Guru Guide

Nobody goes deeper into technology

HACKER'S MANUAL 2016

FULLY REVISED & UPDATED EDITION

EXPAND YOUR LINUX KNOWLEDGE

• THE KERNEL • NETWORKS
• SERVERS • HARDWARE • SECURITY

178 PAGES OF TUTORIALS

MASTER NEW SKILLS YOU CAN APPLY TO ANY PROJECT

Future

TGG08

Get the UK's best-selling Linux magazine

DELIVERED DIRECT TO YOUR DOOR

Order online at **www.myfavouritemagazines.co.uk**

or find us in your nearest supermarket, newsagent or bookstore!

Guru Guide

Nobody goes deeper into technology

HACKER'S MANUAL 2016

HACKER'S MANUAL 2016

EDITORIAL TEAM

MANAGING ART EDITOR
Fraser McDermott

ADDITIONAL ART
Efrain Hernandez-Mendoza

EDITOR
Alex Cox

EDITOR-IN-CHIEF
Graham Barlow

CONTRIBUTORS
Jonni Bidwell, Matt Beilby, Neil Bothwick, Kent Elchuk, Matthew Hanson, Neil Mohr, Les Pounder, Mayank Sharma, Richard Smedley, Mihalis Tsoukalos

MANAGEMENT

CONTENT & MARKETING DIRECTOR
Nial Ferguson

HEAD OF CONTENT & MARKETING, TECH
Nick Merritt

GROUP EDITOR-IN-CHIEF
Paul Newman

GROUP ART DIRECTOR
Steve Gotobed

MARKETING

MARKETING MANAGER
Richard Stephens

PRINT & PRODUCTION

PRODUCTION MANAGER
Mark Constance

PRODUCTION CONTROLLER
Marie Quilter

CIRCULATION

TRADE MARKETING MANAGER
Juliette Winyard
Phone +44(0)7551 150984

LICENSING

LICENSING & SYNDICATION DIRECTOR
Regina Erak
regina.erak@futurenet.com
Phone +44(0)1225 442244
Fax +44 (0)1225 732275

SUBSCRIPTIONS

UK reader order line & enquiries: 0844 848 2852
Overseas reader order line & enquiries: +44 (0)1604 251045
Online enquiries: www.myfavouritemagazines.co.uk

PRINTED IN THE UK BY

William Gibbons on behalf of Future.
Distributed in the UK by Seymour Distribution Ltd,
2 East Poultry Avenue, London EC1A 9PT. Phone: 020 7429 4000

Future Publishing Limited
Quay House, The Ambury, Bath, BA1 1UA, UK www.futureplc.com
www.myfavouritemagazines.co.uk
Phone +44 (0)1225 442244 Fax +44 (0)1225 732275

HACKER'S MANUAL 2016

Welcome!

...to the super-enhanced Hacker's Manual for 2016. Dive in and learn how to hack everything.

Hacking shouldn't have the bad name it has. We do not all wear the black hat of the evil hacker; many of us can embrace the term wearing the white hat of the data defender, or even run with the original definition of the term, coined in the '60s by MIT's Tech Model Railroad Club and Artificial Intelligence Lab: someone using their wiles – generally in a playful way – to achieve a goal. That's certainly what we've gone for here, in this latest edition of The Hacker's Manual.

It's a collection of the most essential features and tutorials from the excellent pages of Linux Format magazine, taking you through everything: choosing the right distro for the right purpose; picking up brand new software and coding skills to solve problems faster and more efficiently; making your network and computers more secure than ever before; and even having a little fun while you're at it.

If you enjoy what you read here, may I highly recommend picking up a subscription to Linux Format magazine? I may. And I shall: you'll get all the latest news, reviews, features and exciting hacker ideas delivered to your doorstep 13 times a year, imparted by the most brilliant team in tech journalism. Head over to page 176 to find out more.

Enjoy your hacking!

Alex Cox, Editor

The Guru Guide Doctrine

Guru Guides are designed to help experienced technology users dive deeper into a subject. Whether you're learning a new programming language or planning to start a new business, each book aspires to be...

- A reference you can keep on your desk or next to your computer and consult time and time again when you need to know how to do something or solve a problem
- A teacher – helping you develop your skills and take with you through your life, applying them at home or even in the workplace
- A challenge – we know that you know the basics so instead of patronising you we'll suggest new things to try and help you take your knowledge to the next level
- Available anywhere – you can take your Guru Guide everywhere thanks to the free digital edition you can download and read on your tablet, smartphone or laptop – see page 178 for more details

How are we doing? Email **techbookseditor@futurenet.com** and let us know if we've lived up to our promises!

Contents

HACKER'S MANUAL 2016

Dive into the world of hacking with this in-depth manual that covers the big topics, from the Linux kernel and wider open-source OS to hacking servers, the web and beyond.

Distros

The distro is the core of Linux, so make sure you get the right one.

Software

Did we say the distro was the core? Forget that: software is what you need.

Security

Hammer up the boards and keep the riff-raff out with these essential secrets.

Do more

Super-maximise your skills and create things you can be super-proud of.

Coding

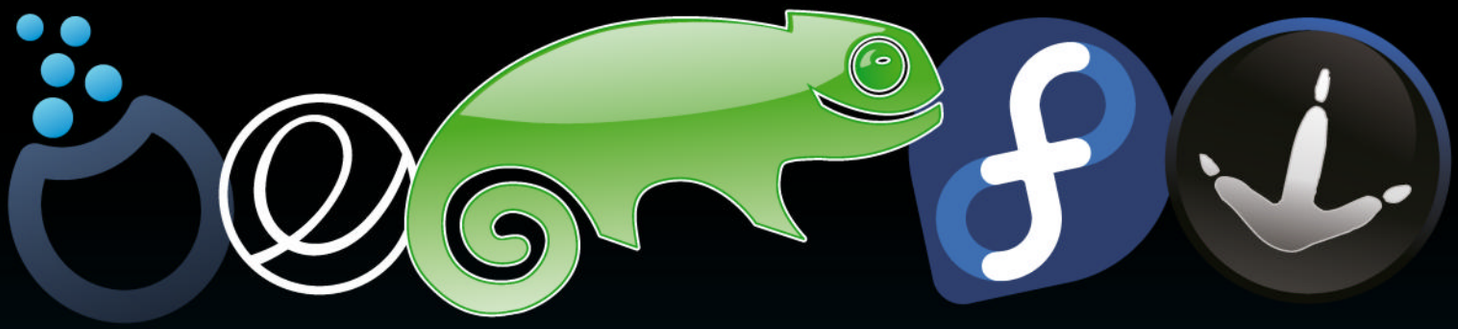
You're not a hacker unless you know your variables from your pointers.

HACKER'S MANUAL 2016

HACKER'S MANUAL 2016

Distros

Because if there was only one form of Linux, we'd be bored



BEST DISTRO OF 2015

Which distro is the one for you? Which is, without doubt, the absolute best? We pick a peck of perfect distros as we head into 2016.

The Linux-verse is teeming with distros of all shapes and sizes, and each of them is a labour of love, but not all deserve a slice of your hard disk. On the face of it, all distros borrow from the same common pool of applications and libraries and you might think they would offer pretty much the same user experience.

However, a Linux distribution (or distro) is more than the sum of its parts. The mainstream distros put in many hours working on open source components to tweak and polish them to suit their particular flavour of Linux.

Things were much simpler in the good ol' days when distro choices were governed by the choice of software or function: OpenSUSE was popular for its rendition of the KDE desktop; Gnome was Fedora's forte; and Ubuntu was the new kid on the block with a novel software centre. Oh, how things have changed. The top distros have a wider mandate and can't afford to just cater to a particular audience anymore.

Another differentiating factor between a regular distro and the popular distros is the amount of time that's spent on building custom tools. The popular distros go that extra mile to create a solid desktop operating system experience and write everything from installers to several critical apps and utilities to manage the desktop. The top distros are also constantly evolving, some more than others. Some distros have the resources of cash-rich multinational corporations fuelling their R&D, such as Ubuntu. But thanks to the nature of open source software that one factor alone doesn't always help corporate-backed projects get a technological edge over pure donation-based, community-supported efforts, such as Linux Mint. Thanks to being in a perennial state of flux, a distro that fails to impress its users with a new feature in one release might win them back as the feature stabilises in future releases.

> "The popular distros go that extra mile to create a solid desktop OS experience."

In the next 10 pages, we'll compare and rate the top desktop distros and help you pick one that showcases the best of Linux and the wider open source community. We've also included the top distros for older computers, distros designed for beginners, rolling release distros for advanced users and server distros for the admins.

THE CONTENDERS

Mageia 5 ■ Ubuntu 15.04 ■ Linux Mint 17.2 ■ Fedora Workstation 22 ■ OpenSUSE 13.2

Installation and update

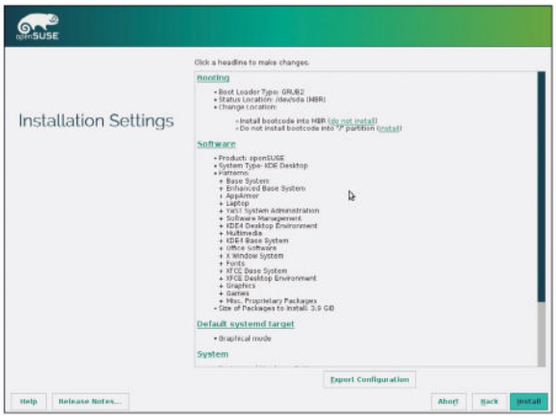
An involved process or an evolved one?

Although some mainstream vendors – such as Dell and Lenovo – have joined the ranks of region-specific vendors, – such as System76 and ZaReason – in offering pre-installed Linux computers, for the majority of users a distro's installation process is still their first encounter with Linux.

A few releases ago, the Fedora project overhauled its *Anaconda* installer, which now employs a 'hub and spoke' model instead of a linear wizard. It isn't the most intuitive installer in the business and it's taken a few releases to stabilise but can even be used with disks with complex layouts now. Advanced users can use the *Anaconda* to create a LVM partition scheme but unlike some other installers it doesn't offer an option to upgrade to a new release. However, the distro's new *FedUp* tool handles the task effortlessly and can use either a network repository (repo) or a DVD image as the package source.

One of the most newbie-friendly installers is the Ubuntu one that's also borrowed by several other distros, including Linux Mint. The installer is easy to use and intuitive enough for new users. The original version in Ubuntu has options to install updates and third-party software, such as codecs. These options aren't available in the Mint version, which automatically installs the codecs and plugins. The installer can also install into a LVM partition and offers the option to encrypt the partition.

Again, this installer isn't designed for upgrading the distro. In Ubuntu this is handled by the *Update Manager* which checks for the availability of new releases and helps you upgrade. By contrast, the recommended method for upgrading Mint is a clean install, but you can also use the *mintupdate* app to upgrade your installation. Also bear in mind that Mint developers don't suggest that you upgrade your installation whenever there's a new release. The current Mint 17.x branch is a LTS release that'll receive security updates and bug fixes until April 2019.



› OpenSUSE's installer lets you save the current configuration into an XML file that can be used for automated installations.

OpenSUSE and Mageia have the two most mature installers of the lot. Both distros have install-only DVDs that weigh over 4GBs and are loaded with software. Both distros offer several desktops, including KDE, Gnome, Xfce and LXDE, while Mageia also includes Cinnamon and Mate.

The OpenSUSE installer allows creation of an LVM partitioning scheme and it can encrypt partitions, and creates users during setup. In addition, it's the only installer that allows you to select a network authentication method, such as LDAP or NIS, as well as a password encryption scheme. The partitioning mode in Mageia's installer can be used in simple or expert mode. The auto-allocate option creates an easy layout with bare minimum partitions in simple mode, while the expert mode offers options with separate partitions based on whether you plan to use the installation as a desktop or a server. Mageia is unique in that it enables you to choose your bootloader and supports *Grub*, *Grub 2* and even *Lilo*. You can also install the distro on machines with UEFI. Both OpenSUSE and Mageia allow you to review all changes the installer is going to make.

Verdict

Fedora Workstation 22
★★★☆☆
Linux Mint 17.2
★★★★☆
Mageia 5
★★★★★
OpenSUSE 13.2
★★★★★
Ubuntu 15.04
★★★★☆

» *The Mint and Ubuntu installers don't have the same flexibility provided by Mageia or OpenSUSE.*

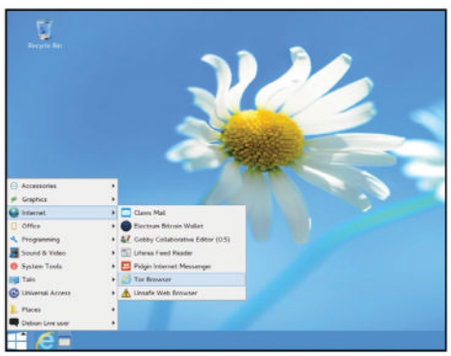
Specialised distros

Besides the desktop distros we compare here there are several other specialised flavours of Linux designed to serve a singular purpose, eg the Debian-based OpenMediaVault distro is perfect for converting an old and unused computer with multiple disks into a NAS server.

Similarly, IPFire is designed to convert a machine into a hardware firewall and router. Then there's TurnKey, a Linux project which produces JeOS appliances for quickly deploying specialised servers, content management platforms and web development platforms.

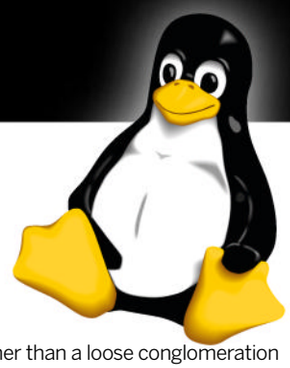
There's also Kali Linux which is loaded with hundreds of tools for penetration testing and security auditing. The Caine distro is similarly designed for computer forensic analysis and includes applications for memory, database and network analysis. If you're concerned about your privacy online look to the Tails Linux distro that ships with a number of internet apps pre-configured for anonymity. The distro uses the Tor network to anonymise all internet activities and includes cryptographic tools to encrypt all files, emails and instant messaging.

Then there are some unique distros that you can install on your disks for repeated use. The upcoming SteamOS from Valve is a Debian-based distro that's designed to run Steam-powered games. But there are tons of non-Steam games which won't run on SteamOS. To play these grab the Play-Linux distro which uses its Ubuntu underpinnings to build a perfect platform optimised for gaming.



› Tails can camouflage itself as a Windows desktop and also includes the *Electrum* bitcoin client.

»



User experience

Navigating the nooks and crannies.

Since all the distros bundle almost the same collection of tools and apps, the one factor that makes or breaks a distro is the user experience. In addition to creating custom artwork, distro developers spend a lot of time tweaking various settings and components to ensure their users get a wonderful experience. All the leading distros spend a considerable amount of effort on making sure their final product is a cohesive unit rather than a loose conglomeration of its parts. They spend time homogenising software to help the applications blend with the rest of the desktop. While the main factor that has a strong bearing on the user experience is the default desktop environment, some distros make tweaks and adjustments to the default settings to deliver a polished product that offers a smooth workflow.

Mageia 5 ★★★★★

Mageia is a wonderfully put together distro that looks elegant with its custom theme and customised KDE desktop. The distro comes from a family of distros that have always been aimed at the desktop user and Mageia 5 continues that tradition. The distro greets users with a welcome app, but unlike many other distros it can do a lot. It informs you about the distro's different repos and lets you install some of the commonly used open source and proprietary apps

Besides the install-only DVDs, Mageia produces installable live mediums for the Gnome desktop which are as robust and reliable as the KDE edition. The distro also has an expansive set of custom tools and utilities that can be used by first-time users and many offer enough flexibility to satisfy advanced users. The project complements its user experience with its vast support infrastructure and detailed documentation.

Fedora Workstation 22 ★★★☆☆

The usability issues with Fedora start with the distro's installer itself which looks prettier than some of the other distros but isn't well laid out. Out of the box, Fedora's Gnome 3 desktop is still very bland and has a deserted look. Unless users enable extensions, they have to grapple with some of its peculiarities, such as a missing bottom panel and the inability to place icons or folders on the desktop. The paginated applications view isn't as effective as the categorised view that is adopted by its peers.

That said, Gnome 3.16 in the latest release features several usability improvements including a new notification system. Fedora's focus has always been on integrating the different desktop environments so that applications from one look like native apps on the other, and the latest release has also made strides on that front.

Pre-installed apps Do you get what you pay for?

Ubuntu, Mint and Fedora produce installable live CDs only, while Mageia and OpenSUSE also have install-only DVDs. All these distros support multiple desktop environments (DE) in different live CDs. However, none of the live CDs allow package selection. OpenSUSE and Mageia lead the others for flexibility as they both offer multiple DEs. Mageia offers the most options, although they both default to KDE.

Once you've picked the DE, both distros enable you to select groups of software for various desktop functions, such as office, multimedia and gaming etc. Furthermore, both distros allow you to install server-specific packages for a web server, database server or a firewall gateway. Finally, you can use both the installers to fine-tune the package selection and even choose individual packages for installation.

Beyond package selection, all the top distros include the usual apps for everyday desktop use. You'll find distro-agnostic apps such as *LibreOffice* and *Firefox*. Some distros require users to equip browsers with plugins to play Flash content or install codecs to handle multimedia files in a proprietary format. Ubuntu enables you to add these during install and while the regular Mint installer adds them automatically, the project has editions for every release without proprietary components. The distros that don't ship with the proprietary bits, notably Mageia, Fedora and OpenSUSE have a well-documented process for adding them.

Verdict

Fedora Workstation 22
★★★☆☆

Linux Mint 17.2
★★★★☆

Mageia 5
★★★★☆

OpenSUSE 13.2
★★★★☆

Ubuntu 15.04
★★★★☆

» *All the distros have a fairly similar selection of default applications.*

Linux Mint 17.2 ★★★★★

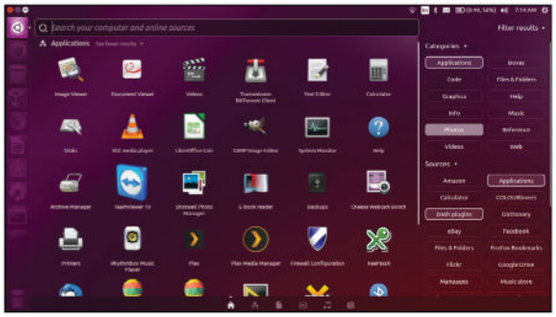
Mint has climbed to the top of the Linux distro charts – at least on **http://distrowatch.com** – by combining the best features of the Ubuntu desktop with a familiar-looking and desktop environment (DE). While it's based on Ubuntu, the distro modifies any tools it borrows to make them more approachable to its user base. One of the best examples of its custom tools is the *Mint Software Manager*, which predates the *Ubuntu Software Center* and is just as slick.

A core strength of Mint is its Cinnamon DE. Cinnamon is based on Gnome 3, but retains the look and feel of Gnome 2. You'll find all the familiar desktop furniture, including a panel at the bottom showing a list of open windows and an Applications menu in the bottom-left corner. Since it's homebrewed, the various components of Cinnamon, such as the file manager are well integrated inside the spiffy desktop.

Ubuntu 15.04 ★★★★☆

Perhaps the biggest contributor to Ubuntu's ease of use is its installer, which can easily carve out space on your disk and set up a dual-boot system without much effort. But while it isn't difficult to get the distro installed, operating its desktop is another matter. While it doesn't look as alien as Gnome 3, Ubuntu's Unity is still visually different to the desktops that most users are familiar with. However, acclimatising to it doesn't take much time and once settled in, you can begin to appreciate the tight integration of the desktop and the apps.

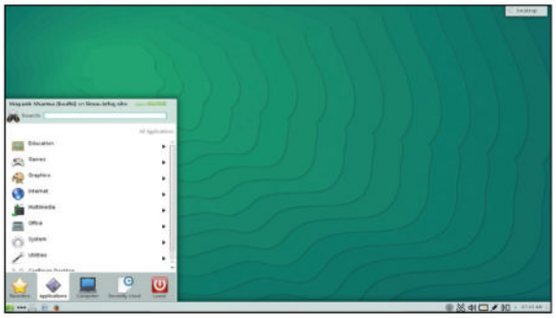
One of the nicest elements of the distro is the Messaging menu that enables you to control your messaging status and presence across various online services. Nifty little tools like this and the *Ubuntu Software Center* give Ubuntu a usability edge over its peers. It's also one of the best documented and most supported Linux distro.



OpenSUSE 13.2 ★★★☆☆

One of the most pleasant-looking distros to the extent that it even customises the splash screens of some of the core apps, such as *LibreOffice*. The distro also tweaks its rendition of KDE with artwork to ensure that all the applications are branded properly with OpenSUSE green, which gives a slick overall look to the desktop.

The distro also gets marks for integrating its tools and settings inside the *Yast* custom control panel for easier access. While it could do with a little polish it really isn't an eyesore, However some of its tools, particularly the package manager, aren't nearly as pretty, eg *Ubuntu Software Center*. That said, it does its job as stated and the distro's one-click install system makes the distro stand out. The distro is also well documented and supported. However, while a looker, it isn't as welcoming to first time users as Ubuntu or Mint.



Package management Flesh out or flush out your distro

While a distro might ship with many applications, sooner or later you'll need to call on the distro's package manager, and virtually every distro has both a command-line package manager and a graphical front-end.

Version 22 of Fedora marked the arrival of *DNF*, which replaces the ageing *Yum*. On the desktop, it relies on *Gnome's Software Tool*. Ubuntu has been leading the pack in graphical package management. The distro's *Software Center* is one of the best tools for fleshing out the distro. Like most package managers, it lists, by default, only packages in official repos. But the distro includes the *Software & Updates* tool, which you can easily enable and add or remove additional repos, and even control how the package manager handles updates.

Mint doesn't borrow much from Ubuntu, its *Software Manager* is visually different, but offers similar options to Ubuntu's manager. The distro also includes the homegrown *MintSources* tool for managing software sources, and the option of *Synaptic* package manager for advanced users.

Package management in both Mageia and OpenSUSE is handled by modules of their respective custom RPM-based control centres. OpenSUSE uses a package manager called *Zypper*, which has a One Click Install system. Mageia's tool, URPMI, isn't as pretty to look at, but is very functional and intuitive enough. Mageia has a tool to enable repos and mirrors as well.

Verdict

Fedora Workstation 22
★★★☆☆
Linux Mint 17.2
★★★★☆
Mageia 5
★★★★★
OpenSUSE 13.2
★★★★☆
Ubuntu 15.04
★★★★☆

» *Mageia has a slight edge for fleshing out the distro without much effort.*

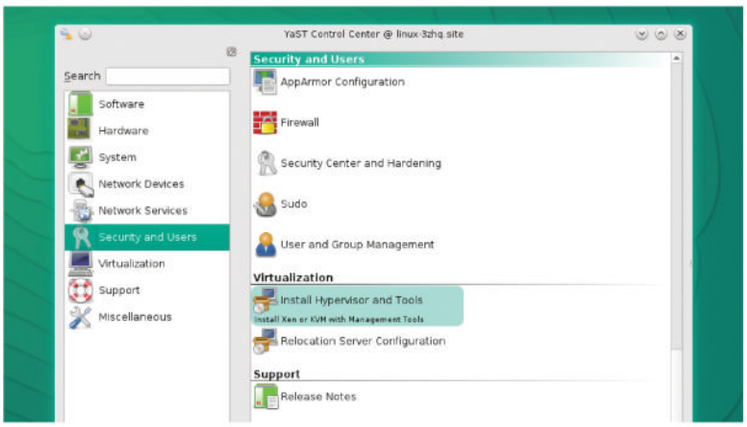
Configuration options

Pop the hood and change the oil.

Linux users have always been able to mould their installation based on their workflow and use. However, the degree of control varies from one distro to another. Some projects, such as Ubuntu, don't offer many tweakable settings. In fact, the distro has received flak for hindering customisation. Every subsequent Ubuntu release has included more customisation, but if you want complete control over your distro you'll need to use a third-party tool, such as the *Unity Tweak Tool*.

Fedora isn't much different. The distro doesn't have a Settings panel of its own and instead relies on the one that ships with Gnome. The Gnome Settings panel isn't very different from Ubuntu's in terms of the bundled configuration options.

While Linux Mint bundles its own custom settings tool for changing the appearance of the desktop and tweak compositing effects, some elements of the settings tool are similar to what Ubuntu offers. The one key difference is the *Device Drivers* tool. In contrast to Ubuntu, the Mint tool has a tweaked user interface and helps users make an informed decision about which drivers to use for their devices.



Mageia and OpenSUSE get additional configuration options, thanks to the *KDE Control Center*.

Both OpenSUSE and Mageia have extensive control panels that you can use to tweak all aspects of the respective installation. OpenSUSE's *Yast* caters to both desktop users and advanced Linux admins. The tool allows tweaking of all the settings for a normal desktop, bootloader and firewall configurations, manage users, set up the network, tune security settings, set up system services and also doubles up as a package manager. It can be used to transform an installation into a *Samba* server, an *Apache* web server etc.

Mageia's *Control Center* offers a similar number of configuration tools. It has modules for managing software, hardware peripherals and system services. Advanced users can employ it to share internet and set up a VPN etc. The distro is working on creating a new Control Center called *ManaTools* which is included as a preview in Mageia 5.

Verdict

Fedora Workstation 22 ★★★☆☆
Linux Mint 17.2 ★★★★☆
Mageia 5 ★★★★★
OpenSUSE 13.2 ★★★★★
Ubuntu 15.04 ★★★☆☆

The most recognisable feature of Mageia and OpenSUSE are their respective configuration control panels.

Default desktops

Balancing form and function.

These days all the top distros offer polished versions of multiple mainstream desktops. Ubuntu is somewhat of an exception in that it only includes and supports its own Unity desktop in the main Ubuntu release and offers other non-Unity desktop as officially supported spins. While the Unity desktop has had numerous usability tweaks and improvements, the desktop still looks different and disorientating to first-time users. If you're willing to adjust, you'll find Unity is well put together and is integrated nicely into Ubuntu.

Using Mageia's Control Center you can configure the graphical server as well as the 3D desktop effects.

Fedora, in many respects is Gnome's flagship desktop and the main Workstation release ships with this desktop. The Gnome 3 desktop is even more disorienting than Unity and you'll most surely have to tweak it before use. Unlike some other Gnome-based distros, Fedora ships with an unmodified Gnome release that's very bland and you'll need to spend some time playing around with its extensions to make the desktop work for you.

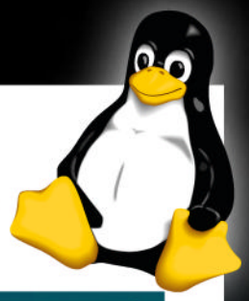
Gnome is also offered as an option on OpenSUSE and Mageia but the default desktop on both these distros is KDE. The KDE desktop builds on the classic desktop metaphor and will not startle first timers. Users familiar with the desktop can explore its revolutionary new features, not all of which are intuitive and easy to comprehend. The good thing is that these stay out of the way and don't trouble users who don't want to use them. KDE also has no shortage of tweakable options. Mint too offers a familiar-looking desktop thanks to the default Cinnamon environment.

Verdict

Fedora Workstation 22 ★★☆☆☆
Linux Mint 17.2 ★★★★☆
Mageia 5 ★★★★☆
OpenSUSE 13.2 ★★★★☆
Ubuntu 15.04 ★★☆☆☆

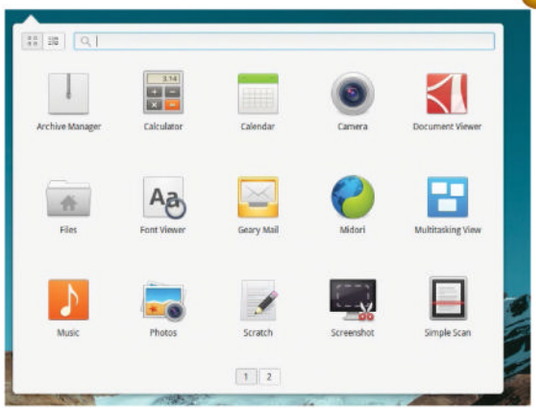
Ubuntu and Fedora lose out to the others for including desktops that take some getting used to.

Beginner-friendly distros



For those who need stabilisers.

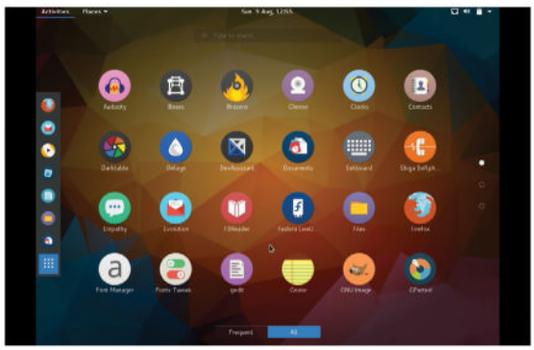
elementary OS "Freya"



This distro has little in common with its base distro, Ubuntu. It ships with its own home-brew Pantheon desktop and has several custom apps, including a Mac OS X-inspired dock. The distro places great emphasis on design and its Apple fixation is evident from the tools it supplies, such as *Snap*, a webcam app, which is similar to Apple's *Photo Booth*. The distro supplies a number of custom tools, such as the *Geary* Mail, *Scratch* text editor and *Audience* video player, which are designed to assist inexperienced users.

The distro even uses its own custom window and compositing manager called *Gala*, which consumes less resources than some of its peers. However, elementary OS doesn't offer many apps out of the box and doesn't include proprietary codecs or ship any non-*GTK* apps which is why it doesn't include the likes of *LibreOffice*.

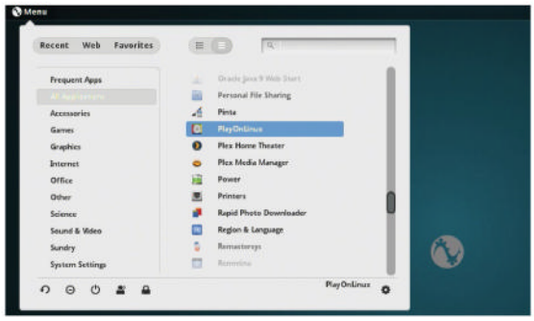
Korora 22



Korora is based on the mainstream Fedora distro and ships separate Gnome and KDE-based live installable editions. In contrast to Fedora's blandness, Korora ships with a heavily customised desktop. The distro has also enabled some Gnome extensions, by default, to iron out some of its navigation issues and includes the *Gnome Tweak Tool* for more customisation. The distro has full multimedia support, and enables third-party repos, such as RPMFusion, *Google Chrome* and *VirtualBox*.

Korora also packs in popular apps and its *Firefox* browser is equipped with useful extensions. The distro has some specialised tools as well, such as the *Audacity* audio editor, *OpenShot* video editor and *Handbrake* video transcoder etc. For package management the distro ships with both Gnome's package manager and *YumExtender*.

Pinguy OS 14.04.2



Another desktop that attracts new users with its intuitive design is PinguyOS. The customised Gnome desktop features a lively dock at the bottom and the Application menu brings up a categorised list of apps, and includes both the *Gnome* and *Ubuntu Tweak Tools*. The distro is chock full of apps and even includes the Plex Media server. Besides the best general purpose and specialised open source apps, it includes several popular proprietary ones, including TeamViewer, Spotify and *Steam for Linux*.

There's also *Wine* that you can manage with the bundled *PlayOnLinux* front-end. If you need more software, it has *Ubuntu Software Center* as well as the *Synaptic* package manager. The distro uses its own repos besides the ones for Ubuntu and Linux Mint Debian.

Verdict Beginner-friendly distros

All three of the desktop distros we've rated, above, have put in a great amount of effort to polish the underlying components of their base distro to a high finish. All three feature incredibly good-looking desktops that are intuitive and functional as well.

Of the three, elementary OS has perhaps put in the most amount of effort into building custom tools and libraries. Everything from the window manager up to its apps is crafted to adhere to its design principles. The one disadvantage with the distro is that it isn't as usable straight out-of-the-box as the others.

Then there's Korora which has turned the clean slate of its Fedora underpinnings into a fully functional smart-looking desktop. The distro is a wonderful starting point for anyone, and its strength lies in its customisation and applications. The distro's weakest point is the *Anaconda* installer inherited from Fedora.

In contrast, Pinguy OS offers the best mix of form and function. Its pleasing desktop environment gives access to its vast number of applications. But make sure you use it only on an adequately specified machine – all its customisations consume a lot of resources and you'll only be able to enjoy Pinguy OS on a machine which has at least 4GB of RAM. On a system with memory lower than that it's best to stick to elementary OS.

»

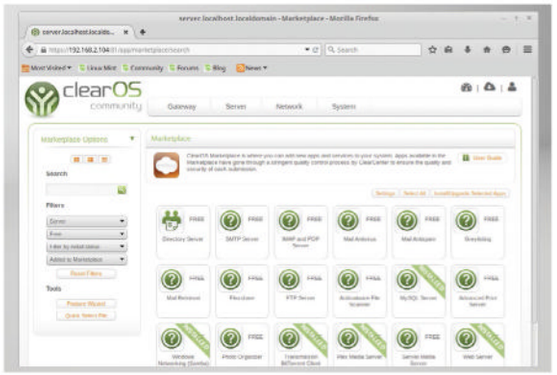
Server distros

For the overseers.

ClearOS 6.6

One of the biggest advantages of the CentOS-based distro is its larger repos of supported server software. The distro offers server options depending on whether you plan to deploy it inside a protected network (like an office), in a publicly accessible network or as a gateway server. The distro supports over 80 free services for various roles including a network server and a cloud server and more. In addition to common servers, you can use it as a seedbox and a Plex Media Server.

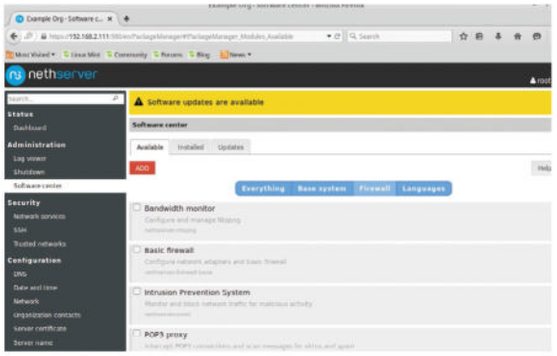
ClearOS also includes several system and network management tools for creating backups, managing bandwidth and RAIDs etc. New admins who aren't sure of the components to install can use the Feature Wizard, which helps pick services.



NethServer 6.6

Also based on CentOS, NethServer enables you to configure the installed server through a web browser. The distro taps into its progenitor's vast repos of software and includes its custom software centre which lists all the supported servers. You can filter through this list depending on the type of server you wish to deploy, such as a firewall, file server, web server and *OwnCloud* server etc.

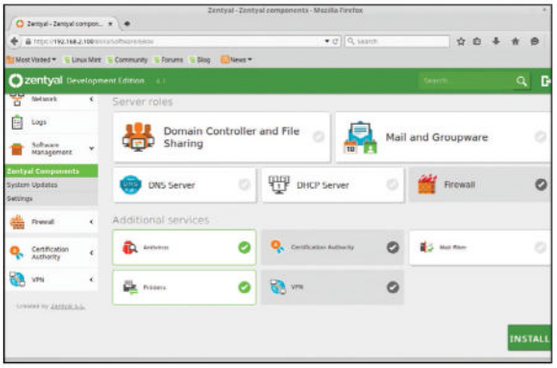
NethServer's browser-based dashboard is well laid out and every section contains a 'Help' button which explains the various options. From the dashboard you can get an overview of the various parameters on the installed server. It also includes a log viewer for tracking the logs of all the installed services. In addition to the in-line documentation, there's detailed guidance on the website, including details for install third-party software.



Zentyal 4.1

Unlike the other two RPM-based distros, Zentyal is based on the Ubuntu Server distro. Zentyal boots to a minimal graphical desktop, but still uses a browser-based interface that's accessible from a remote computer for configuring the installation. In contrast to the other two, Zental isn't an all-purpose server but an office server. Still, you can use a Zentyal installation as a directory server, for filtering email, scanning for viruses, managing printers, deploying VPNs and other core infrastructure services, such as DNS and DHCP, and for issuing and managing secure certificates.

Once installed, you can configure these services from the web interface itself. Zentyal has a polished user interface and its components are nicely integrated. The distro doesn't have an option to install and configure a web server, but you can set up *Apache* from its Ubuntu repositories. If you get stuck, there's a community supported wiki.



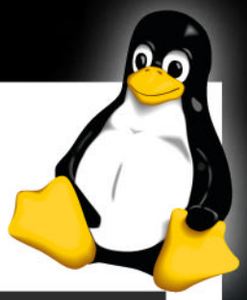
Verdict Server distros

Deploying and configuring a server is an involved process. The three server distros we've covered, above, offer convenience and flexibility, and let you build complex server installations using a point-and-click interface in a fraction of the time it would require you to set them up manually. All three have low barriers to entry and an expansive list of supported servers. In a pinch they are all relatively similar and, ignoring minor usability differences, all offer pretty much the same user experience when it comes to deploying and configuring various servers and their components.

The real contest is the number of servers and services each of them offers. Zentyal brings up the rear since it offers the fewest server options, followed by NethServer and is pipped to the post by our winner, ClearOS. While ClearOS does offer the maximum number of possibilities for fleshing out the base installation, it isn't suitable for all types of deployments, eg if you wish to deploy *OwnCloud*, NethServer is a better bet. Also, none of these servers would impress old-school admins who prefer to build their servers from the ground up. If you are one of these you can go with either Ubuntu Server or CentOS depending on how comfortable you are with their respective package managers. There's also the newly inducted Fedora Server distro, which will enable you to roll out special-purpose servers, but it's yet to make a case for itself in comparison with CentOS.

Rolling releases

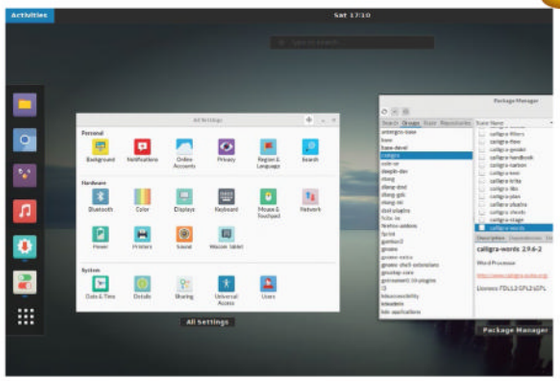
Live on the bleeding edge.



Antergos

A rolling release distro based on Arch Linux, Antergos uses the official Arch repos along with its own custom ones and offers the option to enable the community supported Arch User Repository (AUR) too. Officially, Antergos uses a slightly modified but heavily themed version of the Gnome desktop but the distro's custom installer means you can replace it with a number of desktops: KDE, Cinnamon, Mate, *Openbox* or LXDE.

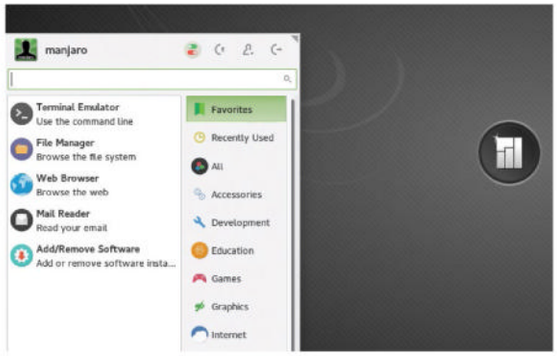
By default, Antergos ships with the *Chromium* browser equipped with Flash plugin. However, during install you can choose *Firefox* as well as some other software that isn't installed by default, such as *LibreOffice*. The distro uses Arch's *pacman* package manager and you can use the graphical *Pamac* front-end to interact with it.



Manjaro 0.8.13.1

Another rolling-release distro, which is based on the ever-popular Arch is Manjaro. This distro uses an installer that's similar to the one used by Antergos. Manjaro recommends using the Xfce desktop but also officially supports the KDE desktop that's available as a separate live installable disc. But community editions are available for other desktop environments, including Gnome, Cinnamon, Mate and *Enlightenment*. Manjaro's default desktop, Xfce is themed and modified.

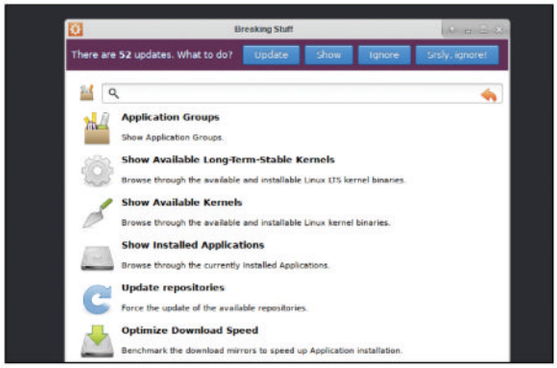
The distro also includes a custom settings manager that doesn't offer very many options but enables you to easily install a different kernel. Manjaro ships with a wide range of apps including *Firefox*, *LibreOffice*, *VLC* and the *Steam for Linux* client. Like Antergos, Manjaro too uses the Arch repository, AUR and uses *Pamac*.



Sabayon 15.07

Gentoo is another highly admired rolling release distro. Based on Gentoo's testing branch, Sabayon retains the rolling-release ethos of its parent, but is a lot more welcoming to first time users. The distro produces different live installable variants based around the Gnome, KDE and Xfce desktop environments.

For installation, Sabayon uses a highly customised version of the *Anaconda* installer that's well laid out and easy to operate. The distro includes proprietary applications, such as *Google Chrome* and some quite bulky open source software too, such as *Gimp*, but no *LibreOffice*. Package management is handled by the excellent *Rigo Application Browser* that's very intuitive and verbose. You can use *Rigo* to install and update individual apps and it'll also help you install Linux kernels. The browser also doubles up as an update manager and can even manage repos.



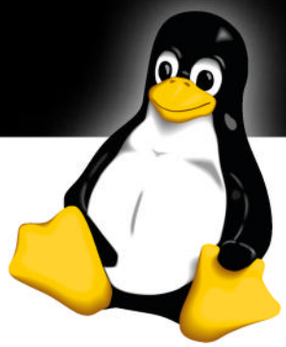
Verdict Rolling release distros

The three distros we've compared on this page, above, all work to reduce the pain of building your system from scratch, which would normally be required by their base distros. However, both Gentoo and Arch are wonderful rolling release distros that give pervasive control to their users for building a system from the grounds up.

Sabayon is perhaps the best Gentoo-based rolling release distro that allows inexperienced users to test the powers of its venerable base. Sabayon's strongest feature is the *Rigo Application Browser*, which is a wonderful graphical front-end to Gentoo's entropy package management system. While fleshing out the distro isn't a tedious job, the distro loses out because of its odd package selection.

If you are looking for a rolling release distro, irrespective of its base, both Antergos and Manjaro are better alternatives built on Arch. There are lots of similarities between the two. Both do a wonderful job of exposing the power and flexibility of Arch to the average desktop user. The distros also have similar installers and use the same graphical package manager. However, Manjaro outdoes Antergos with its installed applications. Antergos also uses the Gnome 3 desktop by default, which is bulkier than Manjaro's default desktop, Xfce. This makes Manjaro even accessible to machines on the lower end of the resource spectrum. Also, the distro has taken pains to ensure that Xfce desktop isn't as dull as the vanilla release.

»



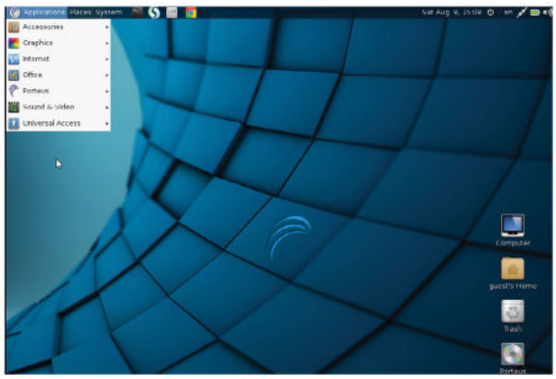
Lightweight distros

For ageing computers.

Porteus 3.1

Porteus is unique in that it doesn't offer a singular download but rather asks its users to build one via its web-based builder. The distro enables you to pick a desktop from KDE 4, Mate, LXDE and Xfce together with a host of popular software, including web browsers (there's *Firefox*, *Chrome*, *Opera*), word processors (*LibreOffice*, *AbiWord*), VoIP client (*Skype*), graphics drivers for Nvidia and AMD Radeon etc. Advanced users can also define and customise boot parameters, such as the tmpfs partition, and enable kernel modules, such as zram.

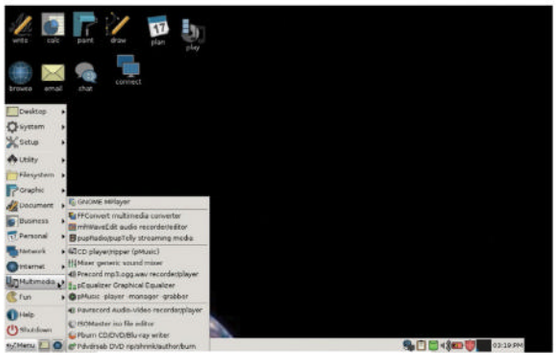
You can use the Porteus installer to install Porteus to a removable USB drive or a fixed hard disk. Porteus is based on Slackware and includes the graphical Unified Slackware Package Manager to help users install apps.



Slacko Puppy 5.7

Puppy Linux is extremely resource friendly too and yet still includes a very functional system. The Puppy Linux project has several official variants. There's Wary Puppy for dated hardware, Lucid Puppy built from Ubuntu's binary packages and Slacko Puppy built from Slackware.

Slacko uses one of the lightest window managers, *JWM*, and there's no beating the distro in terms of out-of-the-box functionality. The distro bundles an application for virtually every imaginable task that you can perform with a desktop computer. It also has all kinds of multimedia applications including graphics viewers and creators and apps to playback, edit and even create multimedia. The included *Firefox* browser is equipped with all kinds of plugins and the distro also has a custom application to download and install the Flash plugin.

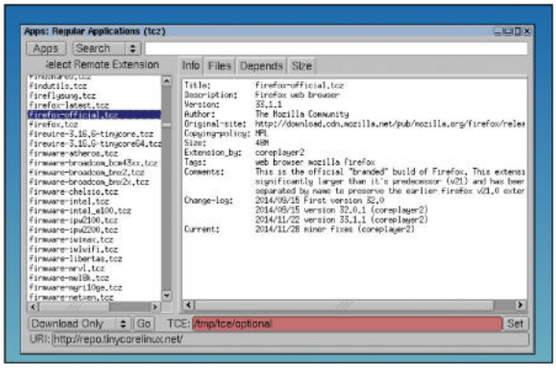


Tiny Core Linux 6.3

Tiny Core is the smallest distro around that boots into a graphical desktop. It isn't a distro that's derived from one of the mainstream distros, but owes its slim stature to a careful choice of lightweight components.

Tiny Core is available in multiple flavours, besides the recommended Tiny Core release that weighs in at a mere 15MB. There is, in fact, an even smaller command line-only 10MB Core release and an all-in-one 72MB CorePlus variant which includes multiple desktops and additional functionality, such as support for wireless network hardware.

As you might expect, the distro is incredibly quick off the blocks and drops you to a plain desktop running the *FLWM* window manager. Tiny Core uses its own package format and its repository is flush with hundreds of popular applications including the *Firefox* browser, *LibreOffice*, *Chromium* and *Thunderbird* etc.



Verdict Lightweight distros

If you need a distro to support older hardware like dial-up modems, look no further than Puppy Linux. But if you're looking for a distro to revive an older machine that's been unable to keep up with the demands of contemporary Linux desktops, then you've got a few options.

Tiny Core Linux is the leanest of the lot. But since the distro doesn't ship with any real applications, you'll have to spend time with its quirky tools converting the basic installation into a usable desktop. Bear in mind that despite the availability of applications and conveniences like automated application installers, it still takes some doing to transform Tiny Core into a regular desktop. In fact, the first application you'll have to download is the distro installer itself, which doesn't ship with the 15MB version. You'll also have to familiarise yourself with Tiny Core's way of doing things.

The lack of familiarity also goes against Slacko. While the distro does include an incredible number of tools for a distro meant for older computers, virtually all of them are the distro's own custom applications with varying degrees of intuitiveness and usability. To its credit, however, the distro includes ample documentation to help you with the transition.

Porteus, on the other hand, manages to find the right balance between familiarity and peculiarity. You get the comforts of using your favourite desktop environment and applications along with the benefits of a lightening fast malleable base.

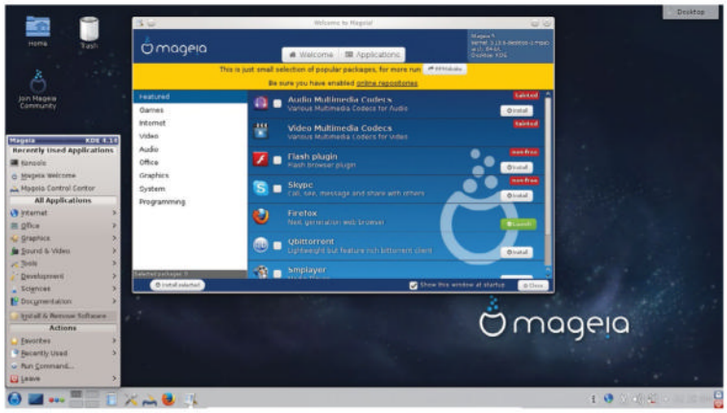
The verdict

Best distro 2015

The one thing free and open source software users don't have a shortage of is choice. The diversity of the sheer number of software on offer makes the task of picking a Linux desktop all the more difficult. For instance, the Ubuntu and Fedora distros are a lot more than single end-user distros; they are complete ecosystems that cater to the broader open source community and power everything from handheld devices to large-scale servers. On the desktop front, both provide a good stable platform for other projects to build on.

If you're not a fan of Ubuntu's Unity desktop environment, you can still benefit from the distro's large software base by using one of its officially supported spins. Similarly, if you find Fedora too bland for desktop use, you can still benefit from its uniqueness by installing the Korora distro.

Despite an active community of contributors, Linux Mint is essentially driven by one individual. The project is primarily supported by donations and can't afford to spare much resources on anything other than engineering the distro in comparison with much bigger projects, such as Ubuntu, Fedora, OpenSUSE and Mageia. Furthermore, the best thing about Mint is its Cinnamon desktop, which is a key catalyst in its meteoric rise. However, Cinnamon is no longer a desktop environment that's exclusive to Mint, and is offered by several other distros, either as an official spin or in their repositories.



› Mageia ships with only open source software but tainting it with popular proprietary applications doesn't take much effort.

This leaves us with two RPM-based, KDE loving distros: OpenSUSE and Mageia. There's nothing inherently wrong with OpenSUSE, but it loses out to Mageia for non-technical reasons rather than technical ones. Mageia is championing the open source movement both on the software and the management front.

Mageia has learnt from the troubled past of its immediate ancestors and is managed in a democratic open source fashion. The distro also offers the widest choice of desktop environments with the aim of making it of use to the most number of users. On top of this, its users can manage their computers using configuration tools that have been worked on and improved for many years. The latest release also makes itself usable on the latest UEFI-enabled hardware. All things considered, Mageia offers the best possible combination of choice, flexibility and ease of use.

1st Mageia 5.0 ★★★★★

» *Builds on the solid foundation of its desktop-centric past to deliver a very malleable distro.*

2nd OpenSUSE 13.2 ★★★★½

» *A very polished distro that can be customised for all kinds of desktop deployments.*

3rd Mint 17.2 ★★★★½

» *Makes good use of its foundation to produce a wonderful desktop that's usable straight out-of-the box.*

4th Fedora 22 ★★★½☆

» *The best supported Linux distribution, and flagship distro for Gnome, which continues to push the envelope.*

5th Ubuntu 15.04 ★★★½☆

» *The ideal distro for users who wish to ride the curve and get a taste of new and upcoming innovations.*

Over to you...

Do you agree or disagree with our result? Share your top distro of 2015 with Linux Format magazine at **lxf.letters@futurenet.com**.

Also consider...

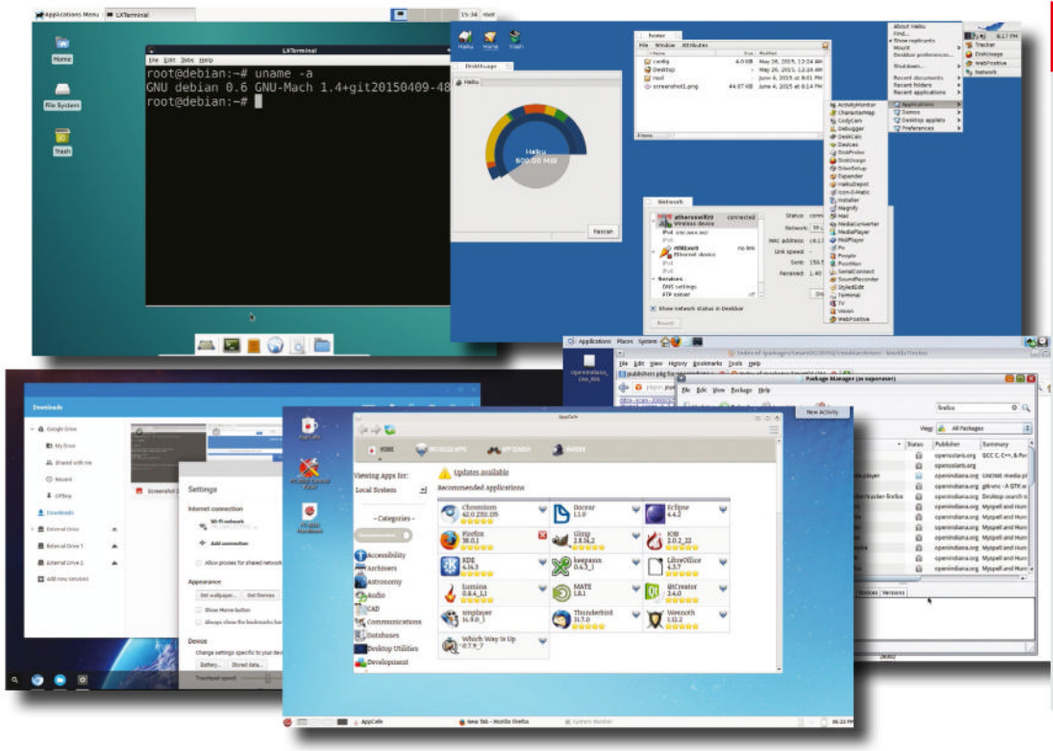
A quick visit to **http://distrowatch.com** will inform you of the immense number of choices on offer. PCLinuxOS and Chakra Linux are two popular semi-rolling releases designed for desktop users, and both use the KDE desktop. However, Chakra is usually one of the first distros to roll out the newest KDE releases.

If you like Ubuntu but not Unity, there's Ubuntu Gnome, Kubuntu and Ubuntu Mate spins. KDE-loving Ubuntu users should also take a look at the Kubuntu-based Netrunner distro. For older hardware you may want to consider Lubuntu and Xubuntu based on LXDE and Xfce desktops respectively.

Mageia's late parent, Mandriva, has also spawned two similar distros with different objectives. There's the OpenMandriva distro that's two-releases old and focuses solely on the KDE desktop, and the ROSA Desktop Fresh with its innovative range of tools for the KDE desktop.

Alternative OSes

We love Linux in all its flavours, but it's not the only game in open source town, so let's follow up our top distro picks by looking at some alternatives.



How we tested...

It's no secret when assessing operating systems that the testing and comparison methods used can affect the results quite significantly. For this roundup we've tried to negate this as much as we can by testing on both virtual machine (in *VirtualBox*) and on real hardware (an HP laptop with a dual-core AMD CPU and Radeon graphics).

Testing a niche OS on a real-world computer may return controversial results, because the user experience will rely on the actual drivers, but we believe that while some people will play with OSes in safe virtual environment others will be curious enough to run them on a spare partition or a separate hard drive. We'll be comparing these OSes in terms of performance, usability, number of available features and applications, online support and development status.

There are plenty of operating systems that are open source but don't use the Linux kernel or, at least, have their own user-land software stack. But why on earth would you want to try them out? Well, it can be useful to study different OS designs; their system tools set and generally how they work, and it has to be admitted that some alternative OSes are very strong in particular tasks. For example, OpenIndiana offers enterprise-grade storage features (thanks to Sun Microsystems of old), PC-BSD has all the advantages of FreeBSD and is very good for web servers (and more), Haiku is a unique project, and not related to Unix-based systems at all, but is very fast, and Chromium OS is the open version of Google's Chrome OS, which powers the increasingly popular, fast and battery-conserving Chromebooks.

> **"We'll highlight OSes that offer the best practical application for the average Linux user."**

So can Google's cloud-based OS compete with classical approach of others? Let's see, and we're also going to discover the purest open source project of GNU/Hurd and put it on the line with our other contenders. Our perspective is going to tend to be more desktop-specific and our goal will be to highlight the OSes that are best offering some practical application for the average Linux user.

Hardware specs

Will they run on your PC natively?

OpenIndiana will likely boot fine from live USB stick or DVD and most of system components will work. There is a community-maintained Hardware Compatibility List (**http://wiki.openindiana.org/oi/Components**), which indicates that there's even an official Nvidia proprietary driver for certain chips in OpenIndiana. Radeon chipsets are supported with basic VGA driver, and most of Wi-Fi chips are reported to work.

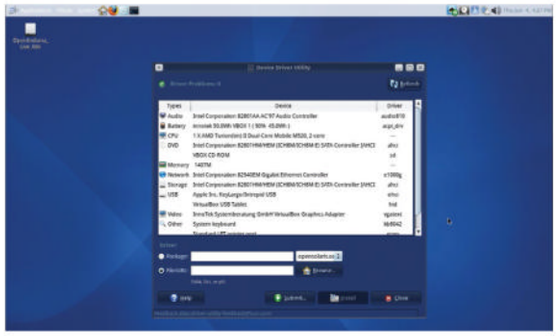
PC-BSD provides the best hardware support and is very close to what we have on Linux. The system offers official Nvidia binaries and Intel drivers for hardware acceleration and a Gallium3D support for most Radeon chips. However, the best OpenGL performance is delivered by *Kwin* in the Plasma desktop environment.

In other aspects PC-BSD matches the hardware compatibility tables of FreeBSD as it uses the same kernel. That means you can install PC-BSD on a real computer and find most of its components working out of the box, including wireless network, printing etc. Of course, not everything is complete: eg Nvidia Optimus will work, but without comfortable switching options between chips, and also some peripherals with Linux-specific blobs can be left unsupported.

Chromium OS has a Linux kernel under the hood and it should deal with most devices acceptably. However, due to forced limitations in the cloud-based OS, it's missing some vital features, such as touchpad support – which is strange considering the OS is targeted at laptops – on some models. In other aspects Chromium OS showed smart chops with perhaps the best support for external peripherals (thanks to the Linux kernel again).

Haiku is a different story. Only two developers work on Haiku's code full-time, so we can't really demand decent hardware support from this tiny OS. Nevertheless, Haiku dealt perfectly with various Wi-Fi adaptors we could find around and surprised us with instant access to WPA2-protected network.

The worst case in comparison with all the others in terms of hardware support is the Debian GNU/Hurd. There's no AGP GART support within Mach, so almost any video chip will be used with the VESA driver; a maximum of 1.7GB of RAM will be used (the rest will be silently ignored); there's no sound support at all; and no USB support (though some keyboards and mice will work thanks to the BIOS emulating legacy interfaces). Finding the right PC configuration on which GNU/Hurd will run would be very tricky.



OpenIndiana has the Device Driver Utility to show you which drivers are currently in use.

Verdict

Chromium OS ★★★★★
Debian GNU/Hurd ★☆☆☆☆
Haiku ★★★☆☆
OpenIndiana ★★★★☆
PC-BSD ★★★★★

You can try booting all the systems, but skip Hurd.

Ease of installation

What does it take to get them up and running?

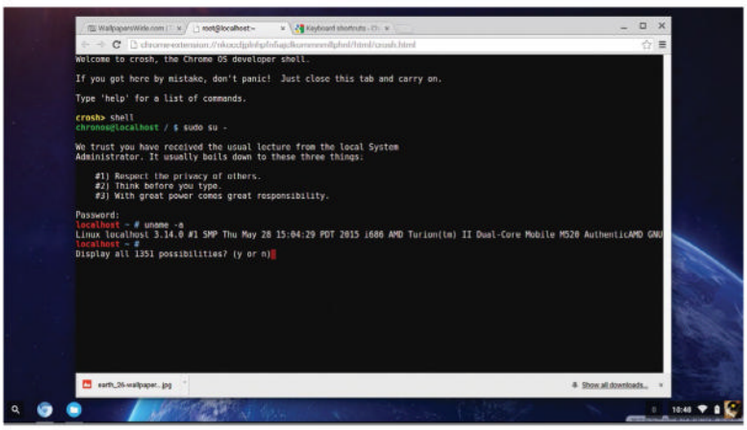
All five contenders in this roundup were all easy to set up in virtual environment. Selecting an ISO as a primary boot device in *VirtualBox* enabled us to run all of them, either in installation or live mode. We also wanted to challenge each OS on real hardware, writing an ISO on a physical media; a USB stick, for instance.

The OpenIndiana website offers a USB image, which, it turns out, is a little tricky to write on USB. Instead, a regular ISO is easier to use, if you know where to download it. (There is a selection on this FTP page here: **http://bit.ly/1lfPr1m**). This may be an obstacle for people new to the OS.

PC-BSD is flashed to USB easily and offers a clean and very good-looking *Qt*-based installer. Logging into the freshly installed BSD system only takes a few minutes with no hassles.

Chromium OS is disappointing in this area as there's only one mode it runs in. Once the OS image is flashed onto a USB drive, it automatically becames a bootable device with a ready-to-run system. The Chromium OS developer guide has an option to install it on the hard drive, but it's not a real installer, but rather simple scripts that flash a driver from a working Chromium OS environment.

Haiku is the simplest OS to run and install. It offers both live mode and a very good (and fast) installer. Debian GNU/Hurd offers several installation modes, as you'd see in Debian Linux, but it has no live mode, and it took about an hour to install the system.



Chromium OS is so simple to use, yet hard for a non-developer to set up.

Verdict

Chromium OS ★★★☆☆
Debian GNU/Hurd ★★★☆☆
Haiku ★★★★★
OpenIndiana ★★★★☆
PC-BSD ★★★★★

It's tie between PC-BSD's and Haiku's hassle-free installations.

Performance

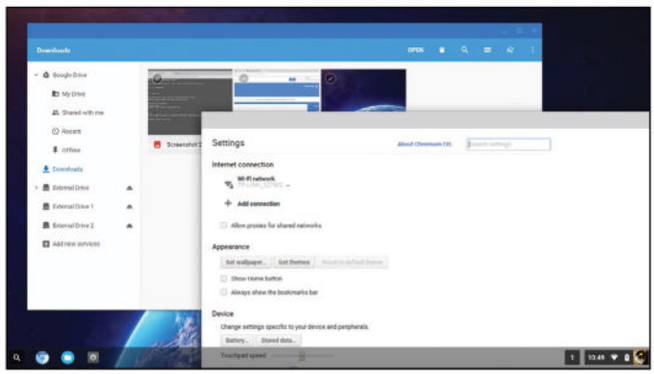
How snappy they are?

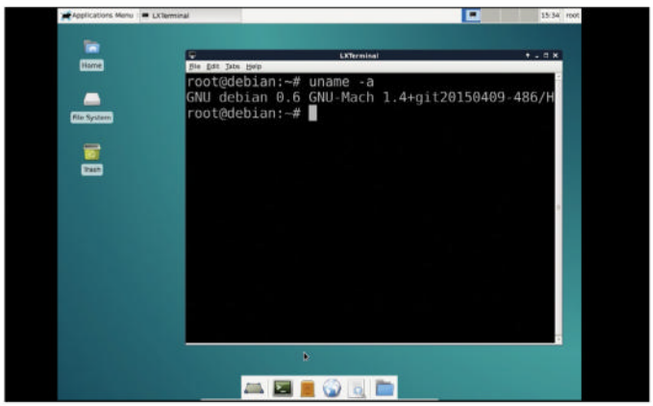
Being used to booting your lightning-fast Linux OS in tens of seconds tends to foster the expectation you can do the same in any other OS. Performance can greatly influence the impression we have of an OS, even if it fails in terms of features.

Performance matters since we want fast installation, fast boot and low latencies between a mouse click on an application's icon and its actual start up. If there's a deviation, we want to find out where it comes from and if it prevents a user from a comfortable computing experience. The difference between th OSes' performance was apparent in the virtual environment, and it was starkly apparent on bare-metal, although your experience will be a little skewed depending on your actual hardware.

Chromium OS ★★★★☆

There can be little or no complaints regarding Chromium OS's performance: it uses the Linux kernel, is based on Gentoo and uses the most recent versions of all system components, wiping off nearly all local applications. After the *X.org* server starts, the rest is handled by the browser, including user login and session, managing windows etc. Chromium OS tries to use pure versions of Gentoo source packages, however a significant number of patches come from Google and the community to optimise Chrome/Chromium OS on certain target hardware, such as Chromebooks. So the OS's performance is very good and if something lags in the Chromium desktop, it's the fault of the browser code, not the underlying base system. Bearing that in mind, you can play with the OS on hardware with 1GB of RAM and a low-end CPU.





Debian GNU/Hurd ★☆☆☆☆

Debian GNU/Hurd has finally brought *X.org* support to Hurd as an out-of-the-box experience. However, from the desktop user perspective the system is incredibly buggy and unstable. To start with, it doesn't bring the graphical desktop under a regular user – we managed to get to the LXDE desktop by issuing `$ startx` under root.

A system running Hurd also feels sluggish and slow to respond, and it isn't easy to determine whether this is due to the unhurried 2D performance with the VESA driver or microkernel I/O issues. We tried to run Phoronix Test Suite for Debian/GNU Hurd, but it turned out that only a few tests would run, such as LAME MP3 encoding, C-Ray, 7-Zip compression etc – and they all indicated a small under run of 4-6% behind the regular Debian GNU/Linux distribution, but still didn't shed any light on why Hurd was so very slow.

Support level and quality

How much help can you get on the web?

Stepping outside the Linux world means that sooner or later you will encounter some problems, so the important question will be: where do you get answers?

OpenIndiana has the large website (**http://wiki.openindiana.org**) with detailed chapters on building, installing and using the system, there is one caveat – the information is targeted mainly for developers and sysadmins, and there isn't anywhere else to look too, other than googling around.

PC-BSD is significantly better in terms of support, as it has a gorgeous community support page (**www.pcbsd.org/en/community**) with an abundance of links to forums, mailing lists, IRC rooms, blogs etc. There are also many non-official PC-BSD resources and Free-BSD websites, that are relevant to both.

Chromium OS has a number of guides at **www.chromium.org/chromium-os**, including Quick Start instructions, but it feels like very little information is being shared with the general public, while the main action still takes place inside Google.

The Haiku project has an official user manual, developer guide and other materials at **www.haiku-os.org/guides** and all information is quite concise.

Finally, the documentation at **www.gnu.org** which makes good on pages at **www.debian.org/ports/hurd** is perhaps the best element in the whole GNU/Hurd project. No wonder, as it's existed since the mid-1990s.

Verdict

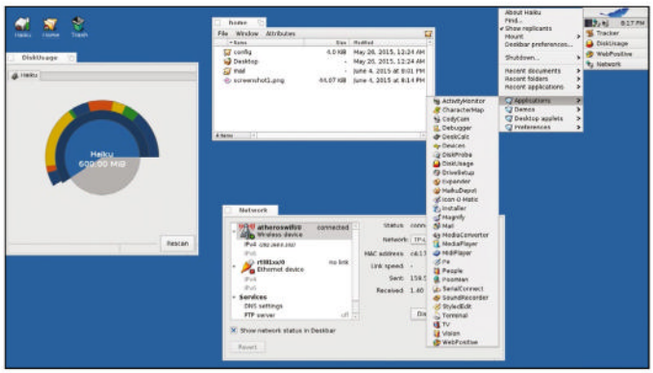
Chromium OS ★★★☆☆
Debian GNU/Hurd ★★★★☆
Haiku ★★★★☆
OpenIndiana ★★★☆☆
PC-BSD ★★★★★

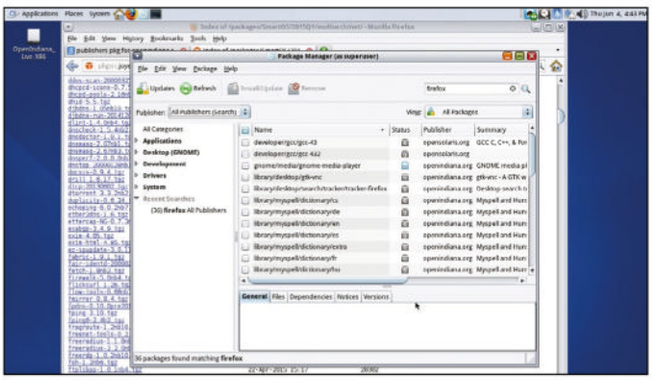
» *BSD systems are strong rivals to Linux in terms of support.*

Haiku ★★★★★

We praised Chromium OS a lot for being very fast and fluid, so you might think it would turn out to be fastest OS in the roundup. Why only four stars then? Well, Haiku runs faster than Chromium OS; faster than any Linux flavour and out and a way faster than other system in our tests.

Haiku surprised us, showing the best figures for each and every task thrown at it. It takes 10 seconds to boot and 1-2 seconds to open any application. Bearing in mind Haiku is 32-bit only with no options, built largely with the ancient GCC2 compiler and without graphic acceleration, which makes our results astonishing. Haiku is a clear winner here with a perfectly optimized graphics stack and tiny footprint in all aspects. Haiku won't shine on CPU-heavy operations and compression, but it's blazingly fast forordinary desktop operations.





OpenIndiana ★★★☆☆

OpenIndiana's performance dips when system configurations use the basic VESA video driver, which lowers the desktop responsiveness. Regardless of video driver, the OS shows noticeable latency when starting and running various applications. The OS relies on ZFS filesystem on its root partition, which adds some marvellous features (such as snapshots) but adds a desktop performance overhead.

OpenIndiana also uses a mixed 32/64-bit mode. The Unix kernel can run in fully 64-bit while most system components are 32-bit – that's why they perform slower. PC-BSD also uses ZFS and while the two are different in most other ways, there is no visible difference on the desktop performance side: file operations and 2D graphics are slower than in Linux, but the lag isn't excessive.

PC-BSD ★★★☆☆

Once installed, PC-BSD boots to the login screen at a pedestrian speed taking a minute or so. The overall desktop performance in KDE4 is rather good, however, in both native and virtualised mode (and PC-BSD automatically enables Guest Additions). It's not as fast as the average Linux distro due to slower ZFS desktop performance compared to ext4 and more basic Gallium3D support, but it's still very usable. Applications such as *Firefox* or *LibreOffice* would start in a few seconds, but *AppCafe* (the PC-BSD software manager) took minutes to initialise, fetch the updates and finally install them – a very unpleasant experience. In many other respects the operating system performed well, it automatically enabled *VirtualBox* Guest Additions and provided accelerated graphics for our Radeon chipset.

Development status

Is the team behind your alternative OS thriving?

An actively maintained OS is crucial for the future of any OS, and each of our OSes has a differing number of developers beavering away on them, and so the time between releases will differ greatly.

OpenIndiana may still be strong thanks to the massive legacy from OpenSolaris community, but the current pace of development is snail-like. The latest release is 151a8, which plopped out of the snail's shell in August 2013; a year after the previous one. The development branch oi_151a9 seems to be alive, but we're not confident about its future.

PC-BSD is much more sprightly, with a new version released every 3-5 months, while Chromium OS boasts hundreds of developers, and its version is synced with the *Chromium* browser releases. However, there are no official ISO images for the Google OS, but rather a set of random builds from various enthusiasts, which resembles a semi-rolling release model.

Haiku OS development is extremely slow, with the latest 'official' release (Alpha 4) dating back to 2012. But the Haiku movement is much more promising with regular events and participation in Google's Summer of Code. Nightly builds of Haiku show off a constant development, even if official releases are far less frequent.

Debian GNU/Hurd had a new release in 2015, based on Debian 8 (Jessie) codebase, which offers hope that Richard Stallman's dream is still alive.

Verdict

Chromium OS ★★★★☆
Debian GNU/Hurd ★★★★☆
Haiku ★★★★☆
OpenIndiana ★★☆☆☆
PC-BSD ★★★★★

» *We're just a little worried about the future of OpenIndiana*

Features and applications

How many useful desktop apps do they offer?

OpenIndiana offers a basic set of desktop applications in its fresh installation (the ISO is less than 900MB) and a few more in two repositories (repos): main and a legacy mirror of old opensolaris.org). There are extra repositories at **http://sfe.opencsw.org** and at **http://smartos.pkg.ec**, but there are very few desktop applications there.

PC-BSD comes with *AppCafe*, a gateway to the system's own repository, and a classical FreeBSD ports support (from command line only). Also, let's not forget a splendid Kldload technology, which enables a BSD system to run Linux binaries, including *Skype*, *Adobe Flash* and some other components, which are available for Linux but not BSD (at least officially).

```
File  Edit  View  Bookmarks  Settings  Help
[atolstoy@pcbsd-5823] ~% su -
Password:
[root@pcbsd-5823] ~# kldstat
Id Refs Address            Size     Name
 1  124 0xffffffff80200000 14eb000  kernel
 2    2 0xffffffff81996000 24e76    drm.ko
 3    1 0xffffffff819bb000 409cc    vboxguest.ko
 4    1 0xffffffff819fc000 105a     vboxvideo.ko
 5    3 0xffffffff819fe000 337a9    crypto.ko
 6    1 0xffffffff81a32000 4ef9     aesni.ko
 7    1 0xffffffff81a37000 1f848    geom_eli.ko
 8    1 0xffffffff81a57000 24dd67   zfs.ko
 9    2 0xffffffff81ca5000 5a75     opensolaris.ko
10    1 0xffffffff81cab000 10725    tmpfs.ko
11    3 0xffffffff81cbc000 a84e8    linux.ko
12    1 0xffffffff81d65000 22949    geom_journal.ko
13    1 0xffffffff81d88000 22bc2    geom_mirror.ko
14    1 0xffffffff81dab000 638a     ums.ko
```

The kldstat command shows what Linux modules are being used by BSD now.

Chromium OS has the only one place to install extra applications from and that's the Chrome Web Store, which is an open marketplace for web apps for both Google Chrome and Chromium OS. But the sad thing is that those web apps can't compete with classic local applications. They aren't entirely awful, but they don't match 'desktop computing applications and it would be totally irrelevant to compare desktop heavyweights, such as *Gimp* with browser extensions on the store.

Haiku has a very modest set of applications, mostly accessible from *HaikuDepot*, its system's package manager. Antiquated sites such as BeBits and Haikuware have been discontinued, though you can find some random apps available for Haiku, such as *Scribus* and a few *Qt4*-based apps. It's not much, but still something.

The GNU/Hurd, Debian team, in contrast, managed to port about 78% of Debian packages to run on the GNU/Mach kernel, but there still aren't any desktops environments other than Xfce and LXDE.

Verdict

Chromium OS ★★☆☆☆
Debian GNU/Hurd ★★☆☆☆
Haiku ★★★☆☆
OpenIndiana ★★★☆☆
PC-BSD ★★★★★

» *Chromium OS lacks the desktop applications of the others.*

First time experience

Are they easy to get used to?

OpenIndiana has a live mode and it welcomes you with a hardened Gnome 2.30 desktop and *Firefox 10* but no *LibreOffice* in its repositories, though *OpenOffice* is there. If you don't mind the antiquated versions of OpenIndiana packages, then you'll feel comfortable. The only real obstacle can be the network card – if a connection isn't found automatically, you'll be faced with a manual setup. (Head to **http://bit.ly/1SYuPk2** for help.)

PC-BSD doesn't have a live mode, and the only challenging part to its install is the partitioner. BSD systems use a different naming convention for disk drives (eg, **/dev/sda**1 will be **/dev/ada0s1a**), but aside from that PC-BSD can be safely installed alongside a Linux distro on different partitions on the same drive. The installer uses KDE4 as default but others are in *AppCafe*.

Chromium OS is tricky to get started with but is easier to work with later on. If you're not a Chromium developer, you can get a pre-built image at **http://bit.ly/ArnoldtheBat** and flash your USB drive with it using `dd`. After you boot, you'll need to access a command prompt with Ctrl+Alt+t to bring up the shell and entering `shell` to access the classic CLI. After that you issue `$ sudo /usr/sbin/chromeos-install` followed by the root password (that's **password** for the images from the link above) and select target device (Note: it will wipe the whole device clean).

As long as Haiku detects your network card you'll be surfing the internet from the *WebPositive* browser in seconds. Haiku is very easy to use and set up, and its interface logic is quite similar to OS X, which is largely thanks to Jean-Louis Gassée, a former Apple executive.

Haiku default browser, WebPositive, is a capable web application based on WebKit.

Debian GNU/Hurd requires extra post-install steps, such as `$ dpkg-reconfigure x11-common` to let users start an X session and some other tricks (see **http://bit.ly/HurdConfig**). But generally the system delivers a horrible experience, with no live mode.

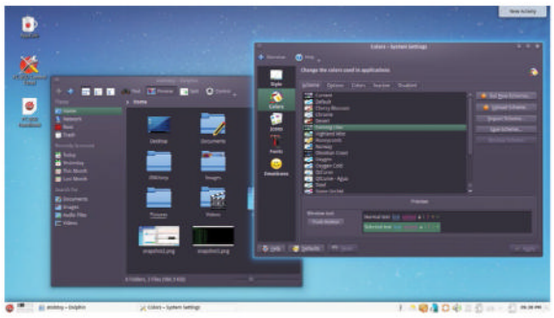
Verdict

Chromium OS ★★★☆☆
Debian GNU/Hurd ★☆☆☆☆
Haiku ★★★★★
OpenIndiana ★★★★☆
PC-BSD ★★★★☆

» *Haiku can outperform any lightweight Linux distro and it also looks cool!*

Alternative operating systems

The verdict



You probably won't notice that you're not in a Linux distribution until you get into a terminal.

The abundance of open source operating systems proves that a community of open-minded developers can do great things, which are worth at least trying out on your home PC. We don't insist that you eventually switch from Linux to another OS, as we love Linux but almost all of them are more or less capable for desktop computing.

PC-BSD is the winner overall with very good performance in almost all the tests we threw at it. The OS is fast, reliable and able to recognise nearly all hardware components and peripherals. It may be missing the live mode, which could garner it even more attention from open source enthusiasts, but the desktop experience with PC-BSD is nearly the same as we'd expect in a decent Linux distribution.

Haiku is a smart OS and really unlike the other OSes. There are builds made with an ancient GCC 2 compiler, which can still run the original BeOS applications together with relatively modern *Qt4* apps. Haiku development is not fast, however, but small changes have accumulated into features that are commendable, such as working WPA2-protected Wi-Fi connections, better USB support and more.

OpenIndiana is ageing, there's no doubt about that. In the past there were great hopes for OpenSolaris and later on for the OpenIndiana/Illumos project, which was supposed to breathe a new life into the 'true' Unix System V and bring it to desktops. However, there's little work being done now and no fresh releases in recent years. OpenIndiana is still worth trying out though, as it has decent support for modern hardware.

Chromium OS is an effort to replace classic desktop computing with so-called 'cloud' computing. If your PC activity fits into the *Chromium* browser, then it may be the system for you. It's fast, sleek and intentionally hides away your system's settings. We appreciate this approach but the truth is that cloud computing using thin clients can't beat classic local apps in terms of features and flexibility.

The fifth place belongs to GNU/Hurd, an infamous attempt to create the most pure basement of a GNU system. Debian developers made a great job of delivering a working distribution with a GNU/Mach microkernel, but it is still far from being stable and usable.

> **"PC-BSD is fast, reliable and able to recognise nearly all hardware components and peripherals."**

1st PC-BSD ★★★★★

Web: www.pcbsd.org **Licence:** BSD licence **Version:** 10.1.2

» Our first choice after Linux, when choosing an open source OS.

2nd Haiku ★★★★★

Web: www.haiku-os.org **Licence:** MIT licence **Version:** Nightly

» A surprisingly usable, ultra-fast and stable OS for computers of all ages.

3rd OpenIndiana ★★★★☆

Web: http://openindiana.org **Licence:** Mostly CDDL **Version:** 151a8

» An old ox, which makes a straight furrow, with some rough edges.

4th Chromium OS ★★★☆☆

Web: www.chromium.org/chromium-os **Licence:** BSD **Version:** 41

» A browser instead of a full-featured OS? Not this time…

5th Debian GNU/Linux ★★☆☆☆

Web: www.debian.org/ports/hurd **Licence:** GPL **Version:** Hurd 0.6

» So many years, but still it's in the early stage of development.

Over to you...

What is your favourite non-Linux open-source OS? We'd love to hear from you. Write to Linux Format at **lxf.letters@futurenet.com**.

Also consider...

We could include a vast array of different operating systems but we will restrict ourselves to mention just four. If you love what Google does, and that isn't everyone we know, why not try Android x86 (**www.android-x86.org**) an unofficial Android port, which runs perfectly on desktops PCs. It doesn't have the drawbacks of Chromium OS while still being a Linux kernel based operating system. Android x86 might not ever become an OS of choice on a desktop, but it runs all those thousands of Android apps perfectly, which could be a painless cure if you don't have an Android-based smartphone.

Another option is ReactOS, which is an open source Windows clone. It has a lot of benefits, even if we're afraid it could lead an average Linux user back in the wrong direction. There are plenty of other open source systems, from the tiny KolibriOS to the massive Darwin forks, so feel free to explore them all.

LINUX VS WINDOWS

Take cover: Microsoft's fired its latest salvo and it's time to square it up against the Linux batallion.

The latest iteration of Windows is here, impressing, confounding and upsetting early adopters. As has become traditional, we pit the Microsoft OS mano-a-mano with Linux to determine the ultimate operating system. Of course, in reality this is comparing apples and oranges (and scoring them with bananas): One is a free codebase which can run on most any hardware imaginable, the other is a proprietary product with an undecouple-able GUI that, until recently, has run only on x86 PCs. Our approach will be to consider features from the Windows 10 build available at press time, together with Microsoft's own PR announcements and compare them with like-for-like equivalents from various Linux distributions.

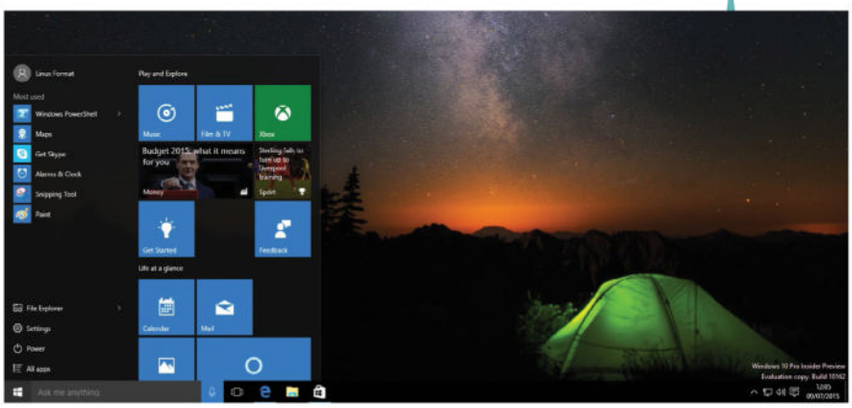
Much of the pre-release hype spoke to Windows 10 heralding a paradigm shift across the Windows landscape. Certainly there are a lot of changes and perhaps most notable is that Windows 10 will be the operating system's last incarnation. That doesn't mean the end of Windows, but rather the beginning of "Windows as a Service". Updates will be pushed to consumers once Microsoft deems them ready, while businesses will be offered a choice of two release channels, dubbed Current and Long Term which offer more rigid release cycles. Individuals who purchase (or are entitled to a free) copy of Windows will see it supported "for the lifetime of that device." Another intriguing development is that users of the pre-release Technical Preview who enroll in the Windows Insider Program can continue to use that and will have the privilege [pain, surely? – ED] of testing new features – and won't have to pay.

> **"We pit the Microsoft OS mano-a-mano with Linux to determine the ultimate OS"**

Windows gone by

We can forgive Microsoft for abandoning its previous strategy of doing discrete releases as it hasn't on the whole worked out well. Windows Vista was received with little affection, mostly because of its demanding system requirements, but let's not forget good ol' user inertia. This is going to get us in trouble, but Vista did have some good points. Sure, the constant user account control (UAC) interruptions were annoying, but they were part of a well-intentioned move to introduce proper account privileges to Windows. DirectX 10 introduced new and exciting multimedia features and the WDDM driver model promised improved graphics performance. But for the most part, Vista was seen as a failure, ignored by users and businesses alike. At its peak it managed a paltry market share of about 21%.

In sum, having a single release of Windows obviates fragmentation problems for Microsoft and upgrade woes for customers. Assuming, of course, that users upgrade in the first place. Many an upgrade-refusenik cites Windows 8 as a reason for staying put and it will be hard to assuage their trepidations and get them to move on. Cosmetically Windows 10 doesn't look or feel all that different to Windows 8.1. This might just be because we Linux enthusiasts prefer to work with grown-up operating systems, but if Microsoft really wanted to avoid naming its latest progeny Windows 9, then 8.2 would be a much better title. Obviously it's a secret how different the underlying codebase really is, but digging around the settings you'll find the same Device Manager that has been kicking about since XP. You'll even find **win.ini** and **system.ini** files which date back to Windows 3.1.



Camping beneath the milky way does not a revolutionary operating system make.

The Microsoft of today is a different beast to that of yesterday. They still enjoy desktop dominance (albeit split between its last five desktop OSes), but this is no longer enough, and CEO Satya Nadella is only too aware of it. The real battle is taking place on mobile devices, and Microsoft barely has a foot in the door. One of the most touted Windows 10 features is platform convergence: PC, Xbox, Windows Mobile devices, giant Surface Hubs and even the Windows 10 build for Raspberry Pi will all run on a unified Windows core, so that one app will run consistently on any of these platforms. For convertible tablet/laptop devices, there's also the Continuum feature, which ensures apps will undergo a seamless UI transition whenever the device is transformed. When Windows Phone 10 is released, it will enable users to plug their phones into a monitor, mouse and keyboard and use it as they would a regular PC. In July 2014 Nadella stated there was already 90% API overlap between mobile, desktop and Xbox code.

Convergence has also been one of Canonical's buzzwords ever since the introduction of its controversial Unity desktop. Two Ubuntu phones have already been released, but these rely on Unity 8 which incorporates the new *Mir* display server. These technologies have a long way to go before they are stable for desktop use, although brave souls willing to try can do so through the Ubuntu Next channel. In all likelihood Microsoft will achieve convergence before Canonical does, but the real challenge for both parties (both small fish in the mobile ecosystem) will be leveraging this feature to win over consumers.

Market share

Windows 7, released three years after Windows Vista, did a reasonable job of righting some of its predecessors perceived wrongs and, credit where credit is due, was generally a much better OS than Vista.

Adoption was fairly cautious, but by Q3 2011 it had surpassed XP. Unfortunately for Microsoft, many of those XP diehards refused to budge and to this day continue not to move. In a way, Microsoft's most successful OS has become its greatest bugbear. Even today, 14 years since being released and over a year after it reached its prolonged End Of Life (EOL) the blue and green XP dinosaur is still roaring (but probably gulping for breath). No doubt Microsoft enjoy the remunerations that go with expensive post-EOL arrangements, but these resources could be better directed elsewhere.

Which brings us to 2012, Windows 8, and the interface formerly known as Metro. While a boon for touchscreen users, desktop users were lost and confused searching for the familiar, and particularly the Start Menu and the desktop. These were hidden behind unintuitive shortcuts or touch gestures. The OS was accused of being in the midst of an identity crisis, with desktop apps and Metro apps rendered entirely at odds with each other. Windows 8.1 was released about a year later and, heeding users protestations, backpeddled on many of the design decisions. Its reception was much warmer, but keyboard and mouse navigation remains awkward. At the time of writing, there are about as many people still using Windows XP as are using 8.1, with both enjoying around a 13% share of the market.

Currently, businesses still languishing with XP are faced with a trilemma: Do nothing, upgrade to the tried and tested Windows 7 or take a gamble and aim for Windows 8.1. The first is not a viable course of action for so many reasons. The second seems like the safest option, but this is an OS that's already six years old, and one for which Microsoft's "mainstream support" program ended earlier this year. Extended support is promised until 2020, but given the glacial pace of certain organisations' (cough UK government cough) migrations, by the time a Windows 7 rollout is complete it'll be getting on time to do it all again. Windows 8.1 may be mature enough by now, but given the similarities between it and its successor, many will skip this release until they judge Windows 10 to be stable enough.

Familiar features

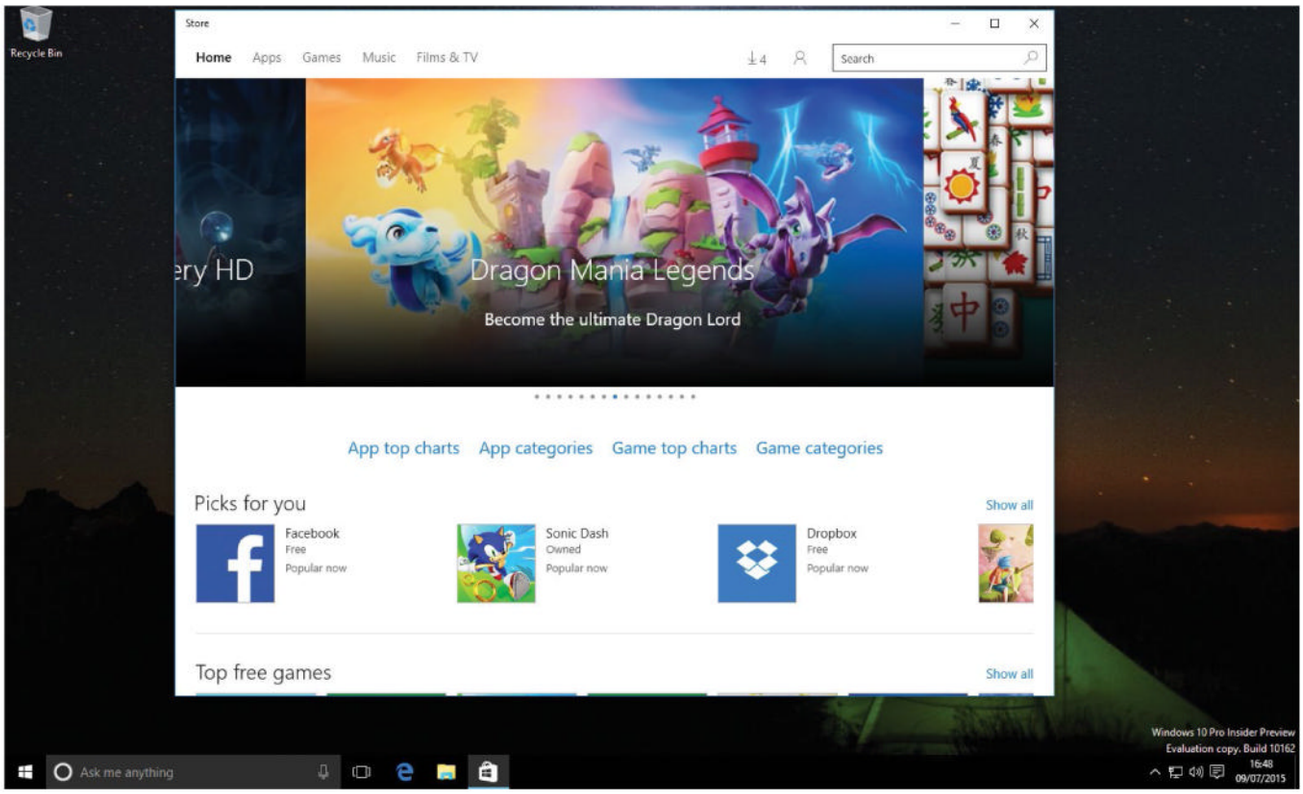
As people do more and more on their desktops – what with multiple browser windows, *Skype* conversations, music players, live streaming setups or whatever is the latest thing the kids nowadays are up to – desktop real estate has become a scarce resource. Thanks to high resolution, widescreen displays the situation isn't as severe as it used to be, but imagine if you had the ability to group lots of different applications or windows together onto a single 'virtual desktop'.

The latest Windows offering lets you do exactly this, with its new Task View feature. Apparently, testing via Windows Insider Program found that users preferred to have only icons from the current desktop visible, so this is the default setting. Previews of all available desktops are available at the click or tap of the Task View button or using the Windows+Tab key combination. At the moment this is a little clumsy though, since invoking the keyboard shortcut places the focus inside the current desktop preview. A couple of extra key presses are required to actually cycle through other desktops and the applications running inside.

Virtual desktops have been available on Windows through third-party programs since the Windows XP days, but more often than not these just used ugly hacks to hide and group various entries on the taskbar. This confuses a number of applications, which are hardwired to believe there can be only one (desktop, not *Highlander*). The discerning reader will, of course, be aware that Virtual desktops have been on Linux since the initial KDE and Gnome releases in the late 1990s, and that they were around, in various guises, long before that in the days of the Amiga 1000 (1985) and the *Solbourne window manager* (1990). It's nice to see Microsoft join the party. Better late than never guys.

Task View in itself is also rather similar to Gnome Shell's Activities Overlay (the screen that shows all running applications). Like Gnome Shell, Windows 10 also features a central notification area (which it has dubbed the Action Center), so that a user's tray is spared domination by dancing icons and toaster popups all vying for their attention. Being able to livesearch applications (and insodoing get unwanted web results) from the Start bar is nice feature, although it's been in Unity and Gnome Shell since their inception. The Unity Dash will even categorise various web results into 'lenses', but obviously it loses points because of the infamous Amazon sponsored results, even if they can be

> **"Virtual desktops have been on Linux since the initial KDE and Gnome releases in the late 1990s"**



Promises of being the ultimate Dragon Lord aside, the windows app store is rather threadbare compared to Ubuntu's.

Future

techradar pro

IT INSIGHTS FOR BUSINESS

THE ULTIMATE DESTINATION FOR BUSINESS TECHNOLOGY ADVICE

- Up-to-the-minute tech business news
- In-depth hardware and software reviews
- Analysis of the key issues affecting your business

www.techradarpro.com

twitter.com/techradarpro facebook.com/techradar


T3
DISCOVER THE FUTURE OF AUTO TECH IN TODAY'S CONNECTED WORLD
ONLINE • PRINT • TABLET
APPLE WATCH
Pre-condition and open your car
BMW i3
The compact electric vehicle to die for
LIFE'S BETTER WITH T3
t3.com

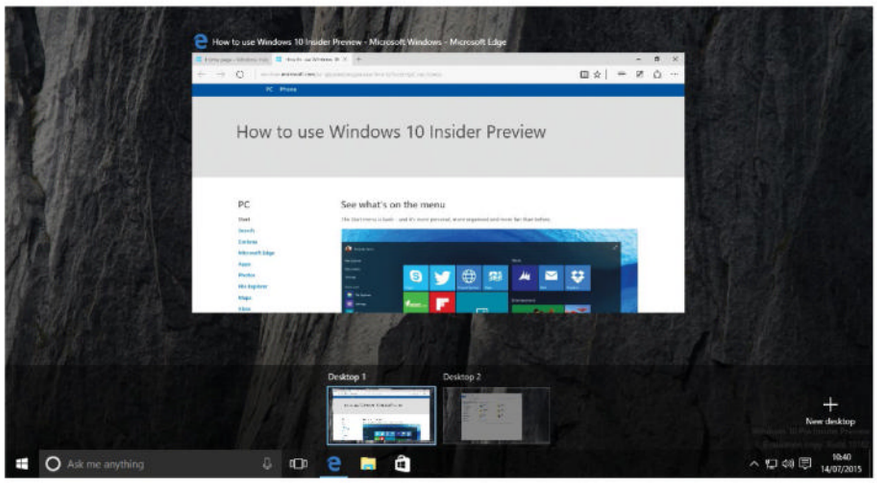
disabled. Being able to see all installed applications is a useful feature. It was vaguely present in Windows 8 (and was in fact the only way to find newly installed applications), but again has been present in a much more useable form in modern Linux desktops for about five years.

Windows Powershell has been around since 2006, and the series sees a fifth instalment with the latest OS. One of its most touted features is that it provides something akin to a package manager. This amazing technology enables you to source software from a trusted repository and install it without having to run the gauntlet of ambiguously worded questions relating to the installation of toolbars, smileys, or other bloatware. Packages can then be cleanly removed with a simple command. The blurb from Redmond calls this Software Discovery, Installation and Inventory (SDII). If only we had something like this on Linux. Oh wait. At present, OneGet (being the title of this new tool) is just a collection of Powershell cmdlets that talks to the repository used by the third-party utility Chocolatey Nuget. This provides just shy of 3,000 packages right now, an order of magnitude smaller than any Linux package manager. In future there will be many other repositories available, perhaps even an official Microsoft one. But at least you'll no longer need to fire up Internet Explorer just to download your favourite browser, it can all be done by opening a Powershell window as administrator and doing:

```
Install-Package -Name Firefox -Provider chocolatey
```

Replace *Mozilla Firefox* with *Google Chrome* if you're that way inclined The `-Provider` argument proved to be necessary for disambiguation with another package called *xFirefox* when we tested, but hopefully things will have been tidied up come the glorious 29 July, when Windows 10 will become available. Naturally, Microsoft will encourage people to use the App Store as their first port of call for new software, but Powershell gurus will enjoy this method. Even if it's not a patch on *APT* or *DNF*.

Gnomes useful way to view your apps is implemented in Windows 10 in a clumsy way.



Microsoft has finally decided that its users are grown up enough to use a new concept it has cooked up called virtual desktops. Little late to the party there, guys.

Windows as a service can in some ways be compared to a rolling-release operating system, such as Arch Linux or Linux Mint Debian Edition. At the same time the multi-branch release model for businesses is vaguely similar to Debian's release model. Indeed, the whole Insider Preview model itself is a big old beta test itself, just like what has been happening with SteamOS over the past year-and-a-bit. But none of these are really Linux ideas, and it's actually quite refreshing to see Microsoft co-opting them. Also pleasant is the fact that this is offered as a free upgrade for those already running a legitimate copy of Windows 7 or later, but this move is largely a deal-sweetener for potential upgraders that are sitting on the fence.

Windows SSH

Another development which isn't strictly part of Windows 10, but which we'll happily include here nonetheless, is that PowerShell is soon to be blessed with SSH functionality. So you will be able to connect to your Windows box and use awkward PowerShell syntax to administer it. While it has always been possible to run a third-party SSH client, such as the venerable *PuTTY*, running a server involved installing the Cygwin environment which is pretty heavy duty. Various bods at Microsoft have pressed for SSH inclusion in the past, but traditionally they have been struck down by management.

Nadella, though, is much more tolerant of what his predecessor might have called 'commie' technology. In fact, Microsoft is going to contribute to the OpenSSH community, and it has just become an OpenBSD (custodians of the OpenSSH project) Gold contributor by flinging a five-figure sum into the pot. Likely this will be greeted with scepticism by some all too willing to quote the often referenced Microsoft strategy: Embrace, Extend, Extinguish. But remember that didn't work with (MS)HTML and it won't work with SSH either. Who knows, maybe we'll even be able to blame them for the next Heartbleed.

System performance

Let's be clear about some thing: our experience of Windows 10 was plagued with bugs and annoyances. But we were testing a preview, and as such it wouldn't be fair to give any credence to them. Bugs notwithstanding, the new operating system, once installed on a suitably specified computer, is impressively quick in general use.

Heeding the frustrations of so many Windows 7 users bemoaning lengthy startup and shutdown times, Microsoft has taken definitive action. So in Windows 8 a new trick was introduced where system processes are summarily dumped to the disk on shutdown, so that they can be speedily reloaded during the next boot. This partial hibernation means that only user processes need to be loaded from scratch, so the time it takes to get to the login screen (assuming the user is not vulgar and passwordless) is slashed. The technique is still in evidence with Windows 10, which managed to boot from an SSD in about six seconds, which is roughly the same time as it takes to get from *Grub* to the *SDDM login manager* on a slimline Arch installation. Day-to-day browsing and poking around the (still largely unpopulated) App Store, was also swift and responsive. The difference is that we've only been using the Windows install for about a week, once a few apps and a few (thousand) obscurely titled runtime libraries are installed the age-old curse of Windows decline will kick in. Our Arch Linux install has been used nearly every day for over a year, has all manner of long-forgotten packages installed, and remains blazing fast. One exception used to be playing Flash videos, which rapidly crippled the system. This was easily solved by uninstalling the Flash plugin because its entirely unnecessary nowadays and serves only as a vector for the delivery of viruses. A modern computer is required to enjoy a smooth-running Windows 10 (see the hardware section), running it on a virtual machine proved particularly painful. By comparison pretty much any computer built in the last 10 years will happily run a lightweight desktop, such as LXQt or Mate, with no fuss whatsoever. Add to that a slightly more modern graphics card (being one that supports at least OpenGL 1.4 and has 128MB of video memory), and it will easily manage a standard Ubuntu installation (the stated minimum requirements are 1GB or RAM and a 1GHz CPU).

> **"One of the strange things that Windows aficionados tend to get excited about is DirectX 12"**

One of the many strange things that Windows aficionados tend to get excited about is the up and coming DirectX 12. Microsoft announced it at GDC in March last year using words including "richer scenes, more objects, and full utilization of modern GPU hardware". Naturally this has implications for gaming, an area where Linux continues to be trumped by Windows. The situation is getting better – there are now over 1,000 Linux games available on Steam. Many triple-A titles have been ported to Linux, and popular FPS adventure game *Dying Light* even saw an unprecedented Linux launch at the same time as its Windows counterpart. Unfortunately, the numbers tell us that Linux gaming is still something of a niche occupation: Around 1% of Steam users (that's a staggering 1.2 million users, extrapolated from the 125 million active accounts) are running it on Linux (even if that doesn't indicate how many are dual-booters). Many Linux users choose to maintain a Windows install solely for gaming where they can enjoy a bigger selection of titles (around 5,000) and more often than not better performance.

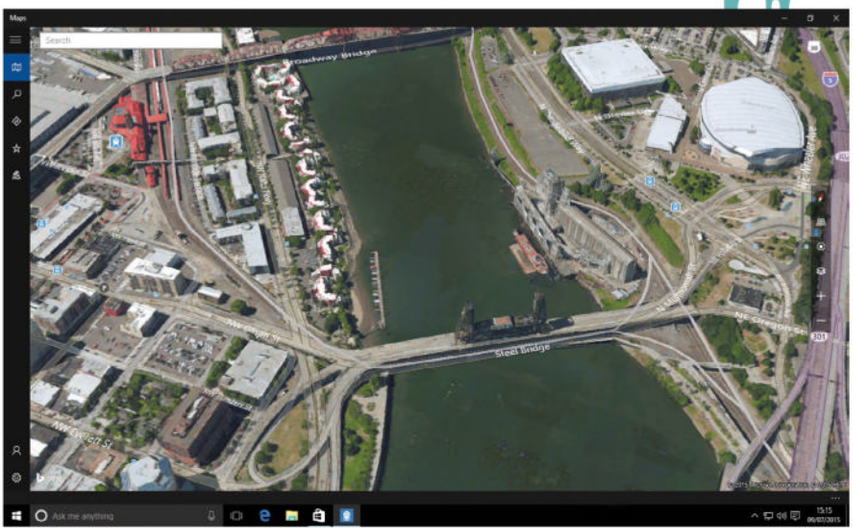
There are a wealth of indie titles available for Linux titles and many of these will run just as swiftly as they do on Windows. High-budget titles though are all-too-often poorly ported. The main issue is the conversion from DirectX to OpenGL, which is often sidestepped by using a wrapper such as *Wine* or E-on. For best results, users still have to resort to the proprietary drivers for most games, and Nvidia (despite its generally poor attitude towards the open source community) tends to trump AMD performance-wise. Mesa, the FOSS implementation of OpenGL, currently only supports up to OpenGL 3.3, which is over five years old. Newer versions of the proprietary drivers support version 4.5, introduced about a year ago. AMD made efforts to break the DirectX stranglehold with its new Mantle technology which AMD promise is coming to Linux, eventually. It saw much fanfare when *Battlefield 4* was launched (boasting a performance boost of up to 45% over Direct3D) but lately, while explicitly stating that it's are not abandoning Mantle, AMD seem to have directed effort elsewhere. Newer OpenGL techniques, dubbed Approaching Zero Driver Overhead (AZDO), offer similar performance boosts, as does DirectX 12.

OpenGL itself is over 20 years old and, like the X protocol, will eventually be phased out. Its stewards, the Khronos Group, has already announced its successor – Vulkan. Valve's Source 2 engine already supports Vulkan and more will follow suit. In the meantime, many major game engines (Cryengine, Unity, Unreal etc) support Linux through OpenGL, so the number of Linux titles is only going to increase.

The *Borderlands* series is one of a growing number of AAA titles available for Linux and is in good company with the recently ported *Bioshock Infinite* and *Shadow of Mordor* incoming.

Desktop & apps

The Windows 10 desktop will not be for everyone – people coming from Windows 7 will have to get their heads around Live Tiles, and some system settings are hard to find. The old Control Panel is still there, but so too is a new one simply called Settings, which you'll find nestled in the Start Menu. Such duality also features in the Start Menu itself, which seems to be composed of two largely autonomous panes: the menu itself and the Live Tiles to the right. Apps can be added, albeit clumsily, from left to right, but going the other way is verboten. In general, re-arranging live tiles was a haphazard affair, sometimes they coherently snapped to the grid, sometimes they wound up at a seemingly random location. Dragging tile groups around proved to be much more reliable.



Windows 10 makes much ado about this Maps application, it's quite neat at showing us Portland, but we could use Bing maps just as well on the web on any platform.

Besides gaming, one hitherto ineluctable point that precluded many from migrating away from Windows was the application ecosystem. Whether its playing the latest games (*see Performance, left*), tinkering with TPS reports in *Microsoft Word*, or pushing pixels in *Adobe Photoshop*, there's always going to be stuff that can't be satisfactorily replicated in a Linux environment. Outside the workplace though, *Microsoft Office* is losing its stranglehold. Most people will find everything they need in *LibreOffice* and many people prefer to work online with Google Docs. *Gimp* is more than sufficient for basic photo editing, but *Photoshop* gurus will still find much to scoff at. As a Linux user, if you do ever find yourself confronted with a DOC file that Google or *LibreOffice* can't comprehend, then you can use Office Online (via a Microsoft account) to convert it to PDF.

The UK government (not exactly known for being digitally progressive) has even selected Open Document Format as a standard. Many major businesses, eager for another excuse to bandy the word 'cloud' around, have successfully transferred to Google Docs, so DOC, that most wretched of file formats, will mercifully not be around forever.

Through Office365 and Creative Cloud Microsoft and Adobe are moving their operations skyward and changing to subscription-based service models. At the moment this still means that the relevant applications still live on your computer, but in future we could see these behemoths transform into web apps and ascend into the cloud. If that happens, and does so in an appropriately standards-compliant manner, then people will finally be able to live the dream and 'run' them on Linux. Open source software is inexorably improving, so by then *Inkscape* and *Krita* could have usurped *Illustrator,* and *Scribus* could have feature-parity with *InDesign*. But don't hold your breath.

Users of Windows 8.1 may lament the demise of its affectionately-titled Charms bar in the new release. However, the shortcuts it housed, particularly the frequently sought for Settings, are now all available from the Start Menu. Windows 10 is surprisingly pleasant to use on a touchscreen device, and while it still has a split-personality feel to it these two egos are sufficiently segregated so it pretty much works like 'old Windows' when used with a keyboard and mouse. Hot corners have been abolished, so there's no danger that letting the pointer stray into some reserved territory in the north-east will trigger a massive occupation of the desktop by a 'Start screen'.

Overall, the Windows 10 desktop is most closely resembled on Linux by Cinnamon, excepting the Live Tiles. Part of the reason for Linux Mint's popularity is this desktop, which is at once modern and traditional, respecting the age-old WIMP (Windows, Icons, Menus, Pointers) paradigm. Plasma 5, the latest incarnation of the KDE Desktop, is another fine choice that retains traditional desktop idioms, and it even works with touchscreens, assuming you can find a touchscreen that works with Linux.

Old systems, new things

For users of older hardware, or just those that don't care for desktop frippery, there are all manner of lightweight desktops available such as Xfce, LXQt and MATE. People seeking a truly beautiful desktop should check out elementaryOS's Pantheon. And then there are Unity and Gnome, the pioneers of brave new desktop territories. There's no denying that these are hard to get used to, but Gnome in particular is gaining something of a following. Once old desktop habits are shed, and a couple of keyboard shortcuts learned, workflows can be made much more speedy.

There's also a new web browser called *Edge*, which is basically Internet Explorer stripped of support for IE6-isms. It looks slick, but the Insider Preview version had some trouble with some websites. Many people will use it just because it's there and has a familiar 'e' icon, but it will be hard to tear people away from *Google Chrome* or *Chromium*, which currently enjoys cross-platform dominance by quite a margin.

The *Edge* browser has done away with the old browser plugin architecture entirely, it doesn't even support Microsoft's own Silverlight, which we can only applaud as that means workarounds will soon be unnecessary.

Hardware and drivers

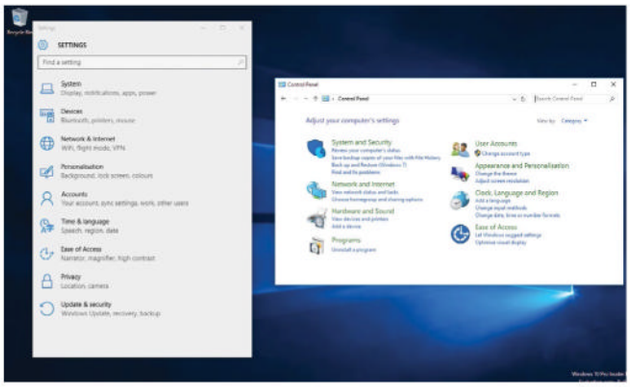
For those who have an older computer dual-booting Windows 7, or even XP, and are considering upgrading to Windows 10, then bear in mind the minimum system requirements: 1GHz CPU, 1GB RAM (2GB for 64-bit), 16GB hard drive and DirectX 9 video card (with WDDM driver). These are pretty modest, especially when we consider the demands that Windows Vista imposed back in the day.

DirectX 9 has been around since 2004, but hardware from that era will likely not meet the driver requirement. Plenty of marginally newer hardware will though, eg the Nvidia Geforce 600 series from late 2004, or AMD's HD2xxx series from 2006 (which back then was made by a company called ATI). These are the minimum requirements though, so don't expect a particularly slick experience using them.

Using only 2GB RAM is no match for a few tabs in *Chrome*, whatever your OS. Also with an old processor, a 1GHz Celeron from back in the day, for instance, you'll be spending a lot of time twiddling your thumbs waiting for Windows to catch up with itself. It's foolhardy to compare raw frequency numbers between old and new CPUs too – multi-GHz processors have been around for 10 years and an old Athlon 64 x2 4800 (2.4GHz) pales into insignificance compared to the similarly clocked Intel Core i3-370M found in many budget laptops.

Credit where credit is due though: It's great to see Microsoft making an effort to support (at least nominally) older hardware, though it is in its interests to unfragment its userbase. There's a pretty lengthy array of graphical features that the OS will automatically tune so that your experience is as slick as possible. Whether manufacturers update their drivers accordingly remains to be seen.

When people begin to consider switching to Linux, they are often concerned about hardware compatibility. The situation here is always improving, but there remain a few unsupported devices: Some older laptop graphics chips are modified by the OEMs, so are no longer recognised by some drivers (although if you encounter such a thing the open source drivers will happily accept your bug report). Likewise, there remain some budget peripherals, such as remote controls and TV cards that lack Linux support. No doubt you'll have seen people on forums complaining about dysfunctional wireless cards, but 90% of the time this is due to missing firmware (which can't be bundled with most distros, but is available in the **linux-firmware** package or failing that can be extracted from Windows drivers using tools such as **b43-fwcutter**).



Windows 10 comes with not one but two control panels to help you tame – and presumably herd – recalcitrant hardware.

> **"New converts to Linux often make the mistake of going and manually hunting for drivers"**

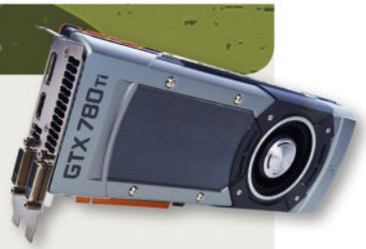
New converts to Linux often make the mistake of going and manually hunting for drivers. This is almost universally a bad idea, your distro will come with drivers for most hardware that's supported on Linux in the form of loadable kernel modules. These are loaded automatically as each bit of hardware is detected, and while they might need minor config tweaks occasionally, it's rare that you'd want to replace them. Some manufacturers do offer Linux driver downloads on their website, but more often than not these have been hacked together by some poor, overworked engineer and will only work with whatever distro said overworked culprit was using. Very often wireless drivers promulgated in this manner are just the relevant parts of the Windows drivers glued to the **ndiswrapper** program.

It's easy to forget that driver problems on Windows arise too. Perhaps more than ever thanks to Windows' driver-signing requirements. A motherboard will require drivers for its chipset, network interface, RAID controller, audio device and various other obscurely named platform drivers. For modern hardware, these will be available from the manufacturer's website. But you'll need to know the precise revision or you'll risk a world of pain. Such downloads often run to hundreds of megabytes, due to various manufacturers' insistence on bundling all manner of bloatware. Linux drivers, in contrast, undergo the scrutiny of the various subsystem maintainers (and possibly even Linus himself) and are guaranteed to be as efficient and well-coded as available hardware knowledge allows.

Graphic drivers

Newer graphics cards will tend to perform better on Windows on release, but eventually the Linux drivers catch up performance wise. They are usually available quite soon after a new graphics card launches too, eg Nvidia already provides drivers for the high-end 980 Ti and Titan X cards. That said, Nvidia's new hardware requires signed firmware blobs to work, and at the time of writing there seems to be some paucity in providing these to the open source Nouveau project. Hopefully this will all be resolved soon.

AMD on the other hand is friendlier towards the open source Radeon driver. Not only does it provide specifications, it pays people to work on it. AMD's latest innovation has been to introduce a common kernel module for both its open source and Catalyst proprietary driver, with the latter's naughty bits annexed to a separate userspace module.

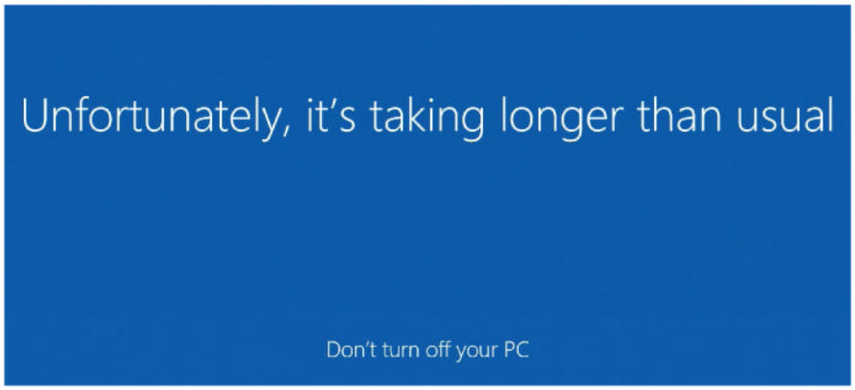


Beyond the desktop

The next edition of Windows Server won't be released until later in 2016, but there are Technical Previews available. The big new feature is in Active Directory Federation Services (ADFS), which allows users from foreign directories and databases to be authenticated by Active Directory domains. ADFS itself has been part of the OS since Windows Server 2003 R2 and enables two realms to establish mutual trust so users from one realm can use their credentials on the other in a fuss-free way.

There are already commercial solutions for authenticating Linux clients against an Active Directory domain controller, and it's possible (though convoluted) to do it using FOSS software. Active Directory uses LDAP and Kerberos which are both open standards. These need to be tied together with Samba and PAM and the domain controller will likely need tweaking as well. In the new edition, this process ought to be much more streamlined.

Centralised authentication in a pure Linux environment can be achieved using the aforementioned protocols, or others such as SASL or NIS. All of these approaches have their advantages and drawbacks, and those coming from a Microsoft background may struggle to recreate the more advanced functionality of Active Directory. It's important to note that Active Directory provides more than just authentication, it handles all the related arcana too – trust, certificates, domains and group policies etc. Many of these are only relevant on Windows systems and the rest can be dealt with using other Linux tools. A common tactic in heterogeneous environments is to have non-Windows machines authenticate to a directory server running something other than AD but which is capable of syncing to and from it; known as deflected integration.



Updates in Windows remain a pain, but we quite like the apologetic tone of this message.

Version 10 of Internet Information Services (IIS) is included in Windows 10, bringing with it support for HTTP/2. Naturally, our top three Linux webservers (*Apache*, *Nginx* and *Lighttpd*) have had support since not long after RFC7540 was published in May. And were supporting SPDY, essentially the parent protocol of HTTP/2, prior to that. Before the 7.0 release, IIS was something of a laughing stock, being little more than a bloated web server that didn't allow more than 10 simultaneous connections. It has grown up now, incorporating a modular extension system and being more scalable on multiprocessor systems. To improve performance IIS uses a kernel-level driver for processing HTTP requests. An IIS vulnerability discovered in April allowed attackers to achieve remote code execution on unpatched systems by exploiting this driver and its status. Linux has had web server bugs too, but its architects know what does and doesn't belong in the kernel.

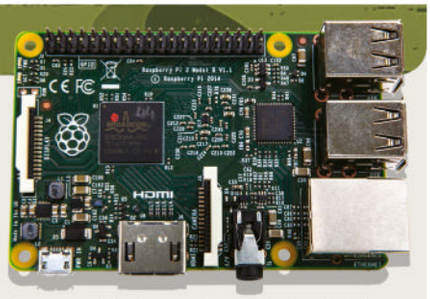
Linux remains the undisputed champion of the server world, which is why it runs most of the internet. We have world-class web servers and databases, industrial grade distros (such as Red Hat Enterprise Linux or the free CentOS) and the advantage of open source on our side. Linux virtual machines tend to be cheaper than their Windows counterparts, and are much more efficient thanks to its modular nature.

> **"Before the 7.0 release, Internet Information Service was something of a laughing stock."**

Windows IoT Pi Edition

Windows Server Core, introduced in 2008 provided a minimal Server OS sans the Explorer shell and many other features not required by most people. Continuing this theme, we now have Windows 10 IoT core, aimed at small Internet of Things devices. At present, builds are available for five devices including the Raspberry Pi 2. This doesn't mean you'll be running *Edge* and have Live Tiles all over your Pi desktop. No indeed, you won't even have a Pi desktop, all code is written in Visual Studio on a Windows 10 machine and uploaded to the Pi. All of the available builds allow programs built on Windows' Universal App Platform to run, which means that they must be programmed in C#, C++ or Javascript and with a XAML, HTML or DirectX presentation layer. You can connect to a Pi running Windows IoT Core using either PowerShell or SSH.

We're pretty far from impartial here, but we think that reducing the Pi to minion status in this way seriously detracts from its appeal. Being able to boot into a proper desktop (even if it is slow and clunky on the original Pi), or run code straight from the Python interpreter, helps new coders appreciate that this diminutive board is very much a fully-functional computer. Of course, if you're a seasoned embedded applications programmer then such a desktop is just going to get in your way. There are all manner of Linux distros designed to be run on embedded devices, including Yocto Sancto and Angstrom. It's also worth mentioning that there are already a huge number of embedded devices already running Linux in one form or another: sat-navs, set top boxes, the TV's on which the latter are set and the list goes on. The latest Tux-flavoured innovation in this area is Snappy Ubuntu Core, which is aimed at the Cloud as well as Things.



Server distros

Do you want to set up your own web, mail or file server, or any combination of these? We compare five distros that cover your needs.

How we tested...

The distros were installed into identical Qemu/KVM virtual machines to make back-to-back comparisons easier. They were also tested on real hardware to make sure they worked in the real world, too. If you are setting up a commercial server you'll either pay for a turnkey system or employ experienced sysadmins. We looked at these servers from the point of view of those wanting to set up a home or small office server, and wanting to spend more time using it than reading man pages. So ease of installation and configuration, along with flexibility were important considerations.

That's not to say that you can't use these in larger environments or that you can't build your own server using a standard distro like Debian, as you can on both counts. You can build from scratch but if you want something that 'just works', read on.

Linux has always been strong in the server space, but which distribution (distro) should you use if you want to set up a server? As with so many of these questions, it depends on what you want.

Just about any distro can be used as a base for a server, although those that install a complete desktop suite are the least suitable as it means removing all that before you add in what you need. That leaves two options: use a distro where the installer lets you choose what you want, such as the Debian net install, or pick a distro that is aimed at the server space and provides a ready to use environment in the way that the home version of Ubuntu and friends provide a ready to use desktop.

If you are looking at commercial use, in a business setting you are going to either pick something with a service contract or you have sysadmins capable of putting together a server suite for you. So we will look at the other server option here, and that's ready to go server distros. Some of these come from familiar names while others are based on well-known distros but are variants that are targeted specifically at server usage. That just leaves the burning question: which one is right for you? Let's find out.

> **"Some of these come from familiar names while others are based on well-known distros."**

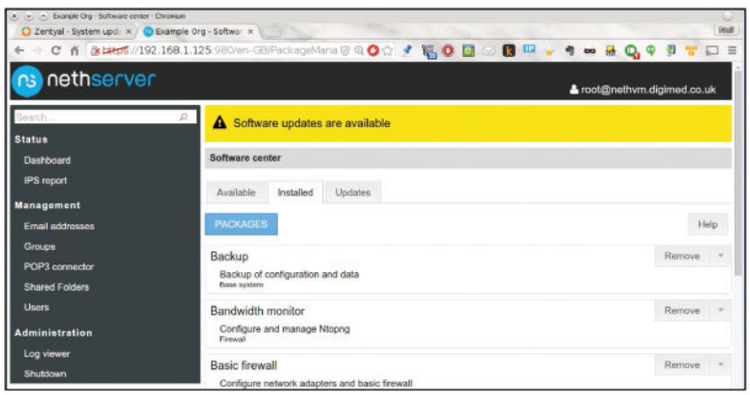
Installation

How easy is it to get onto your computer?

All of these distros are intended to be installed and run from a hard disk, they come as pure install discs, not live CDs, with one exception. The installers are the same as you would see on a desktop distro – usually the text version – so you will need a monitor and keyboard, and maybe a mouse too, attached for the initial installation.

All of them can be run headless after this. In general, there are not many choices to be made during installation, you are unlikely to want a server that dual boots, so it is more or less a case of what goes where.

ClearOS, which is based on CentOS, uses the Red Hat *Anaconda* graphical installer. NethServer has an unattended install option that installs the distro to the first hard disk with default choices, which does use a graphical display even though you cannot interact with it. You'll still need a keyboard attached to press Down+Enter+Enter to select this from the boot menu. TheSSS (it stands for The Smallest Server Suite) does things slightly differently as it boots to a console login from where you can either run servers or the install program, once again this provides minimal options and expects to use the whole disk.



Here is NethServer being installed with no user input whatsoever.

One area that these installers handle surprisingly poorly is hard disk partitioning. Most of them set up a single partition for everything. ClearOS and NethServer are even worse in using LVM but then filling the volume group with a single logical volume, negating the benefits of LVM. Ubuntu Server handled this very well, using LVM but asking how large the root filesystem should be and then allowing you to add further logical volumes and give their mount points. This is particularly important on a server where you usually want to keep the contents separate from the OS, which means having **/var** on its own filesystem.

NethServer's unattended install is a definite bonus, enabling you to get things installed then set it up afterwards. Conversely, ClearOS, Ubuntu and Zentyal let you make more choices during install. Which is best depends on how you prefer to work, but the NethServer approach is better if you are installing more than one server.

Verdict

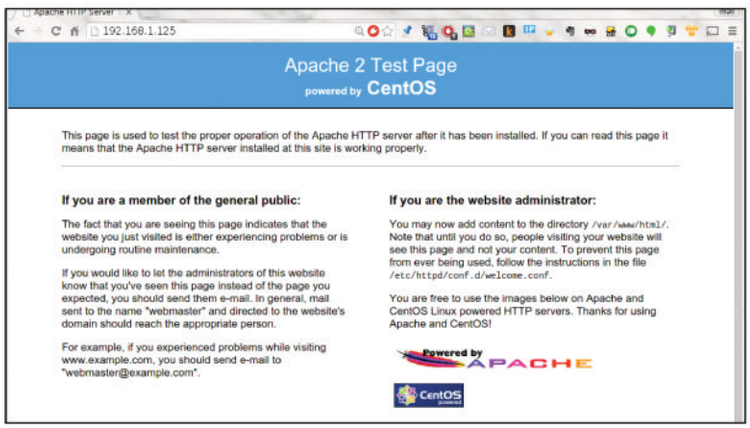
ClearOS ★★★★☆
NethServer ★★★★☆
TheSSS ★★★☆☆
Ubuntu Server ★★★★☆
Zentyal ★★★★☆

» *NethServer wins because of its useful automated install mode.*

Popular services

All servers are not created equal.

The word 'server' is a wide-ranging term, we normally think of a black box serving web pages, emails and files. These are the most popular uses for a server and all of these distros do all of this; with a couple of exceptions. Zentyal doesn't provide web or FTP services as it's intended more as an office server. It's based on Ubuntu, so you can easily install *Apache* or another server if you want to, but you will have to set it up yourself.



Most of the candidates provide *Apache*, but you have to find the content!

TheSSS is an extremely lightweight distro, so it's no surprise that it has the fewest services available. It does provide a web server, and it is *Apache* not one of the lighter alternatives you might expect. FTP is also included but not the more common (these days) file sharing protocols like NFS or *Samba*. The most obvious omission is a mail server, but that wouldn't fit in with the lightweight aspect. Handling thousands of mails for each of a number of users isn't the workload you would give to the sort of hardware TheSSS is aimed at.

The other three, ClearOS, NethServer and Ubuntu Server, all use much the same software for these services: *Apache*, *Postfix* and *Dovecot* (cyrus-imapd on clearOS) so the main differences in this respect are how easy they are to work with, and NethServer has the edge when it comes to administering mail accounts.

Verdict

ClearOS ★★★★★
NethServer ★★★★★
TheSSS ★★★☆☆
Ubuntu Server ★★★★★
Zentyal ★★★★☆

» *Both ClearOS and NethServer have access to many CentOS server packages.*

Web administration

Most servers run without a monitor and need remote administration.

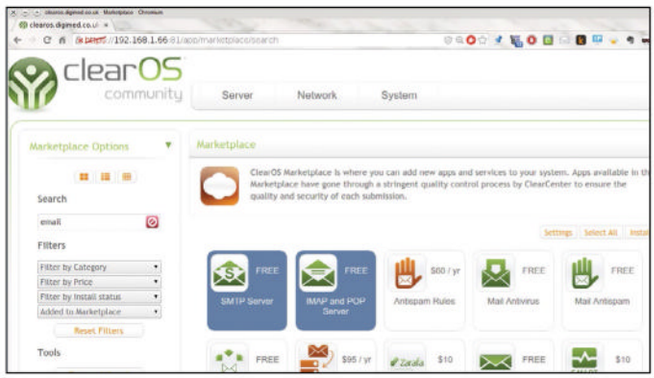
Hard core sysadmins do everything in a terminal using *Emacs*, or even (heaven forbid) *vi*, to edit configuration files. Mere mortals prefer a graphical interface, especially for tasks they aren't familiar with. So it can be important for a server distro to have a good administrative interface and the usual way of doing this, as the servers often run headless, is to run it in a browser session. That way you can administer your server from anywhere on your network. A server distro typically has a lot of components to look after, so it's important for the admin interface to be clear and well organised.

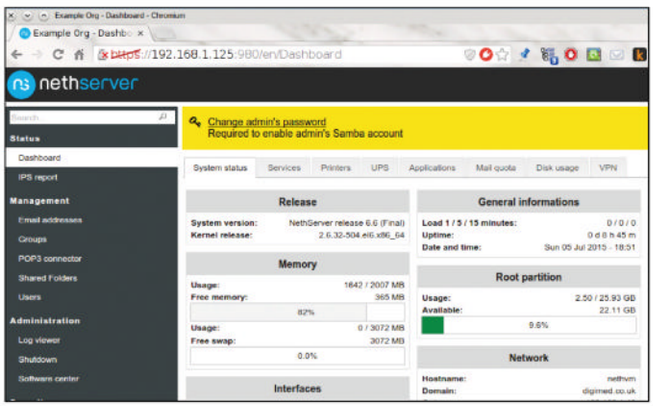
Of course, you shouldn't be locked into using such an interface. If you know what you need to do it's often faster to dive into a shell session, so SSH access is also important.

ClearOS ★★★★★

ClearOS boots to a graphical display showing the details needed to connect to it and a link to change your network settings. When you connect to the ClearOS web interface you are presented with the install wizard, where each page includes a help panel making the process easy for less experienced users. This process does involve registering an account, which may make some people nervous.

The last part of the wizard enables you to choose the applications you want, either by function or name, or you can skip this section and do things yourself later. The ClearOS marketplace provides free and paid applications, it's clear (no pun intended) that this is a free version of a commercial product. The interface is slick and easy to navigate but parts of it, particularly the marketplace, can be slow.





NethServer ★★★★★

NethServer enables you to set the IP address when you install it, so you can load it straight into your web browser after it has booted. The Server Manager interface is clean and well laid out, making it easy to find and change any settings with the minimum of fuss. Each page contains a 'Help' button should you need any explanation of the options. The pages cover user management, service configuration, software installation and updates, network configuration and much more. Setting up the various services you want to run is also done easily from here.

There are options available to backup and restore your system configuration, and daily backups are automatic but you can do them manually more often when you are experimenting with the setup. Scheduled backups of data are also taken care of and can be sent to a network share or USB drive.

Working with Windows

Sometimes a server needs to cater to those less fortunate.

However much we love Linux, and we do quite a lot in fact, there are a vast amount of people that do not use it, so we usually need a server that will work with other operating systems.

For web and mail that's not an issue, there are standard protocols and the server generally doesn't care about the operating system that is being used to talk to it. However, there are some protocols designed for Windows, how well are these supported?

Of most importance is the groupware facilities of Microsoft Exchange, which is considered by many to be an essential requirement for a mail server on a network that includes Windows systems.

ClearOS has *Zarafa* as an additional (paid) option. Zarafa provides MS Exchange-like groupware services. That is, it does what Exchange does, but it's not directly compatible, but it does work with all desktop and mobile platforms. If you want native compatibility with Microsoft Exchange, you should consider *OpenChange* instead, which you will find included as part of Zentyal.

NethServer uses *SOGo*. This is similar to *Zarafa* in that it provides Exchange-like services, but it can also use *OpenChange* for true Exchange compatibility. In contrast, Ubuntu Server isn't restricted by the contents of a web interface, so you can install whatever you want, and all of the above options are available for it.

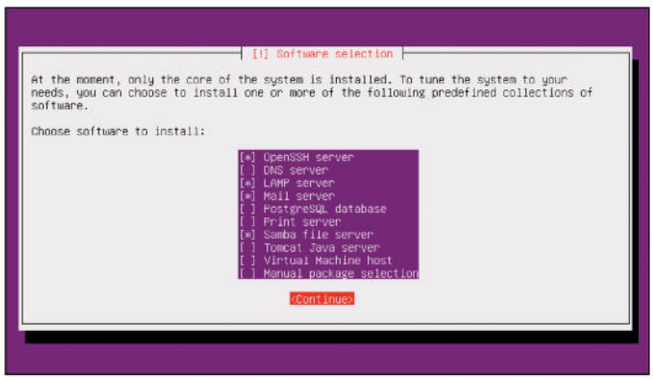
Verdict

ClearOS ★★★★☆
NethServer ★★★★★
TheSSS ★★☆☆☆
Ubuntu Server ★★★★★
Zentyal ★★★★★

» *NethServer has OpenChange, but Zentyal wins for its ease of configuration.*

TheSSS ★★★☆☆

Despite its diminutive size, TheSSS includes a web administration interface. It is pretty basic and mainly an interface to editing various configuration files, but it does help. You still need to use SSH to perform some operations, but you are not left to your own devices as some helper scripts are provided to help you administer the server, run `helpme` at the terminal prompt to see a list of the commands. For a list of server commands, run `server`, which lists various sub-commands for each of the servers. TheSSS does run directly from CD or USB stick so you can try it without installing to your hard drive, but if you are not comfortable using the command line, you will probably find TheSSS is not for you – but what do you expect from a 54MB install image?



Ubuntu Server ★☆☆☆☆

The only remote admin tool installed with Ubuntu Server is OpenSSH. You can install the Zentyal packages, but if you're going to do that you may as well install the Zentyal distro. The only other method of admin documented by Ubuntu is to use Puppet, which is intended for administering multiple systems at once, and not a tool for running a SoHo server. Ubuntu Server is pitched at the pro end of the market and expects an experienced sysadmin to maintain it, which means the UI for the purposes of this comparison is effectively nonexistent.

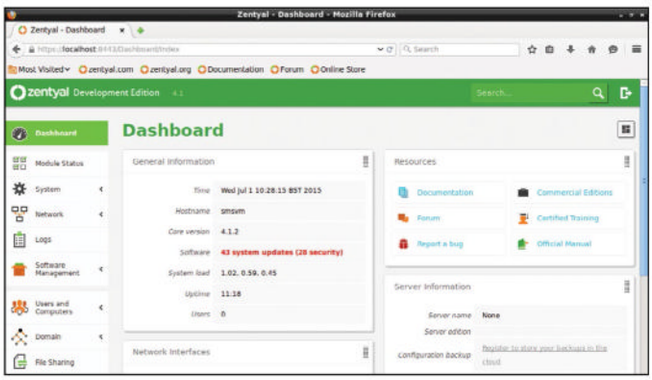
You could also install *Webmin*, a generic web-based system administration program. However, Ubuntu is never going to win in this area and if it is important to you you should consider an alternative.

Zentyal ★★★★☆

Zentyal boots to a full X desktop, running LXDE with *Firefox* open at the configuration login page, even though you've just logged in as that user. You can use this interface from another computer on the network, which is the usual way of doing things, unless you intend to regularly admin the server from its own desktop. A few seconds with a search engine will tell you how to stop the desktop loading, which you shouldn't have to do.

The interface is slightly unintuitive in that pressing the 'Change' button in a module isn't enough to apply changes, you also have to press the global 'Save' button at the top right to commit all changes.

Once you get used to this behaviour, the interface is responsive and reasonably well laid out. This is good as there is very little in the way of online help, and it's not always easy to find what you need on the wiki.



Stability and security

Above all else, a server should be reliable and security bug free.

We use the term 'stability' in this section in the Debian sense of the word, which means not changing too often – none of the software crashed while we were using it. What matters is that there is a solid Linux distribution behind the scenes, and one that will continue to provide timely security updates and fixes for significant bugs. You don't want to be running rapidly changing software on a server, you just want it to work, and to keep on working.

TheSSS is effectively the server version of 4MLinux, which is a completely independent distro. As such it has no big infrastructure behind it, making it unsuitable for anything that you might consider vaguely critical. However, TheSSS is generally unlikely to be suitable for a role where such stability and security is a key factor anyway. It's most suitable, in every way, for a small home network setup, especially if you want to able to run it on old hardware.

The other candidates are all based, directly or indirectly, on major distros. ClearOS and NethServer are both based on CentOS, which is the free rebuild of Red Hat Enterprise Linux, and you don't get much more major than Red Hat. Support, packages and security updates for them will be around for a long time. The same is true of Ubuntu Server and Zentyal (which builds on Ubuntu) as Ubuntu support LTS releases for five years and you also have the work put in by Debian.

Verdict

ClearOS ★★★★★
NethServer ★★★★★
TheSSS ★★☆☆☆
Ubuntu Server ★★★★☆
Zentyal ★★★★☆

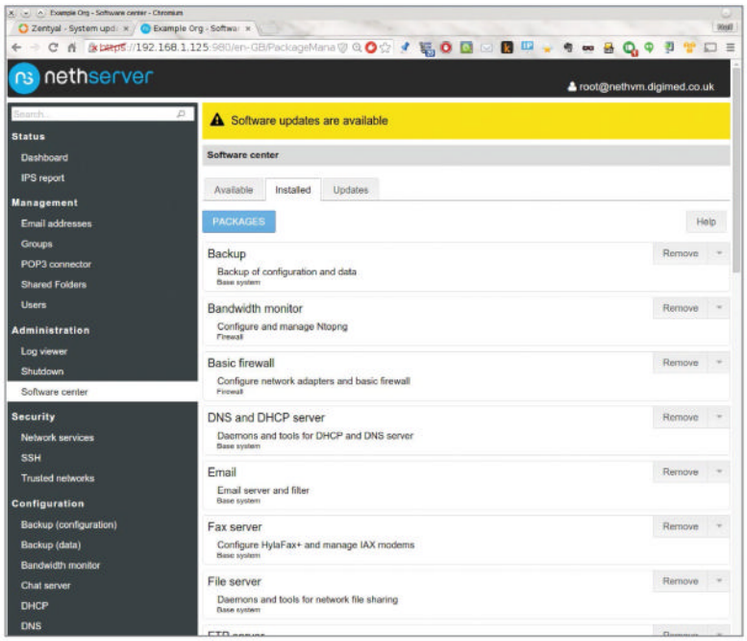
» *ClearOS and NethServer tie. Both have access to the RHEL server software.*

Documentation and support

More features mean more learning – good documentation is vital.

Point and click configuration tools are great, but you really need to have some understanding of what your clicks are actually doing, especially if you intend to expose your server to the internet.

NethServer provides good online help in its web interfaces and, if you need more, there is detailed documentation on the NethServer website, including information on how to install third-party software. The inline help for ClearOS is not quite as detailed but it makes up for it with a wealth of online resources, including manuals, howtos and a knowledge base.



NethServer – detailed inline help, backed up by online documentation.

ClearOS comes in two versions: the free Community release and the paid Professional with support. There's a 30-day free trial of the Professional system. The ClearOS application market also includes paid software. NethServer also has free and supported releases, but it doesn't keep reminding your of the paid support while you are using the free version.

Ubuntu Server has no web interface and so relies purely on the online documentation. That's not such a bad thing, because it is up to Ubuntu's usual excellent standards: being both comprehensive and comprehensible. There's also the option of a support package with Ubuntu.

Zentyal has brief inline help and a community wiki for the main documentation. This includes a section containing official documentation from the Zentyal staff and help in languages other than English. As with most of the other distros, there are community releases and those with paid support.

The documentation for TheSSS is very much like the distro: minimal. To be fair, there is much less to document and detailed information on software configuration is probably best obtained from the upstream websites. This is a purely free distro.

Verdict

ClearOS ★★★★☆
NethServer ★★★★★
TheSSS ★★☆☆☆
Ubuntu Server ★★★★★
Zentyal ★★★★☆

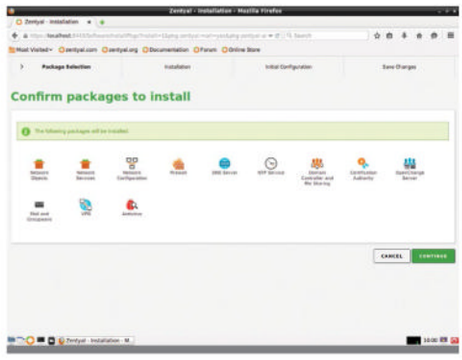
» *The clear help, both inline and on the website, gives NethServer the edge.*

More than LAMP

Server life after sending web pages.

Apart from the usual LAMP and email services, there's a lot more you can use a server for. While TheSSS stopped at the last section, with the exception of a firewall and web proxy, the rest have much more to offer. All of them can be used as a gateway server, sitting between your network and the internet, a firewall in front of it or a VPN connecting remote users directly to your network. With the exception of TheSSS, all of these distros have their roots in big, established distros, so packages are available for anything you want to do.

While Zentyal doesn't provide web or FTP services, it includes almost everything else: file sharing, domain controller, firewall, VPN, you name it. It even includes OpenChange, an implementation of Microsoft's Exchange protocol, making it a good choice for a mixed office network. ClearOS and NethServer provide just about everything, both are based on CentOS and so have access to the wealth of software. Aside from the usual LAMP, mail, FTP, file and print services, this includes web and mail proxies, chat servers, webmail, firewalls, time servers and even a fax server. Because you are responsible for installing and configuring it yourself, Ubuntu Server has the full range of Ubuntu and Debian packages to choose from, but you do the work.



There is so much more to being a server than good old LAMP, as Zentyal shows.

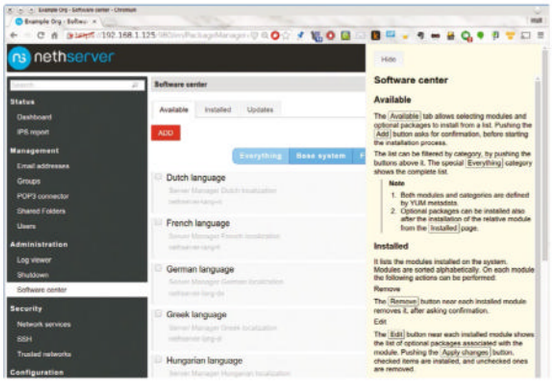
Verdict

ClearOS ★★★★★
NethServer ★★★★★
TheSSS ★★★☆☆
Ubuntu Server ★★★★★
Zentyal ★★★★★

» *Zentyal edges out the others here, thanks to the range of services on offer.*

Server distributions

The verdict



NethServer's interface isn't pretty, but it's very helpful.

There is no simple 'this distro makes the best server' answer here. Which one is best for you depends on what you want to use it for. Most of them have something that sets them apart from the rest, and not always for a good reason. The obvious outsider is TheSSS. This doesn't even try to be a full distro: it is tiny and light, the server equivalent of Damn Small Linux, making it only suitable for light duties. But that also makes it uniquely suitable for such duties, and it is the only distro here that can be run from a CD or USB stick.

Ubuntu Server also stands out as the only candidate without a web configuration tool. This is a deal breaker if you need this, but Ubuntu ticks all the other boxes. It provides all the server features you could want, courtesy of its massive package repositories, and the ease of adding more via PPAs, and provides the five-year support of the LTS releases. It also has a wealth of documentation and community support, which partially alleviates the lack of a GUI.

Zentyal stands out for two reasons: it's excellent integration into a multi-OS environment and its lack of a web server (actually *Apache* is installed for internal use but not available through the interface). This clearly states its purpose as an office server, a role in which it excels. As this is basically Ubuntu Server with the Zentyal front-end, all of the previously mentioned advantages of that distro will apply here too.

That just leaves ClearOS and NethServer to choose between, which is an almost impossible task. Both are based on CentOS 6.6; provide a similar range of service; and have access to the same collections of other software. The automated install and the more comprehensive inline help just swings it for NethServer, but the clearOS GUI won in some areas. If either one of these looks suitable for your needs, you really should try both – it's that close.

> **"If either one of these servers looks suitable for your needs, you really should try both."**

1st NethServer ★★★★★

Web: www.nethserver.org **Licence:** GPL3 **Version:** 6.6

» This enhanced CentOS wins by a short head.

2nd ClearOS ★★★★★

Web: www.clearos.com **Licence:** Various open source **Version:** 6.6

» Very slick repackaging of CentOS 6.6, well worth trying.

3rd Zentyal ★★★★☆

Web: www.zentyal.org **Licence:** Various open source **Version:** 4.1

» How an Ubuntu server should be done.

4th Ubuntu Server ★★★★☆

Web: www.ubuntu.com/server **Licence:** Various **Version:** 14.04 LTS

» Lots of ubuntu goodness, but no administration interface beyond a shell.

5th TheSSS ★★★☆☆

Web: http://thesss.4mlinux.com **Licence:** GPL3 **Version:** 13.0

» Nice for a lightweight option, but no competition for the others.

Over to you...

Do you run your own server. Is it one of these or another setup entirely? Tell Linux Format! Write to **lxf.letters@futurenet.com**

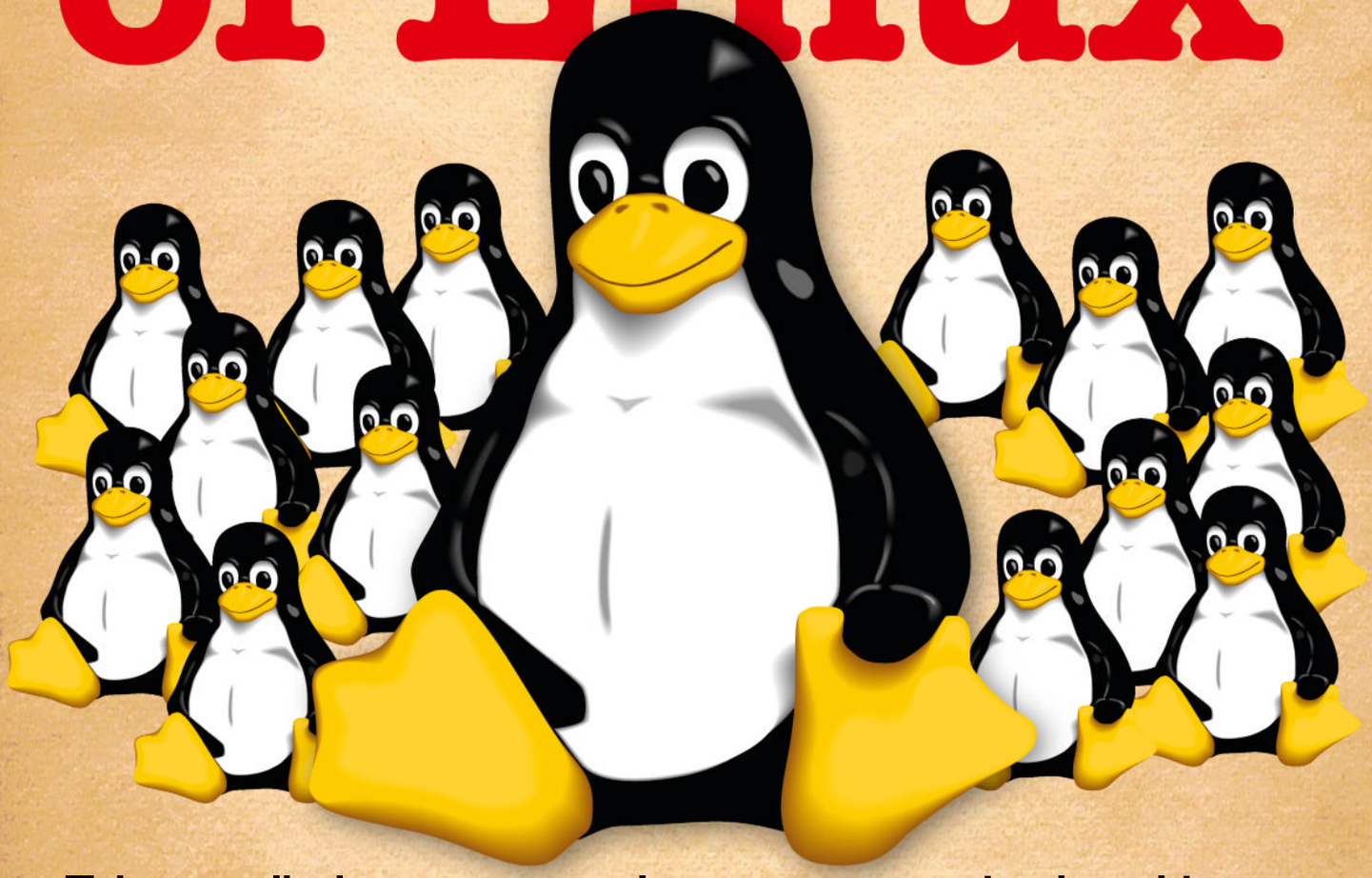
Also consider...

If you don't want to use a pre-packaged server, why not try installing the software you need on a general purpose distribution. Even your favourite desktop distro will be a good starting point, and you can remove the desktop packages once the operating system is ready to run headless and you will be working with something familiar.

Alternatively, you could use Gentoo or Arch Linux to install a server distro from scratch, including exactly what you need and no more. If you want browser based administration, *Webmin* is also a good general purpose option, but It's not quite as slick as some of the interfaces that we've shown you here, but it works with everything.

The decision rests, at least in part, on what you want a server for and why you need it. If you need something for your office setup that just works with the minimum of fuss, at least the top three candidates here will work admirably. If your server is really more of a hobby project, you will learn more by building your own.

15 Years of Linux



Take a walk down memory lane as we examine how Linux has changed over the Linux Format magazine's lifespan.

It was a cold grey morning in May 2000. Winter should have departed but that doesn't happen in Britian. So Reader Zero, seeking respite from the icy rain and miserable population, stumbled into their local newsagent. Zero was hoping for some stimulating and edifying reading material, but was mostly resigned to the notion that the shelves would be populated with the usual feuilletons, corrupt gaming magazines and various 'zines pandering to interests Zero did not possess.

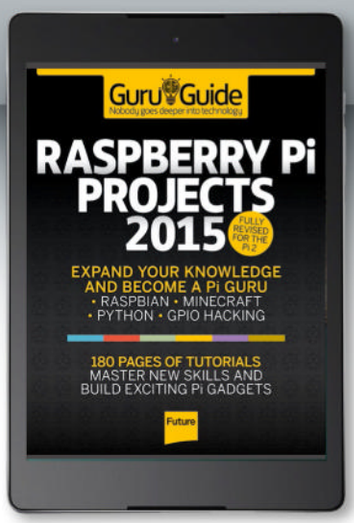
> **"A light in the darkness: 'Join the revolution!' the coverline told our enraptured reader."**

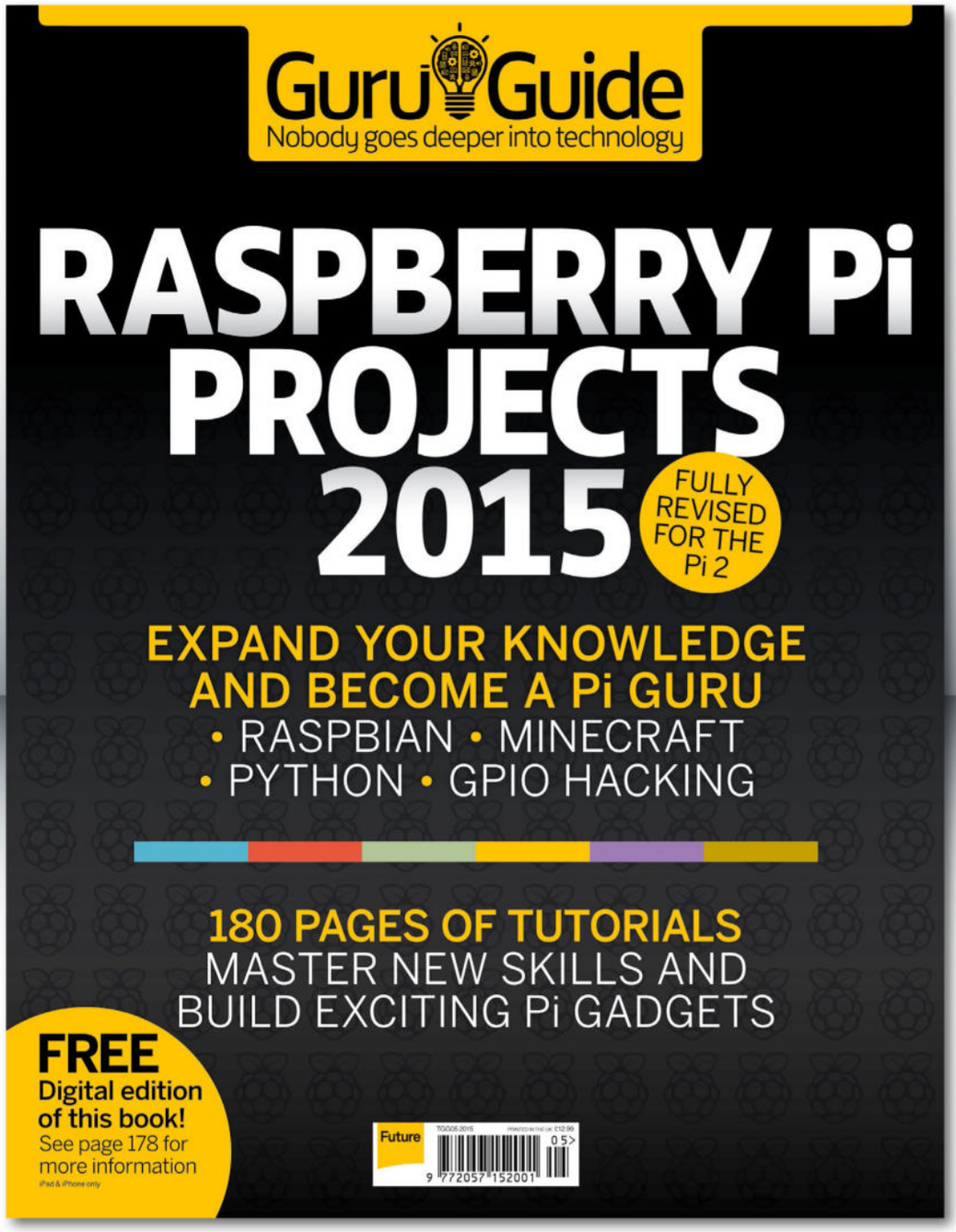
And then he saw it, fluorescent orange, a light in the darkness: "Join the revolution!" the coverline told our enraptured reader. Amazed that frustrated tinkerings at the terminal, considered by their peers an affectation rather than a hobby, could be part of something so exciting and dynamic as a 'revolution', Zero was powerless to resist. There was a free disc too, a whole Linux distribution (Definite Linux) was on there! That would take about a month to download over dial up. And there would be another one in four weeks, and eventually there would be not just a CD but a DVD. Zero's life was changed, and while Definite Linux definitely didn't last long, and the magazine would change hands many times over the next 15 years, it remained a bastion of quality publishing [until Jonni joined – Ed] that would inform, entertain and delight. »

Amazing projects to get the most from your Pi!

OUT NOW!

WITH FREE DIGITAL EDITION



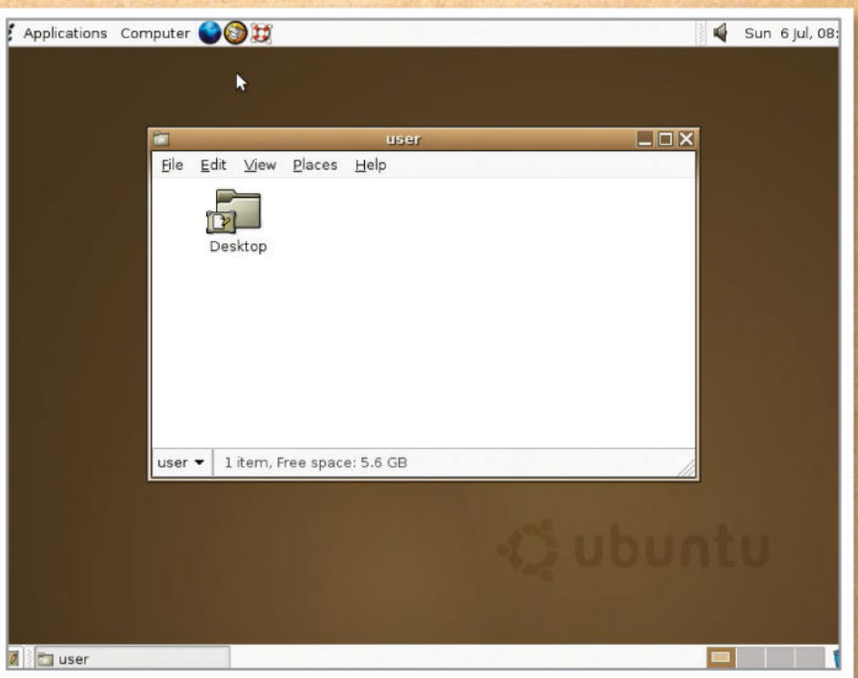


DELIVERED DIRECT TO YOUR DOOR

Order online at **www.myfavouritemagazines.co.uk**

or find us in your nearest supermarket, newsagent or bookstore!

Back when Zero was having their cathartic moment in the newsagents, Linux was already about nine-years old. Some distributions (distros) had already established themselves, and one of the earliest was Softlanding Linux System (SLS), which appeared in May 1992. Unlike its contemporaries, SLS provided more than just the kernel and some GNU tools for preparing filesystems, instead it shipped with a networking stack and the X display server. This was considered ambitious and buggy, and efforts to fix this culminated in Slackware's release in 1993. Also that year, and again in response to frustration with SLS, Debian came into being. Red Hat Commercial Linux appeared the following year, which would engender many popular distros of the late 90s, including Mandrake, Yellow Dog and Definite Linux. KDE was released in 1998, with Gnome following in 1999. Gnome was in part created due to KDE's reliance on the then non-freely licensed *Qt* toolkit. By May 2000, the most popular distributions were Debian 2.1, Red Hat 6.1, Linux-Mandrake 7.0 (this was how it addressed itself back then), Slackware 7.0 and SUSE Linux 6.3. Some of these even featured in the very first **LXF** Roundup.



The Human theme was an attempt to make Ubuntu Linux more friendly, because as everyone knows brown is beautiful, especially if you're a warthog.

What's user experience?

If you're a recent Linux convert who's had to engage in combat with rogue configuration files, misbehaving drivers or other baffling failures, then spare a thought for those early converts whose bug reports and invective utterances blazed the trail for contemporary desktop Linux. Up until comparatively recently, it was entirely possible to destroy your monitor by feeding X invalid timing information. Ever had problems with *Grub*? Try fighting it out with an early version of *Lilo*.

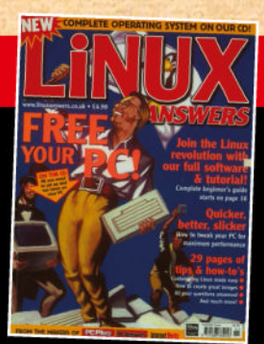
In the early days, even getting a mouse to work was non-trivial, requiring the user to do all kinds of manual calibration. Red Hat released a tool called *Xconfigurator* which provided a text-mode, menu-driven interface for setting up the X server. It was considered a godsend, even though all it did was generate an **XF86Config** file which otherwise you'd have to write yourself. So while Windows users whined about Windows ME being slow and disabling real mode DOS, your average Linux user would jump for joy if their installation process completed. Even if you got to that stage, it would be foolishly optimistic to suppose the OS would boot successfully. Hardware detection was virtually non-existent, and of the few drivers that had been written for Linux, most weren't production quality. Yet somehow, the pioneers persisted – many were of the mindset that preferred the DOS way of working, which began to be sidelined as the millennium approached. Windows users were having their files abstracted away – 'My Computer' epitomises this movement.

> **"Even getting a mouse to work was non-trivial, requiring all kinds of manual calibration."**

In January 2001 Kernel 2.4 was released and with it came support for USB and exciting new Pentium IV processors, among other things. It was of particular importance to desktop users thanks to its unified treatment of PCI, ISA, PC Card and PnP devices as well as ACPI support. The dot-com bubble was just about to burst, but all the excitement and speculation around it meant that many computer enthusiasts had a broadband connection in their home, some even enjoyed the luxury of owning more than one computer. This solved some major entry barriers to Linux: people could now download it much more easily; up-to-date documentation was easily accessible; and when Linux saw fit to disappear one's internet connection (or render the system unbootable), the other machine could be used to seek guidance. But the user experience

Timeline

Pre-history – *Linux Answers*
In late 1999 Future plc published a one-off magazine, this was borne off the back of the success of, the now closed, *PC Answers* and *PC Plus* [the flashbacks! – Ed]. All we'll say is that this was successful enough to launch a monthly magazine...



May 2000 – *Linux Format* #1
Renaming the title in line with Future's most successful print magazines: *Amiga Format* and *PC Format*, *Linux Format* was released with editor Nick Veitch of *Amiga Format* fame and writing talent from *PC Plus* mag. It came with a CD and was an instant hit.

Enjoy software freedom

Discover Linux!

DELIVERED DIRECT TO YOUR DOOR

Order online at **www.myfavouritemagazines.co.uk**

or find us in your nearest supermarket, newsagent or bookstore!

» was still, on the whole, woefully inhospitable. While some installers had evolved graphical capabilities, these more often than not were more trouble than they were worth. Users were expected to understand the ins and outs of disk partitioning, and be able to discern which packages they required from often terse descriptions.

Windows XP was released around October 2001, and while this was seen as a vast improvement over its predecessor, many users found that their machines weren't up to running it. After all, it required 64MB RAM and a whopping 1.5GB of disk space. Remember that BIOSes had only recently gained the ability to address large drives (there were various limits, depending on the BIOS, 2.1, 4.2 and 8.4GB were common barriers). So many people couldn't install it on their hardware, and many that met the minimum specs found the performance rapidly degraded once the usual pantheon of office suites and runtime libraries were installed. This provided the motivation for another minor exodus to Linux, and the retro-hardware contingent continue to make up an important part of the Linux userbase (and berate us for not including 32-bit distros). Before 2006 all Macs had PowerPC processors, and many of these (as well as early Intel Macs), long-bereft of software updates from Apple, now run Linux too.

The Gnome 2 desktop environment was released in 2002 and this would become a desktop so influential that some still seek (whether out of nostalgia, atavism or curmudgeonly dislike of modern alternatives) to reproduce it. It aimed to be as simple, tweakable and intuitive, and it's hard to argue against its achieving all of these adjectives.

Oh, we're so pretty

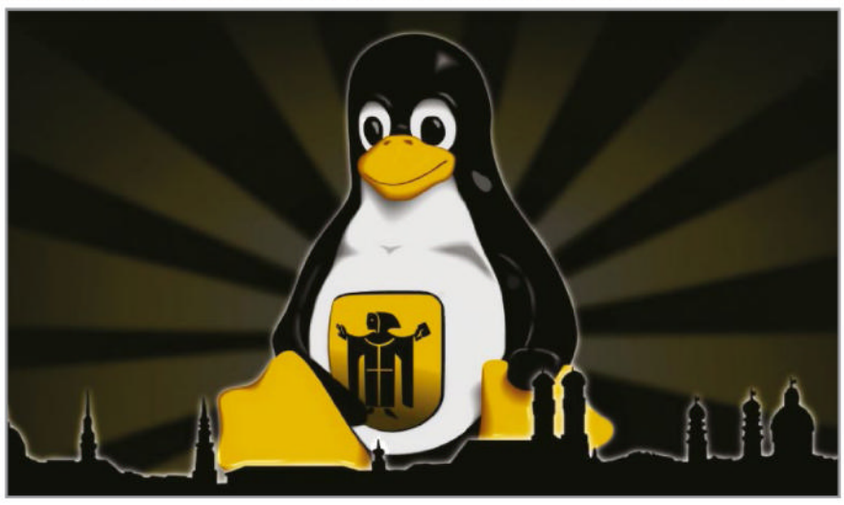
One of the major enablers was its strict adherence to the Gnome Human Interface Guidelines which set out some key principles for application designers. This meant the desktop was consistent not just internally, but in respect to all the *GTK* apps that people would go on to write for it.

> **"Gnome 2: A desktop so influential that some still seek to reproduce it."**

Also released was KDE 3, which vaguely resembled Windows – in that it was cosmetically similar and slightly more resource-demanding than Gnome. People and distributions sided with one or the other. SUSE Linux (predecessor of openSUSE) always aimed to be desktop agnostic, but most of its users preferred KDE. Heeding this, though not until 2009, it changed position and today is the leading KDE-based distro.

In late 2002, 'DVD' Jon Johansen was charged over the 1999 release of the DeCSS software for circumventing the Content Scrambling System (CSS) used on commercial DVDs. This software enabled Linux users to play DVDs, a feat they had been hitherto unable to do since DVD software required a licence key from the DVD Copy Control Agency, one of the plaintiffs in the suit. It later emerged that CSS could be broken much more trivially and Johansen was eventually acquitted. By this time iPods and piracy meant that MP3 files were commonplace. These were, and still are, dogged by patent issues with a number of bodies asserting ownership of various parts of the underlying algorithm. As a result, many distros shipped without patent-encumbered multimedia codecs. The law is murky though, and rights holders have shown restraint in filing suit against FOSS implementations of these codecs. Most distros are prudent and leave it up to the user to install these, although Ubuntu offers users the licensed (but proprietary) Fluendo codecs on install. Fortunately, many of the MP3 patents have expired and many more will have done so by 2017, it doesn't really matter – we have plenty of open formats and codecs now (OGG, FLAC, VPx and x264). It's still technically a DMCA violation to use libdvdcss (a modern and much more efficient way of cracking CSS, used by the majority of media players on Linux) to watch a DVD, but that only applies in some [backwards – Ed] countries and to date, no one has challenged its use.

The city of Munich announced in 2003 that it was to migrate all of its infrastructure from Windows NT to Linux. As well as saving costs, the Bavarians claimed the main impetus for the move was freeing them from vendor lock in. Steve Ballmer visited the mayor personally,



› **The LiMux project branded Tux with Munich's emblem, the Münchner Kindl. Apparently it didn't a hurt a bit. The project is estimated to have saved around €11 million.**

Timeline

November 2000 – LXF007

Corel Linux, the Debian-based distro, was on the CD. The OS may have failed but it was a super-easy introduction to Linux and pointed the way forward for distro developers.

April 2001 – LXF013

The beginning of a new century called for new media, so the DVD age finally hit *Linux Format*! On the first **LXFDVD** you could find SuSE Linux 7.1 and Red Hat 7.0.

May 2001 – LXF014

First a DVD and next an all-new look for **LXF**! The first redesign of the magazine cemented favourites such as HotPicks, Roundup and Answers. The International Space Station was in the Linux news and AMD64 was on the cards.

Graphics drivers and their discontents

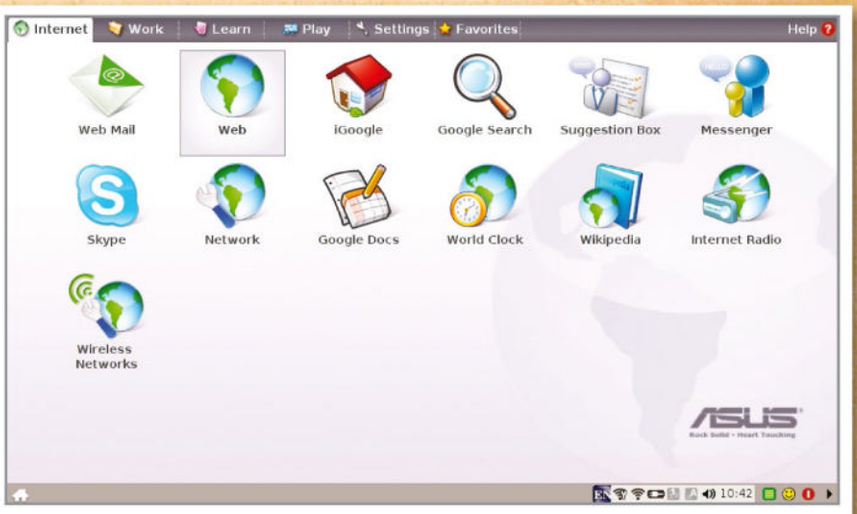
By 2003 Ati (now part of AMD) and Nvidia had both released proprietary drivers to leverage the 3D capabilities of their latest hardware (in 2005 flagship cards were the X1800 and Nvidia's 6800 series). There were open source drivers available, but performance was poor.

ATI were much more forthcoming in releasing device specifications than their opponents, as a result of which their open source drivers developed much more rapidly. Nvidia, through its nv driver, released only some obfuscated source code which left developers puzzled and frustrated. Binary drivers proved troublesome, even with helpful management tools such as Ubuntu's *Jockey*. Repositories would lag behind the latest release, which spurred users into downloading packages direct from the AMD or Nvidia. These were notoriously badly-behaved (we still don't like them now) and would wreak havoc with existing driver arrangements. Since they existed outside the package manager's purview, whenever there was a kernel update the driver module would need to be recompiled. Otherwise there would be no graphics next reboot, which, understandably, some users found upsetting.

This particular situation has been ameliorated thanks to DKMS, but graphics woes continue to be a major source of teeth-gnashing for many users. The story is in many ways still the same: open source drivers are slow and binary ones break things.

In response to the poor performance and lack of 3D support through the nv driver, the nouveau project was announced in 2006. This was a mammoth effort of clean room reverse-engineering, which relied in part on crowd sourced data: Participants would download the REnouveau program which would prod some registers, draw some graphics and then take a snapshot of the register space for developer analysis. It took until 2012 for nouveau to reach a stable release, but it appeared in some distributions some three years earlier, since even in its buggy state it proved superior to nv.

but even his charm and eloquence (and, presumably, offers of hefty discounts) weren't enough to convince the revolutionaries. The project was completed ten years later with some 15,000 machines migrated to the custom 'LiMux' distro. A scare story emerged in 2014 that the city was to revert to Windows, but turned out to be false. It's estimated that the move saved Munich some 11 million euros.

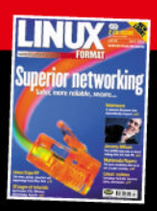


Asus' EeePC Linux was based on Xandros and *IceWM*, but beginners didn't like it, and professionals just replaced it.

0 kernel! My kernel!

After two years in development Kernel 2.6 was released in 2003. This was a vastly different beast to 2.4, featuring scheduler enhancements, improved support for multiprocessor systems (including hyperthreading, NPTL and NUMA support), faster I/O and a huge amount of extra hardware support. We also saw the Physical Address Extension (PAE) so that machines could address up to 64GB of RAM, even on 32-bit architecture. Also introduced was the venerable Advanced Linux Sound Architecture (ALSA) subsystem, which enabled (almost) out-of-the-box functionality for popular sound cards, as well as support for multiple devices, hardware mixing, full-duplex operation and MIDI. The most far-reaching new feature was the old device management subsystem, devfs, being superceded by udev. This didn't appear until 2.6.13 (November 2003), at which point the **/dev** directory ceased to be a list of (many, many) static nodes and became a dynamic reflection of the devices actually connected to the system. The subsystem udev also handled firmware loading, and userspace events and contributed to a much more convenient for desktop users. Although you still relied on such arcana as HAL and ivman in order to automount a USB stick with the correct permissions.

Linux (having already been ported to non-x86 64 bit processors) supported the Itanium's IA64 instruction when it was released in 2001. This architecture was doomed to fail though, and Intel eventually moved to the more conservative AMD64 (or x86-64) architecture, which (we delight in reminding our readers) has been around since 2003. Thanks to open source software, Linux users were running 64-bit desktops right away, while Windows users would have to wait until 2005 for the x64 release of XP. Various proprietary applications (notably *Steam* and its games) run in 32-bit mode, which provides »



April 2002 – LXF026
The second new design for the magazine in as many years! This issue also ran a very popular interview with Samba co-engineer, Jeremy Allison.

May 2002 – LXF027
This issue saw the long awaited results to the reader-voted *Linux Format* Awards 2001. Mozilla won and *Apache* too, while Mandrake picked up best distribution.

February 2003 – LXF037
We asked possibly for the first time: Is this the year of Linux on the desktop? To quote us back then: "I expect 2003 to be a real breakout year." We reviewed LindowsOS 3.0, *Unreal 2003*, while we still liked *IceWM*, KDE and *WMaker*.

» some motivation for distributions to maintain 32-bit releases, but the day will come when these are no longer tenable to maintain, and eventually they will go the way of the 386, no longer supported on Linux since 2013.

Enter the archetype

The 2004 release of Ubuntu 4.10 ('Warty Warthog') was, without a doubt, a major boon for Linux on the desktop. Using the megabucks he'd amassed from creating and selling Thawte, Mark Shuttleworth formed Canonical Inc. The goal was to sell server products and support and at the same time make a desktop Linux "for human beings". Using Debian (it having proven itself by this point) as a base, Canonical added driver tweaks, a very brown Gnome 2 theme and an ambitious six-month release cycle. We also saw the launch of **http://ubuntuforums.org**, where well-meaning but ill-informed members of the community would post 'solutions' to various Ubuntu problems.

In 2004, a sound server called Polypaudio was released by a hitherto unknown developer called Lennart Poettering and some others. At this time desktop environments relied on sound servers to overcome shortcomings in ALSA's dmix system: Gnome was using the Enlightened Sound Daemon (ESD) and KDE was using the analogue Realtime synthesizer (aRts). Polypaudio was designed to be a drop in replacement for ESD, providing much more advanced features, such as per-application volume control and network transparency. In 2006 the project, citing criticism that nobody wants polyps, renamed itself PulseAudio (it was in fact named after the sea-dwelling creature, not the medical condition).

With its new name and increased demand for a sound system comparable with that of OSX or the newly released (and much maligned) Windows Vista, PulseAudio enjoyed substantial development and began to be considered for inclusion in many distros. As is traditional, Fedora was the first to adopt, incorporating it as the default in version 8, released in late 2007. Ubuntu followed suit in 8.04, although its implementation attracted much criticism and resulted in much anti-Pulse vitriol. Poettering at one stage even described his brainchild as "the software that currently breaks your audio". It took some time but eventually Ubuntu (and other distros) sorted out implementation issues, and it now mostly works out of the box.

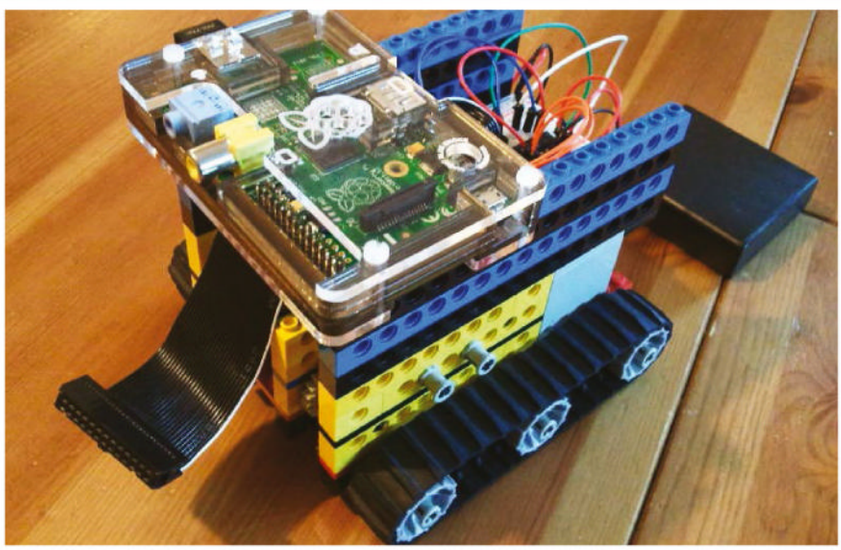
Before tablets, and smartphones that people could afford, netbooks were the pinnacle of portable computing. The first one was the Asus EeePC 701. Due to its low hardware spec (it had a 700MHz processor, 800x480 display and 512MB of RAM) running Windows on it was not an option. Instead it came with a customised version of Xandros Linux, which was functional, but lacking in polish. On the whole most people were unhappy with it, but netbooks still proved great platforms for more experienced Linux users. As newer netbooks were released

Raspberry Pi revolution

The Raspberry Pi was released in 2012. Inspired in part by the success of the BBC Micro (hence the monogram model names) in the early 1980s, the Raspberry Pi aimed to bring practical computer science to the classrooms and bootstrap the UK electronics industry. The low-cost, credit-card sized computer has sold in excess of 5 million units.

While many of these are now empowering young coders, a great deal have become part of diverse man cave projects: The 30-somethings who cut their teeth on BBCs, Spectrums, C64s reliving and reviving the thrills at the interface of coding and creativity. The Pi's GPIO pins mean that all manner of add-ons have been developed, so that the Pi can power anything from robots to remote watering systems.

The lingua franca of Pi projects is Python which, like Basic, is easy to learn. Unlike Basic, though, it is consistent, extensible and won't need to be unlearned should users move on to more advanced languages. The Pi's support for 3D graphics is impressive, but CPU-wise it is more limited. The original Pis struggle to function as a desktop computer, even with the modest Raspbian distribution (although recent work on the *Epiphany* web browser has improved this). In 2015 the Pi received a reboot, gaining a quad-core processor and extra RAM, so now it is a truly multi-purpose computer, and it still only costs £25.



The Raspberry Pi has inspired a whole new maker generation. When the robot overlords rise up we can blame the Foundation...

Time line

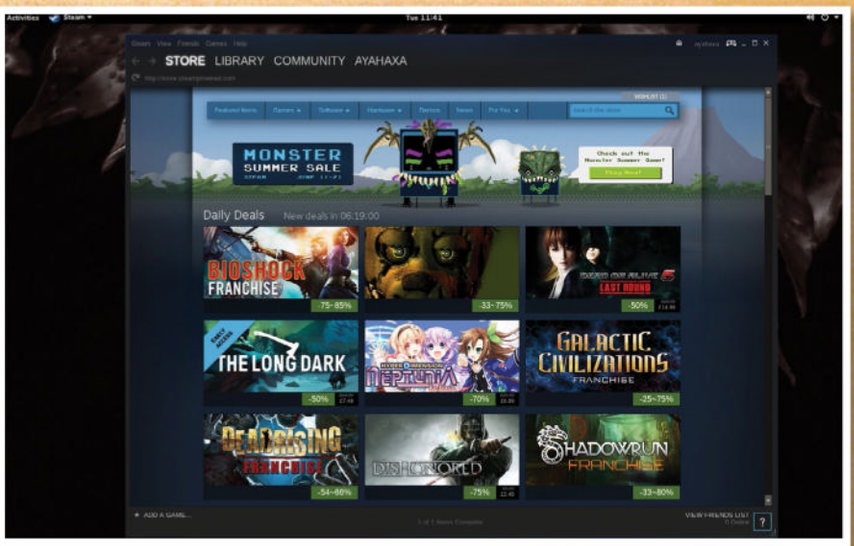
December 2004 – LXF060
The first review of Ubuntu 4.10 by a chap called Jono Bacon, scandalous we're sure; he liked it oddly enough. No, it wasn't on the disc but Mandrake 10.1 was!

January 2005 – LXF061
LXF runs the stalwart Best Distro feature and Mandrake easily wins, poor Ubuntu comes joint 9th. Some chap called Graham Morrison starts as a staff writer and we run Ubuntu on the **LXFDVD** for the first time, alongside Fedora Core 3.

October 2006 – LXF084
The last redesign of **LXF** landed, and the magazine here is largely the design still used today – with the odd section change – the **LXFDVD** was also moved inside the mag.

June 2008 – LXF106
Boy genius, Paul Hudson was promoted to editor/High Commander and we put the Asus Eee PC centre stage, Jonni still has his running Arch Linux...

(many based around the more suitable Intel Atom chips) they started to ship with Windows XP (some seven years after its initial release) and then the crippled Windows 7 Starter Edition. Asus later backpeddled on its Linux enthusiasm: Teaming up with Microsoft it even launched an 'It's better with Windows' campaign, designed to deter people from purchasing Linux-based laptops. This smear campaign used phrases like 'major compatibility issues' and 'unfamiliar environment' to scare people away.



Thanks to *Steam on Linux*, Tux gamers finally have thousands of games to play, and LXF writers can peruse the Summer Sale offerings and still claim to be doing work.

The cost of progress

The year 2010 may be remembered by some as the one Ubuntu started to lose the plot. Up until now, the distro had been going from strength to strength, gaining more users, more stability. It was the poster child for the (dead or irrelevant depending on who you ask) dream of Linux on the desktop. But things started to go awry in the 10.10 release. Its *Ubuntu Software Center* now included paid-for apps (the first one was Fluendo's licensed DVD player) and the Netbook remix used a new desktop environment called Unity. In the 11.04 release though, this became the new shell for the main release too. Ubuntu had long taken issue with the new Gnome 3 desktop, which at the time of the Ubuntu feature-freeze was not considered stable enough to include in the release anyway, and Gnome 2 was already a relic. So in a sense Ubuntu had no choice, but no one likes change, and users were quick to bemoan the new desktops. Ubuntu has persisted with Unity and it's much improved today, but a low point came with the 12.10 release when users noticed 'suggestions' from Amazon as they typed queries into the search lens.

Gnome 3 is not without controversy too – the criticisms it attracted were threefold: First, many preferred the old Gnome 2 way of doing things and this clearly was not that. Second, all the fancy desktop effects required a reasonable graphics card (and also working drivers). There was a fallback mode, but it severely crippled desktop usability. Finally, this appeared to be something designed for use on mobiles or tablets, yet even today mobile Linux (not counting Android) has never taken off, so why should users be forced into this mode of thinking? Many found though, that once some old habits are unlearned and some sneaky keyboard shortcuts are learned (and *Gnome Tweak Tool* is installed), that the Gnome 3 way of working could be just as efficient, if not more so, than its predecessor. KDE users looked on smugly, having already gone through all the rigmarole of desktop modernisation (albeit less drastic than Gnome's) when KDE 4 was released in 2008. Around this point we ought to mention *Systemd* as well, but there's not much to say that hasn't been said elsewhere: the old init system was creaking at the seams, a new and better one came along, it wasn't everyone's cup of tea, but we use it anyway, the internet slanders Lennart Poettering.

> **"The last couple of years have been full of Linux developments and dramas."**

There has always been a niche interest in gaming on Linux, but this was mostly done through *Wine*, which has been around since the mid 90s. Things changed when Valve released its *Steam for Linux* client in 2013. Today there are over 1,000 games available for Linux, with more being ported all the time. Granted, many of the high profile ports incorporate either a *Wine* layer or a wrapper such as eOn, but we are also seeing a good proportion of indie releases running natively. Valve even made an OpenGL version of zombie splatterfest *Left 4 Dead 2*, which outperformed the DirectX/Windows release. Linux users make up about 1% of the Steam userbase at present, but this may change if Valve's plan to conquer the living room [why not the desktop!? – Ed] through Steam boxes, running the Debian-based Steam OS, comes to fruition.

The last couple of years have been full of Linux developments and dramas too, including the Heartbleed bug, a partial resolution to the long-running SCO-IBM lawsuit and a much less adversarial stance from Microsoft. But there just isn't enough space, alas.

July 2009 – LXF120
We celebrate Ubuntu 10.04 by putting it on the cover and interviewing Mark Shuttleworth, again. The man just won't leave us alone!

August 2010 – LXF134
What's this Android thing and how can it even possibly have a chance of taking on the iPhone? We explained why, plus Mint 9 and Fedora 13 on the **LXFDVD**.

April 2012 – LXF156
We reported on this thing called the Raspberry Pi back in **LXF147**, but finally the world could buy this tiny PC marvel and the world loved it.

January 2014 – LXF179…
A new editorial team lands at *Linux Format* Towers as the old team departs for pastures new. **LXF179** is the top-selling issue of the year and **LXF181** is the best seller for almost two years! Thank you for helping us keep **LXF** the UK's best seller!

HACKER'S MANUAL 2016

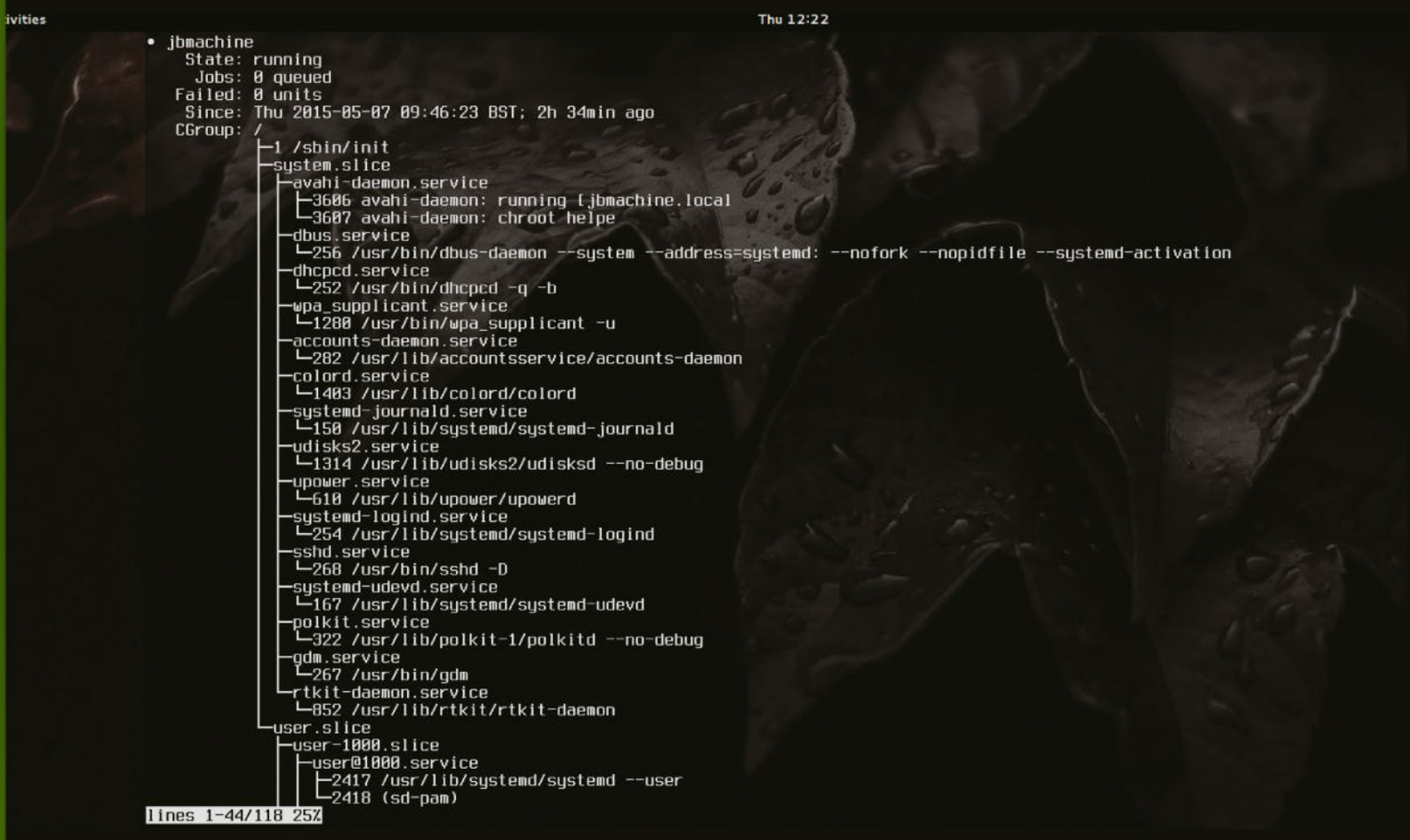
HACKER'S MANUAL 2016

Software

From apps to games to the Linux environment – change it up!

Systemd

Wait! Don't leave. Systemd really isn't going to eat your computer, and in fact it isn't all that bad...



Since being made the default init system by Fedora 15 in 2011, *Systemd* has, despite the controversy, seen steady adoption by other distributions. Having made it into both the latest Debian and Ubuntu versions, only Gentoo and Slackware remain as major stalwarts of ye olde *SysVinit*.

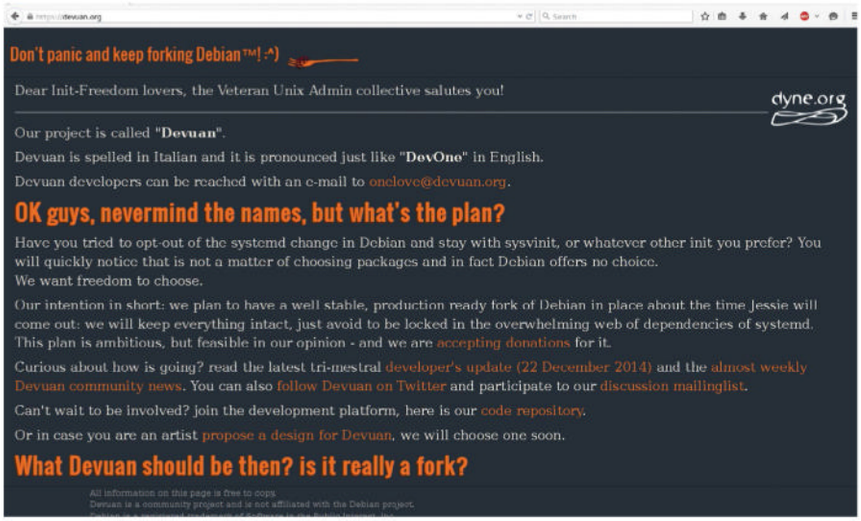
There are, of course, a number of smaller and niche distros that do likewise, but the lack of any major exodus of users to any of these distros provides anecdotal evidence that they are at least satisfied with *Systemd*'s performance and are unswayed by the ideological concerns surrounding it. Indeed, desktop users will typically have witnessed much improved start up times thanks to its parellelisation of startup services and the way it unifies what is a disparate collection of scripts and daemons makes it much more appealing for junior sysadmins, and Linux Format magazine has covered *Systemd* fairly extensively [*Tutorials, p68,* **LXF191**, *Tutorials, p70,* **LXF188**]. But new features are being added all the time and many users are unaware of those that have been there for some time. We'll probe *Systemd*'s innards and see what it's up to, what it can do, and how to stop it doing what we don't want it to. But first some background.

> **"Unifies a disparate collection of scripts and daemons makes it much more appealing."**

Systemd **is a system and service manager. Its primary modus operandi is as an init system, so its main binary is symlinked to the file /sbin/init, which is run as Process ID (PID) 1 after the kernel is loaded.** ***Systemd*** **will then dutifully start all services (making it, literally, the mother of all processes) and continue to manage them until shutdown, whereupon it unloads itself and the machine is halted and powered off.**

The previous init system, known as *SysVinit*, originated in System V – an early version of Unix – and as such is little more than an arcane collection of scripts held together by greybeard magic. This worked well enough, but as Linux distributions (distros) evolved it began to falter.

It defined six runlevels which distros either ignored or abused, and service dependencies and priorities were particularly awkward to work with. So in 2006 Canonical set about developing a replacement, known as *Upstart*. This was entirely backwards-compatible with *SysVInit*, but also provided much better dependency handling and enabled things to be done and responded to asynchronously. Besides Ubuntu, *Upstart* was adopted by all the Red Hat distros as well as Chrome OS. But by 2013 the major distros had all gone the *Systemd* way. In 2014, the Debian Technical Commitee voted to move to *Systemd*, as opposed to *Upstart*, which led to Ubuntu following suit. In a sense, this was the final nail in *Upstart*'s coffin, at least on Linux (*Systemd* doesn't support other kernels, such as the BSDs or Hurd, which is a bone of contention).



Devuan is a Debian fork which eschews *Systemd*. It's still in a pre-alpha state though, so you'd be better off with Slackware, PCLinux OS or Gentoo if you want a *Systemd*-free distro.

Seats and sessions

One reason for *Systemd*'s widespread adoption is its unified provision of desktop-centric features. Its *logind* component (besides usurping the old login service) formalises the concepts of seats, sessions and users, so that – with suitable hardware – managing concurrent local desktop sessions is trivial. While not everyone will use this, a side-effect is that the older ConsoleKit logic is now entirely obselete. Back in the day, anyone not using a full desktop environment would have had to fight with this mess in order to be able to mount USB sticks or shut down the system without requiring root privileges, resulting in many an angry post on many a forum. *Systemd-logind* also enables the X server to be run as a user which increases security. Conversely though, desktop environments, particularly Gnome, have started to rely on *Systemd* components (not the init system itself – this is irrelevant here) which has attracted some ire since installing these components alone (or using them without using *Systemd*'s init system) can be tricky. The commands **reboot**, **halt**, **shutdown** all require root, however *systemd-logind* (together with the **polkit** package) enables these functions to be performed by any locally logged-in user with an active X session. Such a user will be able to turn the computer off with:

```
$ systemctl poweroff
```

provided, of course that no other users are logged in, and if there are the user will be prompted for the root password. You can also substitute **poweroff** for **suspend** or **hibernate** provided their hardware supports it. *Systemd-logind* also handles power and sleep button events, which traditionally have been the job of acpid. These are configured in the file **/etc/systemd/logind.conf** which provides the following self-explanatory defaults:

```
IdleAction=ignore
HandlePowerKey=poweroff
HandleSuspendKey=suspend
HandleHibernateKey=hibernate
HandleLidSwitch=suspend
HandleLidSwitchDocked=ignore
```

Internal (infernal?) Journal

Gone also is ye olde syslog service, (well mostly, *Systemd* can forward messages to a syslog daemon if required). *Systemd*'s *journald* daemon will be more than sufficient for Joe User's log management requirements. Prior to *journald*, messages were gathered from the kernel and any running (or failing) services by a syslog daemon, which would filter those messages into text files in **/var/**

Life without Systemd

Some distros, while using *Systemd* by default, will permit you to use an alternate init system if you so desire. Support for this varies, eg Ubuntu 15.04 makes the process very easy: both *Systemd* and *Upstart* are installed out of the box and you'll find an 'Ubuntu ... (upstart)' entry in the Advanced options for Ubuntu Grub submenu. Those who are seeking a more permanent switch can install the **upstart-sysv** package and run:

```
$ sudo update-initramfs -u
```

For now, most Ubuntu users will not run into any difficulties with (and many will probably not even notice any difference between) the two systems. This will change in the future though, especially after the LTS release next year, as the dust settles and *Systemd* becomes ingrained into the Ubuntu ecosystem.

It would be remiss of us not to mention another init system: *OpenRC*. While technically not a replacement for SysVinit, it does extend and modernise everything that happens after PID1. *OpenRC* is maintained – and used by default in – Gentoo, which up until 2007 used a clunky pure-shell solution. Since udev has been merged into *Systemd*, refuseniks have to use eudev, another Gentoo machination forked from udev prior to its assimilation. But don't fret, you can use both *OpenRC* and Eudev in other distros too: Arch Linux has packages in the AUR. Some de rigueur packages (eg X.org) rely on *Systemd* libraries so you won't be able to purge the beast entirely.

» **log**. Userspace processes would also put their own logs in here directly. In order to prevent this directory becoming humoungus, one would install and configure `logrotate`. With *Systemd* all logs are centralised and can be accessed with the **journalctl** command. Of course, if you still need a syslog implementation then this can be run in tandem with journald, but most people will manage without. Executing `journalctl` will show logs going back as far as journalctl remembers. These are automatically piped through `less` for ease of scrolling. By default, historic logs won't be deleted unless disk space falls below what is specified by the **/etc/systemd/journald.conf** file. There are three options that you may decide you want to tweak here:

» **SystemMaxUse** This specifies the maximum disk space that the journal will occupy, this defaults to 10% of the filesystem storing the journal.

» **SystemKeepFree** The minimum space that *Systemd* will try to keep free on the filesystem holding the logs. If this is set higher than available space, the value is adjusted to the amount of free space when *Systemd* was started.

» **SystemMaxFileSize** The maximum size of each individual journal file. Ultimately this tells *Systemd* how many files to break the logs into, so that when they are rotated this much history will be lost.

History's all well and good, but if one just needs to see logs from today, then the `-b` switch will show only messages from the current boot. Whenever something doesn't work, the Linux aficionado's instinctive response might be to check the output of

```
$ dmesg | tail
```

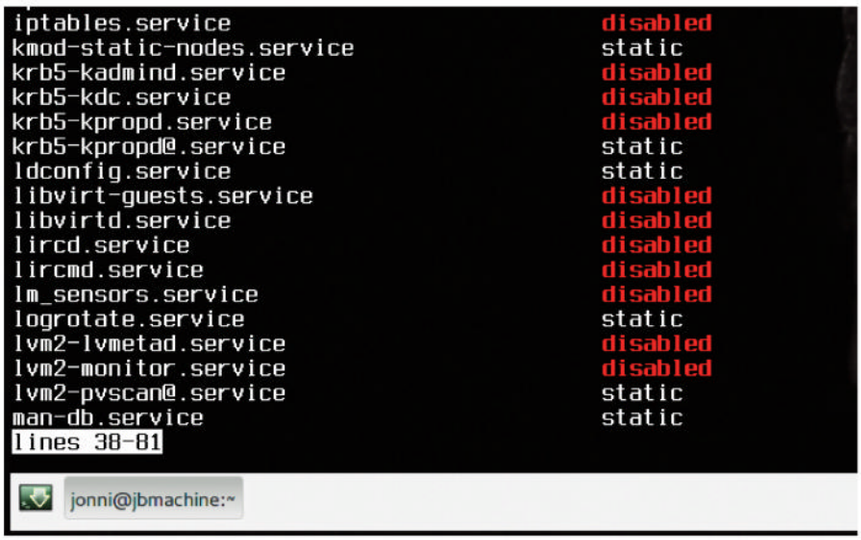
for any telltale error messages from the kernel, or

```
$ tail /var/log/messages
```

for messages from elsewhere. The *Systemd* equivalent is to run

```
$ journalctl -e
```

which allows you to scroll upwards from the end of the journal. Of course, **dmesg** still works, but this way we see messages from sources besides the kernel as well, and the timestamps are automatically displayed in local time, rather than seconds since system boot. If something went wrong on a previous boot, then we can check those logs by adding a number to the `-b` switch. Adding `-1` refers to the current boot (the default for `-b`), `-2` the previous boot and so on. You can also use absolute indexing here, so 1 refers to the earliest boot in *Systemd*'s logs (the same as if you call it without the `-b` option), 2 the next, and so on.



› **Unit files everywhere. These are the lifeblood of *Systemd* and by extension your computer.**

The binary debate

Systemd's logs are stored in a binary format for ease of indexing. This allows for a lot of data to be searched swiftly, but is also something of a bone of contention. Binary logs are more prone to corruption, so in theory a disk failure might only nerf a 4k sector of a text file, but could corrupt the entirety of a *journald* binary.

Text files lend themselves to parsing with Perl, grep, sed, awk and the like, and many sysadmins make use of scripts incorporating these for working with log files. The fact that scripts will no longer work seems to have drawn a fair amount of ire from some sysadmins, but we think such criticism is unwarranted: if you need text files then newer versions of *syslog-ng* will pull them out of journald for free.

Systemd's most fundamental units are imaginatively-titled unit files. The command

```
$ systemctl list-unit-files
```

will display a list of all of them and show their statuses. Unit files all live in either the **system/** or **user/** subdirectories of *Systemd*'s main directory (usually **/usr/lib/systemd/**). Unit files may be services (eg, sshd.service) which start programs, daemons and the like, or they can be more abstract things, such as mountpoints, sockets, devices or targets. Targets are a more flexible interpretation of *SysV*'s runlevels, they define

Systemd – what's not to like?

By far the most vociferous complaint against *Systemd* is its supposed contravention of traditional Unix philosophies around having one tool that does one thing well, and plays nice with other tools that in turn do their thing well.

Systemd stands accused of being a monolithic blob which usurps (among others) udev, cron, PAM, acpid, and logind. Having all these components all rolled up in a single binary running as PID1 upsets some people, but much of the cant and invective flying around is largely ill-informed. The fact that *Systemd* has been so widely adopted ought to corroborate its appropriateness, but instead the naysayers claim a conspiracy, a 'do-ocracy' even, is afoot, where the developers are imposing their preferences on users.

In its praise, *Systemd* provides all kinds of modern features: fair apportioning of resources through kernel cgroups, remotely accessible logs, much improved chroot environments (through *systemd-nspawn* and *machinectl*) and faster boot times, to name but a few. Trying to understand the boot process is always going to be daunting for a novice user, but at least with *Systemd* the problem is easier with components being cleanly divided and using modern syntax: the polar opposite to the Lovecraftian nightmares you would encounter in days of yore.

Of course, *Systemd* is still relatively young, and some upcoming features that have been whispered fuel further concerns: Do we really want to amalgamate PID1 with its own bootloader? Do you want to run a stateless (no static configuration files) system? We'll see how it all pans out.

a set of services to start for a particular purpose. Desktop systems will boot into the **graphical** target by default, which is pretty much runlevel 5 insofar as it (hopefully) ends with a graphical login, such as Gnome's GDM or the lightweight SDDM. Servers will boot into **multi-user.target**, analogous to runlevel 3, which instead boots to a console login. If one examines the **graphical.target** file one will see, besides others, the lines:

```
Requires=multi-user.target
Wants=display-manager.service
```

This tells us that our **graphical** target encompasses everything in the multi-user target, but also wants a display manager to be loaded. The system can be forced into a particular target (but only with root privileges) using, for example:

```
$ systemctl isolate multi-user.target
```

The display-manager.service file is actually a symlink which gets set up when a display manager is installed, it points to a service file. Services are added to *Systemd* targets using the command `$ systemctl enable`, which just makes the requisite symlinks. For example, to start the SSH daemon on next boot, run:

```
$ systemctl enable sshd
```

and you will be informed of *Systemd*'s actions:

```
Created symlink from /etc/systemd/system/
multi-user.target.wants/sshd.service to /usr/
lib/systemd/system/sshd.service.
```

When things go wrong

It is an ineluctable truth that, from time to time, stuff will break [Ed – isn't that the second law of thermodynamics). Sometimes that which breaks will leave in its wake unbootable systems, and nobody likes working with an unbootable system. Commonly, graphics drivers will be at fault, and the system, having failed to start the graphical login manager, will just sit there, helpless and silent. To rectify this, you should reboot (and hopefully the machine will still let you do that gracefully) and add the following option to the kernel commandline (press **e** to edit it from the *Grub* menu):

```
systemd.unit=multi-user.target
```

Booting with this option (by pressing Ctrl-X) will prevent the errant display manager from loading, so that driver problems can (hopefully) be repaired from the command line. For more serious boot-impeding problems, you may have to recourse to the **rescue** or **emergency** targets, or in extreme cases **chroot**-ing in from another OS.

Of course, not everything that breaks will result in an unbootable system. Symptoms might be strange error messages flashing past too quickly to read, or sometimes things will be subject to an annoying 90s timeout before the boot can continue. Besides looking at the journal, you can get a helicopter view of system health with:

```
$ systemctl status
```

which shows any queued jobs and lists all currently running service files and processes (again piped through **less** for your scrolly enjoyment). If the second line reads:

```
# State: degraded
```

(with the adjective coloured in a particularly panic-rousing red) then something is wrong. Typically a unit file has failed to load for some reason. This can be investigated further with:

```
$ systemctl --state=failed
```

Once the rogue unit has been identified, we can use `journalctl` to see if it left any useful information in the journal, eg If the above command reported something wrong with sshd.service we can go on to query anything it recently wrote to the journal by using the command:

```
$ journalctl -eu sshd
```

This will hopefully provide sufficient informations to diagnose and resolve the issue. Restart the service with:

```
$ systemctl restart sshd
```

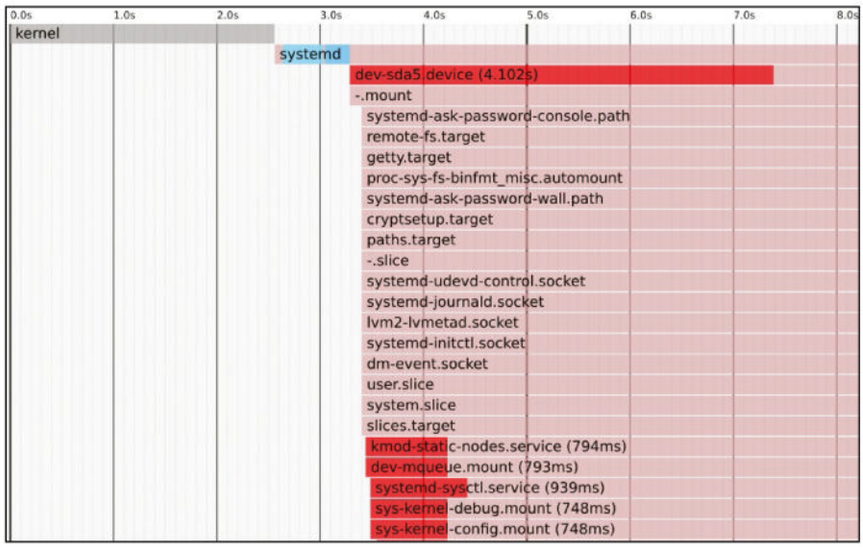
and hopefully all will be well, in which case *Systemd*'s status will change from a worrisome 'degraded' to a business as usual 'running'. Some userspace processes will also write to the journal, which we can also filter by process name (using the `_COMM=` option), absolute path or PID (`_PID=`). Since Gnome 3.12, X.org logs are no longer written to the oft' scrutinised (and now oft' searched for) **/var/log/Xorg.0.log** file. Instead, they now reside in the journal, which you can filter with either:

```
$ journalctl -e _COMM=Xorg
```

or using:

```
$ journalctl -e /usr/bin/Xorg
```

If you're using Gnome on Fedora or Arch Linux, then you will need to use `Xorg.bin` or `gdm-x-session` in the _COMM argument that we've mentioned above.



Nobody enjoys a good plot more than we do, especially one that provides detailed information about the boot process made by systemd-analyze.

> **"Systemd's fundamental units are imaginatively-titled unit files."**

Speed up boot

One particularly nice feature of *Systemd* is its ability to analyse boot times. The command

```
$ systemd-analyze
```

will show you a summary of how much precious time was taken by the kernel and userspace portions of the boot process. For more detail add `blame` to the command which will show you the time taken by individual services. This will list the most time-consuming processes first, but be aware that since things are, to use the *Systemd* parlance, aggressively parallelized", the times listed here may be much longer than the time it takes to get from *Grub* to your login screen/prompt. For our final trick, you can even make a nice SVG plot showing all the glorious timing information using:

```
$ systemd-analyze plot > plot.svg
```

After reading through our guide you'll now find *Systemd* to be a less scary prospect and perhaps slightly less of a villain of the piece in the sometimes ranty sysadmin world.

TOP 100 LINUX TOOLS

Take a stroll through the open source garden as we pick the best apps, tools and utilities available to all Linux kind.

We all have our favourite open source apps that work for us better than any available alternative. But take a moment and step back from the *Emacs* vs *vim* type battles raging on in the Linux-verse and marvel at the sheer number of apps at our disposal. Your distros' software repositories give you access to thousands of apps, and you can install everything from fully featured app suites to nifty command-line utilities literally with the touch of a button.

There are open source apps and tools for all kinds of applications today. There's hardly any use case that isn't catered for by a community contributed app. Many of these apps have proved their mettle and offer features and performance benefits that surpass their proprietary counterparts. They have also proved themselves to be invaluable to home and business users in more than one sense of the word. According to rough estimates on **www.openhub.net**, some popular apps such as *LibreOffice*, *Firefox* and *Apache* would take several hundred person-years to develop and cost millions of pounds. Yet they are all available to you for no-cost.

> **"Many of these apps have proved their mettle and surpass their proprietary counterparts."**

Open source apps come in many shapes and sizes and you can grade them based on their usability. There are feature-rich apps, task-oriented app suites, well put-together tools, and newfangled novelty apps and games. Some ship with well-designed graphical interfaces and others show their more versatile sides when operated from the command-line.

In this feature, we traverse this diverse and vast collection of open source gems on offer and pick the ones that are at the top of their game. In this list of the 100 best apps we've covered a wide range of categories. Whether you are a business owner, an educational institution, a developer, a home user, or a gamer, we've got something for everyone. While you'll be familiar with some of the most popular tools in this list, rest assured there are quite a few that might have missed your attention. If you've been unable to escape the clutches of commercial software, we're sure you'll find quite a few tools on this list that are suitable replacements.

Essential apps

A Linux desktop isn't complete without them.

LibreOffice

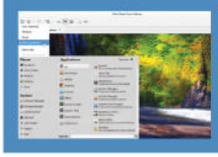
Forked from *OpenOffice.org*, *LibreOffice* has become one of the most popular office productivity suites. It includes programs for word processing, and can create spreadsheets, slide shows, diagrams and drawings, maintain databases, and compose mathematical formulae. It also offers good compatibility with documents in proprietary formats and has recently had a face lift.
www.libreoffice.org

Wine

Despite the increasing number of cross-platform apps that work on Linux, there are some that still only support Windows. This includes big third-party proprietary apps, such as *Adobe Photoshop* or just small niche home-grown tools that you can't do without. For such situations, you can use *Wine*, which generally run these Windows-only apps and games with ease. The project supports over 20,000 apps. Some work flawlessly out-of-the-box while others require minor configuration tweaks.
www.winehq.org

Remmina

With *Remmina* you can access a remote computer from the comforts of your desktop. It supports the widest range of protocols and will connect to all kinds of remote desktop servers. The app is easy to use, and has enough features that make it a viable option for occasional use.



http://remmina.sourceforge.net

Thunderbird

Another gem from the Mozilla Foundation, *Thunderbird* is one of the best email clients, being easy to setup and is brimming with features. Simple setup wizards aid syncing with popular web-based email services and it can manage multiple accounts, supports encryption and is extended through add-ons.
www.mozilla.org/thunderbird

KeepassX

Trying to remember different passwords for the various services is a challenge for most humans (that don't count cards in Las Vegas for fun). You can defer this task to *KeePassX* which stores password in an encrypted database. It can fill in the password automatically and also includes a random password generator.
www.keepassx.org

BleachBit

A distro accumulates a lot of digital gunk over time. *BleachBit* helps you spring clean it and protect your privacy. It also removes temporary and other unnecessary files, and has tools to securely delete files or wipe them.
http://bleachbit.sourceforge.net

OpenSSH

When you need to interface with a remote computer, you cannot do without OpenSSH. It's a family of tools that provides secure tunnelling capabilities by encrypting all traffic and includes several authentication methods, and supports all SSH protocols.
www.openssh.org

Gufw

You may not be using a firewall currently, and if that's because they are difficult to set up then you need *Gufw*. It features an intuitive graphical interface for managing the inbound and outbound traffic rules for various apps and services and even individual ports. Its wizard-like graphical menus are designed especially for inexperienced users.
www.gufw.org

Gufw has profiles and preconfigured rules to aid inexperienced users.

VirtualBox

When *Wine* doesn't cut it you can use *VirtualBox* to run an entire Windows installation inside a virtual machine. The software is also useful for installing experimental apps that you don't want to deploy on a real computer, and for testing other OSes without exposing it to real hardware.
www.virtualbox.org

Clonezilla

This is a cloning solution that's distributed as a live CD and is popular for doing bare metal backup and restoration of individual PCs. It can also deploy an image to multiple computers in a lab. Clonezilla can work with a large number of popular disks, partitions and filesystem types.
www.clonezilla.org

VLC

Distros ship with a functional video player. But if you need more control, there's no beating *VLC*. It supports virtually every video and audio format out there and includes handy CLI tools for advanced users.
www.videolan.org/vlc

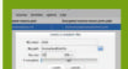
PeaZip

PeaZip is a graphical archiving tool that can work with over 130 different types of archive files and can even create encrypted ones. It integrates with popular desktops and also has a CLI for advanced users.
http://bit.ly/PeaZipSF

Gparted

Use *Gparted* to restructure a disk on your computer. It's available as a live CD and can also be installed inside your distro. *Gparted* can create, resize, move, delete, reformat or check partitions and supports many filesystems.
www.gparted.org

ZuluCrypt



Create an encrypted disk within a file or within a non-system partition or USB disk. *ZuluCrypt* has an intuitive user interface and can be used to encrypt individual files with GPG.
http://bit.ly/zuluCrypt

HomeBank

This is a feature-rich finance app. It can import data from other apps and bank statements in popular formats. It can also detect duplicate transactions and features dynamic reports and is easy to use for budgeting.
http://homebank.free.fr

Internet apps

Get the best of the web with these tools.

Firefox

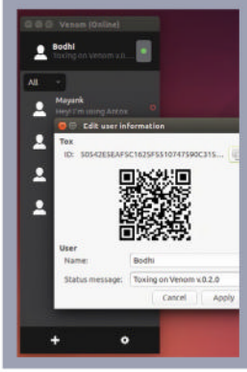
One of the most widely recognised pieces of open source software, Mozilla's *Firefox* web browser is the default browser on virtually every Linux distro. It's pretty responsive and known for its privacy features. You can customise it to the hilt and also extend it with an impressive number of extensions.
www.firefox.com

gFTP

The *gFTP* client is a feature-rich client that'll get the job done, if you need to download files via FTP occasionally. It has a simple two-pane interface that shows the content of the local and remote filesystem. Using *gFTP* you can also transfer files between two remote servers.
http://gftp.seul.org

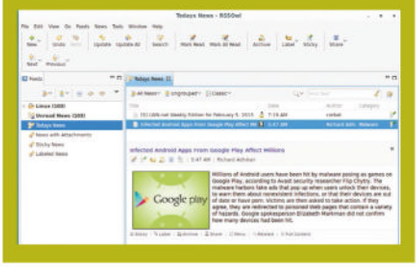
Tox

Privacy conscious users should try the new decentralised IM and VoIP client called Tox. This relies on a distributed network, which uses P2P connections, the same technology used by BitTorrent to provide a direct connection, between users for chats and, unlike other Skype alternatives, Tox uses no centralised servers or supernodes, which could be compromised. All chats are also encrypted using the peer-audited NaCl crypto library.
https://tox.im



RSSOwl

An excellent desktop alternative to Google Reader, RSSOwl is a news aggregator for RSS and Atom News feeds that's easy to configure. The app gathers, organises, updates, and stores news in an easy to use, and saves selected items for offline viewing and sharing.
www.rssowl.org



Jitsi

Jitsi is the best VoIP app, as long as you're not adverse to Java apps. It supports IM and make one-to-one audio and video calls, as well as audio conference calls. It supports many of the widely used IM and telephony protocols, including SIP, XMPP, AIM, ICQ, MSN, etc. Jitsi has all the features you'd expect from a softphone, and more, such as encrypt text chats with OTR and voice and video by establishing a ZRTP session.
https://jitsi.org

Aria2

What makes *Aria2* a unique utility is that it can download the same file at the same time using different protocols. The lightweight CLI app can download via HTTP, FTP, BitTorrent and Metalink and can also open multiple connections to download the file faster.
http://aria2.sourceforge.net

Midori

The go-to browser for anyone concerned about resource consumption, *Midori* is popular with lightweight distros. Despite its lightweight nature and design, Midori has all the features you'd expect from a web browser including a speed dial, tabbed interface, bookmark management and configurable web search as well as an incognito mode.
www.midori-browser.org

FileZilla

For those who use FTP a lot, there's FileZilla. The client supports FTP, SFTP and FTPS protocols and has just about any configuration option you can imagine. It also has a tabbed interface so you can browse more than one server and even transfer files simultaneously between multiple servers.
https://filezilla-project.org

Deluge

BitTorrent is popular for downloading Linux distros and there are numerous download clients. One of the best is *Deluge* which has multiple front-ends, including a graphical and a web-interface. It has features that enable advanced users to tweak it to their liking and also has a nice library of plugins.
www.deluge-torrent.org

Pidgin

Pidgin is a wonderful app for instant messaging over many network protocols. You can sign in with multiple accounts in the single client and chat with many friends in different networks. You can use it to connect to AIM, MSN, Google Talk, Yahoo, Jabber, IRC and more chat networks all at once.
www.pidgin.im

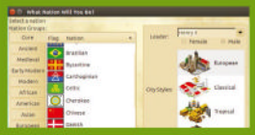
Games

0 A.D.

This is a real-time civilisation-building strategy game that features impressive graphics and intense battle gameplay. It's yet to have a final release but has already won accolades in its current state.
http://play0ad.com

FreeCiv

Another strategy game that challenges players to lead their tribe 4,000B.C to the space age.
www.freeciv.org



Alien Arena

A popular first person shooter with a sci-fi theme and the tournament style deathmatch of *Quake* and *Unreal Tournament*. The game has several game modes and over 60 maps, and is quite configurable.
http://red.planetarena.org

OpenMW

OpenMW is a new game engine that recreates the popular *Morrowind* RPG. The aim of the project isn't to improve game assets or add additional features but to provide gamers a more moddable edition of the game.
https://openmw.org

FlightGear

For fans of aircraft simulators there's *FlightGear* that aims to offer flight across real world terrain. It includes scenery for more than 20,000 airports, and can be extended with your own aircraft and locations.
www.flightgear.org

Office and productivity

Enhance your workflow with these apps.

Calligra

Unless you feel you need *LibreOffice*'s superior compatibility with proprietary formats, you may want to consider *Calligra*. It's a continuation of *KOffice* and unlike *LibreOffice*, *Calligra* has a modern-looking, modular design, and also uses Open Document as its native file format. It ships with a large clutch of apps. In addition to the *Words* word processor, *Tables* for spreadsheets, *Stage* for preparing presentations, and *Kexi* for managing databases, it also benefits from Krita for digital painting.

www.calligra.org

GnuCash

Gnome users have *GnuCash* which is similar to *KMyMoney* in terms of features, but also handles and categorises entries differently. *GnuCash* is a personal and small business accounting app that's based on double-entry for professional reporting and besides dealing with monetary transactions, it can track things such as stocks, bonds and mutual funds.

www.gnucash.org

Xournal

This app is very handy for when you need to scribble bits of information down for later. As well as typing out notes, you can use it with either a mouse or a stylus. It can also be used to add annotations to PDF files.

http://xournal.sourceforge.net

Zathura

This is a simple and a lightweight PDF reader that supports almost all the usual features you'd expect. You can search text strings, jump pages, zoom in and out, rotate pages, add bookmarks and more. In addition to PDFs, it can display DjVu and even encrypted documents.

https://pwmt.org/projects/zathura

AbiWord

The wide gap between rich text editors and word processors is occupied by *AbiWord*. It's lightweight but still offers commonly-used word processing features, which makes it a popular for lightweight distros. It also offers cloud-based collaboration capabilities via its **AbiCollab.net** service.

ProjectLibre

A project management tool helps you stay on top of ongoing projects and *ProjectLibre* is one of the best. It's an award winning app that's used widely by many enterprises around the world. *ProjectLibre* has several useful features and can also visualise tasks with various charts and reports.

www.projectlibre.org

OpenLDAP

OpenLDAP is great for when you want to run a directory server. It implements the LDAP protocol and has all the expected features, including logging, replication, access control, user and group management etc. It also integrates with Active Directory.

www.openldap.org

Gnumeric

AbiWord is usually paired with the lightweight Gnumeric spreadsheet app. However, the app isn't light on features and offers a lot more functionality than proprietary spreadsheet apps. *Gnumeric* will import data from Microsoft Excel files and there are import filters for other apps as well.

www.gnumeric.org

KMyMoney

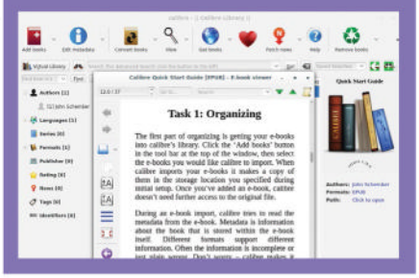
Designed for KDE users, *KMyMoney* is a feature-rich accounting app. It supports different account types, such as Cash, Checking, Savings, etc and can categorise incomes and expenses, and can reconcile bank accounts. If your bank allows it, you can have *KMyMoney* connect to your bank directly to retrieve your account activity.

https://kmymoney.org

Calibre

You can use *Calibre* to manage your collection of ebooks, and supports a wide range of readers and smartphones. The app can import ebooks manually or, if you prefer, by syncing a reading device such as the Kindle. Any files imported can be sorted and grouped by metadata fields, which can be pulled from various online sources, such as **www.goodreads.com**.

www.calibre-ebook.com



Achievo

This is a web-based resource management tool with a simple interface for accessing its CRM, HRM and project management and planning tools. You can also track resources across multiple projects.

www.achievo.org

Okular

The default PDF viewer for KDE and includes a good number of useful features. Besides PDF it can also read a number of other file types, including Postscript, DjVu, CHM, XPS, ePub, TIFF, CBR, and others.

https://okular.kde.org

LaTex

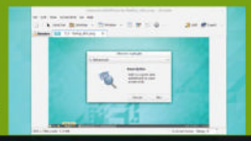
LaTex is a document preparation system and document markup language based on TeX. Its purpose is to simplify TeX typesetting for documents containing mathematical formulae and is widely used in academia.

www.latex-project.org

Shutter

Besides capturing the full screen, Shutter can capture a specific area, or a window. You can also upload to a hosting service.

www.shutter-project.org



ClamAV

While most viruses and trojans will have no effect on Linux, you still can have infected files in your distro that can wreck havoc when accessed on a Windows machine. So be a good admin and use ClamAV to scan files.

www.clamav.net

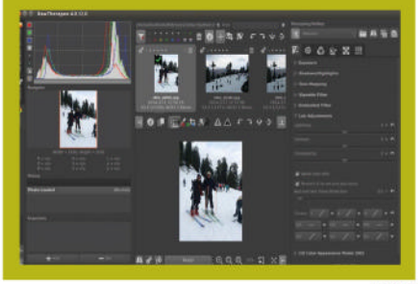
Hobbyist

Follow your passion.

RawTherapee

Do you shoot with a digital single lens reflex camera (DSLR)? Then take a look at *RawTherapee* which includes a wide range of tools for processing and converting RAW files. In addition to basic manipulations, the app has extensive options for working with RAW files. Using the app you can adjust the colour and brightness values of your images, correct white balance, adjust tones, and a lot more. Besides RAW files you can also use *RawTherapee* for editing traditional image files, and it also includes Adobe Lens Correction profiles.

www.rawtherapee.com



Scribus

A comprehensive desktop publishing program. *Scribus* can be used to create professional press-ready online and print documents including brochures, booklets, books and magazines. It has a feature-rich interface and has features, such as PostScript colour separations, support for CMYK and spot colours, ICC profiles, and printer marks. *Scribus* also includes a variety of templates and styles and you also get an array of settings and tools to precisely define and position the various layout elements you require.

www.scribus.net

Krita

Although *Krita* is part of the *Calligra* suite it needs a special mention of its own. *Krita* is a digital painting and illustration app that offers many expressive brushes, HDR painting, filters, perspective grids, painting assistants, and many other features you'd expect from such an app.

www.krita.org

Stellarium

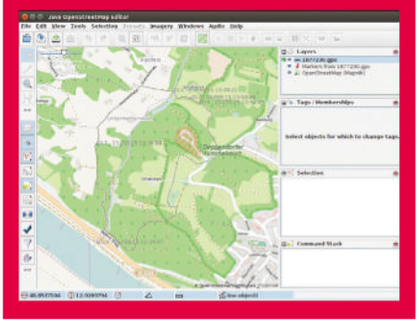
Stellarium is a free open source planetarium for your computer. It calculates the positions of the Sun and Moon, planets and stars, and draws the sky as per the users location and time. It can also draw the constellations and simulate astronomical phenomena such as meteor showers, and eclipses.

www.stellarium.org

JOSM

Keen to contribute to the mapping project, OpenStreetMap? Then use *JOSM*. It's a Java-based offline map editor that can help you plot GPS traces. You can load GPS track-logs into JOSM and start adding streets to OpenStreetMap instantly. Although OpenStreetMap has several other editors available, most contributors use JOSM for their edits, as it lets them upload changes back to OSM quickly and easily enough. JOSM offers several features and can be extended with plugins and styles.

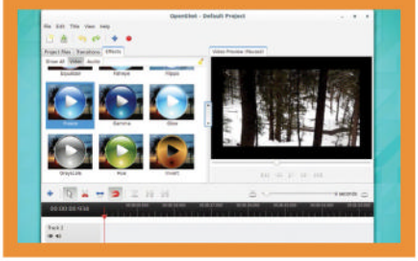
https://josm.openstreetmap.de



OpenShot

There are only a handful of video editors for Linux and *OpenShot* offers the best combination of features and ease of use for the home user. You can use it to combine videos, audio tracks, and still images together and add in captions, transitions, and more, and export the final product in a variety of formats. *Openshot* can also use *Blender* to create 3D animated titles

www.openshot.org



Inkscape

Another pro-quality tool, *Inkscape* offers advanced vector graphics editing and is popular for drawing vector art, line art, and designing logos and graphics. It's brimming with features, such as markers, clones, alpha blending and more, and is often compared to expensive proprietary apps such as *Illustrator* and *CorelDraw*.

www.inkscape.org

Cinelerra

Cinelerra is excellent if you need to edit more than home videos, as it's the most advanced non-linear video editor and compositor for Linux. It supports HiFi video and audio and is resolution and frame-rate independent, which enables it to edit videos of any size. The app has several advanced features, such as overlays, denoising, normalisation, time stretching, color balance, compositing, real time effects and a lot more. It also includes a compositing engine for performing tasks such as keying.

www.cinelerra.org

Media

Comix

Digital comics are distributed as comic book archive files that mainly consist of a series of image files, typically PNG or JPEG files, stored as a single archive file. *Comix* can read digital comics in virtually every format.

http://bit.ly/ComixApp

FontForge

FontForge is a feature-rich app for creating and editing fonts and supports all common font formats. It can extract information from a font file as well as convert from one format to another, and can be used for previews.

http://bit.ly/FontForge

CairoDock

CairoDock is a MacOS X dock-like app. One of its main advantages over other docks is that it doesn't require a compositing window manager to work and can add bling to older low-powered machines.

www.glx-dock.org

Audacity

If you need to work with audio, you should use the powerful A*udacity* sound editor. You can trim audio, combine tracks, and even stack multiple tracks, as well as export to a number of formats and quality settings.

http://bit.ly/AudacityApp

MPD

The Music Player Daemon is an audio player with a server-client architecture, which means you can control it remotely from another computer. It plays audio files, organises playlists and can maintain a music database.

www.musicpd.org

Development

Power tools and programs for power users.

jEdit

This is a text editor for programmers that supports auto indent, and syntax highlighting for more than 140 different programming languages. The app enables you to define complex macros and offers a powerful and user-friendly keyboard mapping system. It's highly configurable and customisable, and you can extend its functionality by adding plugins.
www.jedit.org

Eclipse

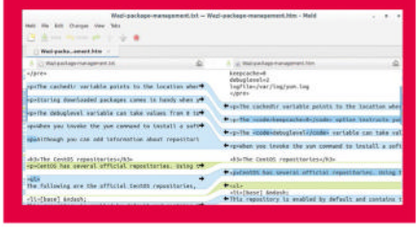
There's no beating *Eclipse*, the most feature-rich IDE. Although Java is its speciality, Eclipse supports a range of languages via plugins. In fact, its plugin marketplace is an indispensable resource. Eclipse does code refactoring and you can use it to extract the selection as a local variable or method. Since it can target multi-person installs, it handles version control very maturely
www.eclipse.org

BlueFish

Do you develop for the web? *Bluefish* is a multi-language editor that's designed for web developers. It supports many programming and markup languages and focuses on dynamic and interactive websites. It supports code block folding, unlimited undo/redo, automatic tag closing, and syntax highlighting. Another useful feature is the snippets bar from where you can add the most common snippets of code for a variety of languages. *Bluefish* also has support for popular open source web apps such as MediaWiki and Wordpress.
http://bluefish.openoffice.nl

Meld

A graphical diff tool, *Meld* enables you to compare two or three files as well as whole directories. It includes features, such as syntax highlighting and direct file editing, and using the tool you can easily isolate and merge the differences. *Meld* can also be used to browse various popular version control systems such as *CVS* and *Subversion*.
www.meldmerge.org



KompoZer

New and experienced HTML programmers will save a lot of amount of time and effort with the *KompoZer* editor. It has an intuitive interface and includes a colour picker, an FTP site manager, CSS editor, customisable toolbars, forms, spell checker, markup cleaner and can also validate code using W3C's HTML validator.
www.kompozer.net

Gimp

Despite its name, Gimp is a powerful, comprehensive image manipulation program. It offers a wide range of tools for professional-quality photo retouching and image manipulation capabilities for free. It also offers a huge list of features and supports all the common graphics file formats.
www.gimp.org

Blender

With *Blender* animators can create 3D printed models, visual effects, art, interactive 3D applications and video games. The app provides a wide range of features that can be used to create 3D animation films. It's a one-stop 3D package and includes a gaming engine, a video sequence editor, production-ready camera and object tracking, a large library of extensions, and an advanced physics engine. It can render fluid dynamics and simulate the movement of elastic objects and clothes.
www.blender.org

Geany

You don't need a full-blown IDE if you only program occasionally, which makes *Geany* a good choice. It's a cross between a plain text editor and an IDE with support for the popular languages and nifty features like a compile/run button, a listing of functions defined in the currently opened file, and much more.
www.geany.org

APTonCD

Suddenly realise that you need to move your Ubuntu installation or need to give a friend a copy of your setup? With *APTonCD* Ubuntu users can back up all of their installed packages to an ISO image, which can then be added as a software source on another installation. You can use this source to restore the packages on to the system or keep everything in the APT cache.
aptoncd.sourceforge.net

Clementine

Use Clementine to play locally stored music and streaming audio. The app has an attractive interface and it also helps organise and transfer music to various devices, and integrates well with popular cloud services.
www.clementine-player.org

Icecast

With *Icecast* you can stream music across the network. *Icecast* supports many audio streams simultaneously and listeners can access a stream via a remote media player and also configure MPD as a source.
www.icecast.org

Amarok

If you use KDE your distro may already include this music player, Amarok. It too integrates with several online audio services, and its features include creating dynamic playlists, bookmarks, scripting, context view.
https://amarok.kde.org

LMMS

LMMS is digital audio workstation that produces music by synthesising sounds, arranging samples, and playing them on a MIDI keyboard. It also has a song editor and plugins to simulate instruments and effects.
www.lmms.io

Kodi

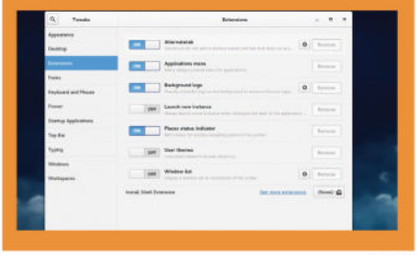
Until recently *Kodi* was known as *XBMC*. It's an excellent option for users who wish to turn their PCs into media hubs. It plays most kinds of media files and works with TVs, IR and bluetooth remote controls.
www.kodi.tv

Utilities

Apps that let you do more with your computer.

Gnome Tweak Tool

Not satisfied with the stock Gnome desktop? Use the *Gnome Tweak Tool* to customise several aspects, including the appearance settings of the desktop. With this tweak app you can also change the behaviour of the Windows and Workspaces, manage extensions and you can even circumvent the design philosophy of Gnome 3 by placing icons, files and folders on the Gnome desktop.
http://bit.ly/GnomeTweakTool



digiKam

One of the best photo management tools for Linux is *digiKam* and it has features that'll appeal to all kinds of users. It recognises all major image file formats and can organise and sort images based on metadata. The app also has plugins to export images to various online services.
www.digikam.org

K3b

Although it's designed for KDE, the *K3b* optical media burning utility is one of the finest for the job. The app can burn multiple El Torito boot images, audio CDs, VCDs, SVCDs, mixed-mode CDs, eMovix CDs, and DVDs. It can also rip DVDs and write ISO images.
www.k3b.org

Grub Customizer

Grub 2 is the most popular Linux bootloader that's used by virtually all major distributions. It's an impressive piece of software with lots of options. The *Grub Customizer* is a simple to use graphical tool,which enables you to quickly customise all aspects the bootloader, including its appearance.
www.launchpad.net/grub-customizer

DOSBox

Relive the good ol' days with *DOSBox* and play your favourite classic DOS games that won't run on your modern hardware. This is an x86 PC emulator that creates an IBM PC compatible computer complete with compatible graphics and sound cards. The app can also simulate networking hardware for multiplayer games on the local network and even over the Internet. The *Wine* project even uses code from *DOSBox* to bolster support for DOS apps.
www.dosbox.com

Avidemux

Avidemux is a video editor and converter that can be used for basic cutting, filtering and encoding tasks. It supports many file types, including AVI, MPEG, and MP4. The app is designed for users who know what they want to do but also provides an intuitive interface so that tasks such as cutting and appending videos are pretty straightforward. The app has some presets and users can also save custom settings that make the app easier for new users to operate.
http://fixounet.free.fr/avidemux

Handbrake

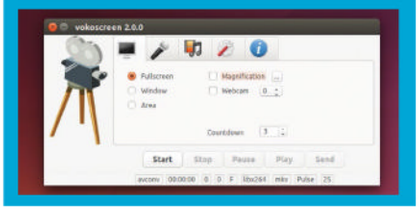
When the need to convert a video arises, *Handbrake*, the video transcoder app does a commendable job. It can convert nearly any format and supports a wide range of video codecs. One of its best features is built-in device profiles for popular devices that make the conversion process easier.
www.handbrake.fr

EasyStroke

Want to control your PC with the flick of the mouse? The *EasyStroke* app lets you define and manage gestures by recording the movements of your pointing device while holding down a specific mouse button. You can then configure actions that'll be executed when the app recognises the defined stroke.
https://easystroke.sourceforge.net

Vokoscreen

A feature-rich screencasting app worthy of note is *Vokoscreen*, which is based on FFmpeg for handling multimedia data. *Vokoscreen* can capture both video and audio, with options to record the entire screen, window or a selected region, along with video from a webcam. The app supports MPEG4, x264, MP3 and Vorbis codecs and can save files in either .AVI and .MKV containers.
The app offers some controls such as the ability to change the video quality and frames captured per second and can be used to make screencasts of games.
www.kohaupt-online.de/hp



Terminal

Ncmpcpp

This is a command-line *MPD* client that's easy to use and customisable. It provides useful features such as the ability to sort playlists, song lyrics, item filtering, fetching artist's info from last.fm, tag editor and much more.
http://bit.ly/Ncmpcpp

Samba

Samba is a suite of programs that enables Linux users to access and use files, printers and other commonly shared resources on a Windows PC on a network and does this by supporting the SMB protocol which.
www.samba.org

rTorrent

Here we have a command-line BitTorrent client with an ncurses interface. You can run it as a daemon and manage it with *screen* and since it supports SSH you can manage your torrents from any remote machine.
http://bit.ly/rTorrent

Links2

There are lightweight browsers and then there's *Links2*. This is a web browser that can render complex pages and even has a pull-down menu. It's also special because it's a CLI browser that you operate via the keyboard.
http://links.twibright.com

Midnight Commander

Before the days of graphical file managers, real hackers used *Midnight Commander*, known as *mc*. It's still your best option if you regularly find yourself in the console environmen a lot.
http://bit.ly/MidnightCdr

Admin tools

Take charge of your distro with these power apps.

Redo Backup

We've mentioned the *Clonezilla* cloning solution earlier in the feature, but if all you need is a tool to swap out an old disk for a new one, then you use *Redo Backup and Recovery*. The tool is designed for inexperienced users and has the simplest of interfaces.

www.redobackup.org

XAMPP

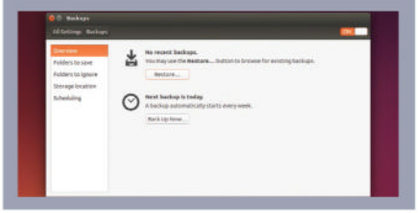
The XAMPP stack gives you a single package that you can use as a sandbox to test and develop web apps. It includes all the necessary components such as *Apache*, *MySQL*, PHP, and Perl as well as several other libraries, modules and tools, such as *phpMyAdmin* and *FileZilla* for managing the stack components. Once installed, you can manage the various services via a graphical control panel.

www.apachefriends.org

Déjà Dup

The app's minimal GUI sets itself apart from the various other backup apps you'll find, and it lets you configure backups within a matter of minutes. *Déjà Dup* is based on *Duplicity* and provides just the right number of features for desktop users who aren't used to the ways of a backup tool.

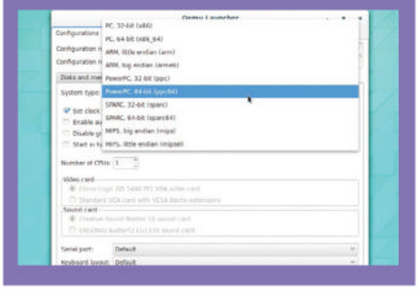
http://live.gnome.org/DejaDup



Qemu

It's is a feature-rich multi-purpose processor emulator and virtualiser. You can use it to create virtual machines and even emulate various hardware architectures. If you have the right hardware on tap (a processor with hardware virtualisation extensions), you can use *Qemu* with *KVM* in order to run virtual machines at near-native speed.

www.qemu.org



Mondo Rescue

Mondo is a unique backup solution that creates bootable backup and restoration disks customised for the system being used. *Mondo* has a text-driven interface and works with a wide range of file systems and can use a variety of media as backup mediums.

www.mondorescue.org

Open Media Vault

When you need more protection for your data than a simple backup then you need to deploy a NAS server. The *Open Media Vault* project is a Debian-based server that offers the power of commercial options in a way that's easy to setup and manage.

www.openmediavault.org

Conky

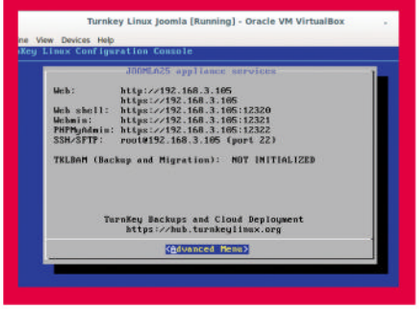
Concerned about the resource utilisation on your PC? *Conky* is a nifty little app that lets you keep an eye on your system. It can monitor and report on the states of various components. The tool is very flexible and highly configurable and can also display information from apps, such as weather updates.

http://conky.sourceforge.net

Turnkey Linux

The Turnkey project produces appliances which you can use to deploy a new server in a jiffy. A Turnkey appliance is a self-contained system that packs in a fully functional web app that runs on top of Just enough Operating System (JeOS) components required to power that particular app. All the appliances are based on Debian but are available in several formats depending on the hardware that you want to deploy it on. Once they're up and running you can manage each appliance using a browser-based interface.

www.turnkeylinux.org



Zentyal

The Zentyal distro has all the components you need to run a gateway server. The distro simplifies the process of setting up, monitoring and controlling the components of the server with a host of custom management tools and helps you configure the servers without mucking about with config files.

www.zentyal.org

Mutt

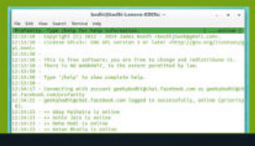
Mutt is to email what *Links2* is to the web browser. It's a text-based mail client that is highly configurable and it supports both POP and IMAP protocols and has all the usual features you'd want from an email client.

www.mutt.org

Profanity

Profanity is a console-based client for the XMPP protocol that supports multi-user chats and OTR message encryption.

www.profanity.im



Canto

Want to do more from the command-line? Get the *Canto* CLI RSS feed reader. It supports RSS, Atom and RDF feeds and imports and exports feeds in OPML format. It has lots of customisation and even configure it with Python.

http://bit.ly/CantoRSS

mpg123

This is an MP3 audio player for the command-line that supports gapless playback. It's so good that its decoding library, libmpg123 is used by other audio players for MP3 playback

www.mpg123.de

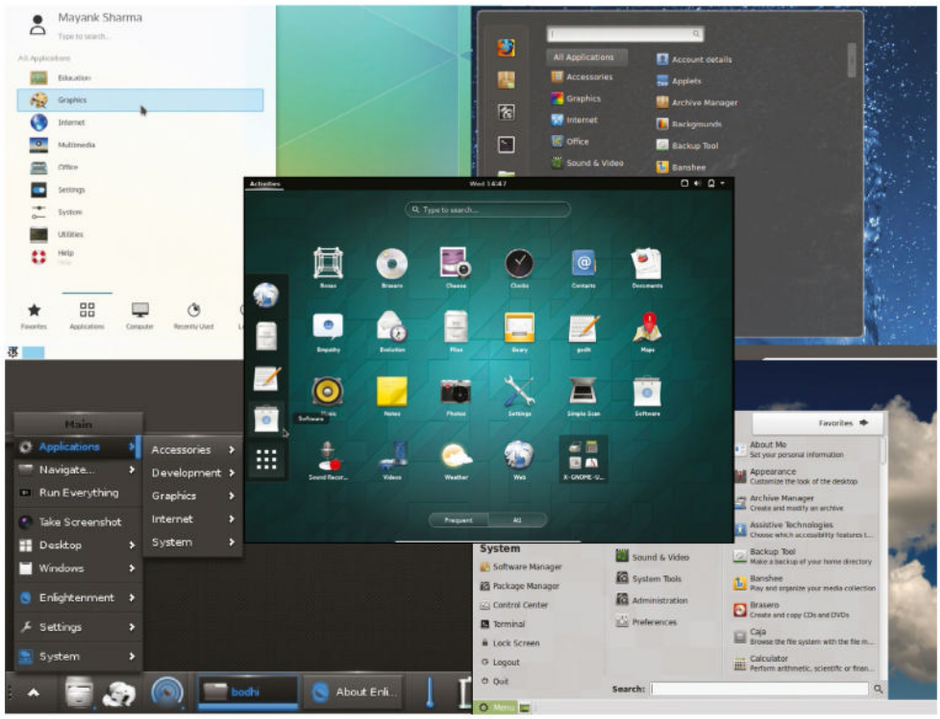
FFmeg

One of the most versatile media conversion utilities, *FFmeg* can manipulate virtually any type of media file in various ways, such as changing bitrate, extract audio, record streams, extract stream and much more.

www.ffmpeg.org

Linux desktops

Not entirely happy with your distribution's default desktop environment? Let's check out some of the mainstream alternatives.



How we tested...

Some distros rally behind a particular desktop environment by actively participating in its development. For example, Fedora, through its corporate sponsor Red Hat, has several Gnome developers on its payrolls. Similarly, many full-time KDE developers draw their paychecks from OpenSUSE.

This being the case, we felt the test needed a neutral environment, so we installed the desktops on top of the main Ubuntu distro that ships with Unity and has nothing to do with the development of any of the desktop environments. However, we also used the native environment suggested by the desktop to fully experience all the components. We'll also compared their level of configurability while commenting on their native configuration tools and any third-party or community-supported extensions.

For most desktop Linux users, a desktop environment is the paramount medium for interacting with their distribution. It's the collection of all the graphical elements that you can see on the desktop of the computer, including windows, toolbars and icons, etc. Desktop environments (DEs) also include a Window Manager that's responsible for the appearance of windows in the GUI.

Like all things Linux and open source, users are spoilt for choice when it comes to selecting a DE. The ability to change and alter the DE is just as important as being able to change and alter the default applications.

> **"The ability to alter the DE is as important as being able to alter the default applications."**

Most major distributions officially support multiple desktops. Fedora, Mageia, OpenSUSE all support KDE, Gnome and a number of other desktops. Then there are distros that officially support only a restricted number of desktops, such as Ubuntu that only supports Unity and Linux Mint that prefers Cinnamon and Mate. However, that still doesn't stop you from replacing the official default DE with another option.

In this Roundup, we'll look at some of the most popular DEs and their advantages. The one mainstream DE missing from our list is Ubuntu's Unity, which despite being open source, is best experienced on Ubuntu only.

Installation and distro support

Who supports them and how do you get 'em?

You can switch to another desktop environment without going through the pains of changing to a new distro. For all intents and purposes, a DE is just another piece of software, and you can install it as you would any other. Every desktop in this Roundup is supported by almost all distros, so they're just a visit to the package manager away.

The Ubuntu project has an officially supported flavour for the KDE, Gnome and Mate desktops.

Gnome is the default DE on many popular distros. The Fedora, Mageia and OpenSUSE projects all release an officially supported installable live CD/DVD version based around the Gnome desktop. In fact, Gnome was also default on Ubuntu until it started using its own shell. However Ubuntu does still use the core Gnome 3 libraries.

Next up on the popularity list is KDE. OpenSUSE, Mageia, and Fedora are some of the top distros that officially support the desktop and produce an installable live CD/DVD image based on KDE. If you prefer the KDE desktop on top of Ubuntu, you can grab the Kubuntu distro. Some distros such as Linux Mint include multiple bundles of the desktop with a different packages, such as **kde-standard**, and **kde-full**.

Cinnamon is the default environment that ships with the Linux Mint distro that also spearheads its development. The desktop is available in the official repositories of Fedora, Mageia and Ubuntu and you can install it via their respective package managers. Mate has carved a name for itself among distros that are designed for older computers.

Enlightenment is one of the oldest desktops in this Roundup and yet it doesn't ship by default on top of any popular mainstream distro. Its biggest promoter was Bodhi Linux which has shut up shop recently, however, you can find Enlightenment in the official repos of virtually every distro.

Once you've installed multiple desktop environments you can easily switch to another one. To do this just log out of the desktop environment. Then tinker with the buttons on the login manager and one of them will reveal a drop-down list of all the installed desktops. Select the one you want to use and the login manager will log you into that desktop.

Verdict

Cinnamon ★★★★★
Enlightenment ★★☆☆☆
Gnome ★★★★★
KDE ★★★★★
Mate ★★★★☆

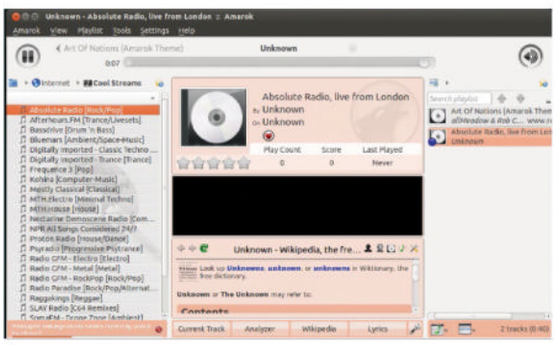
Most of the mainstream distros ship with either a KDE or Gnome desktop.

Default applications

What's shipped in the prepackaged box?

Desktop environments usually ship with their own core applications. For instance, a typical Gnome installation will have a collection of over two dozen core applications for virtually every desktop-related task, from managing images, documents, contacts to playing music and videos.

One of the highlights of Gnome 3 is the tighter integration with online services through Gnome Online Accounts. This enables you to sign into services, such as Google Docs and Flickr, and share data for offline use. Gnome 3 also has a number of apps that use these configured online accounts, such as Gnome Contacts that enables you to search for and edit your contacts, whether stored locally or online.



Unlike the old days, the major DEs have become increasingly interoperable and you can run the apps designed for one desktop on another without anomalies.

KDE also has a similarly large list of applications in its software collection. These compilations consist of packages, such as **KDE-Graphics**, **KDE-Admin** and **KDE-Utilities** and each includes related applications, such as a document viewer, an image viewer, utilities such as an archiving tool and a calculator, and various tools to aid with system administration.

The other desktops don't match up to these two and only include the most essential apps. Cinnamon uses many of Gnome 3's apps with appropriate modifications of its own, such as the *Nemo* file manager forked from Gnome's *Nautilus*. Similarly, the Mate desktop ships with a number of apps that the project has forked from Gnome 2, such as the *Caja* file manager, *Plurma* text editor and the *Eye of Mate* image viewer. Enlightenment brings up the rear. It isn't a complete desktop environment and lacks apps of its own.

Verdict

Cinnamon ★★★☆☆
Enlightenment ★☆☆☆☆
Gnome ★★★★★
KDE ★★★★★
Mate ★★★☆☆

Gnome and KDE shine, but both Mate and Cinnamon offer the essentials.

Appearance

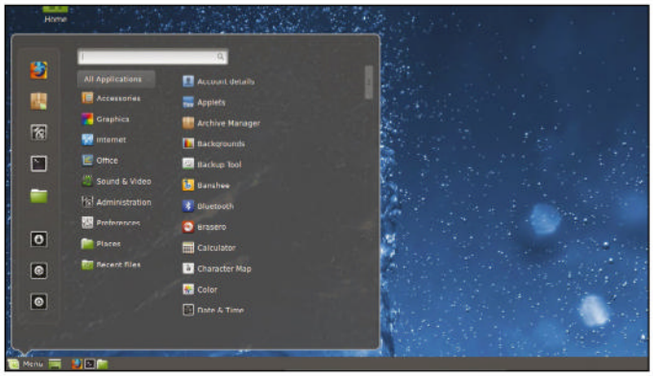
Which gives the best desktop experience?

Before Unity and Gnome 3 came about, the Ubuntu and Fedora distros dominated desktop options. But their respective new releases took DE design in a controversial new direction, re-imagining the desktop for the next gen of computing devices that didn't use the trusty mouse and keyboard combination. In accommodating new touch devices, both DEs alienated a huge swathe of desktop users, who were suddenly forced to learn new ways of interacting with their computers. Mate and Cinnamon were both born out of this controversy. But, as is usually the case with FOSS, in time both Unity and Gnome 3 have become malleable enough to win back some of their old audience. However, what sets them apart from each other is how they look, and how you operate them.

Cinnamon ★★★★★

Cinnamon is the other desktop environment that sticks to the standard desktop metaphor, and came into existence as a result of the community's dissatisfaction with Gnome 3 and Unity.

The Cinnamon desktop is a standard-fare with an icon-laden desktop and a panel at the bottom, which shows notifications alongside a list of open windows and an Applications menu in the bottom left-corner of the screen. Like Mate, the Cinnamon Applications menu is a refresh of the standard menu and extends the categorised text-driven layout of the traditional menu with usability features borrowed from other environments, such as the Favourite Apps bar. It also has widgets that you can place on your Panel as well as the desktop. There's also an Exposé-like hot corner feature that presents an overview of open windows.



Enlightenment ★★★☆☆

The Enlightenment Window Manager was born in 1997, with it's pleasantly different and refreshing view of the desktop, when the viable choices back then were Gnome or KDE.

The default desktop has a Workspace Switcher on the top and a panel at the bottom with an application launcher. You can also left-click anywhere on the desktop to bring up the launcher, and drag and drop icons on the desktop. Enlightenment gives you quite a few new mechanism for interacting with the windows; for example, there are six different options for maximising a window. The latest Enlightenment is still as graphically stimulating as ever. The desktop has a range of widgets you can add known as gadgets. Appealingly, unlike the other desktops, graphical effects on Enlightenment, such as fading menus and maximising windows, work well on older underpowered hardware.

Help and documentation

Need some hand holding?

All the desktop environment projects have adequate avenues for dispensing help and support. Gnome, for instance, has a help portal (**http://help.gnome.org**) for users and there's also **http://wiki.gnome.org** which hosts pages for the various Gnome projects. There are also several mailing lists and IRC channels.

KDE too has lots of documentation suitable for all kinds of users. There's documentation for almost every KDE app on **http://docs.kde.org** and there are also app tutorials and tips in the UserBase wiki. Similarly there's the TechBase wiki for advanced users, which has a Sysadmin's guide. Again, for support subscribe to one of the mailing lists, forums boards or visit the IRC channels.

Mate has a wiki with bit-by-bit instructions to help install the desktop on top of several distros, as well as a list of Gnome 2 apps and their matching Mate app. Users looking for help should head to the official forum boards or the IRC channel. Surprisingly there's no official documentation for Cinnamon itself, although the Linux Mint user guide has a section on it. Enlightenment has a wiki with some details about its components and you can ask for help on IRC channel or the users mailing list.

Many projects such as Gnome and KDE also help you keep in touch with their developers by aggregating their blog feeds in special Planet portals, such as **http://planetkde.org** and **http://planet.gnome.org**.

Verdict

Cinnamon
★★☆☆☆
Enlightenment
★★☆☆☆
Gnome
★★★★☆
KDE
★★★★★
Mate
★★★☆☆

» *All the DEs have some kind of support infrastructure in place for users.*

Gnome 3 ★★★☆☆

Gnome 3 has a revolutionary UI that still takes some getting used to, and we note that its apps look best when viewed full-screen and run inside windows that lack a Minimise button. The desktop begins with the Activities Overview, which gives you access to installed apps and has a launcher-like Favourites bar for pinning frequently used apps.

The Workspace Switcher is folded in the right-hand side of the screen and always lists any additional workspaces; switching to a second workspace and adding windows will automatically add a third workspace. At the top is a search box that matches strings to apps and documents stored locally or linked online services. Omissions that will jar with traditionalists are the inability to fill the desktop with icons and the lack of an options menu when you right-click on the desktop.

KDE ★★★★☆

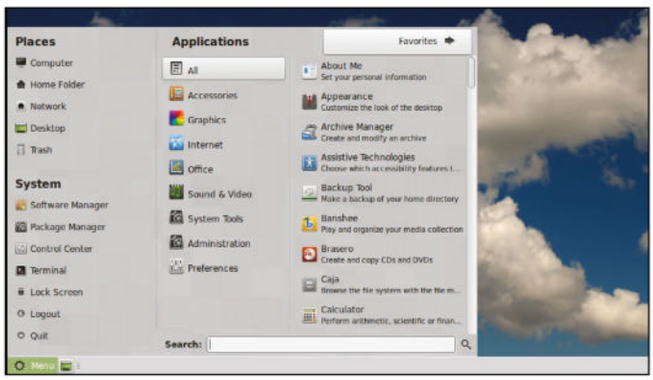
In contrast to Gnome, the KDE desktop is a malleable shell that's all about customisation. The default layout and behaviour of the desktop and the Kickoff app launcher will be feel familiar even to users from non-Linux operating systems.

KDE ships with different Views designed to make the best of the available desktop real-estate for regular screens and netbooks, and doesn't force the user to stick to either. The default layout is the Desktop View, which encourages you to use desktop widgets and you can also add widgets to the panels. Most distros place the Folder View widget on the desktop to display the contents of a folder in a neat little box that you can place anywhere on your screen. The new release features flatter icons and its Activities feature is now more accessible and configurable.

Mate ★★★★★

The Mate desktop is a fork of the Gnome 2 desktop and mimics its look to the letter. Linux Mint's Mate looks different and ships with a panel at the bottom and an application launcher on the left. Clicking on the launcher reveals a three-pane application menu. In the first-pane you get quick links to Places and System tools including the Package Manager, Control Center etc along with the power off options. In the second-pane you get the typical software categories, such as Accessories and Office. You click on any one of these to reveal its contents in the third-pane.

Many of the items in these panels are just controlled via plugins that you can easily turn off from the Panel preferences. You can create icons and shortcuts on the desktop and place files and folders. You can also add a panel on top and add applets to it as you could in Gnome 2.



Performance

How do they impact boot times?

An important criterion for selecting a DE is the age of the hardware that will power it. Newer desktops need accelerated graphics and oodles of RAM. On such a system, you should stick to a DE designed for the regular user, such as Gnome 3, KDE or Cinnamon. On older hardware, Mate and Enlightenment will give you a smoother experience.

Gnome boots up slow on older machines and takes a lot of resources. KDE on the other hand will be more responsive on the same machine and is more resource efficient with every release. However, for a solid experience, you should only use them on a machine with at least 2GB of RAM.

The key difference between Cinnamon and Mate is that the former takes advantage of modern hardware to provide slick graphics while Mate runs more efficiently on older hardware. Mate is often pitched as the desktop for users that crave the productivity of Cinnamon, but lack the resources for it.

On our test machine, Mate booted almost twice as fast as Gnome and KDE. It also managed to shave off about five seconds on average compared to Cinnamon, while having almost an equal memory footprint. Unlike many lightweight DEs, the Enlightenment desktop is full of eye candy that you'd expect from a full-blown DE, but at a fraction of the resources. In fact, in our tests the SparkyLinux Enlightenment edition booted about seven seconds faster than the distro's Mate edition.

Verdict

Cinnamon ★★★★☆
Enlightenment ★★★★★
Gnome ★★★☆☆
KDE ★★★★☆
Mate ★★★★★

» *Gnome still needs a fairly modern system to flex its muscles.*

Extensions and add-ons

I need add-ons, lots of add-ons

Extensions have played a very important role in the acceptance of Gnome 3 and the project has quite a few of them and an innovative way of installing and managing them. The Gnome Extensions website lists a variety of add-ons and extensions that add missing functionality. You can install them from the website itself in just a few clicks. Some of the popular ones enable features that were mainstays of the Gnome 2 desktop and will help ease the transition for users moving to Gnome from the likes of Windows.

One of the remarkable features of KDE 4 is its extensibility. The desktop enables you to even replace the standard Kickoff app launcher with the classic application launcher, or the modern Lancelot launcher. Add-on widgets are known as Plasmoids in KDE parlance and you can find dozens of them for everything from displaying RSS feeds to automatically uploading images to an image sharing website.

Cinnamon ships with an Extensions module in its Control Panel. This doesn't house any extensions by default, so you'll first have to switch to the Available Extensions (online) tab to download them. The tab lists almost two dozen extensions including several Alt+Tab app switching mechanisms, such as the Coverflow App Switcher, and the 3D App Switcher. Other popular extensions include the Desktop Scroller and Wobbly Windows.

Mate doesn't have any official extensions, but the community has contributed some to extend the functionality of some of the core components. For example, the Caja-actions extension, which adds apps to the context-menu and the sound-converter extension which enables you to convert audio files to different formats. There are also a set of plugins for the *Pluma* text editor and the *Eye of Mate* image viewer, as well as some panel applets.

Enlightenment has a range of widgets, which it calls gadgets, that you can place on the Shelf (what Enlightenment calls the toolbar) or add to the desktop.

Verdict

Cinnamon ★★★★☆
Enlightenment ★★☆☆☆
Gnome ★★★★★
KDE ★★★★★
Mate ★★★☆☆

How things change – Gnome 3's many add-ons make it very flexible.

Configurability

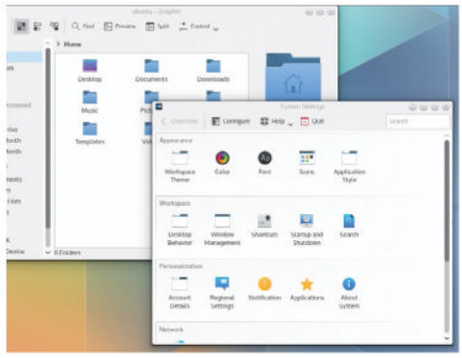
Are they tweakable?

Gnome has basic configuration options including changing wallpapers, configuring online accounts, and basic privacy preferences. For more extensive tweaks you need the third-party *Gnome Tweak Tool* that's available in the official repos of most Gnome-based distros, such as Fedora. With the tool, you can tweak the appearance of the desktop, display icons on the desktop, tweak the top panel and change the behaviour of the windows and workspaces etc.

There's literally no end to KDE's customisation options. Customising KDE is an on-going process and not a one-time affair. The desktop is designed to grow and mutate as your usage requirements. KDE has a well-laid out System Settings module. Settings are housed under the top two categories of the panel. Using the Application Appearance option you can pick the theme for the widgets and also influence individual elements, such as its colour, icons and fonts. Also take a look at the Desktop Effects option, which, as the name suggests, enables you to activate and configure the compositing effects.

Both Cinnamon and Mate include many customisation options in their respective Control Centers. Under Mate the Control Panel lets you influence the styling of the desktop as well as setup hardware and configure system tools. Both desktops include the Appearances module which houses settings for changing the look of the desktop, and Cinnamon also has the Effects module, where you can enable or disable many compositing effects. The bulk of the settings reside under the Preferences section. Using the Panel module, for example, you can tweak the panel's layout, move it to another corner of the screen or auto-hide it to maximise desktop space on smaller devices.

Enlightenment also lets you customise every detail of how it looks, feels and behaves. Its configuration panel is overflowing with options.



Some KDE distros, such as OpenSUSE also include their own configuration wizard.

Verdict

Cinnamon ★★★★★
Enlightenment ★★★★☆
Gnome ★★☆☆☆
KDE ★★★★★
Mate ★★★★☆

The other DEs score higher than Gnome 3 as their customisations are built in.

Desktop environments

The verdict

A desktop is about personal preference. It wasn't like everyone ditched Gnome 3 when it debuted its new fangled version. Nor did everyone embrace Cinnamon or Mate with glee. Each desktop environment is designed with a particular purpose and suits a particular type of Linux user.

Enlightenment is the esoteric desktop of the lot. All that beauty and bling comes at the cost of usability. A better lightweight distro, minus the bling, is Mate. However, it isn't of much use as an everyday desktop without third-party apps.

If you want your desktop to be usable straight out the box, you can't beat Gnome and its default set of apps. But it saps usability with its eccentric layout and to be productive with the desktop you need to rely on a third-party customisation tool. If you have a particular style of working, and want to take charge of the layout and behaviour of your desktop, then nothing is likely to suit you better than KDE. The desktop is so malleable that you can even tweak it to resemble Gnome 3. This is why distros, such as OpenSUSE, Mageia, ROSA and Chakra all look slightly different from each other despite all shipping the KDE desktop. However, KDE is one of the least friendly desktops for new users and all its configuration options might actually be a turn off.

We're awarding this Roundup to Cinnamon, then. The desktop environment is one of the reasons behind Linux Mint's success. The distro was willing to go the extra mile to please users who were turned off by the new Gnome and Unity desktops. While Cinnamon lacks the customisation of KDE, it does offer adequate options. It's also intuitive to use which is why it's offered as an option by a number of leading distributions, such as Mageia, Fedora and OpenSUSE. In fact, with a few exceptions, major distros officially support multiple desktops. OpenSUSE, Fedora, Mageia support KDE, Gnome, Cinnamon and a number of other desktops, so you really should try a couple to see which better suits you.

Some distributions rally behind a particular desktop and offer a much better experience on that.

> **"Cinnamon: the desktop environment is one of the reasons behind Linux Mint's success."**

1st Cinnamon ★★★★★

Web: http://cinnamon.linuxmint.com **Licence:** GPL v2 **Version:** 2.2

» The desktop successfully bridges the old with the new.

2nd KDE ★★★★☆

Web: www.kde.org **Licence:** GNU LGPL **Version:** 5.1

» The desktop of choice for tinkerers.

3rd Gnome ★★★☆☆

Web: www.gnome.org **Licence:** GPL, LGPL **Version:** 3.14

» It's bold and its different and still takes some getting used to.

4th Mate ★★★☆☆

Web: www.mate-desktop.org **Licence:** GPL, LGPL **Version:** 1.8

» For those who like to maintain status-quo.

5th Enlightenment ★★☆☆☆

Web: www.enlightenment.org **Licence:** BSD **Version:** e19

» Ideal for adding bling to old PCs that can't power mainstream desktops.

Over to you...

Do you agree? Or do you use a desktop environment that we've overlooked? Tell Linux Format at **lxf.letters@futurenet.com**.

Also consider...

There's no dearth of desktop environments that you can install on top of your favourite Linux distributions. There's Unity which isn't really supported outside of the Canonical-backed Ubuntu.

Before Mate came along, people looking for a lightweight alternative to the mainstream desktops went with either Xfce or LXDE, and when Gnome 3 came out many people went to Xfce, because of that desktop's similarity to Gnome 2. Then there's LXDE which is designed for low resource usage and has much simpler tools than even Xfce. However, both LXDE and Xfce have officially supported flavours of the Ubuntu distribution, which are called Lubuntu and Xubuntu, respectively.

If you need an even faster desktop, there's the ROX Desktop. It's based on the *ROX Filer* file manager and was inspired by the user interface of RISC OS. Some distros also use the *Openbox* stacking window manager. If you want something even more esoteric, then there's *JWM* which is used by Puppy Linux and works admirably well on older hardware.

Build your own Steam Machine

Explosive AAA gaming has arrived on Linux with over 1,000 titles available. Building your own dedicated gaming box has never been easier.

Gaming on Linux has been plagued with problems in the past, usually because many developers rush out Linux support – or leave it out altogether – and focus on Windows. The hard truth is Microsoft's OS is found on the vast majority of gaming PCs (eg Steam's Hardware Survey (Feb 2015) has the Windows user base at 95.68% out of over 125 million active clients), and that's even with the company leaving a trail of broken promises and an even more broken online infrastructure and DRM – try mentioning Games for Windows Live to a PC gamer and see them visibly shudder.

Thankfully, the tide has turned and gaming on Linux is in rude health. Microsoft's desire to create a walled garden with Windows 8 worried Valve, the video game developer behind the much-loved *Half-Life* series, and the company behind the Steam service, of course, enough to create a Debian-based distro called SteamOS that is squarely focused on gaming.

> **"It's now easier than ever to game on Linux – we get access to the latest titles."**

Although Valve's embrace of Linux left a lot of us wondering what took them so long, it was high profile enough to grab the attention of PC gamers who hadn't considered Linux before. With Valve's backing, an increasing number of developers are porting their games to Linux, while hardware manufacturers, particularly graphics vendors, are making decent strides in supporting Linux through their drivers.

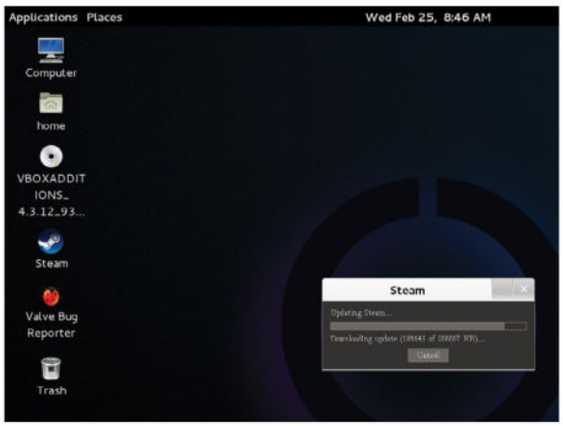
It's now easier than ever to game on Linux – we get access to the latest titles, powerful hardware is supported and we don't have to struggle getting games working via *Wine* or waste money on a Windows licence. Even better, many PC gamers can even see an impressive improvement in performance just by switching to Linux.

You could, of course, buy a Steam Machine from many reputable manufacturers now (such as Alienware, Asus, Cyberpower, Scan and Zotac etc), but to get yourself a dedicated machine for playing Linux games, we think your best bet is to download and install SteamOS yourself. This distro has been designed from the ground up for gaming, with Steam's Big Picture Mode as the default interface. The interface has been specially built for navigating with a control pad on a big screen, such as a TV, though this means if you want to use your machine for tasks other than gaming then SteamOS isn't for you in it's current form.

However, if you want to make the ultimate Linux gaming machine that blows the PS4 and Xbox One consoles out of the water, then head over to **http://bit.ly/BYOSteamOS**.

On this page you'll find two options, the first is to download the default SteamOS beta installation. Although this is probably the most straightforward way of installing SteamOS, it does require a hard drive with a whopping 1TB capacity, which is probably a lot more than what most people have – or even need. The second option is to download the custom SteamOS beta installation. This method gives you more control over the install, using an installer that's based on Debian's default, and it means you can install SteamOS on to a more realistically sized hard drive. Clicking Download, the default SteamOS beta installation takes you to a page which displays the Steam End User Licence Agreement. It's worth reading this to understand what SteamOS and Valve's Steam service is.

Although SteamOS is Linux-based and uses open source software, it's primarily an interface for Valve's proprietary Steam Store. Proprietary drivers are also used, and although Steam is less obnoxious than some DRM-infused store fronts, you should know what you're getting into before you install it. You will, for instance, be able to access the Gnome desktop that's installed as part of SteamOS to install non-Steam programs and games at least.



You can enable a Gnome desktop in the SteamOS settings, which will allow you to run non-Steam programs and games.

> **"The UI has been specially built for navigating with a control pad on a big screen."**

Another thing to consider is that the SteamOS is a 1GB download, so if your internet connection isn't the fastest, it's best to have a cup of tea (or four) while you wait. If you agree to the terms of use click the Download SteamOS Beta button to begin.

Once downloaded you'll need to extract the contents of the **SteamOSinstaller.zip** file onto a USB stick. The USB stick needs to have a capacity of 4GB or higher and will need to be formatted to the FAT32 filesystem.

To format the USB drive to FAT32, insert it and bring up the terminal. Next, type in **df** to bring up a list of the drives installed in your machine. Look carefully at the list to identify your USB stick (capacity is a good indicator). It's almost goes without saying, but what the hell we'll say it anyway, but it's vital you correctly identify your drive before you format it, as going ahead and formatting the wrong one can be devastating.

Once you've identified your USB drive make a note of its path under where it says Filesystem, for example **/dev/sdc1**. You'll need to unmount the drive by using:

```
sudo umount /dev/sdc1
```

where **sdc1** is put the path of your USB drive. Next format the drive with the FAT32 filesystem with:

```
sudo mkfs.vfat -n 'SteamOS' -I /dev/sdc1
```

Once again, where we've written **dev/sdc1**, make sure you put the correct path. Also the **-n 'SteamOS'** part of the code is optional. This just names the partition you've created on the drive for ease of use. If you'd rather not name the partition, feel free to leave this out.

Hopefully, by this point the SteamOS file will have downloaded as a ZIP file. We'll need to unzip the files to the freshly-formatted USB drive. To do this, you'll first need to make sure you have the correct programs installed. As root user type in:

```
apt-get install zip
apt-get install unzip
```

»

The hard stuff for your Steam Machine

When building a machine to play games in the living room you need to consider a few things. For starters, since this is for the living room you'll want it to look good, not take up too much space and run quietly.

For a great looking, yet small PC case we'd suggest going for the BitFenix Phenom Mini-ITX, which can be bought for around £60. Next you'll want a CPU, and although Intel's processors are more expensive than AMD's, they perform better, and will future-proof your Steam machine.

The quad-core Core i5-4570 is a great choice that runs at 3.2GHz and costs around £150. Choosing a case and a CPU narrows down our motherboard options. We've gone for the MSI Z87I AC, which costs around £50, as it's a Mini-ITX board and compatible with our processor. Even better, the board comes with built-in Wi-Fi so you don't have to trail Ethernet cables through your living room.

Next up you'll want to think about a graphics card. For ease of compatibility we've gone with Nvidia. Some micro-ITX cases have limited space for GPUs, so we've gone for the Asus GeForce GTX 970 DirectCU Mini. This is an excellent and tiny card that will run the latest games with ease. It is, however, a bit pricey at £280, but well worth the money. If you want to save some cash then the slightly older Asus Nvidia GeForce GTX 760 2GB GDDR5 DirectCU II Mini is a great choice and costs a more palatable £187.

You'll also want a cooler (such as the Gelid SlimHero for £25), memory (Crucial Ballistix Tactical LP DDR3, 8GB for £70 is a good shout), a power supply unit (GX Lite 500W PSU for £41) and a hard drive (any old one will do, we'd recommend 500GB if you're thinking of having lots of games). Hey presto, you've now got an amazing Steam Machine that blows the PS4 and Xbox One out of the water.

All these lovely components will build a formidable gaming machine.

» Now navigate to the folder where the **SteamOSInstaller.zip** was downloaded (usually Downloads), for example:

```
cd ~/Downloads/
```

then type in

```
unzip SteamOSInstaller.zip -d /path/
```

where **/path/** is enter the path of your USB drive. Next, you'll need to install the USB stick into the machine that you're using for the installation. With the USB stick installed, start up the PC and load up the BIOS. This can usually be done by repeatedly tapping F8, F11, or F12 as soon as your system is turned on. Once in your BIOS make sure that UEFI support is enabled and select the UEFI entry to boot from.

If you don't mind having the entire hard drive formatted and replaced with SteamOS, select the Automated install option from the menu when it appears. If you have additional disks and partitions that you want to keep, and you want to install SteamOS in a select location choose Expert install.

If you've ever used the Debian installer you'll be pretty familiar with what comes next. First, you'll be asked to choose your language, location and keyboard layout. The installer will then begin setting up your hardware which will usually take a few minutes. Once done you'll see your hard drives and partitions. This is where you can decide which partitions and drives to use to install SteamOS – useful if you don't want to use all of your hard drive or if you're planning on going the dual-booting route with SteamOS for gaming and another distro for day-to-day tasks.

Select the free space for installing SteamOS – it should be a minimum of 10GB. Select Create a New Partition if you need to

Peripherals

So you've built an amazing, yet compact, Steam Machine and loaded up SteamOS. Now what? You'll want to get some great gaming peripherals for comfy gaming from your sofa.

Valve itself has been working on a dedicated Steam controller with the lofty ambition that it will combine the convenience of a game controller with the precision of a keyboard and mouse setup. It's certainly a tall order and one that Valve appears to have struggled with as the controller has been delayed until late 2015. While we wait for Valve's official controller, which will cost $50, a number of other companies offer some great alternatives for controlling SteamOS games. Roccat has built a Sova lapboard especially for SteamOS which offers a small mechanical keyboard and large mouse pad that can rest on your lap. You can also use games controllers from game consoles, such as the Xbox 360 and PS4 as SteamOS does a good job of recognising them as soon as you plug them in. If you're a fan of racing games then the good news is that renowned racers, such as *Project Cars* are coming to Linux. What's not so great is the support for steering wheel controllers.

If you have a Logitech controller you can install the LTWheelConf tool. Full instructions on how to use it can be found on the Steam network (**http://bit.ly/LTWHeelConf**).

The Roccat Sova has been built especially for SteamOS devices.

The 20 best games on Linux

Five best open source games

Strife: Veteran Edition
This is an awesome first person shooter built on the open-source Chocolate Doom engine. Grab the game from **http://bit.ly/StrifeVE**.

Stunt Rally - version 2.5
Race and performing stunts in fantastic environments. This game features 167 tracks, 19 cars and a track editor. Download the game at **http://bit.ly/StuntRally**.

Annex: Conquer the World 4.0
If you enjoy real time strategy games, then this open source game is for you. Download the game from **http://annexconquer.com**.

BYOND: Space Station 13 Remake
This remake of a criminally overlooked classic is completely open source. Download the code from **http://bit.ly/SS13Remake**.

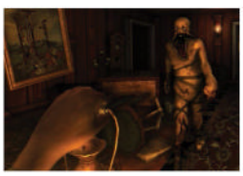
Galaxy Forces: Moon Lander Action!
Hark back to a simpler time for games with this retro-fuelled moon lander shoot-em-up. Download from **http://bit.ly/GalForcesV2**.

Five best AAA games

The Witcher 2: Assassins of Kings
An epic tale of monster-slaying and alchemy, *The Witcher 3* is coming soon, but play this first.

Dying Light
An action survival game presented in first-person. Navigate a dangerous zombie-filled open world to help survivors.

Borderlands 2
This fun and frantic first person shooter makes a post apocalypse world seem like a lot of fun. Play in co-op mode with friends.



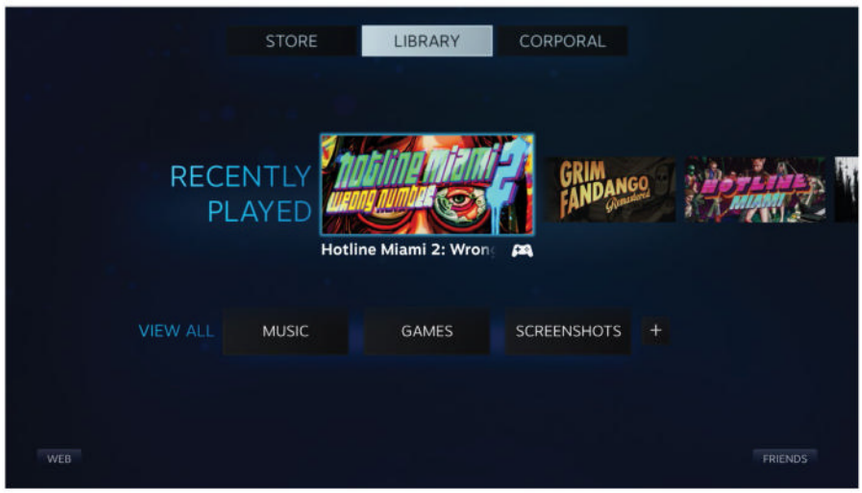
Amnesia: The Dark Descent
Games don't come much scarier than this, so if you're after a good horror game then you'll love this.

Broken Age
The first graphic adventure by Tim Schafer in sixteen years, funded by a record-breaking Kickstarter – and well worth the wait.

and specify the size. Ensure it's Primary, then click Continue, making sure in the Use as Area it has Ext4 Journaling Filesystem, then select Done setting up the partition.

Click on the free space to create another partition. Give it a size of around 10GB – this will be your swap partition. Make it logical, and create it at the end of the available space when the options appear. When you get to the summary screen, double-click Use as and select Swap Area. Double-click the remaining space, leave the partition size as it is and make sure where it says Mount Point you have it set to **/home**. Select Finish Partitioning and Write Changes to Disk, then select Yes. SteamOS will begin configuring and installing itself. Once done a window will appear called Software Selection asking you if you want to install the Debian desktop environment and standard system utilities. Keep both of these ticked and click Continue. Once done your PC will reboot.

Once your system has rebooted, you'll be given the choice to start SteamOS or start it in Recovery Mode – leave it to start normally and SteamOS will continue the installation. Make sure your machine is connected to the internet, as Steam will be installed. Once that's done your machine will reboot once again. The process might create the rescue partition now, so let it do its thing and select to reboot. You'll then be presented with a Debian login screen. Select SteamOS Desktop and click Return to Steam.

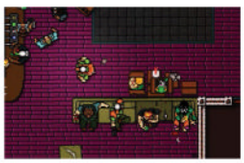


Big Picture Mode makes launching games on a TV with a games controller quick and easy.

If this doesn't work, open up the Terminal and type **steam**. Accept the terms and click OK. There may be some more downloading to be done, but once that's done you'll be thrown into Steam's Big Picture Mode where you'll be able to log in to your existing Steam account, or create a new one.

Alternatively, If you don't want to install a new OS for Steam you could install the Steam for Linux client instead on any Debian-based distro by typing in **apt-get install steam** or **aptitude install steam**. You're now ready enjoy over 1,000 (and counting) titles.

Ten best indie games

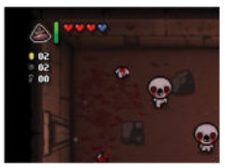


Hotline Miami 2: Wrong Number
The sequel to the ultra-violent and maddeningly addictive indie sensation comes with the same thrills and amazing soundtrack, but it's not for the faint hearted or kids.

Supreme League of Patriots
A classic point and click adventure game with very modern sense of humour brings a cast of crazy characters and fiendish puzzles and combines it with a great art style.

The Fall
The first story in a trilogy, this adventure game excels at world building, story and atmosphere. You play ARID, an artificial intelligence onboard a high-tech combat suit occupied by an unconscious pilot.

Dungeons 2
If you're a fan of Bullfrog's classic game *Dungeon Keeper* then you'll love this spiritual successor, which not only allows you to build devious dungeons to thwart pesky heroes but sees you go topside to attack cities.

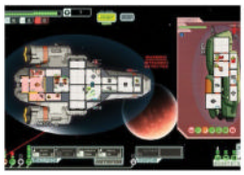


The Binding of Isaac: Rebirth
This is a randomly generated action RPG shooter with *Rogue*-like elements. If you don't understand what we just said, all you need to know is that it's a lot of fun.

This War of Mine
A game like no other. You don't play as an all-powerful soldier, but instead a group of civilians just trying to survive in a besieged city.

Chivalry: Medieval Warfare
Besiege castles and raid villages in this fast-paced medieval first person slasher with a focus on PvP.

Papers, Please
Play the role of immigration inspector for a fictional country. Bureaucracy might not seem thrilling, but this manages to make it so.



FTL: Faster Than Light
Take your ship and crew on an adventure through a randomly generated galaxy filled with glory and bitter defeat in this spaceship sim.

Goat Simulator
Ever wanted to play as a goat? This offers you a chance to live the dream in this completely realistic (not really) simulation.


Guru Guide
Nobody goes deeper into technology
CODING ACADEMY 2015*
FULLY REVISED & UPDATED EDITION
LEARN TO CODE FAST TODAY!
• PYTHON • RUBY ON RAILS
• PERL • PHP
180 PAGES OF TUTORIALS
MASTER NEW SKILLS YOU CAN APPLY TO ANY PROJECT TODAY!
Available at all good newsagents or visit
www.myfavouritemagazines.co.uk/computer

Remote desktops

What's the best client for getting full desktop access from afar?

How we tested...

For testing we used a number of machines: a high-end gaming PC, a Raspberry Pi 2 Model B (where software existed), and we dusted off a slightly sluggish 2.33GHz dual-core machine for some perspective. To see how the candidates fared in low-bandwidth scenarios, we ran the clients through the *Trickle* bandwidth shaper. We simulated slow (25kb/s) and very slow (6kb/s) connection speeds this way. In order to allow each client to reach its full potential, we first paired each client with its partner server to gauge maximum performance. We even used a Windows 7 PC to test the mettle of the RDP clients against the original protocol (the open source xrdp server only implements the protocol parts in the public domain). In the compatibility category we mixed this up to see how different clients and servers interoperate.

While everyone knows the best way to do remote access is SSH, sometimes it's nice (and even necessary) to have access to an entire desktop. Maybe you need to show Auntie Ethel how to change her desktop background, or how to get *nmap* to make a diagram of a rival knitting circle's network.

This surfeit of graphics data presents a problem, especially for the bandwidth-challenged, which a number of technologies aim to solve. Linux favours the VNC protocol, while Windows favours the largely-closed source Remote Desktop Protocol (RDP). There's nothing OS-specific about either of these though. They both work directly on the framebuffer, so the underlying technology works equally well on Windows or Linux. The NX protocol used in *NoMachine NX* challenges both of these with advanced compression and latency reducing tricks which in Linux work on the X protocol directly (or the RDP protocol in Windows). Since 2010 though, the client has been closed source and while once a number of projects aimed to provide open source NX solutions, development of these has largely fallen by the wayside, with the exception being *X2Go*. The *Chrome Remote Desktop* app is still in beta, but will already be of interest to some.

> **"Maybe you need to show Auntie Ethel how to make a diagram of a rival knitting circle's network."**

Get the UK's best-selling Linux magazine

DELIVERED DIRECT TO YOUR DOOR

Order online at **www.myfavouritemagazines.co.uk**

or find us in your nearest supermarket, newsagent or bookstore!

Ease of use

Is it easy to install and navigate?

Finding a distribution for which a *Remmina* package doesn't exist is unlikely as it's rather popular. To get VNC functionality in *Remmina* requires *libvncserver* to be installed, but most distros will sort this out for you. On Arch Linux this package was listed as an optional dependency and needed to be installed manually. Despite the plethora of options everything in *Remmina* is laid out intuitively, so a straightforward connection is straightforward to set up.

TigerVNC, on the other hand, can be rather tricky to locate packages for. Many distros, including Debian and Ubuntu, have opted for the older, and differing by two letters, *TightVNC*. Once you've managed to find some packages though, it's easy enough to find your way around the client. It's modelled after the 'original' *RealVNC* client and as such isn't much to look at. The default options will generally just work so connecting is an easy matter of typing a hostname into the address bar, although you might need to add a **:1** to the end.

Packages for *x2go* are available for most distros, including Raspbian. After installing the server you may need to run **x2godbadmin --createdb** before you can connect. Some packagers seem to have been lazy here. The Qt4 client is easy enough to navigate, but could possibly be laid out in a tidier fashion. It provides reasonable session management through a list on the right-hand side.

Installing the browser part of *Chrome Remote Desktop* app is, as you may imagine, very straightforward. However, setting up a remote server (on Linux) involves installing a Deb package, which by all accounts doesn't work out of the box. On Mint/Ubuntu files needed to be moved or symlinked before *Chrome* would present the option to enable remote connections.

This is a remote desktop clients Roundup, but we're going to go ahead and penalise the app regardless, because this kind of suffering is implicit in its use. Once everything's set up, though, you do get an easy-to-comprehend list of computers and remote assistance requests. *NoMachine* will need to be installed manually, but they do have Deb and RPM packages for you, as well as an installer bundle if these are unsuitable. You will be greeted by a double-whammy of welcome messages, which may help you get your bearings, though the interface is straightforward (if a little garish). Servers can (optionally) advertise themselves on the network so that they are visible to all clients.

This is the second of four wizard-style screens that welcome you to *NoMachine*-ville.

Verdict

Chrome Remote Desktop ★★☆☆☆
NoMachine NX ★★★★☆
Remmina ★★★★☆
TigerVNC ★★★☆☆
X2Go ★★☆☆☆

Remmina and NoMachine are the friendliest of the batch.

Documentation

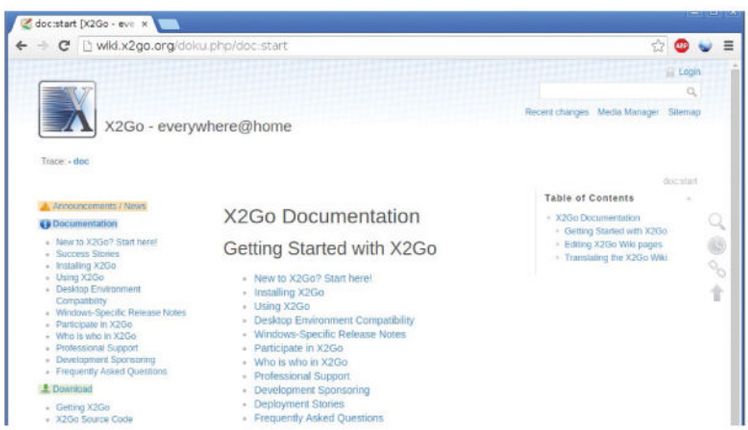
Someone said to read the manual. Is there even a manual?

Remmina is fairly self explanatory to use and has been translated into several languages. If you're feeling brave you can delve deeper into the workings of the **xfreerdp** command that it uses for RDP sessions. Despite its appearance, *TigerVNC* has excellent man pages. They will mostly be of interest to anyone wishing to tweak the server side of things, but the client has command line options for everything in the menus too.

The *X2Go* server comes with a number of command line utilities which could be useful for scripting purposes. They are all thoroughly documented in the provided man pages. The website has plenty of useful information too, not just on how the program works, but concerning future ideas for the project.

There's also quite a handy guide on which desktop environments may fail together with an explanation of why they do. Some remedies are offered for simple cases, such as *IceWM* and *OpenBox*, and the bad news is all laid out clearly for anyone wanting to use a modern desktop.

The *Chrome Remote Desktop* app really needs to provide better documentation for setting up the service. Granted it's still in beta but this is a fundamental issue. Better yet, why not just provide some working packages? The app itself is straightforward enough that Auntie Ethel could use it. *NoMachine*'s documentation is more than adequate, but it loses points for giving you four annoying instruction screens before letting you initially connect.



***X2Go*'s website will help you get started and their mascot will charm you.**

Verdict

Chrome Remote Desktop ★★☆☆☆
NoMachine NX ★★★☆☆
Remmina ★★★☆☆
TigerVNC ★★★☆☆
X2Go ★★★★☆

The outsider of the pack, X2Go scores a surprise victory.

Features

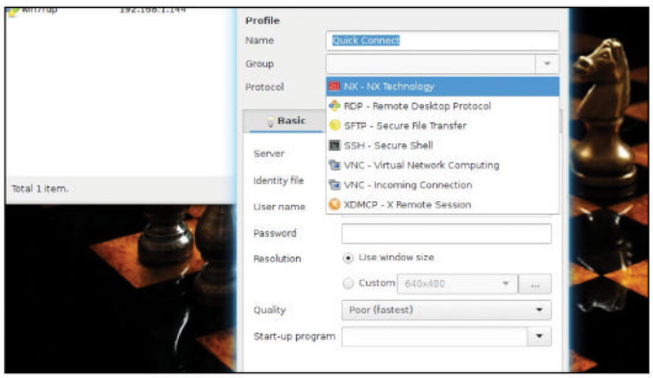
Who has the best bells and whom hath the finest whistles?

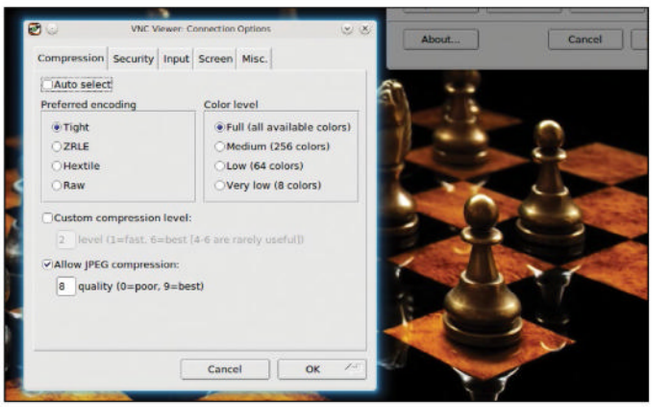
All of the clients on test will, modulo the appropriate configuration and let you connect to your desktop from afar. But they all cater to different needs, have different emphases and do different tricks. In this category we see what features each candidate offers, whether those features are useful or whether they even leave the user in a state of confusion.

All of the clients on test support fullscreen mode, so that (bandwidth permitting) you can pretend you're sitting in front of the remote machine. Some things we mention are strictly properties of the client, whereas some are dependent on the client connecting to the right kind of server. This is particularly true of *TigerVNC*, so much of our praise and criticism here is directed at the server side of things.

Remmina ★★★★☆

Remmina is an impressive client. Besides NX, VNC and RDP, it supports the XDMCP protocol underlying communication between the display server and the display manager. It even enables graphic free connections via SSH and SFTP. If that isn't enough for you, it can be extended through a plugin system. It supports quality presets, which you'll want to modify depending on available bandwidth, as well as scaling options (in case you're viewing on a low powered device). The VNC protocol is unencrypted so it's common to tunnel your connection through SSH. This is straightforward to set up using SSH's -L option, but *Remmina* enables you to do this via its options dialog. Further there's a shared clipboard that works across all protocols. You can even set up the client to listen for an incoming connection to aid with firewall woes.





TigerVNC ★★★☆☆

TigerVNC is a client/server package that has been around since 1999. It supports multiple encodings and compression levels, so that the best possible results can be eked from low-bandwidth connections. *TigerVNC* began life as a fork of the largely-defunct *TightVNC*, and uses the same strategy of dividing the screen into subrectangles (or even subhexagons) and applying the most suitable compression (JPEG, zlib, all manner of other arcane tricks) to each one. It remains compatible with other VNC implementations, but these won't see the advanced compression benefits it offers. The client features a spartan FLTK interface comprising some buttons and an address bar and has all manner of encoding and compression methods, security options (including authentication by TLS certificates) and more. An alternative server, *x0vncserver*, is bundled for controlling an existing X session, rather than starting one anew.

Development status

Do these projects have a future?

Remmina continues to enjoy fruitful development since its inauguration in 2009. But it's merely a front-end to *libvnc* and *xfreerdp*, neither of which receive much attention beyond basic maintenance. As a result, the modern GTK3 UX belies a somewhat ageing interior. And it's what's inside that counts.

The schismatic legacy behind the original VNC (now *RealVNC*), *TigerVNC*, *TightVNC* and even *TurboVNC* is complicated, but it has been to the detriment of open source VNC implementations. *TigerVNC* is in significantly better shape than the rest, but still deserves more attention. In contrast *NoMachine* seems to be relentless in its progress. Shame it's not open source really.

X2Go was started as a project by two high-school students in 2006. Today they and a core team of four others maintain the project. They have noble goals which is good as at least three other projects based on the NX protocol are no longer maintained. From our performance test it's clear that *X2Go* isn't yet comparable to *NoMachine*, but the latter shows the protocol's capabilities and gives the team something to aim towards.

The Linux port of *Chrome Remote Desktop* was announced last July, and its lacklustre performance is forgiveable since doing things the NativeClient way is complicated, but the project could provide packages that work for more than Ubuntu 12.04.

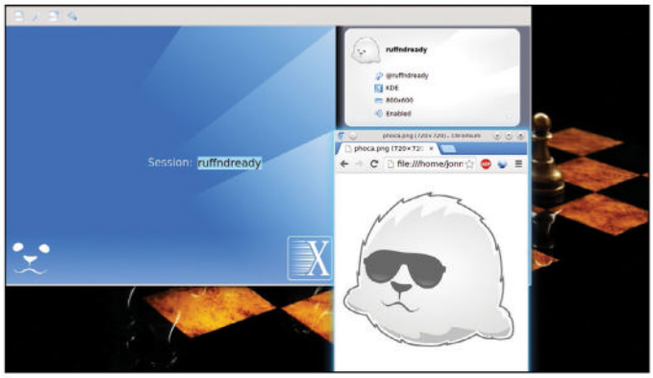
Verdict

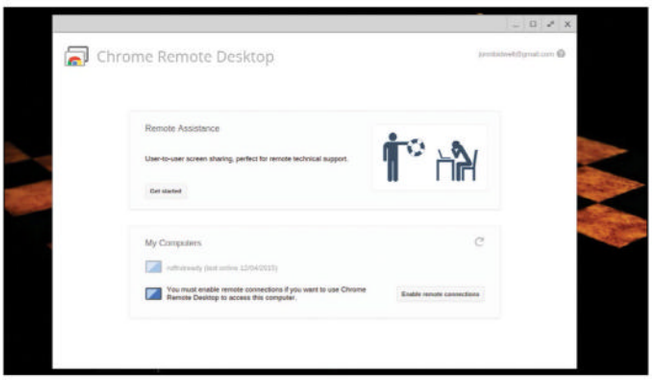
Chrome Remote Desktop ★★★☆☆
NoMachine NX ★★★★★
Remmina ★★☆☆☆
TigerVNC ★★★☆☆
X2Go ★★★☆☆

» *NoMachine is relentless and wants to be on all your machines.*

X2Go ★★★☆☆

X2go uses the NX protocol over SSH. It has many of the features of the *NoMachine* client but without the proprietary chills. It also has a lovely mascot called Phoca. As well as a huge number of compression schemes, the client allows you to choose your connection speed. Connections to sound systems, such as *PulseAudio*, and file systems can be tunnelled through the SSH connection to get around firewalls. As well as rendering the remote desktop on a dedicated display, there's a *Xinerama* extension which enables a desktop to be rendered across multiple screens. Since the *X2Go* server is forked from an old version of X.org, it doesn't support many 3D-accelerated desktops. Gnome 3, Unity and Cinnamon will work in fallback mode, but you'll have better luck with something simpler such as Xfce or Mate.



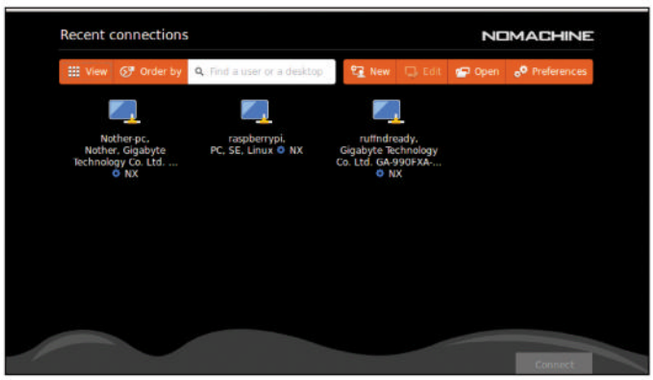


Chrome Remote Desktop ★★☆☆☆

Besides being a web-based remote desktop client *Chrome Remote Desktop* doesn't have much else to brag about. That said, being able to access a machine from a mobile device could come in handy – for instance that machine could be one that you don't have the privileges or inclination to install software. Once the required service package is installed on the server machine(s) then you can access them from the comfort of your browser. Provided your browser is *Chrome* or *Chromium*, and you're logged into the appropriate Google account, that is. To enable remote connections to a machine you must first set up a PIN of at least six digits. Once connected, you get only rudimentary controls for resizing the desktop – the app will decide what sort of quality to provide.

NoMachine NX ★★★★☆

NoMachine NX wins the in-your-face interface award, with its large, branded, black and orange windows. But if you can forgive this and its proprietary nature, then it's something of a powerhouse. Like *x0vncserver* it gives access to the currently running desktop, and like RDP the NX protocol supports audio/USB/drive redirection. It also allows file transfers and remote desktop session recording. The just-released Raspberry Pi package (which is still in alpha) also works fine. *NoMachine* can use UDP to transfer multimedia data which it can encode using H.264, VP8 or MJPEG compression. The latter uses less CPU power so is useful for low-power devices. *NoMachine 4* no longer allows SSH connections, but you can still authenticate by public key.



Protocol & desktop support

Can you use other protocols, or a fancy desktop?

While *Remmina* offers the most diverse selection of protocols, the support it provides is generally inferior to other clients. Most people will use it via VNC, which will support any desktop but lacks OpenGL capabilities on proprietary drivers. Our experience of using it with RDP wasn't particularly impressive, and the NX plugin only works for long deprecated *NoMachine 3* servers.

TigerVNC doesn't need to support other protocols but you ought to get a slightly more enjoyable desktop experience compared to *Remmina*. If you don't then fiddle with the settings. You might need to launch the server with **dbus-launch vncserver** if the server machine is logged into the same desktop environment as your intended VNC session.

X2Go and the rest of our species don't fare so well in this category as they can only connect to their own kind. Further, desktop support is poor: Qt5 doesn't work (so no KDE 5 for you), Gnome and Unity don't work (there are still problems even in fallback modes). Before we installed KDE 4, we were wondering if anything would work.

Hopefully the *Chrome* app people can fix the Qt5 issues soon. When they do though, the desktop experience needs to be improved. It would be a marvel if they could get OpenGL working, but that's a long way off. While *NoMachine* can only connect to it's own kind, its ability to serve even the most complicated desktops is outstanding.

Verdict

Chrome Remote Desktop ★★☆☆☆

NoMachine NX ★★★★☆

Remmina ★★★☆☆

TigerVNC ★★★☆☆

X2Go ★★☆☆☆

» *NoMachine's ability to work anywhere sets it apart.*

Low-bandwidth survival

Is it useable as you approach dialup speeds?

Using *Remmina* to access the Raspberry Pi with the connection artificially limited to 56K dialup speeds was not fun. However, with the colour depth and quality settings appropriately throttled, it was at least usable, provided you were patient. Under such austere compression, though, text can get hard to read, and images hard to discern.

Although getting *TigerVNC* installed on a Raspberry Pi requires using the experimental Jessie edition of Raspbian, the low-bandwidth situation in general is somewhat improved with *TigerVNC*'s advanced encoding. While we found it impressive, it still won't feel and definitely won't look like a local desktop at dialup speeds, so don't expect miracles here.

If you want your desktop to be usable over slow internet connections you can try getting by with 256 colours or get better internet access.

X2Go had difficulties initiating a connection at dialup speed but it eventually managed using Tight compression and a 4K colour palette. At this speed the desktop was barely usable, window redrawing in particular caused major delays. Fortunately most people will have a little more bandwidth, and at 25k/s things were much better.

Chromium doesn't work with **trickle**, but that didn't stop us using the **tc** command to limit bandwidth directly on the interface. Initially, our KDE desktop booted up fine, but it was nigh on impossible to use, with long delays between actions and responses. Very often, the connection would be dropped entirely. As before, allowing speeds of 25k/s made things much more palatable.

Once again *NoMachine* excels, well maybe not excels, but certainly does measurably better than everyone else. Navigating the desktop seemed much more fluid and dynamic, and though text became unreadable immediately after significant window movement, it returned to legibility soon after things had calmed down a bit.

Verdict

Chrome Remote Desktop ★★☆☆☆
NoMachine NX ★★★★☆
Remmina ★★☆☆☆
TigerVNC ★★★☆☆
X2Go ★☆☆☆☆

» *NoMachine wins out – its resilience impressed us.*

Performance

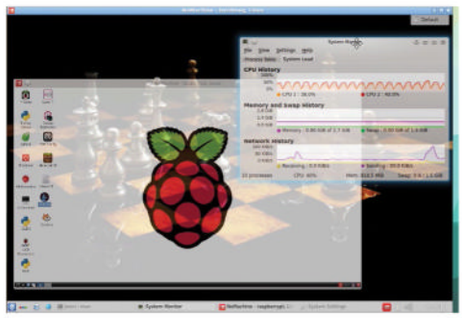
Who's got the go faster stripes?

Remmina uses *libvncserver* which has some support for Tight encoding over VNC. But perhaps it was our hardware, as on Maximum Quality settings connecting over LAN caused some tearing and stuttering on our KDE 4 desktop, mostly when dragging transparent windows around. Turning the quality down a notch made things much smoother, but the extra compression artefacts (particularly on text) are hard to miss. Using 16-bit colour also remedied this, but again keen eyes will spot the dithering.

TigerVNC performs slightly better here, being perfectly capable of playing a full screen YouTube video. Such feats are only possible if reasonably significant CPU power is available otherwise many frames will be dropped. With that caveat, 720p video streaming rarely exceeded 5MB/s, so you'd just about be able to do that on a fast ADSL2+ or cable connection

X2Go doesn't support GLX so our KDE desktop had no transparency effects to contend with. Even so, it was still noticeably less responsive than the others, excluding the *Chrome* app. This is likely due to the server's CPU being overburdened with having to software-render everything. We wouldn't recommend this for doing anything graphically-heavy.

The 'Chromoting' experience was usable, though not very enjoyable. Even with a fast internet connection there's significant lag. It also doesn't work with Qt5, although neither does *X2Go*, which means you can't use this with LXQt.



Going where NoMachine has gone before: It's Raspbian Inside KDE inside LXQt.

This is a shame since this is exactly the kind of desktop for which it would work. The VP8 compression does an OK job of keeping things presentable.

Nomachine's client is the champion; It's the only client that supports OpenGL and playing medium-weight games over LAN proved to be entirely possible. Since NoMachine is available for Windows this provides an alternative to Steam In-Home streaming or *Wine* for playing non-Linux titles.

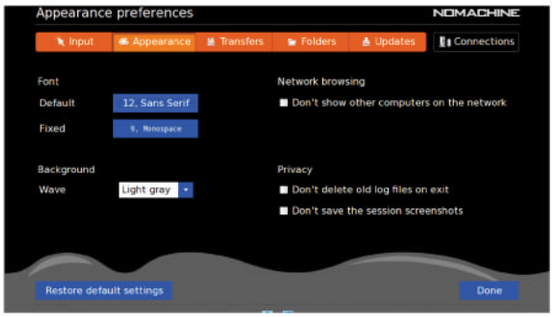
Verdict

Chrome Remote Desktop ★☆☆☆☆
NoMachine NX ★★★★★
Remmina ★★★☆☆
TigerVNC ★★★★☆
X2Go ★★☆☆☆

» *NoMachine knocks the (virtual) socks off of the rest.*

Remote desktop clients

The verdict



The option to change the wave colour in the NoMachine settings was what really swayed us.

And now the moment you've all been waiting for, unless you already looked at the ratings box. Scandalously, we've awarded first place to *NoMachine* and proprietary software, but we had no choice as it outperformed the competition.

Things could change of course – apparently Google has some smart people working for them who might pimp the *Chrome* app a little. We're really keen to see how *X2Go* progresses, but then maybe it will join so many other clients in the NX necropolis, an ignominious end for our new friend Phoca.

Since *NoMachine* no longer supports the old version of the NX protocol, distros have started using *X2Go*'s implementation of *nxproxy*, so hopefully this will stimulate new interest in the package. It's already used by *Remmina* in Arch Linux, and there's an improved Windows client called Pyhoca.

So the potential is there, one easy thing that needs doing is tidying up the interface, particularly pruning, or at least putting under hierarchy, the ludicrous number of compression schemes it offers.

It's interesting how the VNC clients largely failed to compete with *NoMachine*, even *TigerVNC* using the highly-optimised libjpeg-turbo library. Perhaps this speaks to a protocol in its twilight years, or the rise of a new era of NX domination. Then again perhaps some exciting VNC development happens that turns things around shortly after this issue goes to press. (Like the last Roundup where we said there hadn't been a major *VirtualBox* release for years and v5 popped up).

Naturally, some readers will want to stick with open source and *Remmina* is a perfectly good choice here – if you really need to work on a remote desktop then you're probably willing to accept some small performance hit. It's only in comparatively rare situations that you will see any benefit to using *TigerVNC* and it's hard to recommend that one to the kids – who demand svelte icons and layouts as opposed to a glaring textbox that demands input.

> **"It's interesting how the other VNC clients largely failed to compete with NoMachine."**

1st NoMachine NX ★★★★☆

Web: http://nomachine.com **Licence:** Freeware **Version:** 4.5

» A proprietary champion, whatever next?

2nd Remmina ★★★☆☆

Web: http://bit.ly/Remmina **Licence:** GPL **Version:** 1.1.2

» A great all-rounder, but can't compete with the champion.

3rd TigerVNC ★★★☆☆

Web: http://tigervnc.org **Licence:** GPL **Version:** 1.4.3

» The eye of the tiger burns not so bright.

4th X2Go ★★☆☆☆

Web: http://wiki.x2go.org **Licence:** GPLv2 **Version:** 4.0.3.2

» It shows promise, but it's got a long way to go.

5th Chrome RD ★★☆☆☆

Web: http://bit.ly/1GadugV **Licence:** Freeware **Version:** 42

» A neat idea, but needs to mature before it catches on.

Over to you...

Are you satisfied with our verdict? Don't you think that Phoca is the cutest? Let Linux Format know: **lxf.letters@futurenet.com**

Also consider...

If you're interested in game streaming, then Steam's In-Home Streaming is probably still going to beat *NoMachine*. If you have an Nvidia graphics card in your Windows machine, then you can also try using the open source gamestream *Limelight* (**http://limelight-stream.com**) on the client machine, even if that machine is a Raspberry Pi, to a similar end.

We didn't really mention *RealVNC*, which is where the original developers of the protocol are, version 5 of its free client was released in 2012 and the latest update, 5.2, came out in February of 2015.

The 5.0 series is a marked departure from *RealVNC*'s open source licensing though, and now you need to sign up for a free key (with an optional 30-day trial of extra features). But haven't we all seen enough proprietary software for one Roundup? Also don't forget the remote desktop clients that come bundled with desktop environments like Gnome (*Vinagre*) and KDE (*KRDC*). These are fine, but we didn't include them since we're all about trying new things.

HACKER'S MANUAL 2016

HACKER'S MANUAL 2016

Security

The internet's naughty people aren't getting any less naughty.

Source credits

Authors:
Nate Cardozo, Kurt Opsahl, Rainey Reitman

Editors:
Parker Higgins, Dave Maass

Formatting:
Parker Higgins
A publication of the Electronic Frontier Foundation, 2015

Find the original at:
www.eff.org/who-has-your-back-government-data-requests-2015

This work, *Who Protects Your Data?*, is an abridged derivative of *Who Has Your Back? 2015: Protecting Your Data From Government Requests* by the Electronic Freedom Foundation

WHO PROTECTS YOUR DATA?

The **Electronic Frontier Foundation** has released its fifth annual report on online privacy and transparency and explains the implications for all of our data.

We live digital lives: from the videos shared on social networks to location-aware apps on mobile phones; from log-in data for connecting to our emails to our stored documents and, of course, our search history. The personal, profound and even absurd are all transcribed into data packets and whizzed around the fiber-optic arteries of the network.

While our daily lives have upgraded to the 21st century, the law hasn't kept pace. To date, the US Congress hasn't managed to update the 1986 Electronic Communications Privacy Act to acknowledge that email stored for longer than six months deserves identical protections to email stored for less than six months. Congress also dragged its feet on halting the NSA's indiscriminate surveillance of online communications and has yet to enact the strong reforms we deserve. Congress is even on the precipice of making things far worse by considering proposals that would mandate government backdoors (as is the UK government, currently) into the technology we rely on to digitally communicate.

In this climate, we're increasingly looking to technology companies themselves to have the strongest possible policies to protect user rights. But which companies will stand by users, insisting on transparency and strong legal standards around government access to user data? And which companies make those policies public, letting the world – and their own users – judge their stances on standing up for our privacy rights?

For four years, the Electronic Frontier Foundation has documented the practices of major internet companies and service providers, judging each ones publicly available policies and highlighting best practices. Over the course of those first four reports, we watched a transformation take place in the practices of major technology companies.

Overwhelmingly, tech giants began publishing annual reports about government data requests, promising to provide users notice when the government sought access to their data, and requiring a search warrant before handing over user content. Those best practices we identified in early EFF reports became industry standards in a few short years, and we're proud of the role our

> **"While our daily lives have upgraded to the 21st century, the law hasn't kept paces."**

annual report played in pushing companies to institute these changes. But times have changed, and now users expect more.

The criteria we used to judge companies in 2011 were ambitious for the time, but they've been almost universally adopted in the years since then. Now, users should expect companies to far exceed the standards articulated in the original *Who Has Your Back?* report. Users should look to companies such as Google, Apple, Facebook and Amazon to be transparent about the types of content that is blocked or censored in response to government requests, as well as what deleted data is kept around in case government agents seek it in the future. We also look to these companies to take a principled stance against government-mandated backdoors.

In this, the fifth annual *Who Has Your Back?* report, we took the main principles of the prior reports and rolled them into a single category: Industry Accepted Best Practices. We've also refined our expectations around providing users notice and added new categories to highlight other important transparency and user rights issues. We think it's time to expect more from Silicon Valley. We designed this report to take the principles of *Who Has Your Back?* up a notch and see which companies were still leading the pack.

Evaluation criteria

To that end, we went ahead and used the following five criteria to assess company practices and policies:

1 Industry Accepted Best Practices This is a combined category that measures companies on three criteria, and which they must fulfill all of in order to receive credit:

- Does the company require that the government obtain a warrant from a judge before handing over the content of user communications?
- Does the company publish a transparency report, ie regular, useful data about how many times governments sought user data and how often the company provided user data to governments?
- Does the company publish law enforcement guides explaining how they respond to data demands from the government?

2 Tell users about government data requests To earn a star in this category, internet companies must promise to tell users when the US government seeks their data unless prohibited by law, in very narrow and defined emergency situations, or unless doing so would be futile or ineffective.

A notice gives users a chance to defend themselves against overreaching government demands for their data. The best practice is to give users prior notice of such demands, so that they have an opportunity to challenge them in court. We have thus adjusted our criterion from prior years. We now require that the company provide advance notice to users except when prohibited by law or in an emergency and that the company also commit to providing delayed notice after the emergency has ended or when the gag has been lifted. As we were drafting last year's report, we let the companies know that we were going to make this adjustment for 2015 to give them a full year to implement procedures to give delayed notice when appropriate.

> "We've also refined our expectations around providing users notice."

› The EFF raised the bar for the 2015 report.

3 Publicly disclose the company's data retention policies This category awards companies that disclose how long they maintain data about their users that isn't accessible to the user—specifically including logs of users' IP addresses and deleted content—in a form accessible to law enforcement. If the retention period may vary for technical or other reasons, the company must disclose that fact and should publish an approximate average or typical range, along with an upper bound, if any. We awarded this star to any company that discloses its policy to the public—even if that policy is one that EFF strongly disagrees with, eg if the company discloses that it retains data about its users forever.

4 Disclose the number of times governments seek the removal »

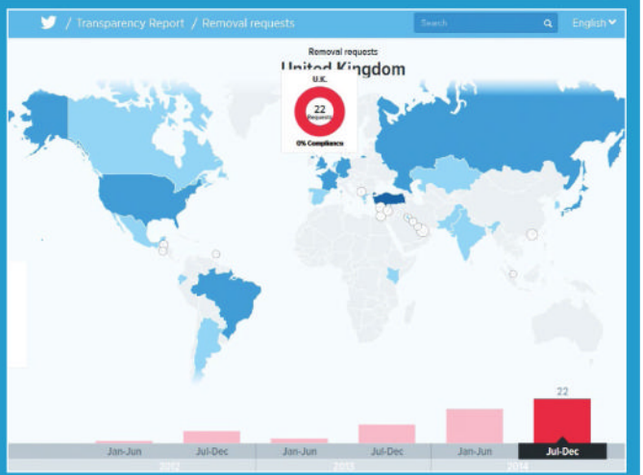
Government removal requests

For more than a year, EFF's lead investigative researcher Dave Maass has been reporting on how Facebook cooperates with prison systems across the United States to block prisoner access to the social network. Facebook had even set up a dedicated 'Inmate Account Takedown Request' form to help prison officials quickly and easily flag prisoner-run accounts for suspension, even when the accounts didn't violate any of Facebook's terms of service.

This practice was the inspiration for EFF's newest category: tracking how often companies are removing content or shutting down accounts at the behest of the government. To earn credit in this category, companies need not refuse all or even any government content removal requests. Rather, they must simply be transparent about how often they are blocking or removing content or accounts.

Though this is simple enough, many companies are falling short in this area including Facebook, the company whose practices inspired the category. We evaluated 24 companies and 15 received credit in this category, though several don't host content.

A particularly strong example of this practice is the data published by Twitter, which includes an interactive map that allows users to mouse over countries and get details about content removal requests over a six-month period.



› Twitter offers a comprehensive breakdown of all its take-down requests and compliance.

» **of user content or accounts and how often the company complies** It's now industry standard practice to have transparency reports. We believe that companies' responsibility to be transparent includes not only disclosing when governments demand user data, but also how often governments seek the removal of user content or the suspension of user accounts and how often the company complies with such demands. We award a star in this category to companies that regularly publish this information, either in their transparency report or in another similarly accessible form. Companies should include formal legal process as well as informal government requests in their reporting, as government censorship takes many forms.

5 **Pro-user public policies: opposing backdoors** Every year, we dedicate one category to a public policy position of a company. For three years, we acknowledged companies working publicly to update and reform the Electronic Communications Privacy Act. Last year, we noted companies who publicly opposed mass surveillance. This year, given the reinvigorated debate over encryption, we are asking companies to take a public position against the compelled inclusion of deliberate security weaknesses or other compelled backdoors. This could be in a blog post, in a transparency report, by publicly signing a coalition letter, or through another public, official, written format. We expect this category to continue to evolve, so that we can track industry players across a range of important privacy issues.

› The full results of the EFF's annual report highlight a very poor result from the popular messaging service, WhatsApp.

	Follows industry-accepted best practices	Tells users about government data demands	Publicly discloses policies on data retention	Discloses government content removal requests	Pro-user public policy: opposes backdoors
Adobe	★	★	★	★	★
amazon.com	★	☆	☆	★	★
Apple	★	★	★	★	★
at&t	★	☆	☆	N/A	☆
COMCAST	★	☆	★	N/A	★
CREDO mobile	★	★	★	★	★
Dropbox	★	★	★	★	★
facebook	★	★	★	☆	★
Google	★	☆	☆	★	★
LinkedIn	★	★	★	☆	★
Microsoft	★	★	☆	☆	★
Pinterest	★	★	☆	★	★
reddit	★	★	★	★	☆
slack	★	☆	★	★	★
snapchat	★	☆	★	N/A	★
SONIC.	★	★	★	★	★
tumblr.	★	★	☆	☆	★
Twitter	★	☆	★	★	★
verizon	★	☆	☆	★	☆
WhatsApp	☆	☆	☆	N/A	★
WICKR	★	★	★	N/A	★
WIKIMEDIA	★	★	★	★	★
WordPress.com	★	★	★	★	★
YAHOO!	★	★	★	★	★

> **"The tech industry stands united against government-mandated backdoors."**

The Good, bad & ugly

We are pleased to announce that nine companies earned stars in every category that was available to them (*see right*). It should be noted that some companies host little or no content, and thus the transparency about government data removal requests may not apply to them. These companies show that it's practical for major technology companies to adopt best practices around transparency and stand by their users when the government comes knocking. Unfortunately, not all companies are embodying such forward-thinking practices. Two major telecoms – Verizon and AT&T – received especially poor results, thus continuing a trend we identified in prior reports where large telecom providers fail to keep pace with the rest of the tech sector.

Notably, some companies that act as Internet service providers (ISPs) and general telecommunications providers are leading the way in adopting strong policies in defence of user rights. In particular, Credo and Sonic again received credit in every category. Comcast is close behind, earning 3 out of 4 possible stars. We hope other telecoms can rise to these standards in the coming years.

It's also clear that the tech industry stands united against government-mandated backdoors. We found that of the 24 companies we evaluated 21 have public statements opposing backdoors, which weaken security and endanger user privacy. ISPs, cloud storage providers, webmail providers, and social networks are overwhelmingly aligned in rejecting government-mandated security weaknesses.

Best practices

These standards were developed over the course of four years of EFF reports, and they encompass three of the main issues at the heart of *Who Has Your Back?*: requiring a warrant before handing over user content, publishing regular transparency reports, and publishing law enforcement guides. The transparency reports and the law enforcement guides help users understand how often and under what circumstances the companies are responding to government data requests, while the warrant for content ensures a strong legal requirement be met before data is handed to law enforcement.

In 2011, no company received credit in all of these categories. This year, 23 of the 24 companies in our report have adopted these principles. It's clear that these best practices truly are accepted by the technology industry, but WhatsApp is notably lagging behind.

Notifying users

This year, we asked companies to do more than simply promise to inform users about government data requests. We also asked them to provide advance notice to users before handing the data to the government.

Linux escapees

The EFF report is very US centric, but as a good portion of the world uses so many of these US-based services, it's a report that affects the majority of us. As open source aficionados, the average *Linux Format* reader is far more aware of the privacy implications and far better set to do something about the situation. The mag has previously looked at *OwnCloud* [see Tutorials, **LXF190**] to see just how easy it is to create your own cloud-based document collaboration and sharing system.

This means it's possible to put into place your own means of escaping corporate rules, regulations and privacy issues. The reality is that not everyone is in such a position and it's in everyone's interest that companies offering online services do so in ways that protect us all without kowtowing to government demands. Or at least make people aware of how their data is store and when – if ever – access is given to government bodies.

Cloud services are only going grow in number, and the amount of data we'll store on them will do the same. **LXF** will be looking at new open source cloud options in the future as more services, such as **www.onlyoffice.com** appear.

› Running your own cloud services, using OwnCloud, is one way to secure your own privacy.

In cases where companies are prohibited from doing so, we asked the companies to promise to provide notice after an emergency has ended or a gag was lifted. Because we knew it would take significant engineering and workflow changes for some of the larger companies to implement these practices, we gave them more than a year's notice that this criterion would be included in the 2015 report.

Two companies, Google and Twitter, who had previously earned credit in our report for telling users about government data requests did not receive credit this year because they didn't have policies in place that tell users after a gag has been lifted or an emergency ended.

Out of the 24 companies, 15 companies we evaluated did meet this stronger criterion, and we're pleased to see the industry is evolving in this way. We were particularly impressed by the strong policy adopted by Dropbox, which states the following:

"Dropbox's policy is to provide notice to users about law enforcement requests for their information prior to complying with the request, unless prohibited by law. We might delay notice in cases involving the threat of death or bodily injury, or the exploitation of children."

Data retention policies

For the first time this year, we extended our evaluated companies to cover whether they were transparent about what deleted data they continued to store. Often, users may not realise that data they delete from an email service provider or off a social network is still stored and available to law enforcement agencies upon request.

Transparency is the first step to educating users about what happens to their deleted data, so we are evaluating companies on their transparency practices in this category. Note that we aren't making specific requirements about a company deleting data after a certain time. Indeed, some companies publicly state that they maintain deleted data and server logs indefinitely – a practice we think is terrible for users. However, for this report, we're just asking companies to be clear about retention periods for data collected that may not be easily viewable to the user (including IP addresses and DHCP data) as well as content that users deleted.

Again, we saw 15 companies out of the 24 that we evaluated receive credit in this category. We were particularly impressed by the clarity and detail of Comcast's disclosures. The company maintains historical call detail records for Xfinity Voice telephone service for two years. This includes local, local toll, and long distance records. In limited instances, older records may be available, but will require additional time and resources to retrieve. For more details about its data retention policy see the Comcast Law Enforcement Handbook at **http://bit.ly/LXFitsthelaw**.

Opposing backdoors

One of the big trends we're seeing across the technology industry is a emphatic rejection of government-mandated security weaknesses. In fact, 21 out of the 24 companies we evaluated took a public position opposing the use of backdoors. This is a powerful statement from the technology community that Congress and the White House should heed.

Many of the companies have signed onto a letter organised by the Open Technology Institute that opposed mandates to intentionally weaken security, which stated the following:

"We urge you to reject any proposal that US companies deliberately weaken the security of our products … Whether you call them 'front doors' or 'back doors', introducing intentional vulnerabilities into secure products for the government's use will make those products less secure against other attackers. Every computer security expert that has spoken publicly on this issue agrees on this point, including the government's own experts."

The EFF's conclusions

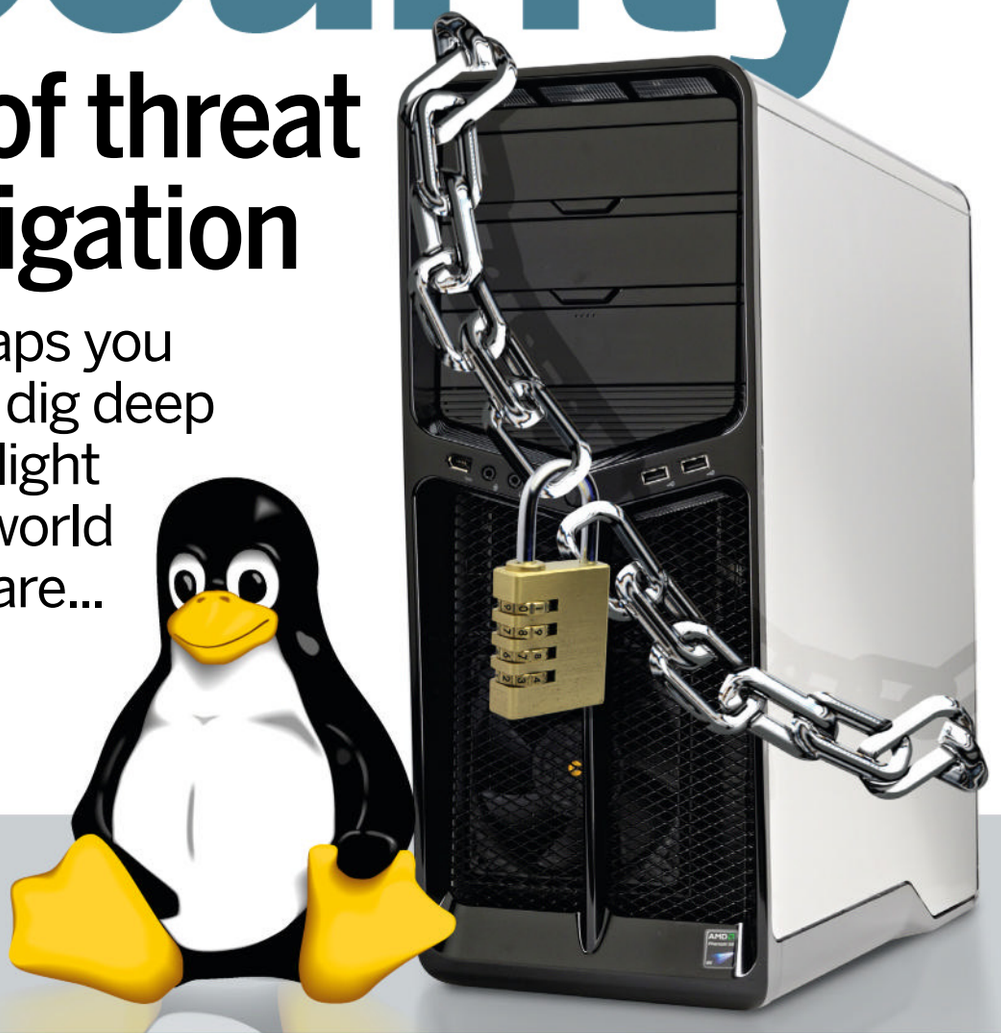
We are pleased to see major technology companies competing on privacy and user rights. Practices that encourage transparency with users about government data requests are becoming the default for companies across the web. While we're only able to judge a small selection of the tech industry, we believe this is emblematic of a broader shift. Perhaps invigorated by the ongoing debates around government surveillance and in response to growing public attention around these issues, more and more companies are voluntarily speaking out about government data requests and giving users the tools to fight back.

We think that this type of transparency can help prompt broader discussion and systematic change about how and when governments access user data and eventually prompt Congress to clarify and improve the privacy laws for digital data. We also recognise that tech companies are in a position to know about and resist overbroad government requests, so we need to do everything within our power to encourage them to speak out and fight back. In handing our data to these companies, we've handed them a huge responsibility to do what they can to stand up for privacy. We're pleased that many of the companies we evaluated are stepping up to the task.

Security

A song of threat and mitigation

Scared? Perhaps you should be. We dig deep to shed some light on the shady world of Linux malware...

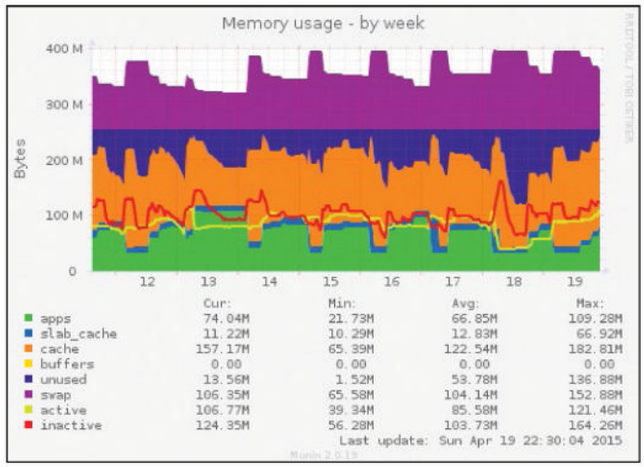


Sometimes in the pub you might overhear someone mansplaining that Linux is 'more secure' than Windows. On one level he's right, desktop Linux users have nowhere near as much to fear in terms of viruses and malware than their Windows counterparts. It's not that they don't exist, but it represents such a tiny portion of the malware ecosystem that it's perfectly reasonable (modulo safe browsing habits) to not worry about it.

This boils down to a simple numbers game: Any survey will put Linux at less than 2% of total desktop market share. With that in mind, it makes much more sense for malware authors to target Windows and (increasingly) Mac systems. Victims can be infected in a number of ways: usually opening dodgy email links and attachments or by visiting compromised websites. Very occasionally an OS vulnerability can be exploited that allows an attacker to remotely execute code on the victim's machine. A compromised – or even a downright malicious – website may try to foist malware onto visiting machines using a variety of techniques. But by far the most prevalent attack vector is the *Flash* plugin. Malfeasant applets can leverage weaknesses here which execute arbitrary code on the remote machine, entirely unbeknownst to the user. It's easy (and in some cases justified) to blame Adobe for shipping dodgy code, but again the real issue is that so many people have Flash installed that it makes good business sense to target them. This is also true for Adobe's *PDF Reader* and Oracle's Java plugin. *Chrome 42* has disabled official support for all NPAPI plugins, citing the large attack surface they levy against the browser.

> **"Malfeasant applets can leverage weaknesses in Flash which execute arbitrary code."**

But the compromised web servers doing the malware-foisting are, more often than not, Linux boxes. And our man in the pub told us these are secure. In fact, there are any number of ways by which a Linux box could end up 'owned' by a villain. And if it's hosting a popular website or sensitive database then all the more motivation for someone to attempt to do so. We often get questions from readers asking how to set up a secure LAMP stack or suchlike, and unfortunately there isn't really an easy answer. There are a few things you should (and plenty of things you shouldn't) do, but there's no accounting for a talented adversary, or some obscure 0-day bug in one of the many components upon which a modern LAMP server relies. That said, let's focus on what we can mitigate against.



› ***Munin*** **comes from the Norse for 'memory'. Once you've gathered some stats, sudden increases in resource demand become much easier to spot.**

It's a config thing

While a compromise could be the result of some new vulnerability with a catchy name and stylish logo, by far the most common cause is good old-fashioned server misconfiguration. A server that is overly permissive about what it lets an outsider access or modify is a server that's asking for trouble. Common mistakes include allowing the web server to write to sensitive files, or having an SQL server accessible to the public (when it need only listen locally or for connections from selected IPs). Alternatively attackers might get lucky through bruteforcing SSH (or other) logins. This shouldn't really be possible – password logins should be disabled (at least for sensitive accounts) in favour of public key auth, and multiple failed login attempts (which are time consuming anyway) should result in a temporary ban.

Thus, check your permissions, have servers only listen on the localhost address where possible (and connect via an SSH tunnel if you need to access them), and have some effective firewall rules in place. In the latter case, it's prudent to lock down outgoing traffic as well as incoming. This might just stop a malevolently installed program from phoning home (they often communicate over IRC) and wreaking havoc. Root logins should be disabled, and authorised users should use **sudo** or **su** to do administrative tasks, since it leaves an audit trail by way of the system log. Assuming then that our front door, as it were, is secure, how else might ne'er-do-wells access our box? Well, that depends on how secure the rest of it is.

PHP scripts provide a common attack surface against web servers, though ultimately any server side language could fall prey to similar attacks. Wherever your web application accepts user input, beware. Since you have no control of exactly what users might input, it's important to sanitise it. Otherwise a malicious user can inject code which, depending on the context, could prove harmful. For example a simple PHP search form might look like:

```
<form method='get' action='search.php'>
<input name="search" value="<?php echo $_GET['search'];?>" />
<input type=submit name='dosearch' value='Search' /></form>
```

Input is passed unchecked to the search.php script, which means a user could inject some JavaScript, for example searching for the string:

```
"><script>alert(0)</script>
```

Results in an alert box. The initial double-quote terminates the HTML attribute **value**, then the right bracket escapes from the **input** element. To guard against these shenanigans, be sure to use the available functions to filter the input. The following code will escape any special characters so they won't cause harm:

```
<?php
$input = "pointy brackets <and> & ampersands?";
var_dump(filter_var($url,FILTER_SANITIZE_SPECIAL_CHARS));
?>
```

While the output in the browser will look the same, if you look at the HTML source generated by the script, you will see that it in fact outputted the string:

```
"pointy brackets &lt;and&gt; &amp ampersands?"
```

The escaped characters are much less use to an attacker. You can also use **FILTER_SANITIZE_STRING** here, which removes (rather than escapes) tags. You could equally well have injected PHP here or, where the input is passed to a database, SQL commands. When using PHP to interface with databases, it's worth using the PDO (PHP Data Objects) API as opposed to MySQLi. This will ensure that data will never be mistaken for instructions.

Once discovered and confirmed, vulnerabilities are referenced through the Common Vulnerabilities and Exposures (CVE) system, although individual products and companies may have their own internal systems too. In the case where information

How to update when you can't update

There are, regrettably, a surfeit of servers running distributions (distros) long past their support window. Admins of these boxes really should get their act together, but if upgrading the OS is out of the question then you should attempt to backport important security fixes. Sometimes people will generously provide packages for your ageing distro, which is convenient but raises a question of trust. In general, you'll have to roll your own packages, incorporating any new security fixes. Source packages for old distros are easy to find (for old Ubuntu versions look on **https://launchpad.net** and **http://archive.debian.org** for Debian).

It's a very good idea to set up a virtual machine that's as close a copy of your aged server as you can manage. You'll also need a working **gcc** toolchain, the set up of which may involve some dependency hell, and you'll also require all the package's build dependencies.

You won't want to do any major version upgrades of vulnerable software since this will likely bork your system, instead patches will need to be adjusted to fit the old version, which will involve some trial and error. If you're using a Debian-based distro then add the patch to the **debian/patches/all directory**, inside the package source's directory, and add the patch name to the file **debian/patches/series**. Then run **debuild** to make the package.

relating to a new vulnerability is embargoed, due to it not being made public, a CVE identifier can still be reserved until it is deemed safe to publicize the details. These will be first disclosed only to relevant people so that patches, or at least suitable workarounds, are available come their announcement. Various distros provide their own security advisories as well, eg **https://security.gentoo.org**. CVE provides a central exchange for rapidly disseminating information about emergent and historic issues.

Failure to apply patches and security updates is asking for trouble. Comparatively few attacks are the result of 0-day exploits and widely available tools enable attackers to scan potential marks for known vulnerabilities. Major distros are quick to patch against newly discovered flaws, so it's important to update affected packages, even if it means minor interruptions as services are restarted. Five minutes of downtime and a few grumbling users are vastly more preferable than having data stolen or having to rebuild the whole system because someone snuck in and installed a rootkit. HP's Cyber Risk report (released earlier this year) claims that 44% of breaches were the result of vulnerabilities that have been public for two to four years, which is a sad indictment against sysadmins. An even worse statistic from Verizon's Data Breach Investigations report is that nearly 97% of successful exploits last year were the result of 10 known issues, eight of which have been patched for over 10 years. While it's easy to read too much into such figures, a fair conclusion to draw is that hackers will go for the low-hanging fruit.

There are some legitimate cases where security updates cannot be applied in the usual way. Embedded systems, for example, don't typically provide any kind of package management. They also tend to run on non-x86 architectures which makes compiling your own binaries something of a pain. The box (*see Open vs Closed, below*) provides some guidelines on how to proceed if you can't update packages by the standard channels, but this is really last resort stuff. Just upgrade your OS and keep it up to date and life will be made a whole lot easier. Debian Jessie will be released by the time your read this, if you're looking for a solid OS with long-term support. Once you've upgraded your ageing scripts/databases/wotnot and got rid of any legacy PHP on your website, you can rest assured subsequent package upgrades probably won't break it for the next three years, thanks to Debian freezing program versions and only applying security fixes.

Crouching malware

Vulnerabilities can be chained together, eg some dodgy PHP might enable an attacker to upload their own scripts to your server, a problem with *Apache* might enable this script to get executed, whereupon it exploits a privilege escalation bug somewhere else that enables it to run as root. At this point your machine is effectively under the control of the attacker and all your data should be considered compromised. Of course, all of this could in theory happen without you noticing: Everything might look and feel perfectly fine, but a tiny Flash applet on your home page may now be serving your visitors a delectable blend of the finest malware. For this reason, it's important not to ignore a security update because the vulnerability it addresses doesn't immediately grant root access. It's beneficial to get into the habit of regularly scrutinising your server logs. These can be quite unwieldy, but there are tools that can help you. *Logwatch* is a particularly handy tool which can summarise accesses to SSH, web, database and any other services you're running into an easily-digestable format. The popular Perl-based *Awstats* provides an attractive web interface for perusing web, FTP or mail server logs.

It's also prudent to keep an eye on system load. The **uptime** command gives you one second, one minute and fifteen minute averages of CPU load, but you can graph historical data using a web-based tool such as *Munin*. The *vmstat* program gives you information about CPU wait times and swap requests which, when found in abundance, point to heavy disk I/O and memory hogging operations. Be on the lookout for any rogue processes. The command

```
ps awwlx --sort=vsz
```

will lists processes sorted by virtual size, which includes shared library and swap usage. So any heavy hitters will be displayed at the end. But rogue programs need not be large, or (in the case of a rootkit) visible at all.

Hidden rootkit

Rootkits are malfeasant programs that use a variety of stealth techniques to evade detection. They can hide inside other programs, the kernel itself, or even your BIOS or other device firmware. In the latter cases, they can be entirely undetectable since any system calls which would ordinarily detect them can be subverted. There are programs, such as *chkrootkit* and *rkhunter*, that can check for some known Linux rootkits. You can also install an intrusion detection program such as *AIDE* which will spot changes to your filesystem, but it can take some configuring. Some rootkits and other malware may depend on a rogue kernel module. You can mitigate against this by enabling module-signing in your kernel. The kernel can generate

Metasploit Framework is a valuable resource for penetration testers, even this ASCII cow agrees

Open vs closed

It's a fairly widespread fallacy that since open source code is public it is inherently more open to attacks. 2014 alone saw an embarrassing **goto** bug in GnuTLS library, the ShellShock bug in *Bash*, and the Heartbleed bug in OpenSSL. While anyone with enough coding experience can, after the fact, snort derisively at the code that caused these vulnerabilities, it doesn't mean that the mistakes are so glaring that they should have been spotted earlier. Reading other people's code is hard, and while projects like OpenSSL review all contributions, they're not going to catch everything. Having their dirty laundry aired in this way may be slightly ignominious, but at least the process from discovery to repair is carried out openly: You can laugh at that unchecked bound, but you can also nod approvingly at a well-executed fix.

Anyone that says proprietary code doesn't suffer this much, need only turn on a Windows machine on the first Tuesday of a given month. In April 2015 there were 11 patches (four of which were critical), and while we'll never know the details, we see phrases like 'privilege escalation and 'security bypass' etc, none of which are things people want in an OS. Such vulnerabilities can also be found through techniques like fuzzing. Once the software patches are released, they can be reverse-engineered and weaponised.

Following the National Cyber Security Survey, CERT-UK is tasked with handling the cyber response to incidents in the UK.

a private key and certificate (which contains the public key) for you, or you can use your own. Any further modules you compile will need to be signed with this key before the kernel will load them. A handy Perl script in the form of **scripts/sign-file** inside the kernel sources directory will do just this, provided you are in possession of the private key. For example, to sign the module acx100 (an out-of-tree driver for certain Texas Instruments wireless chipsets):

```
$perl /usr/src/linux/scripts/sign-file sha512 /
mnt/sdcard/kernel-signkey.priv /mnt/sdcard/
kernel-signkey.x509 acx100.ko
```

Notice how our key and certificate are stored on an SD card. The certificate is public, so we can leave it anywhere, but under no circumstances should you store private keys on the same medium as the data they protect. This is exactly like locking your front door and leaving the key in the lock. Once the signed kernel is compiled you should copy this key to a safe place (ie not anywhere on that system) and securely erase the original. Signing kernel modules is good, but the kernel itself could be poisoned so it allows rogue modules to be loaded. This can be worked around by booting a signed kernel from EFI, which, though beyond the scope of this article, is worth investigating.

Hashed and salted passwords on Linux are stored in the file **/etc/shadow**, which is only readable by root. If an attacker had sufficient resources then they could try and brute force these passwords, so that the credentials could be used to gain access to other systems. Any databases on a compromised machine are ripe for plundering – if the machine is holding personal information then this too can be used to gain access to other systems, or to carry out social engineering attacks. The attacker could move to lock you out of your machine, or just delete everything on it, but that would give the game away.

There's all manner of imaginative fun that an attacker can have with your box. Security researcher, Andrew Morris runs a honeypot (a setup designed to bait and monitor attacks) which recently saw an attacker try and co-opt one of its machine's resources so that they could be provisioned and sold as VPSes (see **http://morris.guru/huthos-the-totally-100-legit-vps-provider**). A common trick used to be to install a cryptocurrency mining daemon, although the rewards nowadays are negligible. However, a vulnerability in the *DiskStation Manager* (DSM) software that runs on Synology NAS devices led to thousands of them being turned into Dogecoin miners. It's thought the attackers netted over $600,000 this way. Synology did issue a fix for DSM in February 2014, but the mass attack continued to generate revenue as many users didn't apply it.

The Metasploit Framework provides an array of modules which enable pen (penetration) testing using already known vulnerabilities. For example, to search for CVE-listed vulnerabilities from last year use:

```
msf > search cve:2014
```

We might be interested in the Heartbleed bug (CVE-2014-0160):

```
msf > use auxiliary/scanner/ssl/openssl_
heartbleed
… > set RHOSTS targetmachine.com
… > set verbose true
… > exploit
```

If a Metasploit module exists for an exploit, then there's a fair chance that said exploit is being used in the wild somewhere, so take the time to test any modules that seem relevant.

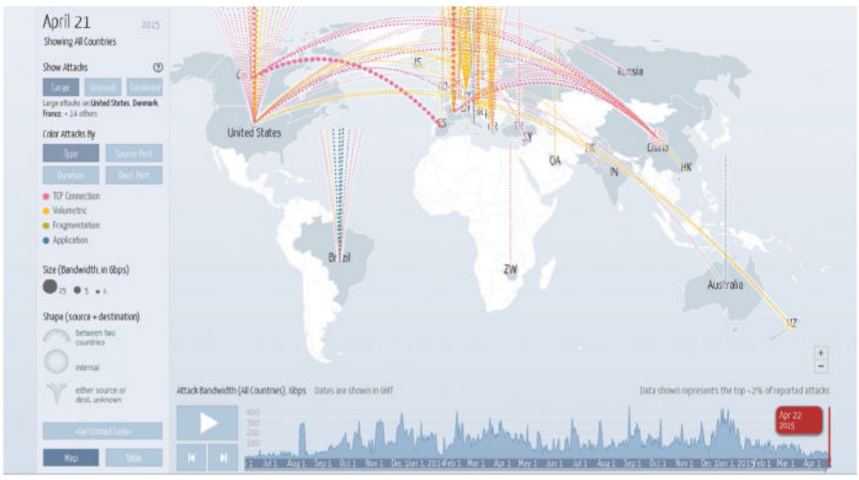
> **"In an ideal world anyone who discovered a 0-day would responsibly disclose the issue."**

We've mentioned 0-day exploits before, without really defining what they are. These are weaknesses which have not been disclosed either publicly or privately. By definition then, no fixes are available and all you can do is hope that you will never get bitten. In an ideal world anyone who discovered a 0-day would heed their moral obligation to responsibly disclose the issue to the appropriate project.

DayZ(ero)

Unfortunately, this won't always happen – cyber criminals from various underground communities will pay top dollar for a handy 0-day, and it's unlikely that they'll use this knowledge honourably. Perhaps more disturbingly, documents leaked by Ed Snowden show that governments (including the USA) are involved in purchasing and stockpiling these exploits. Facebook's bug bounty and *Chrome*'s pwn2own contest provide good motivation for hackers to disclose their vulnerabilities responsibly, but many open source projects lack the resources to offer such financial incentives. In fact, many projects are barely able to support themselves: Werner Koch, citing fiscal pressures, came close to abandoning GPG, the only truly open source public key encryption client. Fortunately, he was bailed out by a grant from the Linux Foundation and also received, following a social media campaign, a generous sum in public donations. Thankfully, many developers working on high-exposure Linux projects are employed or sponsored by corporate entities.

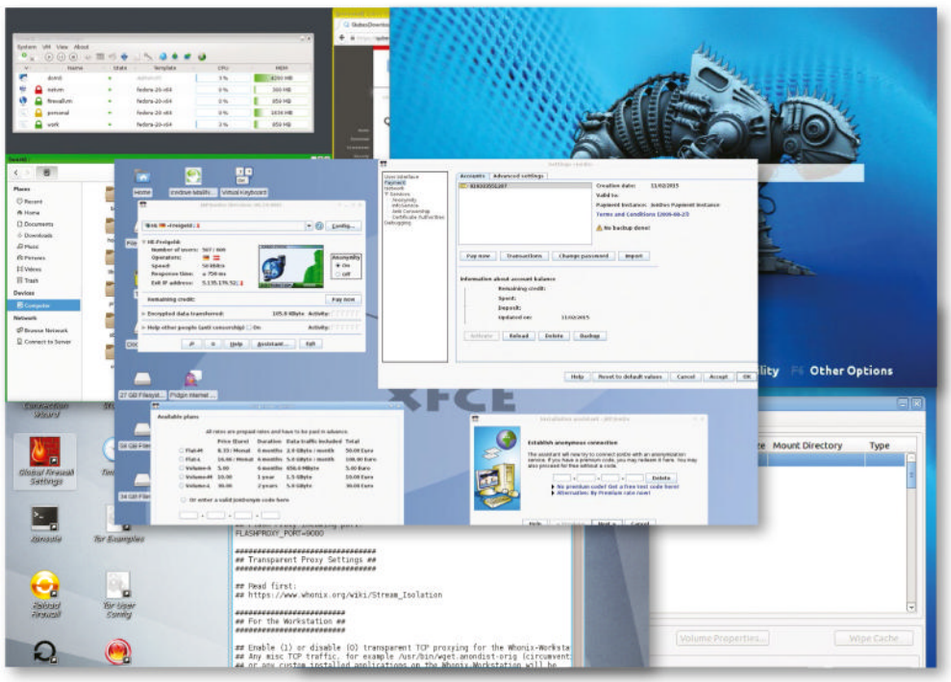
This is merely a glance over the Linux security landscape. There are all many other checks you can do, many other defences you can employ, and, regrettably, many more ways your server can fall victim to an attack. Be vigilant, heed the advisories, and stay safe out there, friend.



If you don't believe DdoS attacks are real www.digitalattackmap.com will prove you wrong.

Privacy distros

Cover your tracks and keep your identity private – we compare special-purpose Linux distros that'll help you stay invisible on the web.



How we tested...

Nearly two years ago mainstream media started discussing PRISM, which raised a lot of concerns about privacy and anonymous access to the Internet. Shortly after that, *Linux Format* magazine came out with great Anonymous distros round-up, which highlighted a noticeable outburst of new releases for Tails, Whonix and other Linux distributions for the security conscious user. This time we revisit the topic with a different selection of contenders and a changed perspective, too. We'll cover: the current state of actively maintained distros; their availability; ease of use; performance; feature set and documentation, and last, but not least; we'll cover the level of compromise they require for regular, general-purpose computing.

There are numerous use cases where someone security conscious may want to use a specialised and non-mainstream Linux distribution instead of a regular one. So we selected five diverse options, each with its own traits and benefits.

Tails is perhaps the most well-established system we're covering, and claims to provide anonymous internet access, circumventing any censorship. Ubuntu Privacy Remix (UPR) provides anonymity together with a strong means of securing your data. It runs only in live mode, encrypts your data and protects it against unsolicited access. Whonix boasts nearly the same features as Tails but goes even further by dividing your workflow into two parts: server and workstation. Qubes OS implements the 'security by compartmentalisation' approach, but this time will face off against other alternatives. Finally, JonDo Live-DVD is a very interesting solution, which grew out of the multiplatform JonDonym, an internet surfing anonymiser with privacy and security in mind.

> **"The winner should be not only secure, but balanced and friendly even to less tech-savvy users."**

Anonymity and security tend to go hand in hand, so we expect an added benefit to be being able to nail down your system so it's secure from would-be hackers. We'll compare all these options with each other in different terms, and the winner should be not only secure, but generally balanced and friendly even to less tech-savvy users.

Availability

What does it take to get them running?

When you decide to try out an anonymous distro, you have to be aware that there's cost involved in using them, but it varies, so lets see what it takes to get our contenders up and running.

Tails is the most well-known distro, and we expected to download its ISO file and write it onto USB stick via some convenient tool like *dd* or front-end like *ImageWriter*. But the process with Tails turns out to be less straightforward, because the image has to be modified with the *isohybrid* utility. So, it went:

```
isohybrid tails-i386-1.2.3.iso -h 255 -s 63
dd if=tails-i386-1.2.3.iso of=/dev/sdc bs=16M
```

Where **/dev/sdc** is your flash drive. After that it works like a charm. The system boots into the live session just like a regular Debian-based distro.

No, it's not a blue SUSE lizard, it's Ubuntu Privacy Remix, which features this cool Protected Pangolin!

Whonix and Qubes OS are significantly harder to launch, and here is why: Whonix comes in the form of two *Virtualbox* machines, one for the Gateway and another for the Workstation. The idea behind this exquisite delivery is to isolate the environment you work in from the internet access point. So, the first thing to do is launch and configure the Whonix Gateway on one VM and then accessing it from another VM, where all work will be done. We didn't find any issues with it, but we have to admit that only advanced users will be able to deploy their workflow under Whonix.

After writing Qubes OS's ISO onto USB stick and booting from it, we discovered that there's no live session, only an installation mode. Qubes OS is based on a recent Fedora release and shares the same installer with it. But the system has some quite surprising system requirements: it wants you to provide it with 4GB of RAM, 32GB for the root partition and prefers built-in Intel video chip, as Nvidia or AMD have some issues in Qubes OS. The system needs such overstated resources due to its 'Security via isolation' approach, which we'll discuss later.

Finally, Ubuntu Privacy Remix and JonDo Live-DVD were extremely easy to launch. Their respective live sessions were fast and easy to use.

Verdict

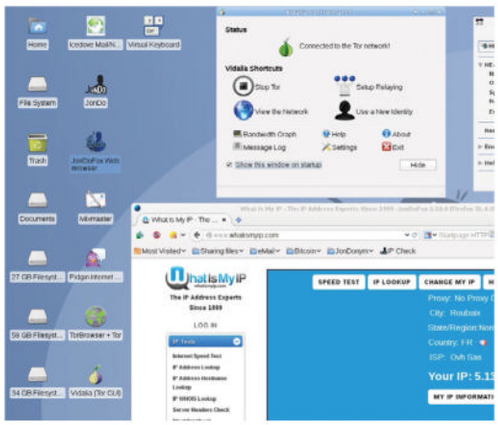
JonDo Live
★★★★★
Qubes OS
★★★☆☆
Ubuntu Privacy Remix
★★★★★
Tails
★★★☆☆
Whonix
★★☆☆☆

» *Easy access to anonymous live sessions wins out.*

Development state

Private and secure today, but how actively are they maintained?

This aspect is often overlooked, but it's vital as regular users will want to have an up-to-date and actively supported distro. The reality is that some secretive distros are abandoned by developers (such as Privatix) or left unmaintained for years (like Liberté). Some may think that it's a matter of new features and fixes, but let's not forget that abandoned Linux distros may have trouble running on modern hardware that has things like UEFI and Secure Boot.



JonDo Live-DVD has embarassingly frequent updates.

Tails is one of the best maintained security distros, with a very fast pace of development. New releases are rolled out every 2-4 months, which means Tails has had six releases during 2014 and went from v0.23 to 1.2.3 rapidly.

The Ubuntu Privacy Remix (UPR) developers, in comparison, don't seem to be in such a hurry, but keep development steady. UPR emerged in December 2008 and has been sticking with Ubuntu LTS releases. The current version is 12.04r1 (Protected Pangolin) which supports new hardware but is still a very lightweight distro.

Whonix is a relatively new project, which started in 2012 and has been very actively developed since then. Now at version 9.6, Whonix continues to get updates every few months.

Qubes OS is similar in that its birth also dates back to 2012, and the project has reached R2 release. Qubes OS's development is very active, with lots of well-documented alpha, beta and release candidate versions published every few months.

But that leaves us with the insanely speedy development record of JonDo Live-DVD. Somewhat staggeringly, JonDo boasts a changelog, which is updated every 5-10 days!

Verdict

JonDo Live
★★★★★
Qubes OS
★★★★★
Ubuntu Privacy Remix
★★★★☆
Tails
★★★★★
Whonix
★★★★★

» *All our participants are in rude health & updated often.*

Web surfing protection

How effectively do they shield you from web threats?

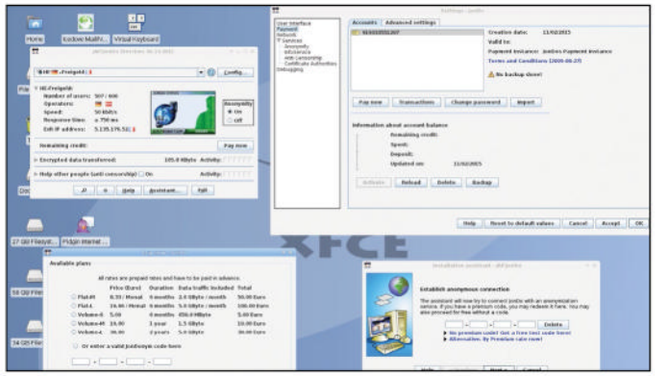
When you're accessing the internet, things become complicated and no one can guarantee that everything you access is 'absolutely' safe. But most of our distros try their best to offer the maximum possible protection.

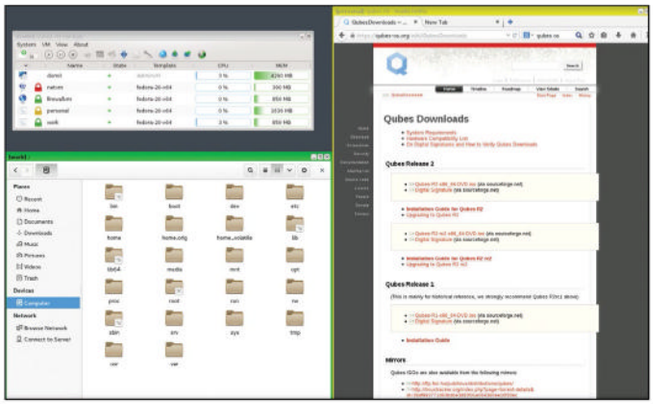
We also assume that while security is a top priority, users will still need to: access webmail; download and upload files; store passwords and sensitive data; and perform other common activities on the internet. Anonymity requires some compromises, such as lower download speeds and a harder password policy, but we also insist on a comfortable web browsing experience. But don't confuse greater security and hardened internet policies with good user data safety. This is different and something we'll cover later.

JonDo Live-DVD ★★★★★

JonDo provides network anonymity using the JonDo *IP changerv* (aka *JonDonym*), which is a Java Anon Proxy, similar to *T*or. JonDo enables web browsing (via a *Firefox*-based *JonDoBrowser*) with revocable pseudonymity and sends requests through a cascade and mixes the data streams of multiple users to further hide the data to outsiders.

It's worth noting that while the whole thing is open source, there are free and commercial plans. The free one can only use destination ports 80 and 443 that are used for the HTTP and HTTPS protocol (enough for web browsing and FTP). The premium service provides additional SOCKS proxies for extra anonymisation and a better connection speed. Generally, we find JonDo safer than Tor, because JonDo is much more centralised and can't include malicious nodes (which is possible in Tor).





Qubes OS ★★★★☆

Qubes OS implements another concept of virtualisation-based isolation. The system runs *Xen* hypervisor with multiple instances of an altered Fedora 20 virtualised on top of it. Qubes OS is divided into several 'domains' and applications can be run as virtual machines (AppVMs).

The standard way of anonymising network traffic is using Qubes TorVM, which connects to the internet and runs Tor. Other applications can be assigned to use this 'Torified' connection. The positive side is that an application doesn't need to be aware of Tor; it runs in regular mode without needing add-ons, and all IPv4 TCP and DNS traffic is routed by Tor. The downside is that you need to configure everything manually. We also noticed that this concept tends to restrain attacks and malware from spreading outside domain/AppVM, rather than prevent them.

Data safety

How safe is your sensitive data within each distro?

Though the most important feature of Tails is its 'amnesia' in live mode, you can install it to your hard drive and use it just like a regular Linux distro. Among all of the benefits of doing that, you'll note that your RAM will be wiped on reboot or shutdown, which will protect against forensic recovery techniques.

Ubuntu Privacy Remix shines when it comes to securing your data. The only way to store it is using the extended *TrueCrypt-Volumes*, which can be stored on removable USB media only (which, in turn, is mounted with a 'noexec' option). There's no way for your data to be left on drive partitions, not even unnoticed or by accident.

Whonix is much less amnesic than most of the others. On the Workstation side all data can be stored persistently, and its up to you how you keep it. You may want to encrypt and protect it with an extra password or store it on isolated location. But generally Whonix doesn't have a strong focus on data security.

Qubes OS is much better for data security, because it's possible to isolate sensitive data in a separate domain/AppVM without network access, but again the security level is heavily dependent on the skill of the user and how disciplined they are. JonDo Live-DVD offers a way for using persistent storage, and we found it to be quite user-friendly. It's ready to use LUKS encrypted USB sticks and drives and provides a special assistant to prepare your media.

Verdict

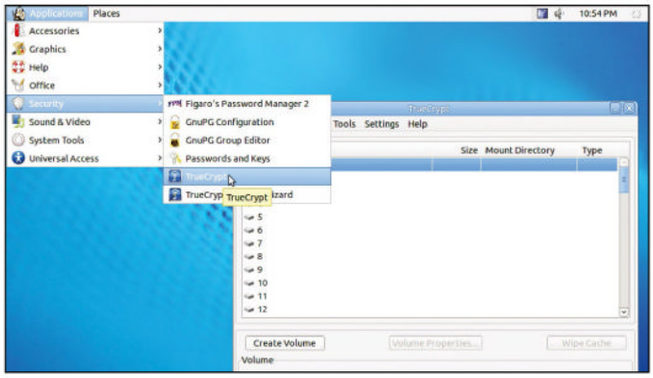
JonDo Live ★★★★☆
Qubes OS ★★★★☆
Ubuntu Privacy Remix ★★★★★
Tails ★★★★☆
Whonix ★★★☆☆

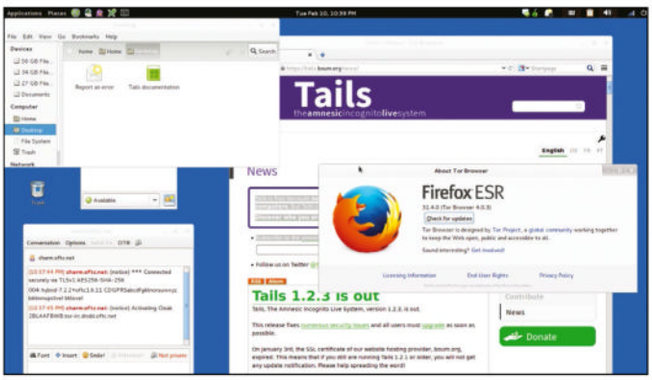
» *This time UPR offers the most security for your data.*

Ubuntu Privacy Remix ★☆☆☆☆

Sad but true, Ubuntu Privacy Remix (UPR) has no networking functionality at all. The system kernel is modified so that it ignores any network hardware, making UPR a perfectly isolated system, which can't be attacked via LAN, WLAN, Bluetooth and Infrared etc. So, there's no web browsing, no cookies, no trojans nor any data downloaded from the web, and no instant messaging or remote or cloud services. Almost all traces of network connectivity are wiped off the UPR, though some are still there. For example, *ifconfig* and *ifup/ifdown* commands are there, but they are virtually helpless, as network hardware is violently disabled.

So in this test UPR fails to be any use for web surfing, even if it is part of the design. If, however, you're paranoid and want a system that avoids being online entirely then UPR will be the right solution.





Tails ★★★☆☆

Tails includes top-notch networking features, and the most important one is Tor, which is an open network of anonymous servers that attempts to prevent your identification and traffic analysis.

This is accompanied by *Vidalia*, a front-end for easy set up, a preconfigured *Firefox* ESR-based web browser, which is equipped with a Tor Button, *HTTPS Everywhere, NoScript* and *AdBlock Plus* extensions.

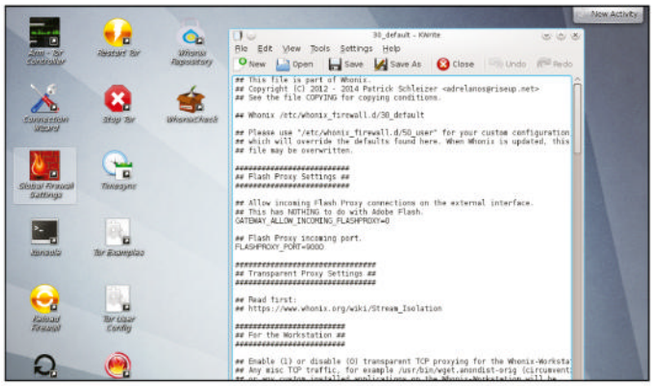
Tails many extras include *I2P* anonymising network, proxy and VPN front-ends, the *Florence* virtual keyboard, application isolation via *AppArmor*, *PWGen* for generating strong passwords and *KeePassX* for managing them, and *AirCrackNG* for wireless networks auditing etc.

Tor and I2P traffic are also divided, thanks to the dedicated *I2P Browser*, and *Pidgin* uses the more secure Off-the-Record (OTR) mode.

Whonix ★★★★★

Whonix also relies on Tor for network anonymity and shares many third-party tools with Tails. So lets point out the differences. Here the Tor client runs on Whonix-Gateway, which provides better protection against IP and location discovery on the Workstation.

The level of IP and DNS protocol leak protection is sometimes the same, but in Tails there's a possibility of misconfiguration, which can lead to IP leak and in Whonix this doesn't exist. Even if the workstation is compromised (eg by someone getting root access), it would still be impossible to find out the real IP. Isolating the proxy server within a standalone VM (or maybe a physical PC) works great. Whonix also makes use of 'entry guards' in Tor (randomising endpoints), which is something that is missing in Tails out of the box.



Performance

How snappily do they run?

More recent Tails uses 3.16.7 kernel and loads into Gnome Shell 3.4 in fallback mode by default. The desktop is very lightweight; nearly as fast as classic Gnome 2 in previous Tails releases, but official system requirements say it needs at least 1GB of RAM to work smoothly, which we think is a bit much.

Ubuntu Privacy Remix was updated to use the Ubuntu 12.04 LTS package base and thus has numerous backports and modern features, yet it remains very easy on resources. UPR uses a classic Gnome 2 desktop, which loads in a couple of seconds. We'd suggest that 512MB of RAM is enough, though UPR can make use of the larger RAM volume as the system implements 'ramzswap' to store swap file in RAM.

JonDo Live-DVD can boot even on very old CPUs, and its XFCE desktop is very fast. However, you'll need 1GB RAM to work smoothly with the Java-based JonDo app and the web browsers.

Whonix is different, again, because you need a host capable of running two *Virtualbox* guest machines at a time. Your host OS and configuration is down to you, but you're going to need at least 4GB of RAM, a spare 12GB of hard drive space. However, the SSD and CPU with hardware virtualisation support are both very welcome.

For Qubes OS you'll need an even beefier machine: a 64-bit CPU, 4GB of RAM and at least 32GB for root partition. Qubes OS is, therefore, the most demanding choice.

Verdict

JonDo Live ★★★★★
Qubes OS ★★☆☆☆
Ubuntu Privacy Remix ★★★★★
Tails ★★★★☆
Whonix ★★☆☆☆

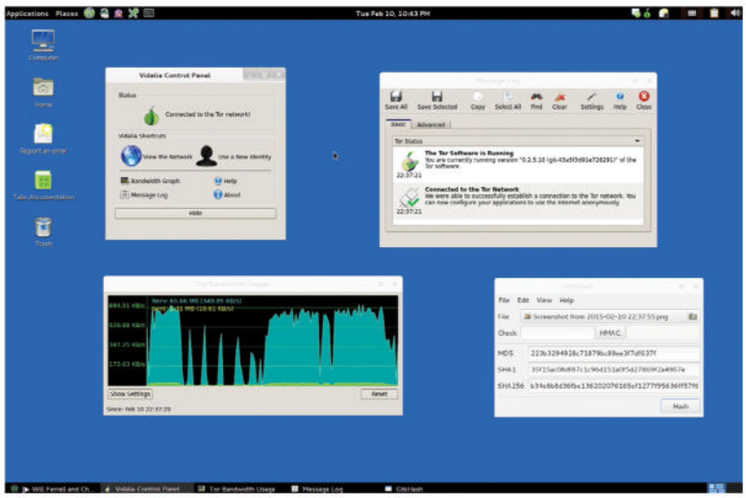
» *Both Tails and JonDo are modest on resources.*

Desktop usability

Can you be anonymous and still enjoy a feature-rich desktop?

Though Tails is 'amnesic', it includes an installer, which can create a persistent partition either on the same USB stick you boot from, or another USB storage device. This makes Tails a pleasant experience for permanent work in live mode. It also includes a vast selection of software, from *LibreOffice* and *Gimp* to *Audacity* and *Sound Juicer*.

JonDo Live-DVD also has a very usable Xfce live desktop, which is packed with all the essential desktop software, but its main advantage is that you can install both the JonDo IP changer and *JonDoFox* browser on any Linux distro. This is a huge bonus, because you can stay with your already-configured Linux box and seamlessly turn anonymous.



The desktop in Tails will be familiar and easy to use for Gnome users.

Ubuntu Privacy Remix (UPR) includes only basic Gnome 2 accessories and very few desktop apps (*Scribus* and *LibreOffice* are the most noticeable examples). The desktop experience in UPR is poor, so much so that even extracting screenshots turned out to be a problem. Worst of all, UPR is made deliberately non-manipulative, so nothing can be fixed from a desktop perspective.

Both Whonix guest machines use the KDE desktop on top of Debian. We really love KDE, but it seems to be excessive on the Gateway side. But the Workstation experience turned out to be very comfortable. Aside from some minor slowdowns and restrictions, because of it being a virtualised and firewalled system, Whonix Workstation can be used as a fully featured desktop.

Qubes OS is an entirely different experience: it's easy to install but can work very slowly later down the line. Its KDE desktop is intuitive, but interaction between domains requires extra skill. For example, copying and sharing files from one domain or AppVM to another has its own logic and clipboard usage is limited.

Verdict

JonDo Live ★★★★★
Qubes OS ★★☆☆☆
Ubuntu Privacy Remix ★★★★★
Tails ★★★★☆
Whonix ★★☆☆☆

» *The best offer familiar software and anonymity tools.*

Documentation and support

Is there any help and where do you get answers to questions?

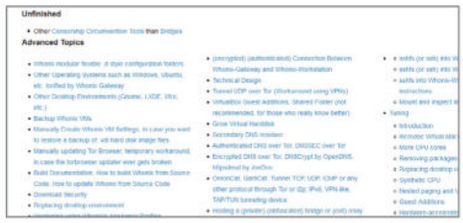
Good wiki pages, FAQs and other helpful documentation are important for any software. This is certainly the case with anonymous distros that can be frustrating even for people familiar with Linux.

Tails offers in-depth end-user documentation with general information, first steps, commonly asked questions and detailed explanations for almost all aspects, even those not related to Tails directly, but it's all essential if you want to study the basics of privacy and encryption. There's even a chat room and a 'request a feature' form.

Ubuntu Privacy Remix has a neat and compact website, yet there isn't that much materials, but the quantity of UPR resources corresponds with its feature set. You can find some helpful how-to guides, such as instructions for creating a personal UPR build (with a custom software set).

Nearly all Whonix documentation resides in a dedicated and detailed wiki portal. We found it to be very comprehensive and more in-depth than the resources Tails supplies – Whonix has more articles, more support options and a very active forum.

The Qubes OS project also has a wiki portal with essential and advanced articles. The OS architecture is explained in detail and there's an FAQ, tutorial slides and user documentation. Qubes OS has many extra features, such as running non-Linux AppVMs, and this is covered in a detailed manual.



The Whonix help section is huge and scrollable. Even advanced and in-depth topics are covered.

There's also a helpful developer's corner, which provides all you need to develop custom solutions.

JonDo has help topics, an FAQ, tutorials, a wiki portal and a forum. Though it looks complete, a thorough review shows many weaknesses. The FAQ is brief, and the wiki is very small. Very few topics are actually covered, which is disappointing.

Verdict

JonDo Live ★★★☆☆
Qubes OS ★★★★☆
Ubuntu Privacy Remix ★★★☆☆
Tails ★★★★☆
Whonix ★★★★★

» *Whonix sneaks in front of Tails for it's level of support.*

Privacy distributions

The verdict

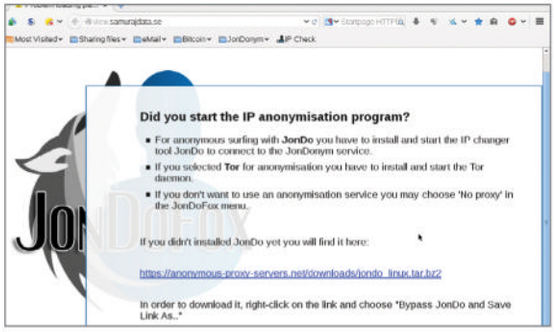
Java Anon Proxy was a 2007 startup, backed by solid research work of many years. Here, we witness the fruit of that work as JonDo Live-DVD clearly outperforms the former king of anonymous web access: Tails. Both projects are premiere quality, however, with balanced features and active development.

It's hard to say whether Tor provides perfect anonymity or not, but it's technically possible to single out a Tor user either through a compromised node or by matching traffic and user behaviour with other details, or even by correlation-timing attacks. On the other hand, JonDo node selection is less random than Tor, and we're not completely sure to what extent you can trust it. Both solutions slow the internet speeds greatly, and the JonDo proxy cascade seems to be even slower than Tor node chain. But connection speed is not top priority, because you're getting well-tested and supported anonymity.

Other participants clearly defined the cost they charge for advanced privacy and security. Whonix forces you to use virtual machine, which is always slower than a host computer, has little or no 3D support and takes extra time and skills to install it for the first time. But once you've done that Whonix can be configured to your need just like any other Debian-based distro.

It would also appear that Qubes OS will only work on quite high specified hardware, but even then it runs even slower than virtualised Whonix. Qubes OS does, however, deliver good anonymity, but its main purpose is to isolate different segments so that one segment can't bring down the others if compromised. You will also have to learn how different software domains communicate with each other.

The approach of Ubuntu Privacy Remix is unconventional, but it's also about anonymity although dealing with it very differently to the others. The project's website shows how you can create your own UPR spin-off and use it as a perfectly isolated system, which leaves no traces on a computer. UPR can also detect virtual environments and eject its ISO from its settings, but all this is solely local, without any connectivity with the outside world.



› ***JonDoFox*** **won't let you surf the internet unless your start Java Anon Proxy.**

> **"JonDo Live-DVD clearly outperforms the former king of anonymous web access: Tails."**

1st JonDo Live-DVD ★★★★★

Web: http://bit.ly/JonDoLive-DVD **Licence:** BSD **Version:** 0.9.71.2

» Fast, portable, effective and easy to use for anonymous web surfing.

2nd Tails ★★★★★

Web: https://tails.boum.org **Licence:** GNU GPLv3 **Version:** 1.2.3

» Balanced for 'mostly' safe internet access. Also a friendly way to try Tor.

3rd Whonix ★★★★☆

Web: www.whonix.org **Licence:** Mainly GNU GPL **Version:** 9.6

» Very usable and super-secure, but the hardware specs are quite high.

4th Qubes OS ★★★☆☆

Web: https://qubes-os.org **Licence:** Mainly GNU GPL **Version:** R2

» Very secure, but like riding a bumpy narrow road between concrete walls.

5th UPR ★★☆☆☆

Web: www.privacy-cd.org **Licence:** Mainly GNU GPL **Version:** 12.04r1

» Consider it as a special-purpose distro for securing sensitive data.

Over to you...

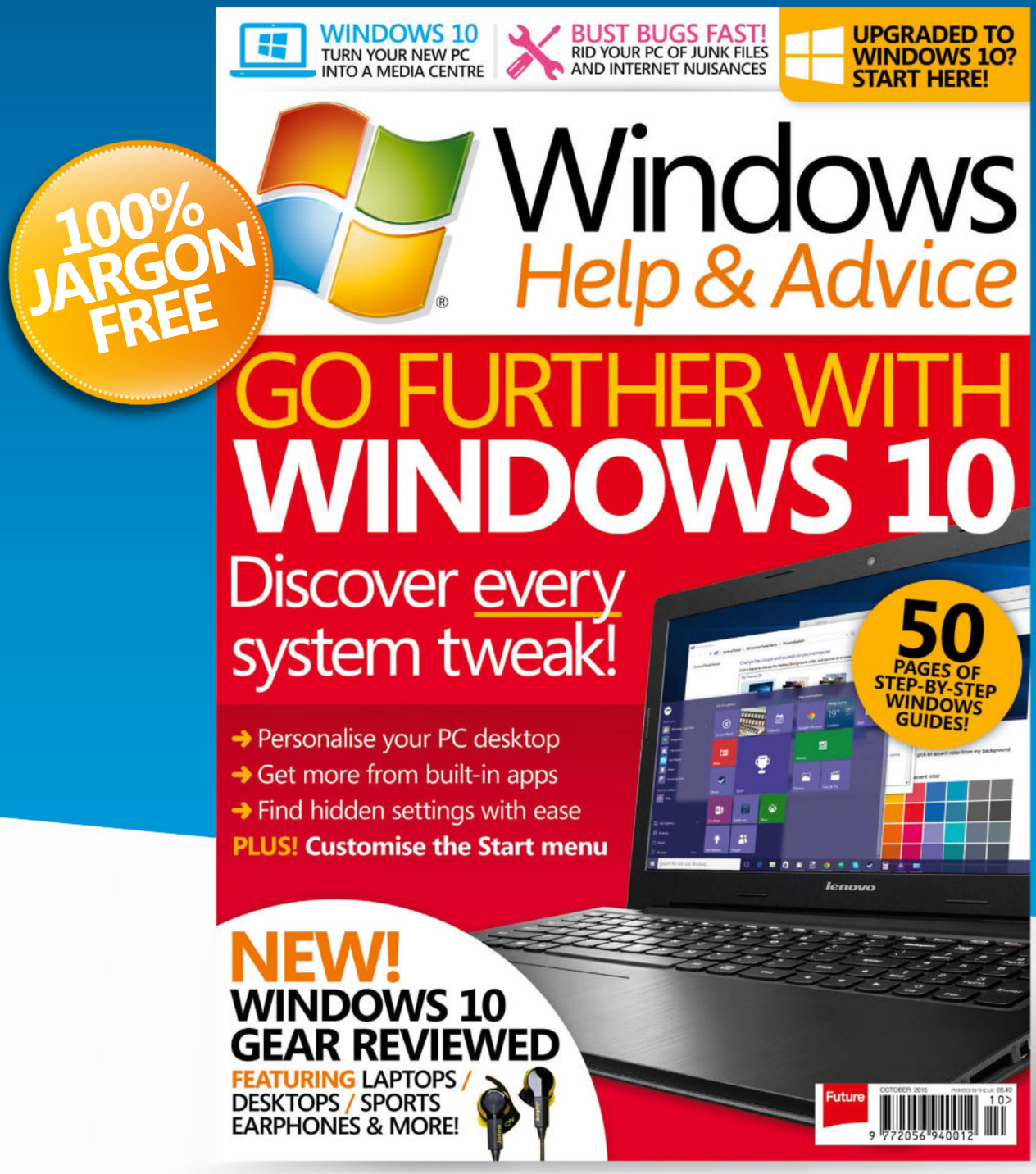
Tell Linux Format about your anonymous web surfing experiences at **lxf.letters@futurenet.com**. What's your favoured distro for privacy?

Also consider...

Many people share the illusion that they can be invisible and unreachable under the Tor network. In fact, this is only true until a user breaks a law or somehow attracts attention from intelligence services. Please use anonymity only for peaceful purposes and at your own risk. On the other hand, you have a right to keep your data away from third-parties, so why not take some measures?

The choice of anonymising distros is larger than what we've covered. Privatix and Liberté both haven't received any updates for a long time, but they are still usable and ready for web surfing on most machines. There are other projects too, such as IprediaOS, Polippix and Mandragora that didn't fit in this Roundup but are worth considering. In fact, it's not too hard to turn your existing Linux install into a digital fortress. Almost all tools for anonymity on Linux are open source, including Tor front-ends, extensions and encryption methods.

THE EASY WAY TO LEARN WINDOWS



AVAILABLE IN STORE AND ONLINE

www.myfavouritemagazines.co.uk

Tor: Set up a Wi-Fi hotspot

Worried about security? Configure a Raspberry Pi as an access point that routes all traffic over the anonymous Tor network.

Do you use *Tor* to prevent big brother from tracking you online? Although it is pretty straightforward to use, it can be quite a hassle to configure *Tor* on all your Internet-enabled devices. You can save yourself a lot of hassle by using a Raspberry Pi as an anonymised wireless access point. The Pi will dole out an IP address and any device that's connected to it will be able to access the Internet via the *Tor* network. To get this project up and running, you'll need a Raspberry Pi along with an SD card with the Raspbian distro. If you haven't done this before, follow the walkthrough to get Raspbian up and running. You'll also need an Ethernet cable. Hook one end into the Pi's Ethernet port and the other into your wireless router. This is how the Pi will connect to the Internet. You'll also need a USB Wi-Fi adaptor that's compatible with the Raspberry Pi. If you haven't got one yet, check the list of compatible adapters that are known to work on the Pi (**http://elinux.org/RPi_USB_Wi-Fi_Adapters**).

Access Point Pi

Once you've setup the Pi, you can configure the Pi from a remote machine via SSH. For the rest of the tutorial, we'll assume the IP address of your Pi is **192.168.2.100**. Fire up a terminal that's connected to the same router as the Pi and enter

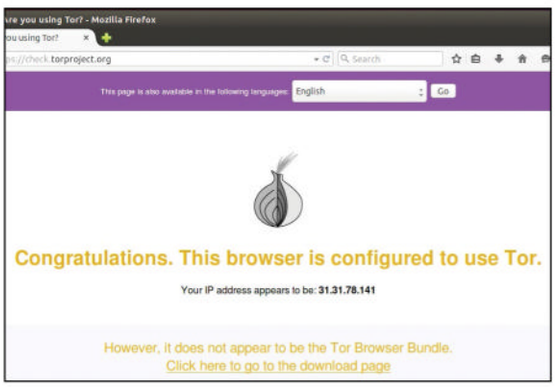
```
ssh pi@192.168.2.100
```

to connect to it. After authenticating yourself into the Pi, use

```
iwconfig
```

to make sure the wireless adaptor is recognised by the device. Now refresh its package list with

```
sudo apt-get update
```



It takes more than *Tor* to stay anonymous. Make sure you read the documentation on the Tor Project's website.

and install the software that will make it act as an access point with:

```
sudo apt-get install hostapd isc-dhcp-server
```

When it's installed, it's time to set it up. Begin by editing the **/etc/dhcp/dhcpd.conf** file that controls the DHCP and automatically assigns IP addresses to all connected devices. Open it in the *nano* text editor with

```
sudo nano /etc/dhcp/dhcpd.conf
```

and comment out the following two lines by adding a # in front of them, so that they read:

```
#option domain-name "example.org";
#option domain-name-servers ns1.example.org, ns2.example.org;
```

In the same file, scroll down and uncomment the word authoritative; by removing the **#** in front.

Then scroll down to the end of the file and add the following lines:

```
subnet 192.168.12.0 netmask 255.255.255.0 {
range 192.168.12.5 192.168.12.50;
option broadcast-address 192.168.12.255;
option routers 192.168.12.1;
default-lease-time 600;
max-lease-time 7200;
option domain-name "local";
option domain-name-servers 8.8.8.8, 8.8.4.4;
}
```

In these lines we define the IP address of our Pi access point (192.168.12.1), the range of the IP addresses it'll hand out to connected devices (from 192.168.12.5 to 192.168.12.50) as well as the address of the domain name servers (8.8.8.8 and 8.8.4.4). You can change any of these values as per your preference. Save the file (Ctrl+X) once you're done.

Setting up a static IP

We'll now edit the **/etc/default/isc-dhcp-server** to specify the interfaces that our new DHCP server should listen to. Open the file and scroll down to the line that reads **INTERFACES=""**. Insert **wlan0** between the quotes so that it now reads **INTERFACES="wlan0"**, and save the file.

Now we'll setup the wireless adaptor (wlan0) and give it a static IP address. First, deactivate the wireless adaptor with:

```
sudo ifdown wlan0
```

command and then open the **/etc/network/interfaces** file. In the file, comment out every existing entry associated with wlan0, such as:

```
# iface wlan0 inet manual
```

Quick tip

If you get Locale errors when connected to the Pi remotely, make sure you don't forward your locale by editing **/etc/ssh/ssh_config** and commenting out the SendEnv LANG LC_* line.

```
# wpa-roam /etc/wpa_supplicant/wpa_supplicant.conf
# iface default inet dhcp
```

Then add the following lines below the line that reads **allow-hotplug wlan0** to set the static IP address for the new access point:

```
iface wlan0 inet static
address 192.168.12.1
netmask 255.255.255.0
```

Save the file and activate the interface with

```
sudo ifconfig wlan0 192.168.12.1
```

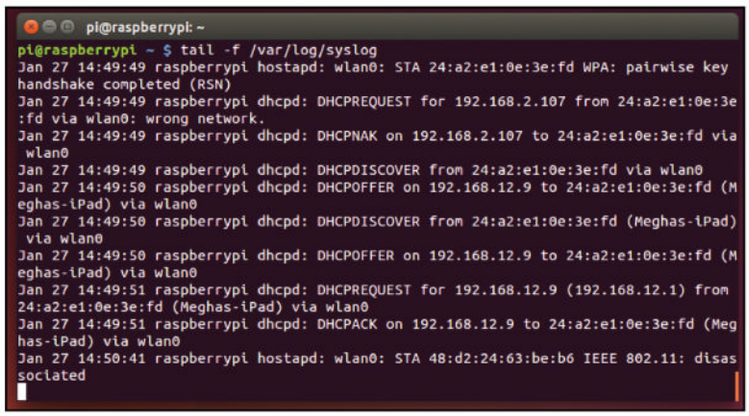
Quick tip

Use the **tail -f /var/log/syslog** command to keep an eye on all system messages. This might come in handy if you are unable to connect to the Pi hotspot.

Make your point

Now that we've defined the wireless access point it's time to configure it. Create a new file called **/etc/hostapd/hostapd.conf** with the following contents:

```
interface=wlan0
ssid=TorSpot
hw_mode=g
channel=6
macaddr_acl=0
auth_algs=1
ignore_broadcast_ssid=0
wpa=2
wpa_passphrase=$$Your_Passphrase$$
wpa_key_mgmt=WPA-PSK
wpa_pairwise=TKIP
rsn_pairwise=CCMP
```



Use the tail -f /var/log/syslog common to keep an eye on the devices connected to your *Tor* hotspot.

We've setup a password-protected network called TorSpot. You can specify a different name for the access point by specifying it in the **ssid=** string. Also change the **wpa_passphrase=** string to specify a custom password. You'll need to enter this password to authenticate yourself to the Pi's access point.

Next up, we'll tell the Pi where to find this configuration file by pointing to it in the **/etc/default/hostapd** file. Open the file, find the commented out line that reads **#DAEMON_CONF=""** and uncomment and edit it to read **DAEMON_CONF="/etc/hostapd/hostapd.conf"**.

NAT setup

We now need to set up NAT to allow multiple clients to connect to the Pi's access point and route all their traffic through the single Ethernet IP. Edit the **/etc/sysctl.conf** file and at the bottom add the following line:

```
net.ipv4.ip_forward=1
```

Save the file and then enter

```
sudo sh -c "echo 1 > /proc/sys/net/ipv4/ip_forward"
```

to activate the forwarding. You'll now have to specify the routing rules that will connect the Ethernet port (eth0) that's connected to the internet and the Wi-Fi access point (wlan0) which is exposed to the devices within your network:

```
sudo iptables -t nat -A POSTROUTING -o eth0 -j MASQUERADE
sudo iptables -A FORWARD -i eth0 -o wlan0 -m state --state RELATED,ESTABLISHED -j ACCEPT
sudo iptables -A FORWARD -i wlan0 -o eth0 -j ACCEPT
```

By default, these rules will be flushed when you restart the Pi. To make them permanent, first run:

```
sudo sh -c "iptables-save > /etc/iptables.ipv4.nat"
```

Then edit the **/etc/network/interfaces** file, scroll down to the very end and add

```
up iptables-restore < /etc/iptables.ipv4.nat
```

what this does is loads the rules when the devices are activated on boot.

Your Pi access point is now all set. To test it restart the DHCP server with

```
sudo service isc-dhcp-server restart
```

and manually enable the access point with our configuration

Your own hostapd

Sometimes even though a wireless adaptor works out of the box on the Raspberry Pi, it might throw errors when it's asked to serve as an access point. This is especially true of cards that use Realtek chipsets, like the one we've used – MicroNext MN-WD152B – which uses the RTL8192CU chipset. While it works right off the bat for browsing the web, it doesn't work with the *hostapd* client in Raspbian's repository. It turns out Realtek has its own version of *hostapd* client which you'll have to use in case you are in the same predicament as us.

To download the file, head to Realtek's download section (**http://bit.ly/RealtekWiFiDrivers**) and select your chipset from the ones listed. This takes you to a page that lists the drivers for your chipsets. From this page grab the driver for Linux, which will download a compressed zip file with a long-windy name. In our case this was called **RTL8188C_8192C_USB_linux_v4.0.2_9000.20130911.zip**. We'll just refer to it as **driver.zip**.

Copy this file to the Raspberry Pi using *scp* using something like:

```
scp driver.zip  pi@192.168.2.100:/home/pi
```

This copies the file to the Pi's home directory. Now extract the file with

```
unzip driver.zip
```

and **cd** into the **wpa_supplicant_hostapd** directory. It'll list several compressed tarballs. Use the **tar zxvf** command to extract the file beginning with **wpa_supplicant_hostapd**. Now **cd** into the **hostapd** directory under the extract directory. This directory has a file named Makefile. Open it in a text editor and replace the

```
CFLAGS = -MMD -O2 -Wall -g
```

line towards the top of the file with

```
CFLAGS=-MMD -Os -Wall -g
```

Save the file and enter *make* to compile the *hostapd* client. It'll take quite some time and when it's complete it'll replace the *hostapd* binary in this directory.

Before using this new version, move out the old version with:

```
sudo mv /usr/sbin/hostapd /usr/sbin/hostapd.orig
```

Then copy over the newly compiled version with the following:

```
sudo cp hostapd /usr/sbin/
```

And give it the right permissions with:

```
sudo chmod 755 /usr/sbin/hostapd
```

You should now be able to get your access point online without any issues.

with the following command [Read the *'Your Own Hostapd' box, p72*, if you get an unknown driver error]:

```
sudo /usr/sbin/hostapd /etc/hostapd/hostapd.conf
```

If everything goes well, the wireless access point (TorSpot) is listed in the list of available Wi-Fi hotspots. You can connect to it from another computer or a smartphone and authenticate using the password you specified in the **hostapd.conf** file. When connected, you should be able to browse the Internet normally.

Once you have tested the new access point, let's cement the settings so that they are activated as soon as the Pi boots up. Start the hostapd and DHCP services with the

```
sudo service hostapd start
```

and

```
sudo service isc-dhcp-server start
```

commands and then update the init scripts with

```
sudo update-rc.d hostapd enable
```

and

```
sudo update-rc.d isc-dhcp-server enable
```

Now restart the Pi with

```
sudo shutdown -r now
```

When the Pi is back up again, you'll be able to connect to the new access point and browse normally.

Torify access

Your Raspberry Pi is now fully functional as a wireless hotspot. However, the data is still not anonymised. So let's add *Tor* to the mix. SSH back into the Pi and install *Tor* with

```
sudo apt-get install tor
```

When it's installed, edit *Tor*'s config file **/etc/tor/torrc** and add the following at the top:

```
Log notice file /var/log/tor/notices.log
VirtualAddrNetwork 10.192.0.0/10
AutomapHostsSuffixes .onion,.exit
AutomapHostsOnResolve 1
TransPort 9040
TransListenAddress 192.168.12.1
DNSPort 53
DNSListenAddress 192.168.12.1
```

These settings inform *Tor* about the IP address of our access point and asks that it anonymises any traffic that flows over it. Next up, we'll change the routing tables so that connections via the Wi-Fi adaptor (wlan0) are routed through *Tor*. First, flush the existing redirection and NAT rules with the

```
sudo iptables -F
```

command go on to

```
sudo iptables -t nat -F
```

command. Since, we'll still want to be able to SSH into the Pi, we'll add an exception for SSH's Port 22 with:

```
sudo iptables -t nat -A PREROUTING -i wlan0 -p tcp --dport 22 -j REDIRECT --to-ports 22
```

We'll now add two rules. The first is a passthrough rule for DNS lookups and the second directs all TCP traffic to *Tor*'s port 9040:

```
sudo iptables -t nat -A PREROUTING -i wlan0 -p udp --dport 53 -j REDIRECT --to-ports 53
sudo iptables -t nat -A PREROUTING -i wlan0 -p tcp --syn -j REDIRECT --to-ports 9040
```

Like before, these rules won't be carried on to the next session. To load them on reboot, all you have to do is save them to the NAT save file like before with

```
sudo sh -c "iptables-save > /etc/iptables.ipv4.nat"
```

In the previous section, we've already configured the **/etc/network/interfaces** file to load the contents of this file when the interfaces are activated.

You can now enable the Tor service with

```
sudo service tor start
```

and update the relevant boot scripts with

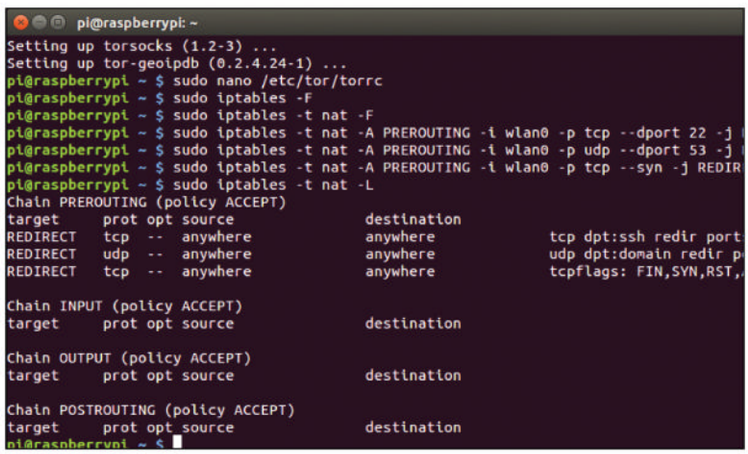
```
sudo update-rc.d tor enable.
```

That's it. Now restart the Pi. When it's back up again, you'll be able to connect to the Pi hotspot, TorSpot, as before. However, unlike as before all your traffic will now be routed through the *Tor* network.

You can verify that this is happening by heading to check **https://torproject.org** from any device that's connected to TorSpot. The page will also list your IP address which will not be that of your ISP. Visit this page from another device connected to TorSpot and it'll show a different address. Congratulations, you can now anonymously browse the web on all your devices!

Quick tip

We've used Google's DNS service in this tutorial, but you can use another service like OpenDNS or your ISPs DNS servers by pointing to them in the **/etc/dhcp/dhcpd.conf** file.



Verify the traffic redirection rules with the sudo iptables -t nat -L command.

Tor-in-a-box options

If you find this tutorial too cumbersome, or want to set up something for a non-technical friend or relative, there are several ready-made hardware solutions that can anonymise all their web traffic in a similar fashion.

There's the OnionPi Pack from AdaFruit (**http://bit.ly/AdaOnionPi**) which includes a Raspberry Pi B+ and a compatible USB Wi-Fi adaptor along with a case for the Pi, cables, SD card and everything else you need to setup your Torrified Wi-Fi hitspot. The bundle costs $80. However, you'll still have to follow the instructions and set it all up yourself.

If you'd rather have something more plug and play, there's the SafePlug from the guys who bought us PogoPlug. It's a $49 device that plugs into your wireless router and once activated routes all traffic over the Tor network. A neater and smaller alternative is the Anonabox (**www.anonabox.com**). It initially launched on Kickstarter but after its funding was suspended it relaunched on Indiegogo. Here it was listed at $51 and surpassed its funding target in early January 2015 and will begin shipping in February 2015. Anonabox is a router that you can directly connect to via Wi-Fi or Ethernet.

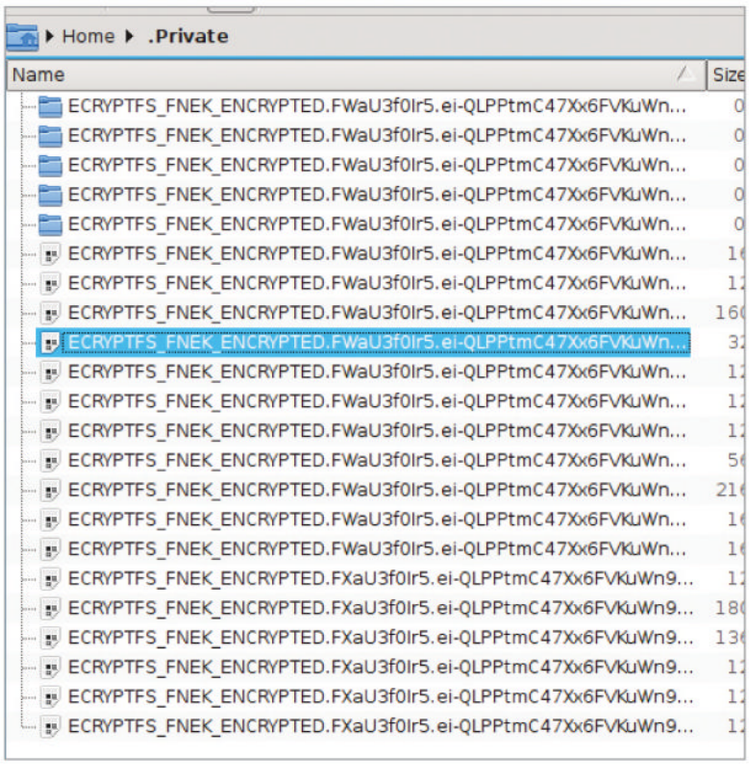
Another router-based option is Portal which stands for Personal Onion Router To Assure Liberty. The project produces a pre-built software image for several TP-Link routers. You can simply flash the Portal firmware image onto these router following the instructions on the project's website (**https://github.com/grugq/portal**).

Encryption: full drive protection

Keep your files safe from prying eyes, even other users of your computer, by creating a stacked filesystem with ecryptfs.

Last year, when everyone was interested in privacy in the aftermath of Edward Snowden's revelations, Linux Format magazine looked at using *cryptsetup* to encrypt whole disk partitions with the Linux kernel's *dm-crypt* facilities, but there are other encryption systems available. There are several ways of encrypting data on your computer.

The method we looked at before encrypted a whole block device, usually a disk partition. This is good for whole system encryption, but makes everything available once the system is booted. There was also *TrueCrypt*, which works with either whole devices or virtual disks (a large file that acts like a disk). Sadly the project was abandoned in 2014, and although there have been a couple of forks many people are still using the 7.1a version (the final, neutered 7.2 version only allows viewing of *TrueCrypt* volumes). Another alternative is for the filesystem to handle the encryption, as ZFS does on Sun systems, but none of the main Linux filesystems provide encryption themselves.



This is how your files look after encryption, and their contents are equally unintelligible.

Introducing ecryptfs

The next option, and the one we are concerned with today, is what is called a stacked filesystem, where you mount one filesystem on top of another, and this is what *ecryptfs* uses (*cryptsetup*, which we've covered before, uses stacked block devices, below the filesystem).

Because *ecryptfs* works on top of the normal filesystem, it's not restricted to entire disk partitions, it can be used to encrypt individual directories. This is the method Ubuntu uses to provide encrypted home directories if you choose that option during installation. It is easiest to explain with an example. The *ecryptfs* filesystem itself is contained in the Linux kernel, but you will need to install the **ecryptfs-utils** package for the tools to work with it. Create two directories called **crypt** and **plain**, then you can create an encrypted directory with this command:

```
sudo mount.ecryptfs crypt plain
```

You will be asked a number of questions, obviously you should choose a password that is both secure and memorable (or store it somewhere safe). Most of the rest can be left as the defaults with the possible exception of Enable Filename Encryption that you may want to set to yes.
Now copy some files to plain then look in **crypt**. You will see the same filenames if you didn't enable filename encryption, otherwise you will see encrypted names. Either way, the contents will be encrypted; try viewing one of the files. Now unmount it with:

```
sudo umount plain
```

The readable versions of the files have disappeared, leaving only the encrypted versions. Run the above **mount** command and the contents of **plain** will reappear. This method of mounting is cumbersome but it illustrates how *ecryptfs* functions. The filesystem you mounted on **plain** is virtual, it exists only in memory, the only data written to disk are the encrypted files in **crypt**. Once you unmount the **plain** version your data is protected, and cannot be read again until you mount it, which requires your password.

Convenient encryption

There is, of course, a more convenient way of setting up an encrypted directory for a user that doesn't require **sudo** or answering questions - run this as your normal user:

```
ecryptfs-setup-private
```

The command will ask for your login password and then a passphrase for the encrypted directory. The former is used to lock the latter, which you can leave that blank and have *ecryptfs* generate a secure passphrase automatically. This

Pros and cons of ecryptfs

Ecryptfs has a number of advantages over *LUKS/dm-crypt*:

» **Back up to cloud** As the encryption is at file level, you can backup your **.Private** directory to a cloud service or external drive without worrying about your data being accessible to others. Just make sure you backup **.cryptfs** and your passphrase some separate and secure.

» **Multi-user security** *Ecryptfs* can encrypt directories separately for each user.

» **Directory** *Ecryptfs* can also be used on system directories and swap, with a suitable fstab entry, but it will prompt for a passphrase.

» **Login to read** A user's data is only available when the user is logged in, and even then *ecryptfs* defaults to making it only readable by that user (and root, of course).

There are, however, some disadvantages too:

» **Many files** It is slower dealing with directories containing many files, although this can be mitigated (at the expense of security) by having *ecryptfs* not encrypt filenames.

» **Large files** Because each file is encrypted separately, the files all increase in size, which can be significant with a large number of small files, like an email or browser cache.

» **Not cross-platform** *Ecryptfs* is Linux only, using features of the kernel, which won't be a problem for everyone. As far as we are aware, there's no reliable way to read Windows files.

creates three directories: **.Private** contains your encrypted data, **Private** is the mountpoint for the decrypted contents and **.ecryptfs** contains files that are used to mount your directory. As the passphrase itself is encrypted, you should make a copy and store it somewhere secure, such as a USB key nowhere near your computer:

```
ecryptfs-unwrap-passphrase ~/.ecryptfs/wrapped-passphrase
>/somewhere/safe/ecryptfs_passphrase
```

Now you can mount and unmount your private data with these commands, or use the desktop icon it provides.

```
ecryptfs-mount-private
ecryptfs-umount-private
```

This creates a single, encrypted directory in your **home**, but what if you want more? Let's say you want your Documents and Accounts directories encrypted but see no point in encrypting Photos or Music (why waste time decrypting large files that hold nothing private). The easy answer is to move the directories into **Private** and create symbolic links back to their original locations, like this:

```
mv Documents Private
ln -s Private/Documents Documents
```

Make sure **Private** is mounted when you do this, then your files will only be available when the *ecryptfs* filesystem is mounted, otherwise it will just show up as a broken link.

Automatic mounting

You give your login password to unlock the *ecryptfs* passphrase to mount the filesystem (you can use the -w option to **ecryptfs-setup-private** if you want to use an independent password) so you may be asking why when you've already just given a password to login, you need to give it again to mount your private files? This is a valid question, if you know it once, I'm sure you can remember it again a few seconds later. If you prefer, you can have your **Private** directory automatically mounted when you login (and unmounted when you logout), thanks to the magic of PAM. As root, insert this line into **/etc/pam.d/common-auth**:

```
auth    required   pam_ecryptfs.so unwrap
```

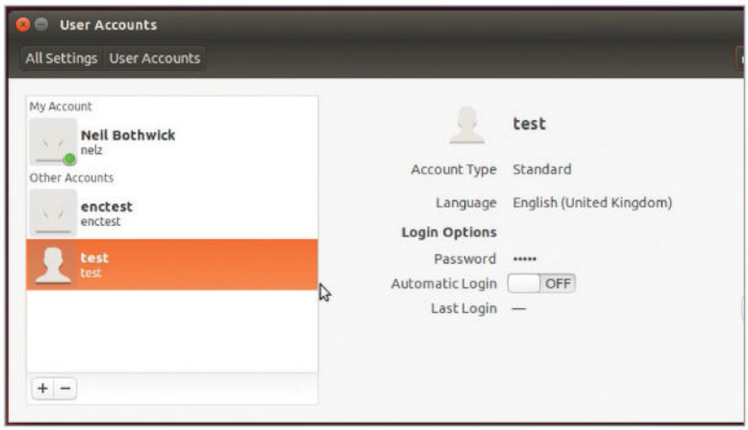
and this one into **/etc/pam.d/common-session**:

```
session optional   pam_ecryptfs.so unwrap
```

Now PAM will mount your *ecryptfs* **home** directory when you login. This will not happen if you have auto-login enabled, otherwise you would have no security at all.

Encrypted $HOME

If all of this looks a little familiar, that is probably because you have used the encrypted **home** directory feature in Ubuntu, which also uses *ecryptfs*. But this a standard kernel feature not restricted to one distro (ChromeOS also uses *ecryptfs* behind the scenes). Ubuntu doesn't just set up a **Private** directory when you install it, but it encrypts your entire **home** directory. So the simplest way to get a fully encrypted **home** directory may seem to be to install Ubuntu and choose that option. There are a couple of reasons you may not want to do this: you may use a different distro or you may already use Ubuntu but don't want to start again with a new installation.



If your distro does not permit root login, like Ubuntu, create a spare user with admin rights when encrypting your home directory.

There's a single command that will convert your entire **home** directory to *ecryptfs*, but there are a couple of caveats. You must have no files in use in the **home** directory, which means that the user mustn't be logged in, and you need free space of up to 2.5 times the current size of your **home** directory for the conversion process (mainly because encrypted and unencrypted copies of your files are stored until the job is done). So log out and log in as another user with admin rights then run:

```
sudo ecryptfs-migrate-home --user <yourusername>
```

After the process completes, you must log in as that user before rebooting, to complete the setup and make sure everything is working. Once that is done and you have verified that your files are there and readable, you can delete the original unencrypted files that are still in **/home/user.some_random_string**. Be aware that deleting that directory does not remove all of your unencrypted data from your hard drive, only the directory table. To be fully secure, you should overwrite all unused space with random data.

```
dd if=/dev/urandom of=somefile bs=4k
rm somefile
```

This creates a file of random data that fills the drive and then deletes it to return the space to you.

Whether you use **ecryptfs-setup-private** or **ecryptfs-migrate-home**, you should use **ecryptfs-unwrap-passphrase** to save the passphrase to a safe place. If you don't keep a copy of your passphrase, you won't be able to access your data if the **.ecryptfs** directory is lost or damaged.

Encrypt drives with zuluCrypt

Insulate your data another way

While you can control access to the data on your computer using user accounts and file permissions, they aren't enough to prevent a determined intruder from gaining access to your private files. The only reliable way to keep your personal data to yourself is to encrypt it. Sure, working with encrypted data is an involved process, but it'll go a long way in reinforcing your security and insulating your data. zuluCrypt is a graphical encryption app that has an intuitive easy to follow interface. Using the app you can create an encrypted disk within a file, a partition and even USB disks. It can also encrypt individual files with GPG.

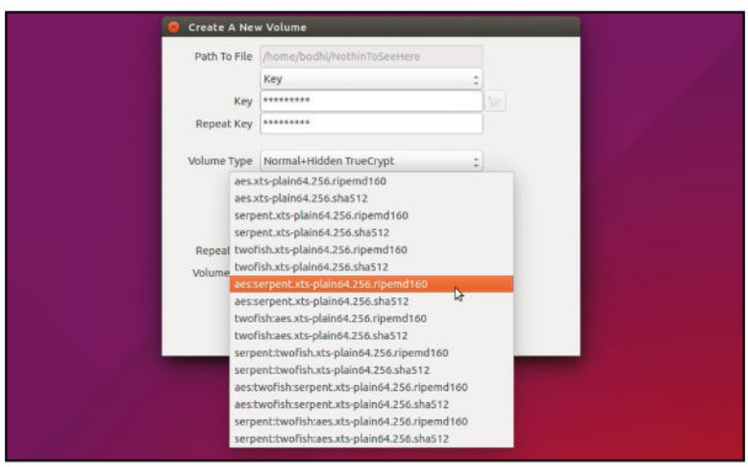
To install zuluCrypt head to **http://mhogomchungu.github.io/zuluCrypt/** and scroll down the page to the binary packages section. The app is available as installable .deb package files for Debian and Ubuntu. Download the package for your distro and extract it with `tar xf zuluCrypt*.tar.xz`. Inside the extracted folder, switch to the folder corresponding to your architecture (i386 for older 32-Bit machines and amd64 for new 64-Bit ones). Both folders contain four binary packages that you can install in one go with the `sudo dpkg -i *deb` command. On other distros you'll have to install zuluCrypt manually. Download the app's tarball and follow the detailed steps in the included BUILD-INSTRUCTIONS file to fetch the dependencies from your distro's repos.

One of the first things you should do after installing is to create encrypted versions of all files that you consider sensitive. Fire up the app and head to zC > Encrypt A File. In the dialog box that comes up press on the button adjacent to the Source field and navigate to the file you wish to encrypt. zuluCrypt will use this information to create a file with the same name and append the .zC extension at the end – or save it elsewhere by clicking on the folder icon adjacent to the Destination field and navigating to a new location.

Next enter the password for encrypting the file in the key field. Make sure the password is a mix of characters and numbers to make it difficult to guess. Also remember that there's no means of recovering the password if you ever forget it, and no possibility of decrypting the file – that's sort of the point! Once you've confirmed the password press the Create button to encrypt the file. This process might take some time depending on the type and size of the file you are encrypting. Once it's done you'll have the encrypted version with the .zC extension in the destination location you specified earlier. Once a file has been encrypted, make sure you delete its original version.

You'll now have to decrypt the file before you can read and make changes.For this, launch zuluCrypt and head to zC > Decrypt A File. Point to the encrypted file in the Source field and alter the location of the unlocked file in the Destination field. Now enter the password with which you encrypted the file and click the Create button. When it's done, the decrypted file will be created in the specified destination. To lock the file again, encrypt it by following the previously outlined procedure.

› zuluCrypt also supports cascade encryption which is the process of encrypting an already encrypted message, either using the same or a different algorithm.



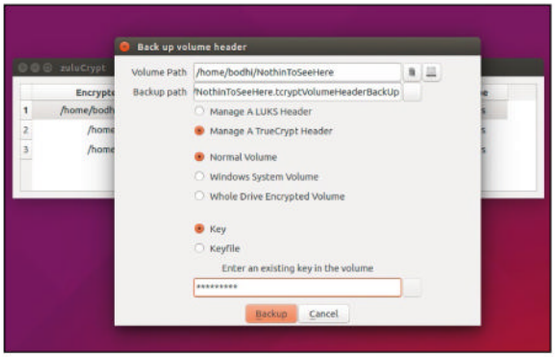
Encrypted data silos

Individually encrypting files works well if you only need to protect a couple of files. Also, it's a cumbersome process and is only suitable for files you don't need to read or modify regularly. If you need to protect a number of files that you access frequently, a better approach is to file them inside encrypted storage areas.

zuluCrypt can perform block device encryption, which means that it can encrypt everything written to a certain block device. The block device can be a whole disk, a partition or even a file mounted as a loopback device. With block device encryption, the user creates the file system on the block device, and the encryption layer transparently encrypts the data before writing it to the actual lower block device. While encrypted the storage areas just appears like a large blob of random data and doesn't even reveal its directory structure.

To create an encrypted storage device within a file, fire up zuluCrypt and head to Create > Encrypted Container In A File. In the window that pops up you'll have to enter the name and complete path of the directory under which you'll house your sensitive data. It's called a file, because it'll appear as a singular file when it's encrypted. You'll also have to specify the size of the directory depending on the size of the files it'll house and the space available of your disk.

When you press the Create button, zuluCrypt pops up another window. First up you'll have to specify a password for encrypting the file. Next, you'll have to select a Volume Type.



You can't unlock a volume without a header. In case the original gets corrupted, create a backup by right-clicking on a mounted volume and selecting the appropriate option.

The default option is LUKS or Linux Unified Key Setup, which is a disk-encryption specification designed specifically for Linux. In addition to LUKS, zuluCrypt can also create and open TrueCrypt, VeraCrypt and Plain volumes. Plain volumes are headerless encrypted volumes and the encryption information is provided by zuluCrypt. Because of this, Plain volumes are application-dependant and not very portable. TrueCrypt or VeraCrypt volumes are better alternatives if the encrypted volume is to be shared between Linux, Windows and OS X computers.

Once you've decided on the type of Volume, you'll have to pick a cipher, an algorithm that does the actual encryption and decryption. An associated attribute of the cipher is the associated size of the key. As the key size increases, so does the complexity of exhaustive search to the point where it becomes impracticable to crack the encryption directly.

The most popular encryption cipher is the Advanced Encryption Standard (AES) which is based on the Rijndael cipher. AES with a key size of 256 bits is widely used as it offers the right balance of speed and security. This is the default cipher in zuluCrypt. However the app supports a large number ciphers including the Twofish algorithm, and Serpent. These two are considered by the US National Institute of Standards and Technology to have a higher security tolerance than AES, but are also slower.

You can safely select the default values for each field, including the default filesystem for the volume (EXT4) and press the Create button. When the process completes, you'll notice a file with the name you specified for the encrypted container with illegible content and size equivalent to what you specified earlier.

Making sense

Before you can store files inside this encrypted volume you'll first have to decrypt and mount it. Head to Open > PLAIN,LUKS,TrueCrypt Container In A File. Use the file button in the pop up window to navigate to the encrypted container file that you've just created. If you wish you can alter the mount name for the file, else just enter the password and press Open. Toggle the checkbox if you only want to read the contents of encrypted volume.

Once your volume is mounted it'll appear in your file system like any other mounted file system. The main zuluCrypt window will also list the volume along with the complete mount path. You can now create directories within this mounted location and create files just like you would on any regular mounted device. When you're done, right-click on the mounted volume in the zuluCrypt interface and select the 'Close' option. This will unmount and encrypt the volume and all you'll have once again is the single encrypted file with illegible content. Mount the file again following the procedure mentioned above to reveal its contents.

If you have issues managing multiple passwords, zuluCrypt gives you the option to create random keyfiles which you can then use to encrypt files and volumes. To generate a keyfile head to Create > Keyfile. Now enter the name for the keyfile and its storage path. From a security point of view, you should make sure the keyfiles are not stored on the same hard disk as the files or volumes it encrypts. In fact it's best to keep these on an external drive which ensures that your encrypted data remains secure even if someone grabs hold of your drive containing the encrypted files and volumes.

To use a keyfile instead of a password, select the keyfile option using the drop-down menu when creating an encrypted volume or encrypting a file. After selecting this option you'll also have to point the app to the keyfile, which will then be used to lock your data.

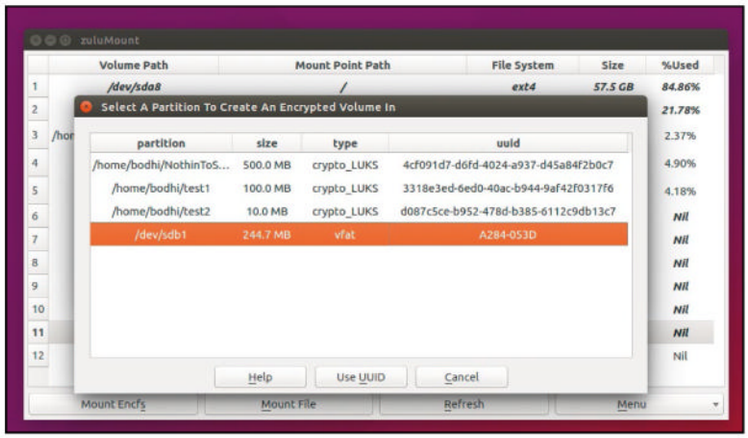
Scramble partitions and disks

If you want to encrypt large amounts of data, it's best to place the encrypted container inside a partition of its own or even on a removable USB drive. Note that when you create such a container zuluCrypt takes over the entire partition or disk, so make sure you've backed up any existing data.

Also, make sure that the destination partition or drive isn't mounted. Use the `mount` command to list all mounted partitions. If the partition you wish to use, say /dev/sdb1, is mounted, you'll first have to unmount it with `sudo umount /dev/sdb1` .

Now launch zuluCrupt and head to Create > Encrypted Container In A Hard Drive. In the window that pops up, zuluCrypt will list all the available partitions that it can use to house the encrypted volume. Note that the devices are listed both by device name and by the associated UUID. If you are creating a container on a removable disk, make sure you toggle the Use UUID option. This will ensure that zuluCrypt always correctly identifies the device. Now double-click on the drive/partition you wish to create the volume on. You can now create an encrypted volume on it using the same exact procedure we used earlier to create an encrypted volume inside a file.

Although at first it might sound cumbersome to use, but over time zuluCrypt will grow on you as you get familiar with the app. There is no easier way for the privacy conscious to keep their data secure.



ZuluCrypt includes the zuluMount tool that can mount all encrypted volumes supported by zuluCrypt and also doubles up as a general-purpose mounting tool.

Beef up security with Kali Linux

Plug the holes in your network defenses with this pentesting distro.

Kali Linux is the swiss army knife for ethical hackers. The distro is loaded with tools for penetration testing that you can use to compromise *your own* network in order to identify the weak points of entry that can be exploited by crackers. The Debian-based distro has more than 300 tools and utilities that are arranged neatly in a categorised and structured menu. Kali Linux is available in multiple flavours and can be used as a Live distro or inside a virtual environment. In this tutorial we'll use some of the common tools in Kali Linux to expose the weak points in our network.

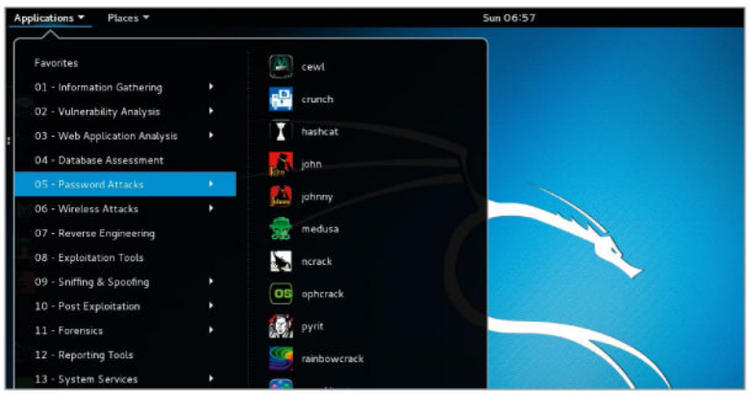
First let's find out what devices are logged on to the network using netdiscover, which can be used for reconnaissance on those WiFi networks that aren't using DHCP. Fire up a terminal inside Kali and type

```
netdiscover -i wlan0
```

which sends out ARP requests over the network and then displays the results on the screen. The process is live, and as soon as new machines come on the network they'll pop up on the screen. Once you have a list of hosts, press [Ctrl] + [C] to stop the scan. With a list of hosts and their MAC addresses you can begin the process of exploiting them.

You will probably need to see what ports are open on these hosts, and the OS they are running. One of the best apps for the job is nmap, which can be easily used via its graphical interface, Zenmap, which lets you run various types of scans on any host within your network. Zenmap ships with 10 common scanning profiles and you can define your own using its extensive options.

The Kali Linux menu is arranged in the order in which a network is usually infiltrated. It begins with tools for infiltration testing and moves on to vulnerability analysis, wireless attacks and exploitation.



Break into WiFi

WiFi Protected Access (WPA) and **WiFi Protected Access 2** (WPA2) are wireless security protocols that were intended to address the security shortcomings of **WEP**. Because the **WPA** protocols dynamically generate a new key for each packet, they prevent the statistical analysis that caused WEP to fail. Nevertheless, they are vulnerable to some attack techniques. WPA and WPA2 are frequently deployed with a pre-shared key (**PSK**) to secure communications between the access point and the wireless clients. The PSK should be a random passphrase of at least 13 characters in length; if not, it is possible to determine the PSK using a brute-force attack by comparing the PSK to a known dictionary. This is the most common attack.

The best way to check whether your wireless network is impervious to attacks or not is to break into it. However, be advised – breaking into a wireless network that isn't under your charge is illegal and shouldn't be attempted.

We'll use the airmon-ng set of tools to crack open the faults in our network. To start, we need to be able to intercept or monitor wireless transmissions; therefore, we need to set the Kali communication interface with wireless capabilities to monitor mode with

```
airmon-ng start wlan0
```

If the command tells you that there might be some processes that could cause trouble, use

```
airmon-ng check kill
```

to kill those processes before reissuing the

```
airmon-ng start wlan0
```

command. The command creates a monitoring interface such as wlan0mon.

When the monitoring interface is up, use

```
airodump-ng wlan0mon
```

to view the local wireless environment. This command lists all networks that it can find within range of the wireless adaptor at that particular point of time. The output includes several key details including the BSSID of each network along with its MAC address, bandwidth information including the channel used, information on the encryption used, and the ESSID which provides the name of the wireless network.

Now locate your network from the list, and make a note of its BSSID and the channel it's on. Then use the information you get to fire up airodump, for example:

```
airodump-ng -c 11 --bssid 28:03:7C:51:10:31 -w /root/
Documents/my-network wlan0mon .
```

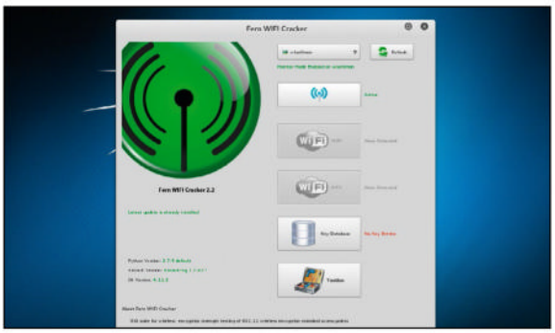
The command will create a bunch of files under the **/root/Documents directory**.

We'll now force one of the devices to reconnect with the router and capture the handshake between them. Make a note of the BSSID of any station and launch a new terminal while leaving Airodump running. In the new terminal window we'll launch what's known as a deauthentication attack where a device is forced to reauthenticate to the access point and re-exchange the secure encrypted WPA2 keys. In the new

Passwords, passwords, passwords.

The biggest threat to security is weak passwords. If you think you have good strong passwords, there are some tools that you can test them with. Hydra is a free brute force password cracking tool which can attack single or multiple user accounts. Primarily a command-line tool, the password cracker also has a graphical user interface that can be used to create complex attacking patterns. Hydra can interact with a huge range of services including HTTP, IMAP, LDAP, POP3, RDP, SMB, VNC and more.

Another popular password cracker included in Kali is John the Ripper. This is used primarily for exposing weak Unix passwords. Like Hydra, John the Ripper is a command-line tool but also has a graphical interface (called Johnny) which does a nice job of exposing its various command line options.



You can use the Fern app to graphically crack your Wireless networks.

terminal window enter

```
aireplay-ng -0 2 –a 28:03:7C:51:10:31 -c 00:1C:50:7D:44:5C
wlan0mon
```

Here the -a switch points to the BSSID of the network and the -c switch points to the MAC address of a station. You might have to repeat this command several times with different stations until you see a bunch of ACKs in the terminal window which indicate that the access point has acknowledged the deauthentication command that was just sent.

Now switch to the original terminal window, still running the Airodump command. If the handshake has been captured you'll notice a number next to the WPA Handshake in the top right-corner of the window. You now have the router's password in encrypted form. We'll now use aircrack to brute force our way through this encrypted password using a wordlist. Kali Linux ships with several wordlists and we'll cycle through all until we find a match. The handshake is captured inside a file under the **/root/Documents directory** with the **-01.cap** extension. To start cracking the password type,

```
aircrack-ng /root/Documents/*-01.cap -w /usr/share/
wordlists/fern-wifi/common.txt
```

This is a time consuming process and you might have to use the other wordlists until one finds your password.

Strengthen your network

As you can see, breaking through a wireless network's security key doesn't take much effort. Depending on the complexity of your password, the process can take anywhere from ten minutes to ten hours. Your only defence against such attacks is a complicated password with various special characters, numbers and mixed cases.

Furthermore, there are a few more things you can do to make the attackers work to gain access to your network. While these won't stand against a determined attacker, they are enough to dissuade the average wardriver looking for free WiFi. You should enable MAC address filtering and if possible add the MAC addresses of all your devices to your wireless router's settings so that only the specified devices can connect to your Wi-Fi network. But know that MAC addresses can be easily spoofed.

You should also disable unnecessary services and protocols, particularly those that are notoriously used for gaining unauthorised access such as SNMP, SSDP and uPnP. If they do gain access to your network, you can limit the damage they can do by disabling administration over WiFi and instead forcing it over a wired connection only.

It's also pretty easy to spoof a network address and trick people into signing into the wrong network. This is easily done with a tool called Airbase which essentially turn your Wi-Fi adapter on Kali Linux into an access point with the same name as another network. Once signed in, the attackers can capture all traffic over the spoofed network including usernames, passwords and all sorts of transactional information. To avoid falling prey to spoofed networks, never log into a network that doesn't require a password and also turn off your distro's ability to automatically connect to Wi-Fi.

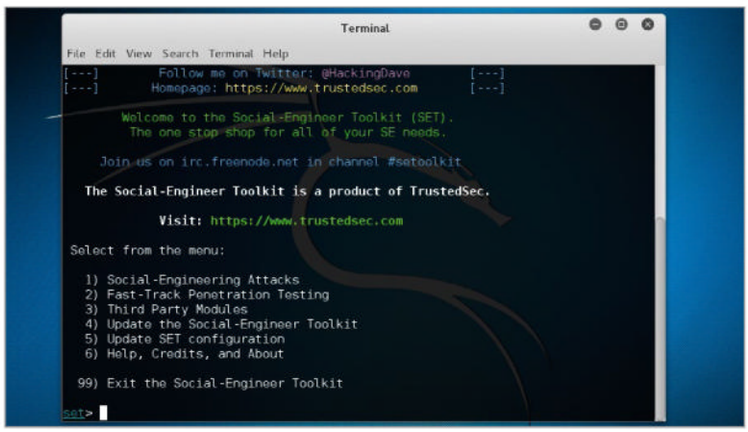
Lastly, you might also want to disable WPS. Kali Linux includes the reaver tool which can exploit a vulnerability in the WPS implementation on many routers and brute force its way to the wireless password. Using the tool is fairly simple. Use airodump and make note of your router's BSSID and channel. Then use

```
reaver -i wlan0mon -b 8D:AE:9D:65:1F:B2 -c 9 -vv
```

to figure out the WPS PIN of your router.

One possible option to circumvent this type of attack is to turn off the WPS function, though it's been reported that this isn't always effective. A better option is to switch to an open source firmware like DD-WRT that doesn't have the WPS functionality in the first place. Also, many new routers can resist brute force attacks by limiting the number of times you can access it. However this too can be circumvented.

All said and done, the best solution to securing a wireless network is to deploy a RADIUS authentication server that's used in conjunction with the WPA2 protocol.



The Social Engineering Toolkit is a collection of scripts to help you cook up various make believe attacks to exploit the human element.

Get more from your Mac

Try the new issue of MacFormat free* in the award-winning app!

bit.ly/macformatipad

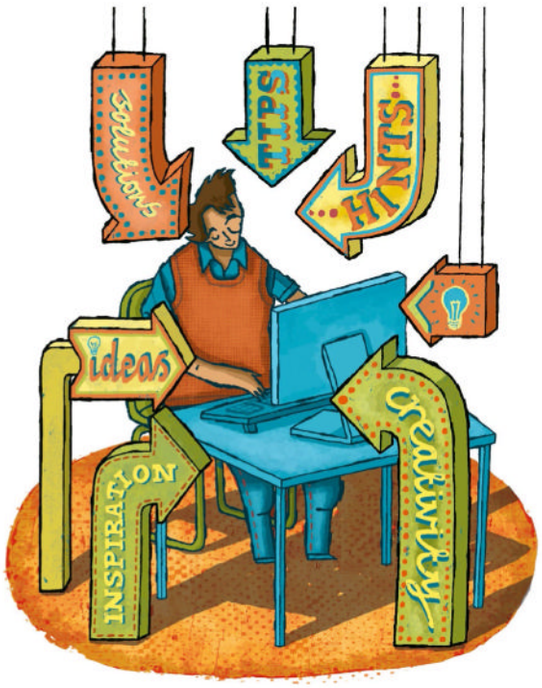
Available on the App Store

Packed with practical tutorials and independent advice – discover why MacFormat has been the UK's best-selling Apple magazine for seven years!

* New app subscribers only

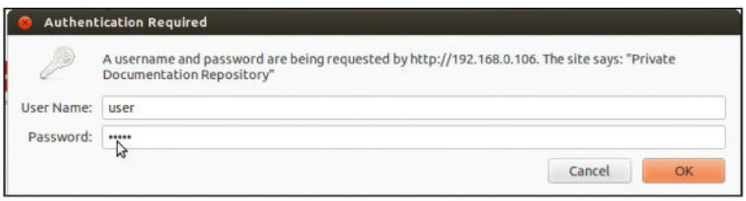
Motion: Detect and record

Learn how to build a livestreaming system, using a Raspberry Pi and a webcam, and how to save your motion-detected video.



We'll assume you have none of the required packages to follow this tutorial on video surveillance and video recording. You will use *Motion* which is the heart of this article. Aside from that, you will require *Apache* (or *Nginx*) and PHP. Although this tutorial is geared towards using a Raspberry Pi, you can use another PC setup if you prefer. Do note, that if you go the *Apache* and PHP route, everything will work very easily without having to make extra changes to the server and PHP.

If you do decide to go with Nginx instead of *Apache* you will need to make some extra changes: such as installing *PHP-FPM*; changing the root folder path for web page files; and editing the following files: **/etc/nginx/sites-available/default**, **/etc/nginx/sites-enabled/default** and **/etc/php5/fpm/php.ini**.



Use simple password protected authentication to keep files secret.

Now, for the synopsis of each package. *Motion* will be used to record video after movement is triggered. The video clips will be written to a folder as Flash SWF files. However, *Motion* still allows you to see the location even without movement, much like a regular security camera.

Once you have those files, you may want to be able to sort through them effectively. Here is where the web server and PHP play their role. With the *Apache* or *Nginx* server, you can serve these files over the web.

Realistically, many files will be accumulated and you may want to create a loop with PHP in order to output each file into a link that can display the video in a popup. In which case a free video popup application, such as *Shadowbox* can be used. Lucky for you, the code (which can be found on the Linux Format website) contains the files that do all that stuff.

With all that covered, you'll have a setup that can deliver your videos. This tutorial will show you various options and their how-to counterparts. Since a camera like this could be used in your home as a security camera, you may want to password protect any web pages or the folder where you keep the videos. Now, if someone did happen to break into your premises and decide to steal or wreck your Raspberry Pi, we'll also guide you through a backup plan that can be used to move your video files to a foreign web server that the robber won't have a clue exists.

Getting things to work

Since this article is about *Motion*, let's install this first:

```
sudo apt-get update
sudo apt-get install motion
```

Now that one installation is out of the way, let's add the rest, which includes *Apache*

```
sudo apt-get install apache2
```

and PHP:

```
sudo apt-get install php5 libapache2-mod-php5 php5-mcrypt
```

Let's move on and make some basic procedures and tests to see everything is working as it should. The main files which you will customise are **/etc/motion/motion.conf** and **/etc/default/motion**. Open up **motion.conf** with your favourite editor. By default, you'll note that the parameters shown below are the opposite of the default values. For example, daemon off becomes daemon on:

```
daemon on
webcam_localhost off
control_localhost off
```

» Save the changes and open up the **/etc/default/motion** file and make the following changes:

```
start_motion_daemon=yes
```

Now, let's fine tune some options. Three changes that are needed are: the frame rate, quality and minimum amount of frames to trigger the motion to record:

```
framerate 30
quality 90
minimum_motion_frames 5
```

Without changing this setting, two frames per second looks way too jerky and will miss out a lot of action, so we change the frame rate from 2 to 30 frames per second. The second change is obvious since it's a quality upgrade. The third change sets the minimum amount of frames of motion that need to be detected. By default, the value is 1. The problem with a number this low is that you can end up with unwanted recordings from things such as lights flicking. Keep in mind that you have many options and can look deeper into the features. A good place to start is on the offical website (**http://bit.ly/MotionConfigFileOptions**).

Some of the other features you might want to consider are: taking a picture at a desired interval, such as every second, every minute or every hour. This feature makes it easy to host a live weather cam, for instance, or to determine if someone is sitting on your couch.

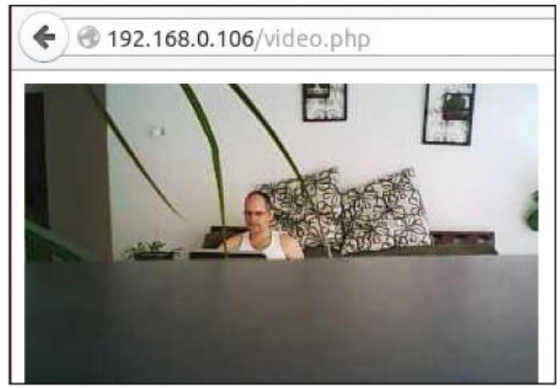
Configuring Motion

Changing all parameters to suit your specific needs is very easy and the **motion.conf** file will often have nice, self-explanatory comments while the website and man page have more information to offer. Obviously, this service doesn't do much without a working, compatible webcam and a list of webcams worth trying with the Raspberry Pi can be found at **http://elinux.org/RPi_USB_Webcams**.

Using a plug and play webcam makes life easy for this task, and one cheap, readily available webcam that works is the Logitech C170. Note: If you are using the Raspberry Pi, the Raspberry Pi cam won't work with *Motion*. To tell if the USB webcam connects OK, run the command **lsusb**.

At this point, you will likely have a working webcam, a working web server and an adequate *Motion* configuration. This is good, but you'll also need to create a folder for the images and set ownership for *Motion*. By default, *Motion* drops the images and SWF files into the **/tmp/motion folder**. It won't create the folder, therefore, you will need to:

```
cd /tmp
mkdir motion
```



› **Alert! Man on sofa. Capturing live video feed.**

```
chown motion:motion motion
```

Now, let's see how everything works. To start with, you can get *Motion* up and running with the command

```
service motion start
```

and you can always restart it with the command

```
service motion restart
```

The first thing you'll need to do to test that everything works fine is to see if you can view the default web page. Since your Pi will have a unique network address, you can just type it in the browser. For example, if the Pi is connected to a router with the IP of **192.168.0.1**, your device could have an IP like **192.168.0.106**. Thus, the URL would also be **http://192.168.0.106**. If you have success returning the default web page, you will see a message stating that everything is working properly. If not, you will get a typical browser error which makes it obvious that something is not quite right.

Now with a working server, let's move on and get down to viewing and recording video. You can test the video in your browser by typing your network IP and port. By default, the *Motion* webcam port will be 8081. Therefore, if you type **http://192.168.0.106:8081** in the browser, you should see your video stream.

A simple setup like this can be beneficial and have many uses aside from security: such as keeping an eye on a new-born while you're working in another room. Since all should be well at this point with the *Motion* service running, you can now go in front of your webcam and jump around a bit. Actually, hand waving will suffice but a few jumping jacks aren't going to hurt. After that, you should be able to browse the new **motion** folder and see a bunch of JPEG files and at least one SWF file.

Quick tip

When you're logged in via SSH and need to edit files, using **vim** to find a string is easy. To do this, all you need is a **/** followed by the string name. Just type **n** to move on to the next one.

Using multiple webcams

Need more than one webcam? No problem. *Motion* enables you to add more easily. You need to open up **/etc/motion/motion.conf** and set up the threads. If you go to the bottom of the file, you see various lines that are commented out followed by the word thread. As you can see, the default location for these new files are in **/usr/local/etc** folder.

To keep it simple, you can change the thread folder to the **/etc/motion** folder. This way, you keep all the editing in one simple location. Now, the first thread would resemble the line below:

```
thread /etc/motion/thread1.conf
```

Once you've set up your threads since you are using multiple webcams, you can create the thread files. Thus, your first thread would be called **thread1.conf**, followed by **thread2.conf**, and so on. The code that you add to these threads just needs to be a few lines. The code samples below display two threads. As you can see, each thread has its own videodevice parameter, custom text that appears on the left-hand side of the video stream and image folder and port number. Here' s **thread1.conf**

```
videodevice /dev/video0
text_left Camera #1
target_dir /var/www/images
webcam_port 8081
```

followed by **thread2.conf**:

```
videodevice /dev/video1
text_left camera #2
target_dir /var/www/images_cam2
webcam_port 8082
```

Now, that you have a motion-detecting device that you can view from your local network, you can move on and make adjustments so that you can see and view your webcam from outside your local network. For some of you, this may be the setup you desire; especially if your webcam and Raspberry Pi are well hidden and unlikely to be tampered with.

However, in addition to storing data in your own home, we will explain how to back up the files to another server for safe keeping, just in case your SD card or hard drive fails, or someone decides to steal, break or ruin your webcam (or webcams) in your home.

Making it web friendly

The whole idea here is to record, store and manage video from a website or by using your IP address that was given to you from your ISP. To find out your IP head to **http://whatismyipaddress.com**. In order to broadcast video out, you'll need to set up port forwarding on your router to enable your network IP to use port 8081.

While you are at it, you may as well do the same for port 80 since this same network IP will be used to display web pages from computers outside your local network; such as your friend across town or a loved one overseas.

After you have made the previous changes to your router's settings, try typing **http://my_ipaddress_from_isp** and **http://my_ipaddress_from_isp:8081**. You should get the same results as you did when you were checking your local IP.

The next step is to clean this up and view organised data view web URLs; such as **http://my_ipaddress_from_isp/video.php** or to view it from another website using an iframe.

In order to show the webcam from the page **video.php**, you just need to use an img tag with the network IP and port. Have a look at the code below, which shows this in complete detail and displays it with the default width and height as specified in the **motion.conf** file:

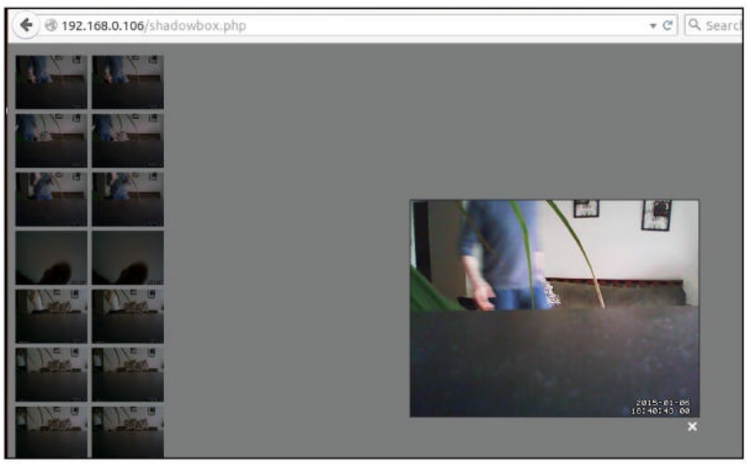
```
<img src="http://192.168.0.106:8081/" width="320"
height="240"/>
```

Now, let's imagine a scenario where you want to stream this livecam from another website using an iframe. Well, all you have to do is make an iframe from another page on the different server. The simple one-liner is shown below.

```
<iframe style="width:320px; height:240px;" src="http://isp_
ipaddress/video.php"></iframe>
```

The next set of code will explain how to display the files that have been saved after motion is detected and recorded.

Displaying stored images and Flash video.



With that said, we will move on and create a simple way to organise the saved files that recorded the movement. The first detail that will need to be changed is to save the images and SWF files into a folder within the web directory. The root web folder is located at **/var/www/html** or **/var/www**. At this point, a light should go off since you have already made several changes to the *Motion* setup. Reopen the **/etc/motion/motion.conf** file and change the target directory. By default, the target directory is located at **/tmp/motion**. The new target is **/var/www/images**:

```
target_dir /var/www/images
```

Viewing recorded video and images

After making changes to **motion.conf**, type the command:

```
sudo service motion reload
```

so that it will now attempt to write any new files to the **/var/www/images folder**. Now, you can easily access the files created by the *Motion* service and display them on the web just like any other typical web page. Although the path has been changed in **motion.conf**, the **images** folder hasn't been created yet. So, make it now.

The folder will be located within the **www** or **html** folder. If it sounds like we're repeating ourselves here, because it means you have been paying attention and are aware that the *Apache* root **web** folder can be in one of two paths:

```
cd /var/www
mkdir images
```

By default, the **www** directory will be owned by the root user and root group. You will want to make some changes; such as all files will be owned by pi and the group will be **www-data**. To change this use:

```
cd /var
chown -R pi:www-data www
```

So, what we are up against now is to make this **images** folder writable by the *Motion* service. As of right now, the other files have adequate ownership and permissions, but, the images folder does not. Well, let's change that right now. The code snippet below has three commands. The first command will add the user motion to the **www-data** group. In case you are wondering, **www-data** is an existing user and group for the *Apache* server. The second command gives the images folder permissions to the *Motion* user and **www-data** group, and the final command makes the folder writable so that the images and SWF files can magically appear in the **images** folder:

```
usermod -a -G www-data motion
chown motion:www-data images
chmod 777 images
```

It's within the **www** folder where you can create a file called **shadowbox.php** that will be used to display content on the web. This file has all the code you need to display a thumbnail from each recorded SWF video, the video itself and the first JPEG image of motion.

The coding process to display the content goes like this: The images directory gets scanned and an array of files is created and sorted in descending order. These files are multiple JPEG files and a single SWF file for each event. All files have a name that starts with the event, followed by the date and finally followed by the sequence.

The date sequence is year, month, day, hour, minutes and seconds. After that, *Motion* just adds the sequence starting from 01 only for the JPEG files.

In order to keeps things simple and uncluttered, the coding for the **shadowbox.php** file will only display a single image and a single SWF file for each event. This simple script also uses *shadowbox* to create popups of the first JPEG image and flash SWF file for each event. Now, you can see all the latest detected motion, down to the first recorded by *Motion*. This file gives all of the results and here is where you may want to customise the output.

If you want to keep these web pages password protected with *Apache*, you can open up the file **/etc/apache2/sites-available/default** and make some minor changes.

If you look for the line **<Directory /var/www/>**, you can add three simple lines to it. The code sample is shown below:

```
AuthType Basic
AuthName "Private Documentation Repository"
AuthUserFile /var/www/.htpasswd
Require valid-user
```

After that, you navigate to the **/var/www** folder and create a blank file called **.htpasswd**, and create the username and password with the simple command displayed below. You will be prompted for a password twice. You simply add it followed by Enter:

```
sudo htpasswd /var/www/.htpasswd add_username_here
```

Since the files can pile up pretty quickly and your disk space can readily disappear, you may want to create a purging system or back them up to another drive. Some backup plans are discussed next.

Backup plans

One tip for determining your backup plan is to watch how much space you routinely tend to use and develop a plan based on those conditions. Simple. For example, if you go through 1GB a week and you have a 8GB card you may want to TAR the images folders, SCP the file to a remote server and remove all files that are more than one-week old. Since the files contain the year, month and date, it's a rather easy process to delete the ones that have expired. The file called **purge.php** is a cleanup file that you'll be able to find on the Linux Format websitef.

This file removes every file that's more than a couple of days old. I will explain the code in a little more detail in a moment. First off, the **images** folder is scanned and all of the files become an array. That array of files then iterates through a foreach loop. A few built-in PHP functions, such as **strstr()**, **preg_replace()**, **substr_replace()**, **substr()**, **date()** and **unlink()** are used to translate all the file names into actual date timestamps that can be used for comparison.

```
73002 -rw-r--r-- 1 motion motion     17211 Jan  6 18:40
73001 -rw-r--r-- 1 motion motion     16824 Jan  6 18:40
73000 -rw-r--r-- 1 motion motion     16939 Jan  6 18:40
72999 -rw-r--r-- 1 motion motion     17084 Jan  6 18:40
72998 -rw-r--r-- 1 motion motion     16833 Jan  6 18:40
72987 -rw-r--r-- 1 motion motion   1665571 Jan  6 18:43
72997 -rw-r--r-- 1 motion motion     16497 Jan  6 18:40
72996 -rw-r--r-- 1 motion motion     16471 Jan  6 18:40
72995 -rw-r--r-- 1 motion motion     15426 Jan  6 18:40
72994 -rw-r--r-- 1 motion motion     14838 Jan  6 18:40
72993 -rw-r--r-- 1 motion motion     14571 Jan  6 18:40
72992 -rw-r--r-- 1 motion motion     15005 Jan  6 18:40
72991 -rw-r--r-- 1 motion motion     15866 Jan  6 18:40
72990 -rw-r--r-- 1 motion motion     15933 Jan  6 18:40
72988 -rw-r--r-- 1 motion motion     15760 Jan  6 18:40
73835 drwxr-xr-x 4 pi     www-data    4096 Jan  6 19:35
73942 drwxrwxrwx 2 pi     www-data  106496 Jan  6 20:49
root@raspberrypi:/var/www/images#
```

All the files in the images folder are named with an event, followed by date and sequence.

Once a timestamp is made from the filename, it goes through a simple **if()** statement and is compared against a time that is set to two days ago from the current time. This part is really easy to change since you just need to change the number 2 to your desired amount of days in the past. Once this criteria is met, the file is deleted with the **unlink()** function. Since this system is only using files without a database, it's rather elementary to move all of these files to your backup location, and since this is copying and moving files, two methods come to mind. One is using a package such as *rsync* and the other is a simple method of compressing the desired files and folders with ZIP or TAR and shipping them to their new destination with SCP. An simple example of SCP is shown below:

```
scp -P 22 /var/www/images.tar pi@example.com:/home/pi/images.tar
```

So there we have it. You've just created your own video surveillance and motion recording system that has several options to suit your needs or that you can customise. Although we've made a rough skeleton and files for you to monitor your video, record files and make backups, you can take this farther if you want. Some simple suggestions would be are to add a responsive template to both the **video.php** and **shadowbox.php** files, and polish up the content with a little CSS magic.

On top of that, you could set up webcams at other sources and have them viewable by the public or friends, depending upon what you want to achieve. Have fun!

Quick tip

You may want to include the time in the file name so backups will not be overwritten. In addition to that, you may want to run a *cron* job that does this procedure on a regular basis.

Nginx and Motion

Nginx doesn't ship ready to go out-of-the-box for *Motion* as *Apache* does. In fact, after you install *Nginx* there's a series of required steps that you must do before you have a smooth operation.

In the case of Raspberry Pi, you can adjust the **worker_processes** value from 4 to 1. You can change this in the **/etc/nginx/nginx.conf** file. This is recommended since the Pi only has a single CPU core.

After that, you will want to change the default web folder since the default points to **/usr/share/nginx/www**. To change this, you open the file called **/etc/nginx/sites-enabled/sites-enabled/default**. The change is shown below so the web folder is **/var/www**:

```
#root /usr/share/nginx/www;
root /var/www;
```

After the previous step, you can quickly install *fastcgi*. The command is below:

```
apt-get install php5-fpm
```

After that, you'll need to open the file **/etc/nginx/sites-available/default** and change a few lines of code so it resembles the content below. Basically, you'll just need to remove a few comments:

```
location ~ \.php$ {
    fastcgi_split_path_info ^(.+\.php)(/.+)$;
    # NOTE: You should have "cgi.fix_pathinfo = 0;" in php.ini
    # With php5-cgi alone:
    #  fastcgi_pass 127.0.0.1:9000;
    # With php5-fpm:
    fastcgi_pass unix:/var/run/php5-fpm.sock;
    fastcgi_index index.php;
    include fastcgi_params;
}
```

We're almost there. Now you'll have to open the file **/etc/php5/fpm/php.ini** and remove another comment so that it looks like the line of code below:

```
cgi.fix_pathinfo=1
```

Finally, make sure to restart *Nginx* after making all of the changes. The command

```
/etc/init.d/nginx restart
```

will do the trick.

techradar.

The home of technology

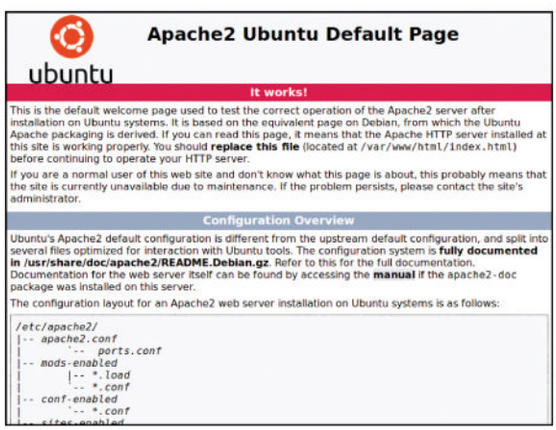
techradar.com

Apache: Ensure a secure start

Apache may be old, but it's tough and it can be used to serve web pages with the strategy and valour worthy of the Apachean people.

The venerable *Apache* HTTP server is considered the granddaddy of web servers, although it's only just celebrated its 20th birthday. Recently we've extolled the virtues of younger, spryer web servers, (in particular *Nginx*, but also *LiteSpeed* and *Lighttpd*), but *Apache* has, since 1996, been the most widely used in the world (by any reasonable metric). Sure, if you're just running a simple website then maybe *Nginx* can serve your pages a few nanoseconds faster, but unless it's a terribly popular website then this is unlikely to trouble you. Indeed compared to *Nginx*, *Apache* might look clunky, or even intimidating, with its diverse configuration files and myriad mysteriously monikered modules. But in this tutorial, we'll try to demystify things: Once we've covered the basics, we'll focus on some security and privacy aspects. It may not be as exciting as an all-singing, all-dancing HTML5 web application, but it might be more helpful.

Once you're all set up and everything seems to be working, let's pause for an over-simplified helicopter view of what it is that *Apache*, or any web server for that matter, really does. Being a server, it will listen for requests, and being a web server, the requests that it will be interested in are HTTP or HTTPS. These may be associated with the server's IP address or a domain name which resolves to this address. A single server can happily serve multiple domains (so-called virtual hosts which we'll study soon), so the first task is to sort out which virtual host the domain part of the URL refers. Then the server studies the remainder of the HTTP request so it can be mapped to the appropriate local resources. These might be static files, eg HTML or images, but could equally be dynamic responses generated on the server-side, eg from PHP or Perl scripts. In the simplest case the part of the URL following the first **/** can be translated to an actual location on the server's filesystem by prefixing with the location of the virtual host's document root, eg **example.com/index.html** might resolve to **/var/www/example/index.html** . This need not always be the case, we can define arbitrarily complicated rewriting rules so that the physical location bears no resemblance to this. For CGI programs the situation is more complicated, but the idea is the same – data from the HTTP request is somehow fed to a script or program that, hopefully without getting exploited, constructs the appropriate HTML. This is then returned to the web server, and in turn the client.



This is what you see on Ubuntu when everything works. Reassuring, but you should disable the default website.

Quick tip

The *Apache* camp have a few things to say about the changes Debian ship in their default config. Read all about it here: **http://bit.ly/DebianDiffs**.

Harden up, bru

If you peruse the (heavily-commented) main configuration file, two things you might notice are the **User** and **Group** directives. When *Apache* daemon is started it initially runs as root, but once it read its configuration files and got its bearings then subprocesses are spawned which run with the credentials specified by **User** and **Group**. It is with these subprocesses that clients have any interaction, so that if anything does go wrong then any attempts at malfeasance won't have root privileges off the bat, which is A Good Thing. Many Linux daemons start this way, since there are certain initial tasks which need root – in the case of *Apache* one such task is binding to port 80 (ports lower than 1024 aren't generally available to mere mortals). The Debian/Mint/Ubuntu convention is to run as the user **www-data** (specified in the file **/etc/apache2/envvars** which is referenced by the main config file), other layouts will use the **http** user. Best practice dictates the *Apache*-running user shouldn't have a login shell and should not be used for any doing anything other than running *Apache*.

As a result of these dropped privileges, any file which you want *Apache* to deal with will have to be readable by **www-data**. Likewise, any directory housing content you wish to be accessible will need to be both readable and executable by this user (the execute bit behaves slightly intuitively for directories on Linux) . Once you start running web applications, then certain files or folders will need to be writable by **www-data** too, but it's best to be as conservative as possible here, eg start with the root being the owner of everything in **/var/www** and give all its subdirectories 755 permissions and files 644. If a program or script fails due to needing to write something, then grant the permissions one

Install and test

Just to confuse you, different distros have chosen to name their *Apache* packages differently. Arch Linux seems to lack imagination, going with **apache**, OpenSUSE and the Debian-based ones have gone with **apache2** and Red Hat's progeny go with the traditional **httpd**.

Once you've appropriately delegated the task to your package manager, it's worth having a look at the main configuration file (possibly to instil a sense of fear, but it also contains some good guidance about how things are arranged). The traditional location here is the rather long-winded **/etc/httpd/conf/httpd.conf** which (again confusingly) is respected by Arch, Fedora etc, the Debian-based distros have opted for **/etc/apache2/apache2.conf** and OpenSUSE has opted for **/etc/apache2/httpd.conf**. Unless otherwise stated, we'll assume a Mint/Ubuntu setup for this article – there's a helpful summary of various distro's *Apache* layouts at **https://wiki.apache.org/httpd/DistrosDefaultLayout** to aid with path and filename translations if you're using something else. The structure (though neither the location nor the content) of *Apache*'s config files is consistent across distros, and while initial configs will vary, most generally ship in a ready for action state. Once you've started the service with

```
$ sudo service apache2 start
```

you can navigate to **http://localhost** and (all going well) you'll see a reassuring 'It works' page. Other distributions may give an empty directory listing, which should also reassure you. You can place your own **index.html** file in the directory **/var/www/html/** (or **/srv/http** on Arch Linux) if you want to display something else.

file and one error message at a time. One thing you should definitely not do is make any file which is read by **root** during the initial startup (eg anything in **/etc/apache2**) writable by **www-data**.

With the *Apache* daemon running, browse to **http://localhost/server-status** . You might see a 'Not Found' error, or (if you're running Ubuntu or Mint) you might see all kinds of information about your web server and ask yourself how the page got there as there's no **server-status** file in the website's root directory (wwwroot). The answer is it came from the **mod_status** module. This status information may look pretty harmless, and can be very useful when diagnosing *Apache*, but it can also prove useful to cyber criminals (as our government seems to prefer to call them instead of 'hackers'). If we weren't using a Debian-derived distro, then disabling **mod_status** would involve removing/commenting out the line:

```
LoadModule status_module modules/mod_status.so
```

from the main config file. However, the Debian family have generously provided some nice scripts for enabling and disabling modules. Inside the **/etc/apache2** directory you'll see, amongst others, directories entitled **mods-enabled/** and **mods-available/**. The former contains symlinks into the latter for each module that is enabled. There are links to **status.load** and **status.conf**, the former contains the above line, and the latter contains various configuration data for the module. The **mods-*** folders enable us to keep the main config file clean. This is A Good Thing, as is the nice suite of scripts the Debian guys provided for managing the symlinks. For example, we can easily disable **mod-status** with:

```
$ sudo a2dismod status
```

You'll need to reload the *Apache* daemon before this change is noted. If you decide you want the status information back again, then it is a simple matter of:

```
$ sudo a2enmod status
```

The **a2ensite** and **a2dissite** commands provide the same convenience for virtual hosts, and **a2enconf** and **a2disconf** do so for modular configuration options. As well as disabling **mod_status**, we can also add the following two lines to **/etc/apache2/apache2.conf** so that we don't betray the *Apache* version number in error pages or HTTP requests:

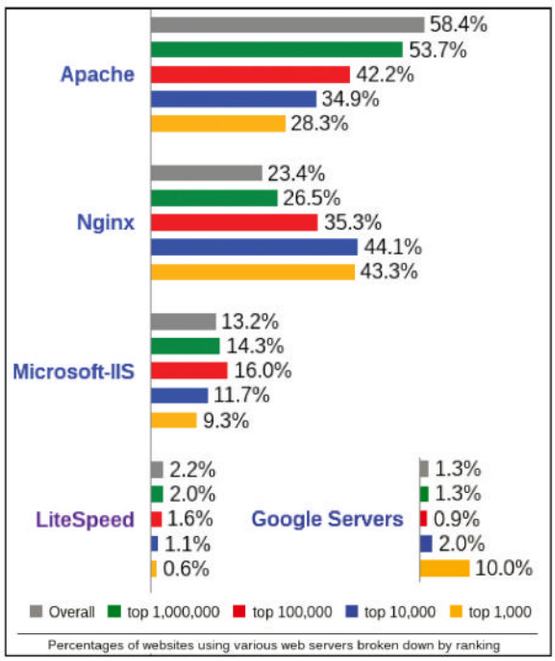
```
ServerTokens Prod
ServerSignature Off
```

By default, if you browse to a directory that doesn't contain an **index.html** file, or other acceptable file specified by the **DirectoryIndex** directive, then you'll get a nice directory listing telling all and sundry the files and directories that reside therein. This is generally not desirable, so we'll turn that off globally by disabling the **Indexes** option for **/var/www/**. Find the appropriate section in **apache2.conf** and add the desired minus sign so that it looks like:

```
<Directory /var/www/>
    Options -Indexes FollowSymLinks
```

Virtual reality

Even if you're only going to be running one website, it's still nice to set it up as a virtual host, if nothing else it keeps the main **apache2.conf** file free of pollution. The default installation on Debian and friends uses a virtual host set up in the file **000-default.conf**, which you should have a look at. We'll use this to set up two domains on our web server. If you don't have access to registered domain names with A records you can still use a bogus **.local** suffix to illustrate the point (or just use hostnames if that's how you roll). Suppose your webserver's local IP is **10.0.1.1** and we wish to set up the two domains below. Then you'll need to add entries in the **/etc/** »



According to this w3techs.com survey, *Apache* is way ahead of the competition. *Nginx* beats it for high-traffic websites, but many people use it as a reverse proxy for *Apache*.

Quick tip

For a great primer on HTTPS read Robert Heaton's blog: **http://bit.ly/HTTPSGuide**.

» **Discover how to customise a wireless router** turn to page 146

» **hosts** file of any machine on your network (including the web server itself) that you want to be able to view this:

```
lxfweb1.local 10.0.1.1
lxfweb2.local 10.0.1.1
```

Alternatively, you can use a dynamic DNS provider to point diverse novelty domain names at your IP. Either way, the next step is to add entries for your website(s) in the **/etc/apache2/sites-available/** directory. We'll copy the default template and tweak it for our two websites above:

```
$ cd /etc/apache2/sites-available
$ sudo cp 000-default.conf lxfweb1.conf
$ sudo cp 000-default.conf lxfweb2.conf
```

We'll store the websites in **/var/www/lxfweb1** and **/var/www/lxfweb2**, so create these directories and add the following lines inside the **<VirtualHost *:80>** section of **/etc/apache2/sites-available/lxfweb1.conf**:

```
ServerName lxfweb1.local
ServerAlias www.lxfweb1.local
DocumentRoot /var/www/lxfweb1
```

Quick tip

It's worth keeping an eye on the access and error logs at **/var/log/apache2**, where you can see who's accessing what and diagnose what's breaking.

Do the same for the **lxfweb2.conf** file, put placeholder content in each DocumentRoot, and enable the two websites:

```
$ sudo a2ensite lxfweb1.conf
$ sudo a2ensite lxfweb2.conf
```

Shazam! Two websites, ready for action. Actually three: if you access the web server by its IP address, or a different domain name that resolves there, you'll get the default site as defined in **000-default.conf**, which you are free to modify. Or indeed disable entirely, should your web server feel that it ought only to be accessed by name and not number.

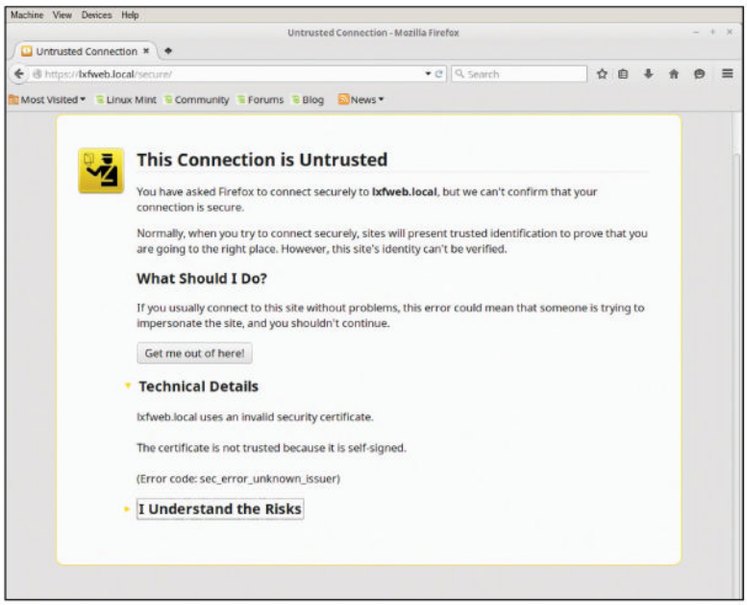
One can control *Apache*'s behaviour on a per-directory as well as a per-site basis. For the former we can strategically place **.htaccess** files. In the appropriate directories, but since these are prone to getting forgotten about we can also use the **<Directory>** directive in the site's configuration file. We're going to add a secure area to our **lxfweb1.local** site, which can only be accessed with a password. First, we'll make the area's directory and put some placeholder content there:

```
$ sudo mkdir /var/www/lxfweb1/secure
$ cd /var/www/lxfweb1/secure
$ echo Classified Facility - no cameras | sudo tee index.html
```

Now edit **/etc/apache2/sites-available/lxfweb1** and add the following near the end of the **<VirtualHost *:80>** section:

```
<Directory /var/www/lxfweb1/secure>
    AuthName "Secure Area"
    AuthType Basic
    AuthUserFile /var/www/.htpasswd
    require valid-user
</Directory>
```

› ***Firefox*** **won't trust a certificate you generate. Hardly surprising, we wouldn't trust you either.**



Used like this, the **Basic** authentication mechanism just checks a file for a matching username and password combination. These files are maintained by the **htpasswd** program which is part of the **apache2-utils** package, which we now install and utilise.

```
$ sudo apt-get install apache2-utils
$ sudo htpasswd -c /var/www/.htpasswd lxfuser
```

You will be prompted for a password for **lxfuser**. The **-c** switch creates a new file, but if you want to add further users then just use the command without it. Now reload *Apache*:

```
$ sudo service apache2 reload
```

When you browse to **http://lxfweb1.local/secure** you will be prompted for a username or password. If you enter incorrect details, then you will continue to be prompted. There are more advanced authentication methods such as verifying users by database or LDAP, or having supplementary admission criteria such as a specific IP address. Have a look at the docs for details: **http://bit.ly/ApacheAuthDocs**. It's important to put the **.htpasswd** file outside of any defined website's **DocumentRoot**. This is in case any misconfiguration (the default config won't let this happen) which could accidentally result in the **.htpasswd** file being served, for example at the URL **http://lxfweb1.local/.htpasswd**. In our case we've got websites defined in subdirectories below **/var/www**, but that directory itself is OK.

HTTP-yeS

Any data sent via an HTTP request or received in the response is done so in the clear. Anyone with access to a machine in between you and the web server can access it, or even alter it. This is hardly satisfactory, especially given that we are wont to transmit personal and financial data. To work around this, we use SSL/TLS technology via the HTTPS protocol. Properly implemented SSL provides two things: Encryption – the data passing between you and the client is obfuscated by high-powered mathematics and Authentication – you can be confident that the website you are fraternising with is indeed what it says it is.

While the mathematics behind encryption has been thoroughly researched (albeit oftentimes poorly implemented), the authentication issue is something of a thorny one. The solution at present is to rely on (ie trust implicitly) a collection of Certificate Authorities (CAs) which provide (at cost to commercial operations, although personal ones are available for free) their sanctioning of a given website in the form of a digital signature on said website's certificate. Your distro maintains a list of those CAs it considers trustworthy in the **ca-certificates** package. From time to time some of these will be revoked due to a scandal, and browsers frequently check in with a Certificate Revocation List so as to minimise potential malfeasance.

First, read and obey the box about generating and signing a certificate (*see Generating a Self-Signed Certificate*). We need to tell your web server to use these credentials for handling HTTPS connections, which usually take place on port 443. You can either offer HTTP in parallel with HTTPS, or you can make your website (or portions thereof) accessible only by HTTPS. A standard *Apache* installation comes with a file **/etc/apache2/sites-available/default-ssl.conf**, which we can modify slightly to suit our purposes, eg, lets enable an

SSL site, as well as the HTTP one, on **lxfweb1.local** from before. As before, copy the default site file

```
$ cd /etc/apache2/sites-available
$ sudo cp default-ssl.conf lxfweb-ssl.conf
```

and change the following lines in **lxfweb-ssl.conf**:

```
<VirtualHost *:443>
ServerName lxfweb1.local
DocumentRoot /var/www/lxfweb1
…
SSLCertificateFile /etc/apache2/ssl/server.crt
SSLCertificateKeyFile /etc/apache2/ssl/server.key
```

We should also preclude old cipher suites to prevent any kind of downgrading-attacks. The old and weak 'export' ciphers which gave rise to the recent FREAK attack, along with many other low-grade ciphers, ought to be disabled by default on most distros' *Apache*/OpenSSL packages. That notwithstanding, said defaults are still often not perfect. We can improve things a little by changing the following lines in **/etc/apache2/mods-enabled/ssl.conf**:

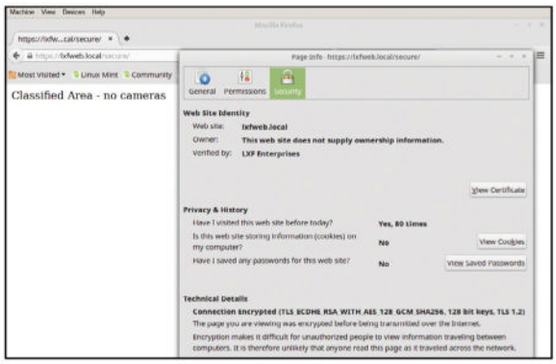
```
SSLHonorCipherOrder on
SSLCipherSuite HIGH:!MEDIUM:!LOW:!aNULL:!eNULL:!EX
PORT:!MD5:!RC4:!3DES:!PSK:!SRP:!DSS
SSLProtocol all -SSLv2 -SSLv3
SSLInsecureRenegotiation off
SSLCompression off
```

Disabling the deprecated SSLv3 protocols precludes the POODLE attack (and also visitors using IE6), disabling compression does so against CRIME. (You may wish to omit this if you're more bandwidth-challenged than paranoid.)

It's worth considering perfect forward secrecy too: The goal of the SSL negotiation process is to come up with a session key known only to the server and the client, and thrown away after use. Newer forms of key exchange do so in a way that generates this key ephemerally: in such a way that a subsequent compromise of the server key alone is insufficient to recover any captured data from the session. Unfortunately the default (either RSA or fixed Diffie-Hellman) key exchanges don't do this, so we should tell *Apache* to use the newer methods by modifying the **SSLCipherSuite** line from above. It's worth giving a few alternatives here since, eg not all browsers support TLS 1.2 which is required for Elliptic Curve crypto. All this makes for a very long line, so just replace **HIGH** above with the following cipher combinations.

```
EECDH+ECDSA+AESGCM:EECDH+aRSA+AESGCM:EECD
H+ECDSA+SHA256:EECDH+aRSA+SHA256:EECDH+aRSA
+RC4:EECDH:EDH+aRSA
```

This favours the newer, faster Elliptic Curve Diffie-Hellman mode, but also allows for the slower but widely-supported Ephemeral DH, all with a variety of ciphers and hashes.



We have entered a secure area, apparently. Newer cipher modes that provide perfect forward secrecy have been properly implemented by TLS 1.2.

Now enable the SSL module and your freshly detailed site and restart *Apache*:

```
$ sudo a2enmod ssl
$ sudo a2ensite lxfweb-ssl
$ sudo service apache2 restart
```

When you browse to your website your browser will (if you didn't pay for a signed cert) give you a big ol' warning about an untrusted CA, which is not surprising. But just this once you can make an exception and continue to the secure site. In *Firefox* you can store this exception, though it will still persecute you about the dodgy certificate.

If you want to redirect all traffic from the HTTP site as well, then add the following line after **ServerName lxfweb1.local** in **/etc/apache2/sites-available/lxfweb1.conf**:

```
Redirect permanent / https://lxfweb1.local/
```

Alternatively, use this second line if you want to force HTTPS for the **secure** directory from the beginning of the tutorial:

```
Redirect permanent /secure https://lxfweb1.local/secure
```

If you're using *Chrome* or *Chromium* then you can forcefully add your certificate to your own keystore using the **certutil** program. Click on the broken HTTPS icon and find the 'Export certificate' option, saving it as, say **lxfweb.crt**. Then import this into your local NSS database with:

```
$ certutil -d sql:$HOME/.pki/nssdb -A -t P -n lxfweb -i
lxfweb.crt
```

While it's nice to get the reassuring padlock icon next to the URL, adding security exceptions like this is potentially dangerous – you might forget that you've done so and, if you're unlucky, your server keys may be stolen. With this an attacker could, at some point in the future, potentially set up a malicious site which your browser would trust implicitly.

And so concludes our introduction and begins your journey into things Apachean. Be careful what (if anything) you make available to the outside world and definitely don't break any laws (or hearts).

Generating a self-signed certificate

A (reputable) CA will only sign a certificate if it pertains to a domain name which you have control over, so if you haven't invested in such a thing (subdomains, eg from dynamic DNS services, don't count) then you can't get your certificate signed officially. But you trust yourself, right? So you can generate and sign your own certificate which will allow visitors to your web server, and if they trust you enough to ignore the warning about an invalid signing authority, then they can confidently connect to your website using SSL, safe in the knowledge that any information passing between it and them is safe from prying eyes. So long as you set it up correctly, that is:

```
$ sudo mkdir /etc/apache2/ssl
$ sudo openssl req -x509 -nodes -days 365
-newkey rsa:2048 -keyout /etc/apache2/ssl/
server.key -out /etc/apache2/ssl/server.crt
```

You will be asked for some address and company details, as well as a Common Name (which you should set to your domain name if you have one) and a contact email address. This will generate a self-signed X.509 certificate, which will be valid for one year and will include a 2048-bit RSA key, (use **openssl --list-public-key-algorithms** to see others available). It's also worth imposing some permissions on the key file and certificate, since if it fell into the wrong hands then you would be susceptible to a textbook Man-in-the-Middle (MitM) attack.

```
$ sudo chmod 600 /etc/apache2/ssl/*
```

Reading certificates and keys is one of the things the root portion of *Apache* does on startup, so these files need not (and should not) be readable by **www-data**.

HACKER'S MANUAL 2016

HACKER'S MANUAL 2016

Do more

Take your Linux skills to the next level and beyond

Future

lifehacker | UK

Helping you live better & work smarter

LIFEHACKER UK IS THE EXPERT GUIDE FOR ANYONE LOOKING TO GET THINGS DONE

- Thousands of tips to improve your home & workplace
- Get more from your smartphone, tablet & computer
- Be more efficient and increase your productivity

---

www.lifehacker.co.uk

twitter.com/lifehackeruk facebook.com/lifehackeruk

BUILD A LINUX PC

Assembling a PC is straightforward, but choosing the components is less so. We look at the options available.

There is a document that has been floating around the internet for at least fifteen years, called 'What if operating systems were airlines'. The entry for Linux Airlines states: "When you board the plane, you are given a seat, four bolts, a wrench and a copy of the seat-HOWTO. html". It's an old joke but as Linux users we are still more accustomed to sometimes having to do things for ourselves than users of other operating systems.

Not that this is necessarily a bad thing, as it means we understand our computers better. Still, at least when you want a new PC you can just go out (or online) and buy one. So why do people build their own? Over the next few pages we will try to answer this question, as well as the more complex questions that arise when you try to do it, such as: how hard is it to do? What are the risks? What about warranties? Will it save me money? Can I build a computer with no Windows? And many more. There are several reasons why you may want to build your own, not least of which is the satisfaction of understanding your computer that little bit more, but this information is not only useful if you want to build a new system from scratch. Much of what we cover here will also be of benefit if you are looking to upgrade an existing computer.

> **"Desktop systems are generally very easy to work on – we find LEGO far more taxing."**

We will been concentrating on desktop systems, which are generally very easy to work on – we find LEGO more taxing. Laptops are another matter, but many of the points about choosing suitable Linux-compatible components still apply and we will end with a look at picking a laptop, or any other type of sealed box, such as one of the popular nettop systems. »

Future

GIZMODO | UK

Not your average technology website

EXPLORE NEW WORLDS OF TECHNOLOGY GADGETS, SCIENCE, DESIGN AND MORE

- Fascinating reports from the bleeding edge of tech
- Innovations, culture and geek culture explored
- Join the UK's leading online tech community

---

www.gizmodo.co.uk

twitter.com/GizmodoUK facebook.com/GizmodoUK

You can buy one of these and get whatever the maker puts inside, or you can assemble it yourself and get exactly what you want.

Why build your own computer? You may be able to save money by sourcing the components yourself, but don't bank on this. What you do get to do is pick the exact specification you want – no throwing away parts to upgrade to what you really wanted. You also get to choose the quality of components you use. A pre-built PC may say it has a 2TB hard drive, but that tells you very little. Most hard drive manufacturers produce several drives of that size, with varying speeds, power consumption and intended usage. You will learn a lot more about components doing it yourself, and while putting together a desktop PC is not difficult, doing so will give you the confidence to dive back inside it when you want to upgrade. You can also upgrade incrementally – once you have a computer you understand, you can increase its hard disk or memory capacity now, then add a faster processor and motherboard next year.

> **"There are still components that are better supported than others."**

Remember Moore's Law

Moore's Law predicts a doubling of computer capacity every two years. Unless you plan to build a new computer every year, you should allow for your needs increasing during its lifetime. Always allow for spare capacity, if you want 16GB of RAM, buy a motherboard that takes 32GB and half-fill it. Your storage use will tend to increase too. 4K video is on the way, so build a system with more drive bays and SATA connectors than you need now. Storage and memory are easy to increase, provided you have left yourself the option. Hopefully your processor and motherboard will last you several years.

What do you need?

There are several standard building blocks needed to build a computer. As a minimum, you will need: processor, motherboard, memory, storage drive, graphics card, case and power supply. You may also need a monitor, keyboard and mouse; but you could be building a media centre that uses a remote control and plugs into a TV, or a headless server such as a NAS box.

Over the next few pages we will look more closely at each of these, explaining the choices to be made. The idea is not to tell you which components to choose but to give you the information to make that decision for yourself based on your particular needs. When you have built your computer, you will then need to install an operating system and software. Naturally, we assume you will install Linux but you may also need to install Windows for gaming or some other particular software needs, so we will look at how to install the two OSes in perfect (well, almost) harmony.

Choosing the components

When it comes to choosing your components, you need to take into account what you will use the computer for now and what other uses are likely. The usage can dramatically alter the choices you make, eg a gaming system needs fast storage but not a lot of it for the OS and current games you are playing, so an SSD is ideal. A NAS is the opposite, lots of space needed but speed is not critical, so a slower spinning hard drive would be better.

Linux users have another aspect to consider: compatibility. While the situation is far better than at any time in the past, there are still components that are better supported than others. Let's work through the main components you will need and look at your options.

Processor

The first decision to be made is which processor to use, as this affects your choice of motherboard and then just about everything else. The choice may seem obvious: get the fastest you can afford, but things are never that simple. The fastest processors are also the most expensive, often by a substantial

Find the price-performance sweet spot for your preferred processor manufacturer.

Linux compatiblity

Hardware support in Linux is very good nowadays. Most devices are supported directly in the kernel. There is no need to go trawling manufacturers' websites for drivers. When you are buying components for your own computer, you need to know whether there's support for the hardware before you buy a component. The first step is to determine the exact component in use, which is not as easy as it sounds. The motherboard maker's website may state it has a Gigabit Ethernet port, but not tell you which chipset, so you will have to do some research.

Typing the full product code and the word Linux into your favourite search engine will normally give plenty of hits. It's worth restricting your search to recent posts, there is no point reading complaints about your choice not being supported a year ago when it is now. Look for posts that give details, such as chipset codes and module names and avoid the rants. Asking in web forums, like our own (**www.linuxformat.com/forums**) or at **www.linuxquestions.org**, about the support for a particular device should get you some first-hand experiences, but search the forums yourself first. Your question may have already been answered and no one like questions from people too lazy to do their own research.

Compatibility problems usually arise because the manufacturer gives no help to the kernel driver developers so everything has to be reverse engineered – this commonly applies to wireless adaptors but some sound cards are also affected. Another reason may be that the hardware you are looking at is so new that support has not yet been added to the kernel of your current favourite distro. You may need to look at a newer, more bleeding edge release to get support. This seems to affect things like network adaptors, wired and wireless, the most.

margin. There's a value for money sweet spot that's usually a couple of steps down from the ultimate. Processor speed is not everything, as memory can have a greater effect on your computer's performance. Going down a step in processor speed and spending the savings on more memory, or an SSD, will usually result in a faster computer. Try leaving a system monitor like *top* running while you use your computer and see how often you use all of your current processor's performance. Unless you are a gamer, or spend your spare time compiling custom kernels, you will be surprised at how rarely your processor usage becomes a limitation.

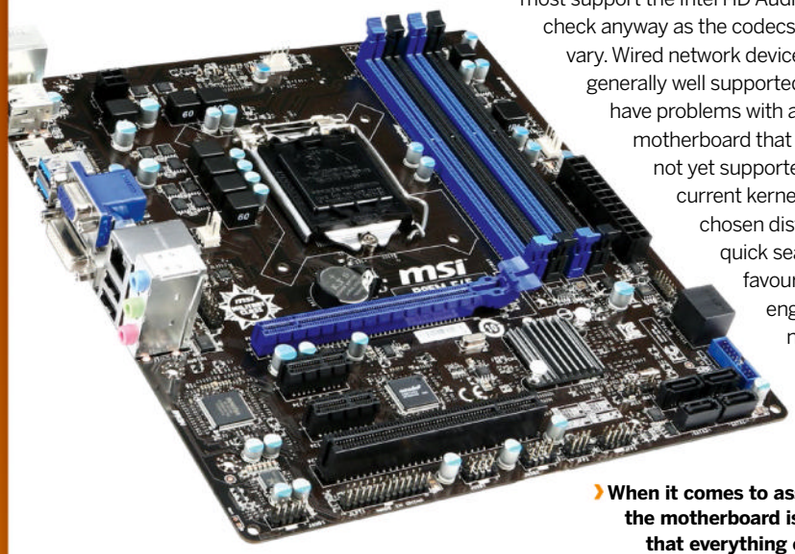
The other choice to be made is between Intel or AMD hardware. You could try a web search on which is best but be prepared to spend some time searching for real facts among the religious fervour. The truth is that processors are so fast nowadays that for a home system, you would be happy with either. Processor choice affects the choice of motherboard, so it may be that, depending on your needs and budget, you end up choosing a CPU to go with the motherboard you want.

Motherboard

If the processor is the heart of your computer, the motherboard is the central nervous system. Everything plugs into this board. In the past, that was all a motherboard did: connected everything together. Now they contain a lot of previously separate components built in, such as network interfaces and sound cards. As Intel and AMD use different CPU sockets, choosing your processor immediately reduces the number of motherboards you have to choose from. Now there are some questions you need to ask, such as do I need more than one network interface? How much memory will I want to fit – now and in the future? How many SATA connectors will I need – don't forget to allow for one for your optical drive and possibly an external eSATA connector too. Also, how fast are the SATA interfaces? This is especially import if you are using an SSD. How many USB ports do I need, and what type? While USB 2.0 is fine for a keyboard and mouse, you will want USB 3.0 connectors for storage devices, and possibly a USB-C connector.

Consider the built-in devices of a motherboard you are interested in, such as network interfaces and sound 'cards'. While the motherboards themselves will work with Linux, will the peripheral devices? Sound cards aren't much of an issue these days as most support the Intel HD Audio standard but check anyway as the codecs in use can vary. Wired network devices are also generally well supported, but you may have problems with a very new motherboard that uses a device not yet supported by the current kernel of your chosen distro. As usual, a quick search of your favourite search engine, using the name of the board and the Linux distribution should turn up and potential problems and solutions. Support for onboard devices may not be a critical issue if you have enough PCI slots you can install alternatives and then disable the built in hardware.

Size can be important. If you are building a system in a large tower case, a full-sized ATX motherboard is easier to work with, but for a media centre you may want a smaller form factor that will fit in an attractive case that can go under your TV.

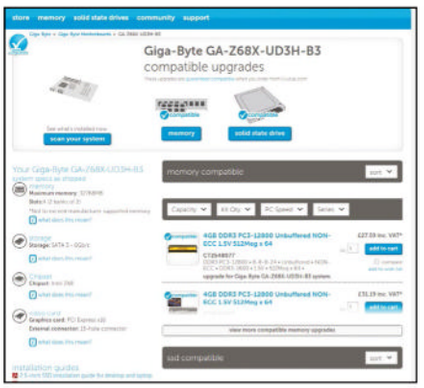


When it comes to assembling a PC, the motherboard is just the thing that everything else plugs into.

Memory may seem unassuming, but it has a greater effect on the performance of your computer than many other components.

Memory

Memory is one of the simplest ways to boost the performance of a modern computer. It enables more tasks and data to be handled at once and any memory that's left over is used by the Linux kernel to cache disk data which, incidentally, is why your computer shows almost no memory free after it has been running for a while. There are three things to consider when buying memory, the physical layout, the size and the speed. The current memory standard is DDR3 and DDR4. Don't try to fit the older DDR or DDR2 sticks into these slots, they are incompatible. Size is obvious, and it is worth getting the most your budget allows. Check your motherboard's manual for the maximum size for each slot because, although the spec allows for up to 16GB per DIMM, most Intel systems are limited to 8GB per unit. Motherboards usually use a dual-channel architecture for memory,



Some memory vendors provide databases that are very handy for seeing which memory is best for your motherboard.

Terminology

You will encounter plenty of abbreviations and jargon when looking at hardware, here are some of the common terms used:

- **DDR 2/3/4** Double Data Rate, a technology using in memory chips.
- **DIMM** Dual Inline Memory Module, this is plug-in memory module.
- **WAF** Wife Acceptance Factor, seen on male-dominated forums and considered important for hardware that will be in the home.
- **Northbridge/Southbridge** Chips on the motherboard that handle core logic and communication between the CPU and other components.
- **SATA** Serial ATA, the current standard for connecting storage devices to computers.
- **ATX** A motherboard configuration specification, meaning most boards are interchangeable in physical format and standard connectors.

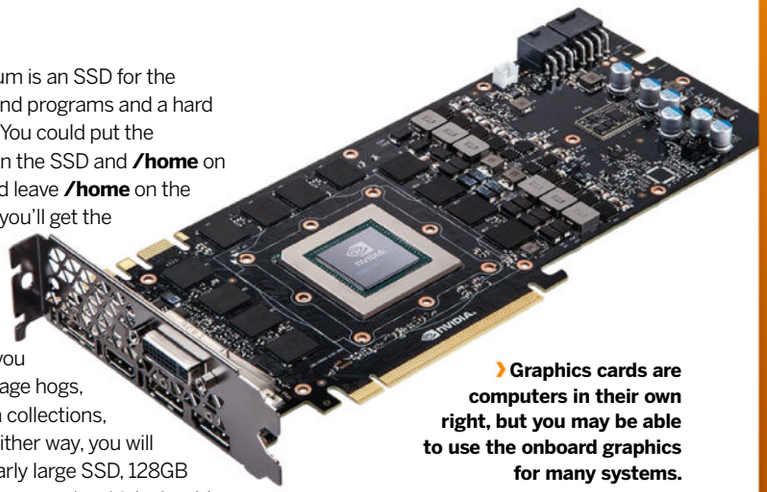
so fit sticks in pairs and check the manual to see which slots go together. This is unnecessary if you are filling all four slots, but unless you are using the maximum size stick, it is better to use two larger sticks. If you want to fit 16GB of RAM, two 8GB sticks give you the option of adding more later, four 4GB sticks does not.

Storage drives

It used to be that the only question to ask about a hard drive was: How big? Now we have to add: How many? And what type? The choice of type is mainly between traditional hard drives with spinning platters and solid state drives (SSDs). (Although there are also hybrid drives that use a spinning disk with an SSD as a cache.) SSDs are much faster than hard drives but cost a lot more per gigabyte, a 256GB SSD costs about the same as a 3TB drive. So an SSD gives much faster booting and program loading but a hard drive holds much more. An SSD also uses less power and is more robust, making it the obvious choice for a laptop. With a desktop, you have the luxury of being able to use more than one drive, so the optimum is an SSD for the operating system and programs and a hard drive for your data. You could put the operating system on the SSD and **/home** on the HD, or you could leave **/home** on the SSD, which means you'll get the benefit of the extra speed for things like your web browser and mail caches. Then you can mount the storage hogs, such as your media collections, on the hard drive. Either way, you will not need a particularly large SSD, 128GB should be more than enough, which should mean you can make sure you get a good one. There is quite a variance in the performance of SSDs, so make sure you get one with decent read and write speeds, in the region of 500Mb/s is good. Apart from that, setting up an SSD is pretty much the same as a hard disk.

SSDs are the obvious choice for a laptop but are idea on a desktop system for installing an OS and the programs you use.

Another option to consider is using multiple drives in a RAID array. This gives you redundancy, if one drive in the array fails your data is still on the other. This isn't a replacement for taking regular backups but it does protect you against a drive failure. With RAID 1, the simplest configuration, two drives are mirrored. All data is written to both drives but read from one (which can give improved read performance as the data comes from whichever drive seeks to it first). Most distro installers will handle installing to a RAID array, but with RAID 1 you can also install to a single drive and add the second to create the array. RAID is handled by the Linux kernel, do not enable any RAID settings on your motherboard. These are so-called fakeRAID and require Windows drivers to work. Just let the installer see that you have two drives and the kernel take care of them.



Graphics cards are computers in their own right, but you may be able to use the onboard graphics for many systems.

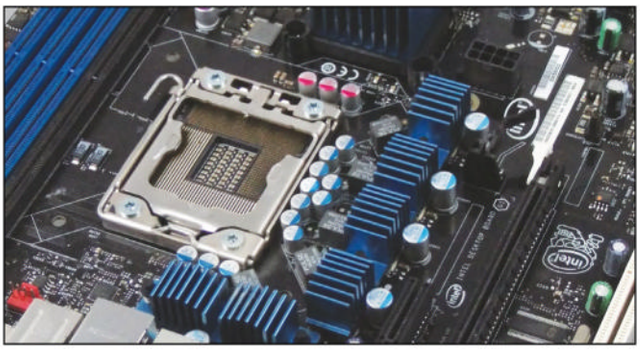
Graphics card

Most motherboards have reasonable onboard graphics these days, so you may not need a separate card. If you want one, the choice is between Nvidia and AMD, and this is another topic likely to provoke religious flamewars.

Nvidia graphics cards will work as far as booting the computer into a graphical display, but the in-kernel nv drivers are limited.

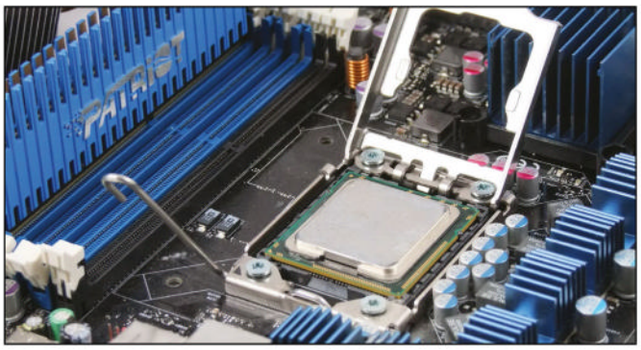
You have a couple of options. The newer open source nouveau drivers work well now – we use them ourselves – and give reasonable performance. If you want the best performance from your graphics card though you will need to install Nvidia's own drivers. As these are closed source they are often not enabled by default so you need to enable the option for restricted or third-party drivers in your distro. You can also download and »

Putting it all together



1 Motherboard layout

Find the diagram of the motherboard's layout and connectors in its manual and keep it open at the section. You should refer to it before making each connection to the board. Many of the connectors are simply header pins on the board, making it easy to put something in the wrong way. Murphy's Law is definitely lurking here. RTFM aplies to hardware as much as software.



2 Zero Insertion Force

Lift the release lever on the CPU socket, insert the CPU aligning the corner dot on the chip with the mark on the socket. It should drop in with no pressure, then lock it with the lever. Remove the tape from the conductive pad on the heatsink and fit it over the CPU, following the instructions that came with the CPU and motherboard. Connect the fan cable to the motherboard.

» install the drivers directly from **www.nvidia.com**, but this is not recommended as then your distro's package manager cannot track and update them. The situation with AMD graphic cards is similar, the potential hassles of running a binary driver versus the lower performance of the open source drivers. The choice between AMD and Nvidia is almost as religious as the Intel vs AMD CPU choice. Unless you need top notch 3D performance, either will work well for you, and if you need that performance you will have to use the binary drivers whichever way you choose. Currently, Nvidia's proprietary drivers are considered better more reliable, but that could easily change.

Case

The computer's case is often left as an afterthought, but a larger, well made case makes life so much easier. A Cheap case will make cable routing harder and will often have sharp edges that will mean bleeding knuckles and blood stains on your nice new motherboard (you can probably sense the first-hand experience in that statement).

A larger case also provides better airflow for improved cooling. If you are building a media centre that goes in the living room, a high WAF (*See Terminology, p34*) is a major consideration, but pay attention to noise levels. What initially seemed like a whisper can become annoying while watching TV. If you choose to go for a case with a window in the side, bear in mind that you will have to be extra careful when routing cables. A rat's nest of cables is never a good idea, but it's even worse when it is on show.

Power supply

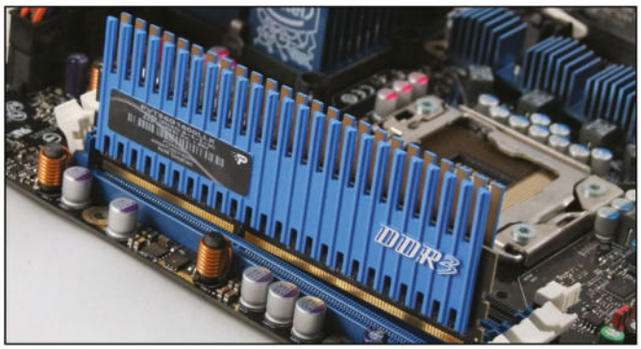
One of the most overlooked components is the power supply (PSU). Avoid the cheap PSUs that are included with cases (in fact, avoid cases that are cheap enough to include a PSU). A PSU must be reliable and good quality. That may seem an obvious statement that applies to all components, but when a PSU fails it can do so in a way that takes out other components. Having a £100 motherboard or hard drive, or both components, wrecked by a £15 PSU is not a good way of saving money.

Other aspects to consider when choosing a PSU are: that it must supply sufficient power for your needs, now and in the future, that it is efficient (any PSU worth its salt is rated 80 Plus, but look for Gold, Platinum and Titanium models) and that it's quiet (particularly for a living room computer). There are several websites where you can list the components you are using and get a recommendation of the power supply you need, for example **http://support.asus.com/PowerSupply.aspx?SLanguage=en**. Check that the PSU you choose has connectors you need, most motherboards take a 24+4-pin ATX connector and a separate CPU power lead. Also ensure you have enough drive power connectors of the right type, some PSUs still come with more of the old Molex connectors than the useful SATA power plugs. If you are using a large case with a bottom mounted PSU, check that the leads are long enough to reach the drives at the top, although you can buy SATA power extender leads cheaply enough that deal with the last two points. You'll be pleased to know there are no special considerations to be taken regarding Linux compatibility!

Specialised systems

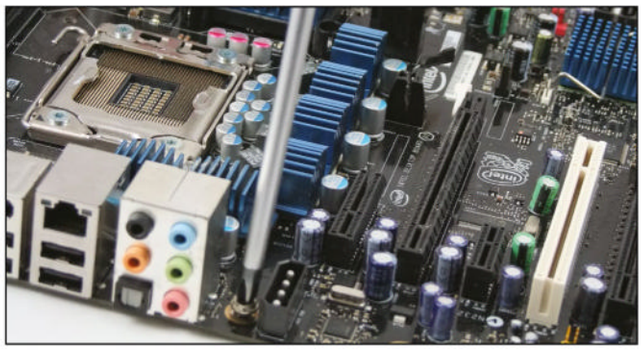
So far we have looked mainly at a general purpose desktop system, but there are some specialist users to consider, such as a home server or a high-performance gaming systems. If you want to build a home server, your requirements will be very different from a desktop or gaming system.

As all data is transferred over the network, a fast SSD is pointless. You could use one for the operating system, but servers are rarely rebooted, so you wouldn't even benefit from the fast boot speed they offer, and they only run a limited number of programs, so loading time is not that important either. Servers generally run headless, so a fancy graphics



3 Memory slots

Insert the memory DIMMS into the sockets on the motherboard. Make sure they are the right way round; slots in the DIMMS should line up with protrusions in the socket and they only go in one way. If you are not using all the slots, consult your motherboard manual for the optimum slots to use and to ensure you take advantage of any dual-channel abilities.



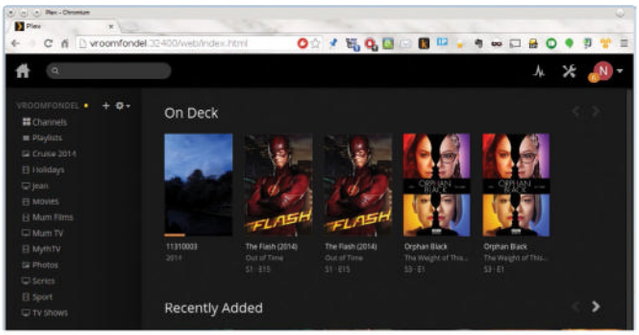
4 Motherboard backplate

Fit the motherboard's I/O plate to the opening in the back of the case and then place the motherboard in the case. There are small posts that it should stand on, you may need to screw them into the case before going ahead and fitting the motherboard. Then secure the motherboard to the posts with the supplied screws – do not overtighten them.

card isn't needed either, you only need a monitor for the installation process and the onboard graphics of most motherboards are more than suitable for that job. What you do need is a decent amount of memory and plenty of storage space. If you are building a file or media server, pick a motherboard with plenty of SATA ports and a case with a similar number of drive bays. However, much storage you add, at some time you will want more. If you are building a web or mail server, storage space is not such an issue, unless you want to serve media files over an internet connection, but plenty of memory helps, especially with *Apache*.

Another type of home server is a media server. This could simply be a repository for your video files or you could be doing the recording on the server too, using *MythTV*'s backend server software or something like Tvheadend. Either way, you will need plenty of space. If you have a server that transcodes video too, maybe something like *Plex* that reformats video to suit the device playing it and the speed of its connection to the server, you will need a decent amount of RAM and a reasonably fast CPU – not a gaming monster but something that's not budget.

As everything works over the network, it goes without saying that it should have a fast network interface, especially if more than one client will be using it at the same time, so make sure it has a gigabit interface, or budget for a decent PCI-e network card.

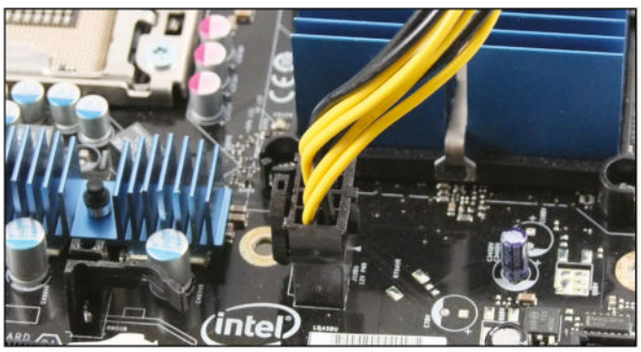


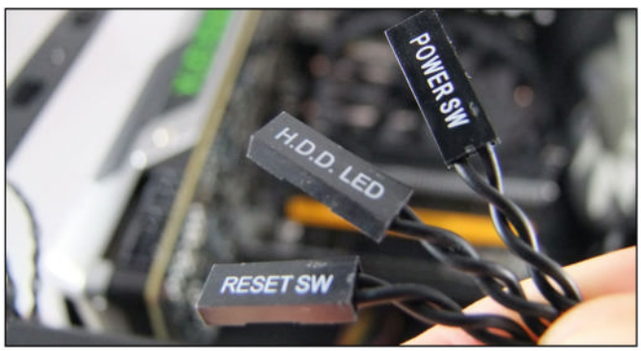
If you are building a server, like this media server, your needs are different. You need plenty of disk space and a fast network connection (or two). You may also need some CPU horsepower for media transcoding.

High-performance gaming

Gaming, and we mean the graphically intense kind and not gambling, places specific requirements on a computer. Performance is the obvious factor, but not just the performance of the CPU. The GPU on the graphics card plays a more important role than the CPU with some games. Clearly you want a fast CPU and graphics card. Many of the higher performance graphics cards take up the space of two PCI-e slots so you need to make sure both your computer's case and motherboard have room for it and that your PSU is capable of powering it in terms of raw power and physical connectors it can offer. Memory is also important, you want plenty of it and the fastest available. Disk storage space is less important than speed, so an SSD is the preferred choice for storage. Gamers always want more performance and one of the ways of getting it is by overclocking their systems. This is where you run your processor and memory at higher than its rated speed. For general purpose use, opinions on overclocking are divided; it does give better »

> **"Many of the higher performance graphics cards take up two PCI-e slots."**





5 Connectors

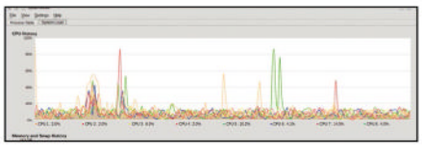
Fit the power supply and connect the power and CPU cables to the motherboard. This is a good time to connect all fan cables. Add your hard drives or SSDs and your optical drive. They may either screw to the case or in some cases they have screwless fittings. Connect the SATA cables from the drives to the motherboard. Connect their power cables for the PSU.

6 Case LEDs

Connect the cables from the case switches and LEDs to the motherboard. These are fiddly connectors but make sure you get them right, the motherboard manual will tell you what goes where. Some of the cables come with a block connector to make the job easier. This is also the time to connect the case's USB ports to the relevant points on the motherboard.

» performance but at the cost of possibly shortened component life and unreliability if you push it too far, plus more heat to get rid of. Some CPUs are locked at a particular speed, so you need to make sure the processor you choose is able to be overclocked. You also need a suitable motherboard. They all offer some control over frequencies and voltages, but some motherboards are intended for overclockers and offer far more control – you'll need to do your research carefully.

To deal with the extra heat, you need a good sized case to maintain plenty of airflow, combined with a decent CPU cooler (the stock cooler supplied with the CPU isn't intended for overclocking) and plenty of fans. A motherboard intended for overclocking may have more fan connectors and some speed control for them. When choosing cooling fans, go for the largest you can fit. A larger fan shifts more air; fast spinning fans create more noise.



› **Even with this four-year-old system, the CPU spends a lot time of doing nothing, but it's nice to have the capacity.**

If noise is an issue, consider water cooling. Overclocking used to be a specialist field that required you to buy and assemble various components then spend ages bleeding the system and performing leak tests (it may surprise you, but water and electronic do not mix well). Nowadays pre-built units are available where you bolt the radiator assembly to a suitable point on the case and clip the heat sink in place of the CPU fan. Some kits also include GPU coolers; as mentioned, gaming works them hard too. Water cooling is not only for gamers and overclockers, it can be use on a standard desktop system to reduce the noise. The loudest noise from such a setup comes from the hard drive stepper motors.

› **Building your own laptop is not a reasonable proposition, but you can still research the components to make sure you get what you need.**

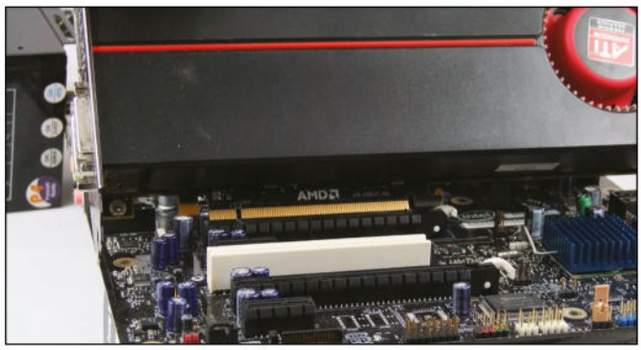
Sealed systems

Assembling a desktop or server computer is a fairly straightforward task, creating your own laptop isn't. While it's possible to buy laptops piecemeal, it's not the usual way of doing things. Still, much of what we have covered in this feature still applies. It is just as important to choose suitable components for a laptop; even more so really because you cannot simply swap them out if you don't like them.

Installing an operating system

Once you have built your computer, you will need to install an operating system. We will not look at that here, the distro installers are well enough documented, but there are a couple of points to consider. New motherboards will use UEFI rather than the older BIOS. They may also have Secure Boot enabled. You will almost certainly need to turn this off in the firmware. Press the relevant key at boot to configure the firmware, the motherboard's manual will tell you which it is (one of the joys of building your own computer is that you get a manual for your motherboard).

You should be able to use UEFI to boot the computer, most distros now work with it, and it's nicer to work with than the old MBR system. If you need to use MBR booting, there will be an option for this in the firmware menu. Some have a UEFI boot option that you can turn off, others have the cryptically named CSM (Compatibility Support Module) that has to be enabled for MBR booting. If you are building a Linux-only system, you can just boot from your favourite DVD and install. If you want to dual boot with Windows, you should install Windows first and then Linux. The Linux installer will pick up the Windows installation and set up a dual-boot menu, whereas the Windows installer would just trample over the Linux bootloader if you did things the other way round.



7 Graphic card

Fit your graphics card into the PCI-16 slot. Check its documentation to see if it needs an external power feed or whether it gets all the power it needs from the PCI slot. Fit any other PCI cards you may be using at the same time. Many new power supplys (PSU) offer modular cable systems, else adaptors are available if your PSU is missing suitable connectors, but ensure it's powerful enough.

8 Tidy cables, tidy mind

Keep your cables tidy, even if you don't have a window in the case. Untidy cables will disrupt airflow and make life harder the next time you try to add or replace a component. If your case doesn't have clips to hold cables out of the way, small cable ties do a very good job of it. You can also often feed cables behind the motherboard tray or inside the hard drive fixtures.

So it is a matter of finding a pre-built laptop within your budget that meets your requirements and then checking that everything will do what you want it to. The same applies if you are looking for a nettop to act as a media player.

The same rules apply for choosing a suitable processor, graphic card, sound card and so on (although the use of the word 'card' here is not strictly correct, almost everything is on the motherboard). There are a couple of important differences: laptop hard drives tend to be slower than their desktop counterparts, making the speed advantage of an SSD even greater. Memory upgrade options are limited, so choose a device with plenty to start with and make sure that it has at least one free slot for expansion. Some laptops have only one slot so upgrading memory means throwing away what you have rather than adding to it.

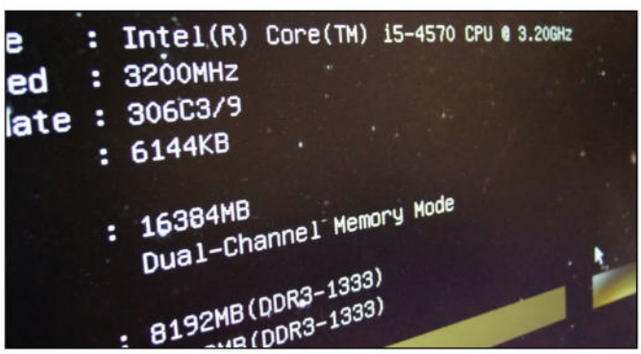
If you have the option, try booting the laptop from a recent live CD, or a live distro on USB. System Rescue CD with the Alt kernel option gives a very recent kernel with the most up to date hardware support. Run `$ ifconfig -a` to see that both wired and wireless interfaces show up and use `$ aplay` to make sure the sound system works all the way from the given file to the speakers. Wireless interfaces can still be a problem with some laptops, but most have a mini-PCI slot these days, in which case you can try another card.

While the Intel vs AMD CPU war and the Nvidia vs AMD graphics war will continue to rage on, Intel provides excellent support for its components. If your laptop has both Intel graphics and wireless cards, it won't go far wrong and everything should just work with no need to install extra drivers. Now grab your favourite distro and enjoy the speed and freedom of Linux!

Careful now

Building a PC isn't difficult, but some care is needed when working with electronic components. Prepare a large working area, clear of any clutter and with a non-conductive surface. Static electricity can kill electronic components, and it can build up on your body without you noticing, until you pick up a component and zap it. You can avoid this by earthing yourself to discharge and static build up. The simplest way to do this is to touch a grounded object, such as a central heating radiator or the kitchen sink before touching a component. You can also buy anti-static wrist straps that connect to a grounded point by flexible cable to keep you static free during your build.

The computer you build will not have a warranty, but the individual components will. As long as the fault wasn't caused by damage while fitting them, any reputable vendor will replace them.



8 Testing, testing

Connect a monitor and keyboard and power up the computer. Hold down the relevant key that loads the BIOS/firmware menu and go straight to the system health page to make sure your CPU is running cool enough. Only then should you check that all of your memory and drives are recognised. Some motherboards may need a setting changed to make all memory visible, but your new PC is now built!

200 BEST-EVER LINUX TIPS

There's always something you don't know. Here are 200 amazing ways to get more from Linux, gleaned from years of navigating its nooks and crannies.

Getting started

Testing and installing Linux distros like a pro.

1 Create a Live distribution with persistent storage
The most popular distros, such as Fedora and Ubuntu, ship with tools that earmark storage space on the live USB disk for saving data that will be available on subsequent reboots.

2 Put multiple live distros on one disk
If you want to test several live distributions you can put them all onto one USB flash drive using either the MultiCD script (which you can find here: **http://multicd.us**) or by using the French MultiBoot LiveUSB tool (**http://liveusb.info/dotclear**).

3 Use an external partitioning tool
While the partitioning tools within the distributions have improved considerably in terms of achieving better control over your disk, it's best to prepare partitions for a Linux installation using third-party tools, such as *Gparted*, which is also installed in the live versions of several distros.

4 Use LVM partitions
One of their biggest advantages to using LVM (Local Volume Manager) is that unlike standard partitions you don't have to estimate partitions at install as you can grow (or shrink) an LVM volume without losing any data.

Get more from the desktop

Be more productive on your favourite desktop.

5 Middle-click to paste
When you highlight some text with your mouse, the text is copied to a special buffer. When you middle-click in a text entry area, a copy of the text that you originally highlighted is pasted into the text entry field.

6 Define keyboard shortcuts
Almost every mainstream desktop allows you to define custom keyboard shortcuts. You'll find the option under the keyboard setting in their respective configuration panels.

7 Touchpad tricks
Move your finger up and down the right side of the touchpad to scroll vertically and tap in the lower-right corner of the touchpad to perform a right-click.

8 Enable workspaces
To enable workspaces in Ubuntu, head to System Settings > Appearance > Behavior and toggle the Enable workspaces option.

9 Install a Dock
On desktops, such as Gnome, cut down the time that it takes to launch your favourite apps by placing them on a Dock, for example the lightweight *Cairo-Dock* which is available in the official repos of most distros.

10 File manager context-menu
The right-click context-menu that is found inside the file manager on most desktop distros are full of useful options that you might have missed, such as the ability to email, compress or restore them to an earlier version.

11 Create Favourites
Place your favourite apps in Ubuntu's Launcher and Gnome's Dash by dragging them from the desktop's respective applications view.

12 Put icons on the desktop
To alter Gnome install the handy *Gnome Tweak Tool* from your distro's repos. Launch the app, head to the Desktop tab and toggle the Icons on Desktop option.

13 Quick Launch menus
Right-click the icons in Ubuntu's Launcher or an app's name in top bar in Gnome to reveal application specific options and actions.

14 Launch commands from the Mint menu
Right-click the Menu applet, choose Configure > Open the menu editor. Then select a sub-menu or create a new one and select 'New Item'. Enter the command in the space that is provided and toggle the launch in the terminal checkbox for CLI apps.

15 Alter power button behaviour
To tweak the setting of the Power button in the GTK-based Cinnamon, head to System Settings > Power Management and use the power button pull-down to select how it responds.

16 Change Panel Layout
To change Cinnamon's default panel layout head to Settings > Panel and use the Panel Layout pull-down menu to select a different style.

17 Add Applets to Panel
Cinnamon ships with several interesting applets that you can add to any panel by right-clicking the panel and selecting 'Add Applets' to the Panel option.

18 Enable compositing
For some bling, enable compositing on Mate by toggling the 'Enable software compositing window manager' option from under Control Center > Windows.

19 Get different widgets on each desktop
To customise the virtual desktops in KDE, right-click the Pager, switch to the Virtual Desktops tab and toggle the option. Now each desktop can have different widgets etc.

20 Run applications as another user
To get an application running as another user (like root) in KDE, right-click the menu icon and select 'Edit Applications', select an existing entry and click 'Copy'. Then navigate to where you want the new entry, click 'New Item', give it any name and click 'Paste'. Switch to the Advanced tab and toggle the 'Run as a different user' option and enter the username of any user.

21 Slideshow wallpaper
Right-click the KDE desktop and click 'Default Desktop Settings'. Use the Wallpaper pull-down menu to select the Slideshow option and point it to a set of images.

22 Enlarge windows horizontally
Xfce users can right-click the Maximise window button to horizontally stretch it across the screen.

Useful keyboard shortcuts

23 Alt+F2 Bring up the Run dialog box.

24 Alt Search through an app's menu via Ubuntu's HUD.

25 Alt+~ Switch between windows on the same app.

26 Alt+Ctrl+ Up/Down/ Right/Left Switch between workspaces.

27 Alt+PrntSc Take a screenshot of the current window.

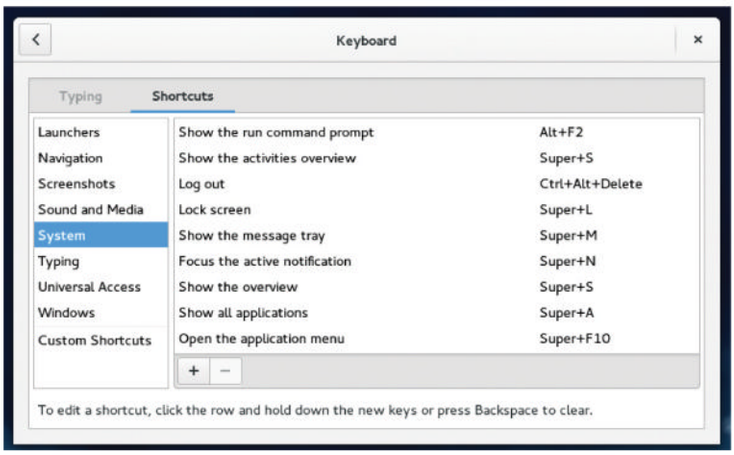
28 Shift+Ctrl+ Alt+r Record a screencast in Gnome.

29 Super+Up Maximise windows in Gnome.

30 Super+Down Minimise windows in Gnome.

31 Super+ Left/Right Snap windows in Gnome.

32 Super+m View any missed notifications in Gnome.



Customise and use keyboard shortcuts to save the time navigating menus.

Tips for your favourite apps

Save time and be more productive with these hidden gems.

LibreOffice

33 Quick change case
Select the words, right-click and head down to Change case menu and select the required option.

34 Enable word completion
Go to Tools > AutoCorrect Options > Word Completion and toggle the 'Enable word completion' and 'Collect words' options.

35 Define Keyboard control
Go to Tools > Customise and click the Keyboard tab to modify any of the shortcuts.

36 Play media files
Head to Insert > Media > Audio or Video and select a media file. Select the media icon in the document to enable media controls.

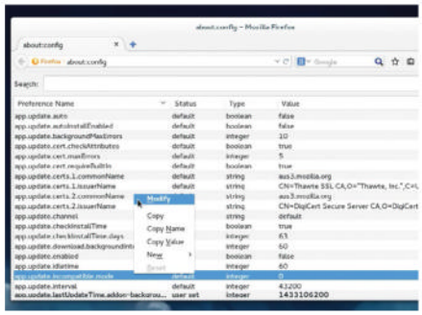
37 Use the Navigator
To swiftly navigate any documents or spreadsheet with the navigator window under View > Navigator.

38 Auto format tables
To auto format them, select some cells and head to Format > Autoformat to choose a different formatting for them.

39 Conditional formatting
Format the cells based on conditions specified under Format > Conditional Formatting > Condition.

40 Protect Sheet
Go to Tools > Protect Document > Sheet to lock access to the sheet with a password.

41 Status bar values
By default the status bar shows the sum of the values in the selected cells. Change the behaviour by right-clicking on the status bar.



› Pop up the hood and take a look inside any Mozilla app with the about:config feature.

Evince

42 Autoscroll PDFs
Right-click inside a document and select the 'Autoscroll' option and use the mouse to control the speed.

43 Make text easier to read
Head to View > Inverted Colors to display white text on a black background.

44 Add Annotations
Select the Annotations option from the drop-down menu in the side pane and switch to the Add tab to add annotations.

Internet apps

45 Speed up the browser (Firefox)
Type `about:config` in the address bar. Then type `network.http` in the filter field and set the network.http.pipelining and network.http.proxy.pipelining parameters to True.

46 Limit RAM usage (Firefox)
Go to `about:config`, filter browser.cache and set the browser.cache.disk.capacity parameter to 30000 if you have 2GB of RAM.

47 Repair folders (Thunderbird)
Right-click the damaged folder, head to Properties and click the 'Repair Folder' button.

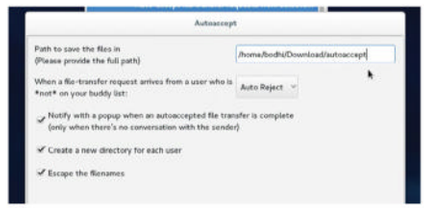
48 Create a mailing list (Thunderbird)
Head to Tools > Address Book > New List and specify which address book list to add addresses to and start adding addresses.

49 Store less mail locally (Thunderbird)
Head to Edit > Account Settings > Synchronisation & Storage for the desired account. Toggle the Synchronise the most recent option and choose the period.

50 Search all messages (Thunderbird)
To search through all mail, including mail only available in full on the server, head to Edit > Find > Search Messages and toggle the 'Run search on server' option.

51 Insert a background image (Evolution)
Toggle the Format > HTML option, head to Format > Page and click 'Browse' under Background image section and pick an image.

52 Advanced search (Evolution)
Head to Search > Advanced Search to create complex search rules. Use the 'Add Condition' button to define parameters.



› Use Pidgin's Autoaccept files plugin to drop files in a folder that you can use with Tip No. 54 to add torrents remotely.

53 Optimise Torrent speed (Transmission)
Use **http://bit.ly/AzureuaUploadCalc** to determine the recommended settings that you can then enter in the Edit > Preferences > Speed and the Network tabs.

54 Monitor directory (Transmission)
Head to Edit > Preferences > Downloading and toggle the 'Automatically add .torrent files from' option and pick a directory.

55 Remote control torrents (Transmission)
Transmission ships with a browser-based interface that can be enabled from Edit > Preferences > Remote.

56 Use a privacy-centric profile (Firefox)
JonDoFox is a *Firefox* profile that automatically integrates with the installed browser and allows you to browse the internet anonymously using a proxy server.

Media players

57 Auto-fetch subtitles (Gnome Videos)
Press Ctrl+Shift+s to open the Movie Subtitles dialog. Now select the language and click 'Find' to look for subtitles on the **www.opensubtitles.org** website.

58 Covert media files (VLC)
Head to Media > Convert/Save, add a file and click 'Convert/Save' button and select the desired codec to convert to.

59 Download online videos (VLC)
Go to Media > Open Network Stream and enter the URL of the video and use the Play pull-down menu and choose 'Convert'. Then select a preset Profile, enter the filename to save and click 'Start'.

60 Record desktop (VLC)
To enable desktop recording, go to Media > Convert / Save > Capture Device. In the Capture mode drop down menu, select

Desktop, then select your frame rate. Finally, click 'Convert/Save', give it a name and click 'Start'.

61 Remote control VLC from a browser (VLC)
Go to Tools > Preference and toggle the 'All' button under Show settings. Now go to Interface > Main Interfaces and toggle the 'Web' option. Then under Main Interface > Lua, set the Lua HTTP Password.

62 Identify a song (Amarok)
Right-click the song you can't recognise, head to Edit Track Details > Tags and click 'Get Tags from MusicBrainz'.

Image editors

63 Move the selection mask (Gimp)
Make a selection, then click the Move tool. Make sure that the Move option is set to 'Selection' in the panel and you can now drag the selection into a new position.

64 Rounded corners (Gimp)
Go to Filters > Decor > Rounded Corners. Then select the 'Edge Radius', which is the amount of curve and optionally customise the other options.

65 Batch process images (Gimp)
Grab and install David's Batch Processor plugin (**http://bit.ly/DavidsBP**) to enable all kinds of tweaks.

66 Automatically write metadata to images (Shotwell)
Head to Edit > Preferences and toggle the Write tags, titles and other metadata to photo files checkbox.

67 Organise photos by events (Shotwell)
By default, *Shotwell* clubs all photos uploaded in one go in a single event. For better organisation you can create new events from a selected group of photos from under Events > New Event.

68 Render RAW files correctly (Shotwell)
To ask *Shotwell* to use the camera's RAW developer, just open an image and toggle the Photo > Developer > Camera option.

KDE apps

69 Bookmarks locations (Konsole)
Use the Bookmarks menu to bookmark any directory. The 'Bookmark Tabs as Folder' option lets you bookmark all open tabs in a single folder.

70 Label tabs (Konsole)
If you've bookmarked a bunch of tabs that you use regularly, you can name them by double-clicking on the tab.

71 Run command on multiple sessions (Konsole)
Use Edit > Copy Input To All Tabs in the Current Window, or Select Tabs if you wish to run the same command, eg on multiple SSH'd hosts.

72 Monitor activity (Konsole)
Enable the View > Monitor for Activity option and KDE will notify you with a popup in the taskbar whenever there's any activity in that Konsole tab.

73 New tab in custom directory (Konsole)
Head to Settings > Edit Current profile and first disable the Start in the same directory as current tab option and then enter the location of the custom start directory in the field above it.

74 Create custom profiles (Konsole)
You can create new profiles with custom fonts and permissions by heading to Settings > Manage Profiles > New Profile. Then customise it by switching to the other tabs such as Appearance.

75 Read-only editor (Kate/Kwrite)
Toggle the Tools > Read Only Mode option to prevent accidentally making changes to an important document.

76 Change highlighting (Kate/Kwrite)
Choose the appropriate highlighting mode for the currently open document by heading to Tools > Highlighting.

VirtualBox

77 Create VM snapshots
To save the current state of a VM, switch to the Snapshot tab in the main interface and click the 'Take Snapshot' button. You can restore snapshots from this interface later.

78 Use USB devices
Head to the Devices > USB Devices and select the USB devices you want to connect which will then be disconnected from the host and made available to the VM.

79 Forward virtual ports
Setup Port Forwarding to ensure any server software inside the VM is accessible from the Internet by heading to Settings > Network > Advanced > Port Forwarding.

80 Enable remote display
If you run *VirtualBox* on a headless server, you can enable the remote display by heading to Settings > Display > Remote Display and toggling the Enable Server checkbox.

81 Manage VirtualBox from a browser
Another useful application for managing *VirtualBox* from a remote computer is *phpVirtualBox*, which recreates the interface inside a web browser.

82 Share clipboard
If you've installed the Guest Editions enable the appropriate option under Devices > Shared Clipboard to copy/paste text between the guest and host.

File manager shortcuts

83 F4 (KDE Dolphin)
Displays the in-line command line.

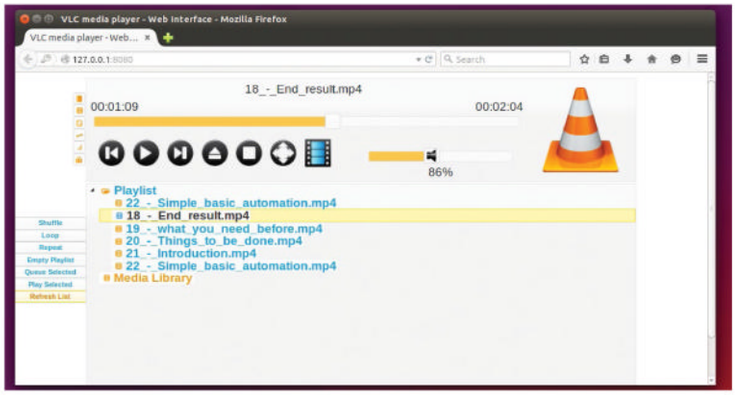
84 F3 (KDE Dolphin)
Splits a single window into two different views.

85 Ctrl+I (KDE Dolphin/ Gnome Nautilus)
View the location bar if hidden (Note: lowercase L).

86 Shift+Enter (Gnome Nautilus) Open the selected folder in a new tab.

87 Ctrl+Shift+ drag the file (Gnome Nautilus)
Creates a soft link to the file.

88 Spacebar (Gnome Nautilus) Preview the selected file if the *Sushi* previewer is currently installed.



Once activated, the *VLC* web interface is available at localhost:8080.

Better manage software

Use the command line to get more from your distro's package manager.

Tips for RPM/Yum/Fedora

89 Install RPMs with Yum
To resolve and fetch dependencies install RPM packages with `yum install <package.rpm>`.

90 Update a particular package
Use `yum check-update <package>` to check for updates for the package which you can install with `yum update <package>`.

91 Search for packages
Use `yum whatprovides <name>` to fetch the name of the package that provides the mentioned file.

92 Install package groups
List all available groups with `yum grouplist` and install any with `yum groupinstall <group-name>`.

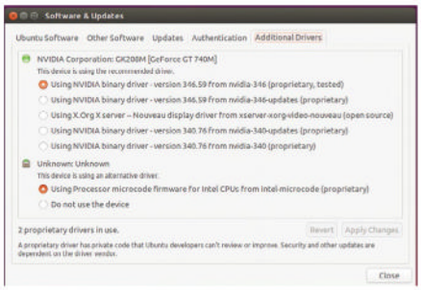
93 Rollback updates
Get a list of actions along with their update IDs with `yum history` and undo one with `yum history undo [update-id]`.

94 Speed up Yum
Install the fastestmirror plugin with `yum install yum-plugin-fastestmirror` and always use the closest mirror to install a package.

Tips for Apt/DPKG/Ubuntu/Mint

95 Backup package list
To install the same packages on another machine, create a list of installed packages with `dpkg --get-selections > pkgs.list`.

96 Replicate on another system
On a fresh installation, first import the list of packages with `dpkg --set-selections < pkgs.list` and then install them with `apt-get dselect-upgrade`.



Use Ubuntu's Additional Drivers tool to install proprietary drivers for your graphics card and other hardware.

97 Uninstall apps
To completely uninstall apps along with their configuration files, use `apt-get remove --purge <app>`.

98 Downgrade packages installed from PPAs
Install the PPA purge tool with `apt-get install ppa-purge` and revert upgraded packages with `ppa-purge <ppa-repo>`.

99 Install dev libraries
To compile a newer version of an app fetch the dev libs of the version in your repos with `apt-get build-dep <app-name>`.

100 Remove archives
Use `apt-get autoclean` to remove downloaded archives of packages that have since been upgraded to newer versions. You can also get rid of them all with `apt-get clean`.

101 Remove unnecessary packages
The `apt-get autoremove` command zaps all dependencies no longer in use.

102 Fix broken dependencies
Use `apt-get -f install` if you get an error while trying to install a Deb package without installing its dependencies first.

103 Use fastest mirror
In Ubuntu's *Software & Updates*, select 'Other' from the Download from the menu and click the 'Select best server' button.

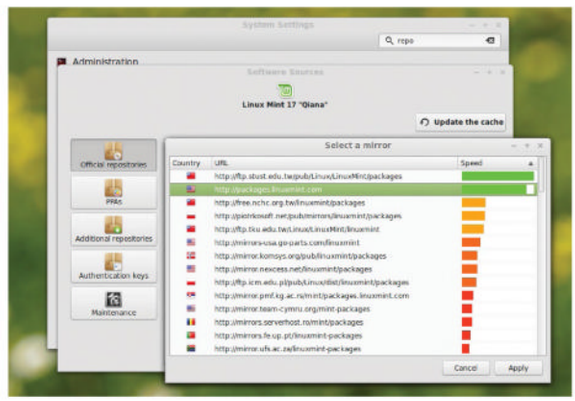
Tips for URPMI/Mageia

104 Fetch a list of dependencies
The command `urpmq -d <pkg-name>` will get a list of required package dependencies.

105 Update all media
Use `urpmi --auto-update` to update the list of available packages.

106 Saves the RPMs
Append the `--noclean` option to prevent urpmi from automatically deleting the downloaded rpms after installing an app.

107 Install from a local directory
Drop RPMs inside a directory and then add it as an installation medium with `urpmi.addmedia backup <directory>`.



Linux Mint has great custom software management tools for easy management of mirrors and PPAs.

108 Install from a URL
Instead of first downloading packages you can install them directly from the web with `urpmi <URL-to-the-rpm>`.

Tips for ZYpp/OpenSUSE

109 List installed packages
The `rpmqpack` command displays a list of all installed packages.

110 Update a package
Use `zypper in <app-name>` to update a package. The command will also install the package if it isn't yet installed.

111 Faster zypper
Use `zypper sh` to enter the Zypper shell which installs packages faster as it keeps all relevant data in memory.

112 Simulate an upgrade
Before you upgrade your installation do a dry run with `zypper -v dup -D`.

113 Backup repos
Save all the configured repos with `zypper lr --export ~/backup-repos.repo`.

114 Restore repos
Use `zypper ar ~/backup-repos.repo` to restore repos from the backed up file.

115 View required patches
Fetch a list of required update patches with `zypper lp`.

116 Install patches
Upgrade apps by applying all available patches with `zypper patch`.

Power user tips

Become a master of your domain.

System administration

117 Monitor remote systems
Launch KDE's *KSysGuard* and go to File > New Tab. Then switch to the new tab and head to File > Monitor Remote Machine and enter the target machine's IP address and connection details.

118 Mount ISO files
Use mount `-o loop <path-to-ISO-file> /tmp/iso-file` to explore the contents of an ISO image.

119 Create virtual consoles
With *tmux* you can create multiple sessions, run different tasks in each, and then switch from one session to another without interrupting the task running inside them.

120 Use tar efficiently
The tar archiver can detect compression formats and `tar xf <compress-file>` is all you need to unpack a file.

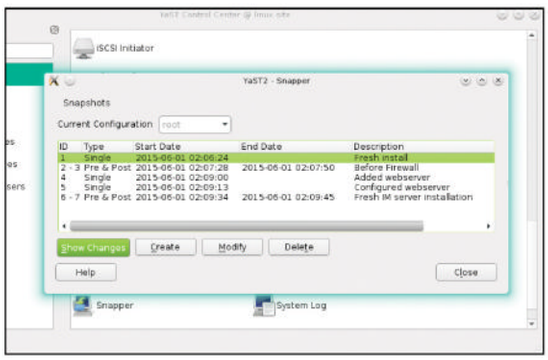
121 Set one-off reminders
You can use `at` with `notify-send` to set short time reminders, such as `echo notify-send "Check on the tea" | at now +4 min`.

122 Schedule a job for multiple times
Use a comma in the crontab file specify multiple times. For example `00 11,16 * * * <task>` executes the task everyday at 11am and again at 4pm (11,16).

123 Run a job within a specific duration
Similarly use a hyphen to specify a range. eg, `00 10-17 * * 1-5 <task>` performs the task from Monday-Friday (1-5) between 10am and 5pm (10-17).

124 Execute a command after every reboot
Use the `@reboot` keyword to run a job whenever the computer starts up.

125 View multiple log files simultaneously
You can install *multitail* from the repos to view multiple file in the following way, eg `multitail /var/log/syslog /var/log/boot.log`.



› OpenSUSE's *Snapper* tool helps you manage snapshots of the distro's btrfs filesystem.

Bash tips

126 View commands matching a pattern
Search through previously executed commands that match a pattern using `history | grep -i <first-few-letters-of-command>`.

127 Reuse arguments from an earlier command
You can use the colon (:) key to reuse the options from the previous command, such as `!!:2` points to the second argument in the previous command.

128 Preview a command before executing
Test your complex *Bash* statements by appending `:p` at the end, such as `ls -l !tar:3:p`.

129 Create shortcuts for commands
You can roll often repeated complex commands into custom ones with `alias`, such as `alias sshbox1='sudo ssh bodhi@192.168.3.111'`. To make aliases permanent add them to the **~/.bashrc** file.

130 Autocorrect CLI typos
You can use `shopt` to autocorrect any common *Bash* typos you tend to create. First, enter `shopt` to display all the available patterns and enable any with `shopt -s`. For example, using `shopt -s cdspell` will find nearest match to misspelt directory names.

131 Create files that are tough to delete
A file with a leading or trailing space in it's name or a hypen (-) cannot easily be zapped from the CLI.

132 Delete tough to delete files
Once you've create a tough file to delete, here are several ways to get rid of files with peculiar names. You can wrap the filename in quotes or use double hyphens, such as `rm " example"` or `rm -- -example`.

133 Delete all files except some
Use the ! operator to remove all files except those that match the specified pattern. For example. `rm ~(*.txt)` will remove all files in the directory that don't end with .txt.

Performance

134 Get details about the hardware
The `dmidecode` command will spit out detailed information about your computer's hardware. For example, using `dmidecode -t 16` will list details about the physical memory. Browse the dmidecode man page for a list of supported DMI types.

135 List process in a hierarchy
You can use `ps --forest` to represent the process tree in ASCII art and clearly identify parent and child processes.

CLI shortcuts

136 Ctrl+a Send the cursor to the start of the command.

137 Ctrl+e Send the cursor to the end of the command.

138 Ctrl+l (lowercase L) Clear the screen but retain what's on the current prompt.

139 Ctrl+k Cut text starting from the command prompt.

140 Ctrl+y Short for 'yank'. Paste the text in the buffer.

141 Ctrl+Shift+c/v Copy and Paste text to the CLI.

Bash shortcuts

142 Shift+PgUp/PgDown Scroll the console.

143 Ctrl+r Search command history.

144 ! <event-number> Repeat a command from history.

145 !! Repeat the last command.

146 Alt+. (dot) Prints the last argument of the previous command.

147 > <filename> Empties specified file.

148 Find memory leaks
To figure out which processes are hogging up the RAM, use `ps --sort mem` which arranges processes in ascending order of memory consumption with the heavy consumers at the bottom.

149 Memory of a particular process
View a detailed memory consumption report of a particular process with `pmap -x <PID>` which displays the amount of resident, non-shared anonymous, and locked memory for each mapping.

150 Trace the execution of a binary
If you have an unknown binary, trace its execution with `stace <binary>` to view all the system calls and signals it makes.

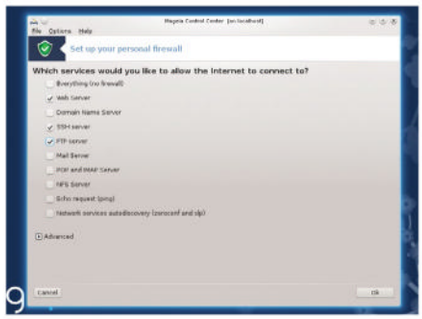
151 Track logged in users
Use the `w` command to get a list of logged in users and their processes. Add the `-f` option to include the hostname of remote users in the output.

152 Kill a graphical app
Type `xkill` in the terminal or the run dialog box which changes the pointer into a cross-hair cursor. Now click on any non-responsive window to kill it. Right-click to dismiss `xkill` without killing a process.

153 Decrease use of swap
If you've got ample RAM, optimise swap usage by editing the **/etc/sysctl.conf** file and changing the value of the vm. swappiness parameter to 10.

Backup

154 Backup the boot sector
A boot sector backup comes in handy when you accidentally wipe out your MBR. Make a backup of a healthy boot sector with `dd if=/dev/sda of=disk.mbr count=1 bs=512` and restore it with `dd if=disk.mbr of=/dev/sda`.



Mageia also includes a Parental Controls for time-based and app-based restrictions.

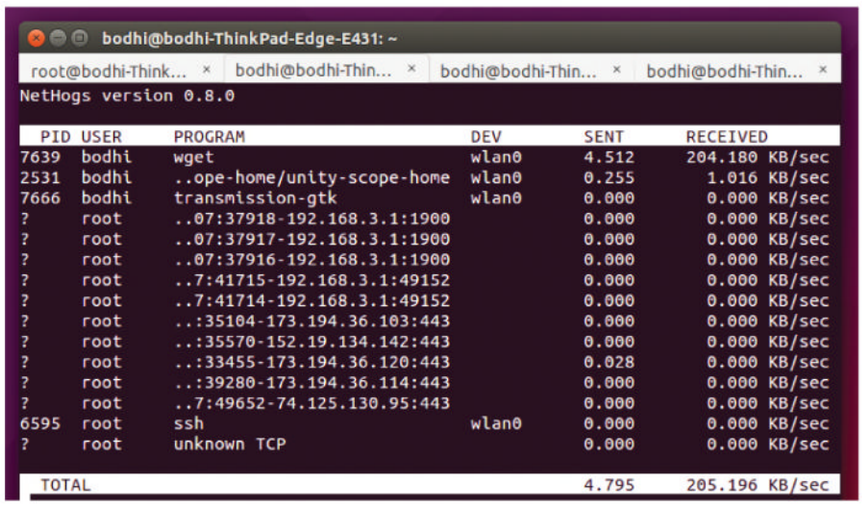
155 Backup partition table
You should also keep a backup of your partition table in the event when a mishap or other zaps this crucial bit of information. Use `sfdisk -d /dev/sda > disk.sf` to backup the table and `sfdisk /dev/sda < disk.sf` to restore the partition table.

156 Monitor the progress of dd
Install the *Pipe Viewer* (pv) tool from your distro's repos and use it to monitor `dd`. For example, `pv -tpreb some-distro.iso | sudo dd of=/dev/sdb bs=4096`.

157 Speed up backups on slower machines
If bandwidth isn't a problem, use `rsync -W` to transfer whole files and save time spent computing the changed blocks and bytes.

158 Track rsync progress
Append the `--progress` option to the `rysnc` command to keep an eye on the data transfer.

159 View changes between source and destination
Use the `-i` option to view the list of items modified by an *rsync* operation, such as `rsync -avzi [source] [destination]`.



Use *Nethogs* to get a real-time view of the bandwidth being consumed by an application.

160 Use rsync over ssh
To transfer *rsync* data over SSH use the `-e ssh` option, such as `rsync -avhze ssh [source] [destination]`.

161 Exclude files
Rsync also lets you skip over certain files that you can specify with the `--exclude` option, like `rsync -avhz --exclude '*.tmp*'` will ignore files with the .tmp extension.

162 Test rsync
First time users should append a `--dry-run` option to all *rsync* operations and scan the output for any unexpected outcomes before running it for real.

163 Limit bandwidth
To make sure the *rsync* operation doesn't hog all the bandwidth restrict its usage with the `--bwlimit option, such as rsync -avhz --bwlimit=50`.

164 Don't backup files on external filesystems
Tar is a popular choice for creating an archive of the disk. Use the `--one-file-system` option with tar to make sure it doesn't backup any mounted partitions (**/media**) or virtual partitions (**/proc**, **/sys**).

Security & Firewall

165 Find which port a program is running on
Use `netstat -ap | grep [app-name]` to see a list of ports that a particular application is communicating from.

166 Disable ping reply
Pings can be used to flood the network and cause network congestion. Disable it temporarily with `echo "1" > /proc/sys/net/ipv4/icmp_echo_ignore_all` or permanently by editing the `/etc/sysctl.conf` file to add `net.ipv4.icmp_echo_ignore_all = 1`.

167 Backup iptables
If you've spent customising the kernel's iptables firewall, make sure you back it up with `iptables-save > ~/iptables.backup`

168 Block a particular domain
First, you need to figure out the domain's IP address with `host -t a www.example.com`. Then use the IP address to get its CIDR with `whois [IP Address] | grep CIDR`. Then use the CIDR to block access to the domain, such as `iptables -A OUTPUT -p tcp -d [CIDR] -j DROP`.

169 Change password for any user
If you've forgotten a password for a user, you can set a new one with `sudo passwd [username]` without being prompted for the current password.

170 Replicate permissions
Use the `--reference` option to copy the permissions of one file to another, such as `chmod --reference=[copy-permission-from-this-file] [apply-on-this-file]`.

171 Securely delete files
Install and use the *shred* utility to delete files so that they cannot be recovered. For example, `shred [file]` will overwrite the file's block with random data several times.

172 Enable built-in Firewall
Some distros such as Ubuntu ship with a simpler front-end to *iptables* firewall, called *UFW*. It's disabled by default but you can enable it with `ufw enable`.

173 Allow incoming connection
UFW denies all incoming connections by default. To tweak this policy and allow connections for common servers do `ufw allow ssh, sudo ufw allow www, ftp`.

Network & Internet

174 Run commands remotely
You can also use SSH to execute commands on a remote machine, such as `ssh [hostname] [command]`.

175 Copy SSH keys to another machine
Use `ssh-copy-id [remote-host]` to securely copy the public key of your default identity to the remote host.

176 Keep connection open
If you frequently get disconnected from remote SSH sessions due to lack of activity, you can enable the KeepAlive option by adding the `ServerAliveInterval 60` line in the **/etc/ssh/ssh-config** file.

177 Browse via SSH tunnel
First create an SSH tunnel to a remote host with `ssh -f -N -D 1080 user@remotehost`. Then change your web browser's Proxy settings and set the SOCKS host to **127.0.0.1** and the port to **1080**.

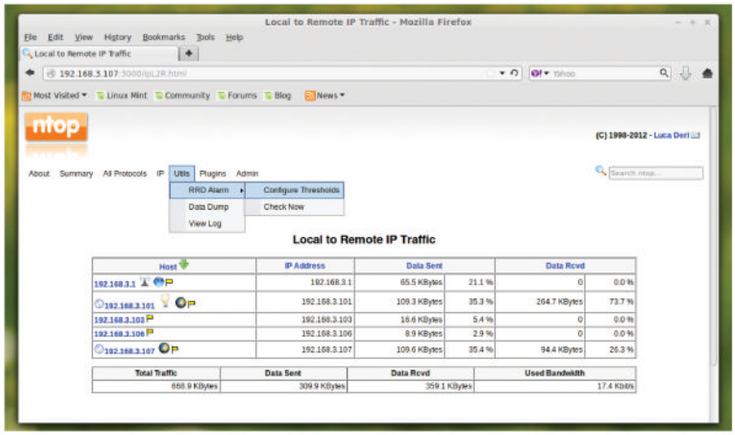
178 Play music over SSH
The command `ssh user@remotehost cat ~/Music/audio.ogg | mplayer` will redirect the output of the remote media file to mplayer on the local machine.

179 Mount partitions over SSH
Use *sshfs* to mount remote partitions such as `sshfs user@remotehost:/home/bodhi /media/remotefs` to mount the remote home directory under the local filesystem.

180 Better monitor network traffic
Ntop is available in the official repos of most distros and gives you detailed analysis of the network traffic via its web-based interface running on port 3000.

181 View network statistics
Use `netstat -s` to view statistics for all protocols or `netstat -st` for only the TCP protocol.

182 Save a webpage
Use `wget` to properly download a webpage. eg, `wget -r -np -k http://www.tuxradar.com/content/dear-edward-snowden` will download all images and change the links in the HTML and CSS files to point to local files.



› ***Ntop* is a versatile tool that can be extended with plugins.**

183 Save multiple files
If you have saved links to multiple downloads in a file, use `cat isos.txt | xargs wget -c` to download them all.

184 Limit data transfer rate
Prevent *wget* from hogging all the bandwidth by imposing limitations, such as `wget --limit-rate=2m` will limit the transfer rate to two megabytes per second.

185 Download files based on modification date
Use curl with the `-z` option to only download files that have been modified after a particular time. For example, `curl -z 29-May-2015 [download-location]`.

186 Upload files
You can use *curl* to connect to a FTP server and upload files, such as `curl -u [user:pass] -T upload.txt ftp://ftp.example.com`.

187 Get definitions
Curl can fetch the definition of a word from a directory server. List them all with `curl dict://dict.org/show:db` and then query one with `curl dict://dict.org/d:shell:foldoc` which fetches the definition of the word 'shell' from the Foldoc dictionary.

188 Simple web filtering
To prevent your computer from accessing a website enter its URL in **/etc/hosts**, such as *127.0.0.1* **www.addictivewebsite.com**.

189 Mirror entire websites
Use the graphical *WebHTTrack* tool available in the official repos of most distros to mirror entire websites and automatically modify links.

190 Regulate bandwidth
You can use *Trickle*, lightweight userspace bandwidth shaper, to control both upload and download speeds. It can also regulate speeds for package managers such as `trickle -d200 apt-get install`.

191 Monitor bandwidth
To monitor bandwidth used by individual network applications use the *nethogs*, a small net top tool that's available in the repos of most distros. Instead of breaking traffic down by protocol it groups bandwith by process.

Top command shortcuts

192 Shift+m Sort by RAM utilisation.

193 k Kill a task from within *top*.

194 1 Track all cores individually within *top*.

195 Shift+w Save the modified configuration permanently.

less command shortcuts

196 / Search forward for a pattern.

197 n Next match.

198 Shift+f Displays new content as it's appended to the file.

199 v Edit the file with your system's configured editor.

200 h View the complete list of shortcuts.

Turbocharge your network

Convert an old PC into a state-of-the-art router with Zeroshell.

If you are responsible for a bunch of networked computers on a small LAN, you can use the Zeroshell distro to rollout various useful network-related services. The Zeroshell distro will transform any computer into a multi-purpose server that offers a lot more services and flexibility than you can wring out of most off-the-shelf routers.

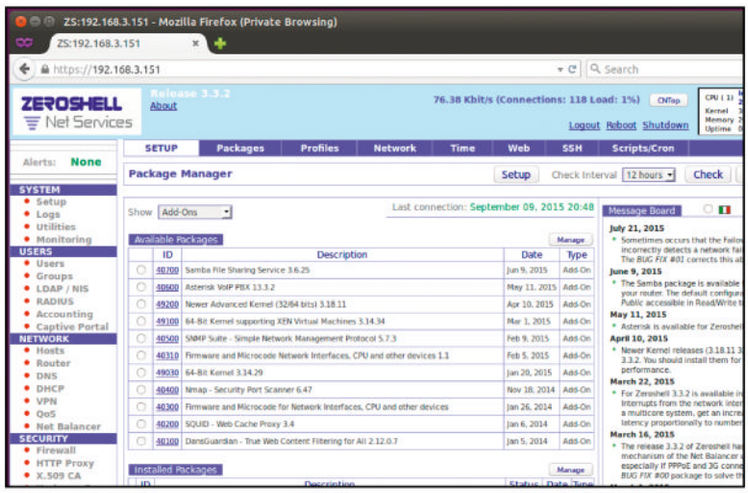
Zeroshell is a small Linux distro that provides various essential network services from DHCP and firewall to VPN and load-balancing. The distro includes a RADIUS server for WPA2 authentication, a Captive Portal instance to create public hotspots and can also be used to shape network traffic and QoS.

The distro has modest hardware requirements and chugs along quite nicely even on an antiquated Celeron box with 1GB of RAM. You can download Zeroshell as an ISO image that you can transfer onto to a CD and install onto the machine's hard disk. Or, you can grab a USB image which will save its configuration locally.

Once you've figured out the hardware you'll use to run Zeroshell, you'll need to decide whether you wish to use Zeroshell to replace your existing router or to supplement it. In case of the former, you'll need to equip the Zeroshell machine with two network cards – one that'll plug into the Internet modem, and the other into a network switch that connects to the other computers on the network. If the Zeroshell server only needs to serve a small number of computers, you can replace the switch with a wireless adapter and turn the Zeroshell machine into a wireless access point.

This is how we'll configure Zeroshell in his tutorial. We'll also keep the router in the equation and connect our Zeroshell server with the router via an Ethernet cable. We can defer the task of doling out IP addresses to the router, which saves us the effort of configuring the routing and DHCP features of Zeroshell and instead allows us to focus on more interesting tasks.

You can extend Zeroshell with several interesting add-ons by supporting the project with your wallets or by promoting it and sending a link to its sole developer.



To get started, boot Zeroshell either from the CD or the USB image. The distro boots up to a customised text-based interface. Before going any further, press [P] to change the default password (zeroshell) for the admin user.

Next up we need to make sure Zeroshell is on the same subnet as the rest of the network. By default Zeroshell assigns itself to the 192.168.0.x subnet. If your existing router is on the same subnet you're in luck. Press [I] and note the IP address shown at the top of the page. That's the address of Zeroshell's web-based interface.

Break the shell

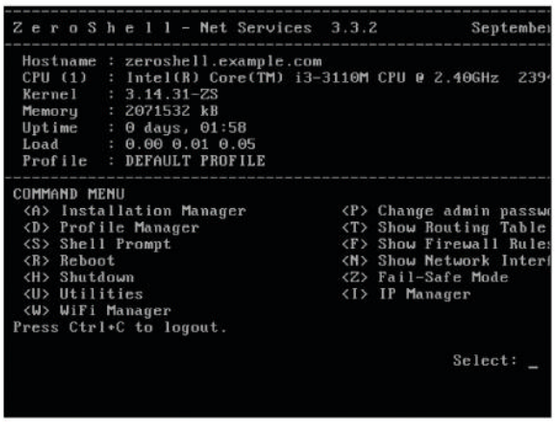
If however you are on a different subnet – let's say your router is located at 192.168.3.1 – then you'll need to change Zeroshell's default address and bring it on the same subnet as the rest of the network.

For this, press [I] to bring up the IP Manager menu. Then bring down the ethernet interface by pressing [S] and following the wizard. Now press [D] and delete the default IP address before pressing [G] to set the default gateway address to your existing router's IP address. In our case, this is 192.168.3.1, and many routers like to sit at x.x.x.1, but yours may be different. Now press [A] to enter a new static IP address for the Zeroshell server, say 192.168.3.151. To bring the changes into effect press [S] to change the status of the ethernet to up. The interface will now change to reflect the new IP addresses. Press [Q] to return to the main menu.

You can now access Zeroshell using a web browser on any computer within the network by pointing that browser at the IP address that you've just set. When it prompts you for login credentials, use the admin username along with the password you defined at the start.

Although the web interface can handle the bulk of its configuration, you'll occasionally need to access Zeroshell's console interface as well. Instead of hopping over to the Zeroshell server, you can remotely access it via SSH. To enable SSH, head to the web interface and click on the SSH tab under the Setup section. In the popup window, toggle the Enabled checkbox. Then enter the subnet of your network (such as 192.168.3.0/24) in the IP address text box and press the + button. Bring the changes into effect with the Save button. You can now ssh into the Zeroshell server from any computer on the subnet with, for example,

```
sudo ssh admin@192.168.3.151
```



Zeroshell lets you save and load different configurations inside profiles which is really handy for experimenting with and testing new features.

Take charge

Next up, let's configure the wireless adapter on the Zeroshell server to act as a wireless access point. For this you'll first need to head to the console-based menu – remember that you can now access this via SSH. In the menu press [W] to bring up the WiFi Manager menu. Once inside press [N] which will kick off a wizard that helps define the settings for the new access point. Zeroshell will prompt you for the SSID of the new access point as well as the encryption mechanism you'd like it to use. While the default options will work in most cases, review each carefully – especially the encryption mechanism.

Once you're through with the wizard your wireless access point should be visible to the devices in the vicinity. However, to hand out IP address to these devices and allow them to browse the Internet, you'll need to create a bridge interface between the wireless adapter and the router that's connected to the Ethernet card.

For this, log in into the web-based interface and head to the Network tab under the Setup section. Then click the button labelled Gateway to make sure the default gateway is set to your router's IP address – 192.168.3.1 in our case.

Close the window and click on the New BRIDGE button. This pops open a window which lists both the ethernet (eth0) and wireless adapter (wlan0) interfaces under the Available Interfaces list. Select each and click the button with the three right arrows to move the selected interface into the Bridged Components list. Do this for both the interfaces, then click Save to activate the new bridged interface. That's it. You can now connect devices to the new wireless access point which will hand out an IP address the same way it takes them to the Internet – via the router.

Furthermore, you can also shield the devices connected to Zeroshell's access point from nasties on the Internet by enabling the Transparent Antivirus Proxy feature. Scroll down to the Security section in the left-hand column and click the HTTP Proxy link. Here, toggle the Enabled checkbox and click the Save button to bring the proxy online. This can take several minutes, since Zeroshell will fetch the latest antivirus definition from ClamAV's website. The Update Log button will help you keep track of the progress.

Once the proxy is active, click on the + icon in the HTTP Capturing Rules section and add two separate Capture Request rules for all traffic passing through the wireless and ethernet adapters. Unless your users are known to frequent the darkest corners of the Internet, you can go easy on ClamAV's server and tune down the number of times Zeroshell checks it for new definitions and updates from the default 12 to, if you're confident, 2. Also make sure you change the default mirror to one that's closer home.

Widespread access

The final feature we're going to enable is VPN access. Configuring an OpenVPN server is quite an involved process which includes pulling in and configuring various pieces of software and generating the appropriate secure certificates. However, Zeroshell ships with OpenVPN, which means all you need to do to use it is to enable it and export the certificates for your clients.

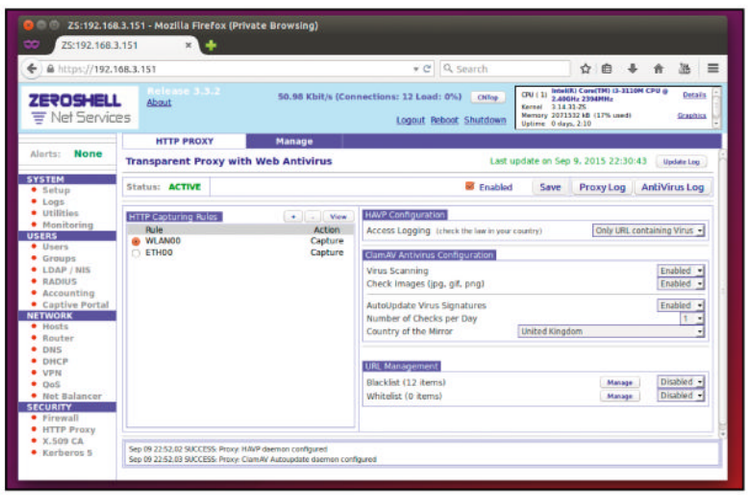
Zeroshell supports different mechanisms for VPN authentication. You can use simple usernames and passwords, X.509 secure certificates, or both – which is what we'll be using. To grab the certificates, click on the Users links under the User section on the left. By default this will list only the admin user. You can use the Add link in the top bar to add more users and repeat the process for each.

For now, select the admin user and click on the tab labelled X509 in the top-bar. From here you can review, revoke and generate a new certificate for the selected user. For the moment though, we'll just save the certificate. Use the pull-down menu to select PEM certificate format and then press the Export button and save the admin.pem file to your local machine.

We'll now grab the certificate for the Trusted Certificate Authority, which in our case is the Zeroshell server itself. Scroll down to the Security section in the left-hand column and click the X.509 CA link. Now switch to the Trusted CAs tab from the top bar, which pops open a window with a list of trusted CAs. Select the only listed entry for our local Zeroshell server and click on the Export button to save the TrustedCA.pem file.

Finally, click the VPN link under the Network section in the left-hand column and toggle the Enabled checkbox. Click on the Save button to bring the server online. That's all there's to it. Now follow the detailed instructions on Zeroshell's website (**http://www.zeroshell.org/openvpn-client/**) to configure your Linux, Windows and Mac OS X clients to connect to your Zeroshell OpenVPN server.

There's a lot more you can do with Zeroshell. Just like OpenVPN, the server ships with a Captive Portal and a RADIUS server installation. All you need to do is enable it and tweak it as per your network.



Along with the antivirus, you can also blacklist URLs from under the transparent HTTPS proxy section.

HTTrack: Website cloning

Need a challenge? Why not quickly get into web development by cloning websites and altering the HTML without prior coding knowledge.

The tool *HTTrack* makes cloning websites quick and easy. But why would you want to clone a website? Well, there are many reasons: for instance, if you've ever had any inclination to build stylish websites, cloning makes it easy to start with a copy of a website you like which you can then modify to suit your needs.

Essentially, almost anything on the web can be captured to provide you a template along with all of the required files (with exceptions of websites created on platforms like Wix and several others). However, you will need to use discretion when making a clone of a website since you likely don't want to violate any copyright laws.

In addition to being a great learning tool, it can benefit website owners as it can be a valuable way to acquire a copy of your website, especially if the web developer is reluctant to hand over the all of the files for some reason or other.

On another level, you can clone from a bloated or outdated CMS platform and end up with lean, fast loading static HTML files. For example, you can clone an outdated *Joomla* or *Wordpress* website to make static HTML files with all images and JavaScript. This is one way around security or compatibility issues of an outdated CMS which are often targets for hackers. Finally, web developers will love this as it's a great tool for making quick copies of your favourite website work, which can easily end up as components in other projects and websites.

At this point, the developer will probably want to hand code, or re-theme the website with a template from a source like Themeforest. The new update from the clone with a more contemporary theme will also likely load much faster than the previous website installation and be easier to maintain, if the changes are only minor.

Although this tutorial is geared to using *HTTrack* for cloning websites, much of the content covered will explain what to do with your clone. Cloning is actually quite easy to do and can you can perform the whole operation within minutes. The more tedious part is knowing how to make changes to the clone. But by the end of the tutorial, you'll have the secrets to creating your cloned template and how to change it to suit your needs. Even if you decide one day to buy a commercial template and build your own website with Linux free of cost, these tips will guide you to getting an excellent finished result. No previous coding knowledge is required for this too, but you will see how to make basic text changes and some basic markup.

Below are a list of commands to install and use *httrack* with a Debian-based Linux operating systems, such as Raspbian and Ubuntu. These installations are required on a clean Linux install. If you already have the installed packages, such as **apache** and **php**, just skip these steps:

```
sudo apt-get update
sudo apt-get install apache2
sudo apt-get install php5 libapache2-mod-php5 php5-mcrypt
/etc/init.d/apache2 restart
sudo apt-get install httrack
Do you want to continue[Y/n]
Type Y and hit Enter on the keyboard.
```

Cloning options

Using the command line, the command below will get the files and links of the main directory. Note: This won't clone the subfolders that could contain data like other HTML or PHP files, shopping cart scripts and more:

```
sudo httrack http://example.com/  -O "/websites/example.com"  -%v -%e0
```

The command below, however, clones a subfolder

```
sudo httrack http://example.com/store/  -O "/websites/example.com/store"  -%v -%e0
```

and while we're covering cloning of a website or subfolders,

Quick tip

To make the most of *HTTrack*, you'll need to get your HTML, CSS and Javascript skills up to scratch as you'll need them to tweak you clone once it's been copied.

for those that want to explore other uses for *HTTrack* you can find a lot more information at **http://bit.ly/HTTrackGuide**.

You should now be able to see the files within the specified directory. In the case above, browse to the **/websites/example.com** folder.

```
cd /websites/example.com
cd example.com
```

Alternatively, you could run *HTTrack* from a browser. You'll need to allow the *Apache* user **www-data** to become a super user and allow it to use the *httrack* package without a password. So, open up that **/etc/sudoers** file and give *Apache* permission to use the desired service(s). Type **nano /etc/sudoers** to open the file:

```
www-data ALL=NOPASSWD: /usr/bin/httrack
```

In order to submit a URL and clone the website, you'll need a basic script that can do this. The code below can do exactly that. The code shown can be copied and pasted into a file stored within the **/var/www** folder. For simplicity, let's call it **cloner.php**. This code will store the clone in the **/home/pi** folder, but you can change the directory to wherever you want the clone to reside:

```
<?php

if (count($_POST) > 0) {
      $url = htmlspecialchars($_POST['url'], ENT_QUOTES,
"utf-8");
      //CREATE CLONE
      shell_exec('sudo httrack http://' . $url . '/  -O "/home/pi/'
. $url . '"  -%v -%e0 2>&1;');
   }

   ?>

   <form name="myform" action="<?php echo
htmlspecialchars($_SERVER["PHP_SELF"], ENT_QUOTES,
"utf-8"); ?>"
         method="post">

      <input class="index-second" type="text" name="url"/>

      <input class="index-fourth" type="submit"
            name="mysubmit"
            value="Clone It"/>
   </form>
```

Just open the URL in a browser, add the URL and submit. The site is stored in the **/home/pi** folder. Depending on the size of the site and files, the clone can take a little while to complete. Often, a website can be cloned in less than a minute. Essentially, using this basic script is a time saver and making the clone can be done by anyone. Let's go on to explain how the script works.

The code contains a simple text box and Submit button, and once the button is clicked, the input is sanitised. After it's sanitised with the **htmlspecialchars()** function, the new variable called **$url** is dumped into the **shell_exec()** function. The **shell_exec()** function is one way to run Linux shell commands within PHP files and by default it's generally enabled, and that is very likely the case with your system. Therefore, if you plan to clone with your Ubuntu, Linux Mint or Rapbian machine at home, you are automatically set up.

However, you may find that the **shell_exec()** function is disabled if you plan to use the script on a web hosting account. The reason that it's often disabled is down to security measures.

Nevertheless, if you have shell access on a web hosting account, you can enable it to use a specified folder or remove it from a blacklist by editing the loaded **php.ini** file. With shared hosting, you can ask if they will enable it for you.

Fixing up the clone

To do a quick check, open the main files called **index.html** in a browser. At this point you should be able to see the web page exactly as you'd expect it to look like. That's a good starting point, but you may want to clean up the code a little since *HTTrack* can leave extra pages and extra code.

For example, your **index.html** file may just be a file that redirects to another page like **index-2.html**. It makes sense to make the redirected page the **index.html** file. It will still work by having two duplicate pages, but, from a web developer's perspective this is messy navigation and having redundant pages is bad practice.

If you happen to know some web design and development and have an editor that can search all files for words or regular expressions, making the change from **index-2.html** to **index.html** is quick and easy. For command liners, you can use the find command to find the files and use sed to change the text:

```
sudo find /websites/example.com/example.com -type f -exec
sed -i 's/index-2.html/index.html/g' {} \;
```

Once you are done with the operation, you will want to change the folder permissions which is currently root, unless your user is in the **sudoers** file.

```
chown -R username:username /websites
```

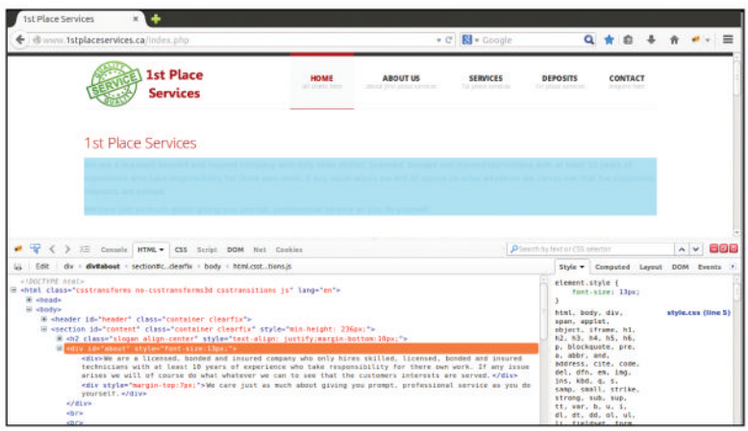
Now, if you have saved **index-1.html** to **index.html**, all links and navigation should work correctly. If you want to clean up the code between the <head> tags, you can still do that too. It may include a comment that *HTTrack* was used to make a website copy. You can remove the entire line that starts with <!-- and end with --> and any other comments that could exist in the header.

Firebug is an extension you can add on to *Firefox* or *Chromium*. However, you'll get many more features using *Firefox* and it's much more efficient to use with this browser.

With Firebug, you can drag your mouse over any web page to work with any desired element on the page. You will have a panel on the bottom and a sidebar on the right. With these, »

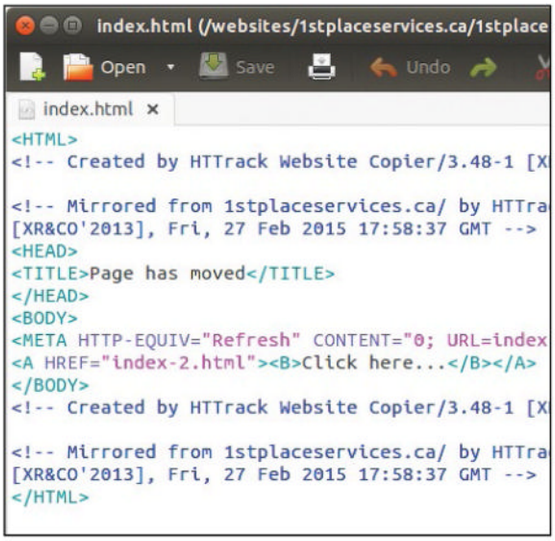
Quick tip

You can add custom PHP/MySQL for dynamic web pages by moving your clone into folder **/var/www** or **/var/html/www**, and renaming the files with a .php extension and changing the menu names to match.



› **Installing the Firebug addon for *Firefox* or *Chromium* makes analysing and changing web page content for display a painless process.**

› ***Gedit*** **is the default Ubuntu text editor. It has a decent colour scheme and can be very efficient for making quick changes to web page files.**



» you can change the HTML and CSS for fine tuning the look of the web page.

In addition to styling and customising a web page, Firebug provides lots of information about images and links. For instance, if you have a clone with absolute path links you want to change them to suit your needs.

From a newcomer's point of view, you can also use Firebug to see the items you want to change or edit. Then, you can look for the match in the website file to make easy changes. After that, just refresh the page and see the changes.

Basic coding

Making changes to a cloned website, such as changing text, links and images is pretty straightforward and can be done with a simple text editor. In Ubuntu, you can do this using *Gedit* or by installing your editor of preference. In other Linux distros you can use the editor that's included or use good alternatives such as *Geany*, *Komodo* and *Netbeans*. For example, you can use *Leafpad* with Raspbian. There are numerous things that you can change, but let's go through a few of the easiest elements to change:

» **Text** Once you open up a file with your editor, you can easily change the text to tailor to your own needs. Until you begin to have a basic understanding of the tags, you should only edit the words you see on the website that you are familiar with.

If you end up getting carried away and start removing tags such as <li>,
, <p>, , <ul>, <img/>, <a> and others, you could end up with a page that looks broken. Again, just stick to changing the words and punctuation until you learn about the other stuff.

» **Links** As with any other template, links are something that will need to be changed. Links have a beginning tag '<a>' and end tag ''. Within the beginning tag is a 'href' attribute that contains the link web address. After that tag closes, the text in between the two tags is the actual text link. Finally, at the other end is the closing tag which makes it complete. A simple link with an absolute URL is shown below:

```
<a href = "http://example.com/page.html">Link text</a>
```

A simple link with an relative URL is shown below. This link to **page.html** is a link where the file **page.html** exists in the same folder as the page which contains the link.

```
<a href = "page.html">Link text</a>
```

» **Images** These are displayed on a web page with a self-closing tag. The code to display an image is shown below:

```
<img src="images/my_image.jpg" />
```

» **List tags** I've included list tags since they are often used in menus. Since the menu of a cloned web page is unlikely to suit your needs, you'll want to know how to change these details. The parent tags for lists are usually . These stand for unordered lists.

The other possible parent list tags are ordered lists . They are usually used to display lists in numerical order, which would not desired in menus.

If you look at the code below and look at the web page in a browser, you can see that the menu items exist between the tags. You can simply delete an entire set, or add them as you need them.

```
<div id="menu">
   <ul>
      <li><a href="index.html">Home</a></li>
      <li><a href="page1.html">Page 1</a></li>
      <li><a href="contact.html">Contact Us</a></li>
   </ul>
</div>
```

Web page structure

We've covered ways to customise a page, but that doesn't actually explain the other aspects of a web page, such as the doctype, head section, body and div tags, so here we go:

» **Doctype** This is located on the first line of the website code. To see any code of a website, you can use Firebug, which we mentioned earlier, or you can use a web browser to view the source code of any web page.

A modern HTML5 Doctype will be at the top of the page

Quick tip

To open and edit your clones on your local Linux machine, you can place them in the **/var/www** folder and access them through the browser with the URL **http://localhost/my_clone_name**.

Altering the PHP configuration

When you make clones with *HTTrack* using a web browser – search the Linux Format site for sample code – your system will end up working with PHP. This adds other factors which need to be considered since PHP using a configuration of its own.

There will be a configuration file that you can always tweak, either on your PC, Raspberry Pi, VPS or dedicated web server, and that file is called **php.ini**.

```
username# php -i | grep "Configuration File"

Configuration File (php.ini) Path => /etc/php5/cli
Loaded Configuration File => /etc/php5/cli/php.ini
username# nano /etc/php5/cli/php.ini
```

Your main concern with this file is to make sure that the script doesn't time out and that the shell_exec() function isn't disabled at any point. The main line that you need to find will begin with **max_execution_time**.

By default, max_execution_time is set to 30 seconds. You can bump that number up and restart *Apache* after saving any new changes, eg, you may want to change it to 5 minutes which is 300 seconds (**60 * 5**). To restart *Apache*, see the following line of code:

```
sudo /etc/init.d/apache2 restart
```

Aside from that, this configuration file also enables and disables functions. By default, you don't have to make any changes because everything is already enabled. But, if you want to have more fine-tuned control of PHP, you can always enable or disable functions by making alterations in this file.

The code snippet below shows an example of disabled functions, including the one this article uses to run *httrack* in the browser.

```
disable_functions =exec,passthru,shell_exec,system,proc_open,popen,curl_exec,curl_multi_exec,parse_ini_file,show_source
```

Firebug primer

Firebug installs in seconds, and as well as being used by a lot of professional web developers, it can also be a very useful tool for someone who just wants to make minor changes to a web page as it can pinpoint the precise location where you need to make your alterations.

To install Firebug, select Tools > Addons > Search for Firebug > Select install. To use the extension, you simply click the icon in the top-right of the *Firefox* browser that looks like a ladybug. After Firebug pops up, click the pointer that shows up on the bottom of the browser window. You should be able to see a window on the bottom and one of the right.

At this point, you can move your mouse around the page and the bottom window will show you exactly the code which is responsible for displaying the relevant content. This will help you quickly locate any page elements that you want to edit. Alternatively, you can right-click on a tag, such as a tag and edit in real time. Then, you can see how the page changes when you add newHTML, such as changing an image.

The left window will display the CSS code for the page. For more advanced designers, you can change many style features here such as widths, paddings, text colour and more.

and look like this: <!DOCTYPE HTML>. It's used with popular HTML5 documents and responsive CSS frameworks, such as Bootstrap and Skeleton. Other Doctypes, like those for XHTML documents will contain more text.

» **HTML tag** After the declaration tag, comes the opening <html> tag. Its closing tag '</html>', is the last piece of text in the entire website.

» **Head tags** The head tags have a beginning and end head tags(<head></head>). In general, all end tags look like the opening tag, but with an extra forward slash. Within the opening and closing head tags, are self-closing meta tags <meta /> with attributes, the <title></title> tags, <script></script> tags and self closing <link /> tags with attributes.

The title tag is important for SEO and displays in the browser when someone views the web page. The meta tags contain info like the page description, keywords, and details for viewing. It can also contain redirects and other data.

» **Body tag** The opening body tag follows the closing </head> tag. The closing </body> tag goes immediately before the closing </html> tag on the bottom.

» **Div and P elements** In a nutshell, the HTML that sits between the <body></body> tags is all of the code that contains the visual aspects of a website. If there's nothing here, the page would be blank. To keep it simple, this code is usually wrapped in opening and closing tags.

With the opening tag, you will often see attributes such as class = "", id="" and style="". These attributes determine how the contents within a div will be displayed on the web page. You can use these attributes to set width, font color, and many other aspects.

» **Comments** HTML comments begin with <!-- and ends with -->. This tag is used for adding notes. It has no influence on how the web page looks at all.

Putting it all together

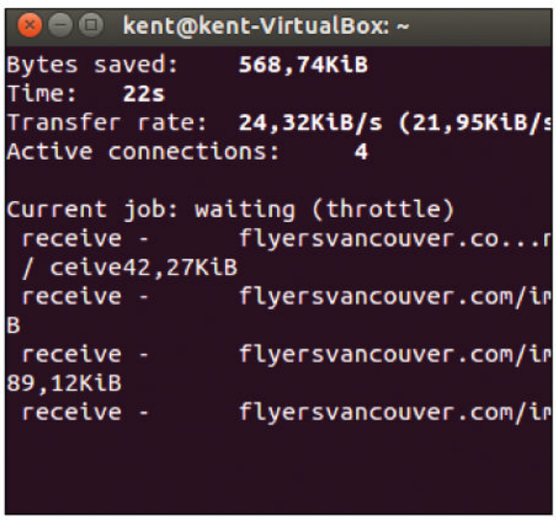
The code below is a simple snapshot of all elements on an HTML page. In a very simplified page, you can see how it pieces together from the top down:

```
<!DOCTYPE HTML>
<html>

  <head>
    <meta name="viewport" content="width=device-width, initial-scale=1">
      <title>My Page Title Goes Here</title>
      <meta name="description" content="My Website Description"/>
      <link rel="stylesheet" href="css/bootstrap.min.css"/>
      <script src="js/bootstrap.js"></script>
  </head>
    <body>
      <div id="menu">
        <ul>
          <li>
            <a href="index.html">Home</a>
          </li>
          <li>
            <a href="page1.html">Page 1</a>
          </li>
          <li>
            <a href="contact.html">Contact Us</a>
          </li>
        </ul>
      </div>
      <div class="container" id="small-row">
        <div class="row" id="small-row">
          <div class="col-md-12"></div>
        </div>
      </div>
    </body>
  </html>
```

And there you have, you are now able to clone the vast majority of websites and go on to edit them in whatever way that you fancy. Although a lot practice is required to be good at coding your own custom web pages, we have tried to get you to a point where you can make common changes that are needed to turn your cloned website into something a little more customised.

For those who would like to go on to build more complex websites in the future, working with clones is a free and a fast way to learn how to build something that actually looks professional straight away.



When you run the httrack package from the command line, you can watch it as it works away at cloning all the different elements of your target website.

Deploy multiple machines

Image and rollout several computers from the comfort of your workstation.

Managing a network of computers is an involved process. Before you can tackle the problem of actively monitoring the machines, you have to install an operating system on each one of them. This is a time consuming task even for a small network with about 10 computers. Computer cloning involves setting up the operating system, drivers, software, and data on one computer, then automatically replicating the same setup on other computers. The technique that's also known as ghosting or imaging is used by system admins for rolling out multiple identical machines over the network without much effort. Fog, which we'll use here, is one of the most popular open source cloning systems.

To use Fog you need to setup an imaging server. The project officially supports several Ubuntu, Fedora, Debian, and CentOS releases, but it's known to work on other distros as well. Before installing Fog make sure the server has a static IP address, which you can ensure from your router's admin page. For this tutorial we'll assume our Fog server is at 192.168.3.51. Also ensure that all the machines in your network are configured to boot from the network card. Finally, make sure your disable any existing DHCP servers on the network as we'll setup the Fog server as a DHCP server and dole out addresses to all the computers on the network.

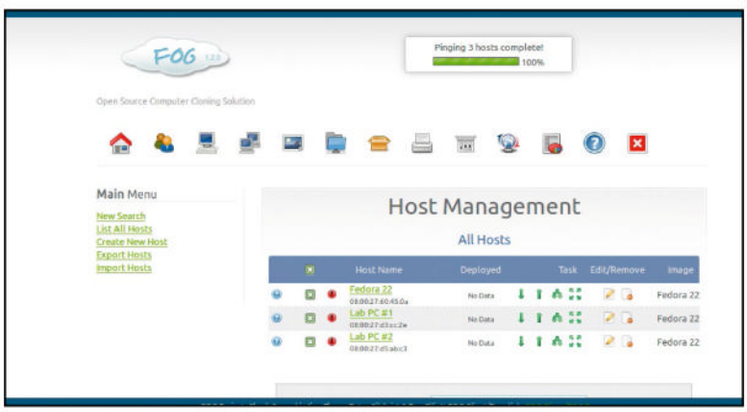
Once you have your network set up, head to the machine that'll be your deployment server and download the latest stable Fog release from Sourceforge (**http://sourceforge.net/projects/freeghost/files/FOG/**). Then fire up a terminal and extract the downloaded tarball with

```
tar xvf Fog_1.2.0.tar.gz -C /opt
```

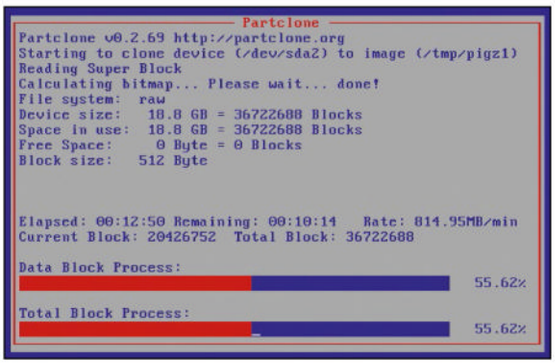
Then change into the bin/ directory under the extracted tarball, and fire up the installation script with

```
sudo ./installFog.sh
```

The installation script will prompt you for several bits of information such as the version of Linux you're running it on, the type of installation, the IP address of the server, the router and the DNS server and whether you'd like to setup the Fog server's own DHCP server. In most cases, it's best to go with the default options suggested by the installer, but make sure you enter the correct IP addresses for the server.



The Fog server's web-based dashboard eases management, even for complex network deployments.



Fog depends on several mature open source tools such as partclone to image a computer.

The script will install various required components. When it's done it'll display a URL for Fog's dashboard (such as **192.168.3.51/fog/management**). Open the link in your web browser and log in with the default credentials (fog:password). On initial launch you'll have to load the default settings into the server's database by clicking on the button on the page. The first order of business when you are at the proper administration dashboard is to create a new user. To do this head to User Management > Create New User.

Create a base image

Now that our imaging server is set up, we'll use it to image a computer. Once a computer has been imaged we can then deploy that image to other computers with a single click.

To begin the process, fire up a browser on the imaging server and head to Fog's dashboard and log in with the default credentials. Then head to Image Management > Create New Image. Use the fields in the form to describe the image. For example, let's assume we are creating an image of Fedora Workstation 22 installation that we'll then install on all our computers in the Science Lab. So we can name the image 'Fedora for Science Lab'.

Next, use the Operating System pull-down menu to specify the operating system of this image, such as Linux. Finally, select the correct disk layout scheme from the Image Type pull-down menu. Our Fedora installation is on a single disk with multiple partition so we'll select the second option.

Advanced Fog features

Fog is a complex piece of software and while we've covered the core feature of the server, it ships with several more. The Fog server is scalable and can manage large networks spread over multiple locations in the same building or around the world. It allows you to arrange hosts into several groups for easier management. One of the most useful features of the Fog server, especially for admins of larger networks, is the multicast ability. Using this feature you can deploy multiple machines in one go.

However, to use this feature successfully you'll need to make sure your Fog host has enough computational and network resources to stream multiple images simultaneously. For such larger networks, you can have multiple Fog installations configured as storage servers. These storage servers share images and take the load of the main Fog server when imaging computers. The distributed storage servers also speed up unicast transfers and introduce data redundancy.

Besides the two most important Fog server tasks that we've covered in this tutorial (upload and download images), you can create several different tasks for any of the hosts in Fog's repository. You can run the Debug task which boots a Linux image to a bash prompt for fixing any boot errors. You can also create a task to remote wipe hosts, to recover files with TestDisk, or to scan for viruses with ClamAV.

The Fog server can also install and manage printers on the network. Depending on the OS on the host you can also use the server to track user access to computers by their Windows usernames and automatically log off users and shut down the computer after a specified period of inactivity. Fog can also install and uninstall apps via snapins.

Now assuming you've already installed Fedora on one of the computers on the network, head to that computer and boot it up. Since the computer is set to boot from the network card, it'll display the PXE boot environment from the Fog server. Scroll down the Fog menu and select the 'Quick Registration and Inventory' option. The Fog server will now scan the computer and add it to its repository of known hosts.

Upload an image

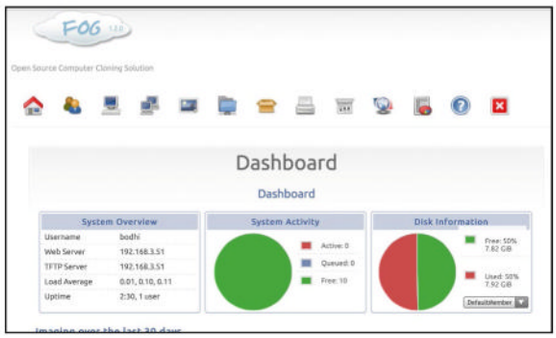
When it's done, shutdown the Fedora computer and head back to the Fog server. Fire up the dashboard and head to Host Management > List All Hosts. The Fedora server will be listed here. By default Fog identifies each host by its MAC address. You can change it to something more meaningful (like 'Fedora 22') by clicking on the edit icon. Here you can change its name and add a brief description to identify this computer. Most importantly, use the Host Image pull-down menu and select the Fedora 22 image you created earlier.

Now that our basic framework in ready, it's time to image the installed Fedora installation. Head back to Task Management > List All Hosts which will list your rechristened Fedora 22 installation. Under the Task section corresponding to this image, click on the green upload arrow. Fog will give you multiple options to schedule the upload task. You can explore the options after clocking some mileage with Fog but for now it's best to go with the default option for instant deployment.

Then head back to the Fedora machine and boot it up. It'll again detect Fog's PXE and automatically image the machine and upload it to the Fog server. The process will take some time depending on the size of the disk it has to image, the processing capabilities of the computers involved and the speed of the local network.

The Fedora computer will restart once it's done uploading the image. You can now use Fog to deploy this Fedora image on all the lab computers with a single click! You can similarly image any other computer on the network, including the new Windows 10 installations.

Once a host is registered you can query its hardware and get compatibility information before imaging it.



You can deploy and image your computers by accessing Fog dashboard from a mobile device like a tablet.

Deploy the image

Before you can deploy an image, you need to register the targets machines as hosts with the Fog Server. The registration process is the same as before. Boot the new computer from the network which should detect Fog's PXE environment. When it does, select the 'Quick Registration and Inventory' option.

When the computer has been added to Fog's repository of known computers, login to the Fog dashboard and head to Host Management > List All Hosts. Click on the edit icon corresponding to the newly added machine and rename it so that it's more identifiable, something like Lab PC #1. Again, remember to use the Host Image pull-down menu to select the Fedora 22 image that we've just imaged from another computer. Repeat the process to register all the computers in the lab with the Fog server. Then edit them in the Fog dashboard to give then an identifiable name and select the Fedora image as the host image.

Now to replicate the Fedora image on to the other lab computers, head to Task Management > List All Hosts. Browse the list of hosts to find the entry for the computer you wish to deploy to and select the corresponding down arrow Deploy image option. After the deploy task has been created, head to the lab computer and power it on. It'll automatically detect the task from the Fog server and copy the image from the server on to the local machine. When it's done, you'll have a mirror copy of the Fedora installation on the Lab computer. Repeat the process to deploy Fedora on other Lab computers as well.

DD-WRT: Hack a wireless router

Find out how to power up the device at the heart of your home network with your very own software for a truly custom spin on routing.

Nowadays a decent router can be relied on to do its own thing without bothering you, making it a great time for home networking. However, it can still be a challenge to get it to do your particular thing instead. If you're ready for a change, the world of custom firmware opens up an embarrassment of configuration choices, as well as an enticing catalogue of new functionality.

With DD-WRT as our firmware of choice, we're going to firmly encourage these sleek and unassuming embedded devices to reach their full huffing, wheezing potential. There will be sweat, there may be tears, but we'll guide you through the process of selecting and installing a firmware, show you some of the nattiest ways to trick it out, and open the door for your own challenges to follow.

DD-WRT is one among many custom firmware for wireless routers, but it beats at the heart of the custom firmware movement, with a broad range of support, relative ease of use, consistent development and a treasure trove of features. Installing DD-WRT isn't a minor tweak, though – it will completely rewrite the way your router operates, potentially opening up functionality, such as: SSH; file and media serving; guest networks; QoS; VLANs; and VPNs in more flavours than you could find in a bag of Revels. However, there are risks commensurate with the scope of the change.

Quick tip

There are other firmware that might reasonably vie for your attention. In particular, various forks of the Tomato project, Merlin's take on AsusWRT (a halfway house between custom and stock firmware strictly for Asus routers) and even OpenWRT, which is the expansive base on which many others are built.

While installing a custom firmware is almost always a beautiful learning experience, sometimes what you learn is how it feels to break a perfectly good router. It probably won't even seem like it's your fault when it happens, but implicit in your willingness to continue is the understanding that it will be your fault, because you were the one who poked it.

Now that's clear, we can continue and the most advisable way forward is to use an older, spare router. Look at it this way – you're going to end this process without a manufacturer's warranty, so you may as well start it without one. You're also less likely to feel a sense of gnawing, visceral guilt if you sneeze and pull out the power adaptor during a firmware update, and proportionally more likely to unlock new features. By contrast, it can take a reasonably long time for custom firmware such as DD-WRT to adapt to new technology (and longer still to make it run reliably), so you may be on a hiding to nothing with this year's super router, even if you're cavalier enough to try it.

Router support

We'll deliver the bad news up front. With no notable exceptions, combination router/modems won't work – BT's famous range of Home Hubs, for example, aren't supported. But all is not lost if you're on VDSL/BT fibre, because you should be able to arrange to use a standalone OpenReach modem instead, and connect up a router of your choice. Other ISPs' combination devices may even have a modem-only mode that enables you to plug in your own router – eg, Virgin Media's Super Hubs fall into this category.

If you do have a standalone router, you can't necessarily just go ahead and plonk a new firmware on it. Some routers don't have the right chipset, some don't have enough flash storage, and some don't have the RAM. Some, frankly, don't have the moxie. All that said, a surprisingly wide range of routers are supported. So how do you know whether yours is one of them?

Your first port of call should be DD-WRT's router database (**www.dd-wrt.com/site/support/router-database**). Simply put your model number into the search field, and then cross your fingers. The database will usually give you a straight yes or no answer, but don't jump for joy when you see your model appear in this list until you have checked that the revision column also matches up with your router – some manufacturers change out the internals almost completely between revisions of the same router model.

Just for fun, try searching for the WRT54G in the router database, and count the iterations. The WRT54G is the granddaddy of DD-WRT, and it has a lot of history. But note that at least one revision isn't supported at all, and that the specs can be wildly different between others. Many have

reduced flash storage space, for instance, and will be limited in which features they can support.

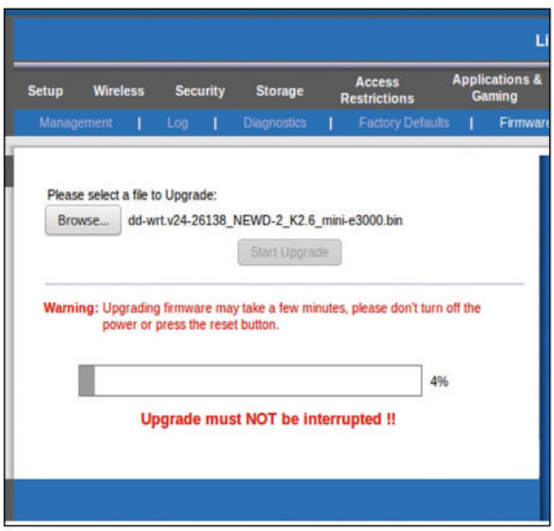
Once you've established that your router is supported, there are two major lights in the darkness: DD-WRT's wiki, and the community forums. The wiki is great for getting a baseline understanding of any issues which might affect your particular router. Start with the Supported Devices page (**www.dd-wrt.com/wiki/index.php/Supported_Devices**). Links from this page often indicate that your router has a specific installation guide, which might just mean that it's a popular model, but it could mean that the flashing process comes with some caveat or special requirement, so be aware.

Firm forum friends

The forums are the best place to find out what's working, right now, for other people using the same hardware (see **www.dd-wrt.com/phpBB2**). You should pay particular attention to threads where users trade blows over their favourite or most stable builds. Look out for the guru posters, who often have long signatures containing details of the many different routers they run, and which firmware versions they're running on them. These guys have done their homework, so make sure you do yours, too, even if that sometimes means leaning across the metaphorical desk to copy their notes.

DD-WRT exists in an ongoing beta, and the newest release is not always going to be the best one for your own particular hardware. There's no shame or loss in using a build which might be significantly behind the bleeding edge. If it's the right fit for your kit, just go for it. With older releases, the main thing you need to concern yourself with is to make sure that you're not exposing yourself and your hardware to any critical security flaws. As a starting point, build revisions between 19163 and 23882 are a poor vintage; any components making use of OpenSSL will be affected by the Heartbleed bug. The good news is that none of the vanilla builds are affected by the *Bash*-specific Shellshock vulnerability; like many embedded device firmwares, DD-WRT relies on *BusyBox* to provide A Shell.

Likewise, the use of uclibc means that the glibc Ghost vulnerability is no concern for today. However, running a custom firmware does tend to send the security ball zipping over the next into your side of the court, so you really do need to keep abreast of any emerging vulnerabilities.



Now is the time for a moment of quiet reflection...

The make or model is usually on a sticker, either on the back or the bottom of your router. Note any version information in addition to the model number.

Now, let's go through a worked example. We have a Cisco Linksys E3000 router, which treads a decent balance between age and relevance. It's around five years old and there's none of that new-fangled wireless AC technology, but it was a powerhouse in its day, with support for simultaneous 2.4GHz and 5GHz wireless bands. The router database shows a firm yes, and there is some specific information on the wiki relating to it. Particular points of note are the implications of it having 60K of NVRAM, and the requirement to flash a trailed build (*see overleaf for the box: Trailed Builds and TFTP*). We'll need to take both of these things into account.

We're lucky, as it happens; on the forums, a build from February 2015 (build 26138) is being touted as stable with the Linksys E series. There's some debate about a bug in the Guest Wi-Fi implementation, but it sounds as though it's going to be worth our time.

The main area for new DD-WRT releases is at **ftp://ftp.dd-wrt.com/betas** and we know from the wiki that E3000-compatible builds are to be found in the broadcom_K26 subfolder. We can pick a mini-trailed release for the E3000 from here with no problem, so we'll get that now, but if we want to move to a larger general build afterwards, then we'll need to remember our 60K NVRAM limit, and pick one of the 60K builds from the same folder. The mega 60K build is (just!) too large for our 8MB flash storage – It's a good job we checked that out, because it came down to counting the bytes – so we'll go with the so-called 'big build' instead.

Firmware update time

Now it's time for us to check and double-check all our sources of information, because we're ready to do the firmware update. The steps that follow are usually applicable, but you should read up on your model to see where any differences might occur.

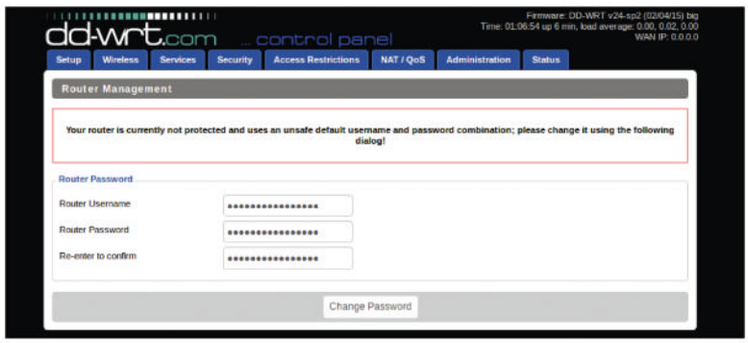
First, you need to connect your computer to the router using a wired connection, and then configure it to have a static IP address on the same subnet as the router. Things are not guaranteed to go wrong if you don't do this, but do you really want to leave the router in charge of business while you're in the process of brainwashing it? The answer is a definite no. No, you don't.

Quick tip

DD-WRT gives you control but it doesn't necessarily give you performance. If blazing fast speed is the only thing that interests you, a manufacturer's own firmware is often faster than the custom alternatives.

Warning

Following this tutorial can destroy your hardware. Future PLC accepts no liability (including through negligence) for any damage, loss of data or costs you might incur as a result of this tutorial. Use this at your own risk.



Success for your firmware update will look like this.

Do a 30-30-30 reset (*see box opposite*), and then log in to your router's web configuration page (with the now factory default username and password). Find wherever your manufacturer has hidden the firmware update section, and browse your computer to find the DD-WRT firmware file you prepared earlier, which is probably a trailed build specific to your router.

Go ahead and do the update using the built-in firmware updater. There may or may not be a progress bar, but ignore it either way. You're going to wait at least five minutes. Use a clock and not your patience to gauge this. Then turn the router off and on again, giving it time to reboot and get its bearings – then, and only then, do another 30-30-30.

Open up a web browser and go to **192.168.1.1**, which is the default IP address for a DD-WRT router, and check that you are indeed looking at a DD-WRT interface. That's the first good sign, and the second is whether it's asking you to change the password, which shows that the 30-30-30 reset after the update has also worked properly.

If all is well, decide whether you're sticking with the build you've just installed or, if you were using a trailed build as an intermediary step, repeat the process again in full, until you have reached your final destination.

Quick tip

NVRAM is the persistent memory in which variables are stored between resets, and it's measured in kilobytes. The more features you use, the more variables you store (VPN certificates are particularly greedy). It's both a limiting factor and a risk; if you reach the end of your NVRAM allocation and keep writing, you can reset or even brick the device.

Configuration work

Now that you're up and running, feel free to do some basic configuration. Get the router set up the way you like it; that's what we came here for. DD-WRT's interface is neat and functional, and you should be able to find the options you're comfortable with, albeit buddying along with a raft of new features. Get your wireless security set up, and then give it a test drive. Now are you ready to try something that you couldn't do before?

How about logging directly into your router via SSH? Yeah, we can do that. We can even do it without a password, using the public key method. To generate an appropriate public/private key pair, enter the following into a terminal on your local machine.

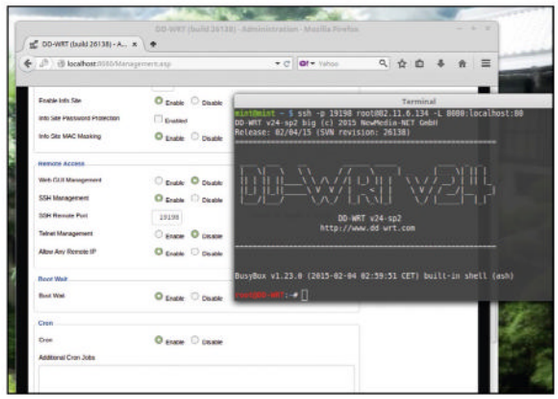
```
ssh-keygen -t rsa -f ~/.ssh/id_rsa_ddwrt
```

You're prompted to set a passphrase, but hitting Enter twice enables you to continue without – choose your balance of security and convenience. Two new files are created under your home directory, in the **~/.ssh/ hidden** folder: **id_rsa_ddwrt** and **id_rsa_ddwrt.pub**, which contain your newly generated private and public keys, respectively. Make sure you keep prying eyes away from the private key, but we'll use the public key to set up easy password-free access to your router.

Go to the Services tab in your new DD-WRT Web GUI, and then click the enable checkbox for SSHd. This expands some new options. It's up to you whether or not you leave password authentication active, but what you do want to do is copy the contents of your **id_rsa_ddwrt.pub** file into the Authorized Keys box. Make sure the entire sequence occurs on a single line. Save and apply these changes. At this point, one simple terminal command on your local machine should let you in through the door:

```
ssh root@192.168.1.1
```

Substitute in the correct local IP of your router, if you've changed it. If you see the DD-WRT message in the terminal, well done, you're in. But you didn't think we were going to stop there, did you? Getting the local access is only half the battle. How about an interesting and powerful way to manage your router from the outside world? Remote access to your router is always going to be a controversial subject but, let's be honest, sometimes it's useful enough to be worth the risk you are taking doing it.



Note that the only user for SSH is root, regardless of what username you set for the Web GUI. The password is the same, if you're using one.

Trailed builds and TFTP

A trailed build could quite accurately be described as a *custom* custom firmware. It's a firmware that's been built specifically for one particular model of router (which is mentioned in the filename). Trailed builds contain headers that check out as legitimate with the manufacturer's own firmware, which then conveniently and quite cleverly enables you to use the existing interface to overwrite itself. A trailed build might not be your end point, however, but more like a transitional step between using stock and custom firmware. Once you have installed a trailed build of DD-WRT, you're generally able to move more freely between different firmware builds – you still need to pick the correct ones, though.

Now let's take a look at tftp, which is quite literally a trivial file transfer protocol. This is necessary for the initial flash of a few routers – older Linksys, Buffalo and Belkin models being the prime examples. It's comparatively rare to require this on Wireless N or newer routers. If you don't need to use tftp, then it's not recommended, regardless of whether or not it's available.

However, it's worth remembering that lots of different routers have a tftp connection available for a limited window during the boot process, because it could be one of the first ports of call if you need to try to recover from a bad flash. Although it's never to be relied upon, it may help bring you back from the brink in a pinch.

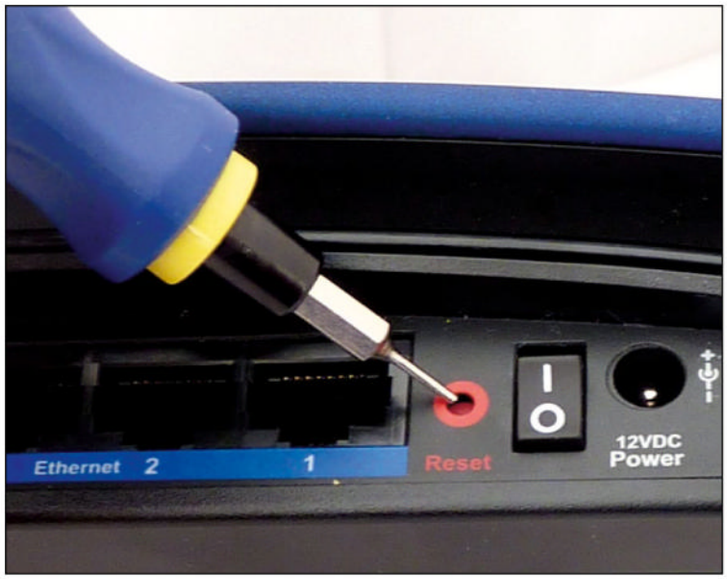
The 30-30-30 reset

Don't underestimate how skew-whiff things can become when the vestigial variables of firmware A come out to play with custom firmware B. The 30-30-30 is a catch-all hard reset for clearing NVRAM and returning most routers to their firmware defaults, which you'll do before and after flashing each new firmware version.

Your router's Reset button is probably on the back of the unit, sometimes inset. Grab a paperclip if you need one, and get into a comfortable position; you are going to be holding your router's reset button down for 90 seconds or more, which is a long, long time for someone with cramp.

Start holding down your router's reset button, and count a full 30 seconds. Not letting go of the reset button, pull the AC plug out of the back of the router. Count 30 more seconds. Keep holding that reset button, and plug the router back in. Count 30 more seconds. Finally, let go of the reset button and throw up some jazz hands to brighten the mood and get your circulation flowing again. Your router should be back to default values for whichever firmware you currently have installed. (You can put your hands down now.)

A handful of older routers, but an ever-increasing number of new AC routers, need to be hard reset in other ways. If the 30-30-30 doesn't return yours to default values, check what does work for your router, and use that method instead.



Yes, you can buy a tool that does the job of a paperclip.

DD-WRT will happily support remote access to the GUI via HTTP or HTTPS. There's no way in this life that you'd want to give the world a shot at the core of your home network without a single security layer, but you might be thinking about allowing HTTPS connections.

Wait, though. Here's a neat trick instead: why not disallow remote Web GUI access altogether, and just connect via SSH? You can then log in and administer the router remotely by command line or set up an SSH tunnel to give you, effectively, local access to the Web GUI. This will work from any location – and you only have to open one door to enable both types of access. Let's look at how this can be done.

First, setting up the remote access to SSH is done in a different part of the DD-WRT GUI to enabling the service. This time you'll want to go to the Management tab under Administration. There's a remote access section here. Don't bother enabling the remote Web GUI Management. Instead, enable SSH Management. You're given the option to select a port for this. You don't need to – and, in fact, shouldn't – use the typical SSH port 22; we'll use port 19198 in this example. We made this up so feel free to make up your own, but don't worry – the connection made on this port will forward through to the SSH service on your router without any extra work on your part.

Now you can SSH to your router from the outside world, in the same way that you do from your local network – the only differences are that you need to specify the port, and use the outward facing IP rather than the local one:

```
ssh -p 19198 root@WANIP
```

You should replace WANIP with the global address of your local network. This can be a DNS name, or an IP address. In the highly likely event that your ISP doesn't provide you with a static IP address, you won't necessarily need to keep track of every change of IP address. DD-WRT supports automatically updating of a number of different dynamic DNS services – take a look at DDNS under the Setup tab for the various options.

So we've come this far, but what about that Web GUI? Well, try starting your SSH session with this command:

```
ssh -p 19198 root@WANIP -L 8080:localhost:80
```

This starts an SSH session as before, but the last part of the command creates a tunnel from port 8080 on your local machine, to port 80 on the router. Now try opening a browser window to the following URL: **http://localhost:8080/**

Wow. Presto. There it is. You've got your Web GUI from a remote location, and it's all encrypted through your SSH session. Now the world, quite literally, is at your disposal.

The gauntlet

Now you've got access via the Web GUI and SSH, what new things are worth trying? Actually, what new things are not worth trying? If that sounds like a challenge, read it as one.

How about building on the SSH tunnelling method we looked at, to have your home router run a SOCKS5 proxy, via which you can encrypt your traffic when you're away from home? If you've got a VPN account, how about connecting with your router as the client? (This can be great for hooking up other, less hackable embedded devices which might not support VPN natively.) Maybe you have a USB mobile broadband dongle? DD-WRT can play with those. Why not try creating an alternative internet feed through your router, for those days when your main ISP connection dies?

If you really want to start playing with fire, you might even find a way to host your own cloud-style file service from a USB hard drive, hanging off the back of your router. It's not like you were planning on turning your router off, were you?

So there we have it. Some absolutely astounding possibilities that would previously have taken all kinds of wizardry to arrange, running on something you probably already had sitting in a cupboard. Remember that routing network traffic is this device's bread and butter, so don't be afraid to make it earn a living!

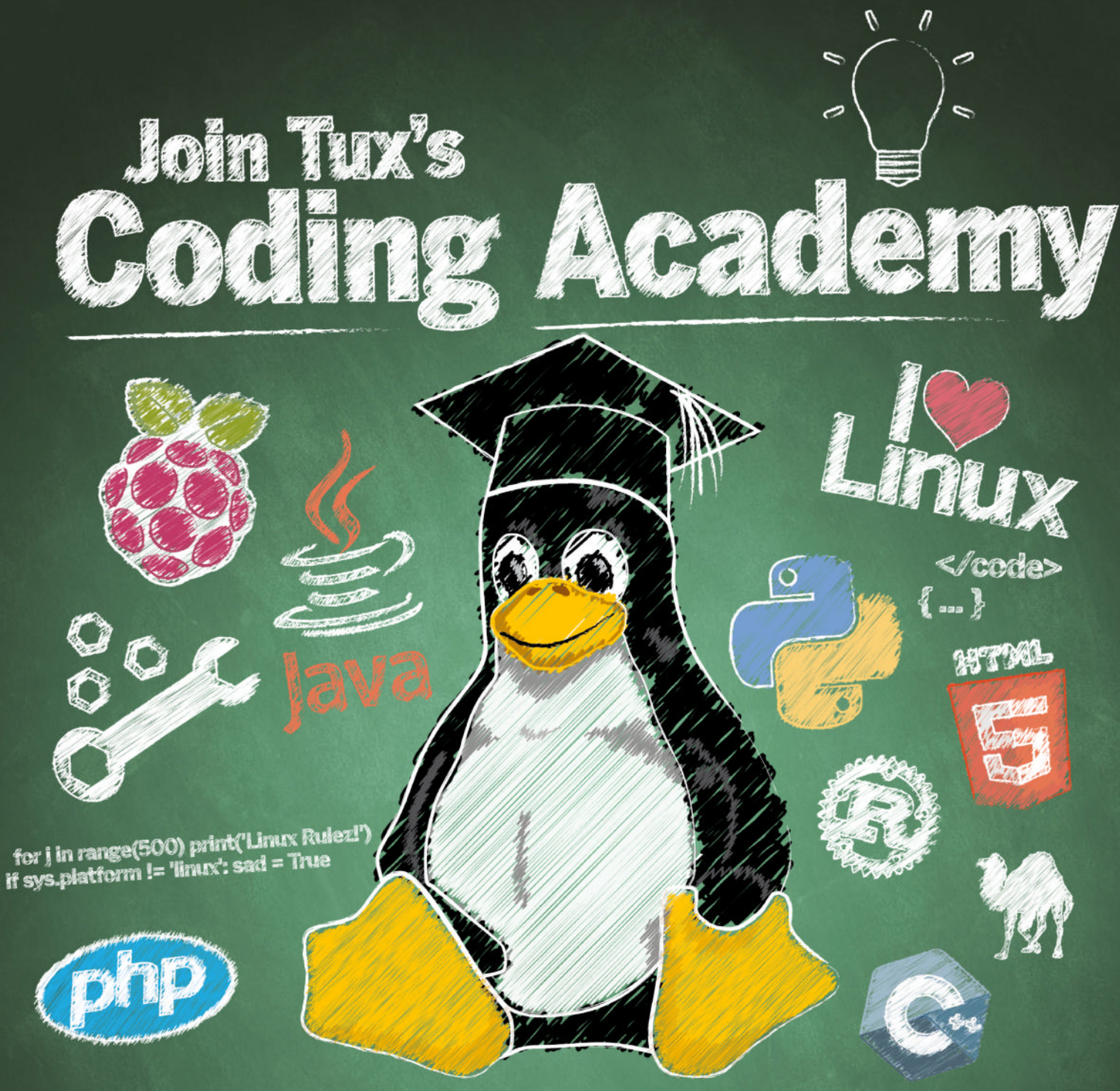
HACKER'S MANUAL 2016

HACKER'S MANUAL 2016

Coding

Programming skills are the mark of the complete hacker.

Join Tux's Coding Academy



Coding isn't scary. Promise. If you've always wanted to learn, there's no better time to get started.

Coding is the new cool. If you don't know how to Python, all your friends are going to be laughing at you...

We're not being flippant. Knowing how to code is almost an essential for the open source fiend. Be it basic Bash scripting or knowing how to read a bit of PHP, coding knowledge is more than useful, it's an essential skill, particularly now that coding is being taught in schools.

Knowing a bit of code helps with using Linux itself, helps you get more out of the terminal, can open doors to a new career or help you troubleshoot why that webpage won't work correctly. Other than taking a bit of time to pick up, there's also no cost and as a FLOSS user, access to every language under the sun is just waiting an **apt-get** away.

So take our hand and let's take just a few minutes to create a fun game in Python, learn which languages are right for you and your projects, then see how you can tackle the new school curriculum and even web development.

Get with the Program

This is going to be a very gentle introduction to programming in Python in which we'll explore the basics of the language, then use the *Pygame* module to make a simple game. That we are able to do this is testament to the power of Python and *Pygame*: much of the tedium inherent in game work is abstracted away and, once we've covered some elementary programming constructions, for the most part we work with intuitive commands.

First follow the instructions in the box to check which version of Python you have and install *Pygame*. Whether you're on a Raspberry Pi or a larger machine, start at the command line. So open up a terminal (*LXTerminal* in Raspbian) and start Python 2.7 with

```
$ python2
```

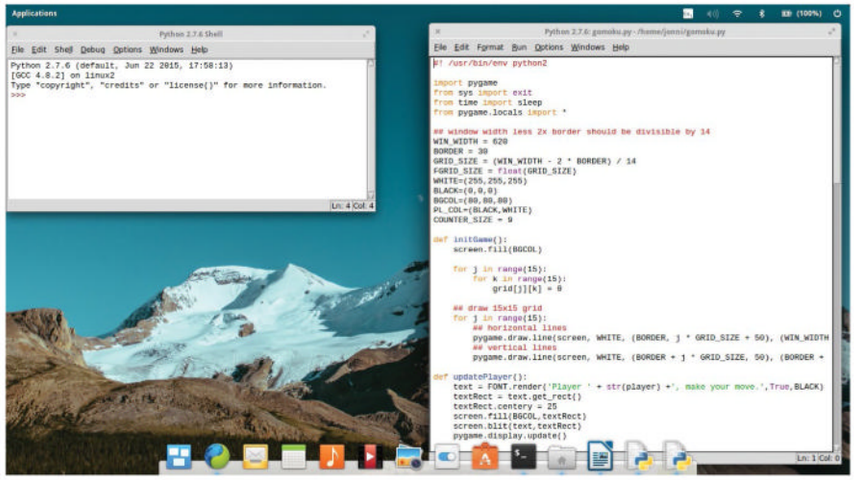
Alternatively, start up *IDLE* and start a command prompt from there by choosing 'Python 2.7 Shell' from the Window menu. Both will start the Python interactive interpreter, wherein Python commands can be typed at the `>>>` prompt and evaluated immediately. It is here that we will forge our first Python program, by carefully typing out the following incantation:

```
>>> print('Hello World')
```

As you might suspect, pressing Enter causes Python to display a global greeting. You can exit the interpreter at any point by pressing Ctrl-D or using the `exit()` command. For larger programs it makes sense to work in a text editor, or an Integrated Development Environment (like IDLE), but the interpreter is a great way to test smaller code fragments. So we'll use it to now to introduce some fundamental coding concepts and constructs. First let's introduce the idea of variables:

```
>>> name = 'Methuselah'
>>> age = 930
>>> print(name, ' is ', age, ' years old.')
```

So we can assign values to variables,



The IDLE development environment is purpose built for Python. You can install it on Ubuntu (or in this case ElementaryOS) with `apt-get install idle`.

change them to our hearts' content, and use `print` statements to see them. Technically, the brackets in the `print` line are only required in Python 3, but they don't do any harm and it's good practice to write, wherever possible, ambidextrous code that will run in both versions. Variables in Python are automatically assigned a type based on their content. So `name` is a string (short for a string of characters) and `age` is an integer (a whole number). You can check this by typing `type(name)` etc.

Some types can be coerced to other types – we can transmute `age` to a floating point number (one with a decimal part) with:

```
>>> age = float(age)
>>> age
```

Just typing the variable name into the interpreter will show you its value, so you can easily see the changes you have enacted. We can convert ints or floats to strings, using the function `str()`. Python can also go the other way, converting a string to a float for example, but this will only work if the original string looks something like its target type: `float('10.0')` will work, but `float('Rumplestiltskin')` will not.

Just as division works differently for floats and ints, so addition works differently for strings. Here the `+` operator stands for concatenation, tacking the latter string onto the end of the former. Thus:

```
>>> 'Hello ' + 'world'
'Hello world'
>>> str(123) + str(456)
'123456'
>>> 'Betelgeuse' * 3
'BetelgeuseBetelgeuseBetelgeuse'
```

The last line shows that we can also multiply strings – division and substraction, however, are not defined for strings.

Data types dictate how data is represented internally, and the effects of this can be quite subtle. For example, in Python 2 the division operator `/` works differently if one of its arguments is a float:

Installing Python and Pygame

If you're using Raspbian on the Raspberry Pi or any flavour of desktop Linux, then the chances are that you already have at least one (probably two) versions of Python installed. While the latest release (1.9.2) is now Python 3 compatible, no distributions are shipping this version yet, so we'll stick to Python 2 (2.7 to be precise) for this tutorial. Check your default Python version by typing:

```
$ python -V
```

If this returns a result that begins with 2, then everything is fine. If, on the other hand, you see 3-point-something, then check Python 2 availability with the command:

```
$ python2 -V
```

Some distros, notably Arch Linux and the recently released Fedora 22, don't ship the 2.7 series by default. Installing *Pygame*, however, will pull it in as a dependency, so let's do that now. Users of Debian-derived distributions (including Raspbian on the Pi) should use the following command to install *Pygame*:

```
$ sudo apt-get install python-pygame
```

Users of other distributions will find a similarly named package (on Arch it's called python2-pygame) and should be able to install it through the appropriate package manager, whether that's *pacman yum*, *zypper* or whatever. Most distributions bundle the IDLE environment with each version of Python installed; if you can't find an icon for it, try running the commands `idle` or `idle2`. If that fails to produce the goods, then go hunting in your distro's repos.

```
>>> 3/2
>>> 1
>>> 3/.
>>> 1.5
```

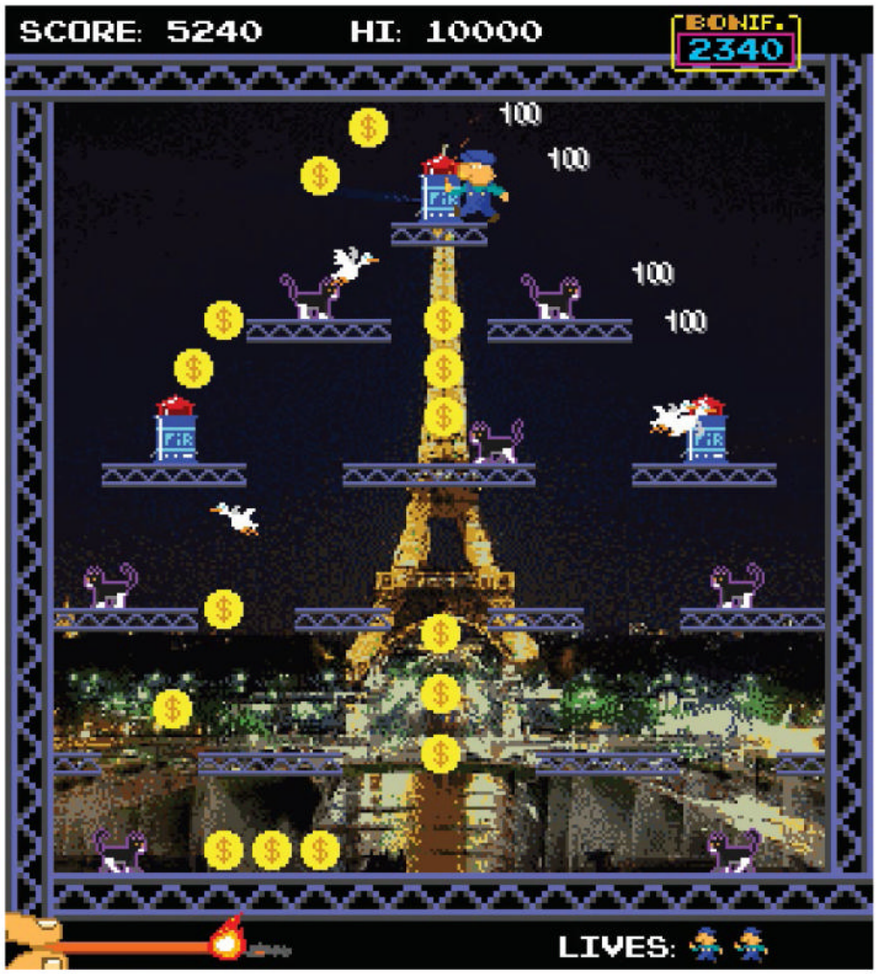
Funny the difference a dot can make. Note that we have been lazy here in typing simply `2.` when we mean `2.0`. Python is all about brevity. (Besides, why make life hard for yourself?) Sooner or later you'll run into rounding errors if you do enough calculations with floats. Check out the following doozy:

```
>>> 0.2 * 3
0.6000000000000001
```

Such quirks arise when fractions have a non-terminating binary decimal expansion. Sometimes these are of no consequence, but it's worth being aware of them. They can be worked around either by coercing floating point variables to ints, or using the `round()` function, which will give you only the number of decimal places you require. We'll see this in practice when we program our Gomoku game later.

Going loopy

Very often programmers desire to do almost the same thing many times over. It could be appending entries to a list, adding up totals for each row in a table, or even subtracting energy from all enemies just smitten by a laser strike. Iterating over each list item, table row or enemy manually would be repetitive and make for lengthy, hard-to-read code. For this reason we have loops, like the humble `for` loop below. When you hit Enter after the first line, the prompt will change to `...` . This is because the interpreter knows that a discrete codeblock is coming and the line(s) following the `for` construct 'belong' to it. Such codeblocks need to be indented, usually using four spaces, though you can use as many as you like so long as you're consistent. If you don't indent the second line, Python will shout at you. Entering a blank line after the `print` statement will end the codeblock and cause our loop to run.



There are all kinds of *Pygame*-powered games. This one, *You Only Get One Match,* features lots of fireworks but limited means of ignition. Check it out at http://bit.ly/LXF202-onematch

```
>>> for count in range(5):
...         print('iteration #', count)
```

There's a few things going on here. We have introduced a new variable, an integer by the name of `count` . Giving sensible names to your variables is a good idea; in this case it implies (even in the absence of any other coding knowledge) that some counting is about to happen. The `range()` function, when used in isolation, returns a list consisting of a range of integers. We'll cover lists in just a moment, but for now we just need to know that `range(5)` looks like [0, 1, 2, 3, 4], which you can verify in the interpreter. So our variable `count` is going to iterate over each of these values, with the `print()` line being issued five times – once for each value in the range.

Another type of loop is the `while` loop.

How to play Gomoku

Gomoku is short for gomokunarabe, which is roughly Japanese for 'five pieces lined up'. The game in fact originated in China some 4,000 years ago. Players take turns to each place a counter on the intersections of a square grid with the goal of forming unbroken lines (horizontal, vertical or diagonal) of length 5. Traditionally the board has 19x19 intersections, but we've gone for the smaller 15x15 board used in some variations. We haven't included an AI (that would be somewhat too complicated for a beginner tutorial) so you'll need to find a friend/other personality with which to play. Alternatively there are plenty of online versions you can play, and KDE users get the similar *Bovu* game with their desktop.

It's easy to figure out some basic strategies, such as always blocking one side of your opponent's 'open three' line or obstructing a 'broken four'. Yet to become a master takes years of practice. The basic rule set as we've implemented it heavily favours the starting player (traditionally black). In fact, work by L. Victor Allis has shown that a good player (actually a perfect player) can force a win if they start. To mitigate against this, big tournaments use a starting strategy called swap2. The first player places two black counters and one white one on the board, and the second player then either chooses a colour or places another black and another white counter on the board and allows player 1 to choose colours. You are free to modify the code to force use of swap2, but it's entirely possible to obey this rule without any code modification: just disobey the first few 'Player x, make your move' prompts.

Rather than iterating over a list, our wiley `while` loop will keep going over its code block until some condition ceases to hold. In the following example, that condition is that the user claims to have been born after 1900 and before 2016.

```
>>> year = 0
>>> while year < 1900 or year >= 2015:
...     year = input("Enter your year of birth: ")
...     year = int(year)
```

Again the loop itself is indented, and again you'll need to input a blank line to set it running. We've used the less than (`<`) and greater than or equal to (`>=`) operators to compare values. Conditions can be combined with the logical operators `and`, `or` and `not`. So long as `year` has an unsuitable value, we keep asking. It is initialised to 0, which is certainly less than 1900, so we are guaranteed to enter the loop. We've used the `input()` function, which returns whatever string the user provides. This we store in the variable `year`, which we convert to an integer so that the comparisons in the `while` line do not fail. It always pays to be as prudent as possible as far as user input is concerned: a malicious user could craft some weird input that causes breakage, which while not a big deal in this example, is bad news if it's done, say, on a web application that talks to a sensitive database. You could change 1900 if you feel anyone older than 115 might use your program. Likewise, change 2015 if you want to keep out (honest) youngsters.

The opposite of listless

We mentioned lists earlier and in the exciting project that follows we'll use them extensively, so it would be remiss not to say what they are. Lists in Python are flexible constructions that store a series of indexed items. There are no restrictions on said items: they can be strings, ints, other lists, or any combination of these. Lists are defined by enclosing a comma-separated list of the desired entries in square brackets. For example:

```
>>> myList = ['Apple', 'Banana', 'Chinese Gooseberry']
```

The only gotcha here is that lists are zero-indexed, so we'd access the first item of our list with `myList[0]` . If you think too much like a human, then 1-indexed lists would make more sense. Python doesn't respect this, not even a little, so if you too think like a meatbag, then be prepared for some classic off-by-one errors. We can modify the last item in the list thusly:

```
>>> myList[2] = 'Cthulhu'
```

Lists can be declared less literally – for example, if we wanted to initialise a list with 100 zeroes, we could do:

```
>>> zeroList = [0 for j in range(100)]
```

This is what is known as a list comprehension. Another example is

```
>>> countList = [j for j in range(100)]
```

which results in a list containing the integers 0 up to and including 99, which could equally well be achieved with `range(100)` in Python 2. However, the concept is more powerful – for example we can get a list of squares using the exponentiation (to the power of) operator `**` :

```
>>> squareList = [j ** 2 for j in range(100)]
```

And after that crash course we're ready to program our own game. You'll find all the code on the disc (in Tutorials, in the file **gomoku.py**) or at **http://pastebin.com/FRe7748B**, so it would be silly to reproduce that here. Instead we'll focus on the interesting parts, in some cases providing an easier-to-digest fragment which you can play with and hopefully see how it evolves into the version in the program.

To dive in and see the game in action, copy **gomoku.py** to your home and run it with:

```
$ python2 gomoku.py
```

On the other hand, if you want to see some code, open up that file in *IDLE* or your favourite text editor. Starting at the first line is a reasonable idea… It looks like:

```
#!/usr/bin/env python2
```

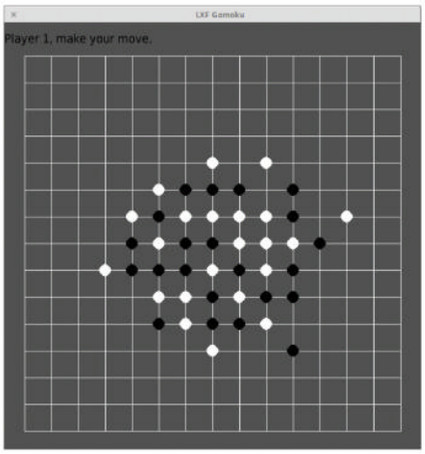
This line is actually ignored by Python (as are all lines that begin with `#`) but it is used by the shell to determine how the file should be executed. In this case we use the *env* utility, which should be present on all platforms, to find and arm the Python 2 executable. For this nifty trick to work, you'll need to make the **gomoku.py** file executable, which is achieved from the command prompt (assuming you've copied the file to your home directory, or anywhere you have write permission) with:

```
$ chmod +x gomoku.py
```

You'll find you can now start the game with a more succinct:

```
$ ./gomoku.py
```

Next we have three `import` statements, two of which (`pygame` and `sys`) are straightforward. The **pygame** module makes easy work of doing game-related things – we're really only scratching the surface with some basic graphics and font rendering. We need a single function, `exit()` , from the `sys` module so that we can cleanly shut down the game when we're done. Rather than importing the whole `sys` module we import only this function. The final import line is just for convenience – we have already imported `pygame`, which gives us access to `pygame.locals`, a bunch of constants and variables. We use only those relating to mouse, keyboard and quitting events. Having this line here means we can access, say, any mouse button events with `MOUSEBUTTONDOWN` without prefixing it with `pygame.locals`.



One of many tense counter-based battles which Jonni ultimately won. [*That's cos you were playing both sides – Ed*]

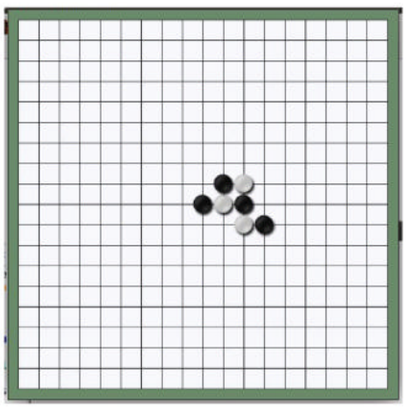
It's all in the (Py)game

Throughout the program you'll notice that some variables are uppercase and some are not. Those in uppercase are either from `pygame.locals` or should be considered constants, things that do not change value over the course of the game. Most of these are declared after the `import` statements and govern things like the size of the game window and counters. If you want to change the counter colours, to red and blue for example, you could replace the values of `WHITE` and `BLACK` with `(255,0,0)` and `(0,0,255)` respectively. These variables are tuples (a similar structure to a list, only it cannot be changed) which dictate the red, green and blue components of colours.

Next you'll see a series of blocks beginning with `def:` – these are function definitions and, as is the case with other codeblocks in Python, they are demarcated by indentation. The `initGame()` function initialises the play area. Here's a simple version that shows what this function does:

```
WIN_WIDTH = 620
GRID_SIZE = (WIN_WIDTH) / 14
WHITE=(255,255,255)
BLACK=(0,0,0)
BGCOL=(80,80,80)
def initGame():
    screen.fill(BGCOL)
    for j in range(15):
        pygame.draw.line(screen, WHITE, (0, j * GRID_SIZE), (WIN_WIDTH, j * GRID_SIZE))
        pygame.draw.line(screen, WHITE, (j * GRID_SIZE, 0), (j * GRID_SIZE, WIN_WIDTH))
pygame.init()
pygame.display.set_caption('LXF Gomoku')
screen = pygame.display.set_mode((WIN_WIDTH,WIN_WIDTH))
initGame()
pygame.display.update()
```

If you add the three import lines to the beginning of this, then this is actually a »



Joel Murielle's graphical *Gomoku* is available from the *Pygame* website. *Pygame* takes all the pain out of working with sprites, notorious troublemakers.

perfectly valid Python program. The `initGame()` function doesn't do anything until it is called at the last line, by which time we've already initialised *Pygame*, set our window title and set the window size to 620 pixels. All variables set up outside of function definitions, hence the five all-caps constants at the beginning and `screen` are accessible inside function definitions; they are known as global variables. Variables defined inside function definitions are called 'local' – they cease to exist when the function exits, even if they have the same name as a global variable – again, something to be aware of in your future coding endeavours. The variable `screen` refers to the 'canvas' on which our game will be drawn, so it will be used extensively later on. The `initGame()` function's first act is to paint this canvas a delightful shade of grey (which you're very welcome to change). Then we use a loop to draw horizontal and then vertical lines, making our 15x15 grid. None of this artwork will appear until we tell *Pygame* to update the display, hence the last line.

Astute readers will notice that the grid overruns ever so slightly at the edges. This is because drawing 15 equispaced parallel lines divides the board into 14, but 620 (our window size) is not divisible by 14. However, when we add in some window borders, since we want to place counters on the edge lines as well, 620 turns out to be a very good number, and we were too lazy to change it. Though rough around the edges, it's still testament to *Pygame's* power and Python's simplicity that we can do all this in just a few lines of code. Still, let's not get ahead of ourselves, our game still doesn't do anything.

Finer points

From here onward, we'll refer to the actual code, so any snippets we quote won't work in isolation – they're just there to highlight things. You'll notice that the `FONT` variable isn't defined with the other constants: this is because we can't use *Pygame's* font support until after the *Pygame's* `init()` method has been called. Let's look at the main game loop right at the end of the code. The introductory clause `while True:` suggests that this loop will go on forever. This is largely correct – we want to keep checking for events, namely mouse clicks or the user clicking the exit button, until the game is done. Obviously we exit the loop when the application quits – clicking the button triggers a `QUIT` event which we react to with the `exit()` function from the `sys` package. Inside the main loop, the first thing we do is call the `updatePlayer()` function, which you'll find on line 32. This updates the text at the top of the screen that says whose go it is, drawing ('blitting') first a solid rectangle so any previous text is erased. Next we loop over the events in the events queue; when the player tries to make a move, the `tryCounterPlace()` function is called, with the mouse co-ordinates passed along.

To keep track of the game, we use a two-dimensional square array (a list of lists) in the variable `grid`. This is initialised as all 0s, and when a player makes a move a 1 or a 2 is entered accordingly. The first job of the `tryCounterPlace()` function is to reconcile the mouse coordinates where the user clicked with a pair of coordinates with which to index the grid variable. Of course, the user may not click exactly on an intersection, so we need to do some cheeky rounding here. If the player clicks outside of the grid (e.g. if they click too far above the grid, so the `y` coordinate will be negative) then the function returns to the main loop. Otherwise we check that the grid position is unoccupied and if so draw a circle there, and update our state array `grid`. A successful move causes our function to return a `True` value, so looking at line 111 in the code we see this causes the next player's turn. But before that is enacted, by the `updatePlayer()` call at the top of the loop, we call the `checkLines()` function to see if the latest move completed a winning line. You'll find details of how this check is carried out in the box.

When a winning counter is detected by our state-of-the-art detection algorithm, the `winner()` function is invoked. This replaces the text at the top of the screen with a message announcing the victor, and the gameover loop is triggered. This waits for a player to push R to restart or rage quit. If a restart is ordered, then the player order is preserved and, since this is updated immediately before `checkLines()` is called, the result is that the loser gets to start the next round.

This is a small project (only about 120 lines, not really a match for the 487 byte *Bootchess* you can read about at **www.bbc.co.uk/news/technology-31028787**), but could be extended in many ways. Graphics could be added, likewise a network play mode and, perhaps most ambitiously, some rudimentary AI could be employed to make a single player mode. This latter has in fact already been done…

Reading between the lines

Part of Key Stage 2 involves learning to understand and program simple algorithms. We've already covered our basic gameflow algorithm – wait for a mouseclick (or for the user to quit), check if that's a valid move, check if there's a line of five, etc. At the heart of that last stage, lies a naïve, but nonetheless relevant, algorithm for detecting whether a move is a winning one.

Consider the simpler case where we're interested only in horizontal lines. Then we would loop over first the rows and then the columns of our grid array. For each element we would check to see that it is non-zero (i.e. there is a counter there) and if the four elements to its right have the same value. In Python it would look like this:

```
for j in range(15):
    for k in range(10):
        pl = grid[j][k]
        if pl > 0:
            idx = k
            while grid[j][idx] == pl and idx < 14:
                idx += 1
            if idx - k >= 5:
                # game winning stuff goes here
```

Note that the inner loop variable `k` reaches a maximum value of only 9. We do not need to check row positions further right than this since our algorithm will reach out to those positions if a potential line exists there. Our variable `idx` effectively measures the length of any line; it is incremented using the `+=` operator, short for `idx = idx + 1`.

The algorithm is easily adapted to cover vertical and diagonal lines. Rather than having four separate functions, though, we've been clever and made a general function `lineCheck()`, which we call four times with the parameters necessary for each type of line checking. Said parameters just change the limits of the `for` loops and how to increment or decrement grid positions for each line direction.

Languages: an overview

One of technology's greatest achievements was IBM's Fortran compiler back in the 1950s. It allowed computers to be programmed using something a bit less awkward than machine code. Fortran is still widely used today and, while some scoff at this dinosaur, it remains highly relevant, particularly for scientific computing. That said, nobody is going to start learning it out of choice, and there are all manner of other languages out there.

Traditionally, you have had the choice between hard and fast languages – such as Java, C and C++ – or easy and slower ones, like Python or PHP. The fast languages tend to be the compiled ones, where code has to be compiled to machine code before it can be run. Dynamic languages are converted to machine code on the fly. However, on some level all programming languages are the same – there are some basic constructs such as loops, conditionals and functions, and what makes a programming language is simply how it dresses these up.

For those just starting coding, it's simply baffling. Opinions are polarised on what is the best language to learn first of all, but the truth is that there isn't one, though for very small people we heartily recommend Scratch. Any language you try will by turns impress and infuriate you. That said, we'd probably not recommend C or Haskell for beginners.

There is a lot of popular opinion that favours Python, which we happily endorse, but then many are put off by the Python 2 versus 3 fragmentation Python has a lot going for it: it's probably one of the most human-readable languages out there. For example, you should, for readability purposes, use indentation in your code, but in Python it's mandatory. By forcing this issue, Python can do away with the curly brackets used by so many other languages for containment purposes. Likewise there's no need to put semicolons at the end of every line. Python has a huge number of extra modules available, too – we've already seen *Pygame* and our favourite, the API for programming *Minecraft* on the Pi.

Beginner-friendly

Other languages suitable for beginners are JavaScript and PHP. The popularity of these comes largely from their use on the web. JavaScript works client-side (all the work is done by the web browser) whereas PHP is server-side. So if you're interested in programming for the web, either of these will serve you well. You'll also want to learn some basic HTML and probably CSS too, so that you can make your program's output look nice, but this is surprisingly easy to pick up as you go along. PHP is cosmetically a little messier than Python, but soon (as in The Matrix) you'll see right through the brackets and dollar signs. It's also worth mentioning Ruby in the accessible languages category. It was born of creator Yukihiro Matsumot's desire to have something as "powerful as Perl, and more object-oriented" than Python.

> **"Although any language will by turns impress and infuriate you, Python has a lot going for it."**

Barely a day goes by without hearing about some sort of buffer overflow or use-after-free issue with some popular piece of software. Just have a look at **https://exploit-db.com**. All of these boil down to coding errors, but some are easier to spot than others. One of the problems with the fast languages is that they are not memory safe. The programmer is required to allocate memory as it is required and free it when it is no longer required. Failure to do this means that programs can be coerced into doing things they should not do. Unfortunately, 40 years of widespread C usage have told us that this is not a task at which humans excel, nor do we seem to be getting any better at it. Informed by our poor record here, a new generation of languages is emerging. We have seen languages contributed from Google (Go), Apple (Swift) and Mozilla (Rust). These languages all aim to be comparable in speed to C, but at the same time guaranteeing the memory safety so needed in this world rife with malicious actors.

Rust recently celebrated its 1.0 release and maybe one day *Firefox* will be written using it, but for now there are a number of quirks and pitfalls that users of traditional languages are likely to find jarring. For one thing, a program that is ultimately fine may simply refuse to compile. Rust's compiler aims for consistency rather than completeness – everything it can compile is largely guaranteed, but it won't compile things where any shadow of doubt exists, even if that shadow is in the computer's imagination. So coders will have to jump through some hoops, but the rewards are there – besides memory safety and type inference, Rust also excels at concurrency (multiple threads and processes), guaranteeing thread safety and freedom from race conditions. »

Programming paradigms and parlance

With imperative programming the order of execution is largely fixed, so that everything happens sequentially. Our Gomoku example is done largely imperative style, but our use of functions makes it more procedural – execution will jump between functions, but there is still a consistent flow. The object oriented (OO) approach extends this even further. OO programs define classes, which can be instantiated many times; each class is a template for an object, which can have its own variables (attributes) and its own code (methods). This makes some things easier, particularly sharing data without resorting to lengthy function calls or messy global variables. It also effects a performance toll, though, and is quite tricky to get your head around. Very few languages are purely OO, although Ruby and Scala are exceptions. C++ and Java support some procedural elements, but these are in the minority.

Functional programming (FP) has its roots in logic, and is not for the faint-hearted. This very abstract style is one in which programs emphasise more what they want to do than how they want to do it. Functional programming is all about being free of side-effects – functions return only new values, there are no global variables. This makes for more consistent languages such as Lisp, Scheme, Haskell and Clojure. For a long time FP was traditonally the preserve of academia, but it's now popular with industry.

Coding in the classrooms

In September 2014, the UK embarked on a trailblazing effort which saw coding instilled in the National Curriculum. When the initiative was announced in 2013, then education secretary Michael Gove acknowledged that the current ICT curriculum was obsolete – "about as much use as teaching children to send a telex or travel in a zeppelin". Far more important was imparting coding wisdom unto the young padewans. Coding skills are much sought-after, as evidenced by industry consistently reporting difficulties in finding suitably qualified applicants for tech jobs n the UK.

The FUZE box gives you everything you need to start fiddling with registers or making GPIO-based mischief.

The 'Year of Code' was launched to much fanfare, though this was slightly quelled as details emerged: a mere pittance was to be added to the existing ICT allocation, and most of this would be spent on training a mere 400 'Master Teachers' who could then pass their Master Skills to lesser teachers around the country. Fans of shadenfreude will enjoy the BBC Newsnight interview with the then Year of Code chief, wherein she openly admits not knowing how to code, despite claiming it is vital for understanding how the world works.

Learning opportunities

Criticism and mockery aside, we're genuinely thrilled that children as young as 5 are, even as we speak, learning the dark arts of syntax, semantics and symbolism. Fear now, ye parents, when your progeny hollers at you (that's what kids do, apparently) "What's the plural of mongoose?" and then follows through seeking clarification on the finer points of recursion and abstraction. Rightly or wrongly, many private firms will benefit from the concerned parents and confused kids resulting from the new computing curriculum. But there are more wholesome directions in which one can seek help.

> **"Without any coding, the ICT curriculum was 'as much use as teaching kids to send a telex'."**

For one thing, you'll always find great tutorials monthly in Linux Format – just look again at the previous five pages, such exceptional erudition. There are also many free resources on the web. Some of them (such as the official Python Documentation) are a little dry for kids, but we'd encourage adults to learn these skills alongside their offspring. Refurbishing an old machine with a clean Linux install will provide a great platform to do just this. All the software you need is free, and distributions like Mint and Ubuntu are easy enough to get to grips with.

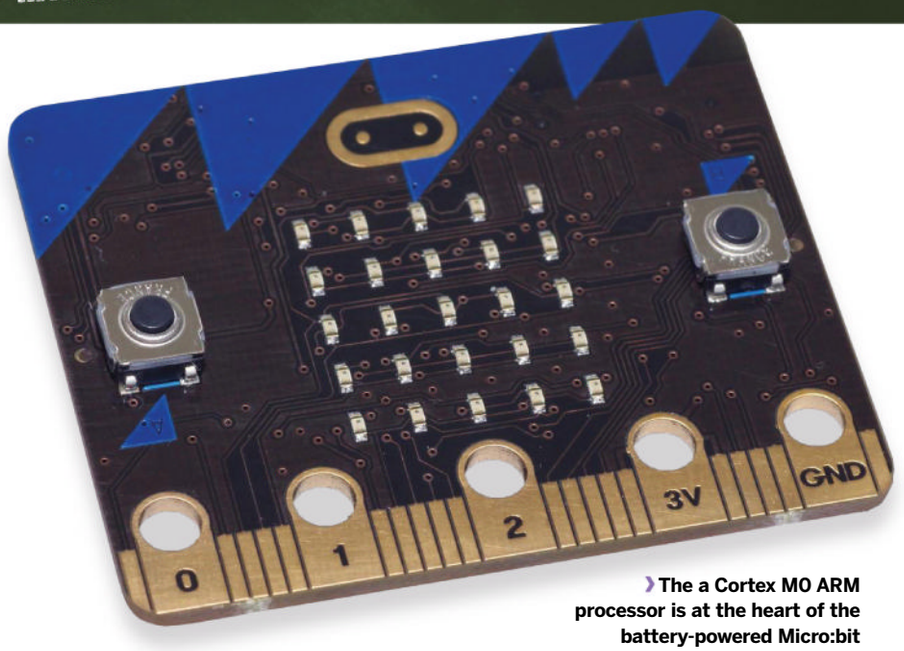
Besides a stray PC, the Raspberry Pi is another great way to provide a learning platform. If you're willing to settle for the older B+ model, then you can get one for about £25, and it'll plug straight into your telly. Of course, you'll need to scavenge a keyboard, mouse, SD card, HDMI cable and possibly a wireless adapter too, if trailing an Ethernet cable to your router is not an option. Mercifully, there are many kits available (for example the Kano, billed as a DIY computer) which will provide these additional peripherals, not to mention the glorious array of Pi cases available to protect it from dust and bumps. We particularly like setups such as the FUZE box, which embed the Pi into a more traditional, albeit chunkier, all-in-one device. Yes, the Pi's tiny form factor is appealing, but with a heavy rotation of USB devices thrown in, it's all too easy to end up in cable-induced tiny hell.

Speaking of tiny things, to coincide with the new learning regimen the BBC will distribute about a million 'Micro: bit' computers to Year 7 pupils. These are even smaller than the Pi, but have no means of output besides a 5x5 LED array. Unlike the Pi, then, they cannot function as standalone computers, requiring instead to be programmed from a more capable device.

Microsoft is generously providing the software and training for this venture, and we are relieved to hear they will not tie the product to their nascent Windows 10 platform, an unfinished beta of which was released at the end of July. The devices have gravity and motion sensors, as well as Bluetooth and some buttons. There are five GPIO rings that could connect further sensors or contraptions. They can be programmed in a number of languages including C++, Python, JavaScript and Blocks – a visual programming language. The Microsoft-provided code editors are all web based, and code entered here must be compiled before being downloaded to the Micro:bit. At present, amidst the pre-launch hush, details are sketchy, but it seems like this compilation all takes place on Microsoft servers. It'll be disappointing if the code editors can't be used offline or no option to compile locally is offered. You can find out more about it at **www.bbc.co.uk/mediacentre/mediapacks/microbit**

Micro:bit spokesbods have been keen to stress that device is in no way intended to compete with the Raspberry Pi, but rather to complement it. The Micro:bit features a 20-pin edge connector which connects it to the Pi or another device so that the machines can work in tandem. Such a pairing will be necessary for the Micro:bit to have

means of communicating with the outside world. The device – indeed the combination of the device and the Pi – has a great deal of potential but there exists some skepticism over whether anything like the excitement generated by the 1982 launch of BBC Micro will be seen again. Back then computers were new and exciting, while these days kids expect them to provide a constant barrage of entertainment in the form of six-second cat videos or 140-character social commentary. Not all of them are going to be thrilled at having to program them. But anything that lowers, if not entirely obliterates, any entry barrier to getting into coding is fine by us. We also look forward to ne'er-do-well students hacking each other's Micro:bits or engaging them in a collaborative DDOS attack on their school's infrastructure.



The a Cortex M0 ARM processor is at the heart of the battery-powered Micro:bit

Get with the program

The syllabus comprises three Key Stages. The first, for 5-6 year olds, covers algorithms in a very general sense. Algorithms will be described in terms of recipes and schedules, to introduce the idea of formalising instructions. The second stage (ages 7-11) introduces core ideas such as loops and variables. Alongside this, candidates will be learning to use web services and how to gather data. The final stage, for secondary students aged 11-14, requires students to learn at least two programming languages and understand binary arithmetic, Boolean algebra, functions and datatypes. Students will also touch on information theory, at least as far as how different types of information can all be represented as a string of bits. They will also gain insights into the relationship between hardware and software.

Throughout the curriculum, students will also learn the vital skills of online privacy and information security – skills the want of which has led to many an embarrassing corporate or governmental blunder. All in all, it's a highly ambitious project, but perhaps such a radical step is necessary to address the skills shortage in this area. With luck the scheme will also lead to much-needed diversification among the coding populace. If it works out, and pupils are learning Python alongside Mandarin or studying Kohonen or Knuth alongside Kant, then we'll be thrilled. »

Code Clubs

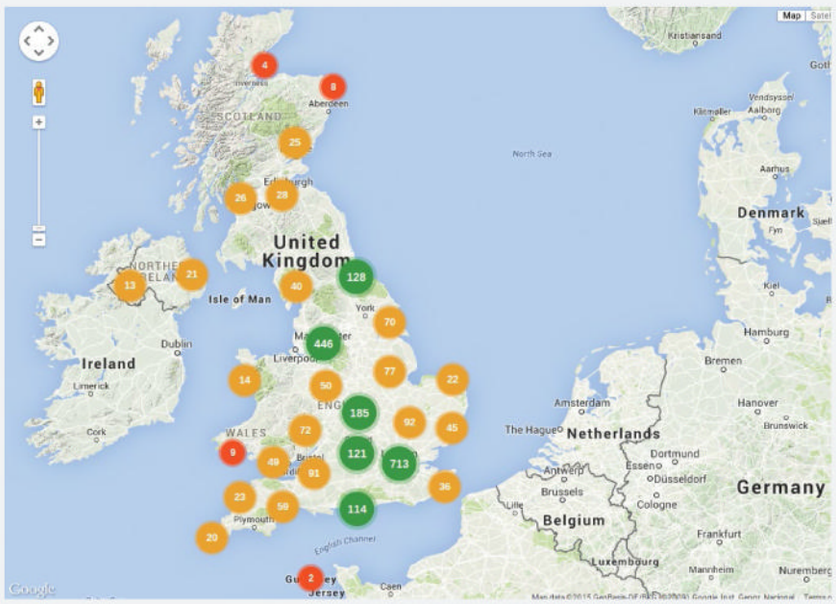
</code>

There are also over 2,000 volunteer-run code clubs across the UK. Code Club, armed with £100,000 courtesy of Google, provides the material, and schools (or other kind venues) provide space and resources for this noble extracurricular activity.

The Code Club syllabus is aimed at 9-11 year olds and consists of two terms of Scratch programming, then a term of web design (HTML and CSS), concluding with a final term of grown-up Python coding. Projects are carefully designed to keep kids interested: they'll be catching ghosts and racing boats in Scratch, and doing the funky Turtle and keeping tabs on the latest Pokémon creatures in Python, to name but a few.

If you have the time and some knowledge, it's well worth volunteering as an instructor. All the tutorials are prepared for you and if you can persuade a local primary school to host you, then you'll get a teacher to maintain order and provide defence against any potential pranksters. You will require a background check, though. Code Club also provide three specialist modules for teachers whose duty it is to teach the new Computing curriculum.

This kind of community thinking is very much in the spirit of open source, providing an access-for-all gateway to coding free from commercial interest and corporate control. The Code Club phenomenon is spreading worldwide too. You'll now find them as far afield as Bahrain and Iceland. The project materials have already been translated into 14 languages with more on the way. You can find out more at **www.codeclub.org.uk**



PHP: Code and calculate

PHP used to stand for Personal Home Page, but these days it's a recursive acronym for PHP: Hypertext Processor. Many of the world's leading web applications, such as WordPress and OwnCloud, are written in PHP. It's used extensively by some of the biggest websites in the world, some of which are so big (Facebook) that they had to write their own engine (HHVM) for it. Rather than using PHP in a traditional setting (running as a module through a webserver), let's install PHP-CLI, the command line edition. This saves us the trouble of setting up Apache or some such (although that's pretty easy nowadays, so you may want to do the tutorial this way and that is fine).

On Debian-based distributions and their derivatives, you'll want to run:

```
$ sudo apt-get install php5-cli
```

Other distros will name their packages differently – on Arch it's called just `php`. Ultimately, if you find your package manager starts trying to install a webserver and all sorts of other gubbins, then you chose the wrong package.

We'll work in the text editor of your choice – you could use `nano` and do the whole thing from the terminal, or you can use something graphical like *IDLE* or *Geany*. Whatever you choose, your first program is going to be short – this short:

```
<?php
echo 'Hello World!';
?>
```

Save this as `hello.php` and then run it with the following command:

```
$ php hello.php
```

You probably knew what was going to happen. Similar to how JavaScript is enclosed in HTML pages via `<script>` tags, PHP code has to be surrounded by the tags on the first and last lines of the snippet there. Codeblocks – which include but are not limited to variable assignments, loops and **if...else..endif** clauses – must be terminated with a semicolon (that most roguish of punctuation marks). If you were to embed the code above in an HTML document on a PHP-enabled webserver, then it would perform its worldly greeting on that webpage. Viewing the page's source through the browser would not show the PHP code, only its output... Spooky.

Let's try a more complicated example. This time we're going to define a function called `fib` which takes an integer `$n` as an argument and returns the `$n`th Fibonacci number. Variables in PHP are prefixed with dollar signs, functions are wrapped in curly brackets and `if` conditions in regular brackets:

```
<?php
function fib($n) {
    if ($n == 0) {
        return 0;
    }
    if ($n == 1) {
        return 1;
    }
    return fib($n - 1) + fib ($n - 2);
}

echo fib(12);
?>
```

Even without any coding experience or exposure to the Fibonacci sequence you'll probably figure out that the 'zeroth' Fibonacci number is 0, and the next one is 1. Subsequent Fibonacci numbers are defined as the sum of their two predecessors, hence the rest of the sequence goes 1,1,2,3,5,8. Amazingly, coding this requires next to no effort – it is perfectly legal to call the function from within itself, and it won't even break. This technique is known as recursion and is vaguely illustrated by the cornflakes packet on which you'll find a smaller picture of the cornflakes packet, which has on it... Maybe not quite like that – thanks to our two base cases the recursion depth is limited, so we won't end up in some infinite depth trap. Be that as it may, this is not the most efficient way to compute Fibonacci numbers. Look at the figure to see the call structure for `fib(5)` – more than half the calls are superfluous. As `n` grows, so too does this duplication of effort: see what happens when you change the function call in the second last line to calculate the 40th Fibonacci number (it's about 100 million, and takes us about 45s to compute). Check out the box opposite for a fix.

> **"Many of the world's leading web apps, like WordPress and OwnCloud, are written in PHP."**

Note that if the user doesn't supply an argument to `fib2.php` (from said box), then PHP will complain since the array `$argv` does not have an entry at index 1. The value of `$argv[0]` is always the name of the program itself, in this case `fib2.php`. Where an argument is provided, we convert it from a string to an int. This converts unusable input to the int 0, which triggers our unhelpful error message. Unfortunately this means our program doesn't work for the zeroth Fibonacci number, but that one's easy to remember. Anything bigger than zero can be found remarkably quickly using our new algorithm. Well, not quite anything – the 93rd Fibonacci number is greater than 2^64, so cannot be stored as a 64-bit integer. PHP rounds this to a floating point number (displayed using exponent-mantissa notation) with limited precision, but eventually (1,477 on our setup) the numbers become too big even for that.

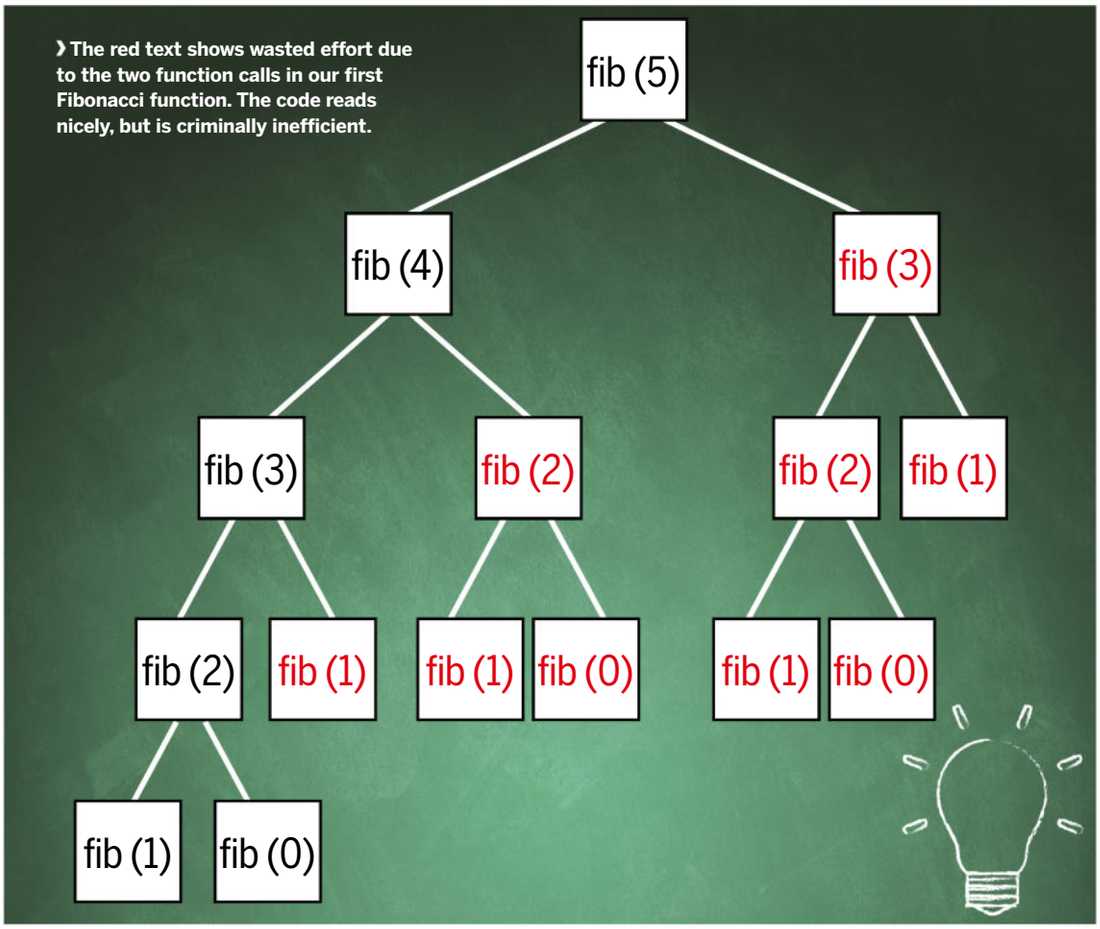
Perhaps it's worth noting that there's an inherent overhead in every layer of recursion and function call you add to your program. So if one really wanted to be Teutonically efficient about it, an iterative solution would be the best one. There's no perceivable speed gains in this case (`fib2 1476` displays an answer almost immediately), so we leave this as an exercise. You'll need to know about **for** loops in PHP, which is good because they feature in the next section.

Caesar cipher

To finish off today's lesson, we're going to use some classic cryptography. Be sure to revisit our glorious treatise on this subject from **LXF189**, but it will suffice to know that Julius Caesar used to encrypt messages by shifting each letter three places down the alphabet, wrapping X, Y and Z to A, B and C respectively. This is slightly tricky to achieve on a computer. One way is to look at the ASCII values of each character of plaintext: uppercase letters A-Z have values 65-90, and lowercase a-z occupy the range 97-122. For simplicity we're going to leave all other characters (e.g. spaces and punctuation) untouched in the ciphertext. So we will loop over `$plaintext` character by character, add the value shift to the ASCII value of each alphabetic character, perform any appropriate wraparounds, get the character represented by this new value and finally append this to our ciphertext.

Once you get that finding the ASCII value is done with the function `$ord` and the character with `$chr` :

```
<?php
function ccipher($plaintext, $shift) {
        $ciphertext = '';
        for ($j = 0; $j < strlen($plaintext);
$j++) {
            $asciicode = ord($plaintext[$j]);
```



› The red text shows wasted effort due to the two function calls in our first Fibonacci function. The code reads nicely, but is criminally inefficient.

Quick tip

Change the value of `$shift` to make other equally useless ciphers – a value of 13 will give you the ROT13 code popular for hiding punchlines and spoilers on many forums.

```
                if ($asciicode >= 65 &&
$asciicode <= 90) {
                    $ciphertext .=
chr(($asciicode + $shift + 13) % 26 + 65);
                }
                else if ($asciicode >= 97
&& $asciicode <= 122) {
                    $ciphertext .=
chr(($asciicode + $shift + 7) % 26 + 97);
                }
                else {
                    $ciphertext .=
chr($asciicode);
                }
            }
            return $ciphertext;
}
$plaintext = 'Veni vidi vinci';
$shift = 3;
echo ccipher($plaintext, $shift);
?>
```

For loops are initiated C-style, giving a list of conditions in brackets. Our iterator variable `$j` starts at 0 and goes all the way to the length of our string. The `and` operator is abbreviated to `&&` and we use the `.=` operator to concatenate strings.

All pretty straightforward, except for the wraparound part. Here we use the modulus operator `%`, which gives you the remainder on division, in our case by 26. In the first case we want this to be 0 if our shifted value is 65, so we add on 13, since 65 + 13 = 78, which is divisible by 26. Dually we add on 7 in the second block. There isn't any room left to write the corresponding `cdecipher()` function, but it's just a question of changing two plus signs to minus signs. See if you can recover the original plaintext this way.

Faster Fibonacci function

Programmers are very keen on writing efficient code. This is not just because they like elegance (though as a rule they do); as we've mentioned, inefficient code might be fine to begin with but can start running as slow as treacle when asked to deal with heavier loads. Here we can use a smarter Fibonacci function which passes the two previous numbers in the sequence, so that no effort is wasted. But we still want to be able to call our function with a single argument, so we'll make these extra arguments optional, setting their defaults to 1 and 0:

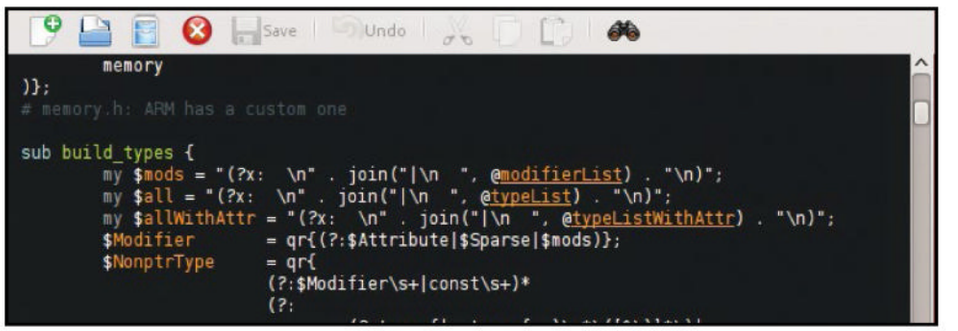
```
<?php
function fib2($n, $oldfib = 1, $olderfib = 0) {
    if ($n > 1) {
        return fib2($n - 1, $oldfib + $olderfib,
$oldfib);
    }
    else {
        return $oldfib;
    }
}
if (!$argv[1]) {
    echo 'no';
}
else{
    echo fib2((int) $argv[1]);
}
?>
```

Besides defining our new improved function `fib2()` we also use the special variable `$argv` to allow the user to provide input on the command line. If one saved this file as `fib2.php`, then the 54th Fibonacci number could be found thusly:

```
$ php fib2.php 54
86267571272
```

Scripting languages

Let's go beyond Bash to see which scripting languages measure up to the needs and wants of a Linux system administrator.



How we tested...

Comparisons, they say, are invidious. This is certainly true for programming languages, where personality and local support are, at least, of equal import to criteria such as speed, and the level of support for different paradigms. Given this, we're presenting a mixture of facts, collective opinions and our own prejudices, but it's a basis for further investigation. The key to a scripting language's usefulness to the sysadmin lies not just in how easily it helps solve problems, but in how many of the solutions have already been written, and are available to download and adapt, and preferably well-documented.

We tried to work across the range of versions installed on a typical network, but insisted on Python 3. Other than that, we've tried to stay in the context of working with what you're likely to find on your network.

Every admin loves time-saving shortcuts, and carries a selection of scripts from job to job, as well as inheriting new ones when arriving in post. The question any new admin asks is which is the best language to learn? (Followed by, where's the coffee?) Veterans of language wars should know that the best language question rarely has a simple or definitive answer, but we thought it would be well worth comparing the most useful choices to make your Linux life easier.

Most scripting languages have been around longer than you think. For example, NewLISP was started on a Sun-4 workstation in 1991. They've borrowed from each other, and elsewhere, and accumulated a long legacy of obsolete libraries and workarounds. Perl's Regular Expressions, for instance, are now found everywhere, and in some cases better implemented elsewhere. So what matters most? How fast the script runs, or how quickly you can write it? In most cases, the latter. Once up and running, support is needed both from libraries or modules to extend the language into all areas of your work, and from a large enough community to support the language, help it keep up with trends, and even to innovate it. So, which scripting language should you learn to improve your Linux life this year?

> **"The question any new admin asks is which is the best language to learn?"**

The learning curve

Online resources, books and good people.

The key questions are: how easy is the language to pick up? Are the learning resources at least adequate? Even if these two questions are answered in the positive, they still need to be backed up by a helpful community to assist you in quickly producing something useful, and help maintain that initial enthusiasm as you hit inevitable problems.

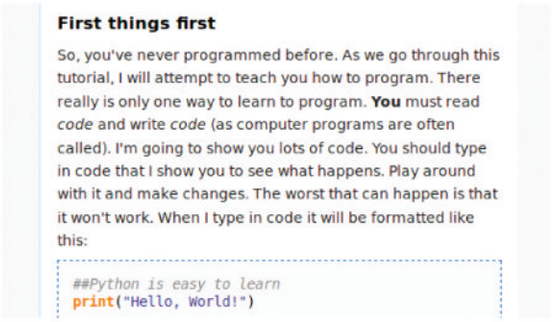
To produce a backup script and test scripts in each of the languages, we started by browsing Stack Overflow. But downloading random code means no consistency between Posix (pure Bourne Shell) scripts, modern Bash, and legacy code that occasionally fails. Fortunately, **www.shellcheck.net** is a great tool for checking the correctness of scripts, and teaches you best practice as it corrects them. The Linux Document Project's (perhaps overly) comprehensive Advanced Bash Scripting Guide (**www.tldp.org/LDP/abs/html**) is also excellent and will help you quickly gain confidence.

Perl's online and built-in documentation is legendary, but we started by running through an exercise from the classic O'Reilly admin book, *Running Linux*, then leapfrogged the decades to No Starch's recent *Perl One-Liners* by Peteris Krumins. Those who eschew the book form should try **http://perlmonks.org**, a source of cumulative community wisdom.

Recent efforts at getting youngsters learning through Code Club (**www.codingclub.co.uk**) and the rest of us through PyConUK education sprints and open data hackdays have shown Python to be easily picked up by anyone. But out-of-date advice, such as the many ways of running subprocesses which persist for compatibility reasons, means careful reading is needed, and it's yet another good reason for starting with Python 3, not Python 2. Head to **www.python.org/about/gettingstarted** for large list of free guides and resources.

Ruby is also an easy sell to learners, and before Rails, command-line apps were what it did best. David B. Copeland's book, *Build Awesome Command Line Applications in Ruby* will save you hours of wading through online documentation, but we were able to get up and running on our test scripts with a couple of web tutorials.

Last, we come to NewLISP: a challenge to programmers schooled only in non-LISP family languages, but you'll be amazed by what it manages to accomplish with just lists, functions and symbols. We dived right in with the code snippets page on **http://newlisp.org**, adapting to build our backup script, and were rewarded with terse, powerful code, that was easier to read than its equally compact Perl counterpart.



From MOOCs to the bookshop, Python learning resources are everywhere.

Verdict

Bash ★★★★☆
NewLISP ★★★★☆
Perl 5 ★★★★☆
Python ★★★★★
Ruby ★★★★★

Python and Ruby are easier to learn, because of good docs and helpful users.

Version and compatibility

Beating the wrong version blues.

The question here is: have I got the right version? Lets start with *Bash*. Every modern Linux distro ships with a version that will run your scripts and anyone else's. *Bash 4*, with its associative arrays, coproc (two parallel processes communicating), and recursive matching through globbing (using ** to expand filenames) appeared six years ago. *Bash 4.2* added little, and is four years old and *Bash 4.3*'s changes were slight.

```
#------------------------
print "\n${l_hlt}" . " "x${l_columns} . "${l_rst}\r"
    . "${l_hlt}  ${l_int}BRANCH:${l_rst}${l_hlt} ${l_git_branch}  (${l_git_sha
}  ${l_rst}\n\$ ";
EOF

}

export COLUMNS
PS1="${PS1}\$(Func_GitCheck)"
Func_GitCheck
richard@luggable:~/work$ bash --version
bash --version
GNU bash, version 4.3.11(1)-release (i686-pc-linux-gnu)
Copyright (C) 2013 Free Software Foundation, Inc.
License GPLv3+: GNU GPL version 3 or later <http://gnu.org/licenses/gpl.html>

This is free software; you are free to change and redistribute it.
There is NO WARRANTY, to the extent permitted by law.
Func_GitCheck
richard@luggable:~/work$
```

As the Unix shell dates back decades, you will find that recent *Bash* versions contain a few unexpected syntax changes.

Perl is still included in the core of most distros. The latest version is 5.20 (with 5.22 soon to appear), but many stable distros ship with 5.18. No matter, you're only missing out on tiny improvements, and just about every script you'd want to write will be fine.

The switch from Python 2 to 3 still catches out the unwary. Run Python 3 if you can and check the documentation if you come unstuck. Python 3.3 is our baseline for Python 3 installs and Python 3.4 didn't add any new syntax features.

Ruby version changes have caused enough problems that painless solutions have appeared, *rvm* enables you to run multiple versions of Ruby, and *bundle* keeps track of the gems you need for each script.

NewLISP's stability and lack of third-party scripts is an advantage here. We cann't, however, guarantee every script will run on the latest versions.

Verdict

Bash ★★★★★
newLISP ★★★★☆
Perl 5 ★★★★☆
Python ★★★☆☆
Ruby ★★★★☆

Ruby's version workaround is good, but Bash's lack of problems is a better result.

Web native scripts

Get your admin scripts moving over HTTP.

Much of a sysadmin's life has migrated to the web, so you'll need a scripting language that has kept pace. We examined both ease of writing our own code, and finding available solutions for doing anything from web interfaces to system stats.

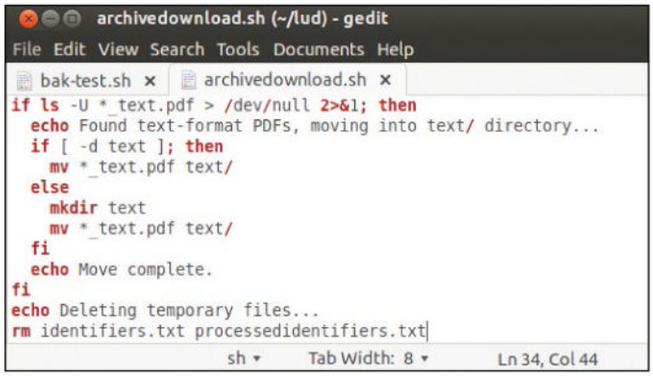
What's noticeable about these languages is the difference in expressiveness and style to produce similar results. However, this is, once again, secondary to personal preference and local support for many admins.

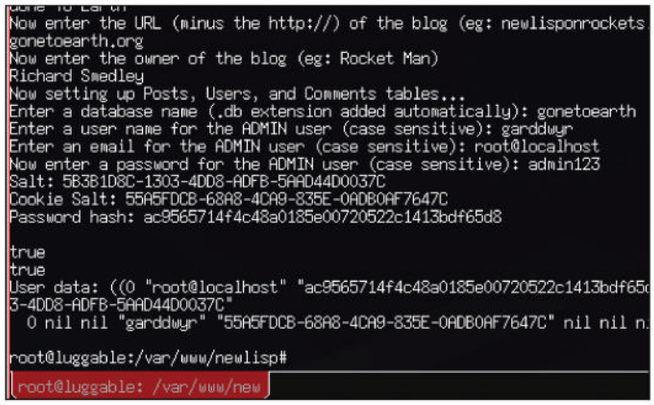
Ruby is quick and enjoyable; Python 'feels right' probably due to it being more human readable; newLISP is astonishingly powerful. But these observations remain partisan clichés without a supportive and maintainable environment to use and develop the code for your own networks.

Bash ★★☆☆☆

While *Bash* will be no one's first choice for a web programming language, it's good to know that when your server doesn't provide for your first choice you can fall back on it thanks to bashlib. This a shell script that makes CGI programming in the *Bash* shell somewhat more tolerable.

Your script will be full of echo statements, interspersed with your commands to produce the desired output. Security considerations mean we wouldn't recommend running this on the open Internet, but it's worth bearing in mind that *Bash* works well as a prototyping language. It's easy to fill a text file with comments describing the broad structure that you want, then fill in the gaps – testing snippets interactively and pasting into **www.shellcheck.net** to check your code as you go. You'll soon be up and running with a proof of concept.





newLISP ★★★★☆

Code Patterns, by NewLISP creator Lutz Mueller, is available on the **www.newlisp.org** website and has chapters on HTTPD and CGI, as well as TCP/IP and UDP communications. If you add in the section on controlling applications, and you'll have everything to get you started.

NewLISP's built-in networking, and simple (or lack of) syntax, makes it surprisingly easy to generate HTML pages of results from, for instance, your monitoring scripts. For a ready built framework, newLISP on Rockets – which uses Bootstrap, jQuery and SQLite – combines rapid application development with good performance.

NewLISP on Rockets provides several functions, from (convert-json-to-list) via (twitter-search) to (display-post-box), which will help you add web functionality. We're impressed but we remain concerned by the small size of the community and the intermittent pace of development.

Community support

Does it have a community large enough to support real work.

DevOps, cloud deployment, test-driven development and continuous integration – the demands on a sysadmin change and evolve, but the requirement to learn something new is constant. Everyone uses *Bash* to some extent but, you'll need to learn *Bash plus* one other.

Perl was the traditional Swiss Army chainsaw of Unix admins through the '80s and '90s, gradually losing ground to Python and then Ruby over the last decade or so. Anyone who started work in the '90s or earlier will be comfortable with it, so finding someone to help with your scripts is often not a problem.

However, the world doesn't stand still, and many tech businesses have standardised on Python, which is used extensively at Google, for example. Much of the software necessary for modern sysadmin work is Python based although the same can be said of Ruby.

Ruby benefits from being the basis of Chef and Puppet, as well as Vagrant and Travis CI, meaning a little familiarity will be helpful anywhere that uses them for deployment. The web frameworks and testing tools written in Ruby have popularised the language at many of the younger web companies.

NewLISP has a much smaller community supporting it, and there aren't many ready made solutions and you may know no-one who uses it. The keenness of the online community goes some way to ameliorate this deficiency, but you have to ask who will maintain your tools when you leave a company?

Verdict

Bash ★★★☆☆
NewLISP ★★☆☆☆
Perl 5 ★★★★☆
Python ★★★★☆
Ruby ★★★★★

» *Ruby has taken mindshare, thanks to some great DevOps software.*

Perl 5 ★★★★☆

Perl was the first web CGI scripting language and has more or less kept pace with the times. It certainly has the libraries, and enough examples to learn from, but with no dominant solution you'll have to pick carefully.

Catalyst, Dancer, and Mojolicious are all good web application frameworks. More likely you'll find everything you need in CPAN. You can glue together a few of the libraries – many of which are already collected together in distros – to handle a pipeline of tasks, such as retrieving XML data, converting the data to PDF files and indexing it on a web page.

Perl's traditional CGI interface is still available, and despite better performing alternatives abstracted through PSGI, you may find that **use CGI;** is all you need to web-enable your script, and remember: 'there's always more than one way to do it'.

```
if ($q->param()) {
    # Parameters are defined, therefore the form has been submitted
    display_results($q);
} else {
    # We're here for the first time, display the form
    output_form($q);
}

# Output footer and end html
output_end($q);

exit 0;

#-----------------------------------------------------------

sub output_top {
    my ($q) = @_;
    print $q->start_html(
        -title => 'Back-up selection',
```

```
                    filename, _, _ = related_attachment
                    content = re.sub(r'(?<!cid:)%s' % re.escape(file
%s' % filename, content)
                self.alternatives[i] = (content, mimetype)

        return super(EmailMultiRelated, self)._create_alternatives(msg)

    def _create_related_attachments(self, msg):
        encoding = self.encoding v settings.DEFAULT_CHARSET
        if self.related_attachments:
            body_msg = msg
            msg = SafeMIMEMultipart(_subtype=self.related_subtype, encod
            if self.body:
                msg.attach(body_msg)
            for related_attachment ∈ self.related_attachments:
                if isinstance(related_attachment, MIMEBase):
                    msg.attach(related_attachment)
                else:
* backup-reporter.py:61:0 Python
```

Python ★★★★★

Python's Web Server Gateway Interface (WSGI), which was defined in PEP 333, abstracts away the web server interface, while WSGI libraries deal with session management, authentication and almost any other problem you'd wish to be tackled by middleware. Python also has plenty of full-stack web frameworks, such as Django, TurboGears and Pylons. Like Rails, for some purposes you may be better off coding web functionality onto an existing script. But Python's template engines will save you from generating a mess of mixed HTML and Python.

Python has many other advantages, from the Google App Engine cloud with its own Python interpreter, which works with any WSGI-compatible web application framework, for testing of scalable applications to supporting a clean style of metaprogramming.

Ruby ★★★★★

Don't imagine for one moment that Rails is a panacea for most sysadmin problems. It's not. And while Sinatra certainly makes it easy to roll out anything web-based in Ruby, even this is overkill for most purposes. That said, Rails does a good job of getting code up quickly and just doesn't drown in all that magic, generated code.

Ruby is ideal for getting any script web-enabled, thanks to gems that are written by thoughtful people who have made sane decisions. Putting a web interface on our backup script, for example, was fun, but distracting as we played with several gems, eg to export reports to Google spreadsheets. Tools like *nanoc*, which generate static HTML from HAML, and some of the reporting gems complement the language's expressiveness, and make adding any functionality to scripts a breeze.

```
log:
development.log

public:
404.html  422.html  500.html  favicon.ico  robots.txt

test:
controllers  fixtures  helpers  integration  mailers  models  t

tmp:
cache  pids  sessions  sockets

vendor:
assets
richard@luggable:~/work/code/ruby/rails/crash/code/blog$
 richard@luggable: ~/work/co
```

Programmability

Managing big scripts requires other programming paradigms.

Before reaching 1,000 lines of code, *Bash* scripts become unmanageable. Despite its procedural nature, there are attempts to make an object-orientated (OO) *Bash*. We don't recommend it, we think it's better to modularise. Functional programming (FP) in *Bash* (**http://bit.ly/BashFunsh**) is also impractical.

Perl's bolted on OO won't be to everyone's taste, but does the job. Perl has fully functional closures, and despite syntactical issues, can be persuaded into FP – just don't expect it to be pretty. For that you should wait for Perl 6.

Python is equally happy with imperative, OO and also manages FP. Functions are first class objects but other features are lacking, even if its list comprehension is very good. Mochi, the FP language (**http://bit.ly/FPMochi**), uses an interpreter written in Python 3.

Ruby is designed as a pure OO language, and is perhaps the best since Smalltalk. It can also be persuaded to support a functional style of programming. But to get FP code out of Ruby, you'll have to go so far from best practices that you should be using another language entirely.

This brings us neatly to NewLISP, an elegant and powerful language with all the functional features at your fingertips. NewLISP uses a pseudo OO implementation in the form of functional-object oriented programming (FOOP), but this doesn't mean, however, that it can cut it for real OO programming.

Verdict

Bash ★★☆☆☆
NewLISP ★★★☆☆
Perl 5 ★★★☆☆
Python ★★★★★
Ruby ★★★★☆

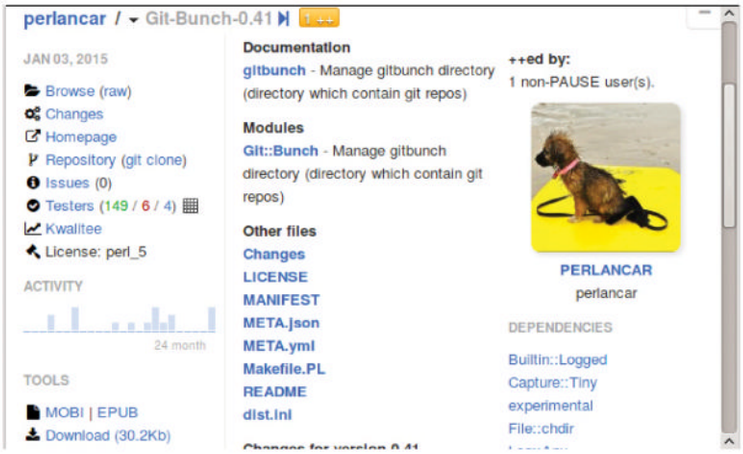
» *Python is a multi-paradigm language and the easiest to maintain.*

Extending the language

Libraries, modules, and getting them working.

None of these scripting languages are as bloated with classes as, for example, Java so that you'll need to use non-core libraries (or modules as they are sometimes called) for writing many scripts. How comprehensive these are, and how easy they are to manage with your script varies greatly.

Perl continues to impress with the mind-boggling choice to be found on CPAN, but its 'there's more than one way to do it' approach can leave you easily overwhelmed. Less obvious, is the magnitude of *Bash* extensions created to solve problems that are perhaps not best suited to any sh implementation.



There's more than one library for that – CPAN is a useful resource for Perl.

Python has excellent library support, with rival choices considered very carefully by the community before being included in the core language. The concern to "do the right thing" is evident in every decision, yet alternate solutions remain within easy reach. At least the full adoption of the *pip* package manager, with Python 3.4, has ensured parity with Ruby.

RubyGems provide the gem distribution format for Ruby libraries and programs, and *Bundler* which manages all of the gems for dependencies and correct versions. Your only problem will be finding the best guide through Ruby's proliferation of libraries. Read around carefully.

NewLisp is not a large language, but it's an expressive one, accomplishing much without the need of add-ons. What modules and libraries that there are address key needs, such as database and web connectivity. There's enough to make NewLISP a useful language for the admin, but not in comparison to the other four choices.

Verdict

Bash ★★★☆☆
NewLISP ★★★☆☆
Perl 5 ★★★★★
Python ★★★★☆
Ruby ★★★★☆

» *The CPAN's longevity and popularity marries well with good organisation.*

Network security

Testing and securing the network – or fixing it afterwards.

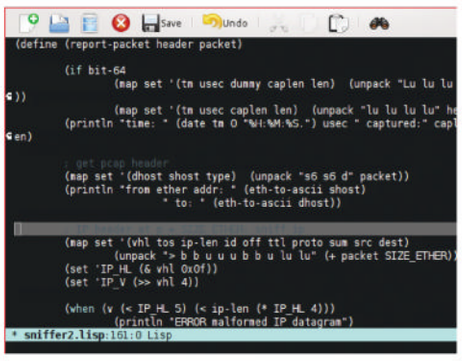
Penetration testing and even forensic examination after an attack will fall under the remit of the hard-pressed sysadmin in smaller organisations. There are enough ready made tools available that you can roll everything you may need into a neat shell script, kept handy for different situations, but writing packet sniffers or tools for a forensic examination of your filesystem in *Bash* isn't a serious option.

Perl has lost some security community mindshare since the early days of Metasploit, but the tools are still there, and are actively maintained by a large user group who aren't about to jump ship to another language. Perl has tools like pWeb – a collection of tools for web application security and vulnerability testing – which is included in distros, such as Kali and Backbox.

Tools such as *Wireshark* are a powerful aide to inspecting packets, but sometimes you'll need to throw together your own packet sniffer. Python not only has Scapy, the packet manipulation library, but provides a socket library for you to easily read and write packets directly.

Ruby's blocks (write functions on-the-fly without naming them) and other features are great for writing asynchronous network code, and its rapid prototyping matches (and even beats) Python. But Ruby's biggest boon is Metasploit, which is the most-used pen-testing software.

In terms of ready rolled tools, you can mix and match as needed, but Perl, Python and Ruby all provide everything you need to quickly examine a network for weaknesses or compromises on-the-fly. Note: Python is featured in more security-related job adverts now.



NewLISP has impressive networking features, even if it lacks the pen-testing tools of the others.

Last, NewLISP isn't well-known among penetration testers and grey hat hackers, but thanks to the networking built in to the language, a function call and a few arguments will create raw packets for pen testing. Once more, NewLISP has clear potential but suffers from its relatively tiny user base.

Verdict

Bash ★☆☆☆☆
NewLISP ★★★☆☆
Perl 5 ★★★★☆
Python ★★★★★
Ruby ★★★★★

» *Python edges ahead of Ruby and Perl, but all three are friends of the pen tester.*

Scripting Languages

The verdict

We admit it's difficult to take the verdict out of a practical context and just declare the best language. For example, *Bash* isn't a strong language, and many time-saving bits of code can be thrown together better with the other four languages, but no-one with tasks to perform at the Linux command line should avoid learning some *Bash* scripting.

Perl is the traditional next step as it's intimately associated with the *nix command line and still found everywhere. It may suffer in comparison with newer languages, but Perl continues to offer not just the Swiss Army Chainsaw of the Linux CLI, but Perl also has a huge and very supportive community.

NewLISP is a pleasant surprise. Yes it has those – Lisp is about lists – but what a compact language for the embedded space as well as the command line. Sadly, the size of the community doesn't match the power of the language, so you'll need to be prepared to back its use with plans to maintain the code yourself.

Python is a powerful, multi-paradigm supporting language. The Python community is large and friendly, and supports everything from education sprints to training teachers. It also backs up community efforts, supporting young learners at Code Clubs, and many other events.

But useful as a community is to the sysadmin, it's often the quick and dirty hacks, readily downloadable and reusable examples, backed by an expressiveness that makes many programming challenges if not trivial, far less of a headache. Rails brought wider attention to Ruby, but Chef, Puppet and Vagrant have helped remind the admin just what can be done with the expressive and eloquent scripting language that was developed by Yukihiro Matsumoto.

```
    private

    def vm_host(vm)
      host_options = {
          user: vm['user'] v 'vagrant',
          hostname: vm['hostname'] v 'localhost',
          port: vm['port'] v '22',
          password: vm['password'] v 'vagrant'
      }

      SSHKit::Host.new(host_options)
    end
  end
end
```

We can't help acknowledging Ruby's power and charms.

Does Ruby edge out Python? Is *Bash* to be ignored? Not for the admin: as they need good knowledge of *Bash* to follow what's going on with the system. And in addition to *Bash*, every sysadmin should know a little Perl, Python *and* Ruby, but have in-depth knowledge of the one that they prefer.

> **"In addition to Bash, every Linux admin should know a little Perl, Python and Ruby."**

1st Ruby ★★★★★

Web: www.ruby-lang.org **Licence:** GPLv2 or 2-clause **Version:** 2.2.0

» Powerful, expressive and very quick to learn.

2nd Python ★★★★★

Web: www.python.org **Licence:** PSFL **Version:** 3.4.2

» Multi-paradigm, encourages good practices and great community.

3rd Perl 5 ★★★★☆

Web: perl.org **Licence:** GPL or Artistic License **Version:** 5.20

» Still a great friend to the sysadmin.

4th newLISP ★★★☆☆

Web: www.newlisp.org **Licence:** GPL **Version:** 10.6.1

» So powerful, it deserves to get more use.

5th Bash ★★★☆☆

Web: www.gnu.org/software/bash **Licence:** GPLv3+ **Version:** 4.3.30

» Doesn't do everything, yet remains essential.

Over to you...

We don't want to start a holy language war, but we would love to hear what you use. Tell Linux Format: **lxf.letters@futurenet.com**.

Also consider...

While *Bash* falls down in some areas, traditional shell scripting could also be represented by Zsh, which has some small but useful differences, such as better access to positional variables, and the ability to extend the shell through widget functions. Nevertheless, it's not a rival to our other choices, and nor is PHP, despite its use in command scripts by some devotees. Instead, our left-field alternative is Rebol (Relative Expression Based Object Language), whose leap towards software freedom two years ago may have come too late to ensure universal popularity. However, Rebol has an elegant design and syntax, and a useful read–eval–print loop (REPL) console.

Rebol's 'dialecting' (using small, efficient, domain languages for code, data and metadata) equips it for just about anything. It's particularly good at dealing with the exchange and interpretation of information between distributed computer systems, but also powerful, terse shell scripts. If you're looking for a new language, give it a try.

Everything you always wanted to know about NoSQL, but were too afraid to ask. Discover why admins love this high-speed system.

The world of databases moves slowly (but perhaps not when transacting), so when a revolution hits it can take decades for the repercussions to be felt. Coined back in 1998, NoSQL are databases that started life not using the then standard query language SQL. But more revolutionary the actual database designs themselves, moved away from the standard relational model altogether for speed and ease of design.

As you might expect by the name, even though NoSQL databases were originally designed not to use SQL, they can now, instead they use various different query languages. While these originally might have appeared in 1998, NoSQL didn't gained prevalence until the late Noughties when it was adopted as a rallying Twitter hashtag for a group of non-relational distributed database projects that were after something small and unique.

If you are wondering whether or not it's worth considering NoSQL databases, you should be aware that according to DB-Engines Ranking (**https://db-engines.com/en/ranking**), *MongoDB*, a popular NoSQL database, is currently the fifth most popular after Oracle, *MySQL*, Microsoft SQL Server and *PostgreSQL* – and even Oracle has a NoSQL version of its famous database.

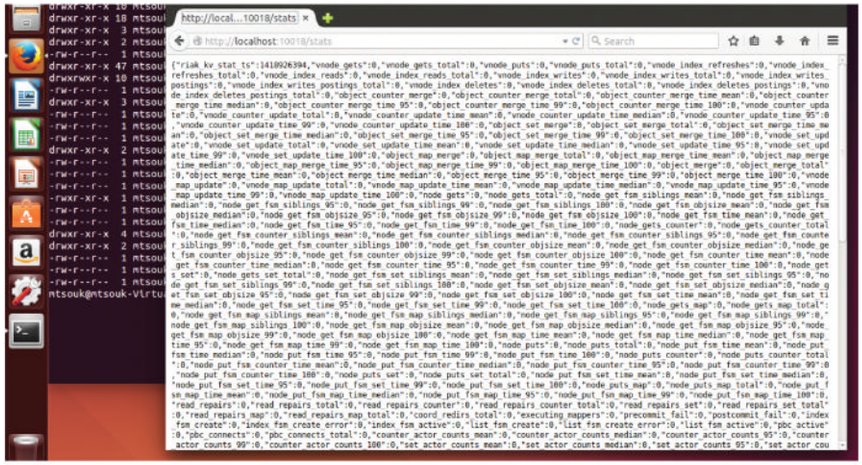
The problem with relational databases is that in order to store complex information you have to deconstruct it into bits and fields, and store it in lots of different tables. Likewise, in order to restore the data, you have to retrieve all those bits and fields and put them back together. Neither of those two tasks is efficient particularly if you have a big and busy website that's storing and querying data all the time.

> **"NoSQL DBs are designed for the web and don't support joins and complex transactions…"**

The next logical step is to use many machines to run your database, but that also creates a problem because relational databases were originally designed to run as single-node systems. So, large companies, such as Google and Amazon, developed their own database systems, *Bigtable* and *Dynamo* respectively, that were quite different from traditional relational database systems, and which inspired the NoSQL movement.

It's quite difficult to define what a NoSQL database is but you can identify a few common characteristics among NoSQL databases: they are non-relational; open source (although not always); schema-less; easy to be distributed on many machines (again, not always) and trying to serve data from the 21st century web culture.

So, NoSQL databases are designed for the web and don't support joins, complex transactions and other features of the SQL language. Their terminology is also a little different, but lets dive into the details.



Every Riak server has a web interface. In this case, we're accessing the server statistics, using http://localhost:10018/stats/ The port number and IP are defined in riak.conf.

The main advantage of a NoSQL database is that they are suited to and efficient for big data and real-time web applications. They also offer easy scalability, and enable you to implement high availability painlessly. They are also generally easier to administer, set up and run, and they can store complex objects. Additionally, it's easier to develop applications for and with NoSQL databases. Their schema can change easily without any downtime because, in reality, they have no schema. Most of them, with the exception of Oracle NoSQL, are open source projects.

Key disadvantages of NoSQL databases include the fact that they require a totally new way of thinking and that you still need a DBA on large and/or critical projects. If your company needs to use both SQL and NoSQL databases, you will have two entirely different systems to program and administer and therefore will need even more people. Being relatively new, they are not as mature as relational databases; therefore choosing a NoSQL database for a critical problem may not always be the safest solution, but this will not be a problem in a couple of years. The last disadvantage is the fact that although they look like they have no schema, you will need to assume an implicit schema in order to do some serious work with your data. This isn't unexpected because as long as you are working with data, you cannot get away with having a schema, even an informal one.

There are several kinds of NoSQL database each of them being good in one or more areas but not all. You can categorise NoSQL databases according to their data model:

» **Document** This is a very common data model. It thinks of the database as a big storage for documents where each document is a multipart data structure that's usually represented in forms of JSON. You can still store documents in any format you want. *MongoDB*, *CouchDB* and *RavenDB* are representative document NoSQL databases.

» **Key-Value** This is also a common data model that's similar to the hash map data structure, where you have a key and you ask the database to return the value stored for that particular key. The value can be anything from a single number to a whole document. The database knows nothing about the stored data. Examples of key-value NoSQL databases include *Riak*, *Redis* and Project Voldemort.

» **Column-family** This is a rather complex data model. You have a 'row key' that enables you to store and access multiple column families. Each column family is a combination of columns that fit together. Row keys must be unique within a column family. The data model might be more complicated than the others but it results in faster retrieval times. Examples of column-family NoSQL databases include *Cassandra* and *Apache HBase*.

» **Graph** This model is totally different from the other three as it is based on the Graph structure. As a logical consequence Graph NoSQL databases handle hierarchies and relationships between things very well; doing similar things with a relational database would be an extremely challenging and slow task. *Neo4j* is a graph NoSQL database. For this article, we'll be using *Riak* as our NoSQL database test case.

Installing Riak

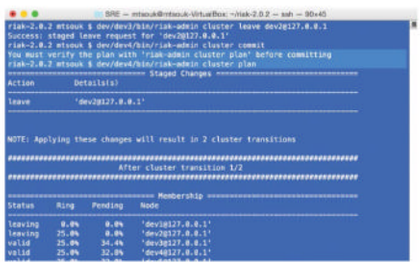
The first thing you should know before installing *Riak* is that you need Erlang on your system. The best way to install *Riak* is by compiling it from source because you have better control and a totally autonomous build of *Riak*. Follow the next steps:

```
$ wget http://s3.amazonaws.com/downloads.basho.com/riak/2.0/2.0.2/riak-2.0.2.tar.gz
$ tar zxvf riak-2.0.2.tar.gz
$ cd riak-2.0.2
$ make rel
```

Alternatively, you can get the *Riak* source code from GitHub and compile it as before:

```
$ git clone git://github.com/basho/riak.git
$ cd riak
$ make rel
```

Both ways should work without any particular problems; we used the first way to compile *Riak*. After successfully compiling *Riak*, you can find its main binary files inside the **./rel/riak/bin** directory. In the same directory that you build Riak, you can run **make devrel** and get eight ready to run Riak databases that we will use as example servers. This is the main



Generating a *Riak* cluster with five nodes is pretty easy. Honest!

Map and Reduce

MapReduce is an advanced querying technique and a tool for data aggregation used in NoSQL databases. It's an alternative technique for querying a database that differs from the usual declarative querying techniques. You give instructions to the database on how to find the data you are looking for and MapReduce tries to find the data. (*See the top of p170 for a simple example of how MapReduce works.*) Using MapReduce can be very tricky sometimes. Nevertheless, it enables you to create queries that would have been extremely challenging to create using SQL.

Once you understand the MapReduce process and practice it, you will find it both very reliable and handy. The MapReduce solution takes more implementation time but it can expand better than an SQL solution. It provides some flexibility that's not currently available in the aggregation pipeline. The tricky thing is deciding whether or not the MapReduce technique is appropriate for the specific problem you are trying to solve. This kind of knowledge comes with experience!

reason to get the *Riak* source code and compile it for yourself.

Before we continue with the rest of the article, we need to introduce you to some terms. First, a *Riak* node is analogous to a physical server. A *Riak* cluster is a 160-bit integer space which is divided into equally-sized partitions. Partitions are, in turn, the spaces into which a *Riak* cluster is divided. Each vnode in *Riak* is responsible for a partition. Vnodes coordinate requests for the partitions they control. A *Riak* cluster can have many nodes that reside on the same or different physical machines. A ring is a 160-bit integer space equally divided into partitions, and a bucket is a namespace for data stored in *Riak*. Internally, *Riak* computes a 160-bit binary hash of each bucket/key pair and maps this value to a position on an ordered ring of all such values.

As you will see later in this article, any client interface to *Riak* interacts with objects in terms of the bucket and key in which a value is stored, as well as the bucket type that is used to set the properties of the bucket.

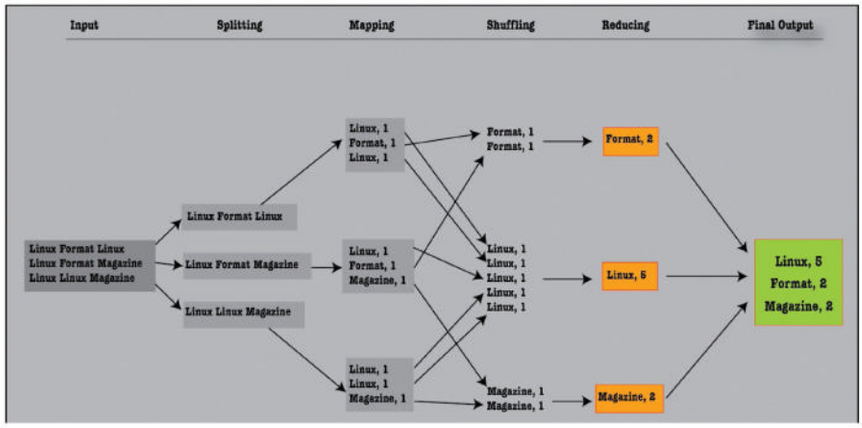
The default mode of operation for *Riak* is to work as a cluster consisting of multiple nodes. *Riak* nodes are not clones of one another by default.

You can start three example *Riak* database servers – you don't have to start all eight Riak servers – by executing the next commands:

```
$ ./dev/dev1/bin/riak start
$ ./dev/dev2/bin/riak start
$ ./dev/dev3/bin/riak start
$ ./dev/dev1/bin/riak start
Node is already running!
```

Each *Riak* server offers a web interface (*see top of p169 for an example of what you will see after connecting to a Riak server*). The port number and the server IP address are defined inside the **riak.conf** file. This is a plain text file that you can edit. The following command reveals the IP and the port number that each Riak server listens to:

```
$ grep listener.http.internal `find ./dev -name riak.conf`
./dev/dev2/etc/riak.conf:listener.http.internal = 127.0.0.1:10028
./dev/dev1/etc/riak.conf:listener.http.internal = 127.0.0.1:10018
./dev/dev3/etc/riak.conf:listener.http.internal = 127.0.0.1:10038
```



A MapReduce example. It may look simplistic but MapReduce is a very powerful technique. Attempting the same with SQL would be extremely difficult.

And so on… Every node in *Riak* has a name associated with it. You can change the name by changing the nodename variable of the **riak.conf** file. The first server (dev1) uses port number 10018, the second *Riak* server (dev2) uses port number 10028 and the third (dev3) uses port number 10038. *Riak* versions prior to 2.0 used a configuration file called **app.config** which has been replaced by **riak.conf**.

The easiest way of finding out if a *Riak* node is up or down is via the **curl** command and the web interface of the node to ping it:

```
$ curl http://localhost:10018/ping
OK
$ curl http://localhost:10038/ping
curl: (7) Failed to connect to localhost port 10038: Connection refused
```

Alternatively, you can use the following:

```
$ ./dev/dev1/bin/riak ping
pong
$ ./dev/dev6/bin/riak ping
Node 'dev6@127.0.0.1' not responding to pings.
```

The advantage of the 'curl way' is that you can run it from a remote machine – provided that the *Riak* server also listens to an external IP address – without having to login to the machine that runs the *Riak* node. You can stop the dev1 *Riak* server by executing the **./dev/dev1/bin/riak stop** command.

Riak uses epmd – the Erlang Port Mapper server – which plays a crucial part in the whole *Riak* operation. The epmd process starts automatically by the **erl** command if the node is to be distributed and there's no running instance present. The epmd process enables *Riak* nodes to find each other. It's an extremely lightweight and harmless process that can continue to run even after all *Riak* nodes have stopped. You may kill it manually after you stop all *Riak* nodes, but this isn't compulsory. The following command lists all names registered with the currently running epmd process:

```
$ epmd -names
epmd: up and running on port 4369 with data:
name dev3 at port 49136
name dev1 at port 55224
name dev2 at port 48829
```

Storing and retrieving data

You can connect to *Riak* dev1 server and store a document using the web interface:

```
$ curl -v -X PUT http://127.0.0.1:10018/riak/LXF/test -H "Content-Type: text/html" -d "<html><body><h1>This is a test.</h1></body></html>"
```

Riak benchmarking

Basho (**http://basho.com**) offers a benchmarking tool for *Riak* written in Erlang. You can get and install it with:

```
$ git clone git://github.com/basho/basho_bench.git
$ cd basho_bench
$ make
```

You should then run **./basho_bench myconfig.config** to get the tool collecting data, and either create a **myconfig.config** yourself or modify an existing config. Existing files reside in the **examples** directory. We used the **examples/basho_bench_ets.config** file as a starting point and added the **{riakclient_nodes, ['dev1@127.0.0.1, 'dev2@127.0.0.1']}.''** line.

Basho Bench creates one Stats process and workers based on what's defined in the concurrent configuration setting in **myconfig.config** file. As soon as these processes are created and initialised, **Basho Bench** sends a **run** command to all worker processes and this initiates the testing. The Stats process is notified every time an operation completes. It also gets the elapsed time of the completed operation and stores it in a histogram.

All the results are inside the **tests** directory. The latest results can be found using the **./tests/current/** soft link. To generate a graph against the current results, run **make results**. [*See the bottom of p171 for a sample output.*]

What is actually stored in the **/riak/LXF** test location is what follows the **-d** option. When you successfully insert a new value, *Riak* will return a 204 HTTP code. As you already know, *Riak* is a key-value store, therefore in order to retrieve a value you need to provide a key to *Riak*. You can connect to *Riak* dev1 server and ask the previously stored document by going to the **http://127.0.0.1:10018/riak/LXF/test** URL. Every URL follows the **http://SERVER:PORT/riak/BUCKET/KEY** pattern. The following command returns the list of available buckets:

```
$ curl -i 'http://127.0.0.1:10018/riak?buckets=true'
HTTP/1.1 200 OK
Vary: Accept-Encoding
Server: MochiWeb/1.1 WebMachine/1.10.5 (jokes are better explained)
Date: Fri, 19 Dec 2014 21:13:37 GMT
Content-Type: application/json
Content-Length: 33

{"buckets":["LXF","linuxformat"]}
```

The following command returns the list of keys in a bucket:

```
$ curl 'http://127.0.0.1:10018/buckets/LXF/keys?keys=true'
{"keys":["test2","test","test3"]}
```

Most of the times, you are going to use a script written in a programming language to access a *Riak* database. The following is a Python script that connects to a *Riak* database, stores and retrieves a document:

```
import riak
# Connect to the cluster
client = riak.RiakClient(pb_port=10017, protocol='pbc')
# The name of the bucket
bucket = client.bucket('python')
# "myData" is the name of the Key that will be used
aRecord = bucket.new('myData', data={
 'Name': "Mihalis",
 'Surname': "Tsoukalos"
 })
# Save the record
aRecord.store()
# Define the key for the record to retrieve
myRecord = bucket.get('myData')
# Retrieve the record!
dictRecord = myRecord.data
# Now print it to see if all this actually worked.
print dictRecord
$ python myRiak.py
{u'Surname': u'Tsoukalos', u'Name': u'Mihalis'}
```

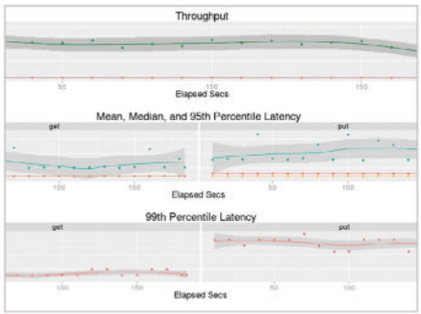
The pb_port value of 10017 is defined in the **./dev/dev1/etc/riak.conf** file using the listener.protobuf.internal parameter. This is the Protocol Buffers port that is used for connecting to the *Riak* Cluster.

Due to the flexibility in the way that a NoSQL database stores data, inserting, querying and updating a NoSQL database is more complex than a database that uses SQL.

Generating a Riak cluster

Creating and manipulating clusters in *Riak* is relatively easy with the help of the **riak-admin** command. If you try to add a node that's not already running to a cluster, you will fail with the following error message:

```
$ dev/dev2/bin/riak-admin cluster join dev1@127.0.0.1
Node is not running!
$ ./dev/dev2/bin/riak start
$ dev/dev2/bin/riak-admin cluster join dev1@127.0.0.1
Success: staged join request for 'dev2@127.0.0.1' to 'dev1@127.0.0.1'
$ dev/dev2/bin/riak-admin cluster join
```



› *Riak* offers a benchmarking tool called *Basho Bench*. The graph is produced with R.

```
dev1@127.0.0.1
Failed: This node is already a member of a cluster
```

Similarly, if you try to join a node to itself, you will get an error message:

```
$ dev/dev1/bin/riak-admin cluster join dev1@127.0.0.1
Failed: This node cannot join itself in a cluster
```

The following command shows the members of an existing cluster:

```
$ dev/dev2/bin/riak-admin status | grep members
ring_members : ['dev1@127.0.0.1','dev2@127.0.0.1']
$ dev/dev1/bin/riak-admin status | grep members
ring_members : ['dev1@127.0.0.1','dev2@127.0.0.1']
$ dev/dev3/bin/riak-admin status | grep members
Node is not running!
```

Another useful command that shows the status of the nodes is the following:

```
$ ./dev/dev1/bin/riak-admin member-status
```

The joining status is a temporary status and will become valid when all changes that are waiting in a queue will be applied and committed. If you want to force changes, you should execute the **riak-admin cluster commit** command.

If you run the **riak-admin member-status** command again you will see the new status of the dev3 node, and the **riak-admin cluster plan** command displays the changes that are about to be applied.

For a node to actually leave the cluster (*see bottom of p169 to see what an interaction with a cluster of five nodes looks like*), you must first review the changes using the **riak-admin cluster plan** command and then commit them with **riak-admin cluster commit**.

So far, you won't have seen any security when interacting with a *Riak* database. Nevertheless, *Riak* supports users and passwords. You can find a lot more information on how *Riak* deals with authentication and authorisation at **http://bit.ly/RiakDocsAuthz**.

Data consistency

Data consistency in databases is critical. ACID (Atomicity, Consistency, Isolation and Durability) is a set of properties that guarantee that database transactions perform reliably. Atomicity means that when you do something to change a database, the change should work or fail as a whole. Isolation means that if other things are taking place at the same time on the same data, they should not be able to see half-finished data. Durability refers to the guarantee that once the user has been notified of the success of a transaction, the transaction will persist, and won't be undone even if the hardware or the software crashes afterwards.

Graph databases perform ACID transactions by default, which is a good thing. On the other hand, not every problem needs 'perfect' ACID compliance. *MongoDB* is ACID-compliant at the single document level, but it doesn't support multiple-document updates that can be rolled back. Sometimes, you may be OK with losing a transaction or having your DB in an inconsistent state temporarily in exchange for speed.

You should carefully check the characteristics of a NoSQL database and decide if it fits your needs. Nevertheless, if data consistency is absolutely critical, you can always implement it in code if it's not fully supported by your NoSQL DB. Keep in mind that this might be non-trivial especially on distributed environments.

PHP: Build a feed aggregator

Combine your favourite RSS feeds and learn a little PHP at the same time.

Have you ever wondered how some websites display combined data from various websites? This is a bit of trickery that can be done using a few methods, including having select database privileges from the sources, scraping data and using RSS feeds – or, in fact, any combination of these three methods. In this tutorial we'll focus on how to retrieve and combine data from other sources using RSS feeds. RSS, which stands for Rich Site Summary (and later Really Simple Syndication), has been around a long time – and stems from Ramanathan V. Guha's work at the Apple Computer's Advanced Technology Group in the mid-1990s – and you probably use them every day to display news feeds among other useful data.

RSS is generally XML files that contain data within organised tags, eg if you go to Craigslist and browse through regular listings you will see the usual list of items. This source code is HTML. But, if you look at the Craigslist RSS feed file for those same items you won't see HTML code in your browser you'll see an XML file that stores each listing within a parent 'item' tag. Each item tag contains information for an entry such as title, date and description. Grabbing and aggregating RSS feeds with a simple PHP script is a fast and simple way to tap into your desired data and output the results you want. These RSS feeds are provided from many large classified and auction websites (such as eBay, eBay Classifieds, Monster and Craigslist etc) and many other websites, such as job websites (eg CareerBuilder and Indeed allow you to obtain XML feeds that can be parsed too).

In many instances, you can just use a specific URL to acquire RSS feeds, while at other times you'll need to use an API with a specified publisher key in order to obtain the feeds. One example of websites that require the publisher key for the API is the popular job site, Indeed.com.

Since we had to pick an example topic, we'll explain how to aggregate RSS from the various sources we've mentioned above. Our examples will show you how to narrow down a job search to location and job position or you could just as easy make this a personal RSS and XML feed aggregator to display bargains of items you want to buy from eBay, Craigslist etc.

In addition to the above, the output will load in your browser on a local web server, such as a LAMP setup running on the Ubuntu distribution (distro). Installing LAMP only takes a minute or two but we'll explain how to do that as well. When your output is shown in the web browser, it will link back to the original post and you can do what you want from there, eg inquire about the job position etc.

Quick tip

You can always run a cron job to open RSS feeds and save them as XML files. Then, you can use these files instead of your URL sources.

Getting started

So let's roll up our sleeves and get ready to find out how to have the edge on data. The first thing you'll need to get started with aggregating and parsing RSS feeds is a working web server along with a few extra packages, such as PHP. Since it only takes a minute, the instructions to set it up are shown next. For those who are thinking to yourself "Not another web server installation" you can skip ahead. To install the *Apache* server, run the code below:

```
sudo apt-get update
sudo apt-get install apache2
```

Next, install the PHP packages with:

```
sudo apt-get install php5 libapache2-mod-php5 php5-mcrypt
```

In this tutorial, we'll use a downloadable script to parse RSS feeds and show the methodology to do it manually. The parser we'll be using is *Magpie RSS*. Note: This is hosted on Sourceforge (**http://bit.ly/MagpieRSS)** but there are alternatives to it that don't use the site, such as *Simple Pie* (**http://simplepie.org/downloads**).

Styling your feeds and links

Since the output is displayed in a basic table, it currently doesn't look that stylish, but you could easily add a plugin that will do the job of smarting up your output. If you do a web search for 'best responsive bootstrap table plugins' you will see that there are many options and tutorials for adding styles, but we'll quickly explain how to using a plugin called Datatables, which can be downloaded at **https://datatables.net/download/download**.

Not only will Datatables provide a resizing table for many devices and is searchable, you can also select how many entries you want to see on a page.

Although we've provided **index_styled.php** on the Linux Format site, I will quickly explain how easy it is to add the plugin. The first step is to go into the file and change the doctype to `html 5`. Thus, the very first tag `<html>` can be changed to `<!DOCTYPE html>`. After that, you need to add a head section with links to the required files. The head data is shown below:

```
<head>
<link rel="stylesheet" type="text/css" href="//cdn.datatables.net/1.10.7/css/jquery.dataTables.css">
<script type="text/javascript" charset="utf8" src="//code.jquery.com/jquery-1.10.2.min.js"></script>
<script type="text/javascript" charset="utf8" src="//cdn.datatables.net/1.10.7/js/jquery.dataTables.js"></script>
<script type="text/javascript">
$(document).ready( function () {
    $('#example').DataTable();
} );</script>
</head>
```

Note that you can use the downloaded files from the Datatables website or you can use links from the CDN (Content Delivery Network). There's just one more thing left to do and that's to add two attributes to the table tag:

```
id="example" class="display"
```

Now, if you load the page in a web browser you will see it looks a lot more stylish.

For those with some responsive coding experience, you may want to take things a little further, eg you could use bootstrap media queries and classes to resize the text and remove columns when the width of the screen being used changes.

Once you download *Magpie*, you place it inside the **www** or **html** folder, eg new installations of Ubuntu with *Apache* will locate the web folders in the **/var/www/html** path while older installation such as Ubuntu 12.04 will use **/var/www** as the root web folder.

Whatever your root folder may be, you will now create a folder called **rss**. Within this folder, you add the extracted *Magpie* folder that will have a name like **magpierss-0.72**. For simplicity right-click the folder and rename it to **magpie**.

In addition to the **magpie** folder, you'll have two files; **index.php** and **customs.php**. The **index.php** file is used to display your content and open it in a browser and **customs.php** contains the functions to take the URLs and gather the data from the RSS sources (Craigslist, Ebay Classifieds, Monster.com, Careerbuilder.com and Indeed.com). Note: Most of the feeds in the coding sample are parsed with *Magpie*, with the exception of those from Indeed and Careerbuilder. The parsing for the latter two are done using the PHP built-in class DOMDocument().

The code used in this tutorial is provided on the Linux Format website, which means the only lines you may want to edit are the actual URLs array that reside near line 3 in the **index.php** file. This is your source list of URLs for the RSS feeds that you want to aggregate. However, you may want to make some changes to the custom parser starting near Line 108 in the file **customs.php**. At the moment, the custom URLs are set to use indeed and Careerbuilder.

You will have to uncomment the URLs for Careerbuilder and Indeed since they both require your publisher key. If you sign up and get your developer keys, all you need to do is replace `YOUR_KEY_HERE` with the proper keys.

The Indeed Publisher Program can be accessed at **www.indeed.com/publisher** while information for how to gather search results for the Careerbuilder website is available at **http://bit.ly/CareerBuilderAPI**.

Now that we've covered the basics, it's time to go over the code and explain how it works. Start by opening the **index.php** file in a web browser. The URL on your local machine would be **http://localhost/rss or http://localhost/rss/index.php**. Near the top of the file, an array called `$urls` is created and contains the feeds you want to aggregate.

Getting RSS feeds from various sources is quite simple, but each provider will have a different method to find them. Some are definitely easier to find than others. Let's start with Craigslist. If you navigate to **http://craigslist.co.uk** and look at jobs in skilled trade/craft in London, you will see many listings. You could just look at the bottom right corner of your screen and click the RSS button to get the feed, or you could narrow your search and get a feed from that.

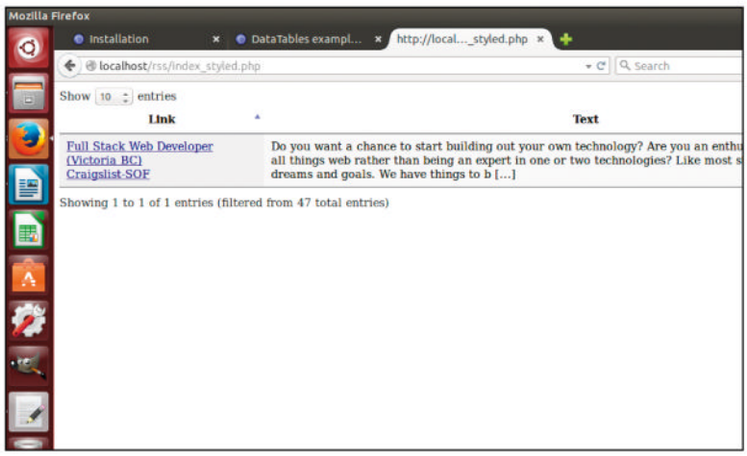
At the upper left of your screen you'll need to check the box 'contract'. Next, click the yellow RSS icon on the bottom right and you will see the feed page. The URL in your browser is an example of what you could add into your feed URLs into the top of the **index.php** page.

Multiple sources

Another example would be gathering eBay RSS feeds, and eBay's feeds aren't so obvious. But, you can narrow your search to whatever you want by adding **&_rss=1** to the end of the URL in your web browser to see a new page with the desired RSS feed.

The next two lines include the file *Magpie* will use to parse and the **customs.php** file, which uses custom functions and checking. Now, things get a little trickier. But, if you look at around line 108 in the **customs.php**, you'll see a function that checks if the $urls array exists. When it confirms the array exists, it runs through a foreach loop for each URL.

If the first set of conditions are met then it checks the URL to see if a part of the string doesn't contain the text 'indeed' »



There are many plugins available, such as Datatables (pictured) that make it easy to stylise your desired output.

or 'careerbuilder'. The PHP built-in function strpos() is used to determine if the word is positioned in the URL. If the URL doesn't contain indeed or careerbuilder, the custom_url($url) function with the URL will be fired. This function is near the top of **customs.php**.

After the URL is passed into the custom_url() function, it will then get tossed into the fetch_rss() function, which is a *Magpie* function located in the **magpie/rss_fetch.inc** file.

In short, *Magpie* grabs the URL and returns $rss_items. At this point, the elements in the array which contain the title, description, link are separated into individual variables with names like $title, $description and $date.

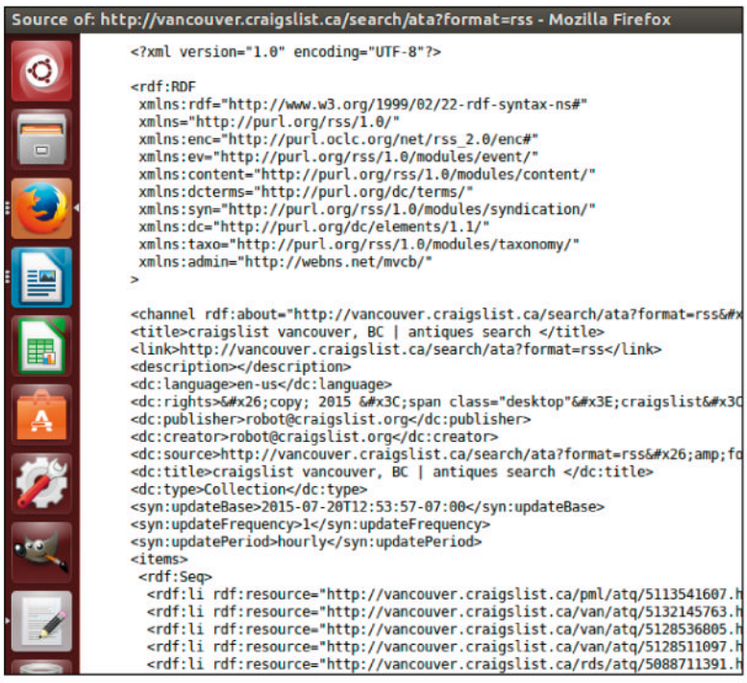
Note that Craigslist uses $item['dc']['date'] for the date while other sources tend to use $item['pubDate']. This sample code makes it easy to check out a feed, but see the tag and add a condition, if necessary.

Once the variables from the feed have been set, there are conditions that create an origin variable called $origin, which are based on the text from the URL. By doing this, you can see the source in the browser when you run the script.

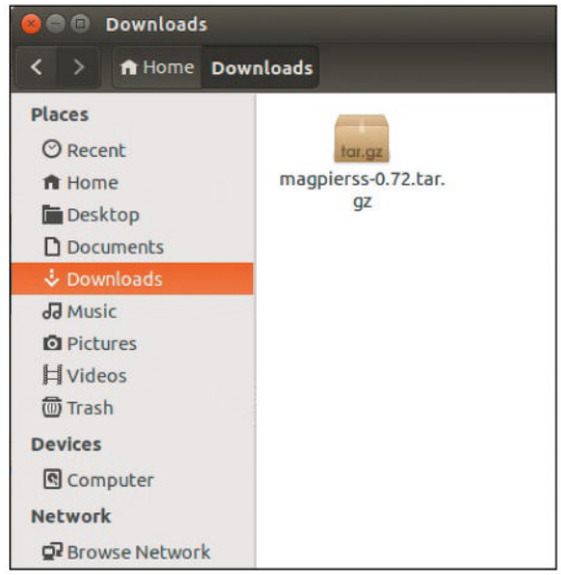
Finally, at the end of this function, the $tot_array grows as each URL runs through the function. The $tot_array contains all the variables, such as the date and title they were created previously. The array contains strings that contain each piece of data that is separated by commas.

Functions and conditions

Now looking back a little, do you remember that the custom_url() function only runs if the URL isn't from Indeed.com or Careerbuilder? What about if it is from either Indeed.com or Careerbuilder? Let's answer that now. Since both Indeed.com and Careerbuilder have different XML tags, they each will go through a set of parsing specific to them. In fact, it's good to remember this because you can apply the same techniques to other XML files that aren't in this coding example – just follow the pattern and make the required adjustments.



The source code of a feed, which contains entries in an organised manner.



***Magpie RSS* can be downloaded from Sourceforge. (Don't worry, it's just a TAR and not an installer.)**

The code is set to check if the URL is from Indeed.com first. If it isn't, the check will be performed for Careerbuilder. Now, let's assume it's your custom API from Indeed.com that you can get free when you sign up.

First, a new DOMDocument is created that's encoded using utf-8. After that, the $dom->load($url) method is called and the URL is loaded. At this point, you will have a list that comes out. With an Indeed XML file, each result from it is listed as 'result' while each listing from Careerbuilder would be 'JobSearchResult'.

For the rest of this example, only the explanation of Indeed.com and its tags will be covered. To show what is being interpreted, you can simply take your API URL from Indeed.com and load it in a browser. It will be an XML file and have all the tags and data for each listing.

Moving on, a foreach loop is used to parse each result. As you can see from the code on the Linux Format website, the getElementsByTagname() method is used and some other coding to get the desired values; just like *Magpie* did with the other RSS feeds.

You will note that there will be some coding and functions such as htmlspecialchars_decode() that is used when creating the value variables such as $jobtitle_value. Basically, decoding converts character code to the desired symbol. With Indeed, you will also see that if there's a $jobkey_value, a new $myurl_value is created. The reason for this is that it will provide a link to the actual company listing, not the Indeed post.

Note that with each foreach loop, the $tot_array gets larger with each loop and contains all listings when the loop comes to an end. Essentially, the $tot_array is everything from both *Magpie* and the custom XML parsing. Now can go back to the **index.php** and carry on where we left off, which was near line 9.

If we got our listings from all of our sources, we have a $tot_array. Code-wise, if the array exists, it runs through a usort() function that calls the cmp() function. This is used to parse the array so the newest listings show up on top.

Once that function runs its course the HTML code displays a table for the results. The left side will be the title

Using scraping

Scraping is an alternative method to using if an RSS feed if one isn't available. This is a technique where source code from a web page is analysed for patterns of data that you might be interested in.

Some of your sources may not supply RSS feeds. But, since they likely provide some kind of listings using HTML web page lists, tables or ordered tags, you can always look at the source code, find the relevant pattern and then scrape the data out.

Often, entries of this kind will have a unique class or some other set of attributes that can make it easy to identify a beginning and end to the useful data.

To scrape a page with PHP, you can use one of its functions, such as file_get_contents() or curl() to retrieve the source code of a page that you fancy. Curl has many more advantages, eg you can use a proxy such as *Tor*. However, you must exercise caution when scraping data. You may want to contact the copyright holder and keep the content unpublished from the web so the content isn't crawled.

Once the content is gathered into a string, you use regular expressions and PHP functions, such as preg_match_all() and preg_match() to separate the file into entries. In the end, you'll have the same data as you would with RSS feeds. You could combine the scraping method with RSS feeds if you want to keep all of your data in one place, with only the addition of a few lines of code.

and link to the post, the middle will be the description and the date will be at the end.

To get a better understanding of how each of the listings is displayed, look around line 28. As you will see, the array_unique() function is used to ensure our results are unique and have no duplicates. Then, near line 33, a foreach loop is used to parse the $tot_array.

For each listing in the array, new variables are made in order to create the title, date, URL and description entries. Since each item in the $tot_array, which is also known as $tot, comes through as a string with commas separating the values, the explode() function is used to make another array called $all.

The array $all contains the actual variables. Since the date was the first item in the array, you will see it is $all[0]. In addition, the title was the second element and took its value from $all[1]. If you have limited array knowledge, keep in mind that all indexed arrays begin with 0, not 1 as you may expect.

Finishing off your aggregator

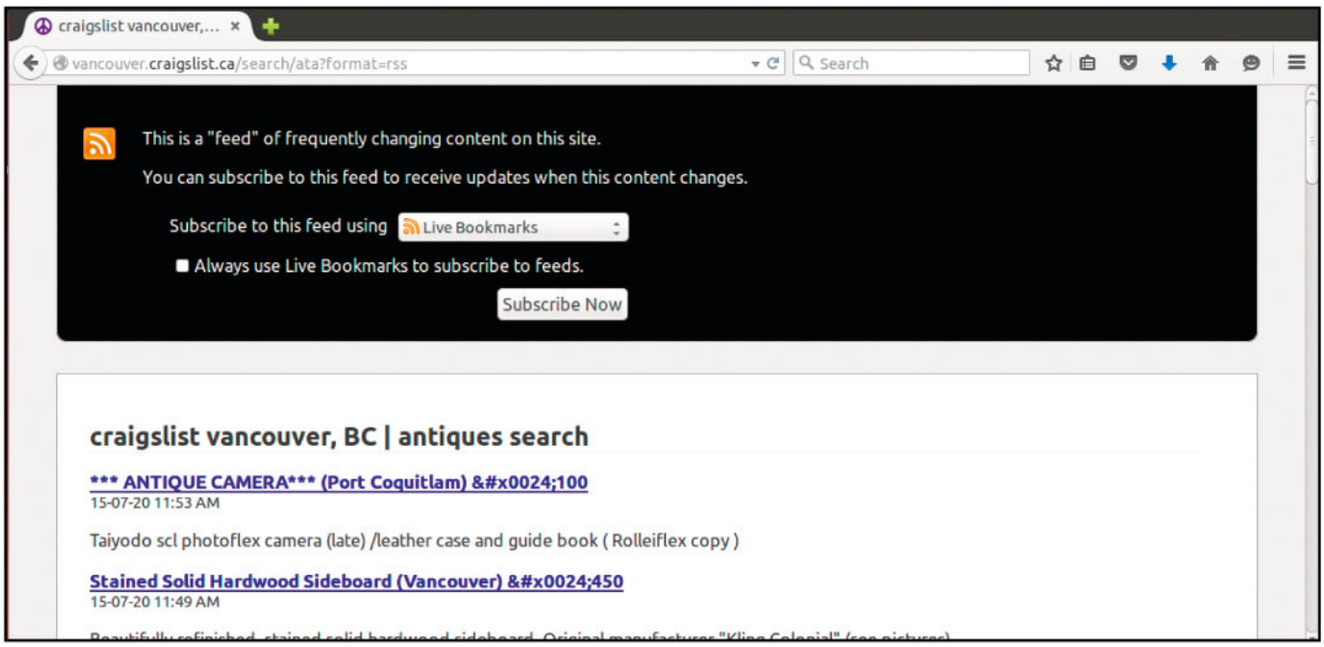
Getting back to the latest date variable, you will see it contains other code. Well, this is because the date function is used to display the results with year, month and day. The strtotime() function is also included to gather the date from the loop located in the $all[0] element and convert it to a Unix timestamp.

If you look at the source code in the **index.php** file, you will see that a table row is displayed for each entry and each value is shown between the <td></td> tags. At the bottom of the file you will also see the usual closing </body> tag and closing </html> tag. With that done, you have a working RSS aggregator and an explanation covering the details of our lean-coded script. But, currently it does look a little too old school and needs some style. As far as usability goes, you could add and draw the URLs from a database so that you could have various aggregated sources at your fingertips.

Another feature you could add easily enough is listing management. Since each listing will have a specific URL, you could make buttons to insert the data into a database, eg one button could be used to make it a favourite and another could be to mark it as a dud that you don't want to see in the list the next time you open the page.

So there you have it. You have your own RSS feed aggregator that will display the source and job description by date. Whether you're looking or comparing jobs, trying to find bargains, or keeping up to date from multiple news sources, this bit of PHP scripting can be used to find the prize data that you are after.

Here's a sample RSS feed shown in your web browser.



Guru Guide
Nobody goes deeper into technology

HACKER'S MANUAL 2016

178 PACKED PAGES! EXPERT GUIDES TO GETTING THE MOST FROM LINUX

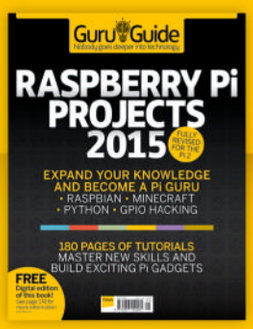
LINUX Completely customise your experience
PRIVACY Lock down every byte of your data
DISTROS Try out the latest cutting-edge distros
NETWORKS Clone websites and entire machines

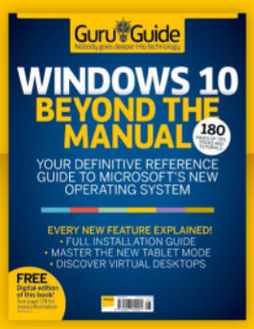
Packed with comprehensive features for the serious Linux user

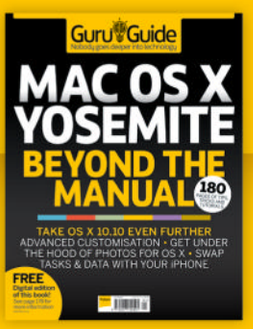
Find tools and techniques to help you work faster and more securely

Use advanced Linux skills to take control of all your networks and hardware

LIKE THIS? THEN YOU'LL ALSO LOVE...







Visit **myfavouritemagazines.co.uk** today!