tSc's Inventory for Debian Family Hardening

Release: October 25, 2016

This is a collection of instructions to optimize and harden Debian and Ubuntu operating systems and their derivatives, yet most of it applies to other Linux distros as well. All procedures have been tested on current installations of Debian Stable and Ubuntu LTS releases.

Not everything listed is necessary and some things you would not want without specific requirement. The focus is on desktops and laptops while suitable audiences are just about anyone—home users, businesses, non-profit and other organizations, independent professionals and government.

Depending on your choices, the covered threat model can span from protecting a home computer from low-level malware to sandboxing vulnerable processes by mandatory access control, to securing data against (publicly known methods of) forensic recovery. In any event, the end result will be a more efficient and secure system compared to default.

The link below is the only official source of this document. If you didn't get it from there, first check if you have the latest release before going further.
http://thesimplecomputer.info/inventory-for-debian-family-hardening

Table of Contents

Variables specific to your system or choice are shown in **orange**.

Preparation

Pre-Prep

sudo
sudo is used throughout this document. If you did not create a root account when you installed Debian, then your regular user account is already in the sudoers group. If you installed Debian with a root account, then your regular user isn't in the sudoers group so you must add it.

Open a root terminal or change to su and run:
```
adduser user_name sudo
```

Then log out and back in.

gedit with pkexec
The command *pkexec* is used to launch gedit as root in this doc. Using polkit to launch GUI applications as root is preferable over sudo or gksu but you must create a configuration file to do this. Skip down to the Permissions area for launching graphical apps as root and use the given gedit polkit action file.

If you just prefer working from the root account, that bypasses the need for both of these.

Bootable USB Stick

This will erase everything on the target device! First unmount it.
```
sudo umount /dev/sdx
```

Then write the image.
```
sudo dd if=/path/to/image.iso of=/dev/sdx
```

ATA Secure Erase

If your mechanical hard drive was manufactured after the year 2001 and is larger than 20 GB in size, then it's likely capable of ATA Secure Erase. This does the best job of wiping an entire drive because it bypasses its onboard S/ATA controller, using the firmware to overwrite absolutely everything, including bad blocks and protected areas and restores the firmware to factory default condition.

For solid state drives, this can be hit or miss. Not all SSDs support ATA Secure Erase and not all which do, implement it correctly. Current secure erase technology for SSDs, sometimes even what's provided by the manufacturers, does not always mean that data on disk really becomes inaccessible.

If a form of secure erase is important to you, you must specifically buy an SSD which was verified to implement ATA Secure Erase properly, or come with a manufacturer toolkit erase function which has been tested for remnant data.

For step-by-step instructions on running Enhanced Secure Erase (which you should prefer over standard Secure Erase when possible), see:
https://ata.wiki.kernel.org/index.php/ATA_Secure_Erase
http://tinyapps.org/docs/wipe_drives_hdparm.html

My only addition to those instructions is that I have been able to successfully do Enhanced SE over USB. This was using external drive enclosures which were all USB 3, but mixed with USB 2 and 3 motherboard ports. This was using hdparm version 9.39 in an old Parted Magic image and v9.43 in Ubuntu 14.04. Oh, and the only way to guarantee non-recovery of data is to physically destroy the drive.

hdparm manual
http://linux.die.net/man/8/hdparm

A more comfortable and possibly safer alternative to hdparm from the terminal would be a Parted Magic live CD/USB (which just automates hdparm for you). Parted Magic isn't free anymore though.
https://partedmagic.com

Persistent malware in hard drive firmware
In February 2015, Kaspersky Lab published a report on Equation Group and some of the malware payloads they use to infect target systems. One was a permanent rootkit which allows the modification of HDD and SSD firmware and persists through subsequent firmware re-flashes.

Though it's in the form of a Windows .dll, we should not be foolish enough to conclude that the threat ends there. HDD firmware hacking has been done by talented individuals, aside from non-state actors, but the integrity of storage firmwares has traditionally received little to no attention from manufacturers.

This is important because while ATA SE is currently as good as it gets for non-destructive (of the drive) data erasure, it's not effective in a situation of firmware infection and it's unclear whether it would be effective in removing malware located in 'unseen' or supposedly empty areas of the drive's storage space.

Zeroing a Hard Drive

If you can not use ATA Secure Erase on a drive, filling it with zeros is the next best thing. This command bypasses write caching and zeroes the drive at its normal write speed. This can still take a long time for large drives.
```
dd if=/dev/zero of=/dev/sdx iflag=nocache oflag=direct bs=4096
```

When the operation finishes, verify it. This outputs a hex dump of the device.
```
sudo xxd -a /dev/sdx
```

If properly zeroed, you'll see similar to the screenshot.
http://epyxforensics.com/sites/default/files/styles/large/public/6_1.png

Background Fill

If you intend to encrypt your partitions, filling a drive with ciphertext makes your encrypted areas indistinguishable from free space. You can choose OpenSSL to fill with AES encrypted nonsense or urandom for pseudo-random nonsense. For more info, see:
http://thesimplecomputer.info/encrypt-your-linux-home-folder-2-ways-and-10-steps#noisefill

OpenSSL
Uses AES in counter mode, seeded with urandom. This writes at your normal drive speed and must be run as the root users.

```
openssl enc -aes-256-ctr -pass pass:"$(dd if=/dev/random bs=128 count=1 
2>/dev/null | base64)" -nosalt < /dev/zero | pv -pterb > /dev/sdx
```

Note: The command above requires package *pv* installed to monitor the fill progress. If you don't want the readout for pv, remove the pipe, pv caller and pv option shown in orange.

urandom
This writes at about 10-20 MB/s.

```
sudo dd if=/dev/urandom of=/dev/sdx bs=4096
```

To check progress of the fill for urandom, enter into a separate terminal:

```
sudo killall -USR1 dd
```

Encryption

Block Device Encryption

Cryptsetup, dm_crypt & LUKS
The standard and alternate Debian and Ubuntu installers can use dm_crypt and LUKS for full disk encryption or individually encrypted partitions. You'll get the choice between AES, Twofish and Serpent ciphers in XTS-plain64 mode with choice of a 128 or 256 bit key created with HMAC-SHA1 using 1 second of PBKDF2.

For more info and a walkthrough of completely manual full disk encryption in Ubuntu so you can further customize your encrypted partitions, see:
http://thesimplecomputer.info/full-disk-encryption-with-ubuntu

If you already have an installed system and want to add encrypted /home and swap partitions, see:
http://thesimplecomputer.info/encrypt-your-linux-home-folder-2-ways-and-10-steps

Back up your LUKS header
If either a key slot or your LUKS header get damaged and you do not have a backup header copy to restore, all encrypted data on that drive or partition will be irrecoverable. Directions to do this are in either of the above two links. Don't skip it!

TrueCrypt
Version 7.1a was the last before the TC disappearance and The Open Crypto Audit Project has finished auditing the code of 7.1a. They found only relatively minor concerns.
http://blog.cryptographyengineering.com/2015/04/truecrypt-report.html

If you want 7.1a, source, binaries and hashes are provided on the project's GIT page.
https://opencryptoaudit.org/

Current TrueCrypt forks (listed alphabetically).
CipherShed - https://ciphershed.org/
Tcnext - https://truecrypt.ch/
VeraCrypt - https://veracrypt.codeplex.com/

File Encryption

Ubuntu's desktop installer can use eCryptfs to encrypt files in /home with AES XTS-plain, a 128 bit key size and SHA256 password hashing. File names are not encrypted.

eCryptFS
Files or folders can be encrypted using eCryptfs (alternative to EncFS, OpenSSL or 7zip, not dm_crypt & LUKS)

eCryptfs is preferable over EncFS because it does not require FUSE and the encryption takes place in the kernel, not userspace, so it's faster. The automated "*ecryptfs-setup-private*" command uses AES-128 CBC and SHA256 but leaves file names in plaintext. If you want differently, you must manually configure eCryptfs. For more info, see:
http://ecryptfs.org/

OpenSSL
Quickly encrypt a single file or folder with a password using OpenSSL (alternative to eCryptfs, EncFS or 7zip).
To encrypt:
```
openssl aes-256-xts -salt -in filename -out filename.enc
```

To decrypt:
```
openssl aes-256-xts -d -in filename.enc -out filename
```

Tomb
Tomb can be used similarly, or to create encrypted containers within a filesystem like with TrueCrypt. Tomb provides stronger encryption and containers would provide overall better data security than encrypting files individually.
https://www.dyne.org/software/tomb/

Set up a cloud storage service to use an encrypted folder. The directions are for EncFS

but the concept is the same for eCryptfs, Tomb and any Truecrypt fork.
http://www.webupd8.org/2011/06/encrypt-your-private-dropbox-data-with.html

Wiping Free Space

See above for wiping an entire hard drive using ATA Secure Erase or zeroing it with dd. To wipe free space in a filesystem, use dd or OpenSSL.

Wipe with AES ciphertext (fast, must be run as root user).

```
openssl enc -aes-256-ctr -pass pass:"$(dd if=/dev/urandom bs=128 count=1
2>/dev/null | base64)" -nosalt < /dev/zero > /wipe_file
```

Disable disk caching and wipe with zeros (fast).

```
sudo dd if=/dev/zero of=/wipe_file iflag=nocache oflag=direct bs=4096
```

Wipe with pseudo-random data (very slow).

```
sudo dd if=/dev/urandom of=/wipe_file bs=4096
```

All 3 commands create the wipe_file in /. After the operation finishes, delete it.

```
sudo rm /wipe_file
```

Installation and Backup

Installation Size

Though not always practical, it is ideal to install a system with only the packages you need. A full desktop installation incurs much bloat resulting in a significantly increased attack surface. Reducing your installation size can be done in two ways: either by stripping out what you don't want from a full desktop installation, or by starting small and building the system up into what you want.

The removal method will still leave remnants, can disrupt dependencies and generally isn't as clean, but is much easier to do provided you know what you're removing. Starting small is best, but not for beginners.

Minimal Debian Installation
With any installer, you'll come to a point near the end of the process where it says that the core system is installed. You can then select other things to add like a standard desktop environment and laptop, system and server utilities.

To keep the core installation and nothing more, deselect everything at that point, then continue with the installation. You'll boot to tty1 of the new system and from there, you can install whatever you want from the command prompt.

Minimal Ubuntu Installation

Must use either the mini.iso which needs network access and has no UEFI support, or a server ISO (no network needed). For either, you'll still need to make sure that none of the additional utilities are selected like you would with Debian.

Note: For either distro, you'll want to disable install-recommends before installing anything if you really want a minimal installation base.

Ubuntu Restricted Extras
Not selecting this will slim down your install size if you don't want the codecs and Adobe Flash. Use this link to see the individual packages this option adds.
`http://packages.ubuntu.com/search?keywords=ubuntu-restricted-addons`

Installation image and repo sources:
`https://wiki.debian.org/SourcesList`
`https://www.debian.org/distrib/netinst`
`https://help.ubuntu.com/community/Installation/MinimalCD`

Network Connection

Whenever possible, it's ideal to install any operating system when disconnected from a network. You would connect and update only after booted into the fresh system and, again, ideally, only after a clean base backup image is made onto external storage.

Debian
This requires post-installation setup for the network connection and package sources.

1. Network configuration automatically happens early in the installation process but with no connection, it will fail. You must then select either “Configure network manually” or “Do not configure network at this time”.

2. For the package mirror steps, select any mirror because it will not actually apply. When the installer tries to configure apt, it will fail. From the red “Bad archive mirror” prompt, select “Go Back”, and choose “Yes” to continue without a network mirror.

3. Continue with the installation and when it's finished, boot into the new system. To configure apt, you'll need to edit `/etc/apt/sources.list` with the repositories for your region. Then you can install anything else you want. For what you should add to your sources.list file, see:
`http://debgen.simplylinux.ch/`

Note: If you want a full networked Debian desktop installation, use the small installer (the netinst.iso image) instead of the CD or DVD install images. The mini installer is a smaller download size but gives you the same base system plus anything additional you select (SSH or web server, desktop environment, etc.).
`https://www.debian.org/distrib/netinst`

Ubuntu
A non-networked installation is much easier with Ubuntu. The installer will warn about no internet connection but it will not affect the installation. You'll also get the full spectrum of

packages from the start and repos will be set to your geographic mirror. You'll need a full desktop or server installation image for this, not the mini installer.

Partitioning

Disk partition tables

GPT is preferable over MBR though it has incompatibilities with older Windows versions.

If your system:

- Is either Linux-only or multi-boots Linux and Apple OS X, then GPT should be used.
- Multi-boots with Windows 8 or newer, GPT must be used.
- Multi-boots a Windows version prior to 8, MBR must be used.

For external storage, GPT should be used unless it's to be shared with 32-bit Windows XP, then you must use MBR.

Assigning a partition table type is done in a live session before the system is installed using Parted, GParted or gdisk. **It erases the current partition table and ALL data on the drive.** There are ways to convert MBR to GPT but this is not worth the effort or risk of data loss.

There are two situations where the Debian or Ubuntu install will automatically change a drive's partition table on installation:

1. When selecting the "*use whole disk and encrypted LVM*" option, which will use the drive's master boot record (MBR area).
2. When installing to a computer set to use UEFI mode in the actual UEFI instead of legacy BIOS mode, in which case GPT is used.

To check what partition table type your drives use from within an installed system:

```
sudo parted --list
```

(This is non-destructive.)

For more info, see:
http://www.linux.com/learn/tutorials/730440-using-the-new-guid-partition-table-in-linux-good-bye-ancient-mbr
http://msdn.microsoft.com/en-us/library/windows/hardware/dn640535%28v=vs.85%29.aspx
https://en.wikipedia.org/wiki/GUID_Partition_Table#Operating_systems_support

Partition layout

Generally speaking, it's best to create system and user areas (`/` and `/home`) on separate partitions. This gives better control over each partition with backup images and fstab mount options, and better disk performance. If you've already installed, you can still move `/home` to a separate partition.
https://help.ubuntu.com/community/Partitioning/Home/Moving

Create swap file or partition

https://help.ubuntu.com/community/SwapFaq

Unless you want to hibernate the computer, there's no general need for a swap partition. Since kernel 2.6, swap files are reportedly “just as fast” as a swap partitions and easier to manage.
https://help.ubuntu.com/community/SwapFaq#Should_I_reinstall_with_more_swap.3F

System Backups

Clonezilla and dd

Clonezilla by CD/DVD or USB is a superb choice for creating drive or partition backup images. It compresses unencrypted data so images take up less space.
http://clonezilla.org/

Alternatively, you can clone a partition or drive with dd. This does not compress data so a dd image uses the full amount of space as does the data being imaged. Encrypted data can't be compressed so for that, Clonezilla uses dd.

To clone a drive or partition with dd:
```
sudo dd if=/dev/sdxy of=/media/path/to/your_image.img bs=4096
```

To restore a dd image:
```
sudo dd if=/media/path/to/your_image.img of=/dev/sdxy bs=4096
```

Unneeded system files

The contents of these folders can be deleted before a disk image is made. They will be automatically recreated in the new system.
```
/tmp
/var/tmp
```

Back up your user's home filesystem

This creates a tarball archive of a user's home directory. It's sloppy but includes absolutely everything in `~/username` and you can then pick through and discard what you don't want.
```
tar --cvpzf home_backup.tar.gz ~/
```

Passkeys

/dev/random & /dev/urandom

Both create strong crypto keys and many other programs are just a front end for them. Contrary to what the /dev/random manual page says, there's nothing wrong with urandom for your entropy needs.
http://www.2uo.de/myths-about-urandom/

The command below generates 3 passkeys, each 64 characters in length. The argument *tr* is the available characters you want to use in the passkey, *fold* is the key length and *head* is the amount of keys you generate. Head needs a minimum value of 1.

```
cat /dev/urandom| tr -dc 'a-zA-Z0-9-_!@#$%^&amp;*()_+{}|:&lt;&gt;?=' |fold -w
64| head -n 3| grep -i '[!@#$%^&amp;*()_+{}|:&lt;&gt;?=]'
```

As an alternative, the package *makepasswd* in the repositories is a GTK+ frontend for urandom.

OpenSSL

Easier to remember the command but base64 does not use all keyboard symbols so an OpenSSL key won't be as strong as a urandom key of the same length. However, not all web servers, services, devices, etc. allow passwords with special characters so base64 may be all you can do.

Create a 32 character long key.

```
openssl rand -base64 32
```

Self-signed SSL/TLS certificates

Create a 4096-bit private RSA key.

```
openssl genrsa -out ~/mykeyfile.key 4096
```

Use your private key to create a certificate signing request.

```
openssl req -new -key ~/mykeyfile.key -out ~/myreq.csr
```

You'll be prompted for identification info for the certificate owner. Leave it blank or fill it in depending on your use. You do not need a challenge password for a self-signed certificate. That is for if you're using a commercial CA like StartSSL or Thawte. Press ENTER to skip through it.

Create a x509 specification certificate valid for 365 days.

```
openssl x509 -req -days 365 -in ~/myrequest.csr -signkey ~/mykeyfile.key -out
~/mycertificate.crt
```

If you don't need the actual certificate signing request (.csr file), you can just create a private key and the certificate with one command.

```
openssl req -x509 -nodes -days 365 -newkey rsa:2048  -keyout ~/mykeyfile.key
-out ~/mycertificate.crt
```

For more info, see OpenSSL's certificate HOWTO:
https://www.openssl.org/docs/HOWTO/certificates.txt

If you want a free SSL certificate for your website signed by a Certificate Authority (as opposed to self-signed), here is a guide to using StartSSL.
https://konklone.com/post/switch-to-https-now-for-free

Package Management

Unnecessary or unwanted packages increase attack surface and waste space. They should be removed or not installed from the start. If unneeded, consider removing remote desktop, telnet, ssh, samba, cups, avahi, zeitgeist and other services, drivers, fonts and compiling tools.

Recommended Packages

Debian and Ubuntu have several degrees of package dependencies which you can read about here. Recommended packages are considered a sort of weak or non-absolute dependency and both distros install them by default.

More often than not, recommended packages aren't needed and not installing them can significantly reduce the amount of packages you pull in with software you do want. That's less bloat and a smaller attack surface. Installing a specific package without recommended dependencies can be done with:

```
sudo apt-get install package_name --no-install-recommends
```

You can also disable automatically installing recommended packages either through Synaptic or by creating a text file for apt, or in the Options menu if you use aptitude.

Synaptic Package Manager
This creates the 99synaptic file below for apt. Synaptic comes with Debian and Ubuntu 12.04 but not 14.04 or 16.04.

```
Settings > Preferences
```

Uncheck:

```
Consider recommended packages as dependencies
```

Then let it reload the repositories.

apt config file
If you don't have Synaptic installed, you'll need to create this file.

```
pkexec gedit /etc/apt/apt.conf.d/99synaptic
```

Edit/add:

```
APT::Install-Recommends "false";
```

Then reload the repos.

```
sudo apt-get update
```

aptitude
Run aptitude in the terminal as root.

```
sudo aptitude
```

Press Ctrl+T to access the menu bar.
`Options > Preferences`
Uncheck:
`Install recommended packages automatically.`

Then reload the repos.

If you want to know what are recommended and suggested for a specific package you have in mind, see:
https://www.debian.org/distrib/packages
http://packages.ubuntu.com/

Repository Mirrors

For Debian, you can change repo mirrors by directly editing `/etc/apt/sources.list` with the new deb url. In Ubuntu, you can change repo mirrors through Software Updater (package name: *update-manager*).

Debian mirrors
http://mirror.debian.org/status.html

Ubuntu mirrors and update intervals
https://launchpad.net/ubuntu/+archivemirrors

Note: You can disable the repositories for source code (the deb-src lines in sources.list) unless you actually need to download the source from the package maintainer. This saves time when updating repo lists.

Note: Changing repo mirrors currently prevents updating versions of Mint (ex. 17 to 17.1). You'll need to open the Update Manager and go to `Official repositories > Restore the default settings`.

Custom sources files
To easily create your own sources.list, see these generator sites.

Debian
http://debgen.simplylinux.ch/

Ubuntu
http://repogen.simplylinux.ch/

Automatic Security Updates

Daily this reloads the repositories, checks for security updates and installs them. Multiple ways to do this depending on your distro.

1. Install and set *unattended-upgrades* manually (Debian and Ubuntu)

2. Software Updater on Ubuntu
3. Synaptic (will need to install it on 14.04 and 16.04)

https://help.ubuntu.com/community/AutomaticSecurityUpdates

Miscellaneous

Back up and restore installed packages from a master list
Create a dump file with all your installed packages.
```
sudo dpkg --get-selections > ~/Desktop/mypackages
```

To restore those packages to a system:
```
sudo dpkg --clear-selections
sudo dpkg --set-selections < ~/Desktop/mypackages
sudo apt-get update
sudo apt-get -u dselect-upgrade
```

Note: This doesn't restore the distinction between automatically and manually installed packages. It also won't remove packages that are installed but not on your restore list.

Ban unwanted packages from the system
Will need to create the file.
```
pkexec gedit /etc/apt/preferences
```
Add:
```
Package: packagename
Pin: version 0
Pin-Priority: -1
```

Tell Synaptic to not mark the new packages from external repos (like Chrome) as auto-removable.
Open Synaptic, select the target packages and:
```
Package > select: Automatically Installed
```

Force removal of a package without taking all its dependencies with it.
```
sudo dpkg -P packagename
```

Disable popularity contest cron job
It's not in all Debian or Ubuntu-based systems so first confirm you have it.
```
ls /etc/cron.daily
```

Comment out the script.
```
sudo mv /etc/cron.daily/popularity-contest /etc/cron.daily/#popularity-
contest
```

Or just delete it.
```
sudo rm /etc/cron.daily/popularity-contest
```

Disabling and Blacklisting Services

Unhide Startup Apps

This will list in the `Startup Applications` menu all applications in `/etc/xdg/autostart`. Out of the box, Cinnamon, Gnome and Unity only allow a few services to be switched off and the rest are hidden. To unhide the rest:

```
sudo sed -i "s/NoDisplay=true/NoDisplay=false/g" /etc/xdg/autostart/*.desktop
```

Then go to `Startup Applications` from the Dash, Overview or panel menu and disable what you don't want. If you're unclear about something, do an internet search to figure out what it's for.

Note: This doesn't really work for Gnome 3.14 (Debian 8) because there's no Startup Application Preferences app anymore. I don't know if the Startup Applications section in Gnome Tweak Took is supposed to take over the old method, but it doesn't (yet?).

systemd

systemd is the successor of the System V init system. Currently systemd is fully implemented in Debian 8 and Ubuntu 15.04 with backward-compatibility for sysv services.

A “unit” is anything that can be controlled by systemd: a system service, PCI device, mount point, etc. To see all units systemd's has loaded and tried to load:

```
systemctl list-units
```

To see all units on the system that systemd has control over:

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

A “masked” systemd unit is totally off limits for systemd to start at any time; it's basically blacklisted. To blacklist a unit or unmask it:

```
sudo systemctl mask service_name.service
sudo systemctl unmask service_name.service
```

For more info on systemd units:
https://www.digitalocean.com/community/tutorials/understanding-systemd-units-and-unit-files

A “target” is the replacement of what we'd previously call a runlevel. Where all services under SysVinit or Upstart could only run under one runlevel at a time, systemd units can occupy different targets independently of each other. If you're in a system with a GUI, the default target is *graphical*. For a headless system with no graphical interface, it will be *multi-user*.

To see the system's default target:

```
systemctl get-default
```

Status of services

To see all loaded services on the system and their state:
`systemctl list-units -t service`

To view a systemd service's status:
`systemctl status service_name.service`

To start or stop a service during runtime:
```
sudo systemctl start service_name.service
sudo systemctl stop service_name.service
sudo systemctl reload-or-restart service_name.service
```

To disable or enable a systemd service on boot:
```
sudo systemctl disable service_name.service
sudo systemctl enable service_name.service
```

For (a lot) more info, see the systemd wiki and Arch's systemd wiki entry.
https://wiki.freedesktop.org/www/Software/systemd/
https://wiki.archlinux.org/index.php/Systemd/Services

Upstart Jobs

Though systemd has won the war, Upstart is still relevant in 12.04 and 14.04. When an Upstart job is set to manual with an override file, the service will still be accessible on demand but not automatically started.

A quick view of `/etc/init` will then show which jobs are disabled. This is easier than inspecting an individual job file's contents and is the main reason to use override files instead of editing the job file directly or deleting it.

To view all Upstart jobs:
`ls /etc/init`

To view a specific Upstart service's run state:
`sudo status service_name`

To deactivate a service, first stop it.
`sudo service service_name stop`

Then create an override file
`echo manual | sudo tee /etc/init/service_name.override`

Upstart init manual.
http://manpages.ubuntu.com/manpages/trusty/en/man5/init.5.html

System V Scripts

User sessions for Debian 7 and Ubuntu 12.04 and 14.04 are runlevel 2. (enter `runlevel` in a terminal). Go to `/etc/rcx.d`, (x being whichever runlevel you are) and find the services you want to disable.

To disable across all runlevels:
```
sudo update-rc.d service_name disable
```

To disable only for individual runlevels, change the S in the script's file name to a K. Example:
```
sudo mv /etc/rc2.d/S17speech-dispatcher /etc/rc2.d/K17speech-dispatcher
```

Note: Debian 8 still has SysV scripts but you should use systemd to control the processes/units. For Ubuntu 12.04 and 14.04, there are still some services you'll need to do this for which don't have Upstart jobs. When Ubuntu switches over to systemd, manipulating SysV scripts will again be unnecessary.

Blacklisting Modules

Use `lsmod` and `lspci` to find the module you want to stop from loading.

For Ubuntu and Debian 7, then add the module name to the blacklist file.
```
pkexec gedit /etc/modprobe.d/blacklist.conf
```

Add: (for example)
```
# blacklist webcam
blackist uvcvideo
```

For Debian 8, you must create a separate blacklist file for each module.
```
pkexec gedit /etc/modprobe.d/module_name-blacklist.conf
```

Add: (for example)
```
# blacklist webcam
blackist uvcvideo
```

Miscellaneous

Remove Unity shopping lenses and scopes
https://fixubuntu.com/

Permissions

Running Graphical Applications as Root

Some points:
- sudo (package *sudo*) is for non-GUI programs
- gksu (package *gksu*) is the graphical 'version' of sudo and is intended for GUI programs. Debian 7 installs with it, Debian 8 and Ubuntu do not.
- gksu is unmaintained and slowly being replaced by pkexec.
- pkexec executes an application as a different user. It's part of the polkit framework

and still transitional so needs tinkering to get working with GUI applications.
- Running gedit with sudo in Ubuntu transfers ownership of `~/.config/dconf/user` from you to root. Then, if you reboot or log out before the 15 minute sudo timeout, you can find your desktop totally default vanilla and you must reclaim the dconf file for your user.
- It's a good idea to configure your most commonly used root access GUI programs to use polkit actions.

Create a polkit action file for (as an example) gedit to use pkexec.
Can also do this for Nautilus, Thunar, Leafpad...just about anything.
Create a file:
```
sudo nano /usr/share/polkit-1/actions/org.freedesktop.pkexec.gedit.policy
```

Add for contents:
```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE policyconfig PUBLIC
 "-//freedesktop//DTD PolicyKit Policy Configuration 1.0//EN"
 "http://www.freedesktop.org/standards/PolicyKit/1/policyconfig.dtd">
<policyconfig>

  <action id="org.freedesktop.pkexec.gedit">
    <message>Authentication is required to run Gedit as root</message>
    <icon_name>gedit</icon_name>
    <defaults>
      <allow_any>no</allow_any>
      <allow_inactive>no</allow_inactive>
      <allow_active>auth_admin</allow_active>
    </defaults>
    <annotate
key="org.freedesktop.policykit.exec.path">/usr/bin/gedit</annotate>
    <annotate key="org.freedesktop.policykit.exec.allow_gui">true</annotate>
  </action>

</policyconfig>
```

To save the file and exit nano, press **Ctrl+X**, then **Y,** then **Enter**. From then on, a root instance of gedit would be launched with:
```
pkexec gedit
```

For more info, see:
http://www.freedesktop.org/software/polkit/docs/latest/pkexec.1.html
https://wiki.archlinux.org/index.php/Polkit

Login

Disable LightDM guest account.
```
pkexec gedit /etc/lightdm/lightdm.conf
```
Edit or add:
```
allow-guest=false
```

User password settings

Descriptions for what these do are in the file. Values are suggestive.
```
pkexec gedit /etc/login.defs
```
Edit:
```
LOGIN_RETRIES 3
LOGIN_TIMEOUT 15
```

Uncomment and set:
```
SHA_CRYPT_MIN_ROUNDS 15000
```

Disable system-wide root login on any terminal.
Doesn't affect single user (recovery) mode or SSH/SFTP. Should first back up the original securetty file.
```
sudo cp /etc/securetty /etc/securetty.bak
```

Then create a new securetty file, leave it blank.
```
echo | sudo tee /etc/securetty
```

Assign root a login password.
Debian asks to configure a root account by default though you can leave the inputs empty for no root acct. In Ubuntu you must make one yourself. If you have encrypted volumes, this does not take the place of a decryption key.

A root account prevents booting into recovery (single-user) mode without first entering the root pw, and disallows resetting user passwords without knowing root's pw. But a root account could decrease security if it has a weak password (but most people's user account passwords are weak too, so net loss is..meh).

To create a root account and assign it a password:
```
sudo passwd root
```

File Access & Histories

umask
umask sets the default permissions for every new file or folder created by every user, including root. 022 (the default) means that the owner can read, write and execute. Any member of the owner's group and any other user can read and execute those files or folders but they have no write access to them. The equivalent chmod would be 755 for folders and 644 for files

027 would mean that owner still can rwx, owner's group can r and x only, and other users can't even access those files or folders. The equivalent chmod would be 750 for folders and 640 for files.

077 would grant the least access. Owner can rwx but owner's group and users have no access at all. The equivalent chmod would be 700 for folders and 600 for files.

In `/etc/login.defs`, Debian and Ubuntu set *USERGROUPS_ENAB* to *yes*. In Ubuntu with a default system-wide umask of 022, an individual user's umask will then be 0002, unless set otherwise by that user on each login or by `~/.profile`.

However, Debian doesn't ship with PAM's umask module loaded by default, so *USERGROUPS_ENAB=yes* doesn't take effect. Debian's setting grants less privileges for group access than Ubuntu, but for that privilege to be granted, other system users must first be part of your user's group.

You can see your user's umask by entering `umask` into a terminal.

Change system-wide default umask
```
pkexec gedit /etc/login.defs
```
Edit:
```
UMASK       077
```

If you're doing this on Debian, you'll also need to tell PAM to load its umask library.
```
pkexec gedit /etc/pam.d/common-session
```
Add:
```
session optional pam_umask.so
```

If you're on Ubuntu and want to limit group access as Debian does, comment out the pam_umask line in `common-session.` So:
```
#session optional pam_umask.so
```

Save the file. Log out and back in.

For more on umask, see:
http://www.cyberciti.biz/tips/understanding-linux-unix-umask-value-usage.html
http://ubuntuforums.org/showthread.php?t=905566&s=a6ca3b2d978b44791ac72569560ff728&p=10060230#post10060230

Home directories
Debian and Ubuntu set the permissions (chmod) of contents of `/home/user_name/*` to 755 for folders and 644 for files. To check the a specific file or folder:
```
stat -c %a file_or_folder_name
```

dmesg
dmesg logs contain sensitive system info. Only allow root to access them.
```
sudo sysctl kernel.dmesg_restrict=1
```

Then make it permanent.
```
pkexec gedit /etc/sysctl.conf
```
Add:
```
#Prevent unprivileged users from accessing dmesg logs
kernel.dmesg_restrict = 1
```

Zeitgeist
For an explanation of what Zeitgeist is, see:
http://zeitgeist-project.com/about/

Zeitgeist consists of several packages and is included to various degrees in Debian and Ubuntu. Docky, Gnome Activity Journal, KDE (along with Nepomuk and KDE activity manager), Synapse and Unity's Dash are the most prominent uses of Zeitgeist though there are plans to integrate it further into Gnome apps.

To see what zeitgeist related packages installed on your system:
```
dpkg -l | grep zeitgeist
```

If nothing on your system has any dependency on Zeitgeist, it's not serving any purpose, and you can remove its main packages: *zeitgest* (a metapackage), *zeitgeist-core* and *zeitgeist-datahub.* This will stop its data logging and eliminate its overhead.

Then remove the databases. This command affects system-wide, so leave the wildcard.
```
sudo rm -r /home/*/.local/share/zeitgeist
```

Disable recent file history for GTK 3 programs (Evince, Nautilus, etc.).
This method does not result in file access errors in the terminal. Use this instead of messing with `~/.local/share/recently-used.xbel`. You'll need to create this file for each user.
```
echo -e "[Settings]
ngtk-recent-files-max-age=0
ngtk-recent-files-limit=0" > ~/.config/gtk-3.0/settings.ini
```

Then delete the current history files (all users).
```
sudo rm /home/*/.local/share/recently-used.xbel
```

Shell command history
This is the history of commands you have run in the terminal. You can see entries by opening a terminal and entering `history` or pressing the up/down arrow keys. This is per-user, not system-wide.

Restrict your history to x amount of lines.
```
gedit ~/.bashrc
```
Edit:
```
HISTSIZE=x
HISTFILESIZE=x
```

Alternatively, save history only for current session.
```
gedit ~/.bashrc
```
Edit:
```
HISTFILESIZE=0
```

To immediately clear entire history, close all terminals but one and enter:
```
history -c -w
```

More shell history commands.
https://chrisjean.com/2009/03/09/command-line-history-in-ubuntu-terminal/

Run Programs in Their Own User Account

It is desirable to restrict some applications to their own standard user account with nearly no system access or privileges beyond there, letting them changeroot where necessary. This is a considerable security improvement but is more conveniently suited to processes which don't require display server access.

For walkthroughs on how to do this, see these links:
Example: Firefox (scroll down to *Hardening Firefox*)
http://blog.internot.info/2014/06/securing-ubuntu-desktop-from-bad-guys.html

Example: Pidgin
http://www.insanitybit.com/2012/08/01/creating-a-new-user-account-for-pidgin/

Filesystem

fstab, tmpfs and TRIM

For better security and disk performance, mount options can be fine-tuned and directories mounted into volatile RAM as temporary filesystems. TRIM for solid state drives is also an important topic but is tumultuous in implementation.

For a detailed walkthrough of fstab, TRIM, tmpfs and other and mount options, see:
http://thesimplecomputer.info/oh-devsda-how-have-i-displeased-thee

Disk I/O

Schedulers
In a comparison by Phoronix on kernel 3.4 (Ubuntu 12.04), CFQ gave the best general performance on platter drives and low-end solid state drives. My own dd writes on kernel 3.8 in Ubuntu 13.04 showed CFQ was slightly faster than Deadline for small writes (10 KB), Deadline was very slightly faster for larger writes (1 MB) and Noop should not be used on platter drives. In Phoronix's tests, Deadline was preferable for higher-end SSDs.

CFQ is the default in Debian and Ubuntu 12.04 while 14.04 uses Deadline. It's okay to mix schedulers but there must be one entry for each drive.
`pkexec gedit /etc/rc.local`
Add:
```
-------
#By default this script does nothing.

echo deadline > /sys/block/sdx/queue/scheduler
echo cfq > /sys/block/sdy/queue/scheduler

exit 0
-------
```

Reboot and verify. The enabled scheduler will be in [brackets].
`cat /sys/block/sdx/queue/scheduler`

Swap and caching

These settings are to make more use of available system memory, but be careful with these if you don't have much RAM. Values are suggestive. Kernel-set defaults (from top to bottom) are 60, 100, 5% and 10%.

```
pkexec gedit /etc/sysctl.conf
```
Add:
```
#Keep data longer in RAM rather
#than swapping out
vm.swappiness = 10
#
#Kernel prefers keeping filesystem
#and inode caches longer in RAM
vm.vfs_cache_pressure = 20
#
#Increase free RAM available to kernel as write
#cache before writing out cached data to disk:
#without interrupting processes
vm.dirty_background_ratio = 15
#
#Increase ceiling for when processes are
#interrupted because kernel write cache
#must be dumped to disk
vm.dirty_ratio = 30
```

Then enable the changes.
```
sudo sysctl -p
```

For more info on kernel sysctl parameters:
https://www.kernel.org/doc/Documentation/sysctl/vm.txt

Increase ureadahead buffer size
Must have ureadahead installed. Debian 7 and Ubuntu do by default. Don't use this with Debian 8 because systemd has its own readahead services.
```
sudo apt-get install ureadahead
```

To view the current readahead buffer size (it's the RA column):
```
sudo blockdev --report /dev/sdx
```

To raise the readhead buffer size (must be in multiples of 512 byte sector sizes):
```
sudo blockdev --setra 4096 /dev/sdx
```
(value is suggestive)

To make it persistent across reboots, add to rc.local:
```
pkexec gedit /etc/rc.local
```
Add:
```
blockdev --setra 4096 /dev/sdx
```

For more info on ureadahead.
http://manpages.ubuntu.com/manpages/trusty/man8/ureadahead.8.html

Gnome Virtual Filesystem

Package *gvfs* is a dependency of Nautilus and Nemo and adds Trash capability to basic file managers like Thunar. One of GVfs's jobs is to log metadata for any storage device mounted over any interface (including network) in the operating system's lifetime. It may be a valuable or unwanted forensics point and a way of accruing clutter.

To minimize Gvfs data collection for all users, you can make a script to purge the folder contents on shutdown. For systems which don't use systemd:

```
pkexec gedit /etc/rc6.d/K99gvfs-clear
```

Add:

```
#!/bin/sh
#Remove gvfs-metadata for all users.
rm /home/*/.local/share/gvfs-metadata/*
fi
```

Then make it executable.

```
sudo chmod +x /etc/rc6.d/K99gvfs-clear
```

For systemd, you need a unit file to run the script. The script can be in any location but for the unit file:

```
pkexec gedit /etc/systemd/system/gvfs-clear.service
```

with the contents:

```
[Unit]
Description=Remove gvfs metadata for all users.
Before=reboot.target shutdown.target halt.target

[Service]
ExecStart=path_to_script

[Install]
WantedBy=multi-user.target
```

Enable the unit.

```
sudo systemctl enable gvfs-clear.service
```

Note: Any way of limiting or purging this data will be a rough hack. File managers and other programs need volume device metadata for mount points like `/` and `/home` so you can't simply remove all modules' execute permission.

Extra gvfs modules

Package *gvfs-backends* is included by default in Debian and Ubuntu. It daemonizes three processes:

1. mtp volume monitor to mount Android devices
2. afc volume monitor to mount iPhones & iPads
3. gphoto2 volume monitor to import photos from cameras.

For these, you can remove the binary's execute permission if you have no use for them. This will be reset if the *gvfs-backends* package is updated.

```
sudo chmod -x /usr/lib/gvfs/gvfs-afc-volume-monitor
sudo chmod -x /usr/lib/gvfs/gvfs-gphoto2-volume-monitor
sudo chmod -x /usr/lib/gvfs/gvfs-mtp-volume-monitor
```

gvfsd and Apache
Noticed on Ubuntu 13.04 with Apache 2.2 running for WordPress. The module *gvfsd-http* makes outbound connections through port 443 to various websites, even when Apache is not exposed to the internet.

Sometimes the IP addresses were sites visited in a browser, sometimes were seemingly unrelated. Removing the module's execute permission stopped the connections.

```
chmod -x /usr/lib/gvfs/gvfsd-http
```

Miscellaneous

Disable unnecessary system logging
This disables kernel logs (not syslog), and mailer logs which you probably don't have installed anyway so won't be populating.

```
pkexec gedit /etc/rsyslog.d/50-default.conf
```

Comment out:

```
#kern.*
#mail.*
```

Then remove the old logs.

```
sudo rm /var/log/kern.*
```

For Debian 8, the file location is:

```
pkexec gedit /etc/rsyslog.d/rsyslog.conf
```

rhosts
Debian and Ubuntu have no .rhosts files by default, but to check if there are ever are any on the computer:

```
sudo find / -name .rhosts -print
```

Network

Firewall & Packet Filtering

iptables controls netfilter, the firewall/packet filter built into the kernel. Depending on your needs, iptables can be difficult to configure so there have been several iptables controllers developed to make adding firewall rules more user-friendly.

The most common in Ubuntu based distros is Canonincal's ufw, included in Ubuntu but not Debian. ufw is NOT a firewall in and of itself—it's purpose is to allow easier control of iptables. More on ufw later.

By default, most linux distros have the firewall set (via iptables) to allow all incoming, forwarded and outgoing connections and if ufw is installed, it's not enabled. Some distros have ports open by default and having no filtering on those ports is bad security, even if you are behind a router or other firewall appliance.

Don't use both iptables and ufw together for firewall rules. It's unnecessary, inefficient, can cause conflicts and complicate troubleshooting. If you want to control the firewall with ufw, leave iptables alone. If you want to use only iptables, keep ufw disabled or remove it entirely.

If you have no internet facing services listening, then all external ports are closed and you technically don't need to bother with any firewall. **BUT**, if you ever install something which opens a port that you're unaware of, then that can be bad news, especially on mobile devices. Why even take the chance? Good security has many layers and using even basic firewall rules on networked devices is always safer than none at all.

To see everything you have listening and on which ports:
```
sudo netstat -tulpna
```

For list of which services traditionally use which ports:
```
cat /etc/services
```

Drop vs Reject
For both iptables and ufw, these links lay out the arguments on whether you should drop or reject unwanted inbound traffic.
http://www.chiark.greenend.org.uk/~peterb/network/drop-vs-reject
http://www.chrisbrenton.org/2009/07/why-firewall-reject-rules-are-better-than-firewall-drop-rules
http://seclists.org/firewall-wizards/2010/Jan/19
https://security.stackexchange.com/questions/43779/benefits-of-reject-over-drop-on-a-single-pc/43783#43783
https://serverfault.com/questions/157375/reject-vs-drop-when-using-iptables
https://unix.stackexchange.com/questions/109459/is-it-better-to-set-j-reject-or-j-drop-in-iptables
https://stackoverflow.com/questions/4907173/ufw-linux-firewall-difference-between-reject-and-deny

tl:dr summary
Are you a:
- desktop behind a router or other hardware firewall? Then you can reject.
- local server with no internet access? Reject.
- heavy torrent user? Reject.
- local server with internet access? Drop.
- mobile device? Drop.

iptables

It's preferable to control the firewall using iptables. This incurs the least overhead and complexity and results in the cleanest rulesets.

To view the current state of iptables' IPv4 rules:

```
sudo iptables -L
```

And for IPv6:

```
sudo ip6tables -L
```

Basic iptables ruleset

Most computers (including mobile) only need 4 firewall rules each for IP versions 4 and 6. The result is simple but effective: Drop or reject any unsolicited inbound traffic and all forwarding traffic. Allow all outgoing traffic, solicited inbound traffic and all on loopback interface (loopback traffic doesn't leave the computer, it's just for local applications to communicate over).

Step 1

Feel free to air-gap and turn off ufw if you have it enabled.

```
sudo ufw disable
```

Then totally strip the current iptables rules down to default.

```
sudo iptables -F && sudo iptables -X
```

Step 2

Now load your rules. If you want to drop incoming unsolicited, use these—**in this order**.

```
sudo iptables -P INPUT DROP
sudo iptables -P FORWARD DROP
sudo iptables -A INPUT -m conntrack --ctstate ESTABLISHED,RELATED -j ACCEPT
sudo iptables -A INPUT -i lo -j ACCEPT
sudo iptables -A INPUT ! -i lo -s 127.0.0.0/8 -j REJECT
```

~OR~

If you want to reject incoming unsolicited, use this instead—**in this order**.

```
sudo iptables -A INPUT -m conntrack --ctstate ESTABLISHED,RELATED -j ACCEPT
sudo iptables -A INPUT -i lo -j ACCEPT
sudo iptables -A INPUT ! -i lo -s 127.0.0.0/8 -j REJECT
sudo iptables -A INPUT -j REJECT
sudo iptables -A FORWARD -j REJECT
```

Step 3

For IPv6, substitute ip6tables for iptables in the commands.
Example:

```
sudo ip6tables -F

sudo iptables -P INPUT DROP #OR REJECT
sudo iptables -P FORWARD DROP #OR REJECT
sudo iptables -A INPUT -i lo -j ACCEPT
sudo iptables -A INPUT ! -i lo -s ::1/128 -j REJECT
sudo iptables -A INPUT -m state --state ESTABLISHED,RELATED -j ACCEPT
```

Step 4

And last, install the package *iptables-persistent* (Debian and Ubuntu only). This will load your rules on each reboot. When asked save your IPv4 and IPv6 rules, choose *YES*.

```
sudo apt-get install iptables-persistent
```

To see your new rules at work:
```
sudo iptables (or ip6tables) -L -v
```

To see your rules in the order in which they're being applied to traffic:
```
sudo iptables (or ip6tables) -S
```

If you want to take it further, read the manual.
http://linux.die.net/man/8/iptables

Here is a list of of advanced iptables rules.
http://www.cyberciti.biz/tips/linux-iptables-examples.html

ufw

By default, ufw is disabled on Ubuntu (not installed in Debian). You don't need ufw unless you're actually going to use it to sculpt firewall rules. Enabling ufw and leaving it with the default deny in and allow out policy gives you nearly the same effect as the iptables rules above. There are 4 exceptions:

1. Inbound ICMP (8) echo requests (pings) are allowed and responded to.
2. Dropped invalid packets are logged.
3. ICMP redirect accepts, SYN cookies and source route accepts are disabled with sysctl switches. These things can also be directly adjusted in `/etc/sysctl.conf`. More on sysctl further down.
4. ufw loads 3 additional netfilter kernel modules for connection tracking: two for FTP and one for NetBIOS. These connection tracking helpers aren't used for anything with ufw's default settings, even if you use FTP and Samba.

Enable ufw with default settings
```
sudo ufw enable
```

Disable extra Conntrack modules
If ufw is enabled and the modules unneeded, they can be disabled.
```
pkexec gedit /etc/default/ufw
```

Comment out:
```
#IPT_MODULES="nf_conntrack_ftp nf_nat_ftp nf_conntrack_netbios_ns"
```

ufw Community wiki
https://help.ubuntu.com/community/UFW

ufw manual for 14.04
http://manpages.ubuntu.com/manpages/trusty/en/man8/ufw.8.html

Static IP Addresses

If you have a stationary computer, give its network interface(s) a static IP (but obviously don't do this with mobile devices). The two ways to do this are either though

NetworkManager or the network interfaces configuration file. Using NetworkManager still lets you manage the interface through the desktop environment's UI (read: turn it on and off) so this is best for computers with a desktop environment. If you assign the ip just through the interfaces file, it's considered Unmanaged by NetworkManager and this is how it's done for servers.

Regardless of which you choose, after the IP is assigned in the computer, you should then go into your router and pair or reserve that same IP for the computer so there's no confusion anywhere.

Interfaces config file
If you prefer to use the interfaces file instead of NetworkManager's UI, then you must tell NM to not manage the NIC.

```
pkexec gedit /etc/NetworkManager/NetworkManager.conf
```
Edit:
```
[ifupdown]
managed=true
```

Take down the network interface so dhclient stops properly.
```
sudo ifconfig eth0 down
```

Then edit the interfaces config file.
```
pkexec gedit /etc/network/interfaces
```
Add:
```
# The primary network interface
auto eth0
iface eth0 inet static
address 123.456.78.9 #(or 123.456.78.9/24, then remove the netmask below)
gateway 123.456.78.0 #(ususally your router's ip)
netmask 255.255.255.0 #(rm this if you used /24 in the address line)
dns-nameservers provider1 provider2 etc.
dns-options rotate
```

Then bring the interface back up and restart the networking services, or reboot the computer to restart everything at once. No need to mess with resolv.conf either.

For more info, see:
http://www.sudo-juice.com/how-to-set-a-static-ip-in-ubuntu-the-proper-way/

DNS

For a list of alternative DNS providers:
http://thesimplecomputer.info/a-list-of-dns-service-providers

DNSCrypt
This application is in neither distro's repositories (yet?). It encrypts traffic between your computer and the DNS provider, but it must use a provider which offers DNSCrypt addresses. DNSCrypt.eu Resolver 1 is the default and the developer also keeps a GitHub page of additional resolvers.

`https://github.com/jedisct1/dnscrypt-proxy/blob/master/dnscrypt-resolvers.csv`

Install DNSCrypt
There are 3 ways to do this:

1. From source
   `http://dnscrypt.org/`
   `https://github.com/jedisct1/dnscrypt-proxy`
2. An unofficial build script by Simon Clausen, who also runs the DNSCrypt.eu resolvers. (Arch and DEB & RPM systems)
   `https://github.com/simonclausen/dnscrypt-autoinstall`
3. An unofficial Launchpad PPA by Pascal Mons (Ubuntu only)
   `https://launchpad.net/~anton+/+archive/ubuntu/dnscrypt`

After DNSCrypt is installed, you must then tell NetworkManager to talk to it (or `/etc/network/interfaces` if you're not using NM to manage connections).

`Network > Wired > Ipv4 Settings`
If you're not using a static IP address, change `Method` to `Automatic (DHCP) addresses only`. If you are static, just fill in the blank for DNS.

Add:
`DNS servers: 127.0.0.2`

Confirm DNSCrypt is working
To quickly check if DNSCrypt is working with the default dnscrypt.eu resolver:
```
sudo tcpdump -i eth0 dst host 176.56.237.171
```

Then go to a website you've either not been to in a long time, or never been to before. This is to avoid the lookup coming from the browser's DNS caching. If DNSCrypt is working, you should see lines in the terminal output for each UDP packet which will include "resolver1.dnscrypt.eu.https:". If it's not working, you won't see any output.

Note: DNSCrypt is not set by default to work with every network you connect to, only those you specify to use the 127.0.0.2 (or /1) for the DNS address. For WiFi, this is a good thing because mobile devices will often use networks that redirect to their own resolvers, like public WiFi welcome pages. Using DNSCrypt then would result in no DNS going beyond the captive portal.

Dealing with a bad DNSCrypt resolver.
DNSCrypt currently doesn't support more than one resolver at a time. This means that if the resolver you're using goes down at any time, you've got to change it to get DNS resolution back.

There are 2 ways to handle this:

1. Go into `/etc/default/dnscrypt-proxy`. Comment out the resolver which isn't working and uncomment a different one which presumably is.
2. Add a second non-DNSCrypt resolver for the system to use as a fallback. It'll only be used when the priority 1 resolver can't be reached and it's a seamless transition from one to the other, and then if P1 resolver comes back online, it'll be used again.

The dnscrypt-proxy config file will also give you a .csv file in `/usr/share/...` which mirrors of the resolver list on GitHub so you don't need to be online to look up a different resolver's info. Don't forget that you can ping even a DNSCrypt server from a VM or different computer on the network.

DNS caching: dnsmasq
Package *dnsmasq-base* is included by default in Ubuntu and Debian 8 but not Debian 7. It is an extension of NetworkManager but it isn't capable of caching. Its purpose is to funnel DNS lookups from all networks on all interfaces to whatever DNS IPs resolvconf detects are in use for the network you're on.

If you don't want to use the network-provided DNS for all networks you connect to, or if you want to use DNS caching on your computer, disable dnsmasq-base.

```
pkexec gedit /etc/NetworkManager/NetworkManager.conf
```

Comment out:

```
#dns=dnsmasq
```

Then you can disable the startup script for dns-clean.

```
sudo update-rc.d dns-clean disable
```

Flush all dns info.

```
sudo /etc/init.d/dns-clean
```

And restart NetworkManager.

```
sudo restart network-manager
```

But if you want to cache resolved IP addresses using dnsmasq, you need the full dnsmasq installation. See here:
https://help.ubuntu.com/community/Dnsmasq

Note: Dnsmasq's cache is stored in RAM so does not survive reboots. For persistent DNS caching, use pdnsd or unbound.

DNS Caching: pdnsd
Not included in either distro by default.

```
sudo apt-get install pdnsd
```

During installation, choose “*manual*”. Then set the configuration files.

```
pkexec gedit /etc/pdnsd.conf
```

Add this entire server block above the *root servers* section:

```
server {
 label="whatever dns service you choose";
 ip="first_provider_ip","second_provider_ip","etcetera...";
 timeout=30;
 interval=60;
 uptest=ping;
 ping_timeout=50;
 purge_cache=off;
 proxy_only=on;
}
```

Then tell it to start on boot.
`pkexec gedit /etc/default/pdnsd`
Edit:
```
START_DAEMON=yes
```

Start the daemon.
`sudo service pdnsd start`

Verify it's working.
Run dig on any domain and note the query time. Then run the dig command again. The second query time should be significantly less, if not zero.
`dig debian.com`

Also for some pdnsd performance changes see:
https://wiki.archlinux.org/index.php/pdnsd#Performance_settings_for_home_broadband_users

DNSCrypt with DNS caching
With this setup, NetworkManager tells pdnsd it wants a DNS lookup and pdnsd then tells DNSCrypt to make the lookup. pdnsd will then cache the returned IP and also keep `/run/resolvconf/resolv.conf` set at 127.0.0.1 so there's no reload scripts needed.

No changes are needed for DNSCrypt, only pdnsd and NetworkManager. The key points of this are the same if you're using Dnsmasq and dnscrypt.org has instructions for using unbound.
`pkexec gedit /etc/pdnsd.conf`
Edit your *server* { … } entry from above with:
```
label="dnscrypt-proxy";
ip="127.0.0.2";
```

Tell NetworkManager (or the interfaces config file) to talk to pdnsd instead of directly to DNSCrypt.
`Network > Wired > Ipv4 Settings`
If you're not using a static IP address, change `Method` to `Automatic (DHCP) addresses only`.

Then add:
```
DNS servers: 127.0.0.1
```

Restart pdnsd and NetworkManager.
`sudo service pdnsd restart && sudo restart network-manager`

Verify resolv.conf is 127.0.0.1
`cat /etc/resolv.conf`

Verify the cache is being used (remember, run this twice for the time difference)
`dig debian.com`

Finish by verifying with tcpdump or Wireshark that DNS traffic is encrypted .

sysctl

Harden and Optimze the TCP Stack

The file `/etc/sysctl.conf` is the same from Debian to Ubuntu and within each release so you can copy between releases. When ufw is enabled, the file `/etc/ufw/sysctl.conf` overrides several settings in the kernel and `/etc/sysctl.d/`. For more detailed info on sysctl switches and how ufw affects them, see:
http://thesimplecomputer.info/pages/adventures-in-linux-tcp-tuning-page2

```
pkexec gedit /etc/sysctl.conf
```

Uncomment:

```
net.ipv4.conf.default.rp_filter=1
net.ipv4.conf.all.rp_filter=1
net.ipv4.conf.all.accept_redirects = 0
net.ipv6.conf.all.accept_redirects = 0
net.ipv4.conf.all.send_redirects = 0
net.ipv4.conf.all.accept_source_route = 0
net.ipv6.conf.all.accept_source_route = 0
```

Add:

```
#Set system's maximum receive buffer size to 16MB
net.core.rmem_max = 16777216
#
#Set max send buffer size to 16MB
net.core.wmem_max = 16777216
#
#Set min, default, and max receive buffer sizes
#sizes per tcp socket (4KB, 82KB, 16MB).
net.ipv4.tcp_rmem = 4096 87380 16777216
#
#TCP send buffer sizes
net.ipv4.tcp_wmem = 4096 65536 16777216
#
#Disable TCP Timestamps
net.ipv4.tcp_timestamps = 0
#
#Decrease time sockets are held as fin-wait
net.ipv4.tcp_fin_timeout = 15
#
#Decrease amount of SYN retries
net.ipv4.tcp_syn_retries = 3
#
#Decrease amount of SYN/ACK retries
net.ipv4.tcp_synack_retries = 3
#
#Decrease the number of keepalive probes
net.ipv4.tcp_keepalive_probes = 3
#
#Enable fix for time-wait assassination hazards
net.ipv4.tcp_rfc1337 = 1
#
#Enable TCP MTU probing
net.ipv4.tcp_mtu_probing = 1
```

If you want to log suspicious incoming packets, uncomment:

```
net.ipv4.conf.all.log_martians = 1
```

Enable the changes.

```
sudo sysctl -p
```

Add a different TCP module to load at boot
Example: TCP Vegas. The process is the same for other modules but you need to find out proper syntax for the module name.

```
gksu gedit /etc/modules
```

Add:

```
tcp_vegas
```

Confirm is enabled on next reboot.

```
cat /proc/sys/net/ipv4/tcp_congestion_control
```

Then add to sysctl.conf:

```
#Use TCP Vegas congestion algorithm
net.ipv4.tcp_congestion_control = vegas
```

Disable IPv6
If you have IPv6 available to you, you should leave it enabled, especially for mobile devices. But if for some reason you want to disable it:

```
pkexec gedit /etc/sysctl.conf
```

Add:

```
#Disable ipv6
net.ipv6.conf.all.disable_ipv6 = 1
```

Enable the change.

```
sudo sysctl -p
```

Network Time

Debian 8 relies on the *systemd-networkd.service* which sets its system time according to the local network's DHCP server. Debian 7, Ubuntu 12.04 and 14.04 use ntpdate for traditional NTP.

For best timekeeping, you want several NTP servers geographically closest to you. By default, Debian 7 uses NTP servers in coordination with the system's installed locale but Ubuntu uses its own NTP server in London.

Configure ntpdate for your region
See the official Network Time Protocol project's website to find servers for your region.
http://www.pool.ntp.org/

Then add them for ntpdate.

```
pkexec gedit /etc/default/ntpdate
```

Edit:

```
NTPDATE_USE_NTP_CONF=no
NTPSERVERS=”0.first.ntp.address 1.second.ntp.address 2.third ntp address
3.fourth ntp address”
```

```
NTPOPTIONS="-bu"
```

Note: The addresses must be separated by a space, and don't include the word “server”.

Alternatively, you can set the network's perimeter appliance, router or gateway as the NTP server. For this to work, the target device must be running an NTP server. Don't do this for mobile devices.

```
pkexec gedit /etc/default/ntpdate
```

Edit:

```
NTPDATE_USE_NTP_CONF=no
NTPSERVERS=”192.168.1.1”
NTPOPTIONS="-bu"
```

ntpdate on demand

When finished configuring or to re-synchronize ntpdate immediately:

```
pkexec gedit /etc/systemd/systemct
sudo ntpdate-debian -bu
```

systemd-timesyncd

systemd-timesyncd uses Simple NTP instead of sourcing the time from DHCP. It allows for multiple NTP servers to be used but it's not enabled by default. To enable it and use this in addition to, or in place of, systemd-networkd.service, see:
https://wiki.archlinux.org/index.php/Systemd-timesyncd

Spoof MAC Address

MAC Changer

MAC Changer is used to spoof the MAC address of an ethernet or WiFi interface.
Package *macchanger-gtk* is an optional graphical frontend for MAC Changer.

```
sudo apt-get install macchanger macchanger-gtk
sudo apt-get install macchanger-gtk
```

Important Each time you disconnect from a network, the NIC's MAC will return to its real one!

From version 1.7.0-5 (in Debian 8), MAC Changer solves this issue. It will ask during installation if you want to randomize the MACs of all interfaces each time the computer connects to a network.

The problem for earlier MAC Changer versions is that NetworkManager doesn't use pre-up anymore so we can't just put a script in there. If you don't have systemd to automate runing MAC Changer before the interface is up, then it is a manual process.

The best you can do is create a script to run MAC Changer on boot. Then also *macchanger-gtk* open (autostart it if you must) when at undesirable WiFi areas so you have a graphical reminder that you need to change the MAC before going online.

Note: To manually change an interface's MAC address through a terminal, the interface must be down (WiFi and/or ethernet turned off).

Startup script (non-systemd)
```
pkexec gedit /etc/rc2.d/S99macchanger
```

Make the contents:
```
#!/bin/sh
/usr/bin/macchanger -a wlan0
exit
```

Note: To spoof your ethernet interface's MAC, add a second line for it (eth0, p2p1, etc.).

Then make it executable.
```
sudo chmod +x /etc/rc2.d/S99macchanger
```

To see your current MAC address compared to your real one:
```
macchanger -s wlan0
```

Adobe Flash Administration

Create a Flash config file
Only for Adobe Flash, not Google's Pepper Flash or any Pepper wrappers. This file overrides the Macromedia website settings.

Create the file.
```
pkexec gedit /etc/adobe/mms.cfg
```

Then select what to add from the full Adobe Flash 11.2 administrator's guide.
https://forums.adobe.com/docs/DOC-1911

TCP Wrappers

This can add an extra layer of security to some services but they are not a substitute for firewall rules or hardening a service's config settings. TCP wrappers are mainly for server applications but wrapper rules are only used by daemons which are compiled to use them.

To check if a specific service supports TCP wrappers:
```
ldd /path/to/binary | grep libwrap.so
```

If so, *libwrap.so* will be present in the output.

Example: For a SSH server, block all network access to other services except for SSH over a specific IP address.
```
pkexec gedit /etc/hosts.deny
```
Add:
```
All EXCEPT: SSHD ip_address
```

Note: for Gnome Shell users, gdm supports TCP wrappers.
https://help.gnome.org/admin/gdm/stable/security.html.en#xdmcpaccess

Network Sharing

NFS

Simple yet capable file server access can be done with the Network File System protocol between *nix, Apple and BSD systems. Kerberos can be added for finer-grained access control. NFS is overall more limited compared to Samba and while it can work with Windows using some 3rd party programs, Samba is usually the preferred choice for Windows network sharing.

Samba

Neither Debian nor Ubuntu include Samba by default but many derivatives do. For full Samba client and server:

```
sudo apt-get install samba
```

Samba User

The Samba daemon should first be given its own standard user account for more stringent privileges. Below I use the name sambauser for the account's username.

```
sudo adduser --system --shell /bin/false --no-create-home --ingroup
sambashare --disabled-login --disabled-password sambauser
```

Then add the new samba user to the samba group.

```
sudo usermod -a -G sambashare sambauser
```

Now you can set up Samba like normal. Set the Guest Account to sambauser or whatever name you chose. If you can configure manually without the GUI tool, that's best. The Samba GUI setup tool is helpful but has a lot of dependencies, and making a Samba user account is still important.

```
sudo apt-get install system-config-samba
```

For pointers when setting up with the GUI tool:
https://www.youtube.com/watch?v=-wUfzdiE4m8

A Samba server configuration tutorial with performance tips.
https://calomel.org/samba.html

Extra tips.
https://wiki.archlinux.org/index.php/Samba/Tips_and_tricks

Samba Project manual pages
https://www.samba.org/samba/docs/man/manpages/

SSH

Debian and Ubuntu include package *openssh-client* but if it's not needed, purge it. The

OpenSSH client also handles SFTP.

Secure OpenSSH server
As a starting point:
http://la-samhna.de/library/brutessh.html

SSH server manual.
http://linux.die.net/man/8/sshd

Harden OpenSSH client
```
pkexec gedit /etc/ssh/ssh_config
```

Uncomment/edit:
```
ForwardAgent no
ForwardX11 no
RhostsRSAAuthentication no
Protocol 2
```

SSH client manual.
http://linux.die.net/man/1/ssh

CUPS Printing

Ubuntu includes a full CUPS installation with network browsing (cups-browsed) enabled by default. Debian does not.

To access the CUPS browser interface:
http://localhost:631

Or use the config file.
```
pkexec gedit /etc/cups/cupsd.conf
```

Debian Wiki: SystemPrinting
https://wiki.debian.org/SystemPrinting

For configuration examples with descriptions, see:
https://www.cups.org/documentation.php/ref-cupsd-conf.html

OpenPrinting database by The Linux Foundation
https://www.openprinting.org/printers

AppArmor and Malware Scanners

Basic AppArmor

Debian does not include AppArmor by default. AppArmor and a few profiles and abstractions are included with Ubuntu (in complain mode) but not all Ubuntu-based distros. What's below will only install and set up AppArmor using the default profiles. Apparmor profiles are located in `/etc/apparmor.d`.

Some programs to restrict with AppArmor are web browsers, email and messaging clients, network connection daemons (Samba, SSH, etc.), DNS handlers, print daemons and PDF and image viewers.

Install AppArmor with the default profiles

```
sudo apt-get install apparmor-notify apparmor-profiles apparmor-utils
```

Start AppArmor (Ubuntu).

```
sudo service apparmor start
```

For Debian, AppArmor requires a few more steps and a reboot. See “*Enable AppArmor*”:
https://wiki.debian.org/AppArmor/HowToUse

To enforce all profiles:

```
sudo aa-enforce /etc/apparmor.d/*
```

To put all profiles into complain mode:

```
sudo aa-complain /etc/apparmor.d/*
```

To put a specific profile into complain mode:

```
sudo aa-enforce program_name
Example: sudo aa-enforce chromium-browser
```

To tell AA to search the syslog for denials:

```
sudo aa-logprof program_name
```

To reload a specific profile after making changes to it:

```
sudo aa-enforce program_name
```

View status of all Apparmor profiles.

```
sudo aa-status
```

More profiles and development profiles for release with future Ubuntu versions can be found here:
https://bazaar.launchpad.net/~apparmor-dev/apparmor-profiles/master/files

Advanced AppArmor

Strong AppArmor profiles will be specific to your system and will require some input from you to complete building them. For stronger base profiles of a few popular internet applications built specifically distros based on Ubuntu 14.04 (but easily fitted to Debian), see:

http://thesimplecomputer.info/apparmor/

See the README file for what works and what doesn't with those profiles. You'll need to finish them off with the logprof tool or manually with syslog denials. Anything more granular than those profiles you must make yourself, specifically to your requirements. This is the nature of AppArmor, one-size-fits-all solutions aren't possible here.

For more info on Apparmor:
http://ubuntuforums.org/showthread.php?t=1008906
https://doc.opensuse.org/documentation/html/openSUSE/opensuse-security/cha.apparmor.profiles.html

Rootkit Scanners

chkrootkit and rkhunter
If you're scanning for rootkits, you'll want both. Usually there is a false positive or several so if you see a suspicious result, do a web search for it.

```
sudo apt-get install rkhunter chkrootkit
```

Set chkrootkit daily cron job.

```
pkexec gedit /etc/chkrootkit.conf
```

Edit:

```
RUN_DAILY="true"
```

Set rkhunter daily cron job with weekly database update.

```
pkexec gedit /etc/default/rkhunter
```

Edit:

```
CRON_DAILY_RUN="yes"
CRON_DB_UPDATE="yes"
APT_AUTOGEN="yes"
```

Run either on demand.

```
sudo chkrootkit
sudo rkhunter --update --propupd && sudo rkhunter --check
```

To run rkhunter with the option to check with the package manager about binary changes (preferred but requires an internet connection):

```
sudo rkhunter --pkgmgr dpkg -c –sk
```

ClamAV

ClamAV is a basic antivirus scanner which can be run on-demand or as a daemon. Daemon mode can scan incoming email with plugins for several mail clients.

```
sudo apt-get install clamav
```

To change ClamAV's update schedule:

```
pkexec gedit /etc/clamav/freshclam.conf
```

Edit:

```
# Check for new database x times a day
```

```
Checks x
```

To choose new mirrors for the database downloads:
```
sudo dpkg-reconfigure freshclam
```

To use ClamAV on demand only, disable it from auto-starting on boot.
```
sudo update-rc.d clamav-freshclam disable
```
or
```
sudo systemctl clamav-freshclam disable
```

To update its database:
```
sudo freshclam
```

Scan something.
```
sudo clamscan -ir /home
```
(For verbose output, make parameters -irv)

Or scan everything.
```
sudo clamscan -ir / --exclude=/proc --exclude=/sys --exclude=/dev
```

ClamAV's graphical frontend is ClamTK.
```
sudo apt-get install clamtk
```

For more on ClamAV:
https://help.ubuntu.com/community/ClamAV

Avast, AVG, BitDefender, Comodo, ESET, Kaspersky and Sophos are some proprietary alternatives to ClamAV.

Web Browsers

Chromium, Chrome

Important Chromium should be installed from the distro's repositories. Google's commondatastorage hosted snapshots are not intended for stable use and don't enable full (SUID) sandboxing.

Default profile locations
Chromium
```
~/.config/chromium/Default
```

Chrome
```
~/.config/google-chrome/Default
```

Add binary switches to Chromium
You can either use the .desktop file method for individual users, or use this method for all users.

```
pkexec gedit /etc/chromium-browser/default
```
Add:
```
CHROMIUM_FLAGS="--no-referrers"
```

Add binary switches to Chrome
This must be done on a per-user basis. Don't modify the original .desktop file in `/usr/share/applications` because it will be overwritten the next time Chrome updates.

First copy the original .desktop file to your local apps area.
```
cp /usr/share/applications/google-chrome.desktop
~/.local/share/applications/google-chrome.desktop
```

Then edit to how you want. For example, to disable referrers in Chrome:
Add:
```
Exec=/opt/google/chrome/google-chrome --no-referrers %U
```

Do this for each instance of “Exec=” in the file; for Incognito, New Window, etc.

Set Chrome to only check for updates once a week
Default is daily.
```
sudo mv /etc/cron.daily/google-chrome /etc/cron.weekly/google-chrome
```

Firefox, Iceweasel

Note: Pale Moon is still based on an older ESR version of Firefox which uses a few different settings.

Profile location:
```
~/.mozilla/firefox/*.default
```

Use Chrome's Pepper Flash in Firefox
See Fresh Player plugin, still in early development stages.
http://www.webupd8.org/2015/01/fresh-player-plugin-sees-new-release.html

Remove default search engines in Debian
This will have to be run again each time Iceweasel updates.
```
sudo rm /etc/iceweasesl/searchplugins/common/*.xml
sudo rm /etc/iceweasel/searchplugins/locale/en-US/*.xml
```

Remove default search engines in Ubuntu
This will have to be run again each time Firefox updates.
```
sudo rm /usr/lib/firefox/distribution/searchplugins/locale/en-US/*.xml
```

about:config
See:
http://thesimplecomputer.info/tscs-firefox-tweak-guide

Note: That page is a supplement to the other sources listed in the article.

Relocate cache to volatile memory

This only applies if you have disk caching enabled. The default cache directory is `/home/user_name/.cache/mozilla/firefox`. To view cache usage info, see: `about:cache`. For Iceweasel, this doesn't mount the offline cache in RAM, only the regular disk cache.

Using /tmp
If you mount `/tmp` as tmpfs, you should use that for the new cache location. In `about:config`, create a new string with the preference name:
`browser.cache.disk.parent_directory`

Make cache location:
`/tmp/firefox`

If you have multiple profiles in the same system user account, separate the cache locations for them.
`/tmp/firefox/profile_name`

Using /run/shm
If you don't mount `/tmp` as tmpfs, use `/run/shm` instead.
`browser.cache.disk.parent_directory; /run/shm/firefox`

Then delete the old cache locations.

Extensions and Search Engines

Content blockers
*uBlock** (Chromium and Firefox)
Blocks ads, annoyances and tracking content using filter subscriptions like AdBlock Plus does. Includes (among other things) hosts file subscriptions, element picker, custom filters and rules, an advanced mode for per-domain control of scripts, frames and images and a real-time traffic monitor to see the full URL of all elements in a page and whether they're blocked or allowed.

* There are two versions of uBlock from separate lead developers. To describe the differences, the (very) tl:dr is that uBlock Origin is considered feature complete. It will be maintained primarily for stability and new features are of very low priority. On the other hand, uBlock (non-Origin) does take feature requests with plans to evolve the extension further.

Here's another (and very down to Earth) explanation.
https://www.ublock.org/faq/

ublock Origin - https://github.com/gorhill/uBlock
uBlock - https://www.ublock.org/

Adblock Plus (Chromium and Firefox)
If you find uBlock isn't your preference, Adblock Plus is still a viable choice.
https://adblockplus.org/

Note: Adblock Edge is discontinued as of June 2015.
https://addons.mozilla.org/firefox/addon/adblock-edge/

Additional blocklists for uBlock and ABP
https://fanboy.co.nz
https://github.com/gorhill/uBlock/wiki/Filter-lists-from-around-the-web

uMatrix (Chromium/Firefox)
A powerful and advanced content blocker which can control 1st and 3rd party CSS, cookies, scripts, plugins, frames and other web requests. Can spoof referrers and user agent strings, has cache control options, uses several HOSTS files for blocking content from undesirable domains and can filter HTTP requests out from mixed-content sites. uMatrix is by gorhill, uBlock's creator.
https://github.com/gorhill/umatrix

NoScript (Firefox)
A powerful content blocker for advanced users. Controls 1st and 3rd party scripts, plugins, frames and other web requests. Can force HTTPS connections and has dedicated protection against click-jacking and cross-site attacks (ClearClick and ABE).
https://noscript.net/

HTTPS Everywhere
For Chromium and Firefox. Forces TLS encrypted connections for popular websites and supports user-added site rules.
https://www.eff.org/https-everywhere

Search Engines
This is just a sort of cheat sheet for OpenSearch URLs without identifiers or other tags. For info on making your own, see:
http://thesimplecomputer.info/search-engines-for-almost-everyone

Chromium/Chrome (and Midori)

Bing
`https://www.bing.com/search?q=%s`

DuckDuckGo
`https://duckduckgo.com/?&q=%s`

Google
`https://encrypted.google.com/search?q=%s`

Ixquick
`https://ixquick.com/do/search?q=%s`

Privatelee
`https://privatelee.com/search?q=%s`

Startpage
`https://startpage.com/do/search?q=%s`

Youtube

```
https://www.youtube.com/results?search_query=%s
```

Firefox
Need to use OpenSearch XML plugins. Either examine and use what's in the Mozilla Add-on pages or see the link above for making your own search plugins.

General Performance

Boot

GRUB boot profile
Do this only after all startup services are configured.
`pkexec gedit /etc/default/grub`
Add:
```
GRUB_CMDLINE_LINUX_DEFAULT="quiet splash profile"
```

Then update GRUB and reboot.
`sudo update-grub && sudo reboot`

When booted, remove “profile” from GRUB and run `update-grub` again.

Ureadahead & systemd-readahead
A new pack file is generated each month by the ureadahead daemon but you can force a new one after the boot profiling above. A new pack file will be generated on next boot and the following boots should be slightly quicker (but realistically is unnoticeable).
`sudo rm /var/lib/ureadahead/pack`

systemd includes its own readahead services and should not be used with ureadahead.
https://wiki.archlinux.org/index.php/Improve_Boot_Performance#Readahead

Boot charts
Boot charts give you a graphical readout of what processes happen when during the boot sequence and how long they take. It's not a performance changer, it helps find where changes could be made. The old *bootchart* package is now part of systemd but still available for non-systemd distros.

Quick instructions for using either:
https://wiki.ubuntu.com/BootCharting

For more info:
http://www.freedesktop.org/software/systemd/man/systemd-bootchart.html

A similar systemd tool is *systemd-analyze*.
https://wiki.archlinux.org/index.php/Improve_Boot_Performance#Using_systemd-analyze

Applications

Preload
Will take a few reboots until it learns what programs you use most often.

```
sudo apt-get install preload
```

For more info:
http://www.hecticgeek.com/2013/05/using-preload-ubuntu-13-04/
http://techthrob.com/2009/03/drastically-speed-up-your-linux-system-with-preload/

Speed up LibreOffice and OpenOffice
https://wiki.archlinux.org/index.php/Openoffice#Speed_up_OpenOffice

Modem Manager
The package *modemmanager* is used by NetworkManager for mobile broadband modems but it often causes delays on shutdown. If you don't need it, purge it.

```
sudo apt-get purge modemmanager
```

TLP power savings script
Intended for laptops.
http://linrunner.de/en/tlp/tlp.html

For TLP's advanced settings, see the config file at `/etc/default/tlp`.

General systemd optimizations
https://wiki.freedesktop.org/www/Software/systemd/Optimizations/

Miscellaneous

Thunderbird
Nearly all `about:config` settings from the Torbirdy changelist can be used to harden Thunderbird, even if not using the Torbirdy plugin or Tor network.
https://trac.torproject.org/projects/tor/wiki/torbirdy/changes

Play encrypted and restricted dvds
Ubuntu

```
sudo apt-get install libdvdread4 && sudo /usr/share/doc/libdvdread4/install-css.sh
```

Debian

```
sudo apt-get install libdvdread4
```

Debian's libdvdread4 package doesn't include an install script for libdvdcss. You'll have to grab *install-css.sh* from Ubuntu's libdvdread package.
http://packages.ubuntu.com/trusty/libdvdread4

For more info on playing DVDs in Debian:
https://wiki.debian.org/CDDVD

Unity static scroll bars
Can use Unity Tweak Tool, dconf tool or set it from the terminal.
```
gsettings set com.canonical.desktop.interface scrollbar-mode normal
```

Desktop wallpaper changer script for Gnome
https://extensions.gnome.org/extension/1000/random-walls/

If you would rather use a cron job for all users on the system, see:
https://stackoverflow.com/questions/24205169/unable-to-change-wallpaper-using-cronjob-on-ubuntu

systemd
https://major.io/2015/02/11/rotate-gnome-3s-wallpaper-systemd-user-units-timers/

Fix for laptop immediately coming out of sleep mode
Affects 16.04 and 14.04, not 12.04 (as far as I've seen).

Problem: When running on battery power and suspending a laptop to RAM, within seconds it comes back out of sleep and is entirely unresponsive. The only way to recover is to hard reset the computer.

Some laptops this happens on every sleep, sometimes it's every 2, 3, or seemingly at random. The problem is due to ACPI wakeup being enabled for some device controller(s) on the computer (usually either USB or ethernet).

Solution: The fix is specific to your machine's PCI assignments so here is only a general walkthrough. First find which device(s) are set to enabled for suspend wakeup.
```
cat /proc/acpi/wakeup
```

You'll see something like this:
```
Device      S-state      Status   Sysfs node
GLAN    S4  *disabled
POP2    S4  *enabled  pci:0000:02:00.0
```

Things like PWRB (power button) or LID0 (lid close sensor) are not the problem, but you want all the others disabled.

Take note of the **enabled* PCI address and run `lspci` to find the device's name on it--that's what needs to be disabled. So the POP2 device above which, let's say for example, lspci tells us is is the ethernet controller, needs to have ACPI wakeup disabled.

Disable it with:
```
echo disabled | sudo tee /sys/bus/pci/devices/0000:02:00.0/power/wakeup
```

Then do some suspend and resume test runs to make sure everything works correctly. If so, add the command to `/etc/rc.local` so it's run on each boot. I've not experienced this problem with Debian 8, but if you try this with a distro using systemd, know that rc.local may not be run.

Add or remove Hibernation entry in Shutdown menu (Ubuntu)
Do this for both “Re-enable hibernate by default...” entries.

```
pkexec gedit /etc/polkit-1/localauthority/50-local.d/com.ubuntu.enable-hibernate.pkla
```

Edit:

```
ResultActive="no"
OR
ResultActive="yes"
```

The End

Problem?
contact@thesimplecomputer.info