

# **The Ubuntu for Network Engineers Cookbook**

**Advance your “Network Engineer of the future” career with the OS of the Cloud**



**Michael Hubbard**

**2023**

# Table of Contents

Installing Ubuntu 18.04.....	7
Using Ubuntu.....	7
Casting your display .....	8
Opening Applications .....	9
Configuring the Dock .....	11
Customizing Gnome.....	11
Gnome extensions .....	13
Managing Files.....	14
Display the Full path in Nautilus .....	14
Easily preview files in Nautilus.....	15
Why do I have a red X on a file or folder?.....	15
Creating a bootable USB stick from an ISO image.....	17
Reference .....	17
Using QEMU to test an ISO or Bootable USB .....	17
Working with the Linux File System.....	18
lsusb .....	19
SSH .....	20
Locating files from the terminal .....	24
Drill - A graphical and CLI tool for searching files.....	24
Working with the Terminal .....	25
Cut/Paste .....	25
Manual Pages .....	25
Tab Completion.....	25
Virtual Console.....	26
History .....	26
<i>Incremental</i> history searching - An extremely handy tool.....	27
Auto-jump .....	29
bd .....	30
Package management.....	31
Display a new line on the prompt in terminal.....	33
Installing and using zsh instead of bash .....	34
Install Oh My zsh .....	35
Add a theme, some plugins and aliases .....	36
Bat - A cat clone with syntax highlighting and Git integration .....	37
Installation .....	38
Enhanced Terminals .....	39
Pick the default terminal program .....	39
Bash Commands.....	40
grep .....	44
References .....	48

Uncomplicated firewall .....	50
The Ubuntu Software Store .....	55
Brasero – ISO, CD, DVD tool .....	55
Cheese webcam .....	55
FBReader - A multi-platform ebook reader .....	55
Flameshot – Screenshot tool .....	55
Glances - An alternative to htop .....	56
Gnome Network Tools - A graphical collection of networking tools .....	56
Meld - A great cross platform graphical file/directory comparison tool .....	56
Openshot - A free, open-source, non-linear video editor .....	57
Qalculate (gtk+ui) – Calculator with Reverse Polish Notation .....	57
Remmina - The GTK+ Remmina Remote Desktop Client .....	58
Foliate - E book reader .....	59
Chapter 2 Tools installed from the Terminal .....	60
ClamAV – Open Source Anti-Virus Scanner .....	60
ClamTk - A graphical frontend to ClamAV .....	61
Etcher - Flash OS images to SD cards & USB drives, safely and easily .....	61
C Compiler .....	61
inxı - A full featured terminal system information tool .....	61
unetbootin .....	62
OpenSSH Server .....	62
Sublime text .....	62
Solaar .....	63
Veracrypt .....	63
exa .....	63
RDFind .....	65
NCDU .....	66
htop .....	66
Installing htop .....	67
Running htop .....	67
iostat .....	67
iostat .....	67
nload .....	68
Python 3 and Python tools .....	69
PIP3 .....	69
XlsxWriter .....	69
Tkinter .....	69
xsltproc .....	69
Speedtest-cli .....	69
Teamviewer .....	69
Wireshark .....	70
CCZE .....	72
Snaps - A new way to install programs on Ubuntu .....	73
How to install a Snap .....	73
Refreshing Snaps .....	73
To list all installed Snaps .....	74
Flatpak .....	76

Managing Network Devices .....	78
Using the GUI to Configure network Connection Profiles .....	78
Details .....	78
Identity .....	80
IPv4.....	81
IPv6.....	82
Security .....	83
Network Manager CLI.....	85
Examples:.....	88
To shut down or bring up an interface: .....	88
List WiFi details .....	90
List all devices on the system .....	90
List details for one Interface .....	90
List All Connection Profiles .....	92
Using Vlan tags.....	94
Monitoring interface status while a switch reboots .....	95
NMCLI Cheat Sheets.....	96
Wireless Terminal commands .....	97
iwconfig .....	97
iwgetid .....	98
iwlist.....	99
Display Saved SSIDs.....	101
Wavemon .....	102
Ethtool .....	105
Set speed/duplex to 100 full.....	105
Advertising speed and duplex.....	105
Display autonegotiation .....	106
Display statistics .....	106
Display only Errors .....	106
Display only Collisions.....	106
Show Features.....	107
Display permanent address .....	108
Flash the NIC LED .....	108
Testing a NIC .....	108
Show Driver Information.....	108
Make Changes Permanent After Reboot.....	109
LLDP client for Linux .....	109
Installation .....	109
LLDP client Usage.....	109
Running interactively.....	110
Networking Tools .....	114
Sipcalc .....	114
Sipcalc using an Interface .....	115
sipcalc -h .....	115
Ipcalc .....	117
ipcalc -h.....	117
Arpscan.....	119

Scan using the adapter's settings .....	119
grepcidr .....	121
nmap .....	122
Reporting .....	125
More Information.....	125
nping .....	125
Traceroute for IPv4.....	126
Tracepath .....	127
Socket Status (ss).....	129
LinSSID .....	132
Minicom.....	132
Installation .....	132
Setup mode .....	133
IPMI Tools .....	134
Installation .....	134
SNMP .....	134
Installation .....	134
Cisco devices .....	135
Remove EXIF data from photos .....	137
Viewing EXIF data .....	137
DNS Enumeration Tools .....	139
DNS Brute Force Wordlists .....	139
DNSENUM .....	139
dnsmap.....	139
dnsrecon.....	139
Installation .....	140
dnswalk.....	142
dnscan .....	142
Anubis.....	142
DNSTracer.....	144
subbrute .....	145
DNSDIAG .....	146
Installation .....	147
dnsping.....	147
dnstraceroute.....	148
dnseval .....	149
CTFR .....	149
MySQL .....	151
TFTP .....	151
Appendix A – A Dell Laptop running Linux .....	165

The following conventions are used in “Ubuntu for Network Engineers”:

*Italic*

Used for URLs, email addresses, acronyms, filenames, and new terms.

**Constant Width (Consolas Typeface)**

Used for program listings and terminal command output

**Constant Width Bold (Consolas Typeface)**

Text that should be typed into the terminal

*Constant Width Italic*

Text that should be replaced by user supplied values

## Installing Ubuntu 18.04

The first question is should I just spin up a virtual machine or use bare metal. I used Linux for a few years as a VM before I bought dedicated hardware. The advantage of using a VM is that if you damage it or decide for any reason it's not the right distribution you can just delete it and try another one. The drawback to a VM is that the hypervisor hides the hardware so you don't really get the experience of installing, configuring and maintaining Linux.

My suggestion if you have never touched Linux is to install Ubuntu as a VM and get some experience with it. You won't have to spend any money, Ubuntu is free, although I usually donate \$10 when I download it to install on a fresh machine, and you can run any of the tools in this book.

I have Kali running on a System 76 Gazelle laptop and Ubuntu on a Dell G5 laptop. I did this because I wanted to get my Linux certifications and I felt that I needed to be running Linux on my daily driver to really learn. The VM can do just about anything the bare metal can do but when I would get stuck on something I would just shut it down and go back to Windows instead of figuring it out.

There are a lot of good tutorials on installing Ubuntu 18.04 available on the Internet so I am not going to cover it here. It is actually very easy, pretty much click, click, next, reboot! The site [linuxconfig.org](http://linuxconfig.org) has a lot of great Ubuntu tutorials, here is link to a tutorial on installing 18.04 – [How to install Ubuntu 18.04](#).

Appendix A covers the installation I did on a Dell G5 laptop that wasn't 100% compatible out of the box. It took a lot more time than I expected to get everything working on the Dell and I documented the steps in the Appendix. Dell and System 76 both sell laptops with Ubuntu pre-installed. The System 76 Oryx Pro will definitely be my next machine. It is a beautiful powerhouse with Nvidia 2080 graphics, up to 32 GB of RAM and massive storage capacity. Unlike the Dell XPS 13/15 lines it has Ethernet, USB A, and the other ports I find handy.

Even if you buy a System 76 or Dell laptop with Ubuntu preinstalled you may still want to take a look at the Appendix since I documented some things like customizing the Grub boot loader and running nVidia and Intel graphics.

In this book I am going to document the tools that I found useful in the transition to Linux for network engineering tasks.

A note on encryption. Ubuntu supports the Linux LUKS full disk encryption. During setup you can select the encrypt full disk option. This is, of course, a good idea for security but if you encrypt and then decide you don't want it or you want to upgrade to the next release of Ubuntu you will need to decrypt. Use the instructions here to decrypt the LUKS volume:

[How to remove LUKS encryption](#)

<https://unix.stackexchange.com/questions/60971/how-to-remove-luks-encryption>

## Using Ubuntu

Switching from Windows to Ubuntu with the Gnome desktop is both easy and hard. How can that be? It's simple, the basic operation of the desktop is similar to Windows. The file manager (Nautilus) is similar to Windows Explorer and of course Chrome or Firefox work just like they do in Windows including syncing, Lastpass, Grammarly, etc. There is a keyboard button called the superkey (on most keyboards it will have the Windows logo) just like the Windows key. You tap it and search for an application or pick from the running applications.

So, what is the hard part? Things like killing a hung application, searching for files from the terminal, checking free disk space are different and will take a little getting used to. Since MACOS is based on Open BSD Unix, once your muscle memory is up to speed you will be able to jump back and forth easily between Windows, Mac and Linux. In a later chapter I will discuss the terminal tools that make Linux so much fun and so powerful for a network engineer.

One of the big advantages for the Linux desktop is that you can customize almost everything since it's free and open source. You can change the terminal you use, the file manager, the way almost anything works and even the desktop itself.

The most popular desktop environments (DE) for Linux are Gnome (Ubuntu's default), and KDE (which is available on Ubuntu) but there are many more. I use the computer for work and just learned to customize Gnome to fit my workflow but I use Terminator for my terminal app and Polo for the file manager.

I listen to a lot of podcasts and KDE Plasma, XFCE and MATE are highly recommended desktop environments (DE) that you may like better than Gnome. If you watched the second episode of Mr. Robot (Season 1 – which I highly recommend) there is a scene where an executive tells Elliot that he runs KDE but understands why Elliott prefers Gnome.

These two short YouTube videos sum it up better than I can:

6 Things to Know When Switching to Linux from Windows  
<https://www.youtube.com/watch?v=wdqubB6hT8>

10 things you can do with Linux that you can't do with Windows  
<https://www.youtube.com/watch?v=JOOkfGv58u0>

The “Choose Linux” podcast is a show that covers a lot of Linux distributions and has a lot of tips. One episode in particular, [\*24: What We Wish We'd Known Earlier\*](#), is well worth the time to listen to. The three hosts talk about things they know that would have been very helpful when they first switched to Linux.

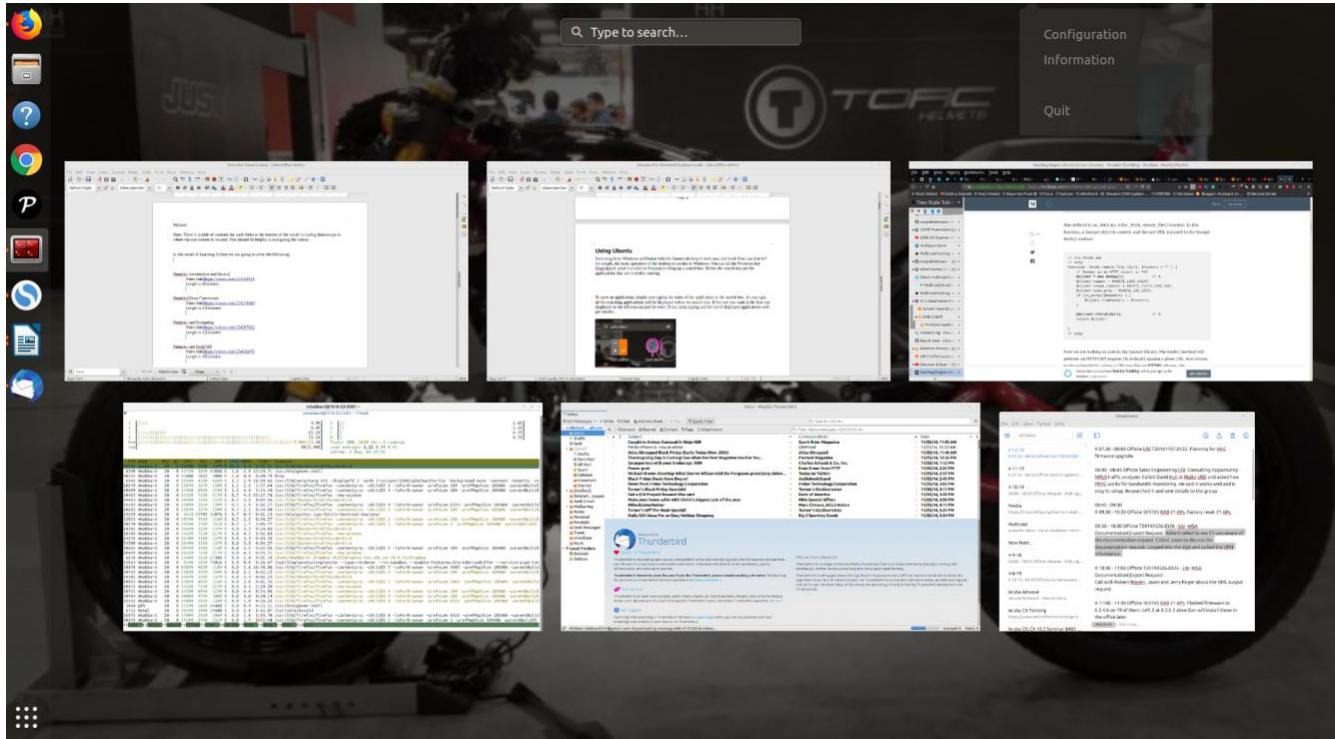
### Casting your display

I have a Vizio “Smart” TV with Chromecast built in. Using the Chrome browser I can click the three dot menu on the right and select “Cast...” to send the output to the TV.

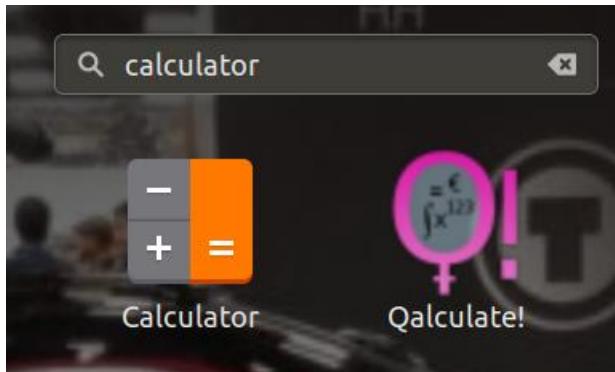
In VLC, select “Playback, Renderer” to send VLC’s output to the TV.  
I don’t have a Chromecast dongle, but I am guessing it would work the same way as the Vizio TV.

## Opening Applications

You can hit the Windows key (Superkey is what it's called in Ubuntu) to bring up a search box. Below the search box are the applications that are currently running. You can use the Tab key and enter or use the mouse to select one of the running applications.

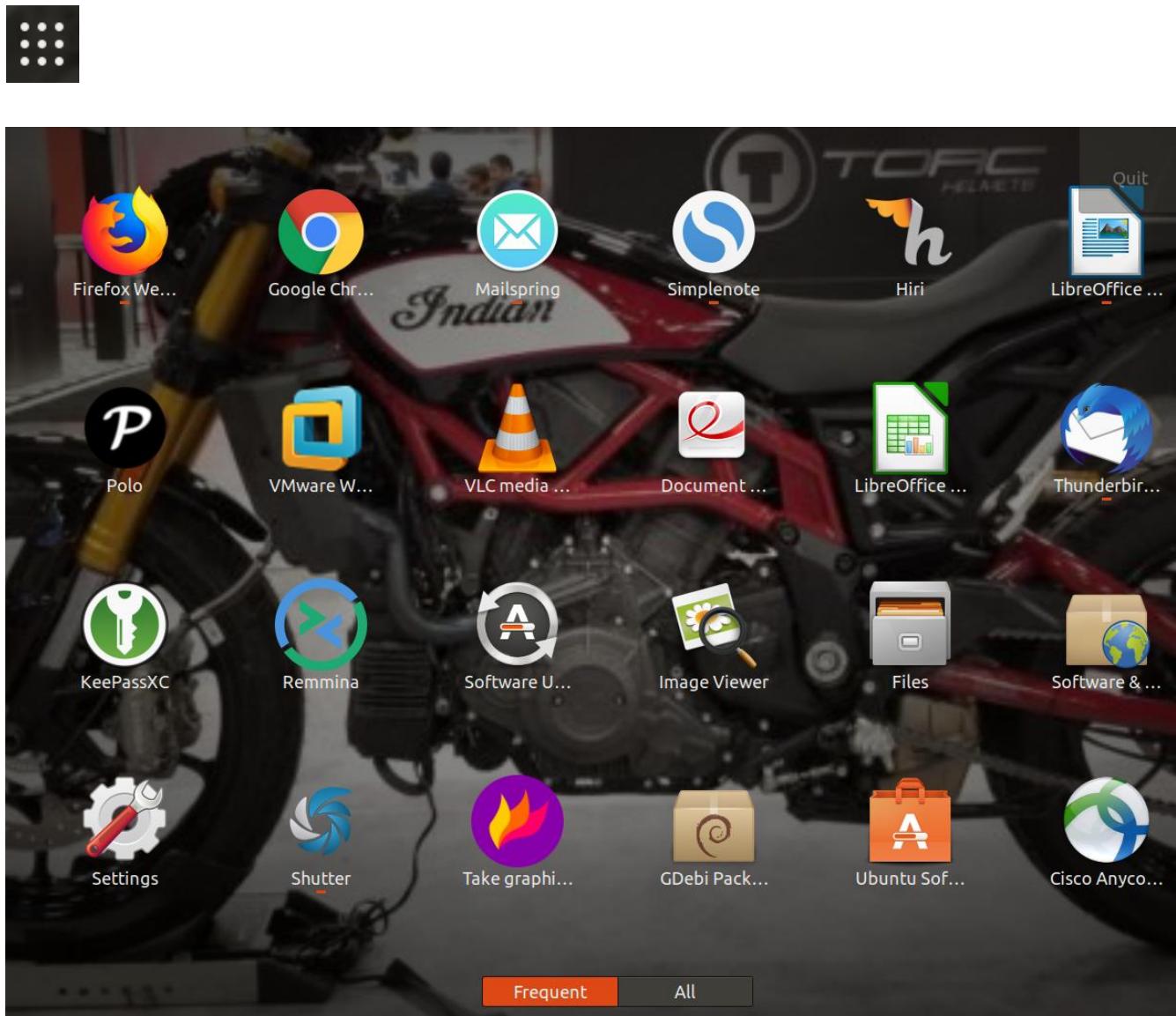


To open an application using search, simply start typing the name of the application in the search box. As you type, all the matching applications will be displayed below the search box. If the one you want is the first one displayed on the left you can just hit enter. If not, keep typing and the list of displayed applications will get smaller. In the screenshot below, hitting enter will open the built in calculator app. Right arrow, Enter will open Qalculate!



You can also use the mouse and click on any application that is displayed to run it.

The dock has a small icon at the bottom, it's three rows of three dots. You can click that icon to see the installed applications in a window. At the bottom, you can select all or frequent. Below is a screenshot with frequent selected. As you can see, it's very easy to get started.



If there are too many to display, a column of circles will appear on the right side of the screen. You can scroll down with the mouse or click on one of the circles to jump to the next page.

If you have several copies of the same application open you can switch between them using the super key and '\~' key (the key to the left of 1 on a US keyboard). You can also bring up the dock, click on the application icon and then select the one you want but the keyboard is much easier and faster.

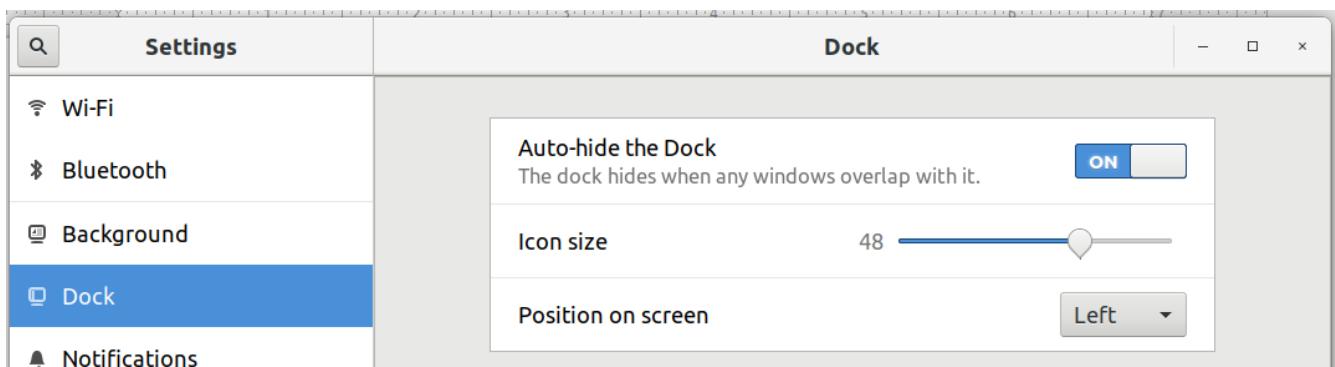
## Configuring the Dock

Like Windows and Mac, Ubuntu with the Gnome DE has a dock. You customize the dock using the Gnome Settings app. To open the Gnome Settings application, click the down arrow in the top right corner of the screen, then select the Wrench/Screw drive icon.



When the Settings application opens, click on Dock. One thing I do to help me use all three operating systems is to customize the dock to be similar across all OSes. I set Windows, Mac and Linux to display it on the left because I am almost 100% of the time on a laptop and the screen is wider than it is high. I also set the dock to auto-hide and display a few commonly used apps.

Using the super key and typing the first few letters is now my preferred work flow rather than clicking on an icon in the dock. One thing I like about Ubuntu is that when I'm using multiple monitors it puts the dock on all the monitors.



## Customizing Gnome

Ubuntu 18.04 ships with Gnome as the desktop environment instead of Unity. I liked Unity better, but with the steps listed below, Gnome is working pretty well for me. Unlike Windows, Linux distributions can use a lot of different desktop environments (DE). Gnome is the default desktop environment for Ubuntu and Redhat (CentOS) so it has a large percentage of the desktop market in Linux. If you want to try others there are several official flavors of Ubuntu that you can download from [Canonical](#).

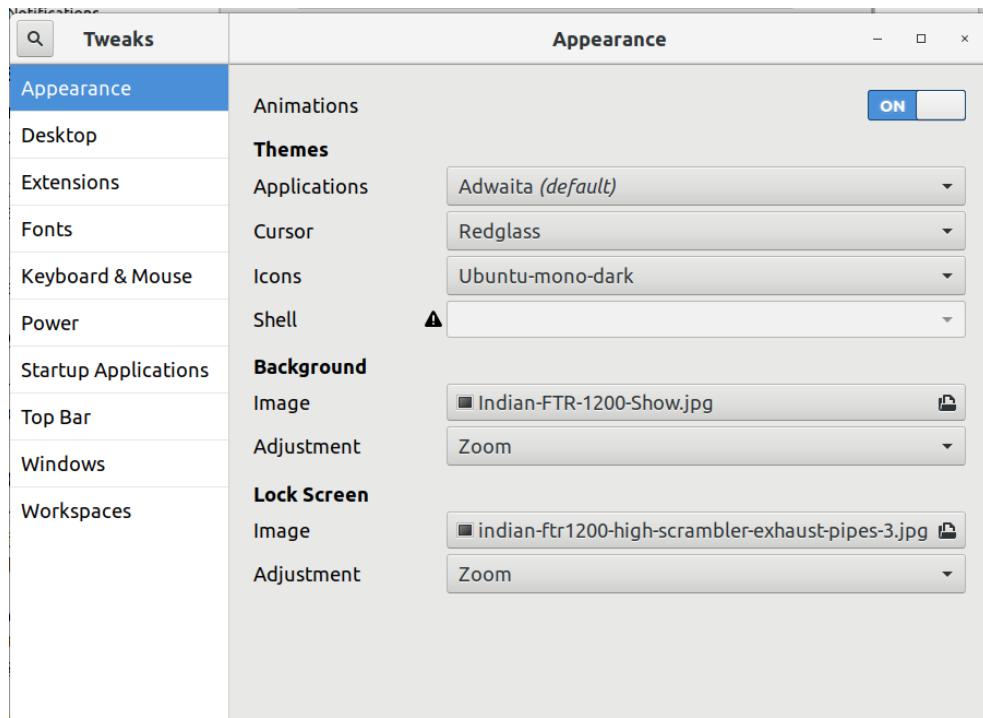
First, install the Gnome Tweak Tool. This tool allows you to change a lot of the Gnome settings to your preference. Open a terminal window, **ctrl + alt + t**, and type the following:

```
sudo apt install gnome-tweak-tool
```

Enter your password and the install will start. When it finishes, hit the super key, type “tweak” and hit enter.

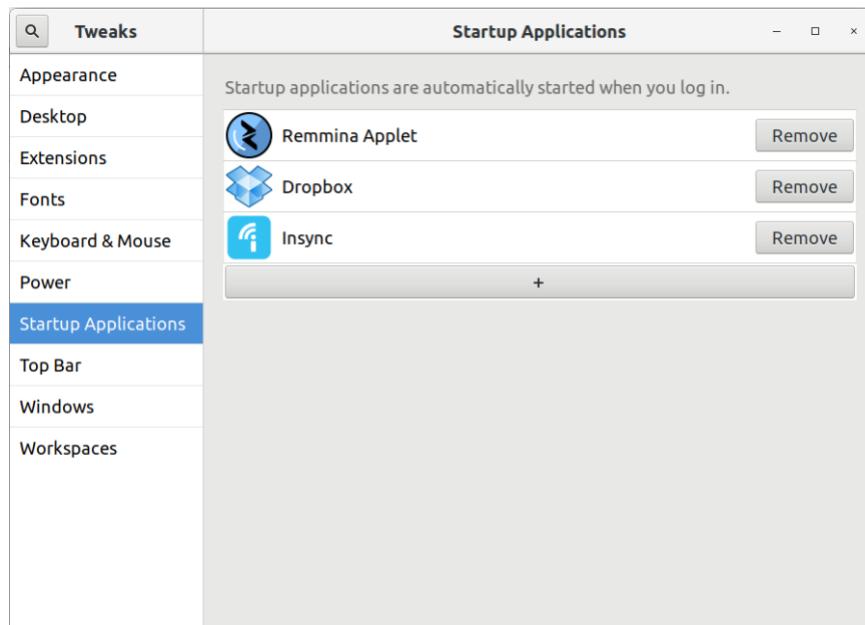


It's very similar to “System Preferences” on the Mac or Control Panel on Windows up to 7.



I won't cover all the categories but you can see that I set a background image, a lock screen image, and a big red cursor.

To set the applications that run on login, use the Startup Applications category.



## Gnome extensions

You can also customize Gnome using extensions. To install extensions, open Chrome or Firefox and go to [Gnome Extensions](#). There are hundreds of extensions available. I have heard that you should limit the number of Gnome extensions to 10 or less due to resource usage but I don't know that for sure. I have booted up and started htop, an application that displays system resource usage and I don't see large amounts of memory or processor in use.

Before you can install Gnome extensions, you have to install a browser plugin. The site will prompt you to install it. Here is link to an article on [\*The 19 best Gnome Extensions\*](#)

Here are the extensions that I have installed:

Disconnect Wifi – by kgshank

Refresh Wifi Connections – by kgshank

Extension Update Notifier – by franglais125

notifications alert – by hackedbellini

Places Status Indicator – by fmuellner

Suspend Button – by laser\_b

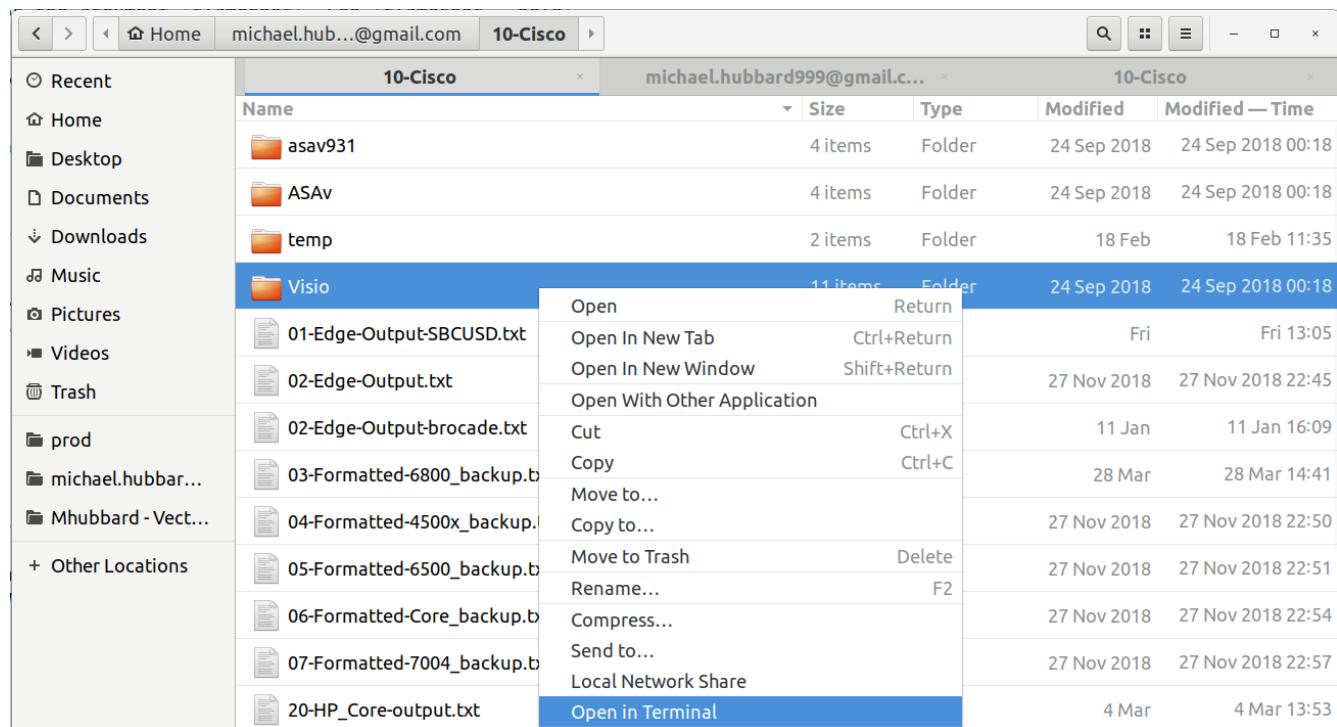
I spoke to the Gnome people at the Southern California Linux Expo 18 (March 2019) and they explained that they don't see the need for any extensions. They prefer to use Gnome work spaces instead of minimizing applications and to use other built in Gnome tools. I tried work spaces and they are very nice.

I had a problem when running VMware workstation in full screen mode. I'm sure there is a work around and I will get back to trying work spaces at some point, but for now I'm using the extensions.

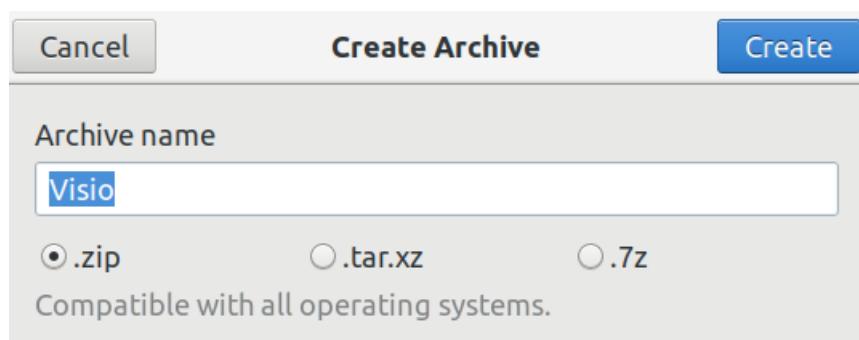
## Managing Files

Nautilus is the file manager for Gnome. It works well and has a feature I love – Tabs! Once you get used to having tabs in your file manager it's hard to use Windows Explorer! Here is a screenshot with three tabs open and the right click menu showing “Open in Terminal”. You can see how convenient this is. On a Mac, the Finder app has the tabs. I don't know why Windows doesn't.

Nautilus has an extensive right click menu. Here are the options for the folder Visio.

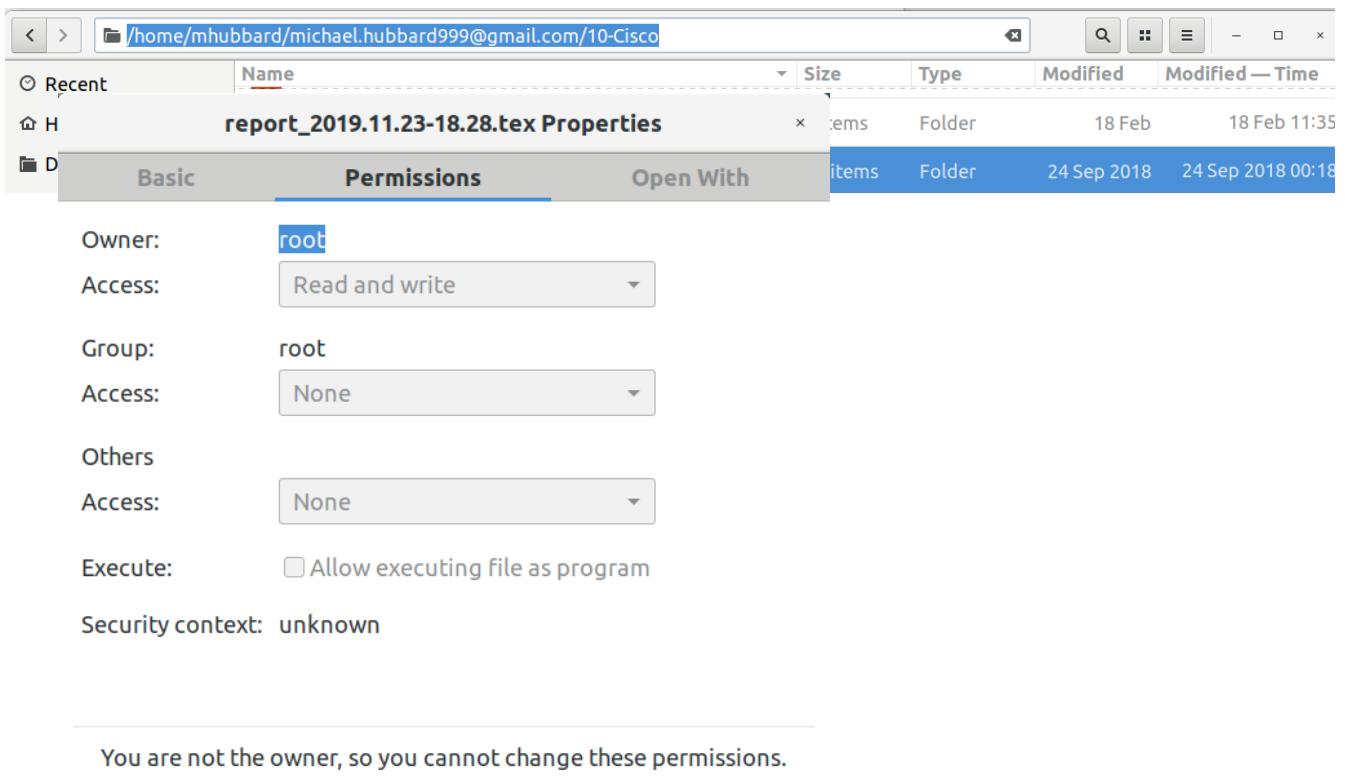


If you click “Compress...” a dialog opens and presents the available applications for compression:



### ***Display the Full path in Nautilus***

Sometimes you want to copy the full path to a file when working in the file manager. Hit **ctrl+l** (lowercase el) and the path will change to a format that you can copy.



### ***Easily preview files in Nautilus***

One thing I missed from Windows Explorer was the preview pane. In Ubuntu running the Gnome desktop you can install "Sushi" to preview files.

[\*How to quickly preview a file in Ubuntu's file manager like quick look in MACOS\*](#)

Then you just hit the spacebar to preview a supported file format.

### ***Why do I have a red X on a file or folder?***

Nautilus, like every other GUI file manager, uses icons to represent files and folders. If the icon has a green check mark in it, you have full access to the file or folder. If the icon has a red X in it, you don't have full permissions to the file or folder.

Here is a screenshot of Nautilus showing one file with a green check mark and one with a red x.

The file with the red x is owned by root and no other user has access to it.

To take ownership of the file:

```
sudo chown mhubbard report_2019.11.23-18.28.tex
```

## **Reference**

Meaning of files-folders with a red x

## **Creating a bootable USB stick from an ISO image**

There are several ways to accomplish this. If you are creating a live USB to try an Ubuntu distro you can use the built in “Startup disk creator” by tapping the super key (Windows Key) and typing software. You will see the Startup disk icon:



Pick the ISO image from the “Source disc image (.iso)” drop down, then select the USB stick, click “Make Startup Disk”. After the image is complete, Ubuntu will use QEMU to test the disc.



You can also use Balena Etcher. The advantage to Etcher is that you can burn any ISO image, not just an Ubuntu based image. To install Etcher, google etcher, from the home page download the .deb file and double click it.

After you burn a disc with Etcher you can use QEMU to test the disc by running:

```
qemu-system-x86_64 -hda /dev/sdx
```

Replace x with the letter for your USB stick.

### ***Reference***

#### ***Using QEMU to test an ISO or Bootable USB***

## Working with the Linux File System

If you have been a Windows user for a long time the hardest part of switching to Linux is the file system. There is no concept of drive letters in Linux. The Linux file system is based on the “File system Hierarchy Standard” maintained by the Linux Foundation.

The top of the Linux file system is called the root. All files and directories are referenced from the root, even if they are stored on different physical or virtual devices. That is so different from Windows that it will take some time to get comfortable with.

Here is a link to a great tutorial on the Linux file system by Abhishek Prakash - [Linux Directory Structure Explained for Beginners](#). His tutorial will get you up to speed on the Linux file system. Abhishek creates Linux tutorials and I recommend that you sign up for his newsletter. The subscribe button is at the top of the page. On Windows you have drive letters and the root of the file system is the “\” character – The good old C:\. But Linux/Mac, or any \*NIX for that matter, doesn’t use drive letters. The file system is usually described as a tree with root at the beginning. Root is shown in the file system as /. Everything is then displayed off the root.

The “/” is another difference that will take a while to get used to. Unix/Linux/Mac use a forward slash instead of the backslash “\” for file system commands. When you use a web browser, you use the forward slash for file commands – <https://github.com/rikosintie> for example.

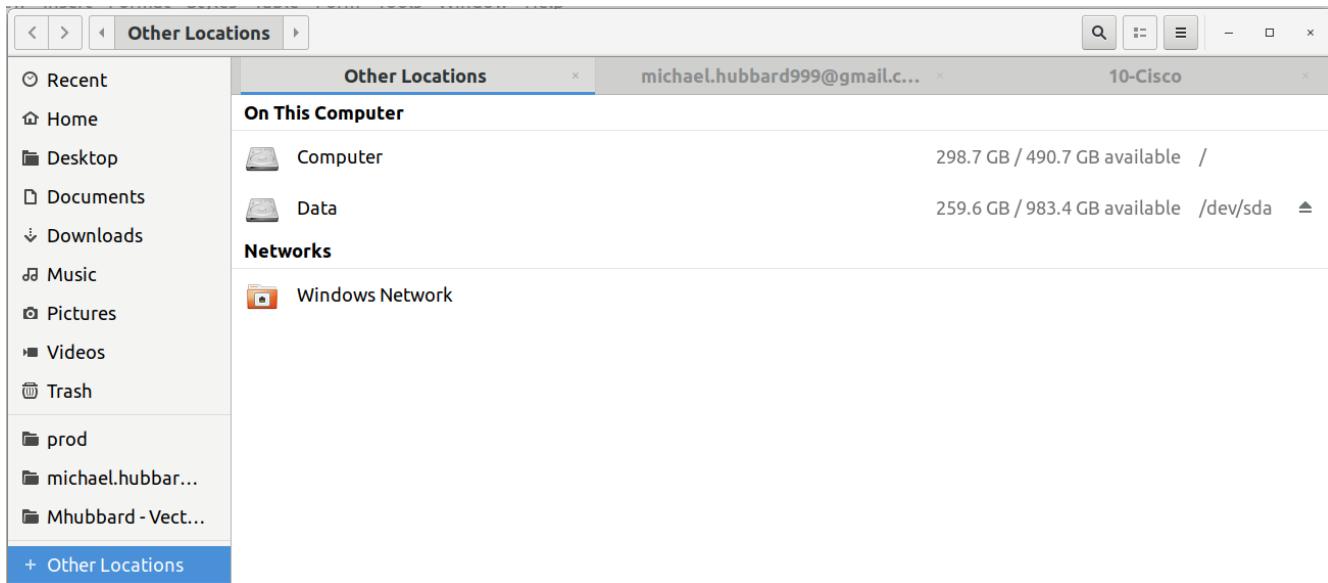
For a quick look at block devices (hard drives, thumb drives, nvme, etc.) on the system, you can use the lsblk command. Here is an lsblk listing on my Dell laptop which has an m.2 NVME drive with the OS on it and 1TB SSD for data.

```
lsblk
NAME      MAJ:MIN   RM    SIZE      RO   TYPE MOUNTPOINT
sda       8:0        0   931.5G  0     disk /media/mhubbard/Data
nvme0n1    259:0     0   465.8G  0     disk
└─nvme0n1p1 259:1     0    513.1M 0     part /boot/efi
└─nvme0n1p2 259:2     0    465.3G 0     part /
```

You can see the nvme drive has two partitions. P1 is the boot partition and p2 has the operating system. Notice that *nvme0n1p2* is type partition and mount is /. That means *nvme0n1p2* is the root of the file system.

The 1 TB SSD is named sda in the operating system and is mounted at /media/mhubbard off of the root. Looking at Nautilus you can see “Computer” which is the OS partition and “Data” which is what I named the SSD.

The display needs a little explanation. For the disk labeled “Computer” it says 298.7 GB / 490.7 GB available. That actually means it’s a 490.7GB drive with 298.7GB available.



Once you get used to it, this seems more intuitive and easier to scale than the drive letter model used by Windows. You will also notice in the lsblk output that Linux/Mac, and again all \*NIX systems, use the forward slash as a delimiter rather than the backslash.

Linux has many “ls” commands for listing things:

- lsblk – list block level devices like disk drives, thumb drives, etc.
- lspci – list the PCI bus devices on the system
- lsusb – list USB bus devices
- lsof – list open files
- lslogins – list logins
- lsmod – list the status of modules inserted into the kernel

I will cover each of these a later in this section.

### **lsusb**

Linux makes it easy to see what USB devices are connected, who the manufacturer is and what the Product ID (PID) and Vendor ID (VID) are.

In this example, I have a USB to Serial adapter connected. It uses the Future Technology Devices International (FTDI) UART. It's connected to Bus 001 and the Vendor ID is 0403, Product ID is 6001.

### **lsusb**

Bus 002 Device 001: ID 1d6b:0003 Linux Foundation 3.0 root hub

Bus 001 Device 002: ID 0c45:6a08 Microdia

Bus 001 Device 003: ID 8087:0aaa Intel Corp.

Bus 001 Device 006: ID 0403:6001 Future Technology Devices International, Ltd FT232 USB-Serial (UART) IC

Bus 001 Device 001: ID 1d6b:0002 Linux Foundation 2.0 root hub

This is a very useful command once you make it part of your skill set. Anytime you connect a USB device you can quickly see who the manufacturer of the chipset is. Especially useful if the device isn't working correctly and want to Google for some information.

Another command that will help here is dmesg. This displays the system messages that have been logged. With the lsusb command above you can see a Future Technology USB-Serial adapter has been inserted.

If I run:

```
dmesg | grep FT232  
[83003.234941] usb 1-3: Product: FT232R USB UART  
[83003.242719] usb 1-3: Detected FT232RL  
[106493.653320] usb 1-3: Detected FT232RL
```

You can see that the system logged the insertion of the UART.

To see what kernel module is loaded for the UART (Note – I am only showing the UART module. Other USB kernel modules were displayed):

```
lsmod | grep usb  
usbserial           49152  3 ftdi_sio
```

In this example the Future Technology USB-Serial adapter is working correctly but you would do the same things to troubleshoot a device that wasn't working.

The big difference over Windows is that support for a device in Linux is built into the kernel and loads when the device is inserted. There are cases where there is no support in the kernel and you will have to go to the manufacturer's site and download a kernel module (driver in Windows). The steps above will quickly let you know that the device isn't being discovered by Linux.

## SSH

\*nix systems have SSH installed by default. Newer versions of the OpenSSH client don't allow weak ciphers.

The OpenSSH client allows you to create SSH keys. The current recommended cipher is Bruce Schnierers ED25519. To create a set of keys using ed25519 run the following in the terminal from the `~/.ssh` directory:

```
ssh-keygen -o -a 100 -t ed25519
```

-o Use the new RFC4716 key format and the use of a modern key derivation function powered by bcrypt.  
-a 100 Use 100 rounds of pbkdf2 (password based key derivation 2)

Specify a strong passphrase when prompted. The passphrase is required anytime you use the key. If you don't password protect the key, and an attacker gets access to the keys, they can log into any server you used them on.

Check the existing keys on your system

```
for keyfile in ~/.ssh/id_*; do ssh-keygen -l -f "${keyfile}"; done | uniq
```

- DSA or RSA 1024 bits: red flag. Unsafe.

- RSA 2048: yellow recommended to change
- RSA 3072/4096: great, but Ed25519 has some benefits!
- ECDSA: depends. Recommended to change
- Ed25519: wow cool, but are you brute-force safe?

Here is what it looked like on my laptop. Looks Like I have some key generation to do!

```
for keyfile in ~/.ssh/id_*; do ssh-keygen -l -f "${keyfile}"; done | uniq
4096 SHA256:569SKPB/kLsw1DqV1SV4J+bE+NYUDfY/LHNbvjvRs+o mhubbard@vectorusa.com (RSA)
3072 SHA256:6e1qwmoi1pbXGQ7hb+GUoPhMxcpeMzTmgg30V0pMDmM mhubbard@HP8600-4.local (RSA)
```

#### Reference

<https://blog.g3rt.nl/upgrade-your-ssh-keys.html>

## Working with services

Gnome includes a tool like “service.msc” on Windows. You start it by tapping the super key and typing “System.”

Ubuntu uses a service called systemd to start and control services. At first it seems confusing and a little difficult compared to just starting “services.msc” on Windows. But like most things, after you do it a few times it’s very logical and easy to use. Here is a cheat sheet for systemd.

<b>Viewing Systemd Information</b>	
systemctl list-dependencies	Show a unit's dependencies)
systemctl list-sockets	List sockets and what activates
systemctl list-jobs	View active systemd jobs
systemctl list-unit-files	See unit files and their states
systemctl list-units	Show if units are loaded/active
systemctl get - default	List default target (like run level)
<b>Working with Services</b>	
systemctl stop service	Stop a running service
systemctl start service	Start a service
systemctl restart service	Restart a running service
systemctl reload service	See if service is running/enabled
systemctl status service	Enable a service to start on boot
systemctl enable service	Disable a service - won't start at boot
systemctl disable service	Show properties of a service (or other unit)
systemctl -H host status network	Run any systemctl command remomtely
<b>Changing System states</b>	
systemctl reboot	Reboot the system (reboot.target)
systemctl poweroff	Power off the system (poweroff.target0)
systemctl emergency	Put in emergency mode (emergency.target)
systemctl default	Back to default target (multi-user.target)

<b>Viewing Log Messages</b>	
journalctl	Show all collected log messages
journalctl -u network.services	See network service messages
journalctl -f	Follow messages as they appear
journalctl -k	show only kernel messages
<b>Runlevels to Targets Cheat sheet</b>	
Sysvinit Runlevel	Systemd Target
0	runlevel0.target, poweroff.target
1, s, single	runlevel1.target, rescue.target
2, 4	runlevel2.target, runlevel4.target, multi-user.target
3	runlevel3.target, multi-user.target
5	runlevel5.target, graphical.target
6	runlevel6.target, reboot.target
emergency	emergency.target

### ***Locating files from the terminal***

The locate tool allows you to search everywhere on the file system.

`sudo apt install mlocate` – Install the tool

`sudo updatedb` - Updates the database, run this periodically especially if you modify the configuration file.

To configure locate:

These two articles have detailed information on customizing locate.

- <https://linux.die.net/man/5/updatedb.conf> - configure locate
- <https://www.tecmint.com/linux-locate-command-practical-examples/>

Example:

```
mhubbard@1S1K-G5-5587:~/Dropbox/Python/Scripts/SIET$  
->locate apsum.py  
/home/mhubbard/Dropbox/Python/Scripts/prod/apsum.py  
/home/mhubbard/Dropbox/Python/Scripts/wlc/apsum.py
```

### **Drill - A graphical and CLI tool for searching files**

From the website, “Search files without indexing, but clever crawling. It uses a different algorithm than most search tools and doesn't use a database like the Ubuntu locate utility. It will search every mount point on the computer and is multithreaded so that it can search many at once. It will also use all available RAM to speed up the search.”

In the GUI, just enter any part of the filename that you can remember. It's not case sensitive and will find the string anywhere in the name.

From the website, [Download Drill](#), you can download the GUI (.deb) and the cli (CLI .deb). It is also available as an appimage if you don't want to install it.

To use the cli version:

`drill-search-cli "filename"`

#### **Example**

```
->drill-search-cli hs-hp  
/home/mhubbard/Downloads/843dc19f-1d21-461b-b126-b46739e2c4d9_HS-HP2600-8-  
LA54_20190321_121957.txt  
/home/mhubbard/Downloads/a77fc3cc-bbdf-4353-8bdf-c0bcba6f148f_HS-HP2600-8-  
LA66_20190322_081901.txt  
/home/mhubbard/Downloads/f595cff7-45b4-4d80-95bd-d8c8e5d5bf18_HS-HP2600-8-  
LA50_20191031_153810.txt
```

## **Working with the Terminal**

Like almost everything else in Linux, you can choose the shell and terminal application you want to use. Ubuntu ships with the Borne Again Shell or BASH, the same as a Mac. When you start the terminal by pressing the super key and typing terminal, you are actually starting the Gnome terminal running BASH as the shell.

I use a terminal emulator called Terminator. It allows you to add new tabs, split screens, log output and do several other useful things. I cover the installation of Terminator in the “Tools Installed from the terminal” section later in the book.

In addition to BASH there are many other shells available – the Fish Shell, Korn Shell, Z Shell, C Shell, etc. I have found BASH to be fine and I like that it is the shell shipped in most distributions so I have stuck with it.

To open a terminal you can press **ctrl+alt+t**. This will open whatever terminal emulator you have set as the default. See “Enhanced Terminals” late in the book for instructions on setting the default.

Here are some tips for working in BASH.

### **Cut/Paste**

Cutting and pasting text in BASH is slightly different than in Windows.

To copy press, **Ctrl+Shift+C**

To paste, press **Ctrl+Shift+V**

The reason for the shift key is that BASH uses **Ctrl+C** to terminate running processes.

### **Manual Pages**

Most Linux terminal commands include “Manual Pages” that give details about the command. To access the Man pages you type “**man <command name>**”. This is where the Linux phrase “*RTFM*” or *Read the fine manual* comes from.

For example,

**man pwd**

The man page opens in the terminal and you press **<enter>** to move down the page. You can press “**e**” to advance one line at a time or “**y**” to backup one line at a time. You can enter a “**\**” and a pattern to search the man page. Once you start a search you can press “**n**” to find the next occurrence or “**N**” to find the previous occurrence.

If you forget the shortcuts you can type **help <enter>** and the help menu will be displayed.

### **Tab Completion**

Bash supports tab completion. This is similar to tab completion on a switch or router. Once you have typed enough to be unique you can press tab to complete the line.

[explainshell.com](http://explainshell.com)

This is a website that has most BASH commands in a much easier format. I recommend trying out explainshell.com when you need to learn how a command works.

## **Virtual Console**

You can also open the shell without a terminal emulator. You would normally do this if something went seriously wrong and you couldn't log in or some other issue occurred.

To open the shell in a *virtual console* enter Ctrl+Alt+F2. There are six virtual consoles – F1, F2, F3, F4, F5, F6. I have found that I usually have to hit Ctrl+Alt+F4 to get a virtual console.

The desktop will disappear and you will be placed into the virtual console with BASH running.

To switch back to the desktop enter Ctrl+Alt+F7. I have found that sometimes I have to hit Ctrl+Alt+F1 and log back into the GUI to return to the desktop. I'm not sure why and I have only used a virtual console once when I broke the login page so I haven't researched it.

### **Note:**

Unlike Windows, Linux/Mac/Unix are case sensitive operating systems. This will take some getting used to because Text.txt is not the same thing as text.txt and the command “cd” will not work if you type Cd or CD!

The Windows PowerShell command line has most of the Linux terminal commands as aliases, if you use it as your Windows command line you can use the same commands on Windows and Linux.

## **History**

BASH keeps a history file of every command you have typed. By default, the file is `~/.bash_history`. If the file isn't there you can use “`echo $HISTFILE`” to locate it. To edit the file use “`gedit ~/.bash_history`”. The file is updated when you close the terminal, so you won't see new entries in the file until you close the terminal.

To view the commands you have typed, enter “`history`” into the terminal. This prints all the commands in the history file plus the commands in the current shell that haven't been written to the file. There will be a number to the left of the command. For example:

```
->history
1986 sudo linssid
1987 sipcalc 10.0.246.0/30
1988 sipcalc 10.0.246.4/30
1989 sipcalc 10.0.246.7/30
1990 sipcalc 10.0.246.8/30
1991 sipcalc 10.0.246.12/30
1992 sipcalc 10.0.246.16/30
1993 sipcalc 10.0.246.20/30
```

To reuse the command with 1989 next to it use:

```
->!1989
sipcalc 10.0.246.7/30
-[ipv4 : 10.0.246.7/30] - 0
```

[CIDR]

Host address	- 10.0.246.7
Host address (decimal)	- 167835143
Host address (hex)	- A00F607
Network address	- 10.0.246.4
Network mask	- 255.255.255.252
Network mask (bits)	- 30
Network mask (hex)	- FFFFFFFC
Broadcast address	- 10.0.246.7
Cisco wildcard	- 0.0.0.3
Addresses in network	- 4
Network range	- 10.0.246.4 - 10.0.246.7
Usable range	- 10.0.246.5 - 10.0.246.6

You can use “**history | less**” to page through the history.

### **Clear the history file**

```
history -c
```

### **Where is the history file located?**

```
echo "$HISTFILE"  
/home/mhubbard/.bash_history
```

### **View history size**

```
echo "$HISTSIZE"  
50000
```

### **Set history size**

```
export HISTSIZE=50000
```

### **Save a record of how you built the system**

One use for the history file is to back up a record of the commands used to build the system. To do this, install Ubuntu, install all the applications you need from the terminal, close the terminal and copy  
~/.bash\_history to a safe location.

The advantage of the history file is that when you open a new terminal you can use the up arrow and replay commands that you have used before. Once you get used to this it's a great feature. In windows you lose the history when you close the command windows.

### **Reference**

<https://www.cyberciti.biz/faq/clear-the-shell-history-in-ubuntu-linux/>

### ***Incremental history searching - An extremely handy tool***

See this page from the “Code in the Hole” blog for more detail: [The most important command line tip  
incremental history searching with inputrc](#)

This is the greatest time saver imaginable! It reads the bash history and lets you search it using the first couple letters of the command and then the up arrow.

To configure it, in terminal enter:  
gedit ~/.inputrc

Then copy paste and save:  
"\e[A": history-search-backward  
"\e[B": history-search-forward  
"\e[C": forward-char  
"\e[D": backward-char

One extra thing: this functionality can be neatly complimented by some choice history settings in your  
~/.bashrc file:

```
export HISTSIZE=1000000
export HISTFILESIZE=1000000000
shopt -s histappend
```

The “shopt -s histappend” appends the history rather than having it over.write

From the “code in the hole” website:

All you need to do to find a previous command is to enter say the first two or three letters and upward arrow will take you there quickly:

Say I want:

```
for f in *.mid ; do timidity "$f"; done
```

All I need to do is enter:

fo

And hit the upward arrow. The command will soon appear.

If you hit the up arrow again it will find the next occurrence.

On a Raspberry Pi, I had to create the ~/.inputrc file first, then this tip worked just like on Ubuntu.

## Reference

Power Shell Usage  
[https://www.ukuug.org/events/linux2003/papers/bash\\_tips/](https://www.ukuug.org/events/linux2003/papers/bash_tips/)

## **Auto-jump**

From the site: A cd command that learns - easily navigate directories from the command line

autojump is a faster way to navigate your filesystem. It works by maintaining a database of the directories you use the most from the command line.

*Directories must be visited first before they can be jumped to.*

I recently found this project on github.com. It learns the directories you use and allows you to jump to them by typing the first few letters of the directory. For example, I keep my Fortinet vpn scripts in “/home/mhubbard/michael.hubbard999@gmail.com/01\_Vector/01\_Forticlient-64bit” which is obviously a lot to type when I want to run one of the scripts. But with autojump I just have to enter

```
j fo
```

from anywhere and it jumps me to the right folder.

## **Installation**

autojump is in the Ubuntu repository so we can use apt to install it:

```
sudo apt install autojump
```

After the install finishes you need to edit ~/.bashrc (gedit ~/.bashrc) and add the following to the end of the file:

```
#start autojump - /usr/share/autojump/
./usr/share/autojump/autojump.sh
```

It will take a while before autojump has a lot of your directories memorized but once it does you will save a lot of time navigating the terminal.

## **Reference**

<https://www.tecmint.com/autojump-navigate-linux-directories-faster/>

<https://github.com/wting/autojump>

## **bd**

### Description

Quickly go back to a specific parent directory in bash instead of typing "cd ../../.." redundantly.

### Installation

```
sudo apt install bd
```

### Example:

I'm currently in

~/Insync/michael.hubbard999@gmail.com/GoogleDrive/Python/Scripts/arp-CX on □ main!

And want to go back to Python

bd Py

/home/mhubbard/Insync/michael.hubbard999@gmail.com/GoogleDrive/Python/

To go back to arp-CX

j arp-C

/home/mhubbard/Insync/michael.hubbard999@gmail.com/GoogleDrive/Python/Scripts/arp-CX

Combined with auto jump, you will save a lot of time navigating your file system.

### From the project Git Repo

Other uses:

Using bd within backticks (`bd <letter(s)>`) prints out the path without changing the current directory.

You can take advantage of that by combining `bd <letter(s)>` with other commands such as ls, ln, echo, zip, tar etc..

Example:

If you just want to list the contents of a parent directory, without going there, then you can use: ls `bd p` in the given example, it will list the contents of /home/user/project/

If you want to execute a file somewhere in a parent directory, `bd p`/build.sh will execute /home/user/project/build.sh while not changing the current directory.

If you reside in /home/user/project/src/org/main/site/utils/file/reader/whatever and want to change to /home/user/project/test, then try cd `bd p`/test

### Reference

<https://github.com/vigneshwaranr/bd>

<https://www.linux-magazine.com/Issues/2018/207/bd-autojump-and-Fasd>

## Package management

Ubuntu has three main methods to install software:

- The software store – A graphical tool where you can search for programs and install them by double clicking. The software store is discussed later in this guide.
- The Aptitude package manager (apt) – a CLI tool that is almost as old as Linux itself.
- The Snap Store – Canonical's proprietary store for snaps. Discussed later in this guide.

Of course, there are a lot of times that you will clone a git repo and build the software, but that is not covered in this guide.

There is a new tool that you can use to install packages from the terminal that is a little easier and faster than using apt:

## Nala package manager

I found this tool in June, 2023. It's one of those things where I didn't know APT was so bad (joking) until I found Nala! I do recommend installing Nala and reviewing the gitlab repo for more information.

### Description

Nala is a front-end for libapt-pkg. Specifically we interface using the python-apt api.

Especially for newer users it can be hard to understand what apt is trying to do when installing or upgrading.

We aim to solve this by not showing some redundant messages, formatting the packages better, and using color to show specifically what will happen with a package during install, removal, or an upgrade.

### Benefits of Nala over Apt

#### Parallel Downloads

Outside of pretty formatting, the number 1 reason to use Nala over apt is parallel downloads. By default, we will download 3 packages per unique mirror in your sources.list file. Opening multiple connections to the same mirror is great for speeding up downloading many small packages. We have the 3 connections per mirror limit to minimize how hard we are hitting mirrors. Additionally, we alternate downloads between the available mirrors to improve download speeds even further. If a mirror fails for whatever reason, we just try the next until all defined mirrors are exhausted.

Note: Nala does not use APT for package downloading and verification

### Fetch

sudo nala fetch

Which brings us to our next standout feature, nala fetch. This command works like most people use netselect and netselect-apt. Nala fetch will check if your distro is either Debian or Ubuntu. Nala will then go get all the mirrors from the respective master list.

Once done we test the latency and score each mirror.

Nala will choose the fastest 3 mirrors (configurable) and write them to a file.

Note: At the moment fetch will only work on Debian, Ubuntu and derivatives still tied to the main repos. Such as Pop!\_OS

## **History**

nala history

Our last big feature is the nala history command.

If you're familiar with dnf this works much in the same way.

Each Install, Remove or Upgrade we store in /var/lib/nala/history.json with a unique <ID> number.

At any time you can call nala history to print a summary of every transaction ever made.

You can then further manipulate this with commands such as nala history undo <ID> or nala history redo <ID>.

If there is something in the history file that you don't want you can use the nala history clear <ID> It will remove that entry.

Alternatively for the clear command we accept --all which will remove the entire history.

Installation

sudo apt update

sudo apt install nala

## **References**

<https://gitlab.com/volian/nala>

<https://phoenixnap.com/kb/nala-apt>

## Display a new line on the prompt in terminal

<https://askubuntu.com/questions/16424/displaying-a-new-line-on-the-prompt>

I like to have a new line inserted after the prompt so that my commands are below the prompt. To do that you just need to edit your .bashrc file and add PS1="\$PS1\n->" to the bottom of the file:

```
gedit ~/.bashrc  
PS1="$PS1\n->"  
save and exit
```

To reload the file:  
exec "\$BASH"

Example:

```
mhubbard@1S1K-G5-5587:/usr/share/nmap/scripts/vulscan/utilities/updater$  
->sudo nmap -sV -sC 192.168.10.181
```

**Note: I now use zsh and the amuse theme which sets a new line and show git status.**

# Installing and using zsh instead of bash

Bash is the default shell on Ubuntu. Bash, the Bourne Again Shell, has been around since the 1990s and works great. But there is another shell, zsh, the I have switched to. The big reason is that there is an add on called “How My ZSH” which is an unbelievably active community that creates themes and plugins for zsh.

Some zsh Features

- Command-line completion.
- History can be shared among all shells.
- Extended file globbing.
- Better variable and array handling.
- Compatibility with shells like bourne shell.
- Spelling correction and autofill of command names.
- Named directories.

<https://www.tecmint.com/install-zsh-in-ubuntu/>

```
sudo apt update  
sudo apt upgrade  
sudo apt install zsh
```

Check the version  
zsh --version

zsh 5.8 (x86\_64-ubuntu-linux-gnu)

This is the current version as of May 2021

You can check your current shell using the echo command:  
echo \$SHELL  
/bin/bash

You can see that the current shell is bash

Run this to make zsh the default  
chsh -s \$(which zsh)

Verify  
grep zsh /etc/passwd  
mhubbard:x:1000:1000:Michael Hubbard,,,:/home/mhubbard:/usr/bin/zsh

Log out of the current bash session and restart terminal to use zsh

exit

Reopen a terminal with ctrl+alt+t  
You will get a message saying zsh needs some configuration:

This is the Z Shell configuration function for new users,  
zsh-newuser-install.

You are seeing this message because you have no zsh startup files  
(the files .zshenv, .zprofile, .zshrc, .zlogin in the directory  
~). This function can help you with a few settings that should  
make your use of the shell easier.

You can:

- (q) Quit and do nothing. The function will be run again next time.
- (0) Exit, creating the file ~/.zshrc containing just a comment.  
That will prevent this function being run again.
- (1) Continue to the main menu.
- (2) Populate your ~/.zshrc with the configuration recommended  
by the system administrator and exit (you will need to edit  
the file by hand, if so desired).

--- Type one of the keys in parentheses --- 0  
1S1K-G5-5587%

I chose 0 to just create the .zshrc file and exit because I am going to install a tool called “Oh My ZSH” to customize the shell.

## Install Oh My zsh

From <https://ohmyz.sh/#install>

```
sh -c "$(curl -fsSL https://raw.githubusercontent.com/ohmyzsh/ohmyzsh/master/tools/install.sh)"
```

When the install shell starts you will see:

```
Found ~/.zshrc. Backing up to /home/mhubbard/.zshrc.pre-oh-my-zsh
Using the Oh My Zsh template file and adding it to ~/.zshrc.
```

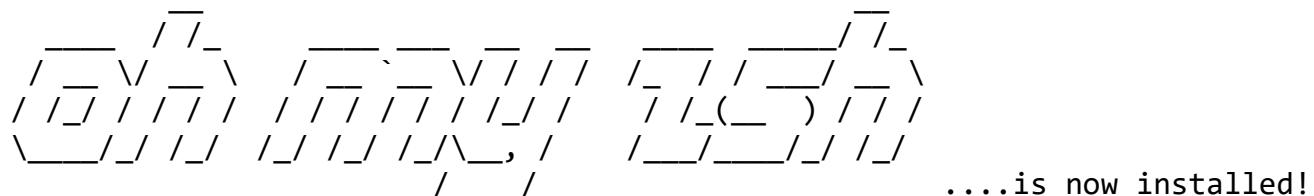
Time to change your default shell to zsh:

Do you want to change your default shell to zsh? [Y/n] y

Changing the shell...

Password:

Shell successfully changed to '/usr/bin/zsh'.



Before you scream Oh My Zsh! please look over the `~/.zshrc` file to select plugins, themes, and options.

- Follow us on Twitter: <https://twitter.com/ohmyzsh>
- Join our Discord server: <https://discord.gg/ohmyzsh>
- Get stickers, shirts, coffee mugs and other swag: <https://shop.planetargon.com/collections/oh-my-zsh>

## Add a theme, some plugins and aliases

Open the `.zshrc` file

gedit `~/.zshrc`

Find the plugin line and change it to

```
plugins=(  
    git  
    zsh-completions  
    zsh-autosuggestions  
    zsh-syntax-highlighting  
    history-substring-search  
    colored-man-pages  
    aliases  
    zsh-docker-aliases  
)
```

Download the plug ins (Autojump should already be installed if you have been following the book. If not follow the instructions in the previous section to install autojump.

```
git clone https://github.com/zsh-users/zsh-completions ${ZSH_CUSTOM:=$HOME/.oh-my-zsh/custom}/plugins/zsh-completions  
git clone https://github.com/zsh-users/zsh-syntax-highlighting.git ${ZSH_CUSTOM:=$HOME/.oh-my-zsh/custom}/plugins/zsh-syntax-highlighting  
git clone https://github.com/zsh-users/zsh-autosuggestions ${ZSH_CUSTOM:=$HOME/.oh-my-zsh/custom}/plugins/zsh-autosuggestions
```

Oh My ZSH offers a lot of themes. I found one that I really like called duellj. To install it change the `ZSH-THEME` line to:

```
ZSH_THEME="duellj"
```

I also like “amuse”. It’s similar to duellj but doesn’t put the username/machine name in the terminal. Since I’m on my personal laptop I don’t need that information. To use “amuse”

```
ZSH_THEME="amuse"
```

<https://github.com/ohmyzsh/ohmyzsh/wiki/Themes>

# Bat - A cat clone with syntax highlighting and Git integration

This is a great upgrade to cat. The automatic paging, syntax highlighting, Git integration and the ability to show non-printable characters makes replacing cat with bat a no brainer.

There are a lot of other features to bat. You should review the official Git repository at <https://github.com/sharkdp/bat>

## Syntax highlighting

bat supports syntax highlighting for a large number of programming and markup languages:

```
▶ bat test.md
File test.md
1 # Markdown example
2
3 Note how we can correctly syntax-highlight the
4 code blocks *inside* the Markdown document.
5
6 Python example:
7
8     ``` python
9     cmd = "bat"
10    print("Hello from {}".format(cmd))
11
12 Ruby example:
13
14     ``` ruby
15     cmd = "bat"
16     puts "Hello from #{cmd}"
17
18
```

## Git integration

bat communicates with git to show modifications with respect to the index (see left side bar):

```
▶ bat src/index.html
File src/index.html
1 <!DOCTYPE html>
2 <html lang="en">
3   <head>
4     <meta charset="utf-8">
5     <title>a changed title</title>
6     <link rel="stylesheet" href="style.css">
7   </head>
8   <body>
9     New lines that have been
10    added since last commit.
11   </body>
12 </html>
```

## Show non-printable characters

You can use the -A/--show-all option to show and highlight non-printable characters:

```
▶ bat -A non-printable.txt
File: non-printable.txt
1 .....space-indented•line•LF
2 | | |tab-indented•line•LF
3 | | |.....mixed-indented•line•LF
4 LF
5 backspace•character•in•quotes:•"•BS"•LF
6 escape•character•in•quotes:•"•ESC"•LF
7 LF
8 line•with•trailing•space••••LF
9 LF
10 line•that•ends•with•carriage•return•CR•LF
11 last•line•LF
```

## Installation

### On Ubuntu (using most recent .deb packages)

... and other Debian-based Linux distributions.

Download the latest .deb package from the [release page](https://github.com/sharkdp/bat/releases) (<https://github.com/sharkdp/bat/releases>) and install it via:

```
sudo dpkg -i bat-musl_0.23.0_amd64.deb # adapt version number and architecture
```

**Important:** If you install bat this way, please note that the executable may be installed as batcat instead of bat (due to a name clash with another package). You can set up a bat -> batcat symlink or alias to prevent any issues that may come up because of this and to be consistent with other distributions:

```
mkdir -p ~/.local/bin
ln -s /usr/bin/batcat ~/.local/bin/bat
```

If you want to use an alias instead of a symlink, add

```
alias cat='batcat'
```

to ~/.zshrc

## Enhanced Terminals

There are several tiling terminal emulators available for Ubuntu. Tiling means that the terminal can be carved up into several "tiles" each of which can be in a different directory and running different programs. Since a network engineer spends a lot of time in the terminal, a tiling emulator will improve your productivity. I use Terminator, here is a link to an article on four others - <https://www.rootusers.com/five-best-terminal-emulators-linux/>

**Terminator** - An enhanced terminal that supports logging, split windows and much more. The logging feature is really nice, as you start a new task in the terminal, just right click and select logging. You are prompted for a filename and everything done in the terminal is logged, not just the keystrokes you type. It's great for capturing nmap scans or other terminal apps.

```
sudo apt install terminator -y
```

Detailed description of Terminator  
[Terminator Linux terminal emulator](#)

Some useful shortcuts in Terminator

ctrl + shift + "+" - increase text size

ctrl + "-" - decrease text size

ctrl + 0 - reset text size to default

Ctrl+Shift+G - reset terminal state and clear window

Ctrl+PageDown - Move to next Tab.

Ctrl+PageUp - Move to previous Tab.

Ctrl+Shift+Right - Move parent dragbar Right.

Ctrl+Shift+Left - Move parent dragbar Left.

Ctrl+Shift+Up - Move parent dragbar Up.

Ctrl+Shift+Down - Move parent dragbar Down.

*Pick the default terminal program*

```
sudo update-alternatives --config x-terminal-emulator
```

Then select your choice from the menu

Example:

```
sudo update-alternatives --config x-terminal-emulator
```

[sudo] password for mhubbard:

There are 2 choices for the alternative x-terminal-emulator (providing /usr/bin/x-terminal-emulator).

Selection	Path	Priority	Status
<hr/>			
* 0	/usr/bin/terminator	50	auto mode
1	/usr/bin/gnome-terminal.wrapper	40	manual mode
2	/usr/bin/terminator	50	manual mode

Press <enter> to keep the current choice[\*], or type selection number: 0

# Bash Commands

Press “ctrl+alt+t” to open the terminal. Once the terminal is open, you can use the following commands for common operations.

~ – the tilde represents your home directory in Linux. You can **cd ~** from any level to return to the home directory. For example:

```
mhubbard@1S1K-G5-5587:~/Dropbox/Python/Scripts/prod$
```

```
->cd ~
```

```
mhubbard@1S1K-G5-5587:~$
```

Using **pwd** to “print the working directory” you will see that I am now at */home/mhubbard*

```
->pwd
```

```
/home/mhubbard
```

```
mhubbard@1S1K-G5-5587:~$
```

**cat** – print the contents of text files to the terminal.

From the man page

NAME

cat - concatenate files and print on the standard output

SYNOPSIS

cat [OPTION]... [FILE]...

DESCRIPTION

Concatenate FILE(s) to standard output.

## Example 1

Print the contents of a text file.

```
mhubbard@1S1K-G5-5587:~/Dropbox/Python/Scripts/prod$
```

```
->cat vlans.txt
```

```
ip address 10.140.1.125 255.255.255.128
ip address 10.140.0.254 255.255.255.0
ip address 10.140.39.254 255.255.248.0
ip address 10.140.47.254 255.255.254.0
ip address 10.141.15.254 255.255.248.0
ip address 10.141.7.254 255.255.248.0
ip address 10.140.45.254 255.255.254.0
ip address 10.141.23.254 255.255.248.0
ip address 172.16.1.254 255.255.255.0
ip address 172.16.202.254 255.255.255.0
ip address 10.140.1.254 255.255.255.128
# ip address 10.100.15.254 255.255.240.0
```

ip address 10.101.0.254 255.255.255.0

### Example 2

combine test1.txt and test2.txt into test3.txt

```
->echo test file 1 > test1.txt  
->echo test file 2 > test2.txt  
->cat test1.txt test2.txt > test3.txt  
->cat test3.txt  
test file 1  
test file 2
```

### Example 3

Concatenate text into a jpeg file. This is useful if you want to send a short text message in a jpeg file. Great for adding notes to photos.

In this example, I have a jpeg file named airbox.JPG and a text file named steg.txt. I will use cat to combine them into test-steg.jpg.

```
cat airbox.JPG steg.txt > test-steg.jpg
```

Now, using cat to display test-steg.jpg (Note: this is the bottom of the file, the beginning will be the gibberish from the jpg)

```
cat airbox.JPG steg.txt > test-steg.jpg
```

k3 qZ #%. This is an example of hiding text inside a jpg file.

**zcat** – Similar to cat but works on gzip archives. The link below has multiple examples of using zcar.

<https://www.tecmint.com/linux-zcat-command-examples/>

**cd** – the change directory command, same as windows. Just like in Windows, you can use .. to go back one level. You can also use directory traversal – cd ../../Python/Scripts/prod/. In this example, I went up two levels from the current directory and then to the Python/Scripts/prod folder.

```
mhubbard@1S1K-G5-5587:~/Dropbox/03_Tools/dnstwist$  
->cd ../../Python/Scripts/prod/  
mhubbard@1S1K-G5-5587:~/Dropbox/Python/Scripts/prod$
```

Once you get used to directory traversal it is much faster to move around in the terminal.

Note:

Directory Traversal works on Windows also but obviously, you have to use “\” instead of “/”.

**cp** – copy files. Format is cp source destination.

Example of copying a file named test.txt from the current directory to the desktop:

```
cp test.txt Desktop/
```

Example of copying a file from a remote directory to the current directory. The “.” represents the current working directory.

```
cp /home/mhubbard/Dropbox/Python/Scripts/prod/apsum.txt .
```

```
pwd
```

```
/home/mhubbard/tftp-root
```

```
ls -l ap*
```

```
-rw-rw-r-- 1 mhubbard mhubbard 1641 Jun 7 15:25 apsum.txt
```

**echo** – copy text to a file.

### Example

Copy the text “test file 1” to a file named test1.txt

```
echo test file 1 > test1.txt
```

**less** – display text files and use cursor keys to navigate. Press Q to quit. For example:

```
less apsum.txt
```

**ls** – list files, similar to dir on windows. You can add -l so show a long format and -la to show the long format and hidden files. The man page (man ls) will list all the options.

**mkdir** – make directory. Usage is simply mkdir dir-name

**mv** – move or rename a file or directory. To rename a file you simply “move” it to a new name. The format is mv source-file destination-file

For example:

```
mv apsum.txt apsum1.txt
```

```
mhubbard@1S1K-G5-5587:~/tftp-root$
```

```
->ls -l ap*
```

```
-rw-rw-r-- 1 mhubbard mhubbard 1641 Jun 7 15:25 apsum1.txt
```

**pwd** – print working directory. This prints to the screen the directory you are currently in.

**rm** – remove. This is different than the windows del command and will take a little getting used to. To remove one file you just use **rm filename**. To remove a directory that has files in it you must add the recursive and force options -rf. To remove the directory ~/Dropbox/Python/Scripts/prod you would enter:

```
rm -rf ~/Dropbox/Python/Scripts/prod/
```

---

### Note

There is no confirmation when you hit enter. Make sure that the path you entered is the path you wanted. Once enter is pressed the directory will be deleted immediately.

---



## grep

One of the best tools in Linux! You can search a directory or recursively through the file system for a pattern inside a file.

From the man page

### NAME

grep - print lines matching a pattern

### SYNOPSIS

```
grep [OPTIONS] PATTERN [FILE...]
grep [OPTIONS] -e PATTERN ... [FILE...]
grep [OPTIONS] -f FILE ... [FILE...]
```

### DESCRIPTION

grep searches for PATTERN in each FILE. A FILE of “-” stands for standard input. If no FILE is given, recursive searches examine the working directory, and nonrecursive searches read standard input. By default, grep prints the matching lines.

### Example 1

I have backups of several switches in a directory. I wanted to pull out the management IP address for each switch. I knew they all started with 10.56.246 and had a 255.255.255.0 mask.

I used a regular expression (regex) to look for 10.56.246. and any number from 0-9. I then passed the results of that to grep again using the “|” operator to strip off the “ip address”, “ntp server” etc.

Regular expressions are beyond the scope of this book but they are very powerful for searching. A Google search for regex will return many articles on how to use them. I like this one [Computer Hope Regex Examples](#)

In the gedit text editor built into Ubuntu you can use regex expressions for search/replace. The actual names of the files were real site names that were four characters long. I wanted to change the real name to site. I just hit **crtl\_h**, selected regex and entered:

^.... in the find box and site in the replace box. The ^ means “start of line” and a “.” means match one character.

```
mhubbard@1S1K-G5-5587:~/Dropbox/01_test/configs/DataCenter$
```

```
->grep -r 10.56.246.[0-9]
site1-6880x.wri: ip address 10.56.246.178 255.255.255.0
site1-6880x.wri: network 10.56.246.178 0.0.0.0
site1-6880x.wri:ntp server 10.56.246.56 prefer
site2-6880x.wri: ip address 10.56.246.58 255.255.255.0
site2-6880x.wri: network 10.56.246.58 0.0.0.0
site2-6880x.wri:ntp server 10.56.246.56 prefer
```

```
site3-6880x.wri: ip address 10.56.246.229 255.255.255.0
site3-6880x.wri: network 10.56.246.229 0.0.0.0
```

Now with the second grep:

```
mhubbard@1S1K-G5-5587:~/Dropbox/test/configs/DataCenter$
```

```
->grep -r 10.56.246.[0-9] | grep 255.255.255.0
site1-6880x.wri: ip address 10.56.246.178 255.255.255.0
site2-6880x.wri: ip address 10.56.246.58 255.255.255.0
site3-6880x.wri: ip address 10.56.246.229 255.255.255.0
site4-6880xx.wri: ip address 10.56.246.214 255.255.255.0
site5-6880x.wri: ip address 10.56.246.221 255.255.255.0
site6-6880x.wri: ip address 10.56.246.241 255.255.255.0
site7-6880x.wri: ip address 10.56.246.219 255.255.255.0
site8-6880X.wri: ip address 10.56.246.133 255.255.255.0
site9-6880x.wri: ip address 10.56.246.51 255.255.255.0
site10-6880x.wri: ip address 10.56.246.180 255.255.255.0
site11-6880x.wri: ip address 10.56.246.194 255.255.255.0
```

If I just want the IP address, I can add the “awk” command to print only the 4th item on the line:

```
grep -r 10.56.246.[0-9] | grep 255.255.255.0 | awk '{ print $4 }'
10.56.246.178
10.56.246.58
10.56.246.229
10.56.246.214
10.56.246.221
10.56.246.241
10.56.246.219
10.56.246.133
```

If you have IP addresses are on multiple vlans you can use:

```
grep -E and the regex '[0-9]{1,3}\.'
```

That uses a “quantifier” to find 0-9 1, 2 or 3 times and then a “.”, the \ is used to tell grep that the “.” isn’t a regex operator but a real dot. It’s called “escaping”.

The “-E” is needed and is called extended regex.

## Example 2

A while back I needed to find a handful of Dell switches on a large subnet full of Dell computers and other devices. Since almost all of the MAC addresses were Dell just looking for MAC addresses didn’t help.

I ran nmap and saved the output to a file.

```
sudo nmap -sS -T4 -sC -p23 -oA powerconnect -n -Pn --open --stylesheet nmap-bootstrap.xsl 10.112.69.1-50
```

Looking at the file it was obvious that the switches returned “Ports: 23/open/tcp//telnet///”.

Contents of snippet from powerconnect.gnmap

```
# Ports scanned: TCP(1;23) UDP(0;) SCTP(0;) PROTOCOLS(0;)
Host: 10.112.69.2 () Status: Up
Host: 10.112.69.2 () Ports: 23/open/tcp//telnet///
Host: 10.112.69.3 () Status: Up
Host: 10.112.69.3 () Ports: 23/open/tcp//telnet///
Host: 10.112.69.4 () Status: Up
Host: 10.112.69.4 () Ports: 23/open/tcp//telnet///
```

A quick grep returned just the IP addresses of the switches. The awk command after the pipe just means print the second parameter in the stream. In this example “Host:” is the first parameter.

```
grep telnet/// powerconnect.gnmap | awk '{ print $2 }'
10.112.69.2
10.112.69.3
10.112.69.4
10.112.69.5
10.112.69.6
10.112.69.7
10.112.69.8
10.112.69.9
10.112.69.10
10.112.69.11
10.112.69.14
10.112.69.15
10.112.69.16
10.112.69.17
10.112.69.18
10.112.69.19
10.112.69.21
10.112.69.22
10.112.69.23
10.112.69.44
```

### Example 3

In this example, I was looking for a set of switches that had IP addresses in the range of 10.255.255. The syslog was massive so opening it in an editor and use the search feature was impractical. But piping the output of the grep/awk combo to the sort command returned only unique IP addresses almost immediately.

```
grep 10.255.255 SyslogCatchAll.txt.001 | awk '{ print $4 }' | sort -u
```

10.255.255.110  
10.255.255.111  
10.255.255.113  
10.255.255.13  
10.255.255.14  
10.255.255.17

#### Example 4

I had a directory with 8 files in it. They contained Vlan IDs, IP addresses, MAC Addresses, Port Numbers and manufacturer names. I need to pull out just the ports with certain manufacturer names. A little grep with some sort thrown in and the job is done!

Here is a listing of the files in the directory:

```
ls -l (Some information removed for clarity)
RPU-Springs-ports.txt
RPUVidSec-ports.txt
UOC-BldgB-Water-ports
UOC1stSW1-ports.txt
UOC1stSW2-ports.txt
UOCCore-ports.txt
UOCbldgB-Sw1-ports.txt
UOCbldgB-Sw2-ports.txt
```

Here is the “grep” command:

```
grep -E
'Uni|Axi|Aru|Chec|Sam|Sony|Hew|Honey|SHA|Pronet|IB|Digibo|Siemens|Tanta|Bosch|
Videx|Industr' *.txt | sort -u -b -k 1 -k 6 -k 5
```

The options:

-E – Interpret PATTERN as an extended regular expression. Without this I would have to use \| instead of just | to get the OR function.

'Uni|AXI... - These are the manufacturers that I wanted to find.

\*.txt – Search all files with extension txt

Sort – Pipe the output of the grep to the sort command

-u - Unique

-b - Ignore leading blanks.

-k - sort a table on any column number by using -k option.

The result: (Truncated for space. There were a lot more ports returned)

RPU-Springs-ports.txt:	905	10.80.152.168	9c1c.12c4.f84a	Gi0/9	ArubaHe
RPU-Springs-ports.txt:	905	10.80.153.196	3c52.82bd.14b1	Gi0/2	HewlettP
UOC-BldgB-Water-ports:	64	10.14.64.4	000b.d801.8b94	Gi1/0/20	Industri
UOC-BldgB-Water-ports:	905	10.80.152.63	9c1c.12c4.f846	Gi1/0/24	ArubaHe
UOC-BldgB-Water-ports:	905	10.80.152.69	0009.9f00.17c4	Gi1/0/13	Videx
UOC-BldgB-Water-ports:	905	10.80.152.80	e4e7.49a3.dde5	Gi1/0/24	HewlettP
UOC1stSW1-ports.txt:	63	10.14.63.212	0004.6368.d0e9	Gi0/44	BoschSec
UOC1stSW1-ports.txt:	65	10.14.65.154	0025.5a22.31e3	Gi0/40	Tantalus
UOC1stSW1-ports.txt:	905	10.195.1.190	fc15.b475.1b69	Gi0/37	HewlettP
UOC1stSW1-ports.txt:	905	10.80.152.189	f09f.fc30.119a	Gi0/39	SHARP

UOC1stSW1-ports.txt:	905	10.80.152.61	0003.2d2a.764c	Gi0/28	IBASETec
UOC1stSW2-ports.txt:	905	10.80.152.216	00a0.0306.d1fa	Gi0/41	SiemensS
UOCCore-ports.txt:	101	10.253.196.2	001c.7f81.2483	Gi2/23	CheckPoi
UOCCore-ports.txt:	260	10.251.80.10	001c.7f81.2480	Gi2/24	CheckPoi
UOCCore-ports.txt:	905	10.80.152.23	0009.9fff.0aa6	Gi4/10	Videx
UOCbldgB-Sw1-ports.txt:	905	10.80.152.44	0009.9f00.21bf	Gi1/0/16	Videx
UOCbldgB-Sw2-ports.txt:	905	10.80.152.63	9c1c.12c4.f846	Gi1/0/13	ArubaHe

These 8 switches had almost 400 ports (48x8) and this took less than a second to do. Manually, I would have had to open each of the 8 files, look for the manufacturer and then copy the information to a new file. I was migrating 72 sites and had to do this for each site. Some sites had as many as 22 IDFs!

I hope that these examples have shown you how useful the grep command is!

## References

- [Linux sort command](#)
- [Linux uniq command](#)

## **touch**

Touch is an odd little utility. If you run “touch filename” and filename doesn’t exist, touch creates an empty file with the current time as the creation date. If you run “touch filename” and filename does exist it changes the timestamp to the current system time and does not modify the file.

### [How can I change the date modified of a file](#)

From the man page:

Name

    touch - change file timestamps.

#### SYNOPSIS

    touch [OPTION]... FILE...

#### DESCRIPTION

    Update the access and modification times of each FILE to the current time.

    A FILE argument that does not exist is created empty.

## Uncomplicated firewall

Ubuntu's built-in firewall is called the Uncomplicated Firewall.

To check its status run the following:

```
systemctl status ufw (active)
● ufw.service - Uncomplicated firewall
  Loaded: loaded (/lib/systemd/system/ufw.service; enabled; vendor preset: enabled)
  Active: active (exited) since Sat 2021-05-01 14:20:29 PDT; 1h 4min ago
    Docs: man:ufw(8)
 Main PID: 371 (code=exited, status=0/SUCCESS)
   Tasks: 0 (limit: 38092)
  Memory: 0B
   CGroup: /system.slice/ufw.service
```

May 01 14:20:29 1S1K-G5-5587 systemd[1]: Finished Uncomplicated firewall.

Warning: journal has been rotated since unit was started, output may be incomplete.

```
systemctl status ufw (inactive)
```

```
● ufw.service - Uncomplicated firewall
  Loaded: loaded (/lib/systemd/system/ufw.service; enabled; vendor preset: enabled)
  Active: inactive (dead) since Fri 2019-03-01 22:01:57 PST; 2s ago
    Docs: man:ufw(8)
 Process: 6170 ExecStop=/lib/ufw/ufw-init stop (code=exited, status=0/SUCCESS)
 Main PID: 589 (code=exited, status=0/SUCCESS)
```

Mar 01 22:01:57 1S1K-G5-5587 systemd[1]: Stopping Uncomplicated firewall...

Mar 01 22:01:57 1S1K-G5-5587 ufw-init[6170]: Skip stopping firewall: ufw (not enabled)

Mar 01 22:01:57 1S1K-G5-5587 systemd[1]: Stopped Uncomplicated firewall.

Warning: Journal has been rotated since unit was started. Log output is incomplete or unavailable.

To stop the firewall, run the following:

```
sudo systemctl stop ufw
```

To start the firewall, run the following:

```
sudo systemctl start ufw
```

UFW works with IPv6. To verify that IPv6 is open:

```
sudo nano /etc/default/ufw
```

And make sure that

IPV6=yes

Display verbose settings:

```
sudo ufw status verbose
```

Status: active

```
Logging: on (low)
Default: reject (incoming), allow (outgoing), deny (routed)
New profiles: skip
```

To	Action	From
--	-----	-----
514/udp	ALLOW IN	Anywhere
69/udp	ALLOW IN	Anywhere
514/udp (v6)	ALLOW IN	Anywhere (v6)
69/udp (v6)	ALLOW IN	Anywhere (v6)

Display UFW logs

```
tail -f /var/log/ufw.log
May  1 15:14:25 1S1K-G5-5587 kernel: [ 3241.624413] [UFW BLOCK] IN=wlp0s20f3
OUT= MAC=3c:6a:a7:3a:e3:e6:9c:8c:d8:c9:17:ae:08:00 SRC=192.168.10.144
DST=224.0.0.251 LEN=221 TOS=0x00 PREC=0x00 TTL=255 ID=49450 PROTO=UDP
SPT=53535 DPT=53535 LEN=201
```

You can also find UFW's logs in the kernel buffer.

```
sudo dmesg | grep '\[UFW'
```

#### Allowing inbound connections

```
sudo ufw allow <port#>
```

For example, to allow ssh

```
sudo ufw allow 22
```

#### Allow a range of ports

You can specify port ranges with UFW. Some applications use multiple ports, instead of a single port.

For example, to allow VoIP connections, which use ports 5060-5061, use these commands:

```
sudo ufw allow 5060:5061/tcp
```

#### Allowing a specific IP address inbound

```
sudo ufw allow from 203.0.113.4 to any port 22
```

## Subnets

If you want to allow a subnet of IP addresses, you can do so using CIDR notation. For example, if you want to allow all of the IP addresses ranging from 10.10.10.1 to 10.10.10.254 you could use this command:

```
sudo ufw allow from 10.10.10.0/24 to any port 22
```

## Connections to a Network Interface

Create a firewall rule that only applies to a specific network interface,  
Sometimes you have multiple USB adapters connected and my want to allow traffic to a specific interface.

In this example, I want to allow ssh to enp60s0

```
sudo ufw allow in on enp60s0 to any port 22
```

## Deleting Rules

Use the UFW status command to display numbers next to each rule:

```
sudo ufw status numbered
```

Status: active

To	Action	From
--	-----	-----
[ 1] 514/udp	ALLOW IN	Anywhere
[ 2] 69/udp	ALLOW IN	Anywhere
[ 3] 514/udp (v6)	ALLOW IN	Anywhere (v6)
[ 4] 69/udp (v6)	ALLOW IN	Anywhere (v6)

To delete the IPv4 rule for tftp

```
sudo ufw delete 2
```

## Reset to default

If you have made changes that you don't want any longer:

```
sudo ufw reset
```

This will disable UFW and delete any rules that were previously defined.

## Reference

<https://serverfault.com/questions/516838/where-are-the-logs-for-ufw-located-on-ubuntu-server>  
<https://www.digitalocean.com/community/tutorials/how-to-set-up-a-firewall-with-ufw-on-ubuntu-20-04>

**dstat** - versatile tool for generating system resource statistics

From the MAN page

## NAME

dstat - versatile tool for generating system resource statistics

## SYNOPSIS

dstat [-afv] [options..] [delay [count]]

## DESCRIPTION

Dstat is a versatile replacement for vmstat, iostat and ifstat. Dstat overcomes some of the limitations and adds some extra features.

Dstat allows you to view all of your system resources instantly, you can eg. compare disk usage in combination with interrupts from your IDE controller, or compare the network bandwidth numbers directly with the disk throughput (in the same interval).

Dstat also cleverly gives you the most detailed information in columns and clearly indicates in what magnitude and unit the output is displayed. Less confusion, less mistakes, more efficient.

Dstat is unique in letting you aggregate block device throughput for a certain diskset or network bandwidth for a group of interfaces, ie. you can see the throughput for all the block devices that make up a single filesystem or storage system.

Dstat allows its data to be directly written to a CSV file to be imported and used by OpenOffice, Gnumeric or Excel to create graphs.

There are a lot more options, just run “man dstat” and you see the entire list. Here are two examples. The first one has cpu (-c), disk (-d), memory (-m), network (-n), and swap file (-s). The second is just the wireless interface.

```
dstat -cdmns
--total-cpu-usage-- -dsk/total- -----memory-usage----- -net/total- ----swap---
usr sys idl wai stl| read writ| used free buff cach| recv send| used free
14 5 80 0 0| 274k 2121k|9701M 717M 332M 7177M| 0 0 | 13M 2035M
15 3 80 2 0| 0 8192B|9693M 725M 332M 7172M| 450B 0 | 13M 2035M
16 3 79 2 0| 0 0|9704M 714M 332M 7183M| 844B 954B| 13M 2035M
17 3 78 1 0| 0 220k|9706M 712M 332M 7183M|3183B 2180B| 13M 2035M
19 5 74 2 0| 0 820k|9701M 716M 332M 7183M| 980B 156B| 13M 2035M
17 3 78 2 0| 0 240k|9690M 728M 332M 7169M| 768B 370B| 13M 2035M
18 4 76 1 0| 0 0|9695M 723M 332M 7174M|1030B 902B| 13M 2035M
17 4 78 1 0| 0 0|9685M 733M 332M 7168M|4413B 2006B| 13M 2035M
21 6 72 1 0| 56k 192k|9692M 726M 332M 7174M| 15k 3991B| 13M 2035M
16 4 79 1 0| 0 0|9696M 721M 332M 7174M|4133B 324B| 13M 2035M
17 4 77 1 0| 0 284k|9693M 725M 332M 7174M|1508B 1472B| 13M 2035M
```

## Monitor a network interface

In this example my wireless adapter is wlp0s20f3

```
dstat -n -N wlp0s20f3
```

```
net/wlp0s20
recv send
 0   0
228B 94B
183B 305B
192B 258B
163B 557B
605B 321B
```

## The Ubuntu Software Store

Ubuntu has its own app store. It has a surprisingly large selection of software. To open the store, simply hit the Super Key and type software. You will see icons for software update and Ubuntu Software. Select Ubuntu Software to open the store. From the homepage you can browse or click the magnifying glass and search.

## Brasero – ISO, CD, DVD tool

Brasero is a simple application to burn, copy and erase CD and DVD media: audio, video or data. It features among other things:

- On-the-fly burning
- Multisession support
- On-the-fly conversion of music playlists in all formats supported by Gstreamer

Ubuntu stopped including Brasero in 18.04 but if you need to burn CD/DVDs you can install it using the Ubuntu software store.

## Cheese webcam

I purchased an endoscope on [Endoscope on Ebay](#) for under \$20. It's very useful for looking behind racks and any area where you can't get your head. The listing said it only worked with Windows and it came with a driver CD, but when I mounted the iso on Windows, Defender popped up saying there was a trojan on it.

Instead of dealing with the driver, I connected the endoscope to my Ubuntu laptop, the driver was in the kernel, and it works perfect with Cheese.

## FBReader - A multi-platform ebook reader

Main features:

- supports several open e-book formats: fb2, html, chm, plucker, palmdoc, ztxt, tcr (psion text), rtf, oeb, openreader, non-DRM'ed mobipocket, plain text, epub, eReader
- reads directly from tar, zip, gzip, bzip2 archives (you can have several books in one archive)
- supports a structured view of your e-book collection
- automatically determines encodings
- automatically generates a table of contents
- keeps the last open book and the last read positions for all open books between runs
- automatic hyphenation (patterns for several languages are included)
- searching and downloading books from [www.feedbooks.com](#) and [www.litres.ru](#)
- partial CSS support for epub files

## Flameshot – Screenshot tool

My favorite screenshot tool, similar to the Windows snipping tool. Powerful and simple to use built-in editor with advanced features. Here is a tutorial for Flameshot:

[Flameshot screenshot software for Linux](#)

A nice feature of Flameshot is that you can start it from the terminal with the following:

```
flameshot gui -d 5000
```

And you will have 5 seconds to switch to the screen you want to capture. It allows you to screenshot dropdown menus that would close if you tried opening Flameshot with the mouse.

## Glances - An alternative to htop

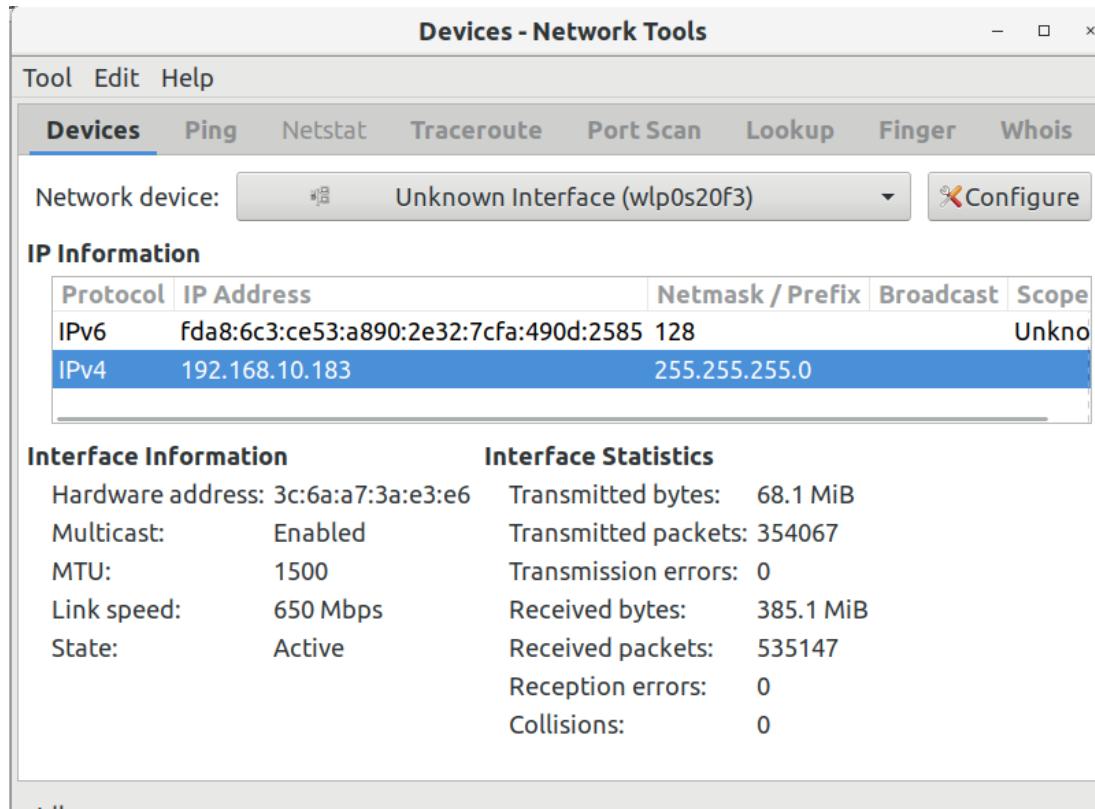
Here is a YouTube tutorial [Linux Tools: Monitoring & Troubleshooting Basics with Glances](#). When you install from the Ubuntu software store you need to click on the permission tab and set what Glances is allowed to do. I selected all.

## Gnome Network Tools - A graphical collection of networking tools.

When you first switch to Ubuntu this is a great tool. Don't put off learning the terminal commands though, Linux offers a lot more than windows. Redhat offers a great cheat sheet for the "ip" command - [IP Command cheat sheet](#)

- Devices - See interface details like IP address, subnet mask, MAC Address, MTU, Link speed for all network devices.
- Ping - Calculates Minimum, Maximum and Average
- Netstat - Routing Table, Open Ports, Multicast addresses
- Traceroute
- Port Scan - A quick port scan for common ports
- Lookup - Basic DNS queries
- Whois - A standard Whois lookup.

See "Networking Cheat Sheet" in the Networking Tools chapter for a tutorial on the most common commands.



**Meld - A great cross platform graphical file/directory comparison tool.**

Available on Linux, Mac and Windows. The

homepage for the project is [Meld](#). It's much better than the compare plugin for notepad++.

### File comparison

- Edit files in-place, and your comparison updates on-the-fly
- Perform two- and three-way diffs and merges
- Easily navigate between differences and conflicts
- Visualise global and local differences with insertions, changes and conflicts marked
- Use the built-in regex text filtering to ignore uninteresting differences
- Syntax highlighting

### Directory comparison

- Compare two or three directories file-by-file, showing new, missing, and altered files
- Directly open file comparisons of any conflicting or differing files
- Filter out files or directories to avoid seeing spurious differences
- Simple file management is also available

After installation is complete, hit the superkey and type meld to launch it. The initial interface lets you choose between file comparison, directory comparison or Version Control view.

Illustration : Meld initial screen

## OpenShot - A free, open-source, non-linear video editor

OpenShot Video Editor is a free, open-source, non-linear video editor. It can create and edit videos and movies using many popular video, audio, and image formats. Create videos for YouTube, Flickr, Vimeo, Metacafe, iPod, Xbox, and many more common formats!

## Qalculate (gtk+ui) – Calculator with Reverse Polish Notation

An interesting calculator that does almost everything including currency conversions and Reverse Polish Notation. I learned on HP calculators and algebraic calculators drive me crazy. I have happy to find Qalculate.

Qalculate is available as a snap so installation is:

```
sudo snap install qalculate
```

Illustration :

## **Remmina - The GTK+ Remmina Remote Desktop Client**

A remote desktop tool that also does VNC and ssh. Remmina is built into Ubuntu. If you want to run the latest development version you can follow the instructions below. Remmina saves its connections to `~/.local/share/remmina`.

To move connections to another Linux machine just copy all the `.remmina` files from the old machine to the new machine.

### [\*\*Remmina Usage FAQ\*\*](#)

To install the latest development version

```
sudo apt-add-repository ppa:remmina-ppa-team/remmina-next
sudo apt update
sudo apt install remmina remmina-plugin-rdp remmina-plugin-secret
```

If you need to support Microsoft's Remote Desktop Gateway with Remmina please see this blog:  
[\*\*Using Remote Desktop Gateway with Remmina\*\*](#)

## Foliate - E book reader

This is a very active project with regular updates. Works with most e-book formats. Foliate is now available as a snap so install is just a matter of:

```
sudo snap install foliate
```

From the developer's site:

A simple and modern eBook viewer

Foliate is a simple and modern GTK eBook viewer.

Features include:

- Two-page view and scrolled view
- Customize font and line-spacing
- Light, sepia, dark, and invert mode
- Reading progress slider with chapter marks
- Bookmarks and annotations
- Find in book
- Quick dictionary lookup

Foliate can open the following file types:

- .epub
- .mobi
- .azw
- .azw3

# Chapter 2 Tools installed from the Terminal

Linux has a great terminal compared to the command-line in Windows. As a network engineer you spend a lot of time in the terminal. There are a lot of great tutorials available on the Internet, this one is a good place to start:

37 Important Linux Commands You Should Know

<https://www.howtogeek.com/412055/37-important-linux-commands-you-should-know/>

The first thing to do is enable HTTPS for the aptitude terminal installation tool since HTTPS have become quite common for Linux repositories:

```
sudo apt update  
sudo apt install apt-transport-https  
sudo apt install apt-show-versions
```

Next we can start installation of the tools we need. Here we will install tools for general use. There is another section on networking tools.

## ClamAV – Open Source Anti-Virus Scanner

<https://www.clamav.net/>

ClamAV® is an open source antivirus engine for detecting trojans, viruses, malware & other malicious threats. There aren't many viruses that target Ubuntu but I like having ClamAV installed so that I can scan CDs/USB flash drives before I use them on Windows. ClamAV is installed from the terminal, the graphical interface ClamTk is installed from the store.

```
install clamav  
sudo apt install clamav
```

```
update the database  
sudo freshclam
```

You only need to run freshclam once. If you receive a message “/var/log/clamav/freshclam.log is locked by another process” it's because the log is locked by ClamAV running in the background. This article - <https://askubuntu.com/questions/909273/clamav-error-var-log-clamav-freshclam-log-is-locked-by-another-process> explains how to troubleshoot this type of error message using the Linux command lsof or List Open Files. The lsof command is useful anytime you need to see what files are open.

Example usage  
`clamscan --info.nse`  
xampp-info.nse: OK

----- SCAN SUMMARY -----

Known viruses: 6759709

Engine version: 0.100.2

Scanned directories: 0

Scanned files: 1  
Infected files: 0  
Data scanned: 0.04 MB  
Data read: 0.02 MB (ratio 2.50:1)  
Time: 10.406 sec (0 m 10 s)

## ClamTk - A graphical frontend to ClamAV.

To install – click this link [ClamTK](#) and download the .deb file. Select “Open with Software Install” and follow the instructions.

## Etcher - Flash OS images to SD cards & USB drives, safely and easily.

This is my favorite tool for creating bootable USB drives. It's cross platform and runs on Windows, Mac and Linux.

Installation is easy:

- Go to [Etcher](#) and click on Downloads. At the time of writing the current version was - etcher-electron-1.4.5-linux-x64.zip
- After the download finishes, double click to open the archive manager
- Extract the appimage file.
- Double click to execute it.

## C Compiler

A GNU C compiler is needed anytime you are building a tool from source. This is unusual on Windows but pretty common on Linux. It's actually very easy and the site normally has the exact commands needed to build from source.

```
sudo apt install gcc
sudo apt install build-essential
gcc --version
```

### More Information

[How to install gcc - the c compiler on Ubuntu 18-04](#)

## inxi - A full featured terminal system information tool

<https://github.com/smxi/inxi>

```
sudo apt install inxi
```

### Examples

```
inxi -F -x -c13 -- all output with extra data
inxi -F -xx -c13 -- all output with extra, extra data
inxi -B -- Battery info
inxi -c13 -- use black output
inxi -C -- CPU information
inxi -D -- hard drive info
```

```
inxi -f -- all cpu flags  
inxi -G -- graphics card info  
inxi -i -- network interface info  
inxi -I -- processes, uptime, memory, inxi version,  
sudo inxi -m -c13 - list memory
```

## unetbootin

A tool for creating live USB drives. Has built in support for many Linux distributions.

```
sudo add-apt-repository ppa:gezakovacs/ppa  
sudo apt update  
sudo apt install unetbootin
```

## OpenSSH Server

Ubuntu comes with an SSH client. If you want to be able to ssh back into your laptop or send files to network devices using SCP you need to install and configure the Open-ssh server. Follow these instruction from [Enable SSH on Ubuntu 18-04](#)

```
sudo apt update  
sudo apt install ssh
```

Useful SSH system commands

```
sudo systemctl start ssh - Start the SSH server  
sudo systemctl stop ssh - Stop the SSH server  
systemctl status ssh - Show server status  
sudo systemctl disable ssh - disables SSH server after next reboot  
sudo systemctl enable ssh - enables SSH after the next reboot.
```

I highly recommend SSH Mastery by Michael Lucas. It's available at [SSH Mastery](#) or Amzon.com. When I switched to Linux my only experience with SSH was Putty. There is so much more to SSH and Michael explains all of it.

## Sublime text

A full featured text editor designed for programmers. There is a trial version and the registered version costs \$75 but the license lets you install it on all the computers you own. I purchased the Real Python Sublime text configuration package from <https://www.realpython.com> and love coding Python in Sublime text. The Realpython package has videos, text and downloads for Linux, Mac and Windows.

To install the latest version of Sublime Text

```
wget -qO - https://download.sublimetext.com/sublimehq-pub.gpg | sudo apt-key add -  
echo "deb https://download.sublimetext.com/ apt/stable/" | sudo tee /etc/apt/sources.list.d/sublime-text.list  
sudo apt-get install sublime-text
```

## **Solaar**

A tool for managing Logitech Unifying dongles under Linux. I use Logitech wireless keyboards on my Odroid, Raspberry Pi and Linux laptop. This tool eliminates the need to configure the dongle on Windows.

```
sudo apt install solaar
```

## **Veracrypt**

Open source encryption tool. I love Veracrypt. It's the follow on to the old Truecrypt project. It has had a professional crypto review and passed. I use Veracrypt to do full disk encryption on USB hard drives. If I leave one at a customer or airport, the data is just random noise. You can also create encrypted partitions if you don't need full disk encryption or encrypted flash drives.

Installation

Go to <https://www.veracrypt.fr/en/Downloads.html>

Download the veracrypt-1.23-setup.tar.bz2 file (or newer)

Run

*sha256sum veracrypt-1.23-setup.tar.bz2* to verify the download

*sudo ./veracrypt-1.23-setup-gui-x* - Follow the prompts

To run Veracrypt

Press the Super key (Windows Key) and type Veracrypt in the search box.

## **exa**

From the Exa website - A modern replacement for ls written in rust.

<https://the.exa.website/>

<https://www.2daygeek.com/exa-a-modern-replacement-for-ls-command-linux/>  
download, unzip, copy to /usr/local/bin/exa

<https://www.mankier.com/1/exa>

Install rust

```
curl https://sh.rustup.rs -sSf / sh
```

Install the dependencies

```
sudo apt install libgit2-24 libgit2-dev cmake git libhttp-parser2.1
```

After the dependencies are installed, the easiest way to install exa, is to download a binary file and place it under /usr/local/bin.

```
wget https://the.exa.website/releases/exa-linux-x86_64-0.7.0.zip
```

```
unzip exa-linux-x86_64-0.7.0.zip
```

```
sudo mv exa-linux-x86_64 /usr/local/bin/exa
```

Example in a folder controlled by Git

```
mhubbard@1S1K-G5-5587:~/Dropbox/Python/Scripts/mac-addr-new$  
- >exa -l --header --git  
Permissions Size User Date Modified Git Name  
drwxrwxr-x@ - mhubbard 26 Sep 22:38 -- __pycache__  
.rw-rw-r--@ 128 mhubbard 5 Jun 23:59 -N mac-addr-6880.txt  
.rw-rw-r--@ 4.4k mhubbard 4 Jun 22:37 -N mac-addr-Nexus-93108.txt  
.rw-rw-r--@ 4.4k mhubbard 4 Jun 22:37 -N mac-addr-nexus.txt  
.rw-rw-r--@ 19k mhubbard 26 Sep 10:56 -- mac-addr.txt  
.rw-rw-r--@ 3.4k mhubbard 6 Jun 0:09 -N mac-addr1.txt  
.rw-rw-r--@ 637 mhubbard 12 May 1:00 -N MAC-Test.py  
.rw-rw-r--@ 34k mhubbard 26 Sep 10:40 -- Mac2IP.json  
.rw-rw-r--@ 24k mhubbard 4 Apr 21:32 -N Mac2IP1.json
```

Example of Tree View

```
mhubbard@1S1K-G5-5587:~/Dropbox/Python/Scripts$ exa -x -g -T
```

```
.  
└── 01-Vector  
    ├── core-backup.txt  
    ├── Core Switch Configuration.docx  
    ├── ISHS-6880X-Core Backup.txt  
    ├── sh-vlan-br.txt  
    ├── shvlanbr.txt  
    └── vlan.py  
└── 10.52.8.22  
    └── ARP-Archive  
        ├── __init__.py  
        └── __pycache__  
            ├── manuf.cpython-34.pyc  
            └── manuf.cpython-35.pyc  
        ├── arp.txt  
        ├── arpout.txt  
        ├── manuf  
        ├── manuf.py  
        ├── sbcusd_mo.txt  
        ├── SBCUSDGome.txt  
        └── test  
            ├── __init__.py  
            ├── manuf  
            └── test_manuf.py  
        «VTPDomain».txt
```

Sort by date modified and show directories first.

```
mhubbard@1S1K-G5-5587:~/Dropbox/Python/Scripts/SIET$
```

```
->exa -l --sort=modified --group-directories-first
drwxrwxr-x@      - mhubbard 26 Sep  2018 tftp
.rw-rw-r--@ 471 mhubbard 24 Apr  2018 vuln_dev_list.txt
.rw-rw-r--@ 2.0k mhubbard 24 Apr  2018 README.md
.rw-rw-r--@ 6.2k mhubbard 24 Apr  2018 sFTP.py
.rw-rw-r--@ 158k mhubbard 25 Apr  2018 DELR.txt
.rw-rw-r--@ 13k mhubbard 27 Apr  2018 siet.py
```

## RDFind

Rdfind is a program that finds duplicate files. The name is short for “redundant data find”. It uses an algorithm that is lightning fast so you can search large folders quickly.

### Installation

```
sudo apt install rdfind
```

### Show the help for rdfind

```
rdfind --help
```

usage:

```
rdfind [options] directory1 directory2 ...
```

Finds duplicate files in directories, and takes appropriate actions

Directories listed first are ranked higher, meaning that if a file is found in several places, the file found in the directory first encountered on the command line is kept, and the others are considered duplicate. options are (default choice within parentheses)

<code>-makesymlinks true false</code>	replace duplicate files with symbolic links
<code>-makehardlinks true false</code>	replace duplicate files with hard links
<code>-deleteduplicates true false</code>	delete duplicate files
<code>-ignoreempty (true)  false</code>	ignore empty files
<code>-removeidentinode (true)  false</code>	ignore files with nonunique device and inode
<code>-makeresultsfile (true)  false</code>	makes a results file
<code>-outputname name</code>	sets the results file name to "name" (default results.txt)
<code>-followsymlinks true false</code>	follow symlinks
<code>-dryrun -n true false</code>	print to stdout instead of changing anything
<code>-checksum (md5) sha1</code>	checksum type
<code>-sleep Xms</code>	sleep for X milliseconds between file reads. Default is 0. Currently, only a few values are supported; 0,1-5,10,25,50,100

If properly installed, a man page should be available as man rdfind.

### Example

```
rdfind -dryrun true /home/mhubbard/Dropbox/Python/Scripts/nmap3
/home/mhubbard/Dropbox/Python/Scripts/nmap3-1
(DRYRUN MODE) Now scanning "/home/mhubbard/Dropbox/Python/Scripts/nmap3", found 69 files.
(DRYRUN MODE) Now scanning "/home/mhubbard/Dropbox/Python/Scripts/nmap3-1", found 55
files.
```

```
(DRYRUN MODE) Now have 124 files in total.  
(DRYRUN MODE) Removed 0 files due to nonunique device and inode.  
(DRYRUN MODE) Now removing files with zero size from list...removed 0 files  
(DRYRUN MODE) Total size is 904251 bytes or 883 KiB  
(DRYRUN MODE) Now sorting on size:removed 24 files due to unique sizes from list. 100 files left.  
(DRYRUN MODE) Now eliminating candidates based on first bytes:removed 3 files from list. 97 files left.  
(DRYRUN MODE) Now eliminating candidates based on last bytes:removed 3 files from list. 94 files left.  
(DRYRUN MODE) Now eliminating candidates based on md5 checksum:removed 0 files from list. 94 files left.  
(DRYRUN MODE) It seems like you have 94 files that are not unique  
(DRYRUN MODE) Totally, 332 KiB can be reduced.  
(DRYRUN MODE) Now making results file results.txt
```

The results.txt file has all of the non-unique files listed.

### More information

<https://rdfind.pauldreik.se/>

## NCDU

An ncurses disk file display tool. Very useful for visualizing disk usage. It depends on the libncurses libraries so you may need to install them first.

```
sudo apt install libncurses5-dev libncursesw5-dev  
sudo apt install ncd
```

To use it, change to the directory you are interested in and run **ncdu**:

Example

```
---/home/mhubbard/Dropbox/Python/Scripts/SIET-----  
 1.6 MiB [#####] /tftp  
 360.0 KiB [##] /.git  
 160.0 KiB [ ] DELR.txt  
 20.0 KiB [ ] siet.py  
 12.0 KiB [ ] sFTP.py  
 8.0 KiB [ ] README.md  
 8.0 KiB [ ] vuln_dev_list.txt
```

## htop

Top is a tool built into most \*nix systems. It displays CPU and memory information very similar to running “show process cpu sorted” on a Cisco device. Since it’s built into most distributions it’s worth knowing it’s there. But it is somewhat limited in what you can do.

Htop is a similar tool but adds the ability to use the mouse, sort, kill processes and use colors in the output. To customize htop, press f2. That brings up an ncurses style menu where you can change options.

### ***Installing htop***

```
sudo apt install htop
```

### ***Running htop***

From a terminal,

```
htop
```

More information

[How to Use htop to Monitor Linux System Processes](#)

### **iotop**

Linux iotop - Check What's Stressing & Increasing Load On Hard Disks

#### **More Information**

[Linux iotop - simple top like io monitor](#)

### **iostat**

A tool for monitoring disk IO.

Install the sysstat package

```
sudo apt install sysstat
```

In the example below, I was copying a 10GB file from the Samsung EVO 960 NVME drive that Ubuntu is installed on to a Samsung EVO 850 SSD. I am using the Linux “watch” utility to update iostat every 10 second.

Notice that the NVME drive was running 10% utilization and the SSD was 100% ! NOTE: Normally you would use a 1 second interval but I couldn't highlight the text and hit ctrl+shift+c fast enough.

```
watch -n 10 iostat -xy --human 1 1
```

```
Every 10.0s: iostat -xy --human 1 1
```

```
1S1K-G5-5587: Fri Jun 21 22:32:09 2019
```

```
Linux 4.15.0-52-generic (1S1K-G5-5587) 06/21/2019 _x86_64_ (8 CPU)
```

```
avg-cpu: %user %nice %system %iowait %steal %idle
```

```
       6.1%   0.0%  19.6%  17.8%   0.0%  56.4%
```

Device	r/s	w/s	rkB/s	wkB/s	rrqm/s	wrrqm/s	%rrqm	%wrrqm	r_await	w_await	aqu-sz	rareq-sz	wareq-sz	svctm	%util
nvme0n1	502.00	147.00	501.1M	12.5M	0.00	3050.00	0.0%	95.4%	0.67	0.08	0.11	1022.2k	87.0k	0.15	10.0%
sda	1.00	759.00	4.0k	474.0M	0.00	0.00	0.0%	0.0%	48.00	182.86	141.13	4.0k	639.5k	1.32	100.0%

## More Information

[Continuously updated iostat](#)

## nload

Monitor network traffic from the terminal.

My wireless adapter is named wlp0s20f3. To run nload and display “human readable” output use the following:

```
nload -u h wlp0s20f3
```

In this case the human readable switch will output the data in bits/kbits/mbytes/gbytes depending on how much data is being transmitted.

## **Python 3 and Python tools**

Ubuntu 18.04 comes with Python 3.6.6 so no install is required.

### **PIP3**

The official installer for python tools

```
sudo apt install python3-pip
```

### **XlsxWriter**

A Python module for writing files in the Excel 2007+ XLSX file format.

<https://pypi.org/project/XlsxWriter/>

### **Tkinter**

The Tkinter library for writing graphical python interface.

```
sudo apt install python3-tk
```

### **xsltproc**

An xml to html converter. Use it with nmap's -oA option to convert the xml output to html.

```
sudo apt-get install xsltproc
```

sudo apt install nload - terminal tool to show network traffic

### **Speedtest-cli**

A simple bandwidth testing tool for the terminal

```
sudo apt install speedtest-cli
```

### **Teamviewer**

A proprietary remote access tool. At least they provide a .deb that installs painlessly under Ubuntu. You can use one account for Linux and Windows.

[Download Teamviewer for Linux](#)

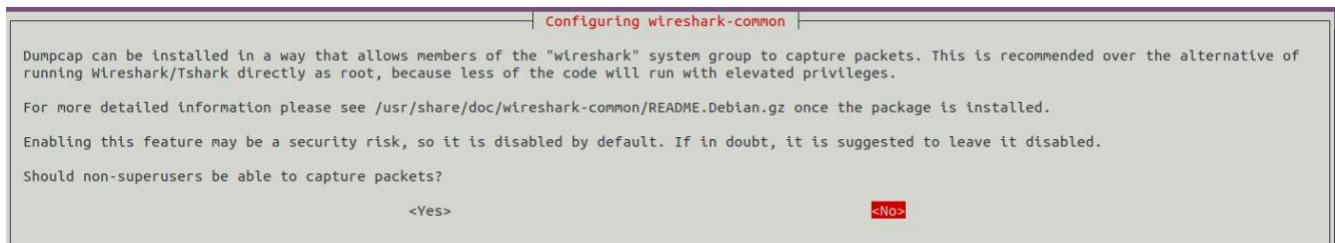
# Wireshark

No network engineer's laptop would be complete without Wireshark! The version in the Ubuntu repository is not as current as the version on the wireshark.org website. To remedy that we will add a custom repository.

## Installation

```
sudo add-apt-repository ppa:wireshark-dev/stable  
sudo apt update  
sudo apt install wireshark
```

During installation you will see a dialog box asking if you want to setup wireshark for non root users. Click yes.



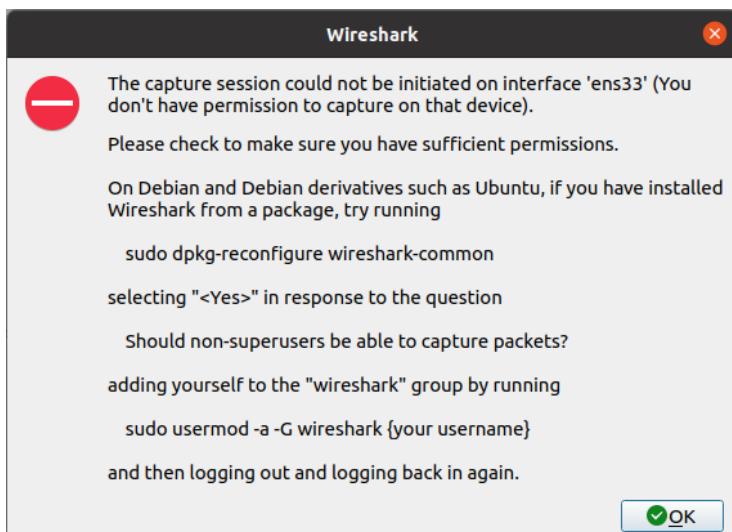
After installation is complete run the following to add your user account to the “wireshark”

```
sudo usermod -a -G wireshark mhubbard
```

Change mhubbard to your username.

That's it! Now you will have the latest version of wireshark, you will be notified to update when new versions are available and you can run wireshark without using sudo.

If you didn't select yes during setup you will be presented with the following dialog when you try to



capture packets.

Run the following to restart the setup and display the dialog

```
sudo dpkg-reconfigure wireshark-common
```

# CCZE

A robust log colorizer.

## Installation

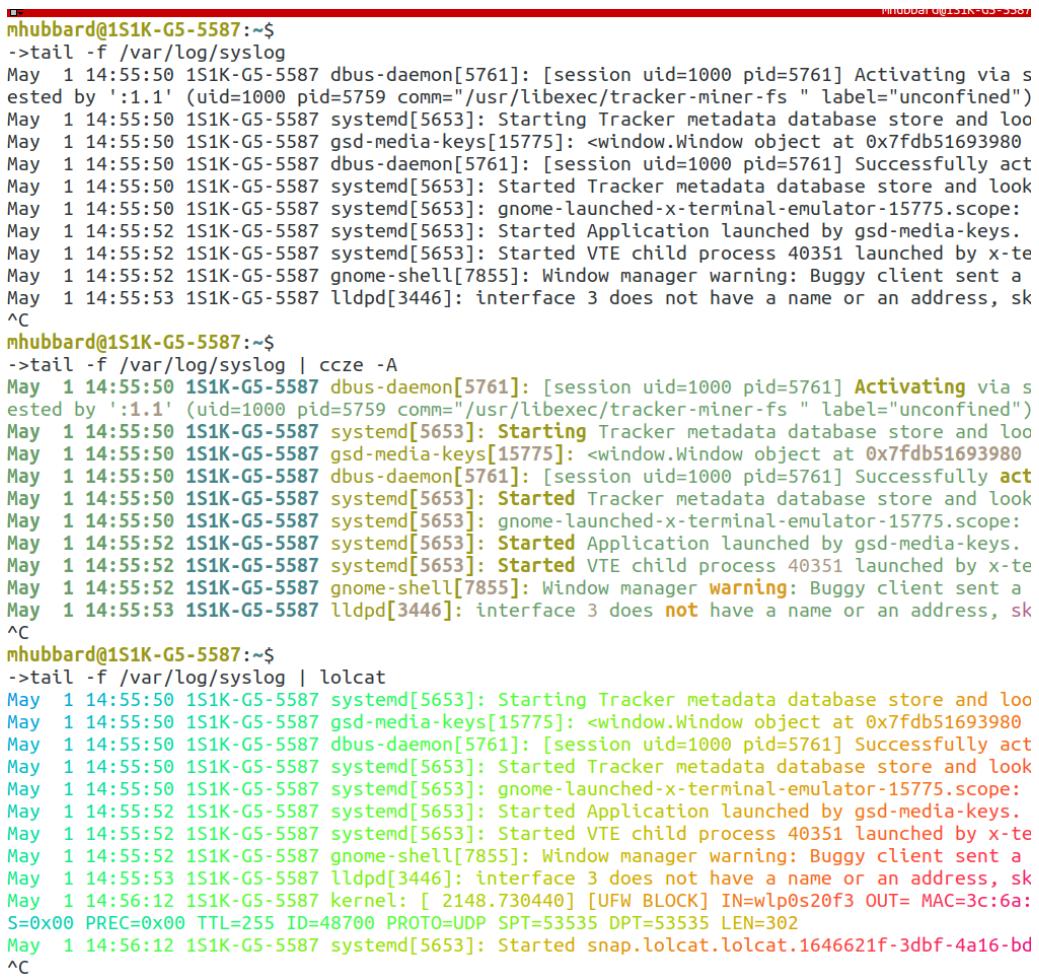
```
sudo apt install ccze
```

## Usage

Pipe the output of any tail/head to ccze

```
tail -f /var/log/syslog | ccze -A
```

Here is a screenshot of a tail of syslog with ccze -A, plain tail and lolcat.



```
mhubbard@1S1K-G5-5587:~$ ->tail -f /var/log/syslog
May 1 14:55:50 1S1K-G5-5587 dbus-daemon[5761]: [session uid=1000 pid=5761] Activating via s
ested by ':1.1' (uid=1000 pid=5759 comm="/usr/libexec/tracker-miner-fs " label="unconfined")
May 1 14:55:50 1S1K-G5-5587 systemd[5653]: Starting Tracker metadata database store and loo
May 1 14:55:50 1S1K-G5-5587 gsd-media-keys[15775]: <window.Window object at 0x7fdb51693980
May 1 14:55:50 1S1K-G5-5587 dbus-daemon[5761]: [session uid=1000 pid=5761] Successfully act
May 1 14:55:50 1S1K-G5-5587 systemd[5653]: Started Tracker metadata database store and look
May 1 14:55:50 1S1K-G5-5587 systemd[5653]: gnome-launched-x-terminal-emulator-15775.scope:
May 1 14:55:52 1S1K-G5-5587 systemd[5653]: Started Application launched by gsd-media-keys.
May 1 14:55:52 1S1K-G5-5587 systemd[5653]: Started VTE child process 40351 launched by x-te
May 1 14:55:52 1S1K-G5-5587 gnome-shell[7855]: Window manager warning: Buggy client sent a
May 1 14:55:53 1S1K-G5-5587 lldpd[3446]: interface 3 does not have a name or an address, sk
^C
mhubbard@1S1K-G5-5587:~$ ->tail -f /var/log/syslog | ccze -A
May 1 14:55:50 1S1K-G5-5587 dbus-daemon[5761]: [session uid=1000 pid=5761] Activating via s
ested by ':1.1' (uid=1000 pid=5759 comm="/usr/libexec/tracker-miner-fs " label="unconfined")
May 1 14:55:50 1S1K-G5-5587 systemd[5653]: Starting Tracker metadata database store and loo
May 1 14:55:50 1S1K-G5-5587 gsd-media-keys[15775]: <window.Window object at 0x7fdb51693980
May 1 14:55:50 1S1K-G5-5587 dbus-daemon[5761]: [session uid=1000 pid=5761] Successfully act
May 1 14:55:50 1S1K-G5-5587 systemd[5653]: Started Tracker metadata database store and look
May 1 14:55:50 1S1K-G5-5587 systemd[5653]: gnome-launched-x-terminal-emulator-15775.scope:
May 1 14:55:52 1S1K-G5-5587 systemd[5653]: Started Application launched by gsd-media-keys.
May 1 14:55:52 1S1K-G5-5587 systemd[5653]: Started VTE child process 40351 launched by x-te
May 1 14:55:52 1S1K-G5-5587 gnome-shell[7855]: Window manager warning: Buggy client sent a
May 1 14:55:53 1S1K-G5-5587 lldpd[3446]: interface 3 does not have a name or an address, sk
^C
mhubbard@1S1K-G5-5587:~$ ->tail -f /var/log/syslog | lolcat
May 1 14:55:50 1S1K-G5-5587 systemd[5653]: Starting Tracker metadata database store and loo
May 1 14:55:50 1S1K-G5-5587 gsd-media-keys[15775]: <window.Window object at 0x7fdb51693980
May 1 14:55:50 1S1K-G5-5587 dbus-daemon[5761]: [session uid=1000 pid=5761] Successfully act
May 1 14:55:50 1S1K-G5-5587 systemd[5653]: Started Tracker metadata database store and look
May 1 14:55:50 1S1K-G5-5587 systemd[5653]: gnome-launched-x-terminal-emulator-15775.scope:
May 1 14:55:52 1S1K-G5-5587 systemd[5653]: Started Application launched by gsd-media-keys.
May 1 14:55:52 1S1K-G5-5587 systemd[5653]: Started VTE child process 40351 launched by x-te
May 1 14:55:52 1S1K-G5-5587 gnome-shell[7855]: Window manager warning: Buggy client sent a
May 1 14:55:53 1S1K-G5-5587 lldpd[3446]: interface 3 does not have a name or an address, sk
May 1 14:56:12 1S1K-G5-5587 kernel: [ 2148.730440] [UFW BLOCK] IN=wlp0s20f3 OUT= MAC=3c:6a:
S=0x00 PREC=0x00 TTL=255 ID=48700 PROTO=UDP SPT=53535 DPT=53535 LEN=302
May 1 14:56:12 1S1K-G5-5587 systemd[5653]: Started snap.lolcat.lolcat.1646621f-3dbf-4a16-bd
^C
```

## Reference

<https://www.systutorials.com/docs/linux/man/1-ccze/>

## Snaps - A new way to install programs on Ubuntu

Installing applications on Linux has been difficult because Arch, Debian and Redhat all use different package managers, and some applications are only available in one of the package manager formats.

Snaps are a new way of packaging applications for Linux distributions that eliminates package dependencies, so the install is easier and Ubuntu 18.04 ships with support for Snaps. There is an official snap store at - <https://snapcraft.io/store> The store is a graphical way to find snaps. You can search based on the name of the snap or by category. There is a surprisingly large selection of software available in the store. Once you find an application you want to install, simply click on install, copy the command and paste it into the terminal.

Some Snap terminal commands:

Search the store from the terminal

***snap find <app name>***

For example:

```
snap find Keepassxc
Name      Version   Publisher   Notes   Summary
keepassxc  2.3.4    keepassxreboot -       community driven port of the windows
application "Keepass Password Safe"
keepassx-elopio 2.0.2  elopio     -       KeePassX is a cross platform password safe
```

## How to install a Snap

Once you find an application, installing easy!

***sudo snap install <app name>***

Note: Some snaps need the --classic switch. This means the snap isn't sandboxed the way normal snaps are. Be sure you trust the snap before using the --classic switch.

Here is Ubuntu's explanation:

With snapd 2.20, a new confinement policy is introduced: “classic”, designed to cater to all your scripting and tooling needs. Snaps declaring their confinement as “classic”, have access to the rest of the system, as most legacy (debian packages for example) packaged apps do, while still benefiting from the ci-integrated store model, with automated updates, rollbacks to older versions, release channels, etc.

## Refreshing Snaps

To refresh (update) all snaps

***sudo snap refresh***

All snaps up to date.

To refresh a specific snap

***sudo snap refresh <app name>***

## To list all installed Snaps

***sudo snap list***

To see all Snap commands simply enter snap and press return. All commands will be listed.

Here are link to ac couple tutorial on snaps

<https://tutorials.ubuntu.com/tutorial/basic-snap-usage#0>

<https://blog.ubuntu.com/2017/01/09/how-to-snap-introducing-classic-confinement>

Snaps I have installed:

**asciinema** - A tool to capture terminal output and save it to asciinema.org. Useful for sharing terminal application demos. Here is an example of the python tool pingsvi in asciinema -  
<https://asciinema.org/a/p9ICnD759vWOzvhJLPDxGMphY>

**Brave** - Much more than a browser, Brave is a new way of thinking about how the web works.

**Cheat** - Cheat allows you to create and view interactive cheat-sheets on the command-line. It was designed to help remind \*nix system administrators of options for commands that they use frequently, but not frequently enough to remember.

The site is <https://github.com/chrisallenlane/cheat>. I use it to save SCP commands, shortcut keys for Terminator and other things that I don't use all the time.

**emu2** - MS Dos emulator. I haven't actually used this yet but it's pretty cool if you have some old DOS programs and would like to run them on Linux. Here's the home page <https://github.com/dmsc/emu2>

**fkill** - A better way to kill a process than the built in kill command. A command-line tool for easily and quickly killing processes.

Once installed, this snap needs to be manually connected to some plugins:-

***sudo snap connect fkill:process-control :process-control***

***sudo snap connect fkill:system-observe :system-observe***

NOTE: This snap is maintained by the Snapcrafters community, and is not necessarily endorsed or officially maintained by the upstream developers.

**Ghex** - A graphical hex editor from Canonical

**Hiri** - A mail client that supports Microsoft Office 365/Exchange. Hiri is not free, there is an annual subscription cost of \$39 or a lifetime for \$119.

**htop** - htop as a snap. Htop is a terminal tool to visualize CPU/Memory/Processes etc.

**hw-probe** - A tool to probe for hardware and upload result to the Linux Hardware Database at the Linux Hardware Project <https://github.com/linuxhw/hw-probe>.

To run a probe:

***sudo hw-probe -all -upload -ID "Description."***

A URL will be onscreen when it finishes. Just paste into a browser and see all of the hardware on your laptop.

See <https://linux-hardware.org/index.php?view=howto> for more information.

**KeepassXC** - A community fork of KeePassX, the cross-platform port of KeePass for Windows.

**Lolcat** – A Command Line Tool to Output Rainbow Of Colors in the Linux Terminal

**Mailspring** - Boost your productivity and send better email with Mailspring, the best mail client for Mac, Linux, and Windows.

**Mumble** - Mumble is an open source, low-latency, high quality voice chat software. Popular in many Linux podcast communities.

**NCDU** - An Ncurses implementation of the Linux DU command. Displays disk usage for every folder and file.

**NotePadqq** - A notepad++ like editor. Has a lot of features and is open source.

<https://notepadqq.com/wp/news/>

**Pac-vs** - SecureCRT replacement. If you had been using PAC-vs on another machine it's easy to move the config over. Follow these steps:

Close Pac-vs

Copy `~/snap/pac-vs/common/.config/pac/pac.yml` from the existing machine to the same folder on the new machine. Note that `.config` is a hidden folder. Press `ctrl+h` to view hidden folders.

Delete `pac.nfreeze` using `rm pac.nfreeze`

Restart Pac-vs

**Powershell** - Microsoft open sourced Powershell. This snap installs Powershell core on Linux. Note that PowerShell requires the `--classic` switch. This removes the strict confinement model from the snap.

**rem** - A small tool for remembering things on the command line.

<https://github.com/mborho/rem>

**Simplenote** - A great cross platform note taking tool. It syncs in the cloud between all your devices.

**Skype** - Microsoft's communicator application

**Slack** - Official Slack client

**Supercalc** - A multi-function calculator that can do conversions, scientific, dates.

**Telegram** - Official desktop client for Telegram

**Termius** - A nice crossplatform SSH client. It has an annual subscription fee. I use it on my iPhone/iPad/Mac and Linux laptop.

**Tusk** - An Evernote client for Linux.

**Visual Studio Code** - Microsoft's Open source code editor. I use it for Python but it has support for almost every programming language.

**VLC** - VLC is a Video playback application.

Here is how to install the "Snaps" mentioned above

```
sudo snap install asciinema --classic
sudo snap install axeman
sudo snap install brave
sudo snap install cheat
sudo snap install emu2
sudo snap install fkill
sudo snap install ghex-udt
sudo snap install glances
sudo snamp install hiri
sudo snap install htop
sudo snap install hw-probe
sudo snap install keepassxc
sudo snap install lolcat
sudo snap install mailspring
sudo snap install mumble
sudo snap install ncdt-kz6fittycent
sudo snap install notepadqq
sudo snap install pac-vs
sudo snap install powershell --classic
sudo snap install rem --classic
sudo snap install simplenote
sudo snap install skype --classic
sudo snap install slack --classic
sudo snap install supercalc-snap
sudo snap install telegram-desktop
sudo snap install termius-app
sudo snap install tusk
sudo snap install vscode --classic
sudo snap install vlc
```

## Flatpak

Like snap, flatpak is a new containerized method for distributing Linux applications. I haven't installed too many Flatpak applications yet but it looks like it is rapidly gaining popularity so you should know it exists.

To install flatpak:

```
sudo add-apt-repository ppa:alexlarsson/flatpak  
sudo apt update  
sudo apt install flatpak  
sudo apt install gnome-software-plugin-flatpak  
flatpak remote-add --if-not-exists flathub https://flathub.org/repo/flathub.flatpakrepo
```

Flatpak requires a restart after installation. Once the reboot is complete you can go to <https://flathub.org> and search for applications.

I have installed the following flatpaks:

**Arduino IDE** - not necessarily a network engineering tool but Arduinos are fun to play with!  
**Marker** - A simple, clean markdown editor.

### Using Google Drive with Linux

Unbelievably, even though Google's cloud is built on Linux, they do not have a native client for Linux. I have found that Insync works great and has all the features of the Windows or Mac client. It's not free, when I purchased it there was a one time charge of \$25.00.

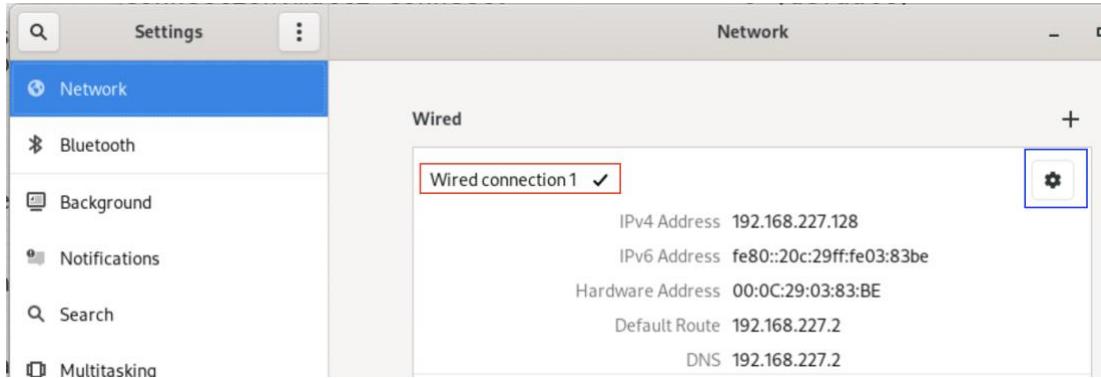
Insync for Google Drive.

# Managing Network Devices

Ubuntu provides Graphical and terminal based tools for managing network devices and connection.

## Using the GUI to Configure network Connection Profiles

Most modern Linux distributions include Network Manager for managing network settings. On Ubuntu desktop installations you can use Settings, Network to modify connection profiles. You will see a default connection profile named "Wired Connection 1" on a new install.



In the box below the name are the IPV4/IPV6 addresses, Hardware (MAC) address, default IPv4 route and DNS server for that connection.

Click on the "Gear" icon to make changes. There are five tabs as described below.

### Details

This tab lists the following information about the connection profile

Link Speed

IP Addresses (IPv4/IPv6)

Hardware (MAC) Address

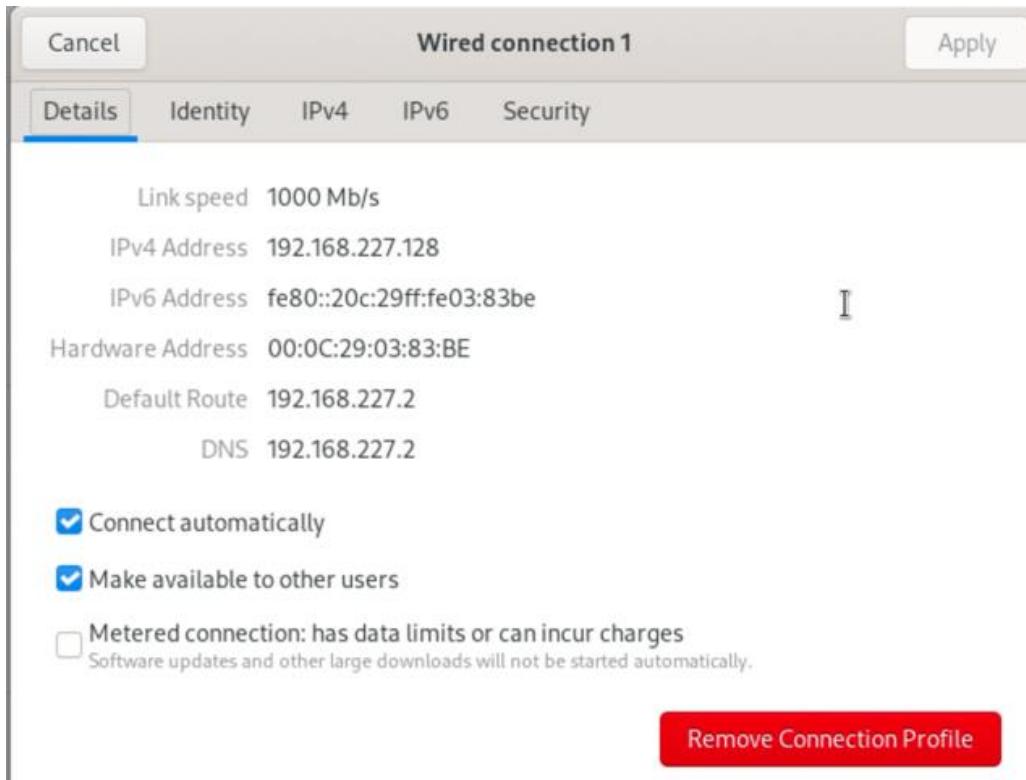
Default Route

DNS Servers

A checkbox for "Connect Automatically"

A checkbox for "Make available to others"

A checkbox for "Metered Connection: Has data limits or can incur charges. Software updates and other large downloads will not be started automatically"



## ***Identity***

Name – You can click into this field and change the connection profile name.

Note: If I am creating several connections while working on a customer network, I use the IP address or Vlan id as the connection name. I find it easier to remember what the connection is used for rather than "Wired Connection 2" or something similar.

View the MAC address

Clone a new mac address

Change the MTU

The screenshot shows a Mac OS X network connection configuration window titled "Wired connection 1". The "Identity" tab is selected. The window includes tabs for "Details", "Identity" (which is active), "IPv4", "IPv6", and "Security". The "Name" field contains "Wired connection 1". The "MAC Address" and "Cloned Address" fields both have dropdown menus. The "MTU" field is set to "1500" with a plus and minus button for adjustment, followed by "bytes". There are "Cancel" and "Apply" buttons at the top right.

## IPv4

This tab allows you to configure IPv4 settings

### IPv4 Method (How an address is assigned)

- Automatic (DHCP)
- Manual - If you need to add multiple static addresses, additional boxes will appear after you enter the first address.
- Shared to other Computers
- Link-Local Only Note: MS calls this Automatic Private Internet Protocol Addressing (APIPA)
- Disable

## DNS

set to automatic or manual

If you choose manual mode separate servers with a ","

## Routes

You can leave it on automatic or turn off automatic and enter your own routes as required

Wired connection 1

Cancel Apply

Details Identity IPv4 IPv6 Security

IPv4 Method

Automatic (DHCP)  Link-Local Only  
 Manual  Disable  
 Shared to other computers

DNS

Automatic

Separate IP addresses with commas

Routes

Automatic

Address	Netmask	Gateway	Metric
			<input type="button" value="Delete"/>

Use this connection only for resources on its network

## IPv6

### IPv6 Method (How an address is assigned)

- Automatic (DHCP)
- Automatic, DHCP only
- Manual - If you need to add multiple static addresses, additional boxes will appear after you enter the first address.
- Shared to other Computers
- Link-Local Only Note: MS calls this Automatic Private Internet Protocol Addressing (APIPA)
- Disable

## DNS

set to automatic or manual

If you choose manual mode separate servers with a ","

## Routes

You can leave it on automatic or turn off automatic and enter your own routes as required

A checkbox

Use this connection only for resources on its network

Wired connection 1

Cancel Apply

Details Identity IPv4 IPv6 Security

**IPv6 Method**

Automatic       Automatic, DHCP only  
 Link-Local Only       Manual  
 Disable       Shared to other computers

**DNS** Automatic

Separate IP addresses with commas

**Routes** Automatic

Address	Prefix	Gateway	Metric	Delete
				<input type="button" value="Delete"/>

Use this connection only for resources on its network

## **Security**

Here is where you configure the 802.1x supplicant.

For authentication, the following methods are supported:

- MD5
- TLS
- PWD
- FAST
- Tunneled TLS
- Protected EAP (PEAP)

Wired connection 1

Cancel Apply

Details Identity IPv4 IPv6 Security

802.1x Security

Authentication

Anonymous identity

CA certificate

No CA certificate is required

PEAP version

Inner authentication

Username

Password    Show password

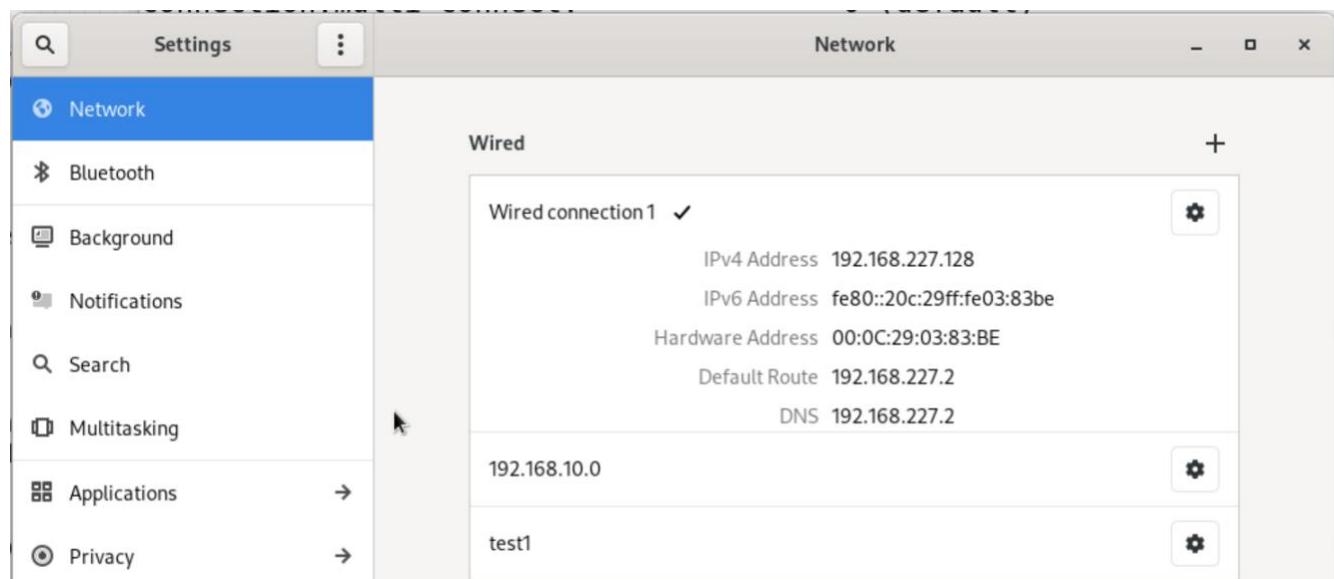
You can also create a new connection profile by clicking the "+" sign. Ubuntu (Debian) really shines with this feature because you can create many different profiles then select which one you need. This is great when testing new networks that have many vlans or anytime you are switching networks and need custom settings.

Each profile will be listed on the Network page in settings. The active connection profile will be expanded to show basic information about the connection. Any other connection profiles will be listed by name but none of the details will be visible. To change to a different profile simply click on it.

In the figure below I have created profiles

192.168.10.0

test1



## Network Manager CLI

nmcli is a command line tool that you can use to query or set adapter information. There is even an interactive shell mode and a terse mode for scripting.

Here is a nmcli tutorial - <https://www.techrepublic.com/article/how-to-use-the-nmcli-command-to-gather-network-device-information-on-linux/>

As always on \*nix systems, you can use the man page to look up commands and options – man nmcli and man nmcli-examples.

Just like on a Cisco or Aruba device you can abbreviate commands as soon as they are unique and use tab completion. You can use the -a switch and nmcli will stop and ask for any missing required arguments.

### Aliases

Before I show examples of nmcli commands I am going to show the aliases I have created. The reason is that I have noticed most users don't create aliases. I have found aliases to be a great time saver and they are very easy to create.

If you are using BASH for your shell, just use

nano ~/.bashrc

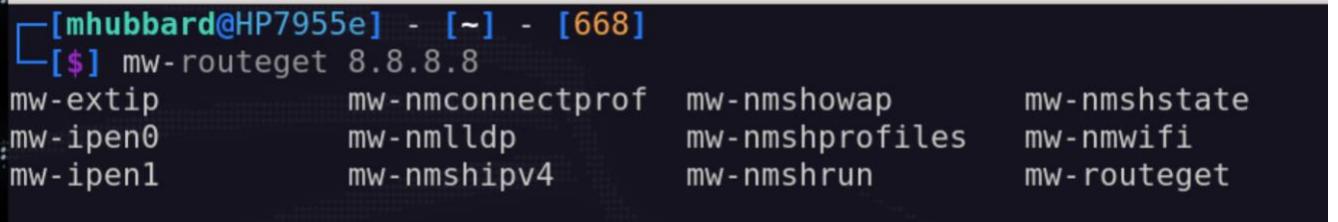
to open the file

If you are using ZSH, just use

nano ~/.zshrc

to open the file.

Then add the following lines at the bottom. I started all my network manager aliases with "mw". That allows me to type "mw [tab]" and see all of my aliases:



```
[mhubbard@HP795e] - [~] - [668]
[$] mw-routeget 8.8.8.8
mw-extip          mw-nmconnectprof   mw-nmshowap      mw-nmshstate
mw-ipen0          mw-nmlldp        mw-nmshprofiles  mw-nmwifi
mw-ipen1          mw-nmshipv4       mw-nmshrun      mw-routeget
```

```
#show status of network manager
```

```
alias mw-nmshrun="nmcli -t -f RUNNING general"
```

```
#show network manager state
```

```
alias mw-nmshstate="nmcli -t -f STATE general"
```

```
#show network connection profiles. $1 is interface name
```

```
alias mw-nmshprofiles='(){nmcli -a -f CONNECTIONS device show $1}'
```

```

#connect to an existing profile. $1 is the profile name
alias mw-nmconnectprof='(){nmcli -p connection up "$1" ifname eth0}'

#show profile IPv4 settings. Profile must be active. $1 is profile name I.E. "Wired connection 1"
alias mw-nmshipv4='(){nmcli -a -f IP4 connection show $1}'

#Show wifi properties
alias mw-nmwifi='nmcli -f GENERAL,WIFI-PROPERTIES dev show $1'

#list available Wi-Fi access points known to Network Manager
alias mw-nmshap='nmcli dev wifi'

#Use mw-nmcli to list llpd neighbors
alias mw-nmlldp='(){sudo nmcli -a -p device llpd list ifname $1}'

#show wifi passwords
alias mw-nmshwif='(){sudo nmcli -a -p device wifi show-password ifname $1}'

```

Then exit nano using ctrl+x, y, [enter]

Here is a screenshot of the nmshipv4 alias for the “test” ssid.

```

[mhubbard@HP7955e] - [/home] - [599]
[$] nmshipv4 test
IP4.ADDRESS[1]:          192.168.10.0      192.168.10.195/24
IP4.GATEWAY:             192.168.10.254
IP4.ROUTE[1]:            dst = 0.0.0.0/0, nh = 192.168.10.254, mt = 1
IP4.ROUTE[2]:            dst = 192.168.10.0/24, nh = 0.0.0.0, mt = 1
IP4.DNS[1]:              9.9.9.9
IP4.DNS[2]:              1.1.1.1
IP4.DNS[3]:              208.67.222.222

```

Here is a screenshot of the nmwifi alias

```

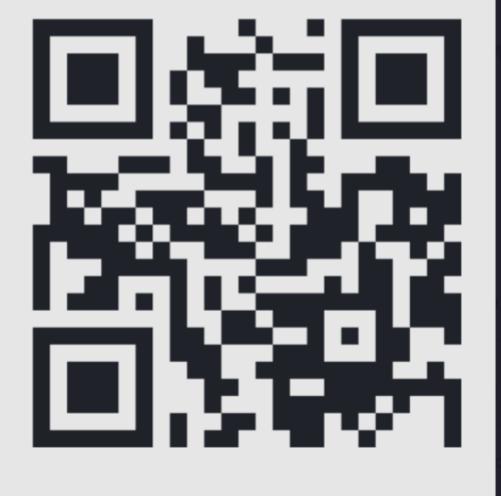
[mhubbard@HP7955e] - [/home] - [598]
[$] nmwifi
GENERAL.DEVICE:           wlan0
GENERAL.TYPE:             wifi
GENERAL.NM-TYPE:          NMDeviceWifi
GENERAL.DBUS-PATH:         /org/freedesktop/NetworkManager/Devices/3
GENERAL.VENDOR:            Ralink Technology, Corp.
GENERAL.PRODUCT:          RT2770 Wireless Adapter
GENERAL.DRIVER:            rt2800usb
GENERAL.DRIVER-VERSION:    5.15.0-kali3-arm64
GENERAL.FIRMWARE-VERSION:  N/A
GENERAL.HWADDR:            00:C0:CA:32:C3:95
GENERAL.MTU:               1500
GENERAL.STATE:             100 (connected)

```

Here is a screenshot of the nmshap alias. Very useful information indeed.

```
[mhubbard@HP7955e] - [/home] - [597]
[$] nmshap
IN-USE BSSID SSID MODE CHAN RATE SIGNAL BARS SECURITY
* 9C:8C:D8:11:7A:F0 test Infra 52 540 Mbit/s 82
```

Here is a screenshot of the nmshwif alias. If you use your camera on the QR code it will offer to connect you to the SSID - test.

```
[mhubbard@HP7955e] - [/home] -
[$] nmshwif wlan0
SSID: test
Security: WPA
Password: Guest111
QR code: 
```

Here is a screenshot of nmshap. This one is very useful when you are new to an environment and want to see the wireless environment.

```
[mhubbard@1S1K-G5-5587] - [~] - [1532]
[$] nmshap
IN-USE BSSID SSID MODE CHAN RATE SIGNAL BARS SECURITY
* 9C:8C:D8:11:7A:E0 test Infra 6 130 Mbit/s 95
  9C:8C:D8:11:7A:F0 test Infra 52 540 Mbit/s 87
  38:17:C3:12:0C:30 test Infra 116 540 Mbit/s 55
  6C:CD:D6:BE:E3:53 NETGEAR23 Infra 9 270 Mbit/s 49
  38:17:C3:12:0C:20 test Infra 11 65 Mbit/s 49
  10:1F:74:63:41:F8 HP-Print-F8-Officejet Pro 8600 Infra 11 54 Mbit/s 49
  20:0C:C8:03:24:35 Belkin.5D98_5GHz_2GEXT Infra 10 270 Mbit/s 35
  7A:A3:51:21:4B:38 SpectrumSetup-E1 Infra 1 130 Mbit/s 20
```

## **Examples:**

All these examples are from the man pages. I used "man nmcli" to display it. I modified some of the descriptions to be more consistent with this tutorial, but all of the information is from the man pages. I used "man nmcli-examples" to get additional examples.

### **Display NetworkManager status**

```
nmcli -t -f RUNNING general  
running
```

### **Display the general status of Network Manager**

```
nmcli -t -f STATE general  
connected
```

### **To shut down or bring up an interface:**

```
nmcli connection down wlp0s20f3  
nmcli connection up wlp0s20f3
```

### **Turn Wi-Fi radio off**

```
nmcli radio wifi off
```

### **Activate the connection profile "Wired connection 1" on interface enp60s0**

The -p option makes nmcli show progress of the activation.

```
nmcli -p connection up "Wired connection 1" ifname enp60s0
```

This is a great command if you use a few different USB Ethernet adapters. You can quickly activate a connection profile on a different device.

Note: This connection profile has spaces in the name. You must enclose it in double quotes or escape the spaces. I recommend not using spaces in connection profile names.

```
nmcli -p connection show Wired connection 1
```

Error: Wired - no such connection profile.

If you want to escape the spaces instead of double quotes use

```
nmcli -p connection show Wired\ connection\ 1
```

### **List all connection profiles.**

```
nmcli connection show
```

### **List all configured connections in multi-line mode**

```
nmcli -p -m multiline -f all connection show
```

### **Show the status for all devices**

```
nmcli device status
```

To list all devices (physical interfaces) enter "nmcli -p dev sho" which is device show with the "pretty" flag. It lists all the devices on your system and their current settings in a "pretty" table. Notice that wireless interfaces are listed as wifi and wired as ethernet.

```
nmcli -p device show
=====
Device details (wlp0s20f3)
=====
GENERAL.DEVICE:      wlp0s20f3
-----
GENERAL.TYPE:        wifi
-----
GENERAL.HWADDR:      3C:6A:A7:3A:E3:E6
-----
GENERAL.MTU:         1500
-----
GENERAL.STATE:       100 (connected)
-----
GENERAL.CONNECTION: test
-----
GENERAL.CON-PATH:   /org/freedesktop/NetworkManager/ActiveConnection/1
-----
IP4.ADDRESS[1]:    192.168.10.183/24
IP4.GATEWAY:       192.168.10.254
IP4.ROUTE[1]:      dst = 0.0.0.0/0, nh = 192.168.10.254, mt = 600
IP4.ROUTE[2]:      dst = 192.168.10.0/24, nh = 0.0.0.0, mt = 600
IP4.ROUTE[3]:      dst = 169.254.0.0/16, nh = 0.0.0.0, mt = 1000
IP4.DNS[1]:         192.168.10.221
-----
IP6.ADDRESS[1]:    fe80::b05c:adfa:b145:a837/64
IP6.GATEWAY:       --
IP6.ROUTE[1]:      dst = ff00::/8, nh = ::, mt = 256, table=255
IP6.ROUTE[2]:      dst = fe80::/64, nh = ::, mt = 256
IP6.ROUTE[3]:      dst = fe80::/64, nh = ::, mt = 600
-----

=====
Device details (enp60s0)
=====
GENERAL.DEVICE:      enp60s0
-----
GENERAL.TYPE:        ethernet
-----
GENERAL.HWADDR:      54:BF:64:3B:9C:68
-----
GENERAL.MTU:         1500
-----
GENERAL.STATE:       100 (connected)
-----
GENERAL.CONNECTION: Static
```

```

GENERAL.CON-PATH: /org/freedesktop/NetworkManager/ActiveConnection/9
-----
WIRED-PROPERTIES.CARRIER: on
-----
IP4.ADDRESS[1]:      204.100.254.205/29
IP4.GATEWAY:        --
IP4.ROUTE[1]:       dst = 204.100.254.200/29, nh = 0.0.0.0, mt = 100
IP4.ROUTE[2]:       dst = 169.254.0.0/16,   nh = 0.0.0.0, mt = 1000
-----
IP6.ADDRESS[1]:      fe80::e1db:4855:2a9b:e590/64
IP6.GATEWAY:        --
IP6.ROUTE[1]:       dst = ff00::/8, nh = ::, mt = 256, table=255
IP6.ROUTE[2]:       dst = fe80::/64,  nh = ::, mt = 256
IP6.ROUTE[3]:       dst = fe80::/64,  nh = ::, mt = 100
-----
```

## List WiFi details

This command will list the connected wireless interface's SSID, Mode, Channel, Rate, Signal Strength and security.

```
nmcli dev wifi
IN-USE      SSID     MODE   CHAN   RATE      SIGNAL      BARS  SECURITY
*          test     Infra   100    405 Mbit/s  76           WPA2
```

## List all devices on the system

```
nmcli device
DEVICE      TYPE      STATE      CONNECTION
wlp0s20f3  wifi      connected  test
docker0    bridge     connected  docker0
enp60s0    ethernet  unavailable
vmnet1     ethernet  unmanaged
vmnet8     ethernet  unmanaged
lo         loopback  unmanaged
```

## List details for one Interface

Now that you have the interface names, you can list all the details for one interface:

```
nmcli dev sho wlp0s20f3
GENERAL.DEVICE:      wlp0s20f3
GENERAL.TYPE:        wifi
GENERAL.HWADDR:      3C:6A:A7:3A:E3:E6
GENERAL.MTU:         1500
GENERAL.STATE:       100 (connected)
GENERAL.CONNECTION: test
GENERAL.CON-PATH:    /org/freedesktop/NetworkManager/ActiveCo
IP4.ADDRESS[1]:      192.168.10.183/24
IP4.GATEWAY:        192.168.10.254
```

```

IP4.ROUTE[1]:      dst = 0.0.0.0/0, nh = 192.168.10.254, mt
IP4.ROUTE[2]:      dst = 192.168.10.0/24, nh = 0.0.0.0, mt
IP4.ROUTE[3]:      dst = 169.254.0.0/16, nh = 0.0.0.0, mt =
IP4.DNS[1]:        1.1.1.1
IP4.DNS[2]:        208.67.222.222
IP6.ADDRESS[1]:    fda8:6c3:ce53:a890:2e32:7cfa:490d:2585/1
IP6.ADDRESS[2]:    fe80::b05c:adfa:b145:a837/64
IP6.GATEWAY:       --
IP6.ROUTE[1]:      dst = ff00::/8, nh = ::, mt = 256, table
IP6.ROUTE[2]:      dst = fe80::/64, nh = ::, mt = 256
IP6.ROUTE[3]:      dst = fda8:6c3:ce53:a890:2e32:7cfa:490d:
IP6.ROUTE[4]:      dst = fe80::/64, nh = ::, mt = 600

```

If you are looking for specific items, you can use grep and the "or" operator. Remember that Linux is case sensitive. To list just ipv4 details use "IP4" instead of "DNS|GA".

```
nmcli device show eth0 | grep -E "DNS|GA"
IP4.GATEWAY:          192.168.227.2
IP4.DNS[1]:            192.168.227.2
IP6.GATEWAY:          --
```

### List all currently active connections

```
nmcli connection show --active
```

NAME	UUID	TYPE	DEVICE
Wired connection 1	9654ef4c-2494-4e7b-9e58-6d58dec3e964	ethernet	eth0

### List all connection profile names and their auto-connect property

```
nmcli -f name,autoconnect c s (connection show)
```

NAME	AUTOCONNECT
Wired connection 1	yes
192.168.10.0	yes
test	yes

### Lists all details for the "Wired connection 1" connection profile

```
nmcli -p connection show "Wired connection 1"
```

Note: shows all properties even if the connection profile isn't applied to an interface

### List details for the "Wired connection 1" active connection, like IP, DHCP information, etc.

```
nmcli -f active connection show "Wired connection 1"
```

Note: This will only return data for active connections. DHCP options from the server are listed. For example:

```
DHCP4.OPTION[3]:      dhcp_server_identifier = 192.168.227.254
```

-f means "field". Valid fields are

6lowpan,802-11-olpc-mesh,802-11-wireless,802-11-wireless-security,802-1x,802-3-  
ethernet,adsl,bluetooth,bond,bond-port,bridge,bridge-  
port,cdma,connection,dcb,dummy,ethtool,generic,gsm,hostname,infiniband,ip-  
tunnel,ipv4,ipv6,macsec,macvlan,match,ovs-bridge,ovs-dpdk,ovs-external-ids,ovs-interface,ovs-

patch,ovs-port,ppp,pppoe,proxy,serial,sriov,tc,team,team-port,tun,user,veth,vlan,vpn,vrf,vxlan,wifi-p2p,wimax,wireguard,wpan and GENERAL,IP4,DHCP4,IP6,DHCP6,VPN, or profile,active.

### List static configuration details of the connection profile "Wired connection 1"

```
nmcli -f profile connection show "Wired connection 1"
```

Note: -f profile returned 96 rows on my laptop.

## List All Connection Profiles

To see all connections, use the connection option. Notice it returns all the wifi SSIDs that my laptop has connected to over time. There is a bash shell script that can use that information to fingerprint your laptop. You can find it here: [Show me your SSID's, I'll Tell Who You Are!](#)

### **nmcli connection**

NAME	UUID	TYPE	DEVICE
docker0	224e7634-38ef-4ad9-8a4d-273baab05f65	bridge	docker0
test	f9620460-92c5-4dfb-9b01-efa8c0372df1	wifi	wlp0s20f3
1S1K-phone	d259ea15-0549-44a2-b818-d0b5b0916ed6	wifi	--
AirConsole	7f103423-e068-476f-895e-9aef413818fd	wifi	--
BGHS	914e7da0-3b24-4a93-b4af-e3d8efa37e80	ethernet	--
BGHS	aece44c6-55e8-4f18-af43-e90c8bd089cc	ethernet	--
DHCP	76ee6a5c-9581-4b29-997e-8b2ffd941c8e	ethernet	--
MOTORBO	42da0d72-2ec9-4cdc-b7b4-7dd9001ecca0	wifi	--
RoTW	7175654c-a88e-4c8e-b18d-d390b741af6d	vpn	--
SBCUSD_ATV	f42c9d3a-7749-4639-9ea1-420d665d5ab2	wifi	--
Static	67eb952f-350a-4f55-8141-57c564795a0b	ethernet	--
Wired connection 1	eabb14c6-a3b7-3e25-ade4-098691196efa	ethernet	--
hhonors_Hampton	9520e0b4-343c-48ef-8bac-6b76dad2c9d2	wifi	--

As always, you can pipe the output to grep.

```
nmcli connection | grep eth
```

To see only connection profiles for ethernet interfaces.

### Show details for the "test" connection profile with all passwords.

Note: without --show-secrets option, secrets would not be displayed.

```
nmcli --show-secrets connection show "test"
```

If you are looking for a PSK password, you can add a grep to the end

```
nmcli --show-secrets connection show test | grep security.psk
```

802-11-wireless-security.psk: SuperSecretPasswd

### List all available connection profiles for your Wi-Fi interface wlp0s20f3.

```
nmcli -f CONNECTIONS device show wlp3s0
```

### List only GENERAL and WIFI-PROPERTIES sections for wlp0s20f3

```
nmcli -f GENERAL,WIFI-PROPERTIES dev show wlan0
```



## **Using Vlan tags**

One big advantage to Ubuntu for network engineers is that it can tag traffic using standard IEEE 801.q tags.

---

Note

you must install the vlan package using “sudo apt install vlan”.

---

There are a couple ways to do this, but nmcli is easy and straight forward. The example below creates the following:

Connection name - v156

Vlan Tag - 56

Interface - enp60s0

```
nmcli con add type vlan con-name v156 dev enp60s0 id 56 ip4 10.112.56.2/23  
gw4 10.112.57.254
```

ip addr will show the interface and IP addresses

```
enp60s0.56@enp60s0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc noqueue state UP  
group default qlen 1000  
    link/ether 54:bf:64:3b:9c:68 brd ff:ff:ff:ff:ff:ff  
    inet 10.112.56.2/23 brd 10.112.57.255 scope global noprefixroute enp60s0.56  
        valid_lft forever preferred_lft forever  
    inet6 fe80::32ee:1b10:8dcf:ed52/64 scope link noprefixroute  
        valid_lft forever preferred_lft forever
```

Notice that it's UP, UP.

### **nmcli connection**

NAME	UUID	TYPE	DEVICE
BGHS	aece44c6-55e8-4f18-af43-e90c8bd089cc	ethernet	enp60s0
docker0	704dbd24-5290-411a-a7f8-606534961bbf	bridge	docker0
v156	79f677e0-0529-474b-ab00-313c9d111cbe	vlan	enp60s0.56

The switchport configuration it's plugged into is:

```
interface GigabitEthernet1/0/12  
description < vlan 56 test >  
switchport trunk encapsulation dot1q  
switchport mode trunk  
spanning-tree portfast  
end
```

and the vlan configuration is:

```

interface Vlan56
  description < Test Network >
  ip address 10.112.57.254 255.255.254.0
  ip helper-address 192.168.10.221
end

```

## Testing the Interface

**ping 10.112.57.254**

```

PING 10.112.57.254 (10.112.57.254) 56(84) bytes of data.
64 bytes from 10.112.57.254: icmp_seq=1 ttl=255 time=2.68 ms
64 bytes from 10.112.57.254: icmp_seq=2 ttl=255 time=1.23 ms
^C

```

**ip route get 10.112.57.254**

```

10.112.57.254 dev enp60s0.56 src 10.112.56.2 uid 1000 cache

```

As you can see, the ping succeeded and the ip route get shows that it is using our vlan 56 interface.

When you are done with the interface use:

**nmcli connection**

to identify the UUID then:

**nmcli connection delete <uuid>**

to delete it.

For example, to delete the v110 connection:

**nmcli connection**

NAME	UUID	TYPE	DEVICE
docker0	704dbd24-5290-411a-a7f8-606534961bbf	bridge	docker0
test	f9620460-92c5-4dfb-9b01-efa8c0372df1	wifi	wlp0s20f3
1S1K-phone	d259ea15-0549-44a2-b818-d0b5b0916ed6	wifi	--
Wired connection 1	eabb14c6-a3b7-3e25-ade4-098691196efa	ethernet	--
hhonors_Hampton	9520e0b4-343c-48ef-8bac-6b76dad2c9d2	wifi	--
v110	cd99c9bf-4100-40be-acf8-2ed0c43bd125	vlan	--
v156	9cb5c90c-7e8b-4f54-ac95-5236bee668da	vlan	--

**nmcli connection delete cd99c9bf-4100-40be-acf8-2ed0c43bd125**

Connection 'v110' (cd99c9bf-4100-40be-acf8-2ed0c43bd125) successfully deleted.

## Monitoring interface status while a switch reboots

Linux has a great little utility called “watch”. You can use it with any other Linux utility that returns text to the screen. In this example I was connected to a switch that I needed to reboot. I wanted to monitor the Ethernet interface and know when it was back up. The default time for watch is 2 seconds but you can override that with the -n (interval) switch. In this case 2 seconds was fine.

**watch nmcli dev sho enp60s0**

```
Every 2.0s: nmcli dev sho enp60s0
1S1K-G5-5587: Tue Jun 18 15:31:03 2019
```

```
GENERAL.DEVICE:          enp60s0
GENERAL.TYPE:            ethernet
GENERAL.HWADDR:          54:BF:64:3B:9C:68
GENERAL.MTU:             1500
GENERAL.STATE:           20 (unavailable)
GENERAL.CONNECTION:      --
GENERAL.CON-PATH:        --
WIRED-PROPERTIES.CARRIER: off
```

Once the switch rebooted:

```
Every 2.0s: nmcli dev sho enp60s0                         1S1K-G5-5587: Tue Jun 18 15:47:36 2019
```

```
GENERAL.DEVICE:          enp60s0
GENERAL.TYPE:            ethernet
GENERAL.HWADDR:          54:BF:64:3B:9C:68
GENERAL.MTU:             1500
GENERAL.STATE:           100 (connected)
GENERAL.CONNECTION:      DHCP
GENERAL.CON-PATH:        /org/freedesktop/NetworkManager/ActiveConnection/13
WIRED-PROPERTIES.CARRIER: on
IP4.ADDRESS[1]:          10.140.55.169/24
IP4.GATEWAY:             10.140.55.254
IP4.ROUTE[1]:            dst = 0.0.0.0/0, nh = 10.140.55.254, mt = 100
IP4.ROUTE[2]:            dst = 10.140.55.0/24, nh = 0.0.0.0, mt = 100
IP4.ROUTE[3]:            dst = 169.254.0.0/16, nh = 0.0.0.0, mt = 1000
IP4.DNS[1]:              10.140.46.67
IP4.DNS[2]:              10.140.7.85
IP4.DNS[3]:              10.140.46.64
IP6.ADDRESS[1]:          fe80::5024:a81a:866c:c798/64
IP6.GATEWAY:             --
IP6.ROUTE[1]:            dst = ff00::/8, nh = ::, mt = 256, table=255
IP6.ROUTE[2]:            dst = fe80::/64, nh = ::, mt = 256
IP6.ROUTE[3]:            dst = fe80::/64, nh = ::, mt = 100
```

## NMCLl Cheat Sheets

<https://www.cheatography.com/nielzzz/cheat-sheets/ubuntu-server/>

[https://learncodethehardway.org/unix/bash\\_cheat\\_sheet.pdf](https://learncodethehardway.org/unix/bash_cheat_sheet.pdf)

<https://www.linuxtrainingacademy.com/linux-commands-cheat-sheet/>

## Wireless Terminal commands

Linux has many built-in or freely available terminal based tools for viewing information the wireless Interface.

### ***iwconfig***

This is an oldie but goodie in Linux! From the MAN pages:

Iwconfig is similar to ifconfig(8), but is dedicated to the wireless interfaces. It is used to set the parameters of the network interface which are specific to the wireless operation (for example: the frequency). Iwconfig may also be used to display those parameters, and the wireless statistics (extracted from /proc/net/wireless).

That last line means that we can display wireless interface statistics using the Linux watch command. Watch is a tool that runs a command at a regular interval and displays the output to the screen. The default is 2 seconds, -n1 sets it to 1 second.

```
watch -n1 iwconfig
wlp0s20f3  IEEE 802.11  ESSID:"test"
    Mode:Managed  Frequency:5.26 GHz  Access Point: 9C:1C:12:96:1D:F0
    Bit Rate=780 Mb/s  Tx-Power=22 dBm
    Retry short limit:7  RTS thr:off  Fragment thr:off
    Power Management:on
    Link Quality=70/70  Signal level=-37 dBm
    Rx invalid nwid:0  Rx invalid crypt:0  Rx invalid frag:0
    Tx excessive retries:9  Invalid misc:10  Missed beacon:0
```

You can set all parameters for the interface using iwconfig. I am not going to cover configuring an interface using iwconfig because the GUI is easily accessible from the menu at the top of the screen. From the Debian Wiki:

“iwconfig is part of the wireless-tools for Linux package maintained by Jean Tourrilhes. Due to the relative complexity of requiring two separate commands to find and sync with a wireless access point, some recommend using frontends provided by GNOME and KDE, or an application called NetGo, to manipulate these settings.”

## ***iwgetid***

The iwgetid command is designed to be used with shell or Perl scripts. You get the same output as iwconfig but in a format that makes it easy to save in a variable. You can tell the command is old because the scheme switch outputs in a format for PCMCIA cards!

```
iwgetid -h
```

```
Usage iwgetid [OPTIONS] [ifname]
```

Options are:

- a,--ap Print the access point address
- c,--channel Print the current channel
- f,--freq Print the current frequency
- m,--mode Print the current mode
- p,--protocol Print the protocol name
- r,--raw Format the output as raw value for shell scripts
- s,--scheme Format the output as a PCMCIA scheme identifier
- h,--help Print this message

Examples

```
iwgetid -a
wlp0s20f3 Access Point/Cell: 9C:1C:12:96:1D:F0
```

```
iwgetid -a -r
9C:1C:12:96:1D:F0
```

```
iwgetid -f
wlp0s20f3 Frequency:5.745 GHz
```

```
iwgetid -f -r
5.745e+09
```

```
iwgetid -m
wlp0s20f3 Mode:Managed
```

```
iwgetid -m -r
2
```

## ***iwlist***

From the man page:

Name “iwlist - Get more detailed wireless information from a wireless interface”.

### Description

Iwlist is used to display some additional information from a wireless-network interface that is not displayed by iwconfig(8). The main argument is used to select a category of information, iwlist displays in detailed form all information related to this category, including information already shown by iwconfig(8).

### SYNOPSIS

```
iwlist [interface] scanning
iwlist [interface] frequency
iwlist [interface] rate
iwlist [interface] keys
iwlist [interface] power
iwlist [interface] txpower
iwlist [interface] retry
iwlist [interface] event
iwlist [interface] auth
iwlist [interface] wpakeys
iwlist [interface] genie
iwlist [interface] modulation
iwlist --help
iwlist --version
```

It is amazing the amount of information you can get from this command.

With this one iwlist command you will get detailed information about all the APs your card can see:

```
iwlist wlp0s20f3 scanning
wlp0s20f3  Scan completed :
          Cell 01 - Address: 9C:1C:12:96:1D:F0
                      Channel:149
                      Frequency:5.745 GHz
                      Quality=64/70  Signal level=-46 dBm
                      Encryption key:on
                      ESSID:"test"
                      Bit Rates:12 Mb/s; 18 Mb/s; 24 Mb/s; 36 Mb/s; 48 Mb/s
                                  54 Mb/s
                      Mode:Master
                      Extra:tsf=00000068fa9f2032
                      Extra: Last beacon: 1902640ms ago
                      IE: Unknown: 000474657374
                      IE: Unknown: 01069824B048606C
                      IE: Unknown: 030195
                      IE: Unknown: 050400010000
```

In this example:

Cell - Address: 9C:1C:12:96:1D:F0 is the radio MAC of the AP

IE stands for Generic Information Events. I do not know how to decode them!

## Transmit Power

Note that the power switch requires sudo.

```
sudo iwlist wlp0s20f3 txpower
```

wlp0s20f3 unknown transmit-power information.

Current Tx-Power=22 dBm (158 mW)

## Bit Rate

```
iwlist wlp0s20f3 rate
wlp0s20f3    unknown bit-rate information.
              Current Bit Rate:866.7 Mb/s
```

## **More Information**

Get more detailed wireless information from a wireless interface

## ***Display Saved SSIDs***

When you connect to a wireless network, the settings, including the username/password, are saved. To view them, open a terminal and enter:

nm-connection-editor

A GUI will open and you can select a saved SSID:

## **Wavemon** Illustration : nm-connection-editor

Another wireless  
for Ubuntu. You  
`sudo apt`

From the MAN  
Wavemon is an  
monitoring  
wireless network  
levels in real-  
showing wireless  
related device

The wavemon  
into different  
screen presents  
specific manner.  
"info" screen  
levels as  
whereas the  
represents the  
moving

Run wavemon with sudo to get the most information.  
`sudo wavemon`

By default, it opens on the information screen.

monitoring tool  
install it with:  
`install wavemon`

pages -  
ncurses-based  
application for  
devices. It plots  
time as well as  
and network  
information.

interface splits  
"screens". Each  
information in a  
For example, the  
shows current  
bargraphs,  
"level" screen  
same levels as a  
histogram.

```
mhubbard@1S1K-G5-5587: ~ 82x32
Interface: wlp0s20f3 (IEEE 802.11), phy 0, reg: n/a, SSID: test
Levels
link quality: 100% (70/70)
=====
signal level: 41 dBm (12589.25 mW)
=====

Statistics
RX: 5,704 (1.97 MiB), drop: 4
TX: 5,289 (2.60 MiB), retries: 537
Info
mode: Managed, connected to: 9C:1C:12:96:1D:F0, time: 20:57m, inactive: 0.0s
freq: 5260 MHz, ctr1: 5290 MHz, channel: 52 (width: 80 MHz)
rx rate: 866.7 MBit/s VHT-MCS 9 80MHz short GI VHT-NSS 2, tx rate: 866.7 MBit/s V
beacons: 12,238, avg sig: -40 dBm, interval: 0.1s, DTIM: 1
power mgt: on, tx-power: 22 dBm (158.49 mW)
retry: short limit 7, rts/cts: off, frag: off
encryption: off (no key set)
Network
wlp0s20f3 (UP RUNNING BROADCAST MULTICAST)
mac: 3C:6A:A7:3A:E3:E6, qlen: 1000
ip: 192.168.10.183/24

F1info F2hist F3scan F4 F5 F6 F7prefs F8help F9about F10quit
```

You can press “s” to bring up the scanning screen. This is useful to see all of the SSIDs your laptop can see along with AP MAC, signal quality, power, channel and frequency.

Note that you can see “hidden” SSIDs also.

```
mhubbard@1S1K-G5-5587: ~ 82x32
Scan window

Belkin.5D98_5GHz_5GEXT      20:0C:C8:03:24:34  40%, -82 dBm, ch 153, 5765 MHz
6test                         9C:1C:12:96:1D:F0  100%, -39 dBm, ch 52, 5260 MHz
ELAB                          9C:1C:12:96:1D:F1  100%, -40 dBm, ch 52, 5260 MHz
ENETG                         A0:04:60:32:02:3E  34%, -86 dBm, ch 11, 2462 MHz
EBelkin.5D98_5GHz_2GEXT      20:0C:C8:03:24:35  43%, -80 dBm, ch 10, 2457 MHz
1PHB1 2.4G \x20               40:16:7E:A2:BB:D8  31%, -88 dBm, ch 10, 2457 MHz
3HP-Print-0C-Officejet Pro 8620 58:20:B1:5C:6E:0C  27%, -91 dBm, ch 6, 2437 MHz
E<hidden ESSID>             96:3B:AD:AF:A8:87  49%, -76 dBm, ch 5, 2432 MHz
EV Family                      92:3B:AD:AF:A8:87  49%, -76 dBm, ch 5, 2432 MHz
EV Family                      92:3B:AD:AF:A3:87  30%, -89 dBm, ch 5, 2432 MHz
E<hidden ESSID>             96:3B:AD:AF:A3:87  27%, -91 dBm, ch 5, 2432 MHz
EORBI09                        7E:D2:94:46:42:F0  30%, -89 dBm, ch 3, 2422 MHz
E<hidden ESSID>             82:D2:94:46:42:F0  29%, -90 dBm, ch 3, 2422 MHz
Etest                          9C:1C:12:96:1D:E0  100%, -39 dBm, ch 1, 2412 MHz
ELAB                          9C:1C:12:96:1D:E1  100%, -40 dBm, ch 1, 2412 MHz
EMySpectrumWiFiC0-2G          A4:08:F5:45:F3:C6  26%, -92 dBm, ch 1, 2412 MHz
ESS

total: 16 Ch/Sg desc 5/2GHz: 3/13 top-3: ch#5 (4), ch#1 (3), ch#3 (2)
2)
F1info F2hist F3scan F4        F5        F6        F7prefs F8help F9about F10quit
```

## Ethtool

Ethtool allows you to view and change the settings of your network interface

### **Set speed/duplex to 100 full**

```
sudo ip link set enp60s0 down
sudo ethtool -s enp60s0 autoneg off speed 100 duplex full
sudo ip link set enp60s0 up
```

Auto-negotiation of speed and duplex is part of the 1000BASE-T specification, that is gigabit ethernet over unshielded twisted pair copper cables.

You can set speed and duplex for 10Mbps and 100Mbps, because autoneg was not required for those versions of Ethernet.

If you like, you can disable the various speeds and duplexes which a NIC advertises, so it only ever advertises 1000/full, but auto-negotiation is still required.

### **Advertising speed and duplex**

advertise - Sets the speed and duplex advertised by autonegotiation.

Example - set advertised speed to 100baseT Full

```
sudo ethtool -s enp60s0 advertise 0x008
```

To set multiple advertisements add the hex values.

Example - set 100baseT Full and 1000baseT Full

```
sudo ethtool -s enp60s0 advertise 0x028
```

The argument is a hexadecimal value using one or a combination of the following values:

0x001	10baseT Half
0x002	10baseT Full
0x004	100baseT Half
0x008	100baseT Full
0x010	1000baseT Half      (not supported by IEEE standards)
0x020	1000baseT Full
0x20000	1000baseKX Full
0x200000000000	1000baseX Full
0x800000000000	2500baseT Full
0x8000	2500baseX Full      (not supported by IEEE standards)
0x100000000000	5000baseT Full
0x1000	10000baseT Full
0x40000	10000baseKX4 Full
0x80000	10000baseKR Full
0x400000000000	10000baseCR Full
0x800000000000	10000baseSR Full
0x100000000000	10000baseLR Full
0x200000000000	10000baseLRM Full
0x400000000000	10000baseER Full
0x200000	20000baseMLD2 Full      (not supported by IEEE standards)

```
0x4000000 2000baseKR2 Full      (not supported by IEEE standards)
0x800000000 2500baseCR Full
0x100000000 2500baseKR Full
0x200000000 2500baseSR Full
0x8000000 4000baseKR4 Full
0x10000000 4000baseCR4 Full
0x20000000 4000baseSR4 Full
0x40000000 4000baseLR4 Full
0x4000000000 5000baseCR2 Full
0x8000000000 5000baseKR2 Full
0x10000000000 5000baseSR2 Full
0x8000000 5600baseKR4 Full
0x10000000 5600baseCR4 Full
0x200000000 5600baseSR4 Full
0x400000000 5600baseLR4 Full
0x10000000000 10000baseKR4 Full
0x20000000000 10000baseSR4 Full
0x40000000000 10000baseCR4 Full
0x80000000000 10000baseLR4_ER4 Full
```

### ***Display autonegotiation***

***ethtool -a enp60s0***

Pause parameters for enp60s0:

Autonegotiate: off

RX: on

TX: on

### ***Display statistics***

***sudo ethtool -S enp60s0***

### ***Display only Errors***

***sudo ethtool -S enp60s0 | grep er***

```
rx_fcs_errors: 0
rx_length_errors: 0
rx_oversize_packets: 0
rx_align_errors: 45
rx_address_errors: 3554
tx_exc_defer_packets: 0
tx_defer_packets: 31
tx_underrun: 0
tx_length_errors: 0
```

### ***Display only Collisions***

***sudo ethtool -S enp60s0 | grep co***

```
tx_single_collision: 8
tx_multiple_collisions: 9
tx_late_collision: 1
tx_abort_collision: 0
```

## Show Features

```
sudo ethtool -k enp60s0
```

Features for enp60s0:

```
rx-checksumming: off
tx-checksumming: off
    tx-checksum-ipv4: off [fixed]
    tx-checksum-ip-generic: off
    tx-checksum-ipv6: off [fixed]
    tx-checksum-fcoe-crc: off [fixed]
    tx-checksum-sctp: off [fixed]
scatter-gather: off
    tx-scatter-gather: off
    tx-scatter-gather-fraglist: off [fixed]
tcp-segmentation-offload: off
    tx-tcp-segmentation: off
    tx-tcp-ecn-segmentation: off [fixed]
    tx-tcp-mangleid-segmentation: off
    tx-tcp6-segmentation: off
udp-fragmentation-offload: off
generic-segmentation-offload: off [requested on]
generic-receive-offload: on
large-receive-offload: off [fixed]
rx-vlan-offload: off [fixed]
tx-vlan-offload: off [fixed]
ntuple-filters: off [fixed]
receive-hashing: off [fixed]
highdma: off [fixed]
rx-vlan-filter: off [fixed]
vlan-challenged: off [fixed]
tx-lockless: off [fixed]
netns-local: off [fixed]
tx-gso-robust: off [fixed]
tx-fcoe-segmentation: off [fixed]
tx-gre-segmentation: off [fixed]
tx-gre-csum-segmentation: off [fixed]
tx-ipxip4-segmentation: off [fixed]
tx-ipxip6-segmentation: off [fixed]
tx-udp_tnl-segmentation: off [fixed]
tx-udp_tnl-csum-segmentation: off [fixed]
tx-gso-partial: off [fixed]
tx-sctp-segmentation: off [fixed]
```

```
tx-esp-segmentation: off [fixed]
fcoe-mtu: off [fixed]
tx-nocache-copy: off
loopback: off [fixed]
rx-fcs: off [fixed]
rx-all: off [fixed]
tx-vlan-stag-hw-insert: off [fixed]
rx-vlan-stag-hw-parse: off [fixed]
rx-vlan-stag-filter: off [fixed]
l2-fwd-offload: off [fixed]
hw-tc-offload: off [fixed]
esp-hw-offload: off [fixed]
esp-tx-csum-hw-offload: off [fixed]
rx-udp_tunnel-port-offload: off [fixed]
```

### ***Display permanent address***

```
ethtool -P wlp0s20f3
Permanent address: 3c:6a:a7:3a:e3:e6
```

### ***Flash the NIC LED***

To blink the LED of Ethernet device enp60s0 for 10 seconds:

```
sudo ethtool -p enp60s0 10
```

The LED will begin blinking, so you know which card you're dealing with. NOTE: the card must support this feature.

### ***Testing a NIC***

```
sudo ethtool -t enp60s0 online
Cannot test: Operation not supported
```

Testing is not supported on all NICs. For example, my Dell laptop has a Qualcomm Atheros Killer E2400 Gigabit Ethernet chip and testing isn't supported.

### ***Show Driver Information***

```
ethtool -i enp60s0
driver: alx
version:
firmware-version:
expansion-rom-version:
bus-info: 0000:3c:00.0
supports-statistics: yes
supports-test: no
```

```
supports-eeprom-access: no  
supports-register-dump: no  
supports-priv-flags: no
```

## ***Make Changes Permanent After Reboot***

If you've changed any ethernet device parameters using the ethtool, it will all disappear after the next reboot, unless you do the following.

On ubuntu, you have to modify /etc/network/interfaces file and add all your changes as shown below.

```
sudo gedit /etc/network/interfaces
```

```
post-up ethtool -s eth2 speed 100 duplex full autoneg off
```

The above line should be the last line of the file. This will change speed, duplex and autoneg of eth2 device permanently.

More Information

<https://www.garron.me/en/linux/ubuntu-network-speed-duplex-lan.html>

<https://www.techrepublic.com/article/how-to-troubleshoot-an-ethernet-interface-ubuntu-server-with-ethtool/>

<https://etherealmind.com/ethernet-autonegotiation-works-why-how-standard-should-be-set/>

[http://www.cisco.com/en/US/tech/tk389/tk214/technologies\\_tech\\_note09186a0080094781.shtml](http://www.cisco.com/en/US/tech/tk389/tk214/technologies_tech_note09186a0080094781.shtml)

<https://www.thegeekstuff.com/2010/10/ethtool-command/>

## ***LLDP client for Linux***

Having an LLDP client on your laptop is very useful when you are working with non-Cisco switches, VoIP phones and any other devices that support LLDP. There are LLDP programs for Windows but most are shareware that time out and then have to be purchased.

<https://fnord.no/2016/04/28/lldp-on-linux/>

### ***Installation***

```
sudo apt install lldpd
```

```
sudo service lldpd restart
```

### ***LLDP client Usage***

lldpcli show neighbors

lldpcli show interfaces

lldpcli show chassis [summary|details]

lldpcli show configuration

lldpcli show statistics

lldpcli update - Make lldpd update its information and send new LLDP PDU on all interfaces.

For detailed help type "man lldpd"

## ***Running interactively***

You can also run it interactively by running lldpcli without any options:

```
mhubbard@1S1K-G5-5587:~/Dropbox/Python/Scripts/SIET$  
->lldpcli  
[lldpcli] $  
[lldpcli] $ sh neighbors
```

---

LLDP neighbors:

---

Interface: enp60s0, via: LLDP, RID: 5, Time: 0 day, 00:04:01

Chassis:

ChassisID: mac 10:1f:74:a4:01:20  
SysName: PROCURVE J9450A  
SysDescr: HP ProCurve 1810G - 24 GE, P.1.17, eCos-2.0  
MgmtIP: 192.168.10.10  
Capability: Bridge, on

Port:

PortID: local 3  
PortDescr: Port #3  
TTL: 120

---

(Note - this is a Windows 7 virtual machine with a hostname of FFKN25S)

Interface: vmnet8, via: LLDP, RID: 4, Time: 0 day, 00:44:06

Chassis:

ChassisID: local EPCEheha  
SysName: FFKN25S  
SysDescr: Intel64 Family 6 Model 158 Stepping 10, GenuineIntel - Windows 6.1 (Build 7601)  
Service Pack 1

MgmtIP: 172.16.209.128  
Capability: Station, on

Port:

PortID: mac 00:0c:29:f5:02:cc  
PortDescr: Intel(R) PRO/1000 MT Network Connection  
TTL: 120

---

[lldpcli] \$ sh interfaces

---

LLDP interfaces:

---

Interface: enp60s0, via: LLDP, Time: 0 day, 03:28:15

Chassis:

ChassisID: mac 54:bf:64:3b:9c:68  
SysName: 1S1K-G5-5587

SysDescr: Ubuntu 18.04.1 LTS Linux 4.15.0-36-generic #39-Ubuntu SMP Mon Sep 24 16:19:09  
UTC 2018 x86\_64

MgmtIP: 192.168.10.183

MgmtIP: fe80::b05c:adfa:b145:a837

Capability: Bridge, off

Capability: Router, off

Capability: Wlan, on

Capability: Station, off

Port:

PortID: mac 54:bf:64:3b:9c:68

PortDescr: enp60s0

TTL: 120

---

Interface: wlp0s20f3, via: unknown, Time: 0 day, 03:46:10

Chassis:

ChassisID: mac 54:bf:64:3b:9c:68

SysName: 1S1K-G5-5587

SysDescr: Ubuntu 18.04.1 LTS Linux 4.15.0-36-generic #39-Ubuntu SMP Mon Sep 24 16:19:09  
UTC 2018 x86\_64

MgmtIP: 192.168.10.183

MgmtIP: fe80::b05c:adfa:b145:a837

Capability: Bridge, off

Capability: Router, off

Capability: Wlan, on

Capability: Station, off

Port:

PortID: mac 3c:6a:a7:3a:e3:e6

PortDescr: wlp0s20f3

TTL: 120

---

Interface: vmnet1, via: unknown, Time: 0 day, 03:46:36

Chassis:

ChassisID: mac 54:bf:64:3b:9c:68

SysName: 1S1K-G5-5587

SysDescr: Ubuntu 18.04.1 LTS Linux 4.15.0-36-generic #39-Ubuntu SMP Mon Sep 24 16:19:09  
UTC 2018 x86\_64

MgmtIP: 192.168.10.183

MgmtIP: fe80::b05c:adfa:b145:a837

Capability: Bridge, off

Capability: Router, off

Capability: Wlan, on

Capability: Station, off

Port:

PortID: mac 00:50:56:c0:00:01

PortDescr: vmnet1

TTL: 120

---

Interface: vmnet8, via: LLDP, Time: 0 day, 03:46:36  
Chassis:  
  ChassisID: mac 54:bf:64:3b:9c:68  
  SysName: 1S1K-G5-5587  
  SysDescr: Ubuntu 18.04.1 LTS Linux 4.15.0-36-generic #39-Ubuntu SMP Mon Sep 24 16:19:09  
  UTC 2018 x86\_64  
  MgmtIP: 192.168.10.183  
  MgmtIP: fe80::b05c:adfa:b145:a837  
  Capability: Bridge, off  
  Capability: Router, off  
  Capability: Wlan, on  
  Capability: Station, off  
Port:  
  PortID: mac 00:50:56:c0:00:08  
  PortDescr: vmnet8  
  TTL: 120

---

[lldpcli] \$ sh chassis

---

Local chassis:

---

Chassis:  
  ChassisID: mac 54:bf:64:3b:9c:68  
  SysName: 1S1K-G5-5587  
  SysDescr: Ubuntu 18.04.1 LTS Linux 4.15.0-36-generic #39-Ubuntu SMP Mon Sep 24 16:19:09  
  UTC 2018 x86\_64  
  MgmtIP: 192.168.10.183  
  MgmtIP: fe80::b05c:adfa:b145:a837  
  Capability: Bridge, off  
  Capability: Router, off  
  Capability: Wlan, on  
  Capability: Station, off

---

[lldpcli] \$ sh configuration

---

Global configuration:

---

Configuration:  
  Transmit delay: 30  
  Transmit hold: 4  
  Receive mode: no  
  Pattern for management addresses: (none)  
  Interface pattern: (none)  
  Interface pattern for chassis ID: (none)  
  Override description with: (none)  
  Override platform with: Linux  
  Override system name with: (none)

Advertise version: yes  
Update interface descriptions: no  
Promiscuous mode on managed interfaces: no  
Disable LLDP-MED inventory: yes  
LLDP-MED fast start mechanism: yes  
LLDP-MED fast start interval: 1  
Source MAC for LLDP frames on bond slaves: local  
Port ID TLV subtype for LLDP frames: unknown  
Agent type: unknown

## Reference

[Vincent Bernat's LLDPD usage instructions](#)

# Networking Tools

One of the greatest things about Linux is the vast number of free tools for network engineers. Here are the ones I have found. I'm sure this is just a small percentage of all the tools available.

## Sipcalc

A great terminal subnet calculator.

Installation

```
sudo apt install sipcalc
```

The simplest use is just sipcalc and a network address:

```
sipcalc 10.56.0.0/21
-[ipv4 : 10.56.0.0/21] - 0
```

```
[CIDR]
Host address           - 10.56.0.0
Host address (decimal) - 171442176
Host address (hex)     - A380000
Network address        - 10.56.0.0
Network mask           - 255.255.248.0
Network mask (bits)    - 21
Network mask (hex)     - FFFF800
Broadcast address      - 10.56.7.255
Cisco wildcard         - 0.0.7.255
Addresses in network   - 2048
Network range          - 10.56.0.0 - 10.56.7.255
Usable range           - 10.56.0.1 - 10.56.7.254
```

In this example, sipcalc returned IPv4 information including the results in dotted quad notation, hex and the Cisco wildcard mask along with the number of hosts in the network!

But, sipcalc supports a lot of options. The -a switch is useful if you are helping someone learn subnetting and they need the masks in bit form:

```
[Classful bitmaps]
Network address       - 00001010.00000000.00000000.00000000
Network mask          - 11111111.00000000.00000000.00000000

[CIDR bitmaps]
Host address          - 00001010.00111000.00000000.00000000
Network address        - 00001010.00111000.00000000.00000000
Network mask          - 11111111.11111111.11111110.00000000
Broadcast address     - 00001010.00111000.00000001.11111111
Cisco wildcard        - 00000000.00000000.00000001.11111111
Network range         - 00001010.00111000.00000000.00000000 -
```

```

          00001010.00111000.00000001.11111111
Usable range      - 00001010.00111000.00000000.00000001 -
                  00001010.00111000.00000001.11111110

```

## **Sipcalc using an Interface**

Another useful switch is the -I or Interface switch. It returns the information for the specified interface.

### **sipcalc -I vmnet8**

```
-[int-ipv4 : vmnet8] - 0
```

```
[CIDR]
Host address      - 172.16.209.1
Host address (decimal) - 2886783233
Host address (hex) - AC10D101
Network address    - 172.16.209.0
Network mask       - 255.255.255.0
Network mask (bits)- 24
Network mask (hex) - FFFFFFF00
Broadcast address  - 172.16.209.255
Cisco wildcard     - 0.0.0.255
Addresses in network - 256
Network range      - 172.16.209.0 - 172.16.209.255
Usable range        - 172.16.209.1 - 172.16.209.254
```

You can see all of the options with:

### **sipcalc -h**

sipcalc 1.1.6

Usage: sipcalc [OPTIONS]... <[ADDRESS]... [INTERFACE]... | [-]>

#### Global options:

-a, --all	All possible information.
-d, --resolve	Enable name resolution.
-h, --help	Display this help.
-I, --addr-int=INT	Added an interface.
-n, --subnets=NUM	Display NUM extra subnets (starting from the current subnet). Will display all subnets in the current /24 if NUM is 0.
-u, --split-verbose	Verbose split.
-v, --version	Version information.
-4, --addr-ipv4=ADDR	Add an ipv4 address.
-6, --addr-ipv6=ADDR	Add an ipv6 address.

#### IPv4 options:

-b, --cidr-bitmap	CIDR bitmap.
-c, --classful-addr	Classful address information.

<b>-i, --cidr-addr</b>	CIDR address information. (default)
<b>-s, --v4split=MASK</b>	Split the current network into subnets of MASK size.
<b>-w, --wildcard</b>	Display information for a wildcard (inverse mask).
<b>-x, --classful-bitmap</b>	Classful bitmap.

#### IPv6 options:

<b>-e, --v4inv6</b>	IPv4 compatible IPv6 information.
<b>-r, --v6rev</b>	IPv6 reverse DNS output.
<b>-S, --v6split=MASK</b>	Split the current network into subnets of MASK size.
<b>-t, --v6-standard</b>	Standard IPv6. (default)

Address must be in the "standard" dotted quad format.

Netmask can be given in three different ways:

- Number of bits [ /nn ]
- Dotted quad [ nnn.nnn.nnn.nnn ]
- Hex [ 0xnnnnnnnn | nnnnnnnn ]

Interface must be a valid network interface on the system.

If this option is used an attempt will be made to gain the address and netmask from the specified interface.

## Ipcalc

Another subnet mask tool. Ipcalc outputs the information in decimal and binary.

Installation

```
sudo apt install ipcalc
```

Example:

```
ipcalc 10.56.0.0/21
Address: 10.56.0.0 00001010.0011000.00000 000.0000000
Netmask: 255.255.248.0 = 21 11111111.11111111.11111 000.0000000
Wildcard: 0.0.7.255 00000000.00000000.00000 111.1111111
=>
Network: 10.56.0.0/21 00001010.0011000.00000 000.0000000
HostMin: 10.56.0.1 00001010.0011000.00000 000.0000001
HostMax: 10.56.7.254 00001010.0011000.00000 111.1111110
Broadcast: 10.56.7.255 00001010.0011000.00000 111.1111111
Hosts/Net: 2046 Class A, Private Internet
```

Like sipcalc, ipcalc has a lot of options. Like most Linux terminal apps the -h switch lists them.

## ipcalc -h

Usage: ipcalc [options] <ADDRESS>[[/]<NETMASK>] [NETMASK]

ipcalc takes an IP address and netmask and calculates the resulting broadcast, network, Cisco wildcard mask, and host range. By giving a second netmask, you can design sub- and supernetworks. It is also intended to be a teaching tool and presents the results as easy-to-understand binary values.

- n --nocolor Don't display ANSI color codes.
- c --color Display ANSI color codes (default).
- b --nobinary Suppress the bitwise output.
- c --class Just print bit-count-mask of given address.
- h --html Display results as HTML (not finished in this version).
- v --version Print Version.
- s --split n1 n2 n3
  - Split into networks of size n1, n2, n3.
- r --range Deaggregate address range.
- help Longer help text.

Examples:

```
ipcalc 192.168.0.1/24
ipcalc 192.168.0.1/255.255.128.0
ipcalc 192.168.0.1 255.255.128.0 255.255.192.0
```

```
ipcalc 192.168.0.1 0.0.63.255
```

```
ipcalc <ADDRESS1> - <ADDRESS2> deaggregate address range  
ipcalc <ADDRESS>/<NETMASK> --s a b c  
split network to subnets where a b c fits in.
```

Notice that you can enter two subnet masks and ipcalc will display both ranges:

***ipcalc 192.168.0.1 255.255.128.0 255.255.192.0***

```
Address: 192.168.0.1      11000000.10101000.0 00000000.00000001  
Netmask: 255.255.128.0 = 17 11111111.11111111.1 00000000.00000000  
Wildcard: 0.0.127.255     00000000.00000000.0 1111111.11111111  
=>  
Network: 192.168.0.0/17   11000000.10101000.0 00000000.00000000  
HostMin: 192.168.0.1     11000000.10101000.0 00000000.00000001  
HostMax: 192.168.127.254 11000000.10101000.0 1111111.11111110  
Broadcast: 192.168.127.255 11000000.10101000.0 1111111.11111111  
Hosts/Net: 32766          Class C, Private Internet
```

Subnets after transition from /17 to /18

```
Netmask: 255.255.192.0 = 18 11111111.11111111.11 0000000.00000000  
Wildcard: 0.0.63.255       00000000.00000000.00 111111.11111111
```

1.

```
Network: 192.168.0.0/18   11000000.10101000.00 0000000.00000000  
HostMin: 192.168.0.1     11000000.10101000.00 0000000.00000001  
HostMax: 192.168.63.254  11000000.10101000.00 111111.11111110  
Broadcast: 192.168.63.255 11000000.10101000.00 111111.11111111  
Hosts/Net: 16382          Class C, Private Internet
```

2.

```
Network: 192.168.64.0/18  11000000.10101000.01 0000000.00000000  
HostMin: 192.168.64.1    11000000.10101000.01 0000000.00000001  
HostMax: 192.168.127.254 11000000.10101000.01 111111.11111110  
Broadcast: 192.168.127.255 11000000.10101000.01 111111.11111111  
Hosts/Net: 16382          Class C, Private Internet
```

Subnets: 2

Hosts: 32764

## Arpscan

One of the best features of Linux is that you have much more flexibility with the network stack than you have with Windows. The arp-scan tool by Roy Hill allows you to use arp instead of ICMP to locate devices on the network.

Why is this important? Today, many devices have a firewall turned on by default. A ping scan won't locate these devices but an arp scan will. If you need to connect to a network that doesn't have DHCP, you must make sure that the address you select isn't in use. Ping can't guarantee that because of firewalls but arp-scan can find devices that don't respond to ping.

Installation

```
sudo apt install arp-scan
```

### Examples

In this example, my Macbook Air has the firewall turned on and an IP address of 192.168.10.175. A ping fails to locate it but arp-scan returns the mac address and manufacture.

```
ping 192.168.10.175
PING 192.168.10.175 (192.168.10.175) 56(84) bytes of data.
^C
--- 192.168.10.175 ping statistics ---
8 packets transmitted, 0 received, 100% packet loss, time 7175ms

mhubbard@1S1K-G5-5587:~$ 
sudo arp-scan -I wlp0s20f3 192.168.10.175
Interface: wlp0s20f3, datalink type: EN10MB (Ethernet)
Starting arp-scan 1.9 with 1 hosts (http://www.nta-monitor.com/tools/arp-scan/)
192.168.10.175      b8:8d:12:08:65:aa  Apple, Inc.

1 packets received by filter, 0 packets dropped by kernel
Ending arp-scan 1.9: 1 hosts scanned in 1.008 seconds (0.99 hosts/sec). 1 responded
```

### Scan using the adapter's settings

In this example we scan the entire subnet based off of the adapter's settings. This example assumes we already have an IP address and mask assigned and are just looking for devices that don't respond to ping.

Note: If you don't know anything about the network you can use wireshark and arp-scan in combination. See my blog for a detailed example - <https://mwhubbard.blogspot.com/2017/04/using-arp-scan-to-find-free-ip-address.html>

```
sudo arp-scan -I wlp0s20f3 --localnet
Interface: wlp0s20f3, datalink type: EN10MB (Ethernet)
Starting arp-scan 1.9 with 256 hosts (http://www.nta-monitor.com/tools/arp-scan/)
192.168.10.13      64:52:99:69:fd:20      The Chamberlain Group, Inc
192.168.10.20      c0:3f:d5:68:0c:cc      (Unknown)
```

192.168.10.50	fc:ec:da:c4:6e:55	(Unknown)
192.168.10.51	fc:ec:da:c4:77:0b	(Unknown)
192.168.10.154	ac:bc:32:7c:97:85	(Unknown)
192.168.10.163	dc:56:e7:49:61:d3	(Unknown)
192.168.10.156	70:81:eb:55:2a:a8	(Unknown)
192.168.10.179	9c:1c:12:c1:61:de	Aruba Networks
192.168.10.201	00:04:7d:0e:24:e5	Pelco
192.168.10.175	b8:8d:12:08:65:aa	Apple, Inc.
192.168.10.174	f0:76:6f:71:4b:10	(Unknown)
192.168.10.221	00:0c:29:4c:aa:4e	VMware, Inc.
192.168.10.239	10:1f:74:63:41:f8	Hewlett-Packard Company
192.168.10.254	08:5b:0e:d4:54:b2	Fortinet, Inc.
192.168.10.160	fc:e9:98:57:e9:ea	(Unknown)
192.168.10.162	ac:3c:0b:7a:13:e9	Apple

17 packets received by filter, 0 packets dropped by kernel  
 Ending arp-scan 1.9: 256 hosts scanned in 2.634 seconds (97.19 hosts/sec).  
 17 responded

Notice that we received several mac addresses that arp-scan listed as unknown. To update the oui.txt file that arp-scan uses run the following:

```
get-oui -v
Fetching OUI data from file://var/lib/ieee-data/oui.txt
Fetched 3823600 bytes
Opening output file ieee-oui.txt
Wide character in print at /usr/sbin/get-oui line 136.
24485 OUI entries written to file ieee-oui.txt
```

Now run the arp-scan again. Now we have a manufacture for all the device on my home network.

```
sudo arp-scan -I wlp0s20f3 --Localnet
Interface: wlp0s20f3, datalink type: EN10MB (Ethernet)
Starting arp-scan 1.9 with 256 hosts (http://www.nta-monitor.com/tools/arp-scan/)
192.168.10.13      64:52:99:69:fd:20  The Chamberlain Group, Inc
192.168.10.20      c0:3f:d5:68:0c:cc  Elitegroup Computer Systems Co.,Ltd.
192.168.10.50      fc:ec:da:c4:6e:55  Ubiquiti Networks Inc.
192.168.10.51      fc:ec:da:c4:77:0b  Ubiquiti Networks Inc.
192.168.10.163     dc:56:e7:49:61:d3  Apple, Inc.
192.168.10.154     ac:bc:32:7c:97:85  Apple, Inc.
192.168.10.160     fc:e9:98:57:e9:ea  Apple, Inc.
192.168.10.162     ac:3c:0b:7a:13:e9  Apple, Inc.
192.168.10.156     70:81:eb:55:2a:a8  Apple, Inc.
192.168.10.179     9c:1c:12:c1:61:de  Aruba Networks
192.168.10.201     00:04:7d:0e:24:e5  Pelco
192.168.10.221     00:0c:29:4c:aa:4e  VMware, Inc.
192.168.10.175     b8:8d:12:08:65:aa  Apple, Inc.
```

```
192.168.10.239      10:1f:74:63:41:f8  Hewlett Packard
192.168.10.254      08:5b:0e:d4:54:b2  Fortinet, Inc.
192.168.10.151      08:66:98:45:7f:86  Apple, Inc.
192.168.10.174      f0:76:6f:71:4b:10  Apple, Inc.
192.168.10.158      04:db:56:ed:ad:58  Apple, Inc.
```

```
18 packets received by filter, 0 packets dropped by kernel
Ending arp-scan 1.9: 256 hosts scanned in 2.630 seconds (97.34 hosts/sec). 18 responded
```

## Additional information

Project website - <https://github.com/royhills/arp-scan>

Project Wiki - <http://www.nta-monitor.com/wiki/>

arp-scan(1) - Linux man page - <https://linux.die.net/man/1/arp-scan>

Arp-Scan Command Tutorial With Examples - <https://www.poftut.com/arp-scan-command-tutorial-examples/>

My python wrapper for arp-scan - <https://github.com/rikosintie/arp-scan>

My blog on arp-scan - <https://mwhubbard.blogspot.com/2017/04/using-arp-scan-to-find-free-ip-address.html>

## grepcidr

<http://www.pc-tools.net/unix/grepcidr/>

This is a great little utility! It's like grep but for CIDR addresses.

Below is a fictitious example from a cutover. I intentionally removed four lines from the “show ip arp” file (arp-b4Cut.txt) to simulate some devices not being up afterward (arp-AfterCut.txt). The output is just the four lines I removed on purpose. NOTE: I did have to search/replace the “Internet “ at the beginning of each line since the -f option expects a list of IP addresses.

You can do a lot more, the man page (man grepcidr) has several examples and Google will find more.

```
grepcidr -vf arp-AfterCut.txt arp-b4Cut.txt
10.100.0.1      0    0015.5d0d.1d38  ARPA    Vlan700
10.100.0.11     214   d02b.20e1.b6bb  ARPA    Vlan700
10.100.0.12     0    Incomplete      ARPA
10.100.0.17     20   88b4.a677.fb6b  ARPA    Vlan700
```

## EXAMPLES from the MAN Page

grepcidr -f ournetworks blacklist > abuse.log

Find customers (CIDR ranges in file) that appear in blacklist

grepcidr 2001:db8::/32 log.1 log.2

Search for this IPv6 network inside two files

grepcidr 127.0.0.0/8 iplog

Searches for any localnet IP addresses inside the iplog file

```
grepcidr "192.168.0.1-192.168.10.13" iplog  
Searches for IPs matching indicated range in the iplog file
```

```
script | grepcidr -vf whitelist > blacklist  
Create a blacklist, with whitelisted networks removed (inverse)
```

```
grepcidr -f list1 list2  
Cross-reference two lists, outputs IPs common to both lists
```

Here is an example run against a file from a core switch with a few thousand ARP records. I wanted just addresses in the 10.140.0.0/23 subnet. Notice that I didn't have to remove the "Internet " from the beginning of each line.

```
grepcidr 10.140.0.0/23 arp-test.txt  
Internet 10.140.0.241 196 24e9.b33e.f080 ARPA TenGigabitEthernet2/16  
Internet 10.140.0.242 35 24e9.b33f.3d40 ARPA TenGigabitEthernet2/16  
Internet 10.140.0.244 137 24e9.b33f.3cc0 ARPA TenGigabitEthernet2/16  
Internet 10.140.0.245 51 24e9.b33f.3d80 ARPA TenGigabitEthernet2/16  
Internet 10.140.0.246 148 24e9.b33f.3dc0 ARPA TenGigabitEthernet2/16  
Internet 10.140.0.247 226 24e9.b33f.3c80 ARPA TenGigabitEthernet2/16  
Internet 10.140.0.248 25 24e9.b33e.f4c0 ARPA TenGigabitEthernet2/16  
Internet 10.140.0.249 40 24e9.b33e.f240 ARPA TenGigabitEthernet2/16  
Internet 10.140.0.250 230 24e9.b33e.f0c0 ARPA TenGigabitEthernet2/16  
Internet 10.140.0.254 - 3c08.f6ff.2640 ARPA TenGigabitEthernet2/16  
Internet 10.140.1.125 - 3c08.f6ff.2640 ARPA TenGigabitEthernet2/15  
Internet 10.140.1.126 0 906c.ac6d.6a73 ARPA TenGigabitEthernet2/15  
Internet 10.140.1.240 80 0011.212d.6701 ARPA Vlan556  
Internet 10.140.1.243 102 58f3.9cf5.0661 ARPA Vlan556  
Internet 10.140.1.251 113 24e9.b33f.3d00 ARPA Vlan556  
Internet 10.140.1.252 9 24e9.b33e.f200 ARPA Vlan556  
Internet 10.140.1.254 - 3c08.f6ff.2640 ARPA Vlan556
```

## nmap

From the introduction at [nmap.org](http://nmap.org)

Nmap ("Network Mapper") is a free and open source utility for network discovery and security auditing. Many systems and network administrators also find it useful for tasks such as network inventory, managing service upgrade schedules, and monitoring host or service uptime.

In addition to the classic command-line Nmap executable, the Nmap suite includes an advanced GUI and results viewer (Zenmap), a flexible data transfer, redirection, and debugging tool (Ncat), a utility for comparing scan results (Ndiff), and a packet generation and response analysis tool (Nping).

In my opinion, you need to be very proficient with nmap to be a good network engineer. If you aren't comfortable, there are a lot of websites with nmap tutorials and [udemy.com](https://udemy.com) has several inexpensive videos. The trick to udemy is to sign up for an account and then look for the courses you want. They

will be pretty expensive. Just add them to your cart but don't check out. Within a couple weeks you will get an email saying they are on sale for \$9.99 each.

## Installation

nmap is in Ubuntu's repository but it is usually quite old. For our purposes we will download the source and build the latest version.

Download from <https://nmap.org/download.html> to ~/Downloads  
The filename will be nmap-<version>.tar.bz2

### Decompress the tarball

```
tar xvjf nmap-<version>.tar.bz2  
cd nmap-<version>.tar.bz2  
.configure
```

If the configuration succeeds, an ASCII art dragon appears to congratulate you on successful configuration and warn you to be careful.

*Build nmap*  
*make*

The next step is a little different on Ubuntu because the root account is disabled by default. So, we have to enable root first:

```
sudo -i passwd root
```

You will be prompted to enter your password.

Now we can do the installation

Become a privileged user for system-wide install: su root  
*make install*

Congratulations! Nmap is now installed as /usr/local/bin/nmap! Run it with no arguments for a quick help screen.

Now we need to disable the root account

```
sudo passwd -l root
```

On Linux the default installation puts the files in the following locations:

/usr/share/nmap  
/usr/share/nmap/nselib - The LUA scripts used by nmap  
/usr/share/nmap/nselib/data - Data files needed for scripts  
/usr/share/nmap/scripts - The location of the NSE scripts

## Example 1

In the arp-scan example above we used my Macbook Air at 192.168.10.175 and Ping failed but arp-scan succeeded. With nmap, if you run it without sudo it uses ICMP for Host Discovery.

```
nmap -sn 192.168.10.175
```

Starting Nmap 7.60 ( https://nmap.org ) at 2019-02-10 20:20 PST

Note: Host seems down. If it is really up, but blocking our ping probes, try -Pn

Nmap done: 1 IP address (0 hosts up) scanned in 3.00 seconds

Now we run nmap with sudo which uses arp if the host is on the same subnet:

```
sudo nmap -sn 192.168.10.175
```

Starting Nmap 7.60 ( https://nmap.org ) at 2019-02-10 20:20 PST

Nmap scan report for 192.168.10.175

Host is up (0.012s latency).

MAC Address: B8:8D:12:08:65:AA (Apple)

Nmap done: 1 IP address (1 host up) scanned in 0.30 seconds

Notice that the host responded and nmap recorded the MAC Address. Keep this in mind when you use nmap on a LAN.

## Example 2

The beauty of nmap is that it has a scripting language and an active community. Here is a script for pulling information out of Ubiquiti devices. In this case I have two Ubiquiti Nano Loco access points in my home lab:

```
sudo nmap -sU -p 10001 --script ubiquiti-discovery.nse 192.168.10.50-51
```

Starting Nmap 7.60 ( https://nmap.org ) at 2019-02-10 21:51 PST

Nmap scan report for 192.168.10.50

Host is up (0.0042s latency).

```
PORt      STATE SERVICE
10001/udp open  ubiquiti-discovery
| ubiquiti-discovery:
|   protocol: v1
|   firmware: WA.ar934x.v8.5.11.39842.190109.1449
|   version: v8.5.11
|   uptime_seconds: 194815
|   uptime: 2 days 06:06:55
|   hostname: Office
|   product: N5L
|   essid: death2all
|   model: NanoStation 5AC loco
|   interface_to_ip:
|     fc:ec:da:c4:6e:55:
|       169.254.110.85
|       192.168.10.50
|   mac_addresses:
|     fc:ec:da:c4:6e:55
MAC Address: FC:EC:DA:C4:6E:55 (Ubiquiti Networks)
Service Info: OS: Linux
```

To compare the firmware versions of multiple devices:

```
sudo nmap -sU -p 10001 --script ubiquiti-discovery.nse -oG ubnt 192.168.10.50-51 |  
grep firmware  
|   firmware: WA.ar934x.v8.5.11.39842.190109.1449  
|   firmware: WA.ar934x.v8.5.11.39842.190109.1449
```

## ***Reporting***

Here are a few of the reporting tools I use for nmap

CSV output

<https://github.com/maaaaz/nmptocsv>

It can take the normal or XML output from nmap as input.

XML output

<https://github.com/honze-net/nmap-bootstrap-xsl/>

This is one of my favorites. You add --stylesheet nmap-bootstrap.xsl when you run the scan and then it creates a beautiful report that you can send in email.

Text output

<http://blog.techorganic.com/2012/09/15/parsing-nmaps-output/>

A bit awkward to run but a nice output

<https://github.com/actuated/nmap-grep>

Comprehensive parsing script for grepable Nmap output files. Provides a summary table, split hosts files, and URLs for web and SMB hosts.

## ***More Information***

<https://nmap.org/book/inst-source.html>

<https://www.tecmint.com/enable-and-disable-root-login-in-ubuntu/>

<https://www.udemy.com> - The complete nmap ethical hacking course

[www.packtpub.com](http://www.packtpub.com) - Nmap: Network Exploration and Security Auditing Cookbook Paulino Calderon

[www.packtpub.com](http://www.packtpub.com) - Mastering the Nmap Scripting Engine Paulino Calderon

## ***nping***

This utility is included with nmap. It allows you to ping devices using tcp or udp on any port. It's great when you have a firewall between you and the device that blocks ICMP.

## **Examples**

Ping using the HTTPS port tcp 443

```
sudo nping -tcp -p443 www.cisco.com
```

```
Starting Nping 0.7.60 ( https://nmap.org/nping ) at 2019-03-01 21:47 PST
SENT (0.1008s) TCP 192.168.10.225:25337 > 96.7.79.147:443 S ttl=64 id=32736 iplen=40
seq=3093962918 win=1480
RCVD (0.2788s) TCP 96.7.79.147:443 > 192.168.10.225:25337 SA ttl=56 id=0 iplen=44
seq=881035284 win=29200 <mss 1460>
SENT (1.1011s) TCP 192.168.10.225:25337 > 96.7.79.147:443 S ttl=64 id=32736 iplen=40
seq=3093962918 win=1480
RCVD (1.2988s) TCP 96.7.79.147:443 > 192.168.10.225:25337 SA ttl=56 id=0 iplen=44
seq=896669537 win=29200 <mss 1460>
SENT (2.1018s) TCP 192.168.10.225:25337 > 96.7.79.147:443 S ttl=64 id=32736 iplen=40
seq=3093962918 win=1480
RCVD (2.1187s) TCP 96.7.79.147:443 > 192.168.10.225:25337 SA ttl=56 id=0 iplen=44
seq=878740795 win=29200 <mss 1460>
SENT (3.1039s) TCP 192.168.10.225:25337 > 96.7.79.147:443 S ttl=64 id=32736 iplen=40
seq=3093962918 win=1480
RCVD (3.1388s) TCP 96.7.79.147:443 > 192.168.10.225:25337 SA ttl=56 id=0 iplen=44
seq=894586121 win=29200 <mss 1460>
SENT (4.1059s) TCP 192.168.10.225:25337 > 96.7.79.147:443 S ttl=64 id=32736 iplen=40
seq=3093962918 win=1480
RCVD (4.1588s) TCP 96.7.79.147:443 > 192.168.10.225:25337 SA ttl=56 id=0 iplen=44
seq=876671119 win=29200 <mss 1460>

Max rtt: 197.631ms | Min rtt: 16.872ms | Avg rtt: 95.972ms
Raw packets sent: 5 (200B) | Rcvd: 5 (230B) | Lost: 0 (0.00%)
Nping done: 1 IP address pinged in 4.20 seconds
```

## Traceroute for IPv4

Ubuntu dropped traceroute for IPv4 from the default installation so you need to install it before running an IPv4 traceroute.

From the man page

traceroute tracks the route packets taken from an IP network on their way to a given host. It utilizes the IP protocol's time to live (TTL) field and attempts to elicit an ICMP TIME\_EXCEEDED response from each gateway along the path to the host.

The only required parameter is the name or IP address of the destination host . The optional packet\_len'gth is the total size of the probing packet (default 60 bytes for IPv4 and 80 for IPv6). The specified size can be ignored in some situations or increased up to a minimal value.

Installation

```
sudo apt install traceroute
```

### Example

```
traceroute 107.170.203.230
traceroute to 107.170.203.230 (107.170.203.230), 30 hops max, 60 byte packets
 1 _gateway (192.168.10.254)  4.411 ms  3.997 ms  3.864 ms
```

```

2 * * *
3 acr02vntrca-gbe-1-1.vntr.ca.charter.com (96.34.97.109) 12.484 ms 13.003 ms 13.883 ms
4 24-180-19-29.static.lsan.ca.charter.com (24.180.19.29) 13.779 ms 13.827 ms 14.075 ms
5 bbr01olvemo-tge-0-1-0-11.olve.mo.charter.com (96.34.2.88) 21.333 ms 20.389 ms 24.041 ms
6 las-b21-link.telia.net (213.248.94.182) 21.259 ms 17.020 ms 23.072 ms
7 sjo-b21-link.telia.net (62.115.116.40) 32.029 ms las-b24-link.telia.net (62.115.136.47) 16.329
ms 15.544 ms
8 palo-b22-link.telia.net (62.115.125.0) 24.806 ms 24.414 ms 24.713 ms
9 * * *
10 * * *
11 * * *
12 107.170.203.230 (107.170.203.230) 23.746 ms !X 23.372 ms !X 23.595 ms !X

```

More information

traceroute(8) - Linux man page - <https://linux.die.net/man/8/traceroute>

A Practical Guide to (Correctly) Troubleshooting with Traceroute Troubleshooting with Traceroute  
[https://www.nanog.org/meetings/nanog47/presentations/Sunday/RAS\\_Traceroute\\_N47\\_Sun.pdf](https://www.nanog.org/meetings/nanog47/presentations/Sunday/RAS_Traceroute_N47_Sun.pdf)

## Tracepath

From the man page

It traces path to destination discovering MTU along this path. It uses UDP port port or some random port. It is similar to traceroute, only does not require superuser privileges and has no fancy options.

Options

-n

Do not look up host names. Only print IP addresses numerically.

-l (lowercase l)

Sets the initial packet length to pktlen instead of 65536 for tracepath or 128000 for tracepath6.

Installation

***sudo apt install iputils-tracepath***

Examples

Trace a path to my Digital Ocean CentOS VPS

***tracepath 107.170.203.230***

```

1?: [LOCALHOST]      pmtu 1500
1: _gateway          3.682ms
1: _gateway          1.472ms
2: no reply
3: acr02vntrca-gbe-1-1.vntr.ca.charter.com      12.945ms
4: 24-180-19-29.static.lsan.ca.charter.com      12.282ms
5: bbr01olvemo-tge-0-1-0-11.olve.mo.charter.com 17.153ms asymm 6
6: las-b21-link.telia.net          14.142ms
7: las-b24-link.telia.net          55.495ms asymm 8
8: palo-b22-link.telia.net        47.295ms
9: no reply
10: no reply
11: no reply
12: 107.170.203.230          35.467ms !H
Resume: pmtu 1500

```

Trace the same path without DNS resolution. This is probably more useful on your internal network than the public Internet.

```
tracepath -n 107.170.203.230
1?: [LOCALHOST]          pmtu 1500
1:  192.168.10.254      1.657ms
1:  192.168.10.254      1.467ms
2:  no reply
3:  96.34.97.109       13.074ms
4:  24.180.19.29        12.283ms
5:  96.34.2.88          11.745ms asymm  6
6:  213.248.94.182      16.195ms
7:  62.115.116.40       20.683ms
8:  62.115.125.0        22.429ms
9:  no reply
10: no reply
11: no reply
12: 107.170.203.230     26.307ms !H
    Resume: pmtu 1500
```

## Additional Information

[tracepath\(8\) - Linux man page](#)

[A Practical Guide to \(Correctly\) Troubleshooting with Traceroute](#)

## Socket Status (ss)

Similar to netstat, from the man page - another utility to investigate sockets. There are a myriad of options, as always, you can use the man pages to see all of them - man ss.

Here are some common options from the man page:

When no option is used ss displays a list of open non-listening sockets (e.g. TCP/UNIX/UDP) that have established connection.

-h, --help

Show summary of options.

-n, --numeric

Do not try to resolve service names.

-r, --resolve

Try to resolve numeric address/ports.

-a, --all

Display both listening and non-listening (for TCP this means established connections) sockets.

-l, --listening

Display only listening sockets (these are omitted by default).

-p, --processes

Show process using socket.

-t, --tcp

Display TCP sockets.

-u, --udp

Display UDP sockets.

-w, --raw

Display RAW sockets.

-x, --unix

Display Unix domain sockets (alias for -f unix).

-S, --sctp

Display SCTP sockets.

## STATE-FILTER

STATE-FILTER allows to construct arbitrary set of states to match. Its syntax is sequence of keywords state and exclude followed by identifier of state.

Available identifiers are:

All standard TCP states: established, syn-sent, syn-recv, fin-wait-1, fin-wait-2, time-wait, closed, close-wait, last-ack, listening and closing.

all - for all the states

connected - all the states except for listening and closed

synchronized - all the connected states except for syn-sent

bucket - states, which are maintained as minisockets, i.e. time-wait and syn-recv

big - opposite to bucket

## Usage Examples

`ss -t -a`

Display all TCP sockets.

`ss -t -a -Z`

Display all TCP sockets with process SELinux security contexts.

`ss -u -a`

Display all UDP sockets.

`ss -o state established '( dport = :ssh or sport = :ssh )'`

Display all established ssh connections.

`ss -x src /tmp/.X11-unix/*`

Find all local processes connected to X server.

An example looking for established tcp sockets

`->ss -t -p state established`

Recv-Q	Send-Q	Local Address:Port	Peer Address:Port	users:(( "firefox", pid=5498, fd=108 ))
0	0	192.168.10.183:52772	74.125.197.189:https	users:(( "dropbox", pid=4558, fd=69 ))
0	0	192.168.10.183:42748	162.125.34.129:https	users:(( "hirimain", pid=23454, fd=16 ))
0	0	192.168.10.183:35626	52.96.31.2:https	users:(( "hirimain", pid=23437, fd=21 ))
0	0	192.168.10.183:35620	52.96.31.2:https	users:(( "firefox", pid=5498, fd=98 ))
0	0	192.168.10.183:34802	52.42.195.146:https	users:(( "firefox", pid=5498, fd=115 ))
0	0	192.168.10.183:49238	192.241.168.185:https	users:(( "hirimain", pid=23437, fd=91 ))
0	0	192.168.10.183:55360	52.96.31.2:https	users:(( "simplenote", pid=23091, fd=92 ))
0	0	192.168.10.183:58596	192.0.84.247:https	users:(( "firefox", pid=5498, fd=120 ))
0	0	192.168.10.183:49780	172.217.14.78:https	users:(( "hirimain", pid=23437, fd=85 ))
0	0	192.168.10.183:49824	52.96.3.130:https	users:(( "firefox", pid=5498, fd=114 ))
0	0	192.168.10.183:47182	35.170.0.145:https	users:(( "firefox", pid=5498, fd=107 ))
0	0	192.168.10.183:60332	172.217.4.142:https	users:(( "dropbox", pid=4558, fd=162 ))
0	0	192.168.10.183:42054	162.125.34.129:https	users:(( "firefox", pid=5498, fd=103 ))
0	0	192.168.10.183:48602	54.152.127.232:http	

If any process, like SSH, is running as root, you will need to use sudo to see the Process ID and File Descriptor.

You can also use modifiers with ss. In this example I have two connections to a Cisco switch:

`3750x#who`

Line	User	Host(s)	Idle	Location
1 vty 0	mhubbard	idle	00:06:36	192.168.10.183

```
* 2 vty 1      mhubbard    idle          00:00:00 192.168.10.183
```

```
->ss -at '( dport = :ssh or sport = :ssh )'  
State  Recv-Q Send-Q Local Address:Port      Peer Address:Port  
LISTEN 0      128     0.0.0.0:ssh           0.0.0.0:*  
ESTAB   0      0       192.168.10.183:52968 192.168.10.52:ssh  
ESTAB   0      0       192.168.10.183:52970 192.168.10.52:ssh  
LISTEN 0      128     [::]:ssh             [::]:*
```

In this example I had been connected to a Microsoft server:

```
->ss -at '( dport = :445 or sport = :445 )'  
State  Recv-Q Send-Q Local Address:Port      Peer Address:Port  
CLOSE-WAIT 1      0       10.67.2.12:42318  10.67.2.224:microsoft-ds  
CLOSE-WAIT 1      0       10.67.2.12:42266  10.67.2.224:microsoft-ds  
CLOSE-WAIT 1      0       10.67.2.12:42324  10.67.2.224:microsoft-ds  
CLOSE-WAIT 1      0       10.67.2.12:42312  10.67.2.224:microsoft-ds  
CLOSE-WAIT 1      0       10.67.2.12:42274  10.67.2.224:microsoft-ds
```

You can also use operators like GT and LT

```
->ss -t sport gt :5000  
State  Recv-Q Send-Q Local Address:Port  Peer Address:Port  
CLOSE-WAIT 32      0       192.168.10.183:46760 162.125.2.3:https  
ESTAB   0      0       192.168.10.183:54650  1.1.1.1:https  
ESTAB   0      0       192.168.10.183:33432  192.241.168.185:https  
CLOSE-WAIT 32      0       192.168.10.183:42588 162.125.34.6:https  
ESTAB   0      0       192.168.10.183:42712  140.82.113.25:https  
CLOSE-WAIT 32      0       192.168.10.183:54128 162.125.2.4:https  
CLOSE-WAIT 32      0       192.168.10.183:40688 8.43.85.14:https
```

Notice that I am using Cloud

```
->ss -t dport lt :5000  
State  Recv-Q Send-Q Local Address:Port  Peer Address:Port  
CLOSE-WAIT 32      0       192.168.10.183:46760 162.125.2.3:https  
ESTAB   0      0       192.168.10.183:54650  1.1.1.1:https  
ESTAB   0      0       192.168.10.183:33432  192.241.168.185:https  
CLOSE-WAIT 32      0       192.168.10.183:36956 162.125.33.7:https  
ESTAB   0      0       192.168.10.183:42712  140.82.113.25:https  
CLOSE-WAIT 32      0       192.168.10.183:54128 162.125.2.4:https  
CLOSE-WAIT 32      0       192.168.10.183:40688 8.43.85.14:https
```

## Additional Information

[Linux for Network Engineers blog](#)

[How to get Socket Status Information \(ss\)](#)

## LinSSID

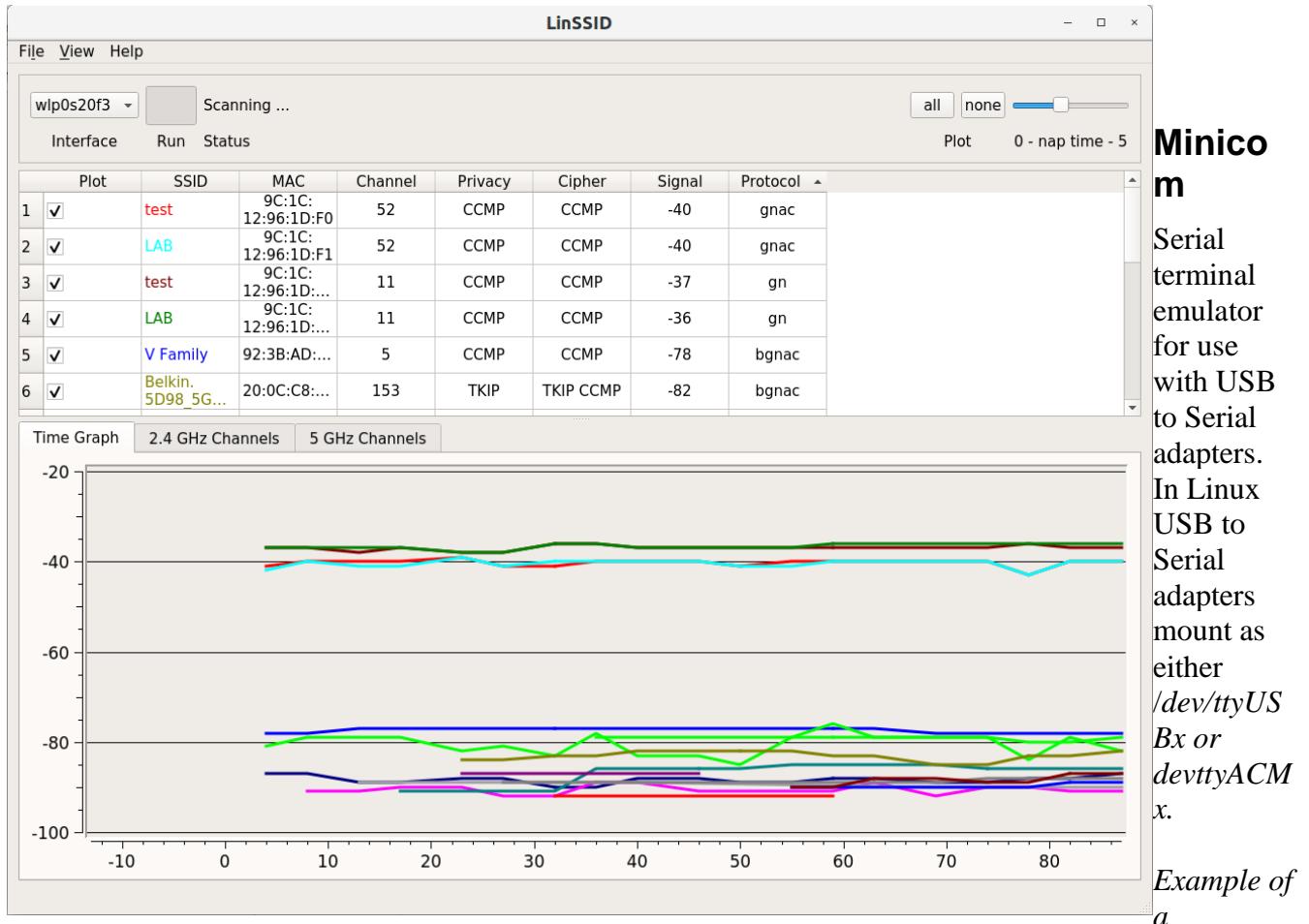
A tool similar to Metageek's inSSIDer. It scans for SSIDs and displays them graphically.

***sudo apt install linssid***

To run it:

***sudo linssid***

A GUI will open but it's paused by default. Click the button next to "paused" and the tool will start scanning. Here is screenshot:



*connecting to the USB mini port on a Cisco switch*

```
ls -l /dev/tty*
crw-rw---- 1 root      dialout 166,  0 Sep 30 13:52 /dev/ttyACM0
```

## Installation

***sudo apt install minicom***

## **Setup mode**

***sudo minicom -s***

scroll down to "Serial Port Setup"

Press "E", "C" to select 9600, N81.

Hit enter a couple times to exit setup and then select "Exit from Minicom"

Now you can run Minicom with:

***minicom***

and have it set to 9600, n81.

If you include a -C and a filename it will log all output.

***minicom -C test.txt***

Press ctrl+a, z to open the minicom menu.

Press ctrl+a, z, x to exit minicom.

Q: Why is error message "Device /dev/ttyUSB0 is locked." shown when executed 'minicom'?

A: It usually occurred by last abnormal close of minicom.

You can fix it by:

***rm /var/Lock/LCK..ttyUSB0***

<https://spremi.wordpress.com/2012/01/10/minicom-enable-access-to-normal-users/>

<https://openmaniak.com/minicom.php>

***sudo usermod -a -G dialout mhubbard***  
***sudo usermod -a -G dialout \$USER***

***ls -l /dev/ttyUSB0***

***cat /etc/group | grep dialout***  
dialout:x:20:mhubbard

***chown root /dev/tty8***  
***chgrp dialout /dev/tty8***  
***chmod 660 /dev/tty8***

## **Additional Information**

<http://manpages.ubuntu.com/manpages/trusty/man1/minicom.1.html> - Manual web page  
[http://processors.wiki.ti.com/index.php/Setting\\_up\\_Minicom\\_in\\_Ubuntu](http://processors.wiki.ti.com/index.php/Setting_up_Minicom_in_Ubuntu)

## **IPMI Tools**

Ubuntu has a package called ipmitool in it's repository. Using ipmitool you can list the users and add a new Administrative user even if you don't have valid credentials. For HP iLo this works on firmware up to v1.6 on iLo3 and up to v1.3 on iLo4. For iLo firmware greater than this for IPMI in general, you will need to have valid administrator credentials to make changes.

### ***Installation***

```
sudo apt-get install ipmitool
```

### **Examples**

Use the list command to see the current users. In this example, iLo version was vulnerable to the "Null Password" exploit so I used FluffyWabbit as the password.

```
ipmitool -I lanplus -C 0 -H 10.0.0.99 -U Administrator -P FluffyWabbit user list
```

Once you run the list command look for an ID that shows ( Empty User).

## **SNMP**

An oldie but goodie! SNMP is a valuable tool for a network engineer.

### ***Installation***

```
sudo apt install snmp
```

Due to license issues Canonical disables MIBs by default. To enable them simply comment out the MIBS line in /etc/snmp/snmp.conf

```
sudo gedit /etc/snmp/snmp.conf
```

```
# As the snmp packages come without MIB files due to license reasons, loading
# of MIBs is disabled by default. If you added the MIBs you can reenable
# loading them by commenting out the following line.
#mibs :
```

save and exit

## Cisco devices

Get the v2 mibs from Cisco's FTP site (<ftp://ftp.cisco.com/pub/mibs/v2/v2.tar.gz>)

To install the MIBs:

```
sudo mkdir /usr/share/snmp/mibs/cisco  
cd /usr/share/snmp/mibs/cisco
```

Copy v2.tar.gz that you just downloaded to /usr/share/snmp/mibs/cisco/v2.tar.gz

```
sudo tar xvfz v2.tar.gz  
sudo gedit /etc/snmp/snmp.conf
```

add this as the first line in the file:

```
mibdirs +/usr/share/snmp/mibs/cisco/auto/mibs/v2
```

## Examples

To display the arp table on a Cisco switch

```
snmpbulkwalk -v 2c -c Sup3rS3cr3t -Oxsq 10.243.1.1 \.1.3.6.1.2.1.3.1.1.2  
iso.3.6.1.2.1.3.1.1.2.82.1.10.243.1.1 "00 2A 10 77 8B 80 "  
iso.3.6.1.2.1.3.1.1.2.94.1.192.168.1.1 "00 2A 10 77 8B 80 "  
iso.3.6.1.2.1.3.1.1.2.96.1.10.243.2.1 "00 2A 10 77 8B 80 "  
iso.3.6.1.2.1.3.1.1.2.96.1.10.243.2.2 "38 0E 4D F9 82 F0 "  
iso.3.6.1.2.1.3.1.1.2.96.1.10.243.2.3 "38 0E 4D 6F 94 9C "  
iso.3.6.1.2.1.3.1.1.2.96.1.10.243.2.4 "6C B2 AE 09 F8 C4 "  
iso.3.6.1.2.1.3.1.1.2.96.1.10.243.2.5 "7C AD 74 50 10 42 "  
iso.3.6.1.2.1.3.1.1.2.96.1.10.243.2.6 "38 0E 4D F9 86 8C "  
iso.3.6.1.2.1.3.1.1.2.96.1.10.243.2.7 "6C B2 AE 09 F5 48 "
```

To display a switch serial number

```
snmpget -v 2c -c Sup3rS3cr3t -O s 10.207.1.26 .1.3.6.1.2.1.47.1.1.1.11.1001  
iso.3.6.1.2.1.47.1.1.1.11.1001 = STRING: "FDO1320X0XP"
```

To display the system information

```
snmpbulkwalk -v2c -Os -c Sup3rS3cr3t 10.243.1.1 system  
sysDescr.0 = STRING: Cisco IOS Software, c6880x Software (c6880x-ADVENTERPRISEK9-M),  
Version 15.2(1)SY7, RELEASE SOFTWARE (fc1)  
Technical Support: http://www.cisco.com/techsupport  
Copyright (c) 1986-2018 by Cisco Systems, Inc.  
Compiled Thu 05-Jul-18 04:32 by prod_rel_team  
sysObjectID.0 = OID: enterprises.9.1.1934  
sysUpTimeInstance = Timeticks: (416930035) 48 days, 6:08:20.35  
sysContact.0 = STRING:  
sysName.0 = STRING: TEST-6880x.home.local  
sysLocation.0 = STRING: <Home Lab>  
sysServices.0 = INTEGER: 78  
sysORLastChange.0 = Timeticks: (0) 0:00:00.00
```

## More information

<https://sourceforge.net/p/net-snmp/mailman/message/26565635/>

<https://networkengineering.stackexchange.com/questions/2900/using-snmp-to-retrieve-the-arp-and-mac-address-tables-from-a-switch>

<https://networkengineering.stackexchange.com/questions/2990/translating-snmpwalk-output-into-human-readable-format/775/>

Cisco MIB Locator - <https://mibs.cloudapps.cisco.com/ITDIT/MIBS/MainServlet>

<https://serverfault.com/questions/440319/did-not-find-zerodotzero-in-module-snmpv2-smi>

## Remove EXIF data from photos

While not specifically network related I wanted to include this tool. When you take a photo with a smart phone it saves a lot of metadata that isn't visible in the photo. This data includes the GPS coordinates where the photo was taken. Ask John McAfee about what can happen when you put a photo on the Internet with GPS data in it!

I subscribe to *Linux Pro Magazine* and I found this [article](#) on using exiftool to remove all metadata.

To clear the metadata of all files that start with airbox in a folder use the following:

```
for i in airbox*.jpg; do echo "Processing $i"; exiftool -all= "$i"; done
```

The article has more detail and examples to just remove geotags.

## Viewing EXIF data

exiftool can display all exif data using the following:

```
exiftool -a -u -g1 airbox.JPG
```

If you want to view the GPS data before removing it you can use:

```
exiftool IMG_4276.JPG | grep GPS
GPS Version ID          : 2.2.0.0
GPS Latitude Ref        : North
GPS Longitude Ref       : West
GPS Altitude Ref        : Above Sea Level
GPS Time Stamp          : 20:53:29.99
GPS Speed Ref           : km/h
GPS Speed               : 0
GPS Img Direction Ref  : True North
GPS Img Direction       : 270.8547973
GPS Dest Bearing Ref   : True North
GPS Dest Bearing        : 270.8547973
GPS Date Stamp          : 2019:06:25
GPS Horizontal Positioning Error: 10 m
GPS Altitude            : 361.7 m Above Sea Level
GPS Date/Time            : 2019:06:25 20:53:29.99Z
GPS Latitude             : 34 deg 7' 22.67" N
GPS Longitude            : 117 deg 19' 22.66" W
GPS Position             : 34 deg 7' 22.67" N, 117 deg 19' 22.66" W
```

There is also a built in tool called ImageMagick that can disp EXIF data. It's actually a suite of tools and the “identify” tool will dump all EXIF data including all GPS data. The tool is designed for photographers and dumps a huge amount of metadata about the camera’s settings. You can use grep to see just the EXIF data

```
identify -verbose IMG_4276.JPG | grep "exif:"
```

Here is the output with just the GPS data:

```
exif:GPSAltitude: 1537038/4249
exif:GPSAltitudeRef: 0
exif:GPSDateStamp: 2019:06:25
exif:GPSDestBearing: 527896/1949
exif:GPSDestBearingRef: T
exif:GPSImgDirection: 527896/1949
exif:GPSImgDirectionRef: T
exif:GPSInfo: 1884
exif:GPSLatitude: 34/1, 7/1, 2267/100
exif:GPSLatitudeRef: N
exif:GPSLongitude: 117/1, 19/1, 2266/100
exif:GPSLongitudeRef: W
exif:GPSSpeed: 0/1
exif:GPSSpeedRef: K
exif:GPSTimeStamp: 20/1, 53/1, 2999/100
exif:GPSVersionID: 2, 2, 0, 0
```

## More Information

[How to View EXIF data of a Photo from Terminal in Linux](#)

# DNS Enumeration Tools

Whether you are checking the security of your public facing servers, preparing to do a pentest or you are an attacker looking for a way in, DNS Enumeration is a good place to start. In this blog I am going to list my favorite enumeration tools.

Several of them are Linux only but will run under Windows using the Windows Subsystem for Linux. If you are running Windows 10 version 1809 you can go to the HyperV Gallery and install Ubuntu 1804. If you don't have Windows 10 you can also use Virtualbox for free to run Ubuntu as a VM.

## DNS Brute Force Wordlists

<https://packetstormsecurity.com/Crackers/wordlists/dictionaries/>  
<https://github.com/TheRook/subbrute/blob/master/names.txt>  
<https://github.com/danielmiessler/SecLists/tree/master/Discovery/DNS>

## DNSENUM

<https://github.com/fwaeytens/dnsenum>  
<https://www.youtube.com/watch?v=raGRvfpsaLw>

multithreaded perl script to enumerate DNS information of a domain and to discover non-contiguous ip blocks.

Think of dnsenum as a supercharged version of a whois query. It not only discovers all of the dns records but it goes a step further and attempts to use google to discover subdomains, discovers BIND versions and more.

```
dnsenum --dnsserver 9.9.9.9 --enum --noreverse -f /usr/share/dnsenum/dns.txt -f --subfile  
~/Desktop/DNS/dnsresults.txt vectorusa.com
```

## dnsmap

Dnsmap is a passive dns mapper that is used for subdomain bruteforce discovery.  
<https://github.com/makefu/dnsmap>

## dnsrecon

DNS enumeration script written in python for performing TLD expansion, SRV record enumeration, host and subdomain brute force, zone transfer, reverse lookup and general record identification.

Github site  
<https://github.com/darkoperator/dnsrecon>

Tutorial  
<http://www.rwbnetsec.com/dnsrecon/>

## **Installation**

Clone the project

```
git clone https://github.com/darkoperator/dnsrecon.git
```

```
cd dnsrecon
```

```
pip3 install -r requirements.txt --user
```

Basic Usage

```
python3 dnsrecon -d <domainname>
```

### **Example 1**

I ran this against the Win2008 DC in my home lab. You can see that it returned all the srv records, Name Server and DNS Server version in just a couple seconds.

```
python3 dnsrecon.py -d pu.pri
[*] Performing General Enumeration of Domain: pu.pri
[-] All nameservers failed to answer the DNSSEC query for pu.pri
[-] Could not Resolve SOA Record for pu.pri
[*]     NS win-l6hbt78g89g.pu.pri 192.168.10.221
[-]     Recursion enabled on NS Server 192.168.10.221
[*]     Bind Version for 192.168.10.221 b'Microsoft DNS 6.1.7601 (1DB15EC5)'
[-] Could not Resolve MX Records for pu.pri
[*]     A pu.pri 192.168.10.221
[*] Enumerating SRV Records
[*]     SRV _kerberos._tcp.pu.pri win-l6hbt78g89g.pu.pri 192.168.10.221 88 100
[*]     SRV _kerberos._udp.pu.pri win-l6hbt78g89g.pu.pri 192.168.10.221 88 100
[*]     SRV _gc._tcp.pu.pri win-l6hbt78g89g.pu.pri 192.168.10.221 3268 100
[*]     SRV _ldap._tcp.pu.pri win-l6hbt78g89g.pu.pri 192.168.10.221 389 100
[*]     SRV _ldap._tcp.ForestDNSZones.pu.pri win-l6hbt78g89g.pu.pri 192.168.10.221 389 100
[*]     SRV _ldap._tcp.pdc._msdcs.pu.pri win-l6hbt78g89g.pu.pri 192.168.10.221 389 100
[*]     SRV _kpasswd._udp.pu.pri win-l6hbt78g89g.pu.pri 192.168.10.221 464 100
[*]     SRV _kerberos._tcp.dc._msdcs.pu.pri win-l6hbt78g89g.pu.pri 192.168.10.221 88 100
[*]     SRV _kpasswd._tcp.pu.pri win-l6hbt78g89g.pu.pri 192.168.10.221 464 100
[*]     SRV _ldap._tcp.gc._msdcs.pu.pri win-l6hbt78g89g.pu.pri 192.168.10.221 3268 100
[*]     SRV _ldap._tcp.dc._msdcs.pu.pri win-l6hbt78g89g.pu.pri 192.168.10.221 389 100
[+] 11 Records Found
```

### **Example 2**

Google seems to have a much larger DNS infrastructure than I do!

```
python3 dnsrecon.py -d google.com
```

```
[*] Performing General Enumeration of Domain: google.com
```

```
[-] All nameservers failed to answer the DNSSEC query for google.com
```

```
[-] Could not Resolve SOA Record for google.com
```

```
[*]     NS ns4.google.com 216.239.38.10
```

```
[*]     NS ns4.google.com 2001:4860:4802:38::a
```

```
[*]     NS ns1.google.com 216.239.32.10
```

[\*] NS ns1.google.com 2001:4860:4802:32::a  
[\*] NS ns3.google.com 216.239.36.10  
[\*] NS ns3.google.com 2001:4860:4802:36::a  
[\*] NS ns2.google.com 216.239.34.10  
[\*] NS ns2.google.com 2001:4860:4802:34::a  
[\*] MX alt4.aspmx.l.google.com 74.125.31.27  
[\*] MX alt2.aspmx.l.google.com 173.194.219.27  
[\*] MX alt1.aspmx.l.google.com 74.125.129.26  
[\*] MX aspmx.l.google.com 74.125.197.26  
[\*] MX alt3.aspmx.l.google.com 173.194.205.26  
[\*] MX alt4.aspmx.l.google.com 2607:f8b0:400c:c02::1b  
[\*] MX alt2.aspmx.l.google.com 2607:f8b0:4002:c03::1b  
[\*] MX alt1.aspmx.l.google.com 2607:f8b0:4001:c15::1a  
[\*] MX aspmx.l.google.com 2607:f8b0:400e:c08::1a  
[\*] MX alt3.aspmx.l.google.com 2607:f8b0:400d:c02::1a  
[\*] A google.com 216.58.219.14  
[\*] AAAA google.com 2607:f8b0:4007:806::200e  
[\*] Enumerating SRV Records  
[\*] SRV \_ldap.\_tcp.google.com ldap.google.com 216.239.32.58 389 0  
[\*] SRV \_carddavs.\_tcp.google.com google.com 216.58.219.14 443 0  
[\*] SRV \_carddavs.\_tcp.google.com google.com 2607:f8b0:4007:806::200e 443 0  
[\*] SRV \_caldav.\_tcp.google.com calendar.google.com 216.58.219.14 80 0  
[\*] SRV \_caldav.\_tcp.google.com calendar.google.com 2607:f8b0:4007:806::200e 80 0  
[\*] SRV \_jabber.\_tcp.google.com alt3.xmpp-server.l.google.com 173.194.205.125 5269 0  
[\*] SRV \_jabber.\_tcp.google.com alt1.xmpp-server.l.google.com 74.125.129.125 5269 0  
[\*] SRV \_jabber.\_tcp.google.com alt2.xmpp-server.l.google.com 173.194.219.125 5269 0  
[\*] SRV \_jabber.\_tcp.google.com alt4.xmpp-server.l.google.com 74.125.31.125 5269 0  
[\*] SRV \_jabber.\_tcp.google.com xmpp-server.l.google.com 74.125.142.125 5269 0  
[\*] SRV \_xmpp-server.\_tcp.google.com alt4.xmpp-server.l.google.com 74.125.31.125 5269 0  
[\*] SRV \_xmpp-server.\_tcp.google.com alt2.xmpp-server.l.google.com 173.194.219.125 5269 0  
[\*] SRV \_xmpp-server.\_tcp.google.com alt1.xmpp-server.l.google.com 74.125.129.125 5269 0  
[\*] SRV \_xmpp-server.\_tcp.google.com alt3.xmpp-server.l.google.com 173.194.205.125 5269 0  
[\*] SRV \_xmpp-server.\_tcp.google.com xmpp-server.l.google.com 74.125.142.125 5269 0  
[\*] SRV \_jabber-client.\_tcp.google.com alt3.xmpp.l.google.com 173.194.205.125 5222 0  
[\*] SRV \_jabber-client.\_tcp.google.com alt3.xmpp.l.google.com 2607:f8b0:400d:c02::7d 5222 0  
[\*] SRV \_jabber-client.\_tcp.google.com xmpp.l.google.com 74.125.142.125 5222 0  
[\*] SRV \_jabber-client.\_tcp.google.com xmpp.l.google.com 2607:f8b0:400e:c08::7d 5222 0  
[\*] SRV \_jabber-client.\_tcp.google.com alt1.xmpp.l.google.com 74.125.129.125 5222 0  
[\*] SRV \_jabber-client.\_tcp.google.com alt1.xmpp.l.google.com 2607:f8b0:4001:c15::7d 5222 0  
[\*] SRV \_jabber-client.\_tcp.google.com alt2.xmpp.l.google.com 173.194.219.125 5222 0  
[\*] SRV \_jabber-client.\_tcp.google.com alt2.xmpp.l.google.com 2607:f8b0:4002:c03::7d 5222 0  
[\*] SRV \_jabber-client.\_tcp.google.com alt4.xmpp.l.google.com 74.125.31.125 5222 0  
[\*] SRV \_jabber-client.\_tcp.google.com alt4.xmpp.l.google.com 2607:f8b0:400c:c02::7d 5222 0  
[\*] SRV \_caldavs.\_tcp.google.com calendar.google.com 216.58.219.14 443 0  
[\*] SRV \_caldavs.\_tcp.google.com calendar.google.com 2607:f8b0:4007:806::200e 443 0  
[\*] SRV \_xmpp-client.\_tcp.google.com alt4.xmpp.l.google.com 74.125.31.125 5222 0  
[\*] SRV \_xmpp-client.\_tcp.google.com alt4.xmpp.l.google.com 2607:f8b0:400c:c02::7d 5222 0

```

[*] SRV _xmpp-client._tcp.google.com alt3.xmpp.l.google.com 173.194.205.125 5222 0
[*] SRV _xmpp-client._tcp.google.com alt3.xmpp.l.google.com 2607:f8b0:400d:c02::7d 5222 0
[*] SRV _xmpp-client._tcp.google.com alt1.xmpp.l.google.com 74.125.129.125 5222 0
[*] SRV _xmpp-client._tcp.google.com alt1.xmpp.l.google.com 2607:f8b0:4001:c15::7d 5222 0
[*] SRV _xmpp-client._tcp.google.com alt2.xmpp.l.google.com 173.194.219.125 5222 0
[*] SRV _xmpp-client._tcp.google.com alt2.xmpp.l.google.com 2607:f8b0:4002:c03::7d 5222 0
[*] SRV _xmpp-client._tcp.google.com xmpp.l.google.com 74.125.142.125 5222 0
[*] SRV _xmpp-client._tcp.google.com xmpp.l.google.com 2607:f8b0:400e:c08::7d 5222 0
[+] 37 Records Found

```

In order to run the Domain Name Brute-Force we need to type:

```
python3 dnsrecon.py -d <domain> -D <namelist> -t brtd
```

## dnswalk

Dnswalk is a DNS debugger. It performs zone transfers of specified domains, and checks the database in numerous ways for internal consistency, as well as accuracy.

## dnscan

<https://github.com/rbsec/dnscan>

## Anubis

<https://www.kitploit.com/2018/01/anubis-subdomain-enumeration-and.html>

<https://github.com/jonluca/Anubis>

```
anubis -h
```

Usage:

```
anubis -t TARGET [-o FILENAME] [-noipbarv] [-w SCAN] [-q NUM]
```

```
anubis -h
```

```
anubis --version
```

Options:

```
-h -help          show this help message and exit
```

```
-t -target        set target (comma separated, no spaces, if multiple)
```

```
-n -with-nmap     perform an nmap service/script scan
```

```
-o -output        save to filename
```

```
-i -additional-info    show additional information about the host from Shodan (requires API key)
```

```
-p -ip            outputs the resolved IPs for each subdomain, and a full list of unique ips
```

```
-a -send-to-anubis-db  send results to Anubis-DB
```

```
-r -recursive      recursively search over all subdomains
```

```
-w --overwrite-nmap-scan SCAN overwrite default nmap scan (default -nPn -sV -sC)
```

```
-v -verbose         print debug info and full request output
```

```
-q --queue-workers NUM override number of queue workers (default: 10, max: 100)
```

```
--version          show version and exit
```

## Example

```
anubis -t vectorusa.com -ipv -o temp.txt
```

-t - The target domain

-i - Show additional information from Shodan. If you don't have a Shodan.io API key you can register and get one for free at shodan.io

-p - outputs the resolved IPs for each subdomain, and a full list of unique ips

-v - Verbose mode

```
anubis -t vectorusa.com -ip -o temp.txt
```

```
d8888 888 d8b
d88888 888 Y8P
d88P888 888
d88P 888 88888b. 888 888 88888b. 888 .d8888b
d88P 888 888 "88b 888 888 888 "88b 888 88K
d88P 888 888 888 888 888 888 888 888 "Y8888b.
d8888888888 888 888 Y88b 888 888 d88P 888 X88
d88P 888 888 888 "Y88888 88888P" 888 88888P'
```

Searching for subdomains for 104.196.200.136 (vectorusa.com)

Testing for zone transfers

Searching HackerTarget

Searching VirusTotal

Searching NetCraft.com

Searching crt.sh

Searching DNSDumpster

Searching Anubis-DB

Searching Shodan.io for additional information

Server Location: Houston, US - 77002

ISP or Hosting Company: Google Cloud

Found 46 subdomains

```
-----
rdp.vectorusa.com: 40.135.170.153
quarantine.vectorusa.com:
vectorresources.vectorusa.com:
tor-conf.vectorusa.com: 40.135.170.143
phpmyadmin.vectorusa.com:
vlamailfrontend.vectorusa.com:
mailgw02.vectorusa.com:
vpn.vectorusa.com: 40.135.168.156
demo-expe.vectorusa.com:
mail.vectorusa.com: 40.135.170.129
link.vectorusa.com: 34.229.2.237
collab.vectorusa.com:
mailgw01.vectorusa.com: 40.135.170.158
dev.vectorusa.com: 40.135.170.151
vectorresourcesinc.vectorusa.com: 40.135.170.151
casht03.corp.vectorusa.com:
tor-av.vectorusa.com: 40.135.170.144
cppm.corp.vectorusa.com:
autodiscover.vectorusa.com: 52.96.26.168
```

```
webmail2k3.vectorusa.com:  
connect.vectorusa.com: 40.135.170.131  
sts.vectorusa.com: 40.135.170.154  
sip.vectorusa.com: 40.135.170.130  
casht01.corp.vectorusa.com:  
torexpe01.vectorusa.com:  
www.vectorusa.com: 104.196.200.136  
crm.vectorusa.com: 40.135.170.151  
tor-access.vectorusa.com: 40.135.170.142  
vlacrm01.vectorusa.com:  
my.vectorusa.com: 40.135.170.148  
lyncwebext.vectorusa.com:  
toraccess.vectorusa.com:  
tor-expe01.collab.vectorusa.com: 40.135.170.131  
tor-expe01.vectorusa.com: 40.135.170.131  
casht02.corp.vectorusa.com:  
.vectorusa.com:  
vflasharepoint.vectorusa.com:  
da.vectorusa.com:  
casht04.corp.vectorusa.com:  
sw.vectorusa.com:  
insight.vectorusa.com:  
vectorusa.com: 104.196.200.136  
archiver.vectorusa.com: 40.135.170.157  
lyncweb-ext.vectorusa.com: 40.135.170.149  
owa.vectorusa.com: 40.135.170.150  
mypassword.vectorusa.com: 40.135.170.133  
Found 19 unique IPs  
40.135.170.153  
34.229.2.237  
40.135.170.144  
104.196.200.136  
40.135.170.143  
40.135.170.154  
40.135.170.142  
40.135.170.151  
40.135.170.149  
40.135.170.133  
40.135.170.148  
40.135.170.131  
40.135.170.157  
52.96.26.168  
40.135.170.130  
40.135.170.129  
40.135.170.150  
40.135.170.158  
40.135.168.156  
Subdomain search took 0:00:01.778'
```

## DNSTracer

dnstracer determines where a given Domain Name Server (DNS) gets its information from, and follows the chain of DNS servers back to the servers which know the data.

<http://www.mavetju.org/unix/dnstracer.php>

```
sudo apt install dnstracer
```

### **Example**

```
dnstracer -s 1.1.1.1 -v vectorusa.com
Tracing to vectorusa.com[a] via 1.1.1.1, maximum of 3 retries
1.1.1.1 (1.1.1.1) IP HEADER
- Destination address: 1.1.1.1
DNS HEADER (send)
- Identifier: 0x260F
- Flags: 0x00 (Q)
- Opcode: 0 (Standard query)
- Return code: 0 (No error)
- Number questions: 1
- Number answer RR: 0
- Number authority RR: 0
- Number additional RR: 0
QUESTIONS (send)
- Queryname: (9)vectorusa(3).com
- Type: 1 (A)
- Class: 1 (Internet)
DNS HEADER (recv)
- Identifier: 0x260F
- Flags: 0x8080 (R RA)
- Opcode: 0 (Standard query)
- Return code: 0 (No error)
- Number questions: 1
- Number answer RR: 1
- Number authority RR: 0
- Number additional RR: 0
QUESTIONS (recv)
- Queryname: (9)vectorusa(3).com
- Type: 1 (A)
- Class: 1 (Internet)
ANSWER RR
- Domainname: (9)vectorusa(3).com
- Type: 1 (A)
- Class: 1 (Internet)
- TTL: 867 (14m27s)
- Resource length: 4
- Resource data: 104.196.200.136
Got answer
```

### **subbrute**

A DNS meta-query spider that enumerates DNS records, and subdomains.

Subbrute is also a python library so you can use it in your own tools.

Written in python, it should work on any OS and the github readme has detailed instructions on how to install it.

<https://github.com/TheRook/subbrute>

To install on Linux just clone the repository  
***git clone https://github.com/TheRook/subbrute.git***

The only requirement is python-dns.

Under Ubuntu/Debian all you need is:

***sudo apt install python-dnspython***

### **Examples**

Minimum usage to return all subdomains

***./subbrute.py google.com***

Tests multiple domains:

***./subbrute.py google.com gmail.com blogger.com***

or a newline delimited list of domains:

***./subbrute.py -t list.txt***

Also keep in mind that subdomains can have subdomains (example: \_xmpp-server.\_tcp.gmail.com):

***./subbrute.py gmail.com > gmail.out***

***./subbrute.py -t gmail.out***

The subdomains enumerated from previous scans can now be used as input to enumerate other DNS records. The following commands demonstrate this new functionality:

```
./subbrute.py google.com -o google.names  
...162 subdomains found...
```

```
./subbrute.py -s google.names google.com --type TXT  
google.com,"v=spf1 include:_spf.google.com ip4:216.73.93.70/31 ip4:216.73.93.72/31  
~all"  
adwords.google.com,"v=spf1 redirect=google.com"  
...
```

```
./subbrute.py -s google.names google.com --type CNAME  
blog.google.com,www.blogger.com,blogger.l.google.com  
groups.google.com,groups.l.google.com  
...
```

## **DNSDIAG**

<https://www.kitploit.com/2018/10/dnsdiag-dns-diagnostics-and-performance.html>

<https://github.com/farrokhi/dnsdiag>

<https://dnsdiag.org/>

From dnsdiag.org

Ever been wondering if your ISP is hijacking your DNS traffic (<https://decentralize.today/is-your-isp-hijacking-your-dns-traffic-f3eb7ccb0ee7#.fevks5wyc>)? Ever observed any misbehavior with your DNS responses? Ever been redirected to wrong address and suspected something is wrong with your DNS? Here we have a set of tools to perform basic audits on your DNS requests and responses to make sure your DNS is working as you expect.

## ***Installation***

```
git clone https://github.com/farrokhi/dnsdiag.git  
cd dnsdiag  
pip3 install -r requirements.txt
```

## ***dnsping***

This is ping for DNS servers.

The common switches are:

- 4 or -6 - Choose IPv4 or IPv6
- C 3 - Number of tries.
- s - DNS Server to use
- t - Record type. A is default. Others are MX, AAAA, NS, PTR,
- v - Verbose

Examples

A Record

```
python3 dnsping.py -4 -c 3 -v -s 9.9.9.9 -t A google.com  
dnsping.py DNS: 9.9.9.9:53, hostname: google.com, rdatatype: A  
34 bytes from 9.9.9.9: seq=0 time=15.305 ms  
google.com. 243 IN A 172.217.4.142  
flags: QR RD RA
```

AAAA Record

```
python3 dnsping.py -4 -c 3 -v -s 9.9.9.9 -t AAAA google.com  
dnsping.py DNS: 9.9.9.9:53, hostname: google.com, rdatatype: AAAA  
48 bytes from 9.9.9.9: seq=0 time=14.287 ms  
google.com. 252 IN AAAA 2607:f8b0:4007:801::200e  
flags: QR RD RA
```

NS Record

```
python3 dnsping.py -4 -c 3 -v -s 9.9.9.9 -t NS google.com  
dnsping.py DNS: 9.9.9.9:53, hostname: google.com, rdatatype: NS  
163 bytes from 9.9.9.9: seq=0 time=18.206 ms  
google.com. 300887 IN NS ns3.google.com.  
google.com. 300887 IN NS ns4.google.com.  
google.com. 300887 IN NS ns1.google.com.  
google.com. 300887 IN NS ns2.google.com.  
flags: QR RD RA
```

## ***dnstraceroute***

Like Traceroute but for DNS!

The common switches are:

- a --asn Turn on AS# lookups for each hop encountered
- c --count Maximum number of hops (default: 30)
- C --color Print colorful output
- s --server DNS server to use (default: first system resolver)
- x --expert Print expert hints if available

Example using Google DNS server.

```
sudo python3 dnstraceroute.py --expert -C -t A -s 8.8.4.4 facebook.com
dnstraceroute.py DNS: 8.8.4.4:53, hostname: facebook.com, rdatatype: A
1      _gateway (192.168.10.254) 2.959 ms
2      *
3      acr02vntrca-gbe-1-1.vntr.ca.charter.com (96.34.97.109) 11.931 ms
4      24-180-19-29.static.lsan.ca.charter.com (24.180.19.29) 14.211 ms
5      bbr01olvemo-tge-0-1-0-11.olve.mo.charter.com (96.34.2.88) 17.241 ms
6      bbr02chcgil-tge-0-2-0-1.chcg.il.charter.com (96.34.3.129) 14.121 ms
7      74.125.51.245 (74.125.51.245) 15.059 ms
8      108.170.247.129 (108.170.247.129) 15.077 ms
9      216.239.41.89 (216.239.41.89) 16.861 ms
10     google-public-dns-b.google.com (8.8.4.4) 14.975 ms
```

==== Expert Hints ====

[\*] No expert hint available for this trace

Example using OpenDNS DNS server. Note that OpenDNS was two hops closer than Google.

```
sudo python3 dnstraceroute.py --expert -C -t A -s 208.67.220.220 facebook.com
dnstraceroute.py DNS: 208.67.220.220:53, hostname: facebook.com, rdatatype: A
1      _gateway (192.168.10.254) 3.214 ms
2      *
3      acr02vntrca-gbe-1-1.vntr.ca.charter.com (96.34.97.109) 16.945 ms
4      24-180-19-29.static.lsan.ca.charter.com (24.180.19.29) 11.714 ms
5      bbr01olvemo-tge-0-1-0-11.olve.mo.charter.com (96.34.2.88) 17.250 ms
6      bbr02chcgil-tge-0-2-0-1.chcg.il.charter.com (96.34.3.129) 14.635 ms
7      peer1.rtr1.lax.opendns.com (206.223.123.85) 14.647 ms
8      resolver2.opendns.com (208.67.220.220) 16.493 ms
```

==== Expert Hints ====

[\*] No expert hint available for this trace

Example using Quad9 DNS server

```
sudo python3 dnstraceroute.py -a --expert -C -t A -s 9.9.9.9 facebook.com
dnstraceroute.py DNS: 9.9.9.9:53, hostname: facebook.com, rdatatype: A
1      _gateway (192.168.10.254) 2.008 ms
2      *
```

```

3      acr02vntrca-gbe-1-1.vntr.ca.charter.com (96.34.97.109) 13.022 ms
4      24-180-19-29.static.lsan.ca.charter.com (24.180.19.29) [AS20115 CHARTER-NET-HKY-NC -
Charter Communications, US] 72.408 ms
5      bbr01olvemo-tge-0-1-0-11.olve.mo.charter.com (96.34.2.88) 17.735 ms
6      bbr02chcgil-tge-0-2-0-1.chcg.il.charter.com (96.34.3.129) 16.433 ms
7      equinix-la.woodyn.net (206.223.123.110) 14.467 ms
8      dns.quad9.net (9.9.9.9) [AS19281 QUAD9-AS-1 - Quad9, US] 16.221 ms

```

==== Expert Hints ====

[\*] No expert hint available for this trace

## dnseval

dnseval is a bulk ping utility that sends an arbitrary DNS query to a give list of DNS servers. This script is meant for comparing response time of multiple DNS servers at once.

Example using public-servers.txt as the list of servers. I added 8.8.8.8/8.8.4.4 to the file.

`python3 dnseval.py -t AAAA -f public-servers.txt -c10 yahoo.com`

server	avg(ms)	min(ms)	max(ms)	stddev(ms)	lost(%)	ttl	flags
8.8.8.8	14.498	12.907	15.592	0.880	%0	1467	QR -- - RD RA -- --
8.8.4.4	15.667	13.452	27.905	4.350	%0	1053	QR -- - RD RA -- --
1.0.0.1	22.177	13.704	92.637	24.760	%0	1313	QR -- - RD RA -- --
1.1.1.1	14.178	13.112	15.532	0.967	%10	1311	QR -- - RD RA -- --
2606:4700:4700::1001	0.000	0.000	0.000	0.000	%100	N/A	-- - - - - - - - -
2606:4700:4700::1111	0.000	0.000	0.000	0.000	%100	N/A	-- - - - - - - - -
195.46.39.39	14.293	13.218	15.388	0.840	%0	1736	QR -- - RD RA -- --
195.46.39.40	14.572	13.018	16.015	0.890	%0	1735	QR -- - RD RA -- --
208.67.220.220	14.539	12.623	17.116	1.336	%0	947	QR -- - RD RA -- --
208.67.222.222	15.125	11.527	26.396	4.073	%0	947	QR -- - RD RA -- --
2620:0:ccc::2	0.000	0.000	0.000	0.000	%100	N/A	-- - - - - - - - -
2620:0:ccd::2	0.000	0.000	0.000	0.000	%100	N/A	-- - - - - - - - -
216.146.35.35	77.689	76.192	78.979	0.787	%0	457	QR -- - RD RA -- --
216.146.36.36	79.335	76.648	90.497	4.212	%0	1278	QR -- - RD RA -- --
209.244.0.3	14.708	13.288	17.192	1.232	%0	1405	QR -- - RD RA -- --
209.244.0.4	15.709	12.883	25.111	3.418	%0	709	QR -- - RD RA -- --
4.2.2.1	14.092	12.942	16.463	0.945	%0	1667	QR -- - RD RA -- --
4.2.2.2	13.648	12.941	14.500	0.547	%0	709	QR -- - RD RA -- --
4.2.2.3	14.383	12.609	16.922	1.168	%0	783	QR -- - RD RA -- --
4.2.2.4	14.176	13.088	15.746	0.746	%0	1404	QR -- - RD RA -- --
4.2.2.5	14.485	12.798	16.889	1.157	%0	783	QR -- - RD RA -- --
80.80.80.80	86.292	75.268	166.986	28.457	%0	1799	QR -- - RD RA -- --
80.80.81.81	77.751	73.909	89.048	4.314	%0	1799	QR -- - RD RA -- --
8.8.4.4	14.048	12.663	15.477	0.835	%0	965	QR -- - RD RA -- --
8.8.8.8	14.350	13.417	15.257	0.597	%0	801	QR -- - RD RA -- --
2001:4860:4860::8844	0.000	0.000	0.000	0.000	%100	N/A	-- - - - - - - - -
2001:4860:4860::8888	0.000	0.000	0.000	0.000	%100	N/A	-- - - - - - - - -
9.9.9.9	14.137	13.236	16.648	1.138	%0	393	QR -- - RD RA -- --
2620:fe::fe	0.000	0.000	0.000	0.000	%100	N/A	-- - - - - - - - -
149.112.112.112	14.097	12.741	15.663	0.833	%0	1225	QR -- - RD RA -- --

## CTFR

Abusing Certificate Transparency logs for getting HTTPS websites subdomains.

<https://github.com/UnaPibaGeek/ctfr>

```
~/tools/ctfr  
python3 ctfr.py --help  
python3 ctfr.py -d vvc.edu -o /home/ctfr/subdomains_vvc.txt
```

## References

DNS Enumeration Techniques in Linux  
<https://resources.infosecinstitute.com/dns-enumeration-techniques-in-linux/>

Is your ISP hijacking your DNS traffic? Is your ISP hijacking your DNS traffic?  
<https://decentralize.today/is-your-isp-hijacking-your-dns-traffic-f3eb7ccb0ee7#.fevks5wyc>

Wikipedia - Domain Name System  
[https://en.wikipedia.org/wiki/Domain\\_Name\\_System](https://en.wikipedia.org/wiki/Domain_Name_System)

What is DNS  
<http://www.howtogeek.com/122845/htg-explains-what-is-dns/>

Fortinet SSL VPN client  
You need to get the VPN client tar ball from someone with a Fortinet contract and then follow the instructions here:  
<http://kb.fortinet.com/kb/documentLink.do?externalID=FD39996>

I found this site that has the client available without a Fortinet account. I didn't use it so I can't vouch for it but it looks like he put a lot of effort into it.  
<https://hadler.me/2018/05/openfortigui-0-6-1/>

After the client is installed, I create a bash script for each customer. The contents look like this:  
cat company1.sh  
. /forticlientsslpn\_cli --server vpn.somecompany.com:443 --vpnuser michaelhubbard

Create the file in gedit, save and then run "chmod +x company1.sh" to make the script executable.

\*\*\*\*\*  
Horst - wifi tool [github.com/br101/horst](https://github.com/br101/horst)  
git clone --recursive <https://github.com/br101/horst>

\*\*\*\*\*  
HPE iLo Java Remote Console  
[HP gen8 ilo4 remote console problem](#)

Download Oracle Java JDK from  
<https://www.oracle.com/technetwork/java/javase/downloads/jdk8-downloads-2133151.html>  
I used Java SE Development Kit 8u192 jdk-8u192-linux-x64.tar.gz

Extract the file

```
tar zxvf jdk-8u192-linux-x64.tar.gz
```

From iLo, click on Remote Console, under Java Integrated Remote Console (Java IRC), click on Web Start and save the file to the downloads folder.

cd to jdk1.8.0\_192/bin and execute:

```
jdk1.8.0_192/bin$ ./javaws ~/Downloads/iLO-jirc.jnlp
```

You will have to accept the security warnings (An scream that HP hasn't written an HTML5 interface like Dell and IBM) and then an interface just like the .net interface will open.

## **MySQL**

<https://dev.mysql.com/doc/mysql-apt-repo-quick-guide/en/>

Go to

<https://dev.mysql.com/downloads/repo/apt/>

and download the mysql-apt-config\_0x.x.x.x\_all.deb file

```
sudo dpkg -i mysql-apt-config_0.8.10-1_all.deb
```

All components are selected. Hit OK

```
sudo apt-get update
```

```
sudo apt install mysql-server
```

```
sudo service mysql status
```

```
sudo service mysql stop
```

```
sudo service mysql start
```

```
sudo apt-get install mysql-workbench-community
```

## **TFTP**

Installing a tftp server

[Set-Up tftp Client Server](#)

```
sudo apt install -y tftpd-hpa
```

```
sudo systemctl enable tftpd-hpa
```

```
sudo systemctl restart tftpd-hpa
```

Modify /etc/default/tftp-hpa as follows

```
sudo nano /etc/default/tftpd-hpa
```

```
TFTP_USERNAME="tftp"
TFTP_DIRECTORY="/home/mhubbard/tftp-root"
TFTP_ADDRESS="0.0.0.0:69"
TFTP_OPTIONS="--secure --create"
```

Note: the “--create” in TFTP\_OPTIONS allows new files to be created in the “TFTP\_DIRECTORY”.

### TFTP Client

To download a file from a tftp server

tftp 10.56.239.150

tftp> get

(files) BRAD-6880x.wri

tftp> ?

tftp-hpa 5.2

To upload a file to a tftp server

put somefile.txt

NOTE: to copy a file to the server you must create an empty file in the “TFTP\_Directory” and set the file to be publicly writable. The easiest way to create the file is with “touch somefile.txt” and then “chmod 666 somefile.txt” to make it writable by anyone. Doing exa -l will list the permissions after the chmod:

.rw-rw-rw- 2.2k mhubbard 13 Dec 2018 somefile.txt

Commands may be abbreviated.

Commands are:

```
connect      connect to remote tftp
mode         set file transfer mode
put          send file
get          receive file
quit         exit tftp
verbose       toggle verbose mode
trace         toggle packet tracing
literal       toggle literal mode, ignore ':' in file name
status        show current status
binary        set mode to octet
ascii         set mode to netascii
rexmt         set per-packet transmission timeout
timeout       set total retransmission timeout
?            print help information
help          print help information
tftp>
```

\*\*\*\*\*

VMware Workstation Linux

allow promiscuous mode

If you want all users to be able to set the virtual network adapter (/dev/vmnet0 in our example) to promiscuous mode, you can simply run the following command on the host operating system as root:

```
chmod a+rwx /dev/vmnet0
```

Secure boot

Could not open /dev/vmmon: No such file or directory.

<https://kb.vmware.com/s/article/2146460>

```
sudo openssl req -new -x509 -newkey rsa:2048 -keyout vmmon.priv -outform DER -out vmmon.der -nodes -days 36500 -subj "/CN=VMware/"
```

```
sudo /usr/src/linux-headers-`uname -r`/scripts/sign-file sha256 ./vmmon.priv ./vmmon.der $(modinfo -n vmmon)
```

```
sudo /usr/src/linux-headers-`uname -r`/scripts/sign-file sha256 ./vmmon.priv ./vmmon.der $(modinfo -n vmnet)
```

```
sudo mokutil --import vmmon.der
```

```
*****
```

openSSH Server

With openSSH server you will be able to use your laptop as an SCP server.

<https://www.maketecheasier.com/setup-enable-ssh-ubuntu/>

Install the package

```
sudo apt install openssh-server
```

Configure

Make a backup copy of the config

```
sudo cp /etc/ssh/sshd_config /etc/ssh/sshd_config.factory-defaults
```

scp from laptop to switch

Add ip ssh logging events so that failed attempts show why

```
sudo service ssh start
```

```
sudo service ssh restart
```

On router

```
copy scp://mhubbard@192.168.10.100/tftp-root/YB_16_03_0007.swi YB_16_03_0007.swi
```

```
copy scp://mhubbard@10.42.52.172/tftp-root/cat3k_caa-universalk9.16.03.07.SPA.bin
```

Troubleshooting

Cisco switch doesn't have ip ssh version 2 enabled

```
%Error opening scp://@10.42.52.172/cat3k_caa-universalk9.16.03.07.SPA.bin (Undefined error)
```

```
systemctl status sshd
ssh.service - OpenBSD Secure Shell server
   Loaded: loaded (/lib/systemd/system/ssh.service; enabled; vendor preset: enabled)
   Active: active (running) since Wed 2018-12-12 21:31:56 PST; 21h ago
     Process: 19309 ExecReload=/bin/kill -HUP $MAINPID (code=exited, status=0/SUCCESS)
     Process: 19305 ExecReload=/usr/sbin/sshd -t (code=exited, status=0/SUCCESS)
  Main PID: 2184 (sshd)
    Tasks: 1 (limit: 4915)
   CGroup: /system.slice/ssh.service
           └─2184 /usr/sbin/sshd -D
```

Dec 13 18:46:11 1S1K-G5-5587 sshd[19813]: Protocol major versions differ for 10.40.250.40 port  
23825: SSH-2.0-OpenSSH\_7.6p1 Ubuntu-4ubuntu0.1 vs. SSH-

\*\*\*\*\*

Unicode characters

<https://help.ubuntu.com/community/ComposeKey>

<https://unicode-table.com/en/#control-character>

Unicode composition

Another means to enter non-keycap characters is to enter them as Unicode character number.

Press Shift+Ctrl+U, release U, enter the hexadecimal (0123456789abcdef) Unicode character code point, then release Shift+Ctrl. An underlined u followed by the number will be displayed as you type.

Alternatively, press (and release) Shift+Ctrl+U, then, while underlined u is displayed, enter the hexadecimal Unicode character code point followed by <Return>.

° = 00b0

\*\*\*\*\*

Docker

<https://docs.docker.com/install/linux/docker-ce/ubuntu/>

\*\*\*\*\*

Webmap

<https://github.com/Rev3rseSecurity/WebMap>

mkdir /tmp/webmap

docker run -d --name webmap -h webmap -p 8000:8000 -v /tmp/webmap:/opt/xml rev3rse/webmap

\*\*\*\*\*

HP Scanner/printer

<https://developers.hp.com/hp-linux-imaging-and-printing/install/install/index>

\*\*\*\*\*

git-cola - GUI for Git

<https://github.com/git-cola/git-cola>

Installation

sudo add-apt-repository ppa:pavreh/git-cola

sudo apt update

sudo apt install git-cola

\*\*\*\*\*

\*\*\*\*\*

Startech USB Crash Cart NOTECONS01

This is a cool little device that connects to your laptop using a USB 3 port and a server's VGA port/usb port. It then displays the server on your laptop eliminating the need for a keyboard/mouse and monitor.

Startech provides drivers for Debian and CentOS distros

<https://www.startech.com/Server-Management/KVM-Switches/Portable-USB-PS-2-KVM-Console-Adapter-for-Notebook-PCs~NOTECONS01#dnlds>

On 18.04 I had to follow the instructions here to make it work.

<https://ubuntuforums.org/showthread.php?t=2375927>

I solved the problem in this way:

moving the libz.so.1 from the folder and creating a link for libz.so.1 in lib/x86\_64-linux-gnu/libz.so.1.

In particular the commands are:

cd /opt/usb-crash-cart-adapter/20180327/guts/ (the directory in which is present libz.so.1)

sudo mv libz.so.1 libz.so.1.old

sudo ln -s /lib/x86\_64-linux-gnu/libz.so.1

\*\*\*\*\*

DNSTwist

<https://github.com/elceef/dnsth twist>

Domain name permutation engine for detecting typo squatting, phishing and corporate espionage.

sudo apt-get install python-dnspython python-geoip python-whois python-requests python-ssdeep  
python-cffi

Docker

If you use Docker, you can pull official image from Docker Hub and run it:

docker pull elceef/dnsth twist

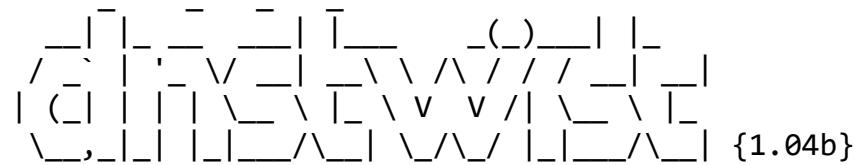
docker run elceef/dnsth twist example.com

## How to use

To start, it's a good idea to enter only the domain name as an argument. The tool will run it through its fuzzing algorithms and generate a list of potential phishing domains with the following DNS records: A, AAAA, NS and MX.

Next run it with the --registered switch to get just the domains that are registered.

```
docker run elceef/dnstwist vectorusa.com --registered
```



Processing 325 domain variants ..70% 12 hits (3%)

Original\* vectorusa.com 104.196.200.136 NS:ns1-09.azure-dns.com MX:vectorusa-com.mail.protection.outlook.com

Bitsquatting vektorusa.com 72.52.178.23 NS:ns1.parklogic.com

Hyphenation vector-usa.com 74.208.236.99 2607:f1c0:100f:f000::248 NS:ns1018.ui-dns.com MX:mx00.1and1.com

Insertion vectoreusa.com 184.168.221.48 NS:ns27.domaincontrol.com  
MX:mailstore1.secureserver.net

Omission vectorus.com 50.63.202.49 NS:ns53.domaincontrol.com MX:mailstore1.secureserver.net  
Omission vectorsa.com 141.8.230.20

Subdomain vec.torusa.com 18.211.9.206 NS:ns1.namebrightdns.com

Subdomain vect.orusa.com 72.52.4.119 NS:ns1.sedoparking.com MX:localhost

Subdomain vector.usa.com 69.10.42.209

Subdomain vectoru.sa.com 141.8.226.34 MX:sa-com-wildcard-null-mx.centralnic.net

Transposition vectoruas.com NS:ns1.moneytones.com

Vowel-swap victorusa.com 208.91.197.194 NS:dns01.gpn.register.com

\*\*\*\*\*

ngrok

ngrok.com

Download the binary and unzip it where you want to run it from.

On the dashboard, under connect your account, grab the auth token and run:

```
./ngrok authtoken 6itiYzJXwW4pAyRjKH7sq_4YDwSMfSsvFDCArqaGpk
```

Result:

Authtoken saved to configuration file: /home/mhubbard/.ngrok2/ngrok.yml

Ookla Speedtest  
sudo apt install speedtest-cli

From a terminal  
speedtest-cli  
Retrieving speedtest.net configuration...  
Testing from Spectrum (71.93.137.125)...  
Retrieving speedtest.net server list...  
Selecting best server based on ping...  
Hosted by Sprint (Anaheim, CA) [44.29 km]: 25.211 ms  
Testing download speed.....  
Download: 110.82 Mbit/s  
Testing upload speed.....  
Upload: 11.75 Mbit/s

---

---

HPE MyRoom  
<https://www.myroom.hpe.com/Download>

Allows HPE support staff remote access.  
Select the .deb file, once downloaded double click and install.

To execute, super key then type myhproom  
Select "login with key"

---

#### Useful Linux commands

Display a binary's dependencies. In Linux a binary executable is called an ELF. In Windows it's called an EXE.

In this example, exa is the ELF file, grep is used to display just the needed dependencies.

```
readelf -d /usr/local/bin/exa | grep NEEDED
0x0000000000000001 (NEEDED) Shared library: [libz.so.1]
0x0000000000000001 (NEEDED) Shared library: [libdl.so.2]
0x0000000000000001 (NEEDED) Shared library: [librt.so.1]
0x0000000000000001 (NEEDED) Shared library: [libpthread.so.0]
0x0000000000000001 (NEEDED) Shared library: [libgcc_s.so.1]
0x0000000000000001 (NEEDED) Shared library: [libc.so.6]
0x0000000000000001 (NEEDED) Shared library: [ld-linux-x86-64.so.2]
```

ldd recurses through all dependencies so the output can be different than readelf. It is not safe to use ldd on an untrusted binary. It can result in the execution of untrusted code!

```
ldd /usr/local/bin/exa
linux-vdso.so.1 (0x00007ffd5537b000)
libz.so.1 => /lib/x86_64-linux-gnu/libz.so.1 (0x00007fa2a6e33000)
libdl.so.2 => /lib/x86_64-linux-gnu/libdl.so.2 (0x00007fa2a6c2f000)
librt.so.1 => /lib/x86_64-linux-gnu/librt.so.1 (0x00007fa2a6a27000)
libpthread.so.0 => /lib/x86_64-linux-gnu/libpthread.so.0 (0x00007fa2a6808000)
libgcc_s.so.1 => /lib/x86_64-linux-gnu/libgcc_s.so.1 (0x00007fa2a65f0000)
libc.so.6 => /lib/x86_64-linux-gnu/libc.so.6 (0x00007fa2a61ff000)
/lib64/ld-linux-x86-64.so.2 (0x00007fa2a73a7000)
```

man ldd(1) says that:

In the usual case, ldd invokes the standard dynamic linker (see ld.so(8)) with the LD\_TRACE\_LOADED\_OBJECTS environment variable set to 1, which causes the linker to display the library dependencies. Be aware, however, that in some circumstances, some versions of ldd may attempt to obtain the dependency information by directly executing the program. Thus, you should never employ ldd on an untrusted executable, since this may result in the execution of arbitrary code.

```
objdump -p /usr/local/bin/exa | grep NEEDED
NEEDED libz.so.1
NEEDED libdl.so.2
NEEDED librt.so.1
NEEDED libpthread.so.0
NEEDED libgcc_s.so.1
NEEDED libc.so.6
NEEDED ld-linux-x86-64.so.2
```

---

Check the status of outbound ports

Black Hills Information Security maintains a service on the public Internet called Allports.Exposed that has all ports open. <https://www.blackhillsinfosec.com/poking-holes-in-the-firewall-egress-testing-with-allports-exposed/>.

You should block all outbound ports that are not required for your company at the firewall. All firewalls should be configured to block TCP ports 135/137,139 and 445 outbound. These are the Microsoft SMB ports. If left open, malware can trick a Microsoft OS to send its NTLM hash and username to a malicious server on the public Internet. Rob Vandenbrink of SANS has a good blog on determining what ports you can block at <https://isc.sans.edu/forums/diary/Egress+Filtering+What+do+we+have+a+bird+problem/18379/>.

Here is a short PowerShell port scanning script they provide that you can use to test ports 1-1024 against allports.exposed.

1. Open up a command terminal.
2. Type ‘powershell.exe -exec bypass’ and hit enter.

```
1..1024 | % {$test= new-object system.Net.Sockets.TcpClient; $wait =  
$test.beginConnect("allports.exposed",$_,,$null,$null); ($wait.asyncwaithandle.waitone(250,$false));  
if($test.Connected){echo "$_ open"}else{echo "$_ closed"} } | select-string " "
```

Alternatively, if you would like to just check for certain ports you can comma-separate each port you would like to scan at the beginning of the script in place of “1..1024”. For example, the following script will only scan ports 21, 22, 23, 25, 80, 443, and 1337.

```
21,22,23,25,80,443,1337 | % {$test= new-object system.Net.Sockets.TcpClient; $wait  
=$test.beginConnect("allports.exposed",$_,,$null,$null); ($wait.asyncwaithandle.waitone(250,$false));  
if($test.Connected){echo "$_ open"}else{echo "$_ closed"} } | select-string " "
```

Here is the same script, but this time we are testing the top 128 ports in use on the Internet as defined by the Nmap project.

```
80,23,443,21,22,25,3389,110,445,139,143,53,135,3306,8080,1723,111,995,993,5900,1025,587,8888,1  
99,1720,465,548,113,81,6001,10000,514,5060,179,1026,2000,8443,8000,32768,554,26,1433,49152,20  
01,515,8008,49154,1027,5666,646,5000,5631,631,49153,8081,2049,88,79,5800,106,2121,1110,49155,  
6000,513,990,5357,427,49156,543,544,5101,144,7,389,8009,3128,444,9999,5009,7070,5190,3000,543  
2,3986,13,1029,9,6646,49157,1028,873,1755,2717,4899,9100,119,37,1000,3001,5001,82,10010,1030,  
9090,2107,1024,2103,6004,1801,19,8031,1041,255,3703,17,808,3689,1031,1071,5901,9102,9000,210  
5,636,1038,2601,7000 | % {$test= new-object system.Net.Sockets.TcpClient; $wait  
=$test.beginConnect("allports.exposed",$_,,$null,$null); ($wait.asyncwaithandle.waitone(250,$false));  
if($test.Connected){echo "$_ open"}else{echo "$_ closed"} } | select-string " ""
```

---

#### Cisco Anyconnect client

<https://www.cocheno.com/2016/08/anyconnect-package-on-the-secure-gateway-could-not-be-located/>

Download the client from the cisco site. It will be similar to this AnyConnect Pre-Deployment Package (Linux 64-bit) anyconnect-linux64-4.6.04056-predeploy-k9.tar.gz.

Unpack it into a folder. Set the shell script as executable.  
chmod +x vpn\_install.sh

Run the shell script as root  
sudo ./vpn\_install.sh

When the script finishes you can use ./vpn to start the client from the terminal or hit the windows key and type “anyconnect” to start the GUI.

NOTE: you must install the Linux packages on the ASA or you will get a message “AnyConnect Package on the secure gateway could not be located”.

AnyConnect Headend Deployment Package (Linux 64-bit)  
anyconnect-linux64-4.6.04056-webdeploy-k9.pkg



## Security

Ubuntu is reasonably secure out of the box, but there are things you can do to tighten it up based on your threat environment. Below are some tools that will let you tune the security to your needs.

Setting up a secure Linux server – This blog is on a Debian server but most of it applies to Ubuntu on a laptop.

<http://www.linux-magazine.com/Issues/2016/183/Securing-Linux>

## Lynis

Lynis is an open source security tool. It helps with auditing systems running UNIX-alike systems (Linux, macOS, BSD), and providing guidance for system hardening and compliance testing. Here are the basics steps to use the Lynis.

### Installation

```
git clone https://github.com/CISOfy/lynis
cd lynis
```

### Lynis commands

audit system	Perform a system audit
show commands	Show available Lynis commands
show help	Provide a help screen
show profiles	Display discovered profiles
show settings	List all active settings from profiles
show version	Display current Lynis version

### Example,

```
./lynis audit system --quick --reverse-colors --auditor "Michael Hubbard"
```

Audit the system, don't pause between tests and list Michael Hubbard as the auditor in the report.

The report is long! If it finds something that doesn't fit best practices it will include a link explaining what it found and how to change it.

### Tips

- If Lynis is not installed as package (with included man page), use --man or nroff -man ./lynis.8
- For systems where the shell background is light, use --nocolors or --reverse-colors
- Use lynis show options to see all available parameters of Lynis

Getting started guide - <https://ciscofy.com/documentation/lynis/get-started/>

## Debian Security Analyzer

***sudo apt install debsecan***

<http://manpages.ubuntu.com/manpages/bionic/man1/debsecan.1.html>

<http://www.enyo.de/fw/software/debsecan/>

To use debsecan we need the version of Debian that Ubuntu is based on.

***cat /etc/debian\_version***

buster/sid

Now run debsecan with these switches to see packages with missing security updates. You can drop the --only-fixed to see all installed Debian packages.

***debsecan --suite buster --format packages --only-fixed***

To get details about security vulnerabilities use the –format detail switch.

***debsecan --suite sid --format detail***

Check the MD5 hashes of installed packages

***sudo apt install debsums***

<https://www.tecmint.com/check-verify-md5sum-packages-files-in-linux/>

***sudo debsums | grep FAILED***

This will show any files that fail the md5 check. If you think malicious activity has occurred you can quickly verify that all of your system files are valid and haven't been changed.

Tripwire

<https://computingforgeeks.com/how-to-install-and-configure-tripwire-on-ubuntu-18-04/>

<https://www.tecmint.com/install-tripwire-ids-intrusion-detection-system-on-linux/>

<https://github.com/Tripwire/tripwire-open-source>

Open Source Tripwire® is a security and data integrity tool for monitoring and alerting on file & directory changes. This project is based on code originally contributed by Tripwire, Inc. in 2000.

It is still maintained but if you want the lastest and greatest you can purchase a copy from

<https://www.tripwire.com/>. Configuration and usage is beyond the scope of this book. The Github site has installation and configuration information.

Samhain

***sudo apt install samhain***

<http://manpages.ubuntu.com/manpages/bionic/man8/samhain.8.html><https://www.linuxquestions.org/questions/linux-security-4/anyone-familiar-with-samhain-on-debian-ubuntu-893234/>

<https://www.la-samhna.de/samhain/>

From the project site:

The Samhain host-based intrusion detection system (HIDS) provides file integrity checking and log file monitoring/analysis, as well as rootkit detection, port monitoring, detection of rogue SUID executables, and hidden processes.

Check if a restart is needed to refresh any packages

There are two different tools for this task. Lynis checks to see if they are installed.

***sudo apt install needrestart***

<http://manpages.ubuntu.com/manpages/xenial/man1/needrestart.1.html>

*needrestart*

***sudo apt install debian-goodies***  
***sudo checkrestart***

apt-listchanges - Show new changelog entries from Debian package archives. Lynis checks to see if this tool is installed.

***sudo apt install apt-listchanges***

<http://manpages.ubuntu.com/manpages/bionic/man1/apt-listchanges.1.html>

Fail2Ban

***sudo apt install fail2ban***

<https://www.techrepublic.com/article/how-to-install-fail2ban-on-ubuntu-server-18-04/>

kolourpaint

Start up applications in tweak tool

Joe Editor

Debugging

tail -f /var/log/syslog

using Remote Desktop Gateway with Remmina

<https://blog.gpunktschmitz.com/1076-using-remote-desktop-gateway-with-remmina>

## **Appendix A – A Dell Laptop running Linux**

I did an impulse purchase of a Dell G5 15" gaming laptop from Walmart in September 2018. It has 16GB of RAM and an Nvidia GTX1060ti video card for hash cracking. Walmart had it on sale for \$879 with free "to Store" shipping.

It came with a 256GB m.2 SATA drive for the OS and a 1TB 2.5" spinning HD. I removed both and replaced them with a 500GB Samsung Evo 960 m.2 PCIe drive for the OS and a Samsung Evo 850 1TB SATA SSD for storage. Fry's had the 1TB SSD on sale for \$197.00.

This model is available from Dell with Ubuntu but Walmart only had the Windows version. Since it is available with Ubuntu I took the gamble that Ubuntu 18.04 would install and run without a lot of hassle. So far that has been pretty much the case.

### **Update the BiOS**

The first thing I did was check for a BiOS update. Dell has an exe file that gets executed by the UEFI so you don't have to boot into Windows! If you are a Windows user that doesn't seem like a big deal but if you don't dual boot and only have Linux installed it is a big step forward.

Here are some reference articles from Dell:

What is BIOS and How to Update the BIOS on Your Dell System

[https://www.dell.com/support/article/us/en/19/sln284433/what-is-bios-and-how-to-update-the-bios-on-your-dell-system?lang=en#How\\_to\\_linux](https://www.dell.com/support/article/us/en/19/sln284433/what-is-bios-and-how-to-update-the-bios-on-your-dell-system?lang=en#How_to_linux)

Updating the Dell BIOS in Linux and Ubuntu Environments

<https://www.dell.com/support/article/us/en/19/sln171755/updating-the-dell-bios-in-linux-and-ubuntu-environments?lang=en>

### **Download the BiOS**

<https://www.dell.com/support/home/us/en/19?app=drivers>

You need to select BiOS from the drop down. Once the page refreshes you have to click on the link for the BiOS. Once that page refreshes you have to click on the "View full driver details" link to see the hashes for the file. Pretty pathetic that the hashes are listed with the file but I guess they are Windows centric and most Windows users don't check the hashes of the files they download.

Manual "nomodeset" Kernel Boot Line Option for Linux Booting

<https://www.dell.com/support/article/uk/en/ukbsdt1/sln306327/manual-nomodeset-kernel-boot-line-option-for-linux-booting?lang=en>

I didn't have any issues with the installer but if you just get a black screen with a cursor you can use this link to work around it.

On the Dell G5 I purchased I did run into a couple problems installing Ubuntu 18.04. The first issue I ran into was that the BiOS didn't recognize the m.2 NVME drive I had installed. A quick google found this Dell KB - <https://www.dell.com/support/article/us/en/04/sln299303/loading-ubuntu-on-systems-using-pcie-m2-drives?lang=en> which resolved that issue.

After that it was just a matter of following the prompts on the installer. I didn't do full disk encryption on this laptop. I use Veracrypt to encrypt external USB drives where I store critical data so I didn't feel that I needed full disk encryption.

If you do want full disk encryption just select it during the installation. Ubuntu uses the Linux Unified Key Setup (LUKS) encryption specification. The advantage of LUKS is that it's used on Ubuntu and Red Hat so it's very well supported and there are a lot of technical articles available on the Internet.

Now it locks up after a few hours.

After I started using it on a daily basis it would lock up. Doing some Googling I found this article that resolved the problem. Turns out that the ACPI in the BiOS expected Windows and I needed to have Linux report an invalid version.

<https://askubuntu.com/questions/19486/how-do-i-add-a-kernel-boot-parameter#19487>

acpi\_osi!= acpi\_osi='Windows 2009'

<https://ubuntuforums.org/showthread.php?t=2396250>

### Grub customizer

This great little utility makes it easy to experiment with kernel options.

<http://tipsonubuntu.com/2018/03/11/install-grub-customizer-ubuntu-18-04-lts/>

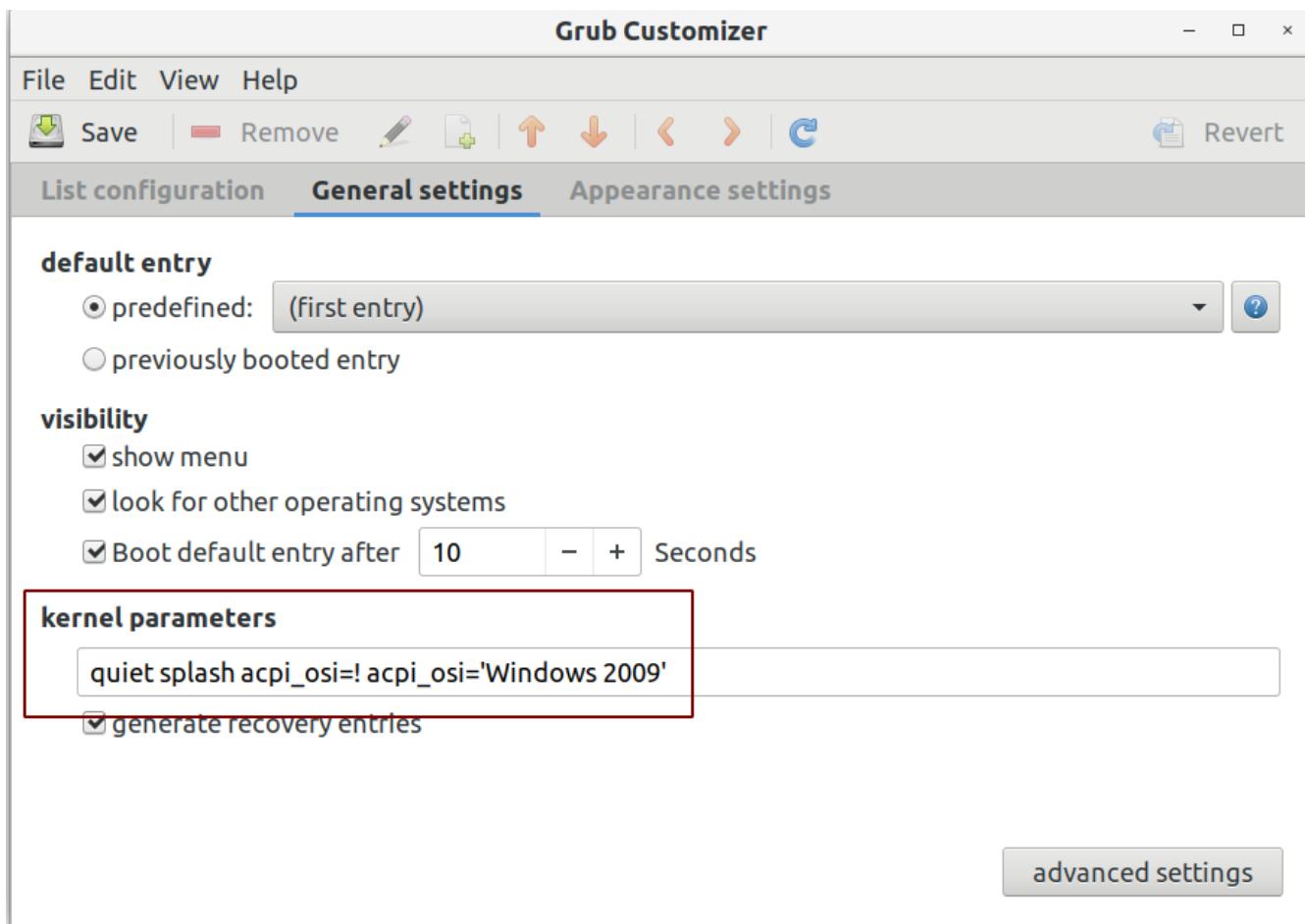
To install it do the following:

sudo add-apt-repository ppa:danielrichter2007/grub-customizer

sudo apt update

sudo apt install grub-customizer

Here is the interface for the grub customizer



You can see the kernel parameters I needed to add in the screenshot.

#### Showing the Grub menu

Normally, during install, if there are no other operating systems, the installer sets the boot variable to GRUB\_HIDDEN\_TIMEOUT=0 . I needed to see the menu for troubleshooting so I changed it to GRUB\_HIDDEN\_TIMEOUT=10 per the instructions in this article.

<https://ubuntuforums.org/showthread.php?t=2398935>

Click the advanced settings button and you will see several options that you can change using the check boxes.

Grub Customizer - settings		
Advanced		
	Add	Remove
is active	name	value
<input checked="" type="checkbox"/>	GRUB_DEFAULT	0
<input type="checkbox"/>	GRUB_TIMEOUT_STYLE	hidden
<input checked="" type="checkbox"/>	GRUB_TIMEOUT	10
<input checked="" type="checkbox"/>	GRUB_DISTRIBUTOR	`lsb_release -i -s 2> /dev/null    echo Debian`
<input checked="" type="checkbox"/>	GRUB_CMDLINE_LINUX_DEFAULT	quiet splash acpi_osi!=! acpi_osi='Windows 2009'
<input type="checkbox"/>	GRUB_BADRAM	0x01234567,0xfefefefe,0x89abcdef,0xefefefef
<input type="checkbox"/>	GRUB_TERMINAL	console
<input type="checkbox"/>	GRUB_GFXMODE	640x480
<input type="checkbox"/>	GRUB_DISABLE_LINUX_UUID	true
<input type="checkbox"/>	GRUB_DISABLE_RECOVERY	true
<input type="checkbox"/>	GRUB_INIT_TUNE	480 440 1

Close

Identifying if Ubuntu has been installed in UEFI mode

<https://help.ubuntu.com/community/UEFI>

Ubuntu installed in UEFI mode can be detected the following way:

if the /etc/fstab file contains an UEFI partition (mount point: /boot/efi) it uses the grub-efi bootloader (not grub-pc)

`cat /etc/fstab | grep efi`

```
# /boot/efi was on /dev/nvme0n1p1 during installation
UUID=CA3C-BE2C /boot/efi vfat umask=0077 0 1
```

from the installed Ubuntu, open a terminal (Ctrl+Alt+T) then type the following command:

```
[ -d /sys/firmware/efi ] && echo "Installed in UEFI mode" // echo "Installed in Legacy mode"
```

Installed in UEFI mode

## Nvidia and Intel built in graphics

Getting both the Intel and Nvidia graphic cards working isn't as easy as it is on Windows but Dell has a KB to help with that.

<https://www.dell.com/support/article/us/en/04/sln298475/a-guide-to-hybrid-video-on-dell-pcs-with-an-ubuntu-operating-system?lang=en>

Now that Ubuntu is working correctly, what's next?

Tweak your touchpad to taste in Linux

<https://www.techrepublic.com/article/tweak-your-touchpad-to-taste-in-linux/>

## Display Brightness

I had problems getting the Dell work with the built in gnome brightness settings. I found this little app the did the trick.

<https://www.debugpoint.com/2016/10/2-ways-fix-laptop-brightness-problem-ubuntu-linux/>

```
sudo add-apt-repository ppa:apandada1/brightness-controller
```

```
sudo apt update
```

```
sudo apt install brightness-controller
```

## Maximum Battery Life

<https://www.linuxuprising.com/2019/03/linux-battery-optimization-tool-tlp-12.html>