Goals

This document covers networking and network security basics in Linux. When you sit down at a Linux system for the first time, we want you to have the basics needed to learn what's running, assess what's on the system, and make basic changes.

The course is designed to have minimal requirements; students are not expected to be experts with Linux or networking.

Each student should have their own Linux system with sudo privileges. This can be done in multiple ways, and we cover multiple approaches below. These steps should be completed far in advance of the class. A student can take the class without one, though they will not get as much out of it.

It should be obvious, but this should not be a production Linux system. We'll be making changes as root, and while the chances are low, some of the following commands could make the system inaccessible or stop functioning normally.

Unlike a normal webcast, this will be almost entirely hands-on. The instructor will demonstrate the commands on his/her Linux system and the students will try them out themselves on their own Linux systems.

This class will focus on the command line. While it's only half of the Linux experience, it's 1) always available, 2) accessible via a keyboard and monitor, a cloud server console, and ssh, and 3) there even when the graphical desktop is intentionally disabled or unable to start because of a display issue.

How the day will work

You'll need a Linux system, and to get the most out of the day, you'll need to be able to run commands with root privilege (which means putting "sudo" in front of them or running them while logged in as root.

For each section:

- I'll give a quick introduction to the commands in question.

- I'll show you running them on my system.

- You can give them a try yourself. I encourage you to type them in by hand instead of cutting and pasting; it helps you learn and it avoids issues where characters like dashes or quotes get mangled by word processors.

- If you run into problems:
    - Recheck that you typed it correctly. The difference between a single and double dash, or using a double quote, single quote or backquote can easily mean the difference between a working command and a non-working command
    - If you're not able to figure out what's wrong, ask! You can use the "Questions" section of GotoMeeting for simpler questions; a volunteer will be there answering. If it's a question you'd like to ask of the other attendees, post it in the Wild West Hackin' Fest Discord, in the "#live-chat" channel. See the "#workshop-resources" channel for updates to this doc and other downloads. (If you're not yet on this Discord, there will be an invitation link in GotoMeeting's "Chat" area)

- You're welcome to try other options or look up more about the command with "`man command`".

- When you feel you've had enough time to try the command out, please press the "hand" icon in GotoMeeting. We'll be using the "hands raised" to get a sense of how many people are ready to go on to the next section, and when we move on we'll ask everyone to press the "hand" again to bring your hands down.

- You're absolutely welcome to continue working on a section even if the instructor is moving ahead! Remember that this class will be recorded, and when that recording is up, you can replay it if there were sections you'd like to go over again.

Getting a Linux system

Note on Operating system

The instructions in the class assume the student has Ubuntu Linux on their test system. If you choose to use a different flavor of Linux:

- Ubuntu and variants: Equivalent, no changes needed
- Debian variants, Raspbian, Raspberry Pi OS: Similar, a few commands may be different
- Fedora, Centos, RHEL, other Linuxes: Acceptable, admin commands will differ
- Anything else: You're on your own. You're welcome to use other Linux flavors, but expect to spend a little time looking up the equivalents to Ubuntu admin commands.

Where to install Linux

You can install Linux on 1) a physical computer with a hard drive you can write over with the Linux install, 2) a virtual machine running on VMWare workstation/player, Oracle Virtualbox, or KVM (you're on your own with other virtualization tools), or 3) A cloud server. We can't give instructions for all of them, but wanted to make sure you had an inexpensive option for taking this class, so we have instructions below for creating and deleting a cloud server in DigitalOcean. (We get nothing back for this or any other recommendation.)

Cloud server

Sign up for a Cloud instance running Linux. Make sure you can ssh to the system and run commands as root (test with "sudo whoami", which should return "root").

If you don't have a preferred cloud provider, we'll give instructions on setting up a server on DigitalOcean.

DigitalOcean cloud server

Go to https://www.digitalocean.com . Sign up for an account there, providing a credit card for billing. Once that's created, log in at https://cloud.digitalocean.com and follow these instructions to create a virtual machine for this class.

1. Click "Create" in the upper right and select Droplets (their name for cloud server).
2. Pick your preferred Linux operating system; Ubuntu 18.04.3 (LTS) or 20.04 (LTS) are fine.
3. Since we don't need high performance to do these labs, the "Basic" plan is fine. Below basic are the system sizes you can pick. Click on the left arrow to show the less expensive ones. I'd suggest the $10/month ($0.015/hour, 2GB memory, 1 cpu, 50GB

disk, 2TB transfer) option, though you can adjust as needed. At this rate you should be able to do the 4 hour lab for under US $0.10 .

4. No additional block storage is needed.
5. Pick a datacenter to host the machine; it should be geographically close to you. Under the physical location are sublocation numbers (1, 2, 3); pick on that's not greyed out.
6. No VPC is needed.
7. Check off "IPv6", but leave "User data" and "Monitoring" unchecked.
8. For authentication it's a little simpler to pick "Password" and enter a root password. Feel free to use SSH Keys if they're available.
9. Leave "How many droplets" at 1. Give the machine a hostname that reminds you of what it will be used for, like "linux-lab".
10. Leave Tags, Project, and Backups blank.
11. Digital Ocean will take anywhere from a few seconds to a few minutes to create your cloud server. You can watch the progress bar as it's being made, and when done, you'll see "linux-lab" with its IPv4 address in your list of available droplets. Click on the hostname to see the details of this system.
12. In the details page, you'll see the ipv4 and ipv6 addresses of your new cloud server.
13. To ssh to this system, run:

`ssh root@`*`either.ip.address`*

, accept the ssh host key fingerprint, and enter the password you provided above. If you're using a graphical ssh tool, fill in "root" for the username, either the ipv4 or ipv6 address for the system to ssh to, and choose connect.

14. You should see a welcome screen with basic system information and a "`root@linux-lab:~#`" prompt. Here's where you can run the commands we'll be doing in the lab. You can open up multiple ssh sessions and run separate commands in each.
15. When the lab is done you need to decide if you want to keep the cloud server and keep paying for it. If you're done with it, go back to https://cloud.digitalocean.com , click on the linux-lab droplet, click on "Destroy" in the center left of the display below the IPv4 address, and click on "Destroy this droplet". The cloud server, all your changes, and any files you placed on it will be permanently deleted. This is the *only* way to stop the billing for it; until you destroy it you'll continue to be charged for it at $10/month *whether the system is running or shut down*.

Your own Linux system/virtual machine

Install virtualization software. Both VMWare workstation and Oracle Virtualbox are good possibilities; both run on Windows, Mac, and Linux, and give you the ability to run Linux client instances, among others.

Download the installer file from your Linux distribution's web site (almost always with an ".iso" extension) and save it to disk.

Inside the virtualization program, create a new virtual machine. Generally this should be a 64 bit Linux instance, and you may even be able to specify exactly which Linux distribution you'll use. For the types of tasks we'll be doing, 4GB or memory should be fine, though you may be able to get away with less at the cost of some speed. A 20GB disk is likely appropriate. At some point in creating the virtual machine you'll be asked where to find the ".iso" file for the OS; unless you've moved it, it'll be in your Downloads directory.

Make sure that the ssh server/ssh daemon/openssh-server is installed and accepting connections. Once you're able to ssh into the system, make sure you can run commands under "sudo". Try running "sudo whoami" and make sure you see "root".

References:
https://ubuntu.com/download/

Topics

Getting started

- Booting up the first time
    - If you're logging in with keyboard and monitor
        - Connect HDMI cable and turn on monitor first
        - Connect keyboard and mouse
        - Do not connect network cable
        - Connect power cable
    - If you're logging in with ssh
        - Connect the ethernet cable to the system
        - If you're on a raspberry pi:

```
ssh pi@raspberrypi.local
```

        - If you're on a cloud server, the user is likely root:

```
ssh root@IP.add.ress
```

        - Otherwise check the documentation for your OS for the username:

```
ssh username@IP.add.ress
```

    - If you're logging in on the console
        - The console will show a request for a username
        - If you're on a Raspberry Pi, User "pi", password "raspberry"
        - If you're on a cloud server, the user is probably root
        - Otherwise, see the documentation for the default username and password
    - **Change password now**

```
passwd
```

    - Now connect Ethernet cable
    - Confirm Internet live

```
ping -c 5 8.8.8.8
```

        - If not, enable dhcp.
            - Ubuntu 20.04 edit /etc/netplan/99_config.yaml

```
network:
  version: 2
  renderer: networkd
  ethernets:
    enp0s3:
      dhcp4: true
    enp0s8:
      dhcp4: true
```

        - And reboot.
    - Patch system

```
sudo apt update && sudo apt full-upgrade
```

    - Install needed tools for today

```
sudo apt install -y lshw procps atop gkrellm nmap tcpdump tshark \
iproute2 net-tools netcat curl wget ufw rsync openssh-client
```

    - Optional graphical tool install (only if you have a graphical desktop)

```
sudo apt install -y wireshark gufw
```

    - Create a new user for experimentation

```
sudo useradd -m --groups sudo labuser
sudo passwd labuser
```

        - Log out and log back in as that user
        - Once in, use "sudo" in front of any commands that need root privileges

References:
https://www.digitalocean.com/docs/droplets/resources/console/

- Command line essentials
    - Please don't cut and paste the commands you see: 1) You learn better when you type them out, and 2) Quote characters may get screwed up in these documents

    - Output of `commandX` scrolls off the screen too fast

```
commandX | less -S
ps axf | less -S
```

    - I only want to see output lines with `abcd` in them

```
commandX | grep -i abcd
ps ax | grep agetty
```

    - I get an error saying I can't run commandX because it needs root privileges

```
sudo commandX
sudo ls -al /root
```

    - I want to save the output to a file to look at later

```
commandX >my_outputfile.txt
sudo lsof -n >lsofn.txt
```

    - I want to save the output and any error lines to a file to look at later

```
commandX >my_outputfile.txt 2>&1
```

    - I need more details about a commandX

```
man commandX
info commandX
commandX -h
commandX --help
```

    - I need to copy a file from this computer to a different computer

```
scp -p local_file.txt other_computer_name:/path/to/remote_file.txt
```

    - I need to copy a file from a different computer to this computer

```
scp -p other_computer_name:/path/to/remote_file.txt local_file.txt
```

    - Look at a file

```
less filename.ext
```

        - "q" to quit any time you're in less
        - Search for a line containing *searchstring*

```
/searchstring<Enter>
```

        - Search again for the same thing

```
/<Enter>
```

    - Simple editor: "nano"

- Immediately in edit mode
- Help on last two lines
- "^" is "ctrl" key, so "^X" is ctrl-X
- If the file you're editing is owned by root or a user other than the one you logged in as, prepend "`sudo`", such as "`sudo nano /etc/resolv.conf`"
- Can use "`nano` *`filename`*" when asked to run "`vim` *`filename`*"

- System hardware and networking
    - Seeing hardware

```
sudo lshw -short
sudo lshw
sudo lshw -short -C memory
sudo lshw -short -C processor
sudo lshw -short -C storage
sudo lshw -short -C disk,volume
sudo lshw -short -C network
sudo lshw -short -C display
```

    - Seeing storage

```
lsblk --fs
```

    - Basic network setup
        - Seeing and interpreting network settings

```
ifconfig
ifconfig -a
ip
ip address
route -n              #older systems, deprecated
ip route
```

            - Fields of interest
            - Finding your IP address
            - Loopback interface

        - Manually setting an IPv4 address and default route
            - Don't actually do the following until needed - it will kill your existing IP address

```
~~sudo ifconfig eth0 12.13.14.15 broadcast 12.13.14.255 up~~
~~sudo route add default gw 12.13.14.1~~
```

        - Do I have a live Internet connection?

```
ping -c 5 -n 8.8.8.8
```

        - Is my DNS working?

```
ping -c 5 -n www.google.com
```

        - Using external service to find your external IP
            - Why use an external service?

```
echo 'GET /' | nc icanhazip.com 80
```

        - Disabling wifi when connecting to Network Security switch
    - System monitoring

`top`

- `M` key to sort by memory usage
- `P` key to sort by processor usage

`atop`

- More detailed view
- Down to specific network interfaces and drives

`gkrellm`

- Graphical display
- Last 60 seconds
- Individual drives, CPUs, and network devices
    - Helps you understand the effects of running programs on system load
- Plugins available
- Windows, Mac, Linux
- Handy for monitoring multiple systems on a single screen

References:

https://ezix.org/project/wiki/HardwareLiSter
https://github.com/lyonel/lshw
https://www.atoptool.nl/
http://www.gkrellm.net/

Signposts

- Where am I?

```
pwd
ls -al
```

- Changing to a different directory

```
cd /tmp
ls -al
```

    - See what happens to your prompt?

- Top level

```
cd /
ls -al
```

- Handy file manager: mc

```
sudo apt install mc
mc
```

    - Function key hints at the bottom of the screen
    - Handy for copying/moving files
    - Can set either panel to be a remote system via ssh

```
<F9> l h machine_name <Enter>
```

    - Also installs a simple but powerful editor, mcedit

```
mcedit passer.py
```

- Important directories

/

    - Top level directory, everything on a Linux system is under here

/bin/
/sbin/
/usr/bin/
/usr/sbin/

    - Most programs are in these.

/boot/

    - Small disk partition that includes files needed to boot the system

/dev/

    - Directory holding objects that look like files but refer to specific devices on the system (like "/dev/sda" for the first scsi disk)

/etc/

    - Configuration files for services

/home/

- Directory above all home directories (except for the root user)

/lib/

- (and others starting with "lib") Library files that are shared between installed programs to save disk space and memory

/media/

- Files on removable media show up here

/mnt/

- Files on network shares primarily

/opt/

- Software not installed by the package manager generally goes under here

/proc/

- The kernel creates fake "files" under here; you can read from (and sometimes write to) these files to get system status and change configurations on the fly
  - Examples:

```
cat /proc/cpuinfo
cat /proc/uptime
```

  - First number: uptime in seconds, second number, total idle time on all processors

/root/

- "root" user's home directory

/run/

- Files specific to this boot (erased at boot time, so don't leave anything there)

/snap/

- Programs installed as "snaps" (Ubuntu packaging mechanism) go here.

/srv/

- Data directories for running services (web content, for example)

/sys/

- Like /proc/, a virtual filesystem showing system state

/tmp/

- Temporary files (commonly a ramdisk - erased at shutdown - so don't leave things there. Former employer. :-))

/usr/

- Primary storage for programs, libraries, support files that don't regularly change.

/var/

- System and application files that do change during normal operation.

References:

`man mc`

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

https://en.wikipedia.org/wiki/Filesystem_Hierarchy_Standard

SSH

- SSH access
  - Installing ssh client on different operating systems
    - Mac: already there
    - Windows: Putty, cygwin
    - iOS: SSH Term
    - Android: Termux
    - Linux: already there
  - Logging in as a normal user

```
ssh [user@]ip.address
```

  - Not working - why?

```
ssh -v [user@]ip.address
```

  - Changing your password

```
passwd [account_name]
```

  - Running privileged commands with sudo
  - Installing an ssh key
    - Do the following **on your laptop**
    - Check for existing keys

```
ls -al ~/.ssh/id_*
```

    - If you don't already have an ssh key:

```
mkdir -p ~/.ssh/
chmod 700 ~/.ssh/
ssh-keygen -b 3072 -f ~/.ssh/id_rsa -t rsa
```

    - Now install the key on the remote computer:

```
wget http://www.stearns.org/ssh-keyinstall/quick-ssh-keyinstall
chmod 755 quick-ssh-keyinstall
./quick-ssh-keyinstall root@linux-lab labuser@linux-lab
```

    - This only allows you to ssh from your laptop **to** linux-lab; it does not enable ssh'ing **from** linux-lab back to your laptop.
  - Running graphical tools with -X

```
sudo apt install xterm
```

    - Now, from your laptop:

```
ssh -X linux-lab xterm
```

    - Processing done remotely
    - Keystrokes, mouse movements and display locally
    - Heavy graphics/motion video mean **tons** of network traffic

References:
https://www.oreilly.com/library/view/ssh-the-secure/0596008953/
https://www.activecountermeasures.com/ssh-no-more-memorizing-ip-addresses/

Packages

- Installing new packages
  - What's already installed?
    - Debian/Ubuntu

```
sudo apt list --installed
```

    - Fedora/CentosRHEL

```
rpm -qa | sort
```

  - Package search
  - What package contains a program?
    - Debian/Ubuntu systems

```
sudo dpkg -S /bin/*rsync* /usr/bin/*rsync* /sbin/*rsync* /usr/sbin/*rsync* 2>/dev/null
apt search nginx
```

    - Fedora/Centos/RHEL

```
sudo yum provides *bin/rsync
sudo yum search nginx
```

  - Install the package
    - Debian/Ubuntu systems

```
sudo apt install packagename
```

    - Fedora/Centos/RHEL systems

```
sudo yum install packagename
```

  - Remove packages
    - Debian/Ubuntu systems

```
sudo apt remove packagename
sudo apt autoremove             #To remove dependencies
```

    - Fedora/Centos/RHEL systems

```
sudo yum remove packagename
```

- Package to try installing and removing: "alpine"

References:
https://blog.packagecloud.io/eng/2015/03/30/apt-cheat-sheet/
https://cheatsheet.dennyzhang.com/cheatsheet-apt-a4
https://access.redhat.com/sites/default/files/attachments/rh_yum_cheatsheet_1214_jcs_print-1.pdf
https://www.digitalocean.com/community/tutorials/package-management-basics-apt-yum-dnf-pkg

Alternatives:
Synaptic (GUI for apt)
dnf (replacement for yum. Yum is deprecated but still widely available on rpm systems)
rpm (Low level package management tool)

File transfer

- Copying files back and forth
  - Copying files to the Linux system

```
scp -p local.file linuxhostname:/tmp/
rsync -av -e ssh local.file linuxhostname:/tmp/
```

  - Copying files back from the Linux system

```
scp -p linuxhostname:/etc/services ./
rsync -av -e ssh linuxhostname:/etc/services ./
```

  - Making the second directory look like the first

rsync -av -e ssh /local/dir/ remotesystem:/remote/dir/

    - Both must end in a "/"
    - Optional --delete

- Downloading a file from a web server

```
curl -s http://www.stearns.org/test_linux_lab/downloadme.txt -O
wget -q http://www.stearns.org/test_linux_lab/downloadme2.txt
cat downloadme.txt
cat downloadme2.txt
```

- Viewing a web site in text mode

```
lynx http://www.google.com
links http://www.whitehouse.gov
```

  - Both may need to be installed

- Network mapping
  - Nmap basic host scan

```
nmap -A --top-ports 40 -sT 127.0.0.1
```

    - Can provide hostnames, IPs, and/or wildcards
  - **PERMISSION**
  - Reading results
  - Scanning for open tcp ports
  - Scanning for udp ports
    - UDP port issues

```
nmap -sU 127.0.0.1
```

References:
https://securitytrails.com/blog/top-15-nmap-commands-to-scan-remote-hosts

- Services
  - Listing open ports

```
sudo ss -p | less -S
sudo ss -tuanp | less -S
sudo netstat -anp
```

    - Listening on 0.0.0.0 vs [::]
    - Listening ports: "LISTEN"
    - Connections: "ESTAB" or "ESTABLISHED"
    - Shutting down: various "WAIT" labels
    - UDP ports: not as much detail

  - Seeing running processes

```
sudo ps axf
```

  - Listing running services
    - systemctl

```
sudo systemctl --all
sudo systemctl list-unit-files
```

    - service

```
sudo service --status-all
```

  - Running a server for testing
    - Installing the server

```
sudo apt install lynx nginx
```

    - Starting it

```
sudo systemctl start nginx
```

    - Stopping it

```
sudo systemctl stop nginx
```

    - Enabling it on next boot

```
sudo systemctl enable nginx
```

    - Disabling it on next boot

```
sudo systemctl disable nginx
```

    - Checking its current status

```
systemctl status nginx
```

  - Starting up a client program

```
lynx http://127.0.0.1      #"qy" to quit
```

    - Try lynx when nginx started, and again when stopped

  - Connecting two windows with netcat

- Two separate windows on the same system (parameter is "dash ell")

```
nc -l 9999
nc 127.0.0.1 9999
#Type something in the first window
#Type something else in the second
```

- On two different systems

```
nc -l 9999
nc ip.of.the.other.system 9999
#Type something in the first window
#Type something else in the second
```

References:
https://www.howtogeek.com/681468/how-to-use-the-ss-command-on-linux/
http://www.stearns.org/doc/nc-intro.current.html

Network topics

- Enabling routing
  - What's routing?
  - Is routing enabled?

```
cat /proc/sys/net/ipv4/ip_forward
```

  - Enable routing

```
echo 1 | sudo tee /proc/sys/net/ipv4/ip_forward
cat /proc/sys/net/ipv4/ip_forward
```

    - Why not `sudo echo 1 >/proc/sys/net/ipv4/ip_forward`

  - Disable routing

```
echo 0 | sudo tee /proc/sys/net/ipv4/ip_forward
cat /proc/sys/net/ipv4/ip_forward
```

  - Permanent change
    - Edit /etc/sysctl.conf

```
sudo nano /etc/sysctl.conf
```

    - Add either:

```
net.ipv4.ip_forward = 1
```

    - or:

```
net.ipv4.ip_forward = 0
```

    - And reboot.
- Other kernel settings: similar command line and sysctl syntax
  - As a general rule it's safe to cat any of the variables under /proc/

```
cat /proc/uptime           #Like the uptime command
cat /proc/modules     #Like lsmod
cat /proc/interrupts
```

  - Find the PID of an interesting process with:

```
ps axf
```

  - See the process details with:

```
cd /proc/pid/
ls -al
cat cmdline
ls -al exe
```

- Firewalling
  - UFW vs netfilter vs iptables vs nftables
  - Command line ufw and/or graphical gufw (http://gufw.org/)
  - Live blocking an open service

        - Opening needed ports and blocking by default
        - Firewalling a host (INPUT/OUTPUT) vs firewalling a router (FORWARD)
        - List firewall rules

```
iptables -L -n
```

        - List firewall rules with counts of the packets that matched

```
iptables -L -nxv
```

        -
        - Block incoming traffic to a service
            - Create

```
iptables -I INPUT -p tcp --dport 80 -j DROP
```

            - Log traffic but no block

```
iptables -I INPUT -p tcp --dport 80 --syn -j LOG
```

            - Delete

```
iptables -D INPUT -p tcp --dport 80 --syn -j LOG
iptables -D INPUT -p tcp --dport 80 -j DROP
```

        - Block incoming attacker
            - Create

```
iptables -I INPUT -s 68.183.227.196 -p tcp --dport 22 -j DROP
```

            - Log traffic but no block

```
iptables -I INPUT -s 68.183.227.196 -p tcp --dport 22 --syn -j LOG
```

            - Delete

```
iptables -D INPUT -s 68.183.227.196 -p tcp --dport 22 --syn -j LOG
iptables -D INPUT -s 68.183.227.196 -p tcp --dport 22 -j DROP
```

        - Order matters
        - Option of using firewall with cloud provider/host machine

    - Setting up tunneling
        - ssh
            - Encrypt traffic going to a remote web server

```
#On your laptop, in three separate windows
sudo tcpdump -i eth0 -qtnp 'tcp port 80'
ssh -L 8000:127.0.0.1:80 linux-lab
lynx http://127.0.0.1:8000
```

            - Why don't we see port 80 traffic going to the lab machine?

    - Running a network monitor
        - Zeek/Bro
            - Not included in Ubuntu any more
            - To install (thanks to Jamey!)

```
echo 'deb http://download.opensuse.org/repositories/security:/zeek/xUbuntu_20.04/ /' | sudo
tee /etc/apt/sources.list.d/security:zeek.list
```

```
curl -fsSL https://download.opensuse.org/repositories/security:zeek/xUbuntu_20.04/Release.key
| gpg --dearmor | sudo tee /etc/apt/trusted.gpg.d/security:zeek.gpg > /dev/null

sudo apt update

sudo apt install zeek zeekctl
```

- Start with

```
sudo zeekctl deploy
```

- All commands that used to have "bro" in the name now have "zeek".

References:

`man sysctl`

https://netfilter.org/

https://netfilter.org/projects/nftables/index.html

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

Alternatives:

Snort (www.snort.org)

Suricata (https://suricata-ids.org/)

Openvpn (https://openvpn.net/)

Packets

- What network interfaces do I have?

```
ifconfig
ifconfig -a
```

  - Difference?
- Packet capture and replay
  - Watching packets with tcpdump

```
sudo tcpdump -qtnp -i eth0
```

  - Only looking at specific packets with a filter

```
sudo tcpdump -qtnp -i eth0 'not tcp port 22'
```

    - Why the filter?
  - Saving packets to disk

```
sudo tcpdump -qtnp -i eth0 -c 100 -w saved_packets.pcap
```

  - Reading packets from disk later

```
sudo tcpdump -qtnp -r saved_packets.pcap
```

- Analysis
  - Tshark (command line) and wireshark (graphical)
  - Passer
  - And *tons* of others.

References:
http://www.stearns.org/doc/pcap-apps.html (not updated since 2008)

Locking down the system

- Enable firewall
    - Use gufw

```
sudo apt install gufw
#From your laptop
ssh -X linux-lab gufw
```

- Patch

```
sudo apt update && sudo apt -y upgrade && sudo apt -y autoremove
```

- Set up auto patching

```
sudo apt install unattended-upgrades
```

- Restrict to just ssh key login
    - Can every account log in with an ssh key?
    - Do the admins have the ability to run commands under sudo?
    - If so, disable passwords:

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

    - Edit the PasswordAuthentication line to be

```
PasswordAuthentication no
```

    - And restart sshd:

```
sudo systemctl restart sshd
```

    - (Note, that will *not* kill your active connections, including the one you're typing on)

- Restrict root login

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

    - Edit the PermitRootLogin line to be

```
PermitRootLogin prohibit-password
```

    - And restart sshd:

```
sudo systemctl restart sshd
```

    - (Again, won't kill your active connections, including the one you're typing on)

- Restrict ssh access to just a few ips
    - https://cloudcone.com/docs/article/how-to-restrict-ssh-access-only-to-specific-ips/

- Remove unneeded packages
    - Planning to strip down? Start with the "server" or "cloud" option instead of desktop.
    - Use apt or yum commands above to find and remove unneeded packages

- Disable unneeded network services
    - List open ports
    - Decide which ones you need
    - For the rest, find the service that started them and shut the service down
        - Remember to both *stop* and *disable* the service

- Encrypted root filesystem
    - Easy during install
    - Cloud server: need to connect to console to type root filesystem passwd, so may not be available

References:
https://ubuntu.com/server/docs/package-management
https://libre-software.net/ubuntu-automatic-updates/

Advanced topics:
Require 2nd factor on login
Log remotely

Common problems

- Drive is full
    - See usage

```
df -h
df -i
```

    - Find top level directory that's the problem

```
du -sh /*
```

    - Repeat for next level down

```
du -sh /var/*
```

    - Note; ignore types cgroup, squashfs, tmpfs, profs, debugfs, tracefs, fusectl, configfs, binfmt_misc, udev

- Can't seem to do DNS lookups
    - Nameservers are defined in /etc/resolv.conf (Note, **not** "resolve")

less /etc/resolv.conf

    - "nameserver a.b.c.d" lines list dns servers in order of preference

```
nameserver 2001:558:feed::1
nameserver 2001:558:feed::2
nameserver 75.75.75.75
nameserver 75.75.76.76
```

    - This file commonly rewritten when connecting to or disconnecting from a network
    - First, can you reach these machines at all? Ping each one (using the addresses you found in /etc/resolv.conf):

```
ping6 -c 5 -n 2001:558:feed::1
ping -c 5 -n 75.75.75.75
```

    - Check that you can get a DNS answer from each with:

```
dig +short @75.75.75.75 www.whitehouse.gov. A
```

    - If one's not working, remove it from /etc/resolv.conf

```
sudo nano /etc/resolv.conf
```

- Diagnosing problems
    - Kernel log

```
dmesg
```

        - Right after boot contains the boot-time messages that give you hints about hardware and filesystem issues
        - Later on, contains security and critical messages

    - Regular logs

```
tail -f /var/log/messages /var/log/syslog /var/log/auth.log
```

- Who's holding a file open?

```
sudo lsof -n >lsofn
cat lsofn | less -S
cat lsofn | grep -v 'mem.*/usr/lib/x86_64-linux-gnu/' | less -S
```

References:

```
man df
man ping
man dig
man dmesg
man lsof
```

Advanced topics

- Strace
    - Runs a program and shows what system calls it's making
    - Crashing chat program
    - Watch a running program
        - Find its process ID

```
ps axf | less -S
```

        - Watch just file accesses:

```
strace -p pid --trace=file
```

        - All system calls

```
strace -p pid
```

        - Save output to a file on disk

```
strace -p pid -o my_trace.txt
less -S my_trace.txt
```

        - With all of the above, press Ctrl-C to stop strace, but leave the original program running
        - To start and watch a running program (similar options, but not -p)

```
strace sleep 10
```

References:

```
man strace
```

- Ansible
    - All of this is on your laptop
    - Install ansible

```
sudo apt install ansible
```

    - Create /opt/local/etc/ansible/hosts, and add (substituting the IP of your linux system)

```
[labsystems]
12.13.14.15
```

    - Run inventory gathering

```
ansible labsystems -m setup | less -S
```

        - Happens over an ssh connection
    - This isn't 5% of what ansible does

        - Powerful tool for provisioning
        - Takes a system and a profile; checks that the system matches the profile, and makes any needed changes to bring it in line
        - Reduces snowflakes and manual effort in system management

- Selinux
    - Core concepts
        - What is a program supposed to be doing?
        - Kernel
        - Profile
        - Enforcing/permissive/disabled
        - Not enabled on Ubuntu
        - Enabled on Fedora/RHEL/Centos
    - Test if enabled (package "policycoreutils")

```
sestatus
```

    - Switching between enforcing and permissive

```
setenforce enforcing
setenforce permissive
```

    - Enable/disable
        - First edit /etc/selinux/config

```
sudo nano /etc/selinux/config
SELINUX=disabled
SELINUX=permissive
SELINUX=enforcing
```

        - Force relabel on next boot

```
sudo touch /.autorelabel
```

        - Reboot required to switch

```
sudo reboot
```

    -

References:

```
man strace
https://www.ansible.com/
man selinux
```

- References
    - https://www.activecountermeasures.com/why-is-my-program-running-slowly/
    - http://www.stearns.org/doc/network-monitoring.current.html
    - https://www.linux.org/forums/getting-started.148/
    - For commandX, try:

```
man commandX
commandX --help
commandX -h
info commandX
```

    - https://assets.ubuntu.com/v1/7a13f770-ubuntu-server-guide.pdf
    - https://www.amazon.com/Ubuntu-Linux-Unleashed-Helmke-Matthew-ebook-dp-B08F5K61KK/dp/B08F5K61KK/
        - I've not read this book, but have read other "____ Linux Unleashed" books, have been very impressed with them, and the table of contents for this one looks similarly complete.
    -