2020

Lecture - CSCI 275: Linux Systems Administration and Security

Moe Hassan
CUNY John Jay College

NYC Tech-in-Residence Corps

Ch01- Starting with Linux

Learning what Linux is

Learning where Linux came from

Choosing Linux distributions

Exploring professional opportunities with Linux

Becoming certified in Linux

Where is Linux found?

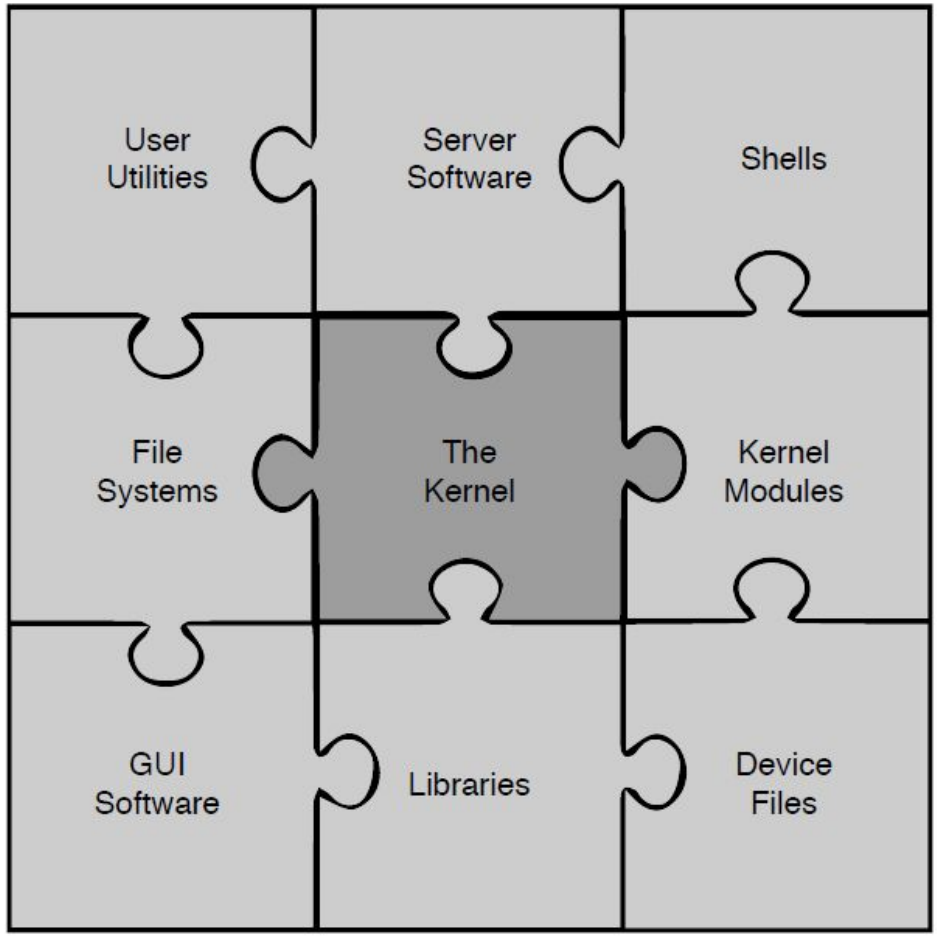
- Google runs thousands upon thousands of Linux servers to power its search technology
- Its Android phones are based on Linux.
- Facebook builds and deploys its site using what is referred to as a *LAMP stack* (Linux, Apache web server, MySQL database, and PHP web scripting language)—all open source projects.
- Financial organizations that have trillions of dollars riding on the speed and security of their operating systems also rely heavily on Linux
- Foundation of “cloud” IS Linux

Introducing Linux

- Linux is an operating system, much like Microsoft Windows
- Linux itself is a kernel, not a full OS
  - Kernel is open source
- Many components come together in a distribution, or distro, to form a complete OS
  - Some distros are free; others are commercial

- A kernel is a software responsible for:
  - Interfacing with hardware devices
  - Allocating memory to individual programs
  - Allocating CPU time to individual programs
  - Enabling programs to interact with each other
- Kernels are not interchangeable.
- Linux OS kernel:
  - Called Linux
  - Created by student Linus Torvalds in 1991
  - Runs on many various platforms & hardware

Components of the Linux OS



Features that make up Linux and other OS:

Detecting and preparing hardware—When the Linux system boots up (when you turn on your computer), it looks at the components on your computer (CPU, hard drive, network cards, and so on) and loads the software (drivers and modules) needed to access those particular hardware devices.

Managing processes—The operating system must keep track of multiple processes running at the same time and decide which have access to the CPU and when. The system also must offer ways of starting, stopping, and changing the status of processes.

Managing memory—RAM and swap space (extended memory) must be allocated to applications as they need memory. The operating system decides how requests for memory are handled.

Providing user interfaces—An operating system must provide ways of accessing the system. The first Linux systems were accessed from a command-line interpreter called a *shell*. Today, graphical desktop interfaces are commonly available as well.

Controlling filesystems—Filesystem structures are built into the operating system (or loaded as modules). The operating system controls ownership of and access to the files and directories (folders) that the filesystems contain.

Providing user access and authentication—Creating user accounts and allowing boundaries to be set between users is a basic feature of Linux. Separate user and group accounts enable users to control their own files and processes.

Offering administrative utilities—In Linux, hundreds (perhaps thousands) of commands and graphical windows are available to do such things as add users, manage disks, monitor the network, install software, and generally secure and
manage your computer.

Starting up services—To use printers, handle log messages, and provide a variety of system and network services, processes called *daemon processes* run in the background, waiting for requests to come in. Many types of services run in Linux.

Programming tools—A wide variety of programming utilities for creating applications and libraries for implementing specialty interfaces are available with Linux.

Advanced features of Linux

- **Clustering**—Linux can be configured to work in clusters so that multiple systems can appear as one system to the outside world. Services can be configured to pass back and forth between cluster nodes, while appearing to those using the services that they are running without interruption.
- **Virtualization**—To manage computing resources more efficiently, Linux can run as a virtualization host. On that host, you could run other Linux systems, Microsoft Windows, BSD, or other operating systems as virtual guests. To the outside world, each of those virtual guests appears as a separate computer. KVM and Xen are two technologies in Linux for creating virtual hosts.
- **Cloud computing**—To manage large-scale virtualization environments, you can use full-blown cloud computing platforms based on Linux. Projects such as OpenStack and Red Hat Enterprise Virtualization can simultaneously manage many virtualization hosts, virtual networks, user and system authentication, virtual guests, and networked storage.
- **Real-time computing**—Linux can be configured for real-time computing, where high-priority processes can expect fast, predictable attention.
- **Specialized storage**—Instead of just storing data on the computer's hard disk, many specialized local and networked storage interfaces are available in Linux. Shared storage devices available in Linux include iSCSI, Fibre Channel, and Infiniband. Entire open source storage platforms include projects such as Ceph and GlusterFS

Linux History

- Began in 1991 with a message from Linus Torvalds.
- Minix was a UNIX-like operating system that ran on PCs in the early 1990s
- Like Minix, Linux was also a clone of the UNIX operating system

- Bell Labs employees Ken Thompson and Dennis Ritchie set off on their own to create an operating system UNIX.
- UNIX offered – filesystem, I/O redirection, portability.

$ **cat file1 file2 | sort | pr | lpr**

- Portability required high level programming language, gave birth to “C” by Brian Kernighan and Dennis Richie.
- UNIX was commercialized, running on special hardware.

Linux History

- Richard Stallman starts GNU project in 1984.
- GNU GPL License allows – Author retains rights, free distribution, copyright is maintained.
- No warranty on GNU software.

- Linux kernel was the piece that was needed to complete a whole UNIX-like OS under GPL.
- Linux can be described as an open source UNIX-like operating system that reflects a combination of SVID, POSIX, and BSD compliance.
- Commercial vendors include - IBM, Red Hat, SUSE, Oracle, HP, Dell, Computer Associates, Intel, Cisco Systems, and others

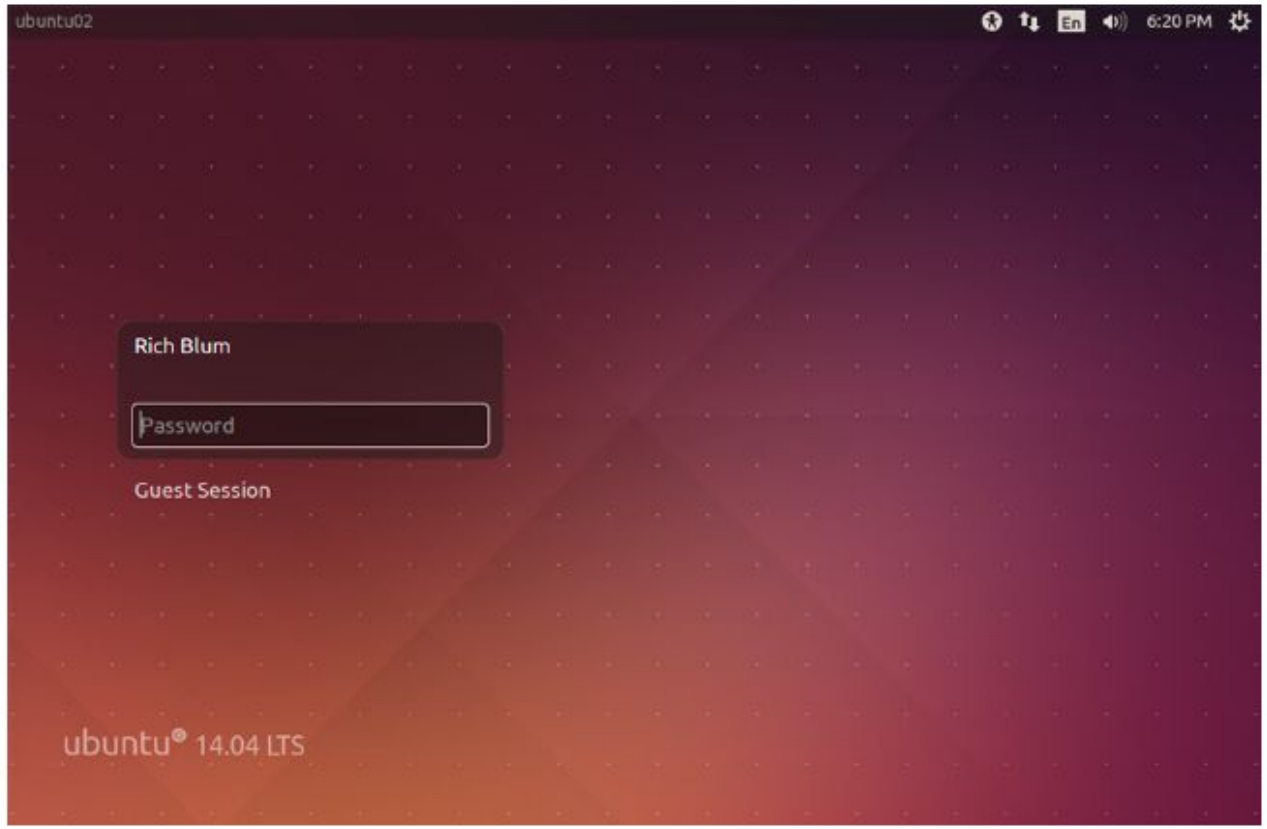
What is Open Source?

- Source code = Programming code human beings use to write software programs
- Compiler = Software that turns source code into binary code, but does not execute the binary code
- Translator = Software that turns source code temporarily into binary code and executes it
- Binary code = Code that machines need to run/understand software programs
- Open source = Freely available source code

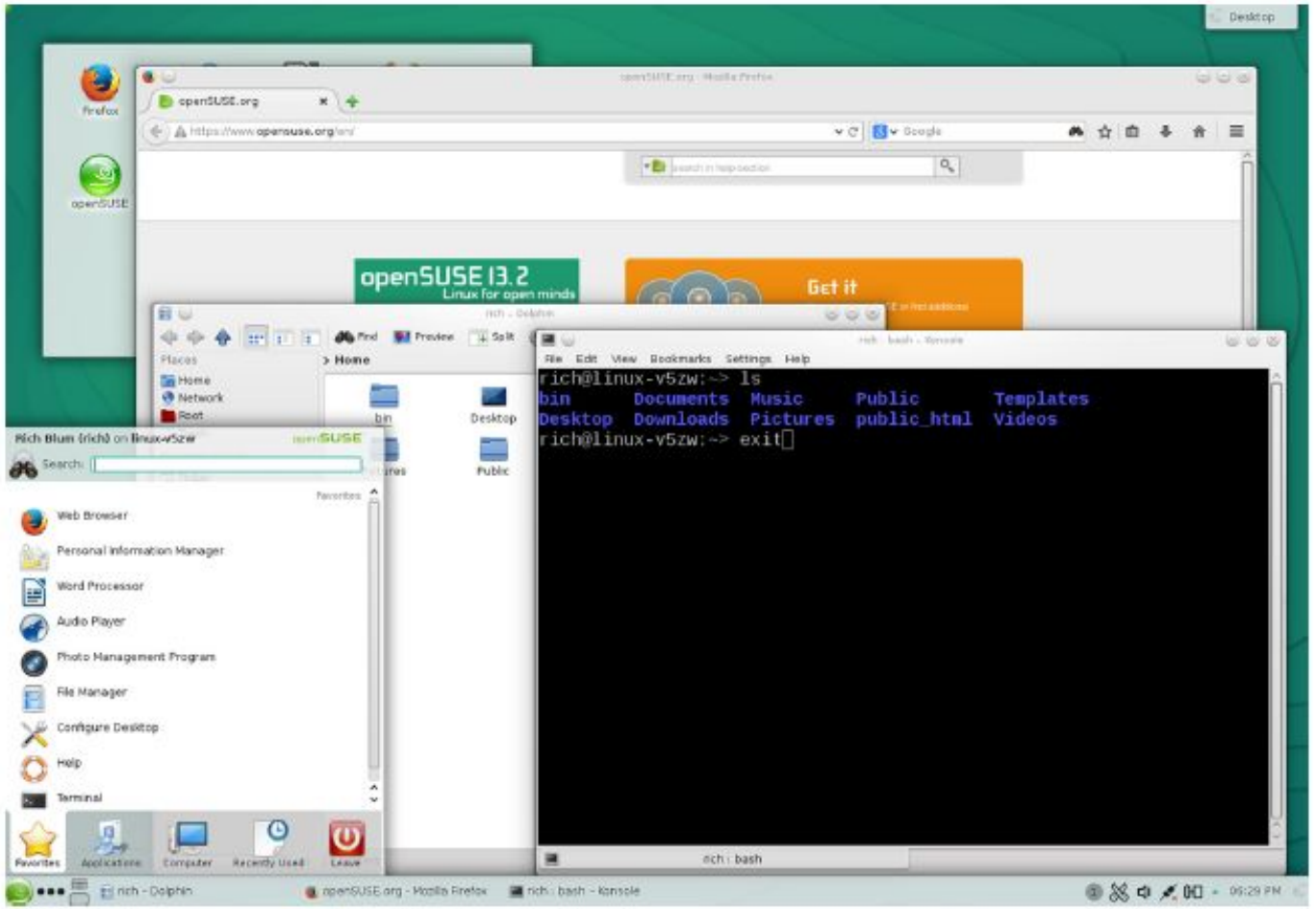
Linux Distributions

- Commercial
  - Red Hat Enterprise Linux
  - SUSE
- Home or amateur
  - Fedora
  - Linux Mint
  - Ubuntu
- Security enhanced
  - Kali Linux
  - Alpine Linux
- Live
  - Manjaro Linux
  - Antegros
  - Live versions available of Fedora, Linux Mint

ubuntu


ubuntu02
En
6:20 PM
Rich Blum
Password
Guest Session
ubuntu® 14.04 LTS

Open SUSE


openSUSE 13.2
Linux for open minds
Get it
rich@linux-v5zw:~> ls
bin Documents Music Public Templates
Desktop Downloads Pictures public_html Videos
rich@linux-v5zw:~> exit
Web Browser
Personal Information Manager
Word Processor
Audio Player
Photo Management Program
File Manager
Configure Desktop
Help
Terminal

Shells

- *Shell*: A software program that allows a user to issue commands to the system
  - Command line interface (CLI)
  - Graphical User Interface (GUI)
- Different shells are available, with different features, functions, and syntax
- BASH shell is most popular CLI, and is assumed in this book

Which Distro to Choose

- Red Hat Enterprise Linux (RHEL), Fedora, or CentOS
  - Collectively referred to as Red Hat-based distros
  - Fedora and CentOS are free
  - RHEL is subscription-based
- Linux Mint, Ubuntu, or Debian
  - Collectively referred to as Debian-based distros
- Kali
  - Security-based, contains many security-related tools

Choosing a Red Hat distribution

RPM package management—Tarballs are fine for dropping software on your computer, but they don't work as well when you want to update, remove, or even find out about that software. Red Hat created the RPM packaging format so a software package could contain not only the files to be shared, but also information about the package version, who created it, which files were documentation or configuration files, and when it was created. By installing software packaged in RPM format, that information about each software package could be stored in a local RPM database. It became easy to find what was installed, update it, or remove it.

Simple installation—The anaconda installer made it much simpler to install Linux. As a user, you could step through some simple questions, in most cases accepting defaults, to install Red Hat Linux.

Graphical administration—Red Hat added simple graphical tools to configure printers, add users, set time and date, and do other basic administrative tasks. As a result, desktop users could use a Linux system without even having to run commands.

Choosing Ubuntu or another Debian distribution

Uses deb packaging format and tools to manage software.

Debian also has a reputation for stability.

More than 130 active distributions use debian.

Debian derivative that has achieved the most success is Ubuntu.

Ubuntu added features that Debian lacked, hugely popular.

Ubuntu offered many options to run, new user friendly, emulators.

Professional Opportunities with Linux Today

■ **Linux talent is a high priority**—Hiring people with Linux expertise is a priority for 77 percent of hiring managers.

■ **Career advancement with Linux**—As for career opportunities, 86 percent of Linux professionals reported that Linux knowledge increased career opportunities.

■ **More Linux recruiting**—Of the hiring managers surveyed, 46 percent reported that they planned to increase recruitment of Linux talent from the previous year (up 3 percent from the previous year).

How companies make money with Linux

- **Software subscriptions**
- **Training and certification**
- **Bounties**
- **Donations**
- **Boxed sets, mugs, and T-shirts**

Linux Certification Options

Comptia Linux+	RHCSA	RHCE
Hardware and System Configuration	Understand essential tools	Bonding
Systems Operation and Maintenance	Operate running systems	Route IP traffic
Security	Configure local storage	Firewalls
Linux Troubleshooting and Diagnostics	Create and configure filesystems	Kernel tunables
Automation and Scripting	Deploy, configure, and maintain systems	Kerberos authentication
	Manage users and groups	Configure iSCSI
	Manage security	System reports
		Shell scripting
		Remote logging
		SELinux

A Summary of Common Linux Distributions

Distribution	Availability	Package format	Release cycle	Administrator skill reqs
Arch	Free	pacman	Rolling	Expert
CentOS	Free	RPM	approx 2-year	Intermediate
Debian	Free	Debian	2-year	Intermediate to expert
Fedora	Free	RPM	approx 6-month	Intermediate
Gentoo	Free	Ebuild	Rolling	Expert
Mint	Free	Debian	6-month	Novice to intermediate
openSUSE	Free	RPM	8-month	Intermediate
Red Hat Enterprise	Commercial	RPM	approx 2-year	Intermediate
Scientific	Free	RPM	approx 6-month	Intermediate to expert
Slackware	Free	tarballs	Irregular	Expert
SUSE Enterprise	Commercial	RPM	2–3 years	Intermediate
Ubuntu	Free	Debian	6-month	Novice to intermediate

Ch02- Creating the Perfect Linux Desktop

Understanding the X Window System and desktop environments

Running Linux from a Live CD/DVD

Navigating the GNOME 3 desktop

Adding extensions to GNOME 3

Using Nautilus to manage files in GNOME 3

Working with the GNOME 2 desktop

Enabling 3D effects in GNOME 2

Linux is all about choices!

- Choice of distribution – RPM based or DEB based. Specialized ones are also available – Gentoo, Slackware, ArchLinux

- Choice of desktop environment – Full featured such as GNOME, KDE and lightweight one such as LXDE, Xfce

- Choice of installation – run as “live” without installing on hard disk, or install permanently on hard disk.

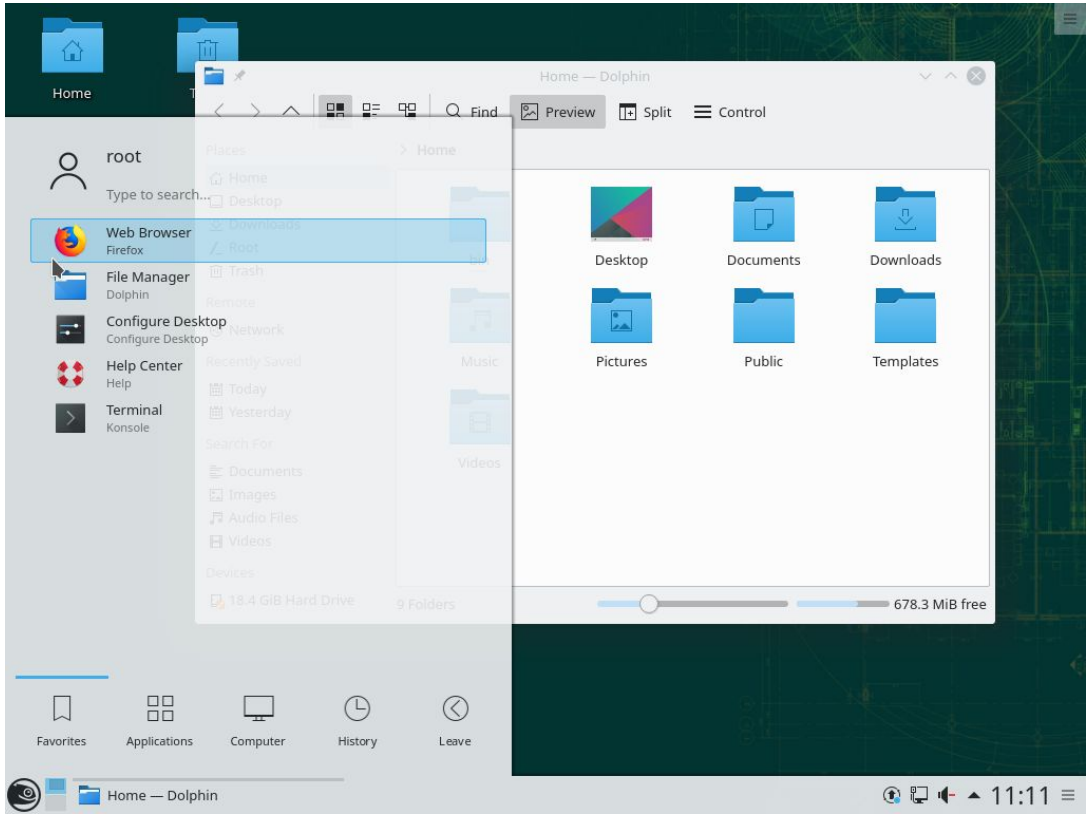
Linux Desktop Technology (X Window)

- Linux desktop environment is based on X Window System. It was built to be lightweight, networked desktop framework.
- Works like backward client/server model.
- Created when terminals were used to manage larger centralized computers. Applications ran on larger computer but displayed on terminal screens, delivered over network.
- Linux does not require X Window -> Only plain-text, command-line interface.
- Linux can install multiple desktop environments. Change at login.

GNOME 3

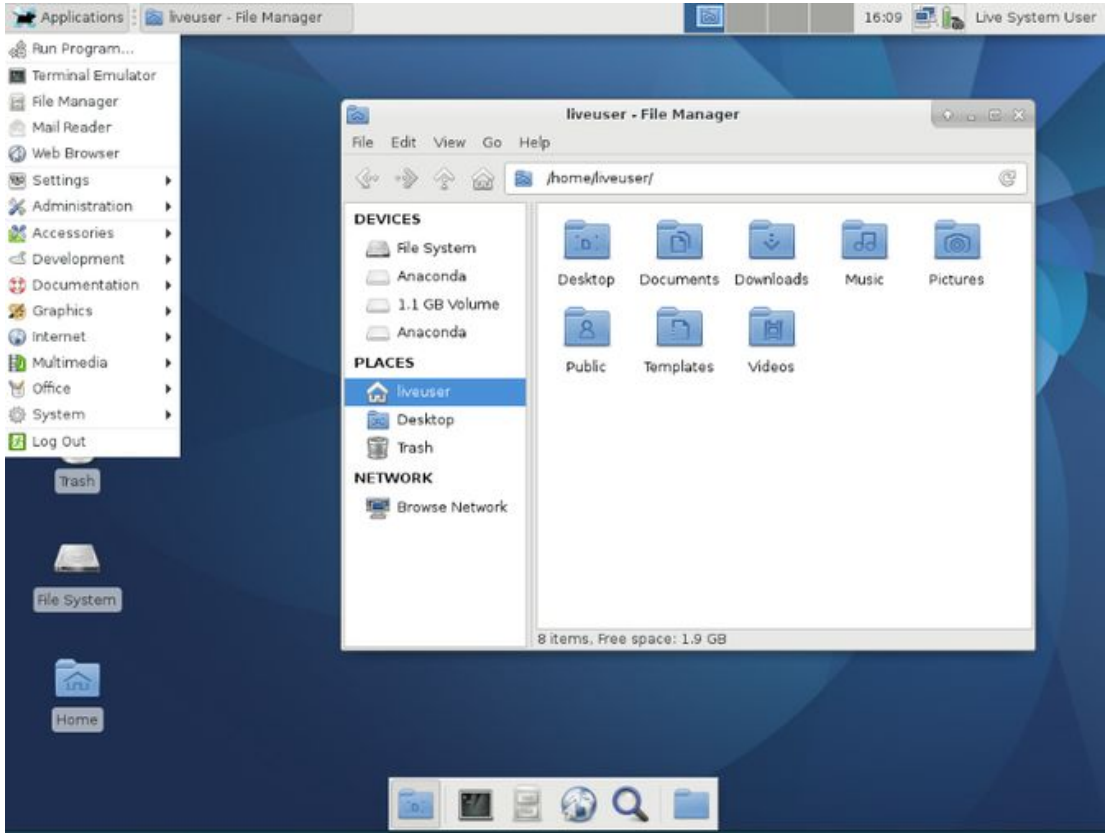
GNOME is the default desktop environment for Fedora, Red Hat Enterprise Linux, and many others. Think of it as a professional desktop environment, focusing on stability more than fancy effects.

KDE – K Desktop Environment



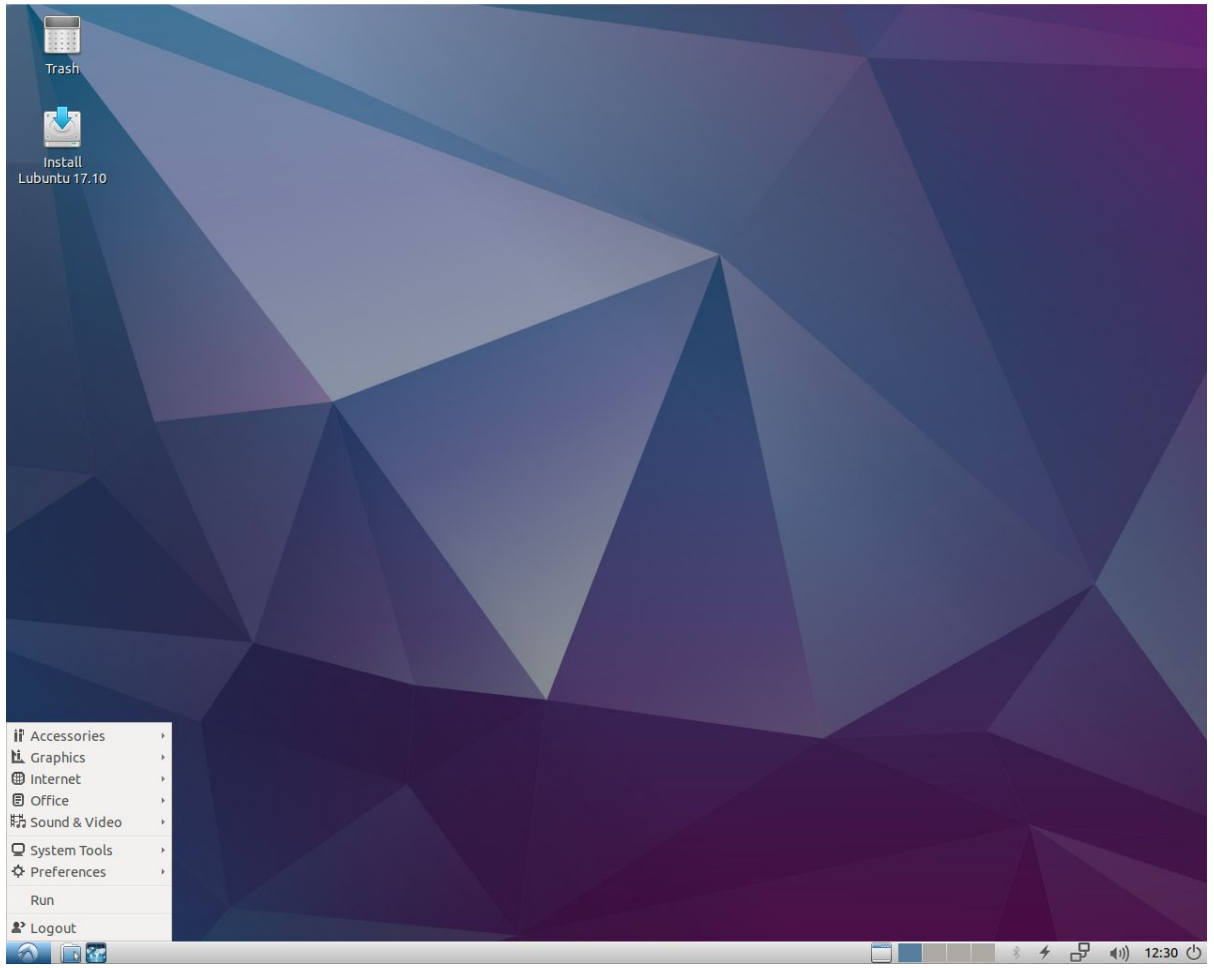
KDE is probably the second most popular desktop environment for Linux. It has more bells and whistles than GNOME and offers more integrated applications. KDE is also available with Fedora, RHEL, Ubuntu, and many other Linux systems.

Xfce



The Xfce desktop was one of the first lightweight desktop environments. It is good to use on older or less powerful computers. It is available with RHEL, Fedora, Ubuntu, and other Linux distributions.

LXDE



The Lightweight X11 Desktop Environment (LXDE) was designed to be a fast-performing, energy-saving desktop environment. Often, LXDE is used on less-expensive devices (such as netbook computers) and on live media (such as a live CD or live USB stick).

Installing CentOS Linux as a virtual machine

- 1) Download latest version of Virtualbox from here - https://www.virtualbox.org/wiki/Downloads

- 2) Download this file named "CentOS-7-x86_64-LiveGNOME-1908.iso" from here - http://mirror.es.its.nyu.edu/centos/7.7.1908/isos/x86_64/ (or from http://mirrors.rit.edu/centos/7.7.1908/isos/x86_64/

- 3) Proceed to installing Virtualbox software on your system. Once done, have a look at this tutorial - https://www.virtualbox.org/manual/ch01.html

- 4) After installing Virtualbox, you can install CentOS. See this tutorial - https://linoxide.com/how-tos/centos-7-step-by-step-screenshots/

Summary

The GNOME desktop environment has become the default desktop environment for many Linux systems, including Fedora and RHEL. The GNOME 3 desktop (now used in Fedora and Red Hat Enterprise Linux 7) is a modern, elegant desktop, designed to match the types of interfaces available on many of today's mobile devices. The GNOME 2 desktop (used through RHEL 6) provides a more traditional desktop experience.

Besides GNOME desktops, you can try out other popular and useful desktop environments. The K Desktop Environment (KDE) offers many more bells and whistles than GNOME and is used by default in several Linux distributions. Netbooks and live CD distributions sometimes use the LXDE or Xfce desktops.

Ch03- Using the Shell

Understanding the Linux shell
Using the shell from consoles or terminals
Using commands
Using command history and tab completion
Connecting and expanding commands
Understanding variables and aliases
Making shell settings permanent
Using man pages and other documentation

Bourne **A**gain **Sh**ell (sh)

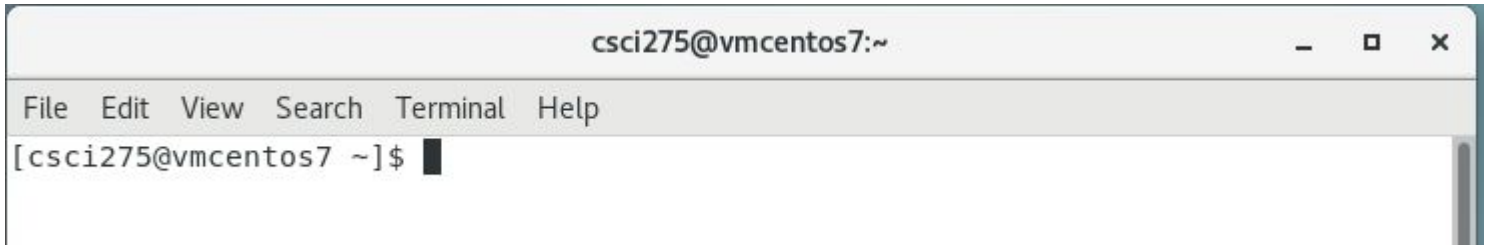
Shell is always available for both RPM and Debian based Linux

Provide means to – run programs, file management, compile computer code, manage computer in general.

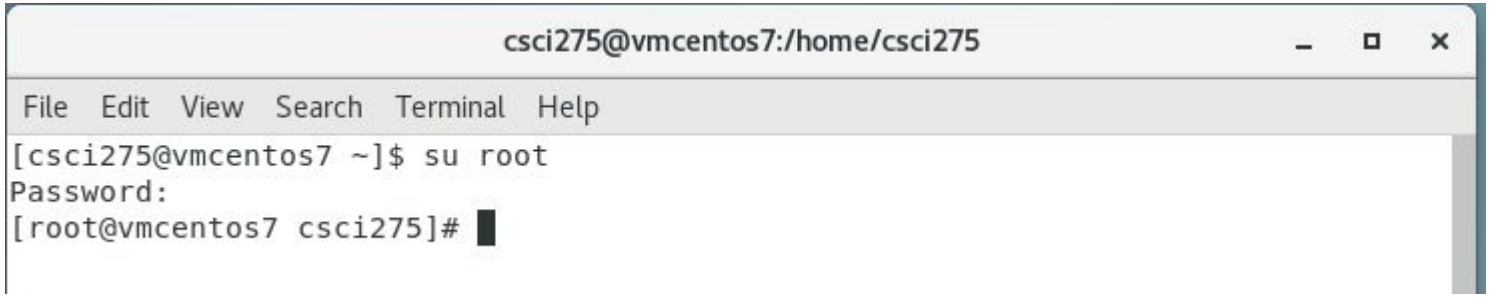
Other shell examples – C shell (csh), Korn shell (ksh), tcsh, ash

Shell is a command language interpreter.

[username@system ~]$ | ~ indicates current directory name



User: csci275
Computer name: vmcentos7
Privilege: normal user ($)



User: root
Computer name: vmcentos7
Privilege: root user (#)

Ch04 - File system management

Learning about the Linux filesystem
Listing fi le and directory attributes
Making fi les and directories
Listing and changing permission and ownership
Making copies and moving files

Linux Filesystems versus Windows-Based Filesystems

- Linux is very case sensitive whereas Windows is not. For example – **Cat.txt** is different from **cAT.txt** or **cAt.txt** in Linux.
- Windows uses letters (such as C:\, E:\ etc). In Linux, all storage devices are connected to the filesystem hierarchy.
- Windows uses backslash (\) and Linux uses forward slash (/)
- Windows uses file extensions (such as .docx or .txt). In linux, it is not essential.
- Every file and directory in a Linux system has permissions and ownership associated with it. Security varies among Microsoft systems.

Using file-redirection metacharacters

■ <—Directs the contents of a file to the command. In most cases, this is the default action expected by the command and the use of the character is optional; using less bigfile is the same as less < bigfile.

■ >—Directs the standard output of a command to a file. If the file exists, the content of that file is overwritten.

■ 2>—Directs standard error (error messages) to the file.

■ &>—Directs both standard output and standard error to the file.

■ >>—Directs the output of a command to a file, adding the output to the end of the existing file.

Using brace expansion characters

By using curly braces ({}), you can expand out a set of characters across filenames, directory names, or other arguments you give commands. Examples:

```
$ touch memo{1,2,3,4,5}
$ ls
memo1 memo2 memo3 memo4 memo5

$ touch {John,Bill,Sally}-{Breakfast,Lunch,Dinner}
$ ls
Bill-Breakfast Bill-Lunch John-Dinner Sally-Breakfast
Sally-Lunch
Bill-Dinner John-Breakfast John-Lunch Sally-Dinner

$ touch {a..f}{1..5}
$ ls
a1 a3 a5 b2 b4 c1 c3 c5 d2 d4 e1 e3 e5 f2 f4
a2 a4 b1 b3 b5 c2 c4 d1 d3 d5 e2 e4 f1 f3 f5
```

Listing Files and Directories

- Most used command is “ls” which can be aliased. Type “`alias ls`” to view them.

```
$ ls -la /home/joe
total 158
```

drwxrwxrwx	2	joe	sales	4096	May 12 13:55	.
drwxr-xr-x	3	root	root	4096	May 10 1:49	..
-rw-------	1	joe	sales	2204	May 18 21:30	.bash_history
-rw-r--r--	1	joe	sales	24	May 10 1:50	.bash_logout
-rw-r--r--	1	joe	sales	230	May 10 1:50	.bash_profile
-rw-r--r--	1	joe	sales	124	May 10 1:50	.bashrc
drw-r--r--	1	joe	sales	4096	May 10 1:50	.kde
-rw-rw-r--	1	joe	sales	149872	May 11 22:49	letter

1st column: **-** for File, **d** for Directory, **l** for Link
2nd column: This field specifies the number of links or directories inside this directory.
3rd column: The user that owns the file, or directory
4th column: The group that file belongs to, and any user in that group will have the permissions given in the third field over that file.
5th column: The size in bytes
6th column: The date of last modification
7th column: The name of the file or directory

File Permissions and Ownership

Permission is shown for: user (u), group (g), other (o), and all users (a).

READ (**r=4**): View what's in the file. See what files and subdirectories it contains.

WRITE (**w=2**): Change the file's content, rename it, or delete it. Add files or subdirectories to the directory. Remove files or directories from the directory.

EXECUTE (**x=1**): Run the file as a program. Change to the directory as the current directory, search through the directory, or execute a program from the directory. Access file metadata (file size, time stamps, and so on) of files in that directory.

File Permissions and Ownership

Each permission (read, write, and execute) can be set using r=4, w=2, and x=1.

To make permissions wide open for yourself as owner, you would set the first number to 7 (4+2+1), and then you would give the group and others read-only permission by setting both the second and third numbers to 4 (4+0+0), so that the final number is 744. Any combination of permissions can result from 0 (no permission) through 7 (full permission).

The following `chmod` command results in this permission: `rwxrwxrwx`

```
# chmod 777 filename
```

The following `chmod` command results in this permission: `rwxr-xr-x`

```
# chmod 755 filename
```

The following `chmod` command results in this permission: `rw-r--r--`

```
# chmod 644 filename
```

The following `chmod` command results in this permission: `----------`

```
# chmod 000 filename
```

Changing permissions with chmod (letters)

File permission can be changed using + and – signs along with letters to indicate what changes [read (r), write (w), and execute (x)] and for whom [user (u), group (g), other (o), and all users (a)]

For example, start with a file that has all permissions open (`rwxrwxrwx`).

The following chmod command results in this permission: `r-xr-xr-x`
`$ chmod a-w file`
The following chmod command results in this permission: `rwxrwxrw-`
`$ chmod o-x file`
The following chmod command results in this permission: `rwx------`
`$ chmod go-rwx file`
The following chmod command results in this permission: `rw-------`
`$ chmod u+rw files`
The following chmod command results in this permission: `--x--x--x`
`$ chmod a+x files`
The following chmod command results in this permission: `r-xr-x---`
`$ chmod ug+rx files`

Moving, Copying, and Removing Files

- To change the location of a file, use the mv command. By default, the mv command overwrites any existing files if the file you are moving to exists.
- To copy a file from one location to another, use the cp command.
- To remove a file, use the rm command.

CAUTION:
When you override the `-i` option on the `mv`, `cp`, and `rm` commands, you risk removing some (or lots of) files by mistake. Using wildcards (such as `*`) and no `-i` makes mistakes even more likely. That said, sometimes you don't want to be bothered to step through each fi le you delete. You have other options:

■ As noted with the `-f` option, you can force `rm` to delete without prompting. An alternative is to run `rm`, `cp`, or `mv` with a backslash in front of it (`\rm bigdir`). The backslash causes any command to run unaliased.

■ Another alternative with `mv` is to use the `-b` option. With `-b`, if a file of the same name exists at the destination, a backup copy of the old file is made before the new file is moved there.

Ch06 - Managing Running Processes

Understanding Processes

Listing Processes

Managing Background and Foreground Processes

Killing and Renicing Processes

Limiting Processes with `cgroups`

Linux Processes

- *Multitasking* running program is referred to as a means that many programs can be running at the same time.
- An instance of a running program is referred to as a *process.*
- Every process has process ID, and parent ID (exception is `init`, PID=1)
- From a shell, you can launch, pause, stop, or kill processes
- Common tools to manage process include - `ps,fg,bg,top,kill,jobs`

Understanding Processes

- There is one `vi` command, but can be used by multiple users, therefore will have different/unique PID
- Processes running as root user will have more privileges.

Commands that display information about running processes get most of that information from raw data stored in the `/proc` file system. Each process stores its information in a subdirectory of `/proc`, named after the process ID of that process. You can view some of that raw data by displaying the contents of fi les in one of those directories (using `cat` or `less` commands).

Listing processes with `ps`

```
$ ps u
USER PID %CPU %MEM VSZ RSS TTY STAT START TIME COMMAND
jake 2147 0.0 0.7 1836 1020 tty1 S+ 14:50 0:00 -bash
jake 2310 0.0 0.7 2592 912 tty1 R+ 18:22 0:00 ps u
```

- The most common utility for checking running processes is the `ps` command.
- `ps` shows which programs are running, the resources they are using, and who is running them, `u` shows username
- The first process (2147) shows that the user named jake opened a `bash` shell after logging in
- The next process (2310) shows that jake has run the `ps u` command
- `USER` column shows the name of the user who started the process
- `PID` column shows the unique ID number referred to as a process ID, can be used to terminate (`kill`) it
- `%CPU` and `%MEM` columns show the percentages of the processor and RAM usage
- `VSZ` (virtual set size) shows the size of the image process (in kilobytes)
- `RSS` (resident set size) shows the size of the program in memory
- `STAT` column indicates sleeping (`S`) or running (`R`) processes
- `START` shows the time the process began running
- `TIME` shows the cumulative system time used.

Managing Background and Foreground Processes

- Linux allows moving move active programs between background and foreground by using ampersand (`&`) to the end of a command line.

```
$ find /usr > /tmp/allusrfiles &
[3] 15971
```

```
$ jobs
[1] Stopped (tty output) vi /tmp/myfile
[2] Running find /usr -print > /tmp/allusrfiles &
[3] Running nroff -man /usr/man2/* >/tmp/man2 &
[4]- Running nroff -man /usr/man3/* >/tmp/man3 &
[5]+ Stopped nroff -man /usr/man4/* >/tmp/man4
```

- You can also use the `at` command to run commands in such a way that they are not connected to the shell.
- To stop a running command and put it in the background, press Ctrl+Z. To bring it back into the foreground to run (the `fg` command) or start it running in the background (the `bg` command).

```
$ fg %1
```

The above command will reopen “myfile” in *vi* editor.

```
$ bg %5
```

The above will start job 5 from above and status changes to:

```
[5] Running nroff -man man4/* >/tmp/man4 &
```

Killing and Renicing Processes

- The `kill` or `killall` command can send a kill signal to any process to end it, reread configuration files, pause (stop), or continue after being paused.
- Different processes respond to different signals. Processes cannot block SIGKILL and SIGSTOP signals, however.
- Signals that you might send most commonly from a command include SIGKILL (9), SIGTERM (15), and SIGHUP (1).

```
$top
PID USER PR NI VIRT RES SHR S %CPU %MEM TIME+ COMMAND
10432 chris 20 0 471m 121m 18m S 99.9 3.2 77:01.76 bigcommand
```

To kill the “bigcommand” process which is consuming 99.9% of CPU, issue the following:

```
$ kill 10432
```

The *killall* can kill multiple processes by “name” instead of PID.

```
$ killall -9 testme
```

Killing and Renicing Processes

- The `nice` and `renice` commands can be used to set or change the processor priority of a process.
- Every process running on your system has a nice value between –20 and 19, default value is set to 0. The lower the nice value, the more access to the CPUs the process has.
- A regular user ($) can set the nice value (0 to 19) only on the user's own processes and when changing, it must be higher than current value. The root user (#) however can set the nice value on any process to any valid value, up or down.

In this example, first increasing `nice` value by 5 for updatedb process, verify using `top` command and then `renice` it:

```
# nice +5 updatedb
# top
PID     USER PR NI VIRT RES SHR S %CPU   %MEM     TIME+   COMMAND
20284   root 25 5 98.7m 932 644 D 2.7    0.0      0:00.96 updatedb
# renice -n -5 20284
```

Limiting Processes with `cgroups`

Cgroups can be used to identify a process as a *task*, belonging to a particular *control group*. Tasks can be set up in a hierarchy where, for example, there may be a task called daemons that sets default limitations for all daemon server processes, then subtasks that may set specific limits on a web server daemon (httpd) or FTP service daemon (vsftpd).

As a task launches a process, other processes the initial process launches (called *child processes*) inherit the limitations set for the parent process. Those limitations might say that all the processes in a control group have access only to particular processors and certain sets of RAM. Or they may allow access only to up to 30 percent of the total processing power of a machine.

Ch08 – Learning System Administration

Doing graphical administration

Using the root login

Understanding administrative commands, config files, and log files

Working with devices and file systems

Understanding System Administration

- Linux is multi-user environment. Administrators are tasked with:
  - ❑ File systems management
  - ❑ Software installation
  - ❑ User accounts
  - ❑ Network interfaces
  - ❑ Servers
  - ❑ Security Features
  - ❑ Hardware management (`lsmod, lspci, lscpu, lsusb, modprobe`)

  and more +
- Administer systems using either `su` or `sudo` command and also by using graphical administration tools.

HTTP status code classes

HTTP status codes are generally divided into five different classes. The first digit of the three-digit code shows which class it belongs to. The different classes are:

- **Class 1xx – Informational:** If an HTTP status code 1xx is transmitted, the server informs the client that the request is in motion. This class combines codes that are responsible for delivering information to the client during the request.
- **Class 2xx – Success:** A 2xx code announces a successful operation. If this code is transmitted, it means that the client's request was received by the server, understood, and accepted. 2xx codes are often sent at the same time as the desired website information, and the user often only takes notice of the website they have requested.
- **Class 3xx – Redirection:** A 3xx code shows that the server's request was received. In order to ensure the request is successfully processed, further steps are needed from the client'send. 3xx codes appear during redirections and forwardings.
- **Class 4xx – Client error:** If a 4xx code appears then there's been a client error. The server has received the request, but cannot perform it. The reason behind this is usually an incorrect request. Internet users will be made aware of this error by receiving an automatically generated HTML page.
- **Class 5xx – Server error:** A 5xx code is shown when the server has failed to perform the request. These server error codes report that the request cannot be performed at present or is not possible at all, which then leads to an HTML error page.

source : https://www.ionos.com/digitalguide/hosting/technical-matters/the-most-important-http-status-codes-at-a-glance/

Using browser-based admin tools

- Red Hat Enterprise Linux OpenStack Platform (RHELOSP)
- Red Hat Enterprise Virtualization (RHEV)
- Webmin
- CentOS Web Panel
- Ajenti
- Froxlor
- VestaCP
- Cockpit Project
- Sentora

Using the root user account

- Switches to another user account
  - Example: `su - student`
- Opens a new shell in which the identity has been switched
- The - option enables you to switch as if you were logging in directly, so that the user's initialization files are executed
- To use su you must be the root user or you must have the password for the account being switched to
- Use the exit command to close the shell

Gaining administrative access with sudo

Assign root privilege for any command they run with `sudo`.
■ Assign root privilege for a select set of commands.
■ Give users root privilege without telling them the root password because they only have to provide their own user password to gain root privilege.
■ Allow users, if you choose, to run sudo without entering a password at all.
■ Track which users have run administrative commands on your system. (Using su, all you know is that someone with the root password logged in, whereas the sudo command logs which user runs an administrative command.)

Exploring Administrative Commands, Configuration Files, and Log Files

- Administrative commands
  - `/sbin`—Contained commands needed to boot your system, including commands for checking filesystems (fsck) and turn on swap devices (swapon).
  - `/usr/sbin`—Contained commands for such things as managing user accounts (such as `useradd`) and checking processes that are holding files open (such as `lsof`). Commands that run as daemon processes are also contained in this directory. Daemon processes are processes that run in the background, waiting for service requests such as those to access a printer or a web page. (Look for commands that end in d, such as `sshd`, `pppd`, and `cupsd`.)

Exploring Administrative Commands, Configuration Files, and Log Files

- Administrative configuration files
  Every configuration file is stored as plaintext file, easy to read and change but prone to errors.
- Here are some key directories:
  - $HOME
  - `/etc` (under this, `/cron*, /cups, /httpd, /init.d, /mail, /pcmcia, /postfix, /ppp, /rc?.d, /security, /skel, /sysconfig, /systemd, /xinetd.d`)
- These directories contain various configuration files with parameters defined.

Ch10 – Getting and Managing Software

Installing software from the desktop

Working with RPM packaging

Using yum to manage packages

Using rpm to work with packages

Installing software in the enterprise

CentOS : Graphical Application Installer

Why use `rpm`/`yum` commands?

- More Repositories – more sources to obtain software from.
- Beyond Desktop Applications – Install more applications beyond default installation
- Flexibility – yum and rpm commands provide more controls
- More complex queries – Above commands provide more information on packages, groups
- Software validation – Both commands allows integrity checks
- Managing software installation – commands provide more scalability, automation

Understanding Linux RPM and DEB Software Packaging

- Early days – grab source code > compile it to runnable binaries > drop it onto your computer. Resulting *tarball* file contains executable, documentation, configuration files, and libraries.

- This method of installing software makes it difficult to do these things – Get dependent software, List the software, Remove the software, Update the software.

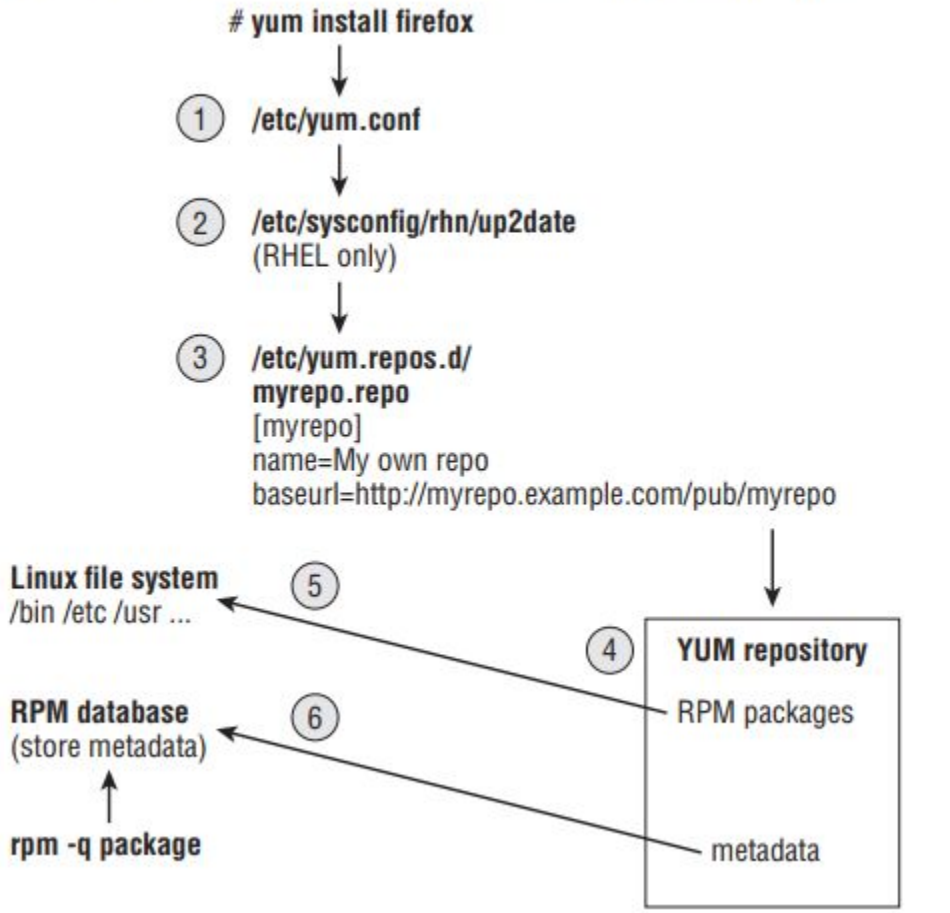
- To deal with these problems, packages progressed from simple *tarballs* to more complex packaging. With only a few notable exceptions (such as Gentoo, Slackware, and a few others), the majority of Linux distributions went to one of two packaging formats—DEB and RPM

Managing RPM Packages with YUM

The Yellowdog Updater Modified (YUM) project solves the dependencies headache. Distribution or 3rd party repositories should provide necessary libraries and components that might be needed for specific package.

Repositories can be http, ftp or local cd/dvd media. The locations of these repositories would then be stored on the user's system in the `/etc/yum.conf` file or, more typically, in separate configuration files in the `/etc/yum.repos.d` directory.

Local and remote activities when installing an RPM with YUM.



Managing Software in the Enterprise

■ **Kickstart files**— Installation parameters can be saved and reused for automation.

■ **PXE boot** – Using network boot and kickstart files, systems can booted using its NIC card, pull down necessary files/configuration from a central server.

■ **Satellite server (Spacewalk**)— Redhat proprietary solution for managing RH systems – updates, configuration are managed.

Ch12 – Managing Disks and Filesystems

Creating disk partitions

Creating logical volumes with LVM

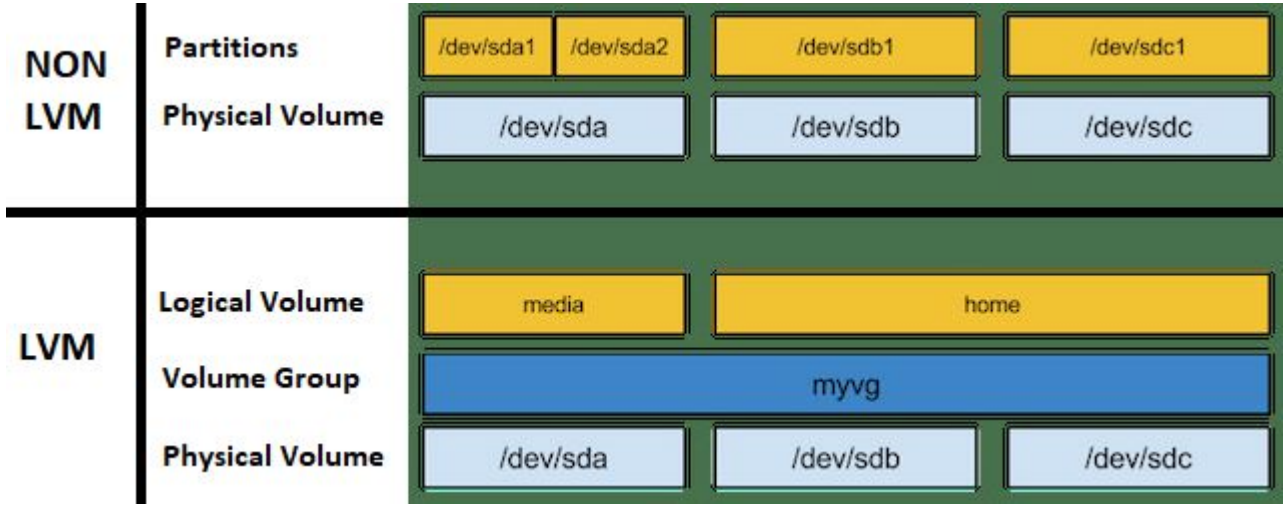
Adding filesystems

Mounting filesystems

Unmounting filesystems

Introduction:

- Disks are permanent storage device, usually divided into partitions with a structure also known as filesystem. Tasks include partitioning, adding filesystems, and managing those filesystems in various ways.
- Logical Volume Manager allows for a layer of abstraction between the OS and the hardware. Dynamic partitions, allows creating/resizing/deleting while OS is running



Understanding Disk Storage

- Disks are divided into partition(s), then formatted with a filesystem (swap area, or LVM physical volumes).
- Disks are used for permanent storage; random access memory (RAM) and swap are used for temporary storage.
- Your CPU can access data much faster from RAM than it can from hard disk.
- A swap space is a hard disk swap partition or a swap file where your computer can “swap out” data from RAM that isn’t being used at the moment and then “swap in” the data back to RAM when it is again needed.

Understanding Disk Storage

- Another special partition is a logical volume management (LVM) physical volume. LVM physical volumes enable you to create pools of storage space called *volume groups*.

- For Linux, at least one disk partition is required, assigned to the root (/) of the entire Linux filesystem. Additional include `/home`, `/var` and `/tmp`

- The business of connecting disk partitions to the Linux filesystem is done automatically and invisibly to the end user. An entry in the `/etc/fstab` file tells Linux each partition's device name and where to mount it

Using Logical Volume Management Partitions

- With LVM, physical disk partitions are added to pools of space called *volume groups*.

■ Add more space to a logical volume from the volume group while the volume is still in use.
■ Add more physical volumes to a volume group if the volume group begins to run out of space. The physical volumes can be from disks.
■ Move data from one physical volume to another, so you can remove smaller disks and replace them with larger ones while the filesystems are still in use—again, without downtime.

Mounting Filesystems

- Most of the hard disk partitions created when you install Linux are mounted automatically (`/etc/fstab`) for you when the system boots.
- The `mount` command is used not only to mount local storage devices, but also to mount other kinds of filesystems (NFS, Samba, new disk, USB) on your Linux system.
- To see filesystem types that are loaded in your kernel, type `cat /proc/filesystems`

Using the mount command

Any user can type `mount` (with no options) to see what filesystems are currently mounted on the local Linux system.

If device was not auto-mounted, it can be mounted manually:

```
# mkdir /mnt/myextdrive
# mount -t ext3 -o ro /dev/sdb1 /mnt/myextdrive
```

Another reason to use the mount command is to remount a partition to change its mount options.

```
# mount -t ext3 -o remount,rw /dev/sdb1
```

Also used for mounting disk images:

```
# mkdir /mnt/mycdimage
# mount -o loop whatever-i686-disc1.iso /mnt/mycdimage
```

Using the umount command

- Umount command (either a directory name or a device name) detaches the filesystem from its mount point in Linux.

```
# umount /mnt/test
# umount /dev/sdb1
```

- In general, it's better to use the directory name (`/mnt/test`) because the umount command will fail if the device is mounted in more than one location.

- An alternative for unmounting a busy device is the `-l` option.

Summary

Managing filesystems is a critical part of administering a Linux system. Using commands such as `fdisk`, you can view and change disk partitions. Filesystems can be added to partitions using the `mkfs` command. Once created, filesystems can be mounted and unmounted using the `mount` and `umount` commands, respectively.

Logical Volume Management (LVM) offers a more powerful and flexible way of managing disk partitions. With LVM, you create pools of storage, called volumes, that can allow you to grow and shrink logical volumes, as well as extend the size of your volume groups by adding more physical volumes.

Ch13 – Understanding Server Administration

Administering Linux servers

Communicating with servers over networks

Setting up logging locally and remotely

Monitoring server systems

Managing servers in the enterprise

Server Administration Skill

Remote access	Diligent Security	Continuous monitoring
GUI typically not available on servers Need to connect remotely (login, copying, execution etc) Use Terminal to perform tasks (software, user, backup, configuration management) Common tools are built on the Secure Shell (SSH) facility.	Server must be able to accept requests for content from remote users and systems. Important to open ports to services that are needed and lock down ports that are not needed. You can secure services using tools such as `iptables` and `firewalld` (firewall tools), TCP wrappers (to allow and deny service access), and Security Enhanced Linux (to limit the resources a service can access from the local system).	Servers usually stay on 24x7, 365 days per year. You can configure tools to monitor each server, gather log messages, and even forward suspicious messages to an e-mail account of your choice. You can enable system activity reporters to gather data around the clock on CPU usage, memory usage, network activity, and disk access.

Configure the server (use `vim` to edit)

- Most Linux servers are configured using plain text files in the `/etc` directory, though they do not offer immediate error checking.
- The main configuration file in Fedora and RHEL is `/etc/httpd/conf/httpd.conf`. The configuration directory is `/etc/httpd/conf.d/`.
- Most server software packages are installed with minimal configuration and lean more toward being secure than totally useful out of the box. Example - mail servers (`sendmail` or `postfix` packages) and DNS servers (bind package). Both installed with default options and starts up on reboot, but only listens to `localhost`.

Start the server (checklist)

User and group permission

- Daemons runs as usrs and groups instead of root.
- For example, `httpd` runs as apache and `ntpd` runs as the ntp user.

Daemon configuration files

- Often, a service has a configuration file for the daemon stored in the `/etc/sysconfig` directory
- For example, options you set in the `etc/sysconfig/rsyslogd` file are passed to the `rsyslogd` daemon when it starts up.

Port numbers

- Packets of data go to and from your system over network interfaces through ports for each supported protocol (usually UDP or TCP).
- Common server ports inclue: SMTP (25), HTTPS (443), FTP (21), SSH (22), DNS (53), NTP (123), SNMP (161), Syslog (514)

Secure the server

Password Protection

- Do not use weak passwords
- Disallaow direct root login, instead use su or sudo to become root.
- Pluggable Authentication Module (PAM)
- Using key-based (passwordless) authentication

Firewalls

- iptables can allow, drop or reject packets.
- firewalld extends iptables functionality by creating zones.
- Example:

```
iptables -n -L -v --line-numbers
```

TCP Wrappers

- A TCP Wrapper is a library that provides simple access control and standardized logging for supported applications that accept connections over a network.
- Using the /etc/hosts.allow and /etc/hosts.deny files, you can allow or deny access to those services that have the TCP Wrappers features enabled (libwrap).

SELinux

- SELinux is in enforcing mode, doesn't impact applications but impacts services
- SELinux makes sure that a web server, FTP server, Samba server, or DNS server can access only a restricted set of files on the system (as defined by file contexts) and allow only a restricted set of features (as defined by Booleans and limited port access).

Configuration Files

- Within the configuration files of most services are values you can set to further secure the service. For example, for file servers and web servers, you can restrict access to certain files or data based on username, hostname, IP address of the client, or other attributes.

Managing Remote Access with the Secure Shell Service

- Most linux distributions include ssh client/server (`yum list installed | grep ssh`)
- Check sshd status (`systemctl status sshd.service`) and start if not running (`systemctl start sshd.service`)
- Connect to remote system: `ssh johndoe@10.140.67.23` (info will be saved at `$ cat .ssh/known_hosts`
- Transfer files (memo) to remote systems (/tmp): `$ rsync -avz rpmpkgs/ root@192.168.0.101:/home/`
- `scp` has limitations (lost attributes, symbolic links, repeated), consider `rsync` instead. In addition, `sftp` can be used.

Checking System Resources with System Activity Reporter (`sar`)

- The `sar` command is part of the sysstat package
- While `top` shows real time information, `sar` shows historical information
- Common usage include these options:

-u for CPU usage

-d for disk activity output

-n for network activity

Refer to the `sar, sadc, sa1`, and `sa2` man pages for more information on how sar data can be gathered and displayed.

```
[root@vmcentos7 log]# sar -h
Usage: sar [ options ] [ <interval> [ <count> ] ]
Main options and reports:
      -b    I/O and transfer rate statistics
      -B    Paging statistics
      -d    Block device statistics
      -F [ MOUNT ]
            Filesystems statistics
      -H    Hugepages utilization statistics
      -I { <int> | SUM | ALL | XALL }
            Interrupts statistics
      -m { <keyword> [,...] | ALL }
            Power management statistics
            Keywords are:
            CPU   CPU instantaneous clock frequency
            TEMP  Devices temperature
            USB   USB devices plugged into the system
      -n { <keyword> [,...] | ALL }
            Network statistics
            Keywords are:
            IP    IP traffic  (v4)
            ICMP  ICMP traffic(v4)
            TCP   TCP traffic (v4)
            UDP   UDP traffic (v4)
      -q    Queue length and load average statistics
      -r    Memory utilization statistics
      -R    Memory statistics
      -S    Swap space utilization statistics
      -u [ ALL ]
            CPU utilization statistics
      -v    Kernel table statistics
      -w    Task creation and system switching statistics
      -W    Swapping statistics
      -y    TTY device statistics
```

Managing Servers in the Enterprise

Deployment

- Servers are deployed over and over again with similar configuration
- Use PXE boot, and kickstart file to install Linux automatically
- Additional tools available – Chef, Puppet, Ansible, Vagrant

Monitoring

- Too many servers to monitor, requires centralized monitoring tool
- These tools collect metrics like – cpu, memory, disk, other activities and info
- Tools include – Nagios, SolarWinds, Zabbix, Centreon, Observium

Summary

- Although many different types of servers are available with Linux systems, the basic procedure for installing and configuring a server is essentially the same.
- The normal course of events is to install, configure, start, secure, and monitor your servers.
- Basic tasks that apply to all servers include using networking tools (particularly SSH tools) to log in, copy files, or execute remote commands.
- Tools for gathering data and reviewing the log data later are very important when administering Linux servers.
- The rsyslog facility can be used for local and remote logging.
- The sar facility gathers live data or plays back data gathered earlier at 10-minute intervals.
- To watch disk space, you can run df and du commands.

Ch22 – Understanding Basic Linux Security

Implementing basic security

Monitoring security

Auditing and reviewing security

Understanding Security Basics

- Implementing physical security
- Implementing disaster recovery
- Securing user accounts
- Securing passwords
- Securing the filesystem
- Managing software and services
- Advanced implementation

Implementing physical security

■ A lock or security alarm on the server room door

■ Access controls that allow only authorized access and identify who accessed the room and when the access occurred, such as a card key entry system

■ A sign stating “no unauthorized access allowed” on the door

■ Policies on who can access the room and when access may occur, for groups such as the cleaning crew, server administrators, and others

Implementing disaster recovery

- What data is to be included in backups
- Where backups are to be stored
- How long backups are maintained
- How backup media is rotated through storage

Some utilities – `rsync`, `Amanda`, `bacula`, `cpio`, `tar`, `dump/restore`

Securing user accounts

- One user per user account.

- Limit access to the root user account. All sudo use (who, what, when) is recorded in `/var/log/secure`

- Set expiration dates on temporary accounts.

```
# usermod -e 2099-01-01 tim
```

- Remove unused user accounts. Find files owned by user > Expire or disable account > backup files > remove or reassign files > delete account

Securing passwords

■ Do not use any variation of your login name or your full name.
■ Do not use a dictionary word.
■ Do not use proper names of any kind.
■ Do not use your phone number, address, family, or pet names.
■ Do not use website names.
■ Do not use any contiguous line of letters or numbers on the keyboard (such as “qwerty” or ”asdfg”).
■ Do not use any of the above with added numbers or punctuation to the front or end, or typed backward.

■ A password should be at least 15 to 25 characters in length.

■ A password should contain: lowercase, uppercase, numbers, and special characters

Securing the filesystem

■ Managing dangerous filesystem permissions (`rwxrwxrwx`)

■ Secure the password files (`/etc/passwd`)

- users should not be able to modify the /etc/passwd directly

■ Locking down the filesystem (`/etc/fstab`)

- You can set the nosuid option to prevent SUID and SGID permission-enabled executable programs running from there. Programs that need SUID and SGID permissions should not be stored in /home and are most likely malicious.

Managing software and services

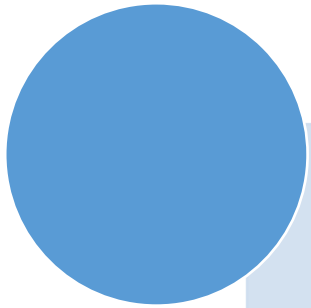
- Updating software packages
  - Remove unnecessary software
  - Frequently update existing software (bug fix + enhancement)
- Keeping up with security advisories
  - Check vendor alerts + other resources
- Disable unnecessary services
  - Speeds up system performances, improves security

`systemctl status|stop|disable` *`sshd`*

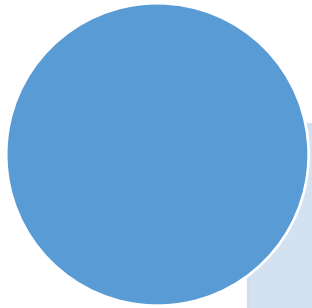
Advanced implementation

- Some other important security topics as you are planning your deployments - cryptography, Pluggable Authentication Modules (PAM), and SELinux.

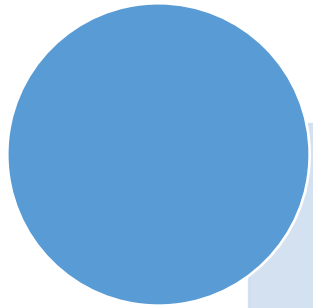
Monitoring Your Systems



Monitoring log files



Monitoring user accounts



Monitoring the filesystem

Conducting compliance reviews

- Similar to audits in other fields, such as accounting, audits can be conducted internally or by external personnel. These reviews can be as simple as someone sitting down and comparing implemented security to your company's stated policies. However, a more popular method is conducting audits using penetration testing.
- *Penetration testing* is an evaluation method used to test a computer system's security by simulating malicious attacks. It is also called pen testing and ethical hacking. No longer do you have to gather tools and the local neighborhood hacker to help you conduct these tests.

Conducting security reviews

Conducting a security review requires that you know current best security practices. There are several ways to stay informed about best security practices. The following is a brief list of organizations that can help you.

- United States Computer Emergency Readiness Team (CERT)
- The SANS Institute
- Gibson Research Corporation

Summary:

- Basic Linux security practices such as managing user accounts, securing passwords, and managing software and services form the foundation for all other security on your Linux system. With that foundation in place, ongoing monitoring of your system includes watching over system log files, checking for malicious intrusions, and monitoring the filesystem.

- Reviews of your security policies are also important to keep up on a regular basis. Audits assist in ensuring that your Linux system is secured and the proper security policies and practices are in place.

Ch25 – Securing Linux on a Network

Managing network services

Controlling access to network services

Implementing firewalls

Auditing Network Services

- A *network service* is any task that the computer performs requiring it to send and receive information over the network using some predefined set of rules – web, email, file, print server etc.
- A Linux server has the potential to provide thousands of services. Many of them are listed in the `/etc/services` file.
- Many Linux distributions come with unneeded network services running. An unnecessary service exposes your Linux system to malicious attacks.

Evaluating access to network services with `nmap`

- A wonderful tool to help you review your network services from a network standpoint is the `nmap` security scanner.

■ **TCP Connect port scan**—For this scan, nmap attempts to connect to ports using the Transmission Control Protocol (TCP) on the server. If a port is listening, the connect attempt succeeds.

■ **UDP port scan**—For this scan, nmap sends a UDP packet to every port on the system being scanned. UDP is another popular protocol in the TCP/IP network protocol suite. Unlike TCP, however, UDP is a connectionless protocol. If the port is listening and has a service that uses the UDP protocol, it responds to the scan.

https://nmap.org/book/man.html

Evaluating access to network services with `nmap`

state	description
open	This is the most dangerous state an nmap scan can report for a port. An open port indicates that a server has a service handling requests on this port. Think of it as a sign on the door, “Come on in! We are here to help you.” Of course, if you are offering a public service, you want the port to be open.
closed	A closed port is accessible, but there is no service waiting on the other side of this door. However, the scan status still indicates that there is a Linux server at this particular IP address.
filtered	This is the best state to secure a port that you don’t want anyone to access. It cannot be determined if a Linux server is actually at the scanned IP address. It is possible that a service could be listening on a particular port, but the firewall is blocking access to that port, effectively preventing any access to the service through the particular network interface.
unfiltered	The nmap scan sees the port but cannot determine if the port is open or closed.
open | filtered	The nmap scan sees the port but cannot determine if the port is open or filtered.
closed | filtered	The nmap scan sees the port but cannot determine if the port is closed or filtered.

Using `nmap` to audit your network services advertisements

- You probably want lots of people to visit your Web site (`httpd` service). You probably don't want everyone on the Internet able to access your SMB file shares (`smb` service).
- You may be tempted to skip the scans from inside your organization's internal network. Don't. Malicious activity often occurs from a company's own employees or by someone who has already penetrated external defenses.
- Security settings on various network components, such as the server's fi rewall and the company's routers, should all be considered when conducting audit scans.

Using `nmap` to audit your network services advertisements

- You probably want lots of people to visit your Web site (`httpd` service). You probably don't want everyone on the Internet able to access your SMB file shares (`smb` service).
- You may be tempted to skip the scans from inside your organization's internal network. Don't. Malicious activity often occurs from a company's own employees or by someone who has already penetrated external defenses.
- Security settings on various network components, such as the server's fi rewall and the company's routers, should all be considered when conducting audit scans.

Controlling access to network services

- Completely disabling an unused service is fine, but for needed network services, you must set up access control. This needed access control can be accomplished, via the `/etc/hosts.allow` and `/etc/hosts.deny` files, for selected services on Linux systems that incorporate TCP Wrapper support.

1. The `hosts.allow` file is checked.
   - If the remote system's address is listed:
     - Access is allowed.
     - No further TCP Wrapper checks are made.
   - If the remote system's address is *not* listed, the TCP Wrapper check process continues on to the `hosts.deny` file.

2. The `hosts.deny` file is checked.
   - If the remote system's address is listed, access is denied.
   - If the remote system's address is *not* listed, access is allowed.

Working with Firewalls

Firewalls can be placed into different categories, depending upon their function. Each category has an important place in securing your server and network.

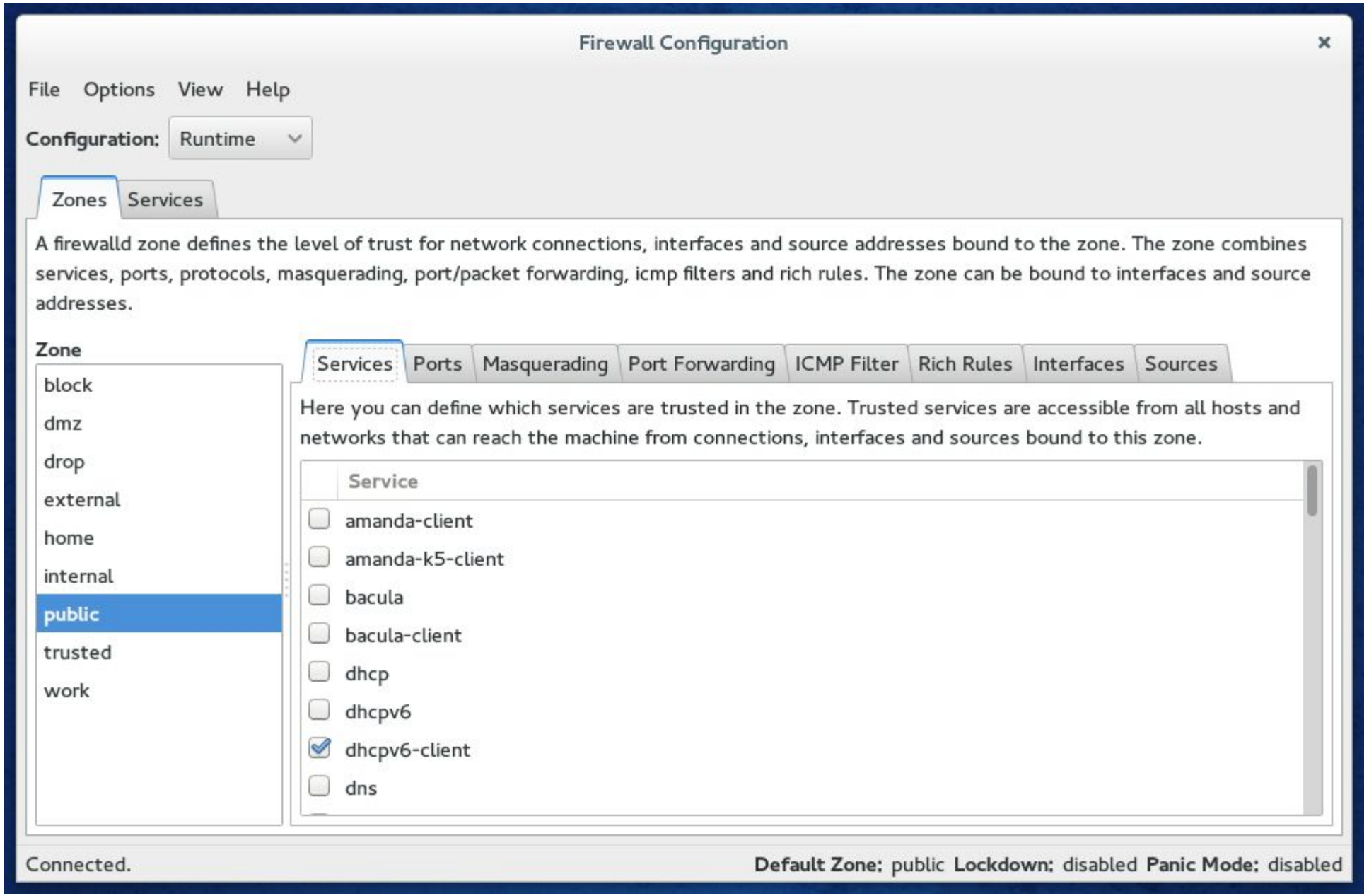
■ A firewall is either network-based or host-based. – Network-based one is protecting the entire network or subnet. A host-based firewall is one that is running on and protecting an individual host or server.

■ A firewall is either a hardware or a software firewall. Firewalls can be located on network devices, such as routers. Firewalls can be located on a computer system as an application.

■ A firewall is either a network-layer filter or application-layer filter. A firewall that examines individual network packets is also called a *packet filter*. An application-layer firewall filters at the higher layers of the OSI reference model.

Implementing firewalls

- On a Linux system, the firewall is a host-based, network-layer, software firewall managed by the `iptables` utility. With `iptables`, you can create a series of rules for every network packet coming through your Linux server. You can fine-tune the rules to allow network traffic from one location but not from another. These rules essentially make up a network access control list for your Linux server.
- Fedora, RHEL, and other Linux distributions have added the `firewalld` service on top of `iptables`, to provide a more dynamic way of managing firewall rules.


Firewall Configuration
File Options View Help
Configuration: Runtime
Zones Services
A firewalld zone defines the level of trust for network connections, interfaces and source addresses bound to the zone. The zone combines services, ports, protocols, masquerading, port/packet forwarding, icmp filters and rich rules. The zone can be bound to interfaces and source addresses.
Zone
block
dmz
drop
external
home
internal
public
trusted
work
Services Ports Masquerading Port Forwarding ICMP Filter Rich Rules Interfaces Sources
Here you can define which services are trusted in the zone. Trusted services are accessible from all hosts and networks that can reach the machine from connections, interfaces and sources bound to this zone.
Service
amanda-client
amanda-k5-client
bacula
bacula-client
dhcp
dhcpv6
dhcpv6-client
dns
Connected.
Default Zone: public Lockdown: disabled Panic Mode: disabled

Protecting your network services can be simplified after you determine and remove any unneeded network services. The `nmap` utility helps you here. Also, you can use `nmap` to audit your Linux server's advertising of network services. These audits assist in determining what firewall modifications are needed.

For needed network services, access control must be implemented. TCP wrappers can assist in this activity. On a per-service basis, access can be allowed or denied, fine-tuning access to each network service.

Recent versions of Fedora and RHEL have added the `firewalld` service as a front-end to the `iptables` firewall facility that is built into the Linux kernel. Using the `firewalld-config` window, you can easily open ports in your firewall to allow access to selected services. The `netfilter/iptables` firewall facility is a host-based, network-layer, software firewall. It is managed by the `iptables` and `ip6tables` utilities. With these utilities, a series of policies and rules can be created for every network packet coming through your Linux server. These policies and rules essentially make up an access control list for your Linux server network.