Let's Lose the Shock-Mystery of Linux - A Pain-free Intro

Defcon 33 – BIC Village

D.J. Davis, A+, CISSP

(ZeroRingDefender)

August 8, 2025

Objectives

Overview, Awareness, General understanding of Linux OS

We talk about:

- Microsoft Windows OS vs Linux
- Linux Distributions
- Command line
- Files, file systems
- Design features of Unix / Linux

Our Game Plan

- We will not go "heavy" into comparing distributions
- Installing Linux is covered in the online appendix to this presentation
- This slide deck will be available on Github

~~ Kindly defer questions to the end of the presentation

Disclaimer --

This presentation is not affiliated with my work. These are my own personal ideas. Feel free to choose and use selectively.

About D.J.

- Background:

  Ops, Dev, Mainframe Eng, Network Eng, WAN Design Eng, IT Integration, Sustaining Eng, Computer Security

- Locale:

  Live/work/play in Washington DC region

- My humble beginnings … …

  COBOL/CICS, EasyTrieve Plus, Business BASIC (Electronic Cash Register PC, police/dispatch Midrange)

- Alma Mater:

  Virginia Commonwealth University - Go Rams!!

  BS Business, Info Sys; MS Business, Info Sys - IT Management

Reasons to Learn / Use Linux

- The most-used OS for web servers, other servers on Internet
- Generally free to acquire and use. Most software is open source, free
- Will run most all computer language compilers, interpreters
- Most versions will run in 4 GB RAM up to a Supercomputer
- Some versions will run in 2 GB, 1 GB or less
- Linux is the basis of the Android OS, Apple MacOS
- Linux is an underlying component of VMware, some Cisco equipment, and is becoming a component in MS Windows
- Great for having, using, learning many computer languages
- Great for studying an operating system, its scripts, and its underlying computer code
- Excellent tools for hardware/software hacking and Security work
- A rich suite of commands including dd, grep !!!
- Very stable, reliable, and powerful OS. **Essentially a FREE version of Unix**

Annd … … What is Special About Unix ???

- The first port-able operating system. Starting with version 4, almost all of Unix is written in the high-level C language (machine-independent) instead of low-level, machine-dependent Assembly language. This makes it easier for the OS to be ported / copied / moved / implemented to different computer platforms
- Modular design. “Unix philosophy”: An OS should provide a simple set of tools; each which performs a limited, well-defined function
- Inter-process communication between programs/processes (“pipes | ”). The output of one program can become the input of another program on the command line
- Program and command input/output redirection < , > , >>
- 3 standard files are opened automatically for each process: stdin, stdout, stderr

Partial List of Linux Distributions … … …

- Puppy Linux **Kali** **Ubuntu** Debian CentOS Linux CentOS Stream
  Red Hat Enterprise Linux (RHEL) Gentoo **Fedora**
  OpenSUSE Scientific Linux CloudLinux Elementary OSLinux
  Mint Arch Linux Manjaro Oracle Linux Slackware Mageia
  Clear Linux Rocky Linux AlmaLinux Asahi Linux Lubuntu
  SUSE Linux Knoppix VzLinux Peppermint OS Zorin OS
  BlackArch Linux SUSE Liberty Linux Navy Linux Tizen

Some Differences Between Linux commands and MS Windows commands

clear vs cls

ls vs dir

traceroute vs tracert

grep vs find

format, parted vs diskpart

ps vs tasklist

Filepath sep: / vs \

cp vs copy

date vs date / time

man vs help

Device: /dev/null vs NUL:

rm vs del, erase

mv vs rename

cd vs cd

mkfs vs format

(*) read vs pause

Linux Live vs MS doesn't have USB-bootable version except 3rd-party Live11

(*) To implement functionality of Microsoft's pause command, we use: read -n 1 -s -r -p "Press any key to continue . . . " ; echo

Other Differences Between Linux and MS Windows (1)

- **Linux commands are case-sensitive**; Windows commands are not
- Linux unified file system has one root directory for the computer system
- Microsoft OSes have a root directory for each disk drive
- Amount of work in GUI vs CLI: Linux – more CLI vs Windows – more GUI
- Free utilities and languages available: free, unlimited and ubiquitous in Linux
- Linux GUIs can vary notably. Windows GUI is fairly standard across versions
- Command-line interpreters: bash, several others vs only MS-DOS style Command Prompt, Powershell
- Scripting: bash scripts vs MS-DOS-style Batch files, Powershell scripts

Other Differences Between Linux and MS Windows (2)

- Languages installed by default in Linux: tcl, python, bash, perl
- File systems: ext2/ext3/ext4 + FAT32, FAT64, NTFS vs NTFS, FAT32, FAT64
- Super-user account: root vs Administrator
- Windows Administrator account can be renamed; Linux root is not renamed
- CLIs installed by default in Linux (Ubuntu): dash sh tclsh bash rbash
- Daemon vs Service
- Mounting / unmounting disk volumes, Flash drives is easier in Windows

- Command General Format:

  Linux: <command> <switches e.g. -s -p -sp --max> <value 1> <value 2> ...

  Windows: <command> <value 1> <value 2> ... <switches e.g. /s /p >

Where did Linux come from? (1)

- In 1969, AT&T / Bell Labs begins to write Unix for use in the Bell (telephone) system
- December 1969 Linus Torvalds is born
- In 1972 Unix is rewritten in C. It was written in Assembly language
- In late 1970s, AT&T begins to license Unix for use in universities and commercial use
- In 1987 Andrew Tanenbaum releases MINIX with its complete source code made available to universities for study in courses and research
- In 1987: Computer science textbook “Operating Systems: Design and Implementation” is written by Andrew S. Tanenbaum, with help from Albert S. Woodhull. The book describes the principles of operating systems and demonstrates the source code of Tanenbaum's MINIX, a free Unix-like operating system that is designed for teaching purposes

Where did Linux come from? (2)

- In 1988 Linus Torvalds starts attending the University of Helsinki, interrupts in 1989 for Military Service, and returns to the University around 1990
- In 1991 Linus buys an 80386 clone, acquires a copy of GTS MINIX, and begins his work on the Linux kernel
- The first Linux prototypes are released publicly in late 1991
- Linux Version 1.0 is released on March 14, 1994
- Torvalds first encounters the GNU Project in fall of 1991 when another Swedish-speaking computer science student, Lars Wirzenius, took him to the University of Technology to listen to free software guru Richard Stallman's speech. Torvalds would ultimately switch his original license (which forbade commercial use) to Stallman's GNU General Public License version 2 (GPLv2) for his Linux kernel after complaints of distributors being unable to recoup their costs due to a non-commercial clause

What is Linux - Components

- OS - software... makes use of a box of HW... run programs... communicate with it

- Kernel (the brain, heart, central nervous system of an OS --more on the next slide …)
- System Boot loader – LILO, GNU **GRUB**, others
- Command-line Interpreter (CLI) - sh, ksh, Bourne, **bash**
- Graphical User Interface (GUI) – GNOME (simpler, easier), KDE (more configurable)
- Package Manager - RedHat Package Manager (rpm, yum, dnf) or Debian/APT (apt get)
- System startup scripts – SystemV / SysVinit (older), **SystemD** (newer)
- File System - EXT3 **EXT4** XFS ZFS Btrfs FAT32 FAT64
- Utility programs / commands

- Different Distributions of Linux (e.g. Fedora, Ubuntu, Kali) have different packages installed, by default

What the Kernel Is; What the Kernel Does

Central, core of operating system. Low-level functions:

- Memory - allocation / Deallocation
- CPU scheduling, Process creation and control
- I/O - communication with hardware via Device Drivers
- Inter-process communication
- Date/Time services, Timers
- Resource protection - file and process permissions
- System Call interface (counted 638 system calls)

Kernel functions:

- Allocates memory to programs
- Allocates CPU time to programs
- Implements a security model to protect/isolate programs, users, the OS
- Provides a programming interface (APIs, System Calls “Syscalls”) so programs can use the resources on the computer (ex. memory, CPU, hardware, files, system clock)

Or… … What is NOT in the Kernel

- CLI, GUI
- Commands, Utility programs
- User programs
- Sometimes hardware drivers, file system drivers, other software modules

Current Linux Kernel

- Version: 6.16 (even minor numbers are prod, odd numbers are test)
- Where to find kernel file: /boot
- To display kernel version: uname –a or uname -r

uname -r

6.11.0-29-generic

uname -a

Linux Dell 6.11.0-29-generic #29~24.04.1-Ubuntu SMP PREMPT_DYNAMIC Thu June 26 14:16:59 UTC 2 x86_64 GNU/Linux

https://github.com/torvalds/linux Source code

https://www.kernel.org/doc/html/latest Docs, and link to Linus Torvalds' kernel source code on Git

Syscall, strace illustration

root@Dell:**/usr/include/x86_64-linux-gnu/asm** # ls

amd_	hsmp.h	hwcap2.h	mtrr.h	sembuf.h	swab.h
a.out.h	ioctl.h	param.h	setup.h	termbits.h	
auxvec.h	ioctls.h	perf_regs.h	sgx.h	termios.h	
bitsperlong.h	ipcbuf.h	poll.h	shmbuf.h	types.h	
boot.h	ist.h	posix_types_32.h	sigcontext32.h	ucontext.h	
bootparam.h	kvm.h	posix_types_64.h	sigcontext.h	unistd_32.h	
bpf_perf_event.h	kvm_para.h	posix_types.h	siginfo.h	unistd_64.h	
byteorder.h	kvm_perf.h	posix_types_x32.h	signal.h	unistd.h	
debugreg.h	ldt.h	prctl.h	socket.h	unistd_x32.h	
e820.h	mce.h	processor-flags.h	sockios.h	vm86.h	
errno.h	mman.h	ptrace-abi.h	statfs.h	vmx.h	
fcntl.h	msgbuf.h	ptrace.h	stat.h	vsyscall.h	
hw_breakpoint.h	msr.h	resource.h	svm.h		

Syscall, strace illustration

root@Dell:/usr/include/x86_64-linux-gnu/asm# **less unistd_32.h**

```
#ifndef _ASM_UNISTD_32_H
#define _ASM_UNISTD_32_H

#define __NR_restart_syscall 0
#define __NR_exit 1
#define __NR_fork 2
#define __NR_read 3
#define __NR_write 4
#define __NR_open 5
#define __NR_close 6
#define __NR_waitpid 7
#define __NR_creat 8
#define __NR_link 9
#define __NR_unlink 10
#define __NR_execve 11
#define __NR_chdir 12
#define __NR_time 13
#define __NR_mknod 14
#define __NR_chmod 15
#define __NR_lchown 16
```

Syscall, strace illustration

```
root@Dell:/home/me/pgms# strace ./hello_asm
execve("./hello_asm", ["./hello_asm"], 0x7ffdedfeb650 /* 49 vars */) = 0
[ Process PID=69753 runs in 32 bit mode. ]
write(1, "Hello, world!+\n\0", 16Hello, world!+
)     = 16
exit(0)                                 = ?
+++ exited with 0 +++
```

```
; a comment - hello_asm.asm
section .data
    helloString db "Hello, world!",10,0      ; String, LF, ASCIIZ
    len1 equ $ - helloString                 ; length of string (15)

section .text
    global  _start
_start:

                                             ; Write String to Stdout w Syscall
    mov     edx,len1                         ; load string length
    mov     ecx,helloString                  ; load pointer to the string to write
    mov     ebx,1                            ; load file handle (1 is stdout)
    mov     eax,4                            ; load system call number (sys_write)
    int     0x80                             ; invoke Interrupt to call OS

                                             ;Exit Program
    mov     ebx,0                            ; load exit code (0 = normal completion)
    mov     eax,1                            ; load system call number (sys_exit)
    int     0x80                             ; invoke Interrupt to call OS
```

Syscall, strace illustration

```
root@Dell:/home/me/pgms# strace ./hello_c
execve("./hello_c", ["./hello_c"], 0x7ffc9066e640 /* 49 vars */) = 0
brk(NULL)                               = 0x8900000
mmap(NULL, 8192, PROT_READ|PROT_WRITE, MAP_PRIVATE|MAP_ANONYMOUS, -1, 0) = 0x71054c528000
access("/etc/ld.so.preload", R_OK)      = -1 ENOENT (No such file or directory)
openat(AT_FDCWD, "/etc/ld.so.cache", O_RDONLY|O_CLOEXEC) = 3
fstat(3, {st_mode=S_IFREG|0644, st_size=62899, ...}) = 0
mmap(NULL, 62899, PROT_READ, MAP_PRIVATE, 3, 0) = 0x71054c518000
close(3)                                = 0
openat(AT_FDCWD, "/lib/x86_64-linux-gnu/libc.so.6", O_RDONLY|O_CLOEXEC) = 3
read(3, "\177ELF\2\1\1\3\0\0\0\0\0\0\0\0\3\0>\0\1\0\0\0\220\243\2\0\0\0\0\0"..., 832) = 832
pread64(3, "\6\0\0\0\4\0\0\0@\0\0\0\0\0\0\0@\0\0\0\0\0\0\0@\0\0\0\0\0\0\0"..., 784, 64) = 784
fstat(3, {st_mode=S_IFREG|0755, st_size=2125328, ...}) = 0
pread64(3, "\6\0\0\0\4\0\0\0@\0\0\0\0\0\0\0@\0\0\0\0\0\0\0@\0\0\0\0\0\0\0"..., 784, 64) = 784
mmap(NULL, 2170256, PROT_READ, MAP_PRIVATE|MAP_DENYWRITE, 3, 0) = 0x71054c200000
mmap(0x71054c228000, 1605632, PROT_READ|PROT_EXEC, MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE, 3, 0x28000) = 0x71054c228000
mmap(0x71054c3b0000, 323584, PROT_READ, MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE, 3, 0x1b0000) = 0x71054c3b0000
mmap(0x71054c3ff000, 24576, PROT_READ|PROT_WRITE, MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE, 3, 0x1fe000) = 0x71054c3ff000
mmap(0x71054c405000, 52624, PROT_READ|PROT_WRITE, MAP_PRIVATE|MAP_FIXED|MAP_ANONYMOUS, -1, 0) = 0x71054c405000
close(3)                                = 0
mmap(NULL, 12288, PROT_READ|PROT_WRITE, MAP_PRIVATE|MAP_ANONYMOUS, -1, 0) = 0x71054c515000
arch_prctl(ARCH_SET_FS, 0x71054c515740) = 0
set_tid_address(0x71054c515a10)         = 69760
set_robust_list(0x71054c515a20, 24)     = 0
rseq(0x71054c516060, 0x20, 0, 0x53053053) = 0
mprotect(0x71054c3ff000, 16384, PROT_READ) = 0
mprotect(0x403000, 4096, PROT_READ)     = 0
mprotect(0x71054c566000, 8192, PROT_READ) = 0
prlimit64(0, RLIMIT_STACK, NULL, {rlim_cur=8192*1024, rlim_max=RLIM64_INFINITY}) = 0
munmap(0x71054c518000, 62899)           = 0
fstat(1, {st_mode=S_IFCHR|0620, st_rdev=makedev(0x88, 0), ...}) = 0
getrandom("\xda\x73\x02\xc3\x0c\x6d\x1d\xac", 8, GRND_NONBLOCK) = 8
brk(NULL)                               = 0x8900000
brk(0x8921000)                          = 0x8921000
write(1, "Hello, World \n", 14Hello, World
)       = 14
exit_group(0)                           = ?
+++ exited with 0 +++
```

```c
/* a comment - hello_c.c    */        // also a comment
// compile with: gcc hello_c.c -o hello_c
#include <stdio.h>
int main ()
{
   printf ("Hello, World \n");
}
```

How to Log In

- We log in to the GUI or text command line depending on computer's Run Level
  - Runlevel 0 - Shutdown / Power-off / Halt
  - Runlevel 1 – Text-based Rescue mode / Operator console only
  - Runlevel 2 – Text-based Multi-user without networking
  - Runlevel 3 - Text-based Command-line Multi-user with networking
  - Runlevel 4 – (Undefined, User definable)
  - Runlevel 5 - GUI Multi-user with networking
  - Runlevel 6 – Reboot

- Runlevel 3 - Text-based, local or network (ssh) Enter Username / Password
- Runlevel 5 – GUI (log in via GUI or ssh). Select/enter Username. Enter Password
- In Debian, Ubuntu, Runlevels 2 thru 4 are the same as Runlevel 5 (GUI)
- A change in the Runlevel calls associated shutdown and startup scripts that place the system in the desired mode

What's Available in the GUI

- Productivity programs - Libre Office (documents, spreadsheets, presentation, drawing)
- Calc (Calculator by The GOME Project -U)Files browser (Files, The GNOME Project -U)
- E-mail / (Mozilla Thunderbird -U)
- Calendar (Calendar by The GNOME Project -U)
- Software Center / software updater
- Games (AilseRiot Solitaire, Mines, -U)
- Text Editor (gedit -F) / Text Editor (Text Editor The GNOME Project -U)
- Terminal (GNOME Terminal -U)
- Settings (typical settings like MS-Windows, Android, iOS)
- System Utilities (Trash, Drivers, Sys Monitors, Image/Doc viewers, Music player / Video player (Videos/Totem -U)
- Web browser (Firefox -U)
- Help
- Remote Desktop Client (Remmina)
- Camera app (Cheese GNOME -U)

Using the Terminal...

date

uptime

clear

exit

top

press q to exit the "top" program

whoami

who

who –a

free

runlevel

time

Try these three, separately: time date time uptime time free

Using the Terminal...

echo

echo "This is a message“

echo "My Commands" ; date ; uptime ; whoami ; who –a

ps

ps -a

Using the Terminal...

man

man uptime

press spacebar to page; press q to exit the "top" program

df	Disk free; displayed in disk blocks
df -h	Displayed in human-readable units: kB, MB, GB
df -k	Displayed in kB
ls	List files (directory information; default is filenames only)
ls -l	List files in long format (details: permissions, ownership, size, date, filename)

Intro to the Linux (unified) File System Organization

- / - root of Linux file system
- /boot - Linux kernel files, boot files
- /etc - (pronounced: ET-sy) System config files including password/shadow file
- /home - home directories of users
- /root - home directory for root superuser
- /media - mount point for temporary devices such as USB drives/dev - system devices (terminals, disk names, null device, etc)
- /tmp - directory to store temporary files
- /usr - **U**NIX **S**ystem **R**esources. System-wide, read-only files incl programs, libraries, documentation
- /usr/bin - regular, non-privileged commands (binary programs)
- /usr/sbin - superuser commands (binary programs)
- /usr/lib lib32 lib64 - 32-bit and 64-bit program library files
- /bin - link to /usr/bin
- /bin/sbin - link to /usr/sbin
- /var - logs, spool files, backups, etc
- /proc - Information on running processes, system data. Usually read-only
- /sys - parameters for the Linux kernel (can read and set)

Moving around the filesystem (1)

pwd ls cd

ls -l ls -a ls -ltr ls -Ral

.filename files are hidden. Use ls –a to list

. and .. this directory; parent directory

cd absolute vs cd relative

by using full path or by using the . and .. placeholders

Moving around the filesystem (2)

chmod - Change mode (file permissions)

chown - Change owner

Note: You can change your file to ownership of someone else and lose access to it

stat - provides extended status information on a file

```
root@hp1:~# stat /etc/passwd  File: /etc/passwd  Size: 2717                Blocks: 8         IO Block: 4096   regular
fileDevice: 179,2Inode: 2099909     Links: 1Access: (0644/-rw-r--r--)  Uid: (    0/    root)   Gid: (    0/    root)Access: 2025-
03-23 10:44:44.969604232 -0400Modify: 2024-09-13 14:45:15.399999846 -0400Change: 2024-09-13
14:45:15.399999846 -0400 Birth: 2024-09-13 14:45:15.399999846 -0400
```

File Permissions

ls -l

```
-rw-r--r--  1  brooks-n    inventors     247394   Jan  02 2025 boeing-rockets-missles-planes
-rw-r--r--  1  dean-m      inventors     247394   Jan  07 2025 ibm-pc-color-monitor
-rw-r--r--  1  fox-b       inventors     247394   Jan  03 2025 linux-bash-shell
-rw-r--r--  1  west-j      inventors     247394   Jan  09 2025 new-electret-microphone
-rw-r--r--  1  johnson-l   inventors     247394   Jan  08 2025 super-soaker-fun
-rw-r--r--  1  Jackson-s   inventors     247394   Jan  02 2025 touchtone-phone
-rw-r--r--  1  croak-m     inventors     247394   Jan  04 2025 voip-sounds-to-life
```

User (u)	Group (g)	Other (o) i.e everyone on the system
Read Write eXecute	Read Write eXecute	Read Write eXecute
4 + 2 + 1	4 + 2 + 1	4 + 2 + 1
= 7	= 7	= 7

rwx = 7 r-x = 5 rw- = 6 r-- = 4 - - - = 0 (no access)

Example: chown 740 boeing-rockets-missles-planes (removes read access from others who are not owner or in group)

File Permissions

For Files:

Read means – the user, group members, or others (everyone) can read the file contents
Write means – the user, group members, or others (everyone) can re-write the file contents
eXecute means – the user, group members, or others (everyone) can run the file if it is a script or program

For Directories:

Read means – the user, group members, or others can list files (ls) and copy files in directory
Write means – the user, group, or others can add and delete files in directory (this requires execute permission also)
eXecute means – the user, group, or others (everyone) can enter the directory

Note: a user can have permission to read a file without having permission to enter the directory

The root super-user generally has access to all user files

/proc Files and Directories (1)

- There is a sub-directory (PID number) for every process
- acpi
- asound
- bootconfig
- buddyinfo
- cgroups
- cmdline
- consoles
- cpuinfo
- crypto
- devices
- diskstats
- dma
- execdomains
- fb
- filesystems
- fs
- interrupts
- iomem
- Ioports
- kallsyms
- kcore
- keys
- key-users
- kmsg
- kpagecgroup
- kpagecount
- kpageflags
- loadavg
- Locksmdsta
- meminfo
- miscmodules
- mounts
- mtrrpage
- typeinfo
- partitions
- schedstat
- slabinfo
- softirqsstat
- swapssys
- rq-trigger
- timer_list
- uptime
- version
- version_signature
- vmallocinfo
- vmstat
- zoneinfo

/proc Files (2)

- cmdline - bootstrap / bootloader command-line, parameters from system start
- cpuinfo - detailed data on system CPUs and their feature set
- loadavg - system 1 min, 5 min, 15 min load averages for top, uptime commands
- meminfo - sizes of memory allocation by category
- mounts - mountable file systems
- swaps - info on swap files
- timer_list - system timers
- uptime - system uptime data that is displayed with the uptime command
- version - version info of running OS
- version_signature - version info of running OS
- vmallocinfo - detailed virtual memory allocations
- vmstat - memory info that is displayed with vmstat command

More On Using the Terminal

Command-line editing - up-arrow (previous commands), backspacing and typing

Ctrl-C	Terminate a running program
Ctrl-S, Q	Pause, resume display of a program
Ctrl-D	End of input to a program that is accepting input
Ctrl-Z	Place a running program in the background
fg bg	Place program execution in the Foreground or Background

Copy / paste in CLI vs Graphical editor

CLI:	Shift + Ctrl + C / Shift + Ctrl + V
Graphical editor:	Ctrl + C / Ctrl + V

3 Files are Auto-Associated with Each Running Process

stdin - Standard input (file # 0) by default, the keyboard

stdout – Standard output (file # 1) by default, the user's screen

stderr - Standard error – error messages (file # 2) by default, the user's screen

- In Linux, everything is considered as a file
- The standard input/output can be redirected to files. Also, it can be sent (piped) to other programs because those programs are reading / writing files

In Linux, EVERYTHING is a File

- What about Windows – Windows uses APIs. A good example is Powershell applets. “Billions” of them … …
- FILES in Linux: Keyboard. Terminal display. Disk partitions. Entire disk drives. OS parameters
- Many system operations can be performed by opening, reading/writing, and closing a file

More On Using the Terminal

>	Output Redirection write	>>	Output Redirection append
<	Input Redirection		
2>&1	Combine stderr into stdout		
|	Pipe		
&	background execution		

```
ls  >  list_of_my_files.txt
sort  <  unsorted_groceries.txt  >  sorted_groceries.txt
Find / -name syscall* 2>&1 /home/testuser1/syscall_search_results.txt
cat  /etc/password  |  less
who  |  wc -l      ;    ls -l  |  wc -l
gedit  hello_asm.asm  &
```

What can be a Command (file) in Linux?

- Program... compiled, interpreted, or a script. Run from the command line / CLI (not usually run from GUI)
- Can be a file on disk
- Can be Built into shell/CLI (see: man builtins)
- Generally available system-wide (from any directory, by most/all users)
- Can be restricted to certain users (remember /usr/bin vs /usr/sbin)

What If Our System Doesn't Have a Command that We Desire ?

root@hp1:/home/user5/Pgm # iostat

Command 'iostat' not found, but can be installed with:

apt install sysstat

root@hp1:/home/user5/Pgm #

Sometimes, (if you are running as root), Linux will ask if you want the package to be installed

How Linux Uses All Available Memory (1)

- Linux attempts to make good use of all free memory (RAM)
- Slowly, over time, Linux allocates free memories to Buffers, Cache
- Buffers: an extension of the virtual memory sub-system
- Cache: stores file system blocks of data that are read from disk. When a file is read from disk (including directory blocks), a subsequent reading of the file or directory data will be from fast memory; not from the slower disk
- This behavior improves disk performance
- But… it can appear to some programs that memory is up to 99% used
- The surplus memory in the Buffers and Cache is available and is allocated to programs as soon as it is needed
- adage: Unused memory is WASTED memory

Commands: top, free

see article: Linux Behavior with Unused System Memory - Why 99% Memory Usage Might Not Be Alarming

How Linux Uses All Available Memory (2)

vi session is ended:

Mem Total	Mem Used	Mem Free	Mem Buffers	Mem Cached
126256	30520	95736	780	8352
-/+ buf/cache	21388	104868		
Swap Total	Swap Used	Swap Free		
262136	64	262072		

System is Idle for five minutes:

Mem Total	Mem Used	Mem Free	Mem Buffers	Mem Cached
126256	30392	95864	**800**	8352
-/+ buf/cache	21240	105016		
Swap Total	Swap Used	Swap Free		
262136	64	262072		

After being idle, the Buffers increases by 20 kB.

System is Idle for 30 additional minutes:

Mem Total	Mem Used	Mem Free	Mem Buffers	Mem Cached
126256	30724	95532	**932**	**8544**
-/+ buf/cache	21248	105008		
Swap Total	Swap Used	Swap Free		
262136	64	262072		

A half hour after the previous check, the Buffers increase by 132 kB and the Cache increases by 192 kB.

Compatibility of Windows (USB) Drives

- Frustration: You're backing up something to a USB and the program stops after copying 4 GB
- A disk formatted with FAT32 has a max file size of 4GB. USB drives are formatted with Windows in mind
- Windows standard file types (Win 10/11): FAT32, NTFS, exFAT
- Linux and Mac do not support NTFS; use exFAT
- In Windows it is easy to format a USB drive as exFAT
- Depending on the distro exFAT support *might* need to be enabled in Linux. This is an easy google search

Some Modular Components in Linux

- Kernel
- Boot loader
- Device drivers
- File systems
- Command-line interpreters / Shells, GUIs
- Package Manager
- System Startup scripts

Linux and Computer Viruses, Malware

- Linux and its file system is sometimes thought to be more secure, more resilient to viruses
- This might be debatable, but a standard Linux user typically has less access to write system files than a Windows system
- The Linux file permissions facility is arguably simpler than Windows
- In some Linux distributions, logging in to the computer as root is often disabled by default

su and sudo

- The CLI user prompt is $ while the root prompt #
- In several default Linux installations, a root password is not entered
- The user runs privileged commands with su (switch user) or sudo
- /etc/sudoers contains list of users who can su / sudo
- The original user is (indirectly) added to the file during the install process
- The installer user might add the user to the admin group, and the admin group might be spelled adm

dd (1)

- Incredibly handy utility to copying & converting files
- Can copy files, disk partitions, MBR, disk drives
- Backup and restore files by copying between files and partitions / disks
- Handy for backing up a moderate size disk (< 1 TB) to a USB drive
- By default, the output file is the same size as the input file. All free space is copied

dd if=/dev/xxxxxx of=/<filepath>/<filename>
dd if=/dev/mmcblk0 of=pc2-disk-image-Aug-23-2024.img

- dd if=/<filepath/<filename> of=/dev/xxxxxx
- Example: dd if=/run/media/liveuser/Seagate Backup Plus Drive/pc2-disk-image-Aug-23-2024.img of=/dev/mmcblk0

dd (2)

- The destination file can be compressed. For example: gzip
- Often the output file size is about 15% of the original
- A backup/restore with compression does take longer to run
- Interesting origin of dd: https://en.wikipedia.org/wiki/Dd_%28Unix%29

dd if=/dev/xxxxxx | gzip -c > /<filepath>/<filename>

dd if=/dev/mmcblk0 | gzip -c > pc2-disk-image-Aug-23-2024.img.gzip

gunzip -c /<filepath>/<filename> | dd of=/dev/xxxxxx

gunzip -c /run/media/liveuser/Seagate\ Backup\ Plus\ Drive/pc2-disk-image-Aug-23-2024.img.gzip | dd of=/dev/mmcblk0

Things that are Awkward or Buggy in Linux

- Different distros have different commands (from different packages. example: adduser, useradd, or both commands installed)
- mount command and parameter options can be difficult for unusual disk formats
- If you pull out a USB without unmounting the filesystem, Linux can get nasty with you regarding that filesystem
- Formatting a disk drive can be somewhat "involved“
- udev rules are buggy for custom VMs and hardware
- Fedora: Upgrade only 1 or 2 versions at a time. eg. V37 to 39 then to 40
- Fedora: Apply same-version updates BEFORE upgrading to new version!

Summary

- Linux comes from Minix (educational look-alike of Unix), kernel by Linus, Other OS software by GNU
- We have discussed several easy differences between Linux and Windows
- We have looked at some core factors that go into a Linux Distribution
- We have listed some programs available in a Linux GUI
- We have looked at some command-line features and Linux design

q & a

Now GO PLAY with it!!!

Thank you!

D.J. Davis (ZeroRingDefender)
ZeroRingD@gmail.com
Twitter/X: @ZeroRingD

https://github.com/IPV3/DC33-BIC/

< Slide deck - Set up, back up, restore Linux - Other resources />