

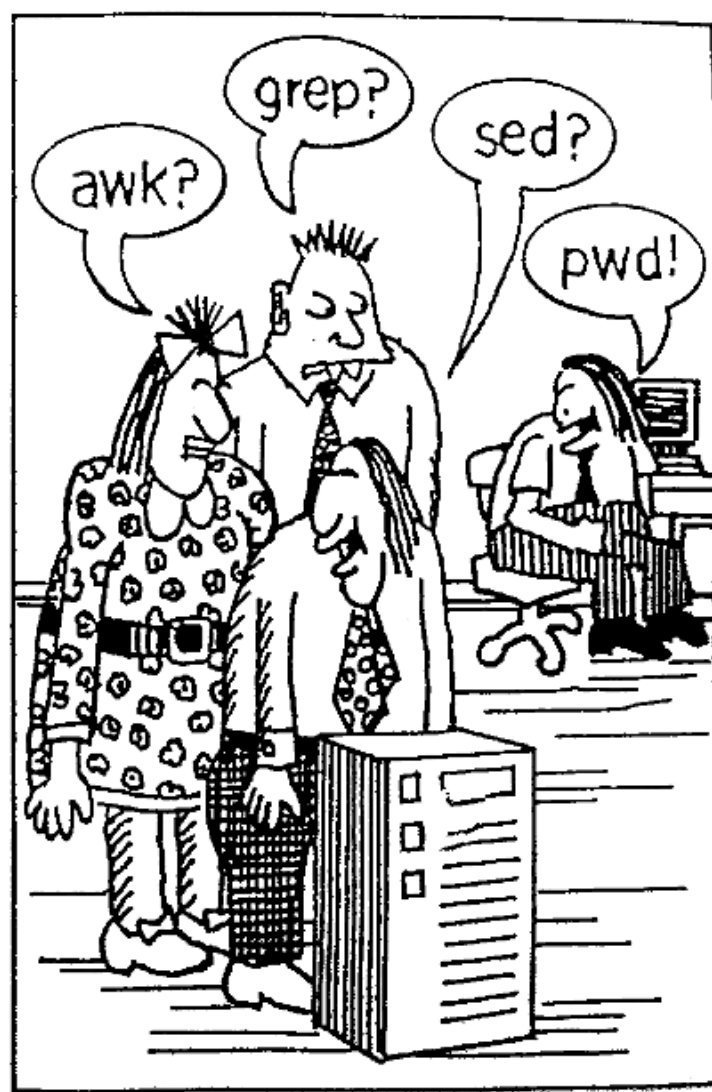
GNU/LINUX TUTORIAL

PART I: BASICS

§1: GNU, LINUX, AND DISTRIBUTIONS

WHY LINUX AND COMMAND LINE? 🤔

- One of the best operating systems.
- Efficient work flow from the terminal.
- Remote access is via command line.
- Linux is everywhere: servers, containers, IoT.



ORIGINS

- **UNIX**: operating system released in 1973. Written in C.
- Has inspired many **UNIX-like** operating systems.
- **POSIX**: a family of standards defining UNIX-like systems.

GNU/LINUX = GNU + LINUX

- **Linux**: free kernel written by Linus Torvalds. In collaborative development since 1991. Millions of lines of code in C.
- **GNU** (= *GNU's not a UNIX*) Project, founded by Richard Stallman in 1983: together with the kernel makes a UNIX-like operating system.
- **GNU/Linux** is sometimes abbreviated to **Linux**.







A FEW DISTRIBUTIONS



- A **distribution** comes with
 - custom Linux kernel build,
 - GNU software,
 - additional software,
 - package manager.
- **Debian, Ubuntu, Raspberry Pi OS;**
- **Fedora; Arch; Gentoo;**
- **Alpine:** often used in containers.

OUR CHOICE IS DEBIAN

- Versions

	Packages	Stability
Stable	 older	 stable
Testing		
Unstable	 latest	 not tested

- Current stable:

Version 11 "Bullseye" (2021)

§2: FIRST PRACTICE

- Run our `debian-playground` image in Docker.
- You will see the **Bash** prompt:

```
user@cosmos:~$ █
```

- `user` : user
- `cosmos` : host
- `~` : abbreviation for home directory
- `$` : invitation to type commands



```
user@cosmos:~$ echo 'Hello, World!'
```

```
Hello, World!
```

```
user@cosmos:~$ echo "Hello, $(whoami)!"
```

```
Hello, user!
```

USEFUL COMMANDS

Command	Action
<code>reset</code>	reset console
<code>clear</code>	clear screen
<code>history</code>	command history
<code>exit</code>	exit the shell
<code>logout</code>	log out of the shell

Command history is stored in `.bash_history` in your home dir.

BUILTINS VS. EXTERNAL COMMANDS

- `type <command>` : command type.
- `whereis <name>` : find binary by name.

```
user@cosmos:~$ type cd
cd is a shell builtin
user@cosmos:~$ whereis cd
cd:
```

```
user@cosmos:~$ type --all ls
ls is aliased to `ls --color=auto'
ls is /bin/ls
user@cosmos:~$ whereis ls
ls: /bin/ls /usr/share/man/man1/ls.1.gz
```

USEFUL KEYS (C=Ctrl, M=Alt)

Key	Action
Tab	completion
C-a/e	cursor to line start/end
M-b/f	previous/next word (back/forward)
C-k	delete everything till line end (kill)
C-w	delete previous word
C-y	paste what we cut (yank)
C-r	reverse history search (C-r for more)

MORE KEYS

Key	Action
-----	--------

<code>C-c</code>	sends <code>SIGINT</code> to the running process
------------------	--

<code>C-d</code>	sends end-of-file symbol to console
------------------	-------------------------------------

<code>C-z</code>	stops running process (use <code>fg</code> to resume)
------------------	---

CAT (CONCATENATE)

```
cat file1.txt [file2.txt ...]
```

Will print to standard output files
`file1.txt`, `file2.txt`, etc. concatenated.

SEE SOMETHING WITH CAT

```
user@cosmos:~$ cat /etc/issue  
Debian GNU/Linux 11 \n \l
```

```
user@cosmos:~$ cat /etc/debian_version  
11.4
```

```
user@cosmos:~$ cat /etc/os-release  
PRETTY_NAME="Debian GNU/Linux 11 (bullseye)"  
NAME="Debian GNU/Linux"  
VERSION_ID="11"  
VERSION="11 (bullseye)"  
VERSION_CODENAME=bullseye  
ID=debian  
HOME_URL="https://www.debian.org/"
```

§3: GETTING HELP WITH MAN AND INSTALLING PACKAGES WITH APT

- We have a very basic system at hand.
- Let's install some software.
- `apt` = Advanced Package Tool.
- Specific to Debian-based distributions.
- Other systems come with their own package managers.

BASIC USE

apt	command	Action
	list	list available packages
	list --installed	list installed packages
	update	download list of packages
	upgrade	install package upgrades
	search <pkg>	package lookup
	info <pkg>	package info
	install <pkg>	install a package
	remove <pkg>	remove a package
	autoremove	remove no needed dependencies

SUDO



- Package management requires `root` privileges.
- Use `sudo` command

```
user@cosmos:~$ apt update
Reading package lists... Done
E: Could not open lock file /var/lib/apt/lists/lock - open (13: Perr
E: Unable to lock directory /var/lib/apt/lists/
user@cosmos:~$ sudo apt update
. . . . .
```

INSTALL MAN

We'll install some documentation.

```
user@cosmos:~$ sudo apt update
```

```
. . . . .
```

```
user@cosmos:~$ sudo apt install man-db manpages
```

```
. . . . .
```

--HELP



```
user@cosmos:~$ tar --help
```

```
Usage: tar [OPTION...] [FILE]...
```

```
GNU 'tar' saves many files together into a  
single tape or disk archive, and can restore  
individual files from the archive.
```

```
. . . . .
```

MAN

```
user@cosmos:~$ man tar
```

TAR(1)

GNU TAR Manual

TAR(1)

NAME

tar - an archiving utility

SYNOPSIS

Traditional usage

```
tar {A|c|d|r|t|u|x}[GnSkUW0mpsMBiajJzZhPlRvwo] [ARG...]
```

UNIX-style usage

Command

Action

`whatis cat`

short summary

`man cat`

show manual page

`apropos`

search for `<keyword>` in

`<keyword>`
descriptions

CALENDAR



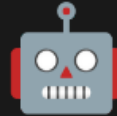
```
user@cosmos:~$ sudo apt install ncal
```

```
user@cosmos:~$ cal
```

August 2022

Su	Mo	Tu	We	Th	Fr	Sa
	1	2	3	4	5	6
7	8	9	10	11	12	13
14	15	16	17	18	19	20
21	22	23	24	25	26	27
28	29	30	31			

UUID



```
user@cosmos:~$ sudo apt install uuid-runtime
```

```
user@cosmos:~$ uuidgen
```

```
050ff192-3dbf-4d32-b64a-e4a3f2bb8e72
```

CURL



```
user@cosmos:~$ sudo apt install curl
```

```
. . . . .
```

```
user@cosmos:~$ curl wttr.in
```

```
Weather report: Guanajuato City, Mexico
```

```
  \  /      Sunny
  .-.-      19 °C
- (  ) -    ↙ 8 km/h
  `-'       10 km
  /  \      0.0 mm
```

POSSIBILITIES OF CURL ★

- Various protocols.
- Inspection of headers (`-i` and `-v`).
- Sending extra headers (`-H`).
- GET, HEAD, POST, PUT, DELETE (`-X`).
- Sending file contents in POST (`-d`).
- Read `man curl`.
- Postman is a proprietary bloat 🤢

WGET



```
user@cosmos:~$ sudo apt install wget lzip
```

```
. . . . .
```

```
user@cosmos:~$ wget https://data.iana.org/time-zones/releases/tzdb-2022a.tar.lz
```

```
. . . . .
```

```
user@cosmos:~$ tar -xvf tzdb-2022a.tar.lz
```

```
. . . . .
```

OLD SOFTWARE

- Don't expect latest software versions in "stable" packages.
- Especially if the Debian release cycle is slower.
- Latest versions are in "unstable".

```
user@cosmos:~$ apt info nodejs
```

```
Package: nodejs
```

```
Version: 12.22.12~dfsg-1~deb11u1
```

```
# 18.x: current Node.js version,
```

```
# 12.x: old, not supported anymore.
```

INSTALLING FROM .DEB FILES



- Packages are stored in `.deb` files.
- Can be installed directly.
- Prefix file name with `./`:

```
$ sudo apt install ./code_1.70.0-1659589288_amd64.deb
```

```
$ apt info code
```

```
Package: code
```

```
Version: 1.70.0-1659589288
```

```
Priority: optional
```

```
Section: devel
```

```
Maintainer: Microsoft Corporation <vscode-linux@microsoft.com>
```

```
Installed-Size: 359 MB
```

```
Provides: visual-studio-code
```

```
. . . . .
```


§4: WORKING WITH FILES

- Everything is a file, e.g. a directory or device.
- Common file systems: **ext4**, **Btrfs**.
- **Inode**: data structure representing contents and metadata.
- **Symbolic links**: pointers to files.
- File names: 255 bytes max, case-sensitive.
- Path separator is `/`.
- `.` is the current directory and `..` is the parent directory.

STAT



```
user@cosmos:~$ stat /etc/shadow
```

```
File: /etc/shadow
```

```
Size: 671          Blocks: 8          IO Block: 4096    regular
```

```
Device: 40h/64d Inode: 10857          Links: 1
```

```
Access: (0640/-rw-r-----)  Uid: (   0/   root)   Gid: (  42/   sha
```

```
Access: 2022-08-05 17:50:36.095309399 +0000
```

```
Modify: 2022-08-05 17:45:16.236327091 +0000
```

```
Change: 2022-08-05 17:45:16.247327091 +0000
```

```
Birth: 2022-08-05 17:45:16.236327091 +0000
```

```

size in bytes
device ID
(hex / dec) 512-byte blocks
allocated IO block size
in bytes file type
$ stat /etc/shadow
File: /etc/shadow
Size: 671 Blocks: 8 IO Block: 4096 regular file
Device: 40h/64d Inode: 10857 Links: 1 owner group ID group
Access: (0640/-rw-r-----) Uid: (0/ root) Gid: (42/ shadow)
Access: 2022-08-05 17:50:36.095309399 +0000 last file access/read
Modify: 2022-08-05 17:45:16.236327091 +0000 last modification of contents
Change: 2022-08-05 17:45:16.247327091 +0000 last modification of inode (metadata)
Birth: 2022-08-05 17:45:16.236327091 +0000 file creation
access rights inode number
(oct / human)

```

```
user@cosmos:~$ stat --format=%F /etc/passwd
regular file
user@cosmos:~$ stat --format=%F /home
directory
user@cosmos:~$ stat --format=%F /proc/1/exe
symbolic link
user@cosmos:~$ stat --format=%F /dev/tty
character special file
```

As you see,

`/etc/passwd`, `/home`, `/proc/1/exe`, `/dev/tty`

are all files, but of different kind.

Command Action

<code>pwd</code>	print working directory
<code>ls</code>	list files in the working directory
<code>tree</code>	print file tree (package <code>tree</code>)
<code>ls foo</code>	list files in <code>foo</code>
<code>cd foo</code>	go to <code>foo</code> directory
<code>cd</code>	go to your home directory

SOME OPTIONS OF LS

- `-1` : one file per line,
- `-a`, `--all` : show hidden files,
- `-h`, `--human-readable` : human-readable size,
- `-l` : long listing format,
- `-p` : append `/` to directories,
- `-R`, `--recursive` : recursive list,
- `-s` : sort by size, largest first,
- `-t` : sort by time, newest first,
- `-r`, `--reverse` : reverse sort.

```
user@cosmos:~$ ls -1ahlp /etc
```

HIDDEN FILES 🙈

- By convention, files starting with **.** are hidden.

```
user@cosmos:~$ ls -l
user@cosmos:~$ ls -la
.
..
.bash_history
.bash_logout
.bashrc
.profile
```


HIDDEN FILES 🙈

- Used to store local settings and configuration.
- Bash will have these in your home dir: `.bash_history`, `.bash_logout`, `.bashrc`, `.profile` (second session).
- Familiar example: Git stores its local data in `.git`.
- Your IDE configuration is in `/home/user/.vscode`, `/home/user/.idea`, etc.

Command

Action

```
mkdir foo
```

create directory `foo`

```
mkdir -p foo/bar/baz
```

create together with parents

```
mv foo bar
```

move `foo` to `bar`

```
cp foo bar
```

copy `foo` to `bar`

```
cp -r ./foo/ bar
```

copy directories recursively

```
rm foo.txt
```

delete `foo.txt`

```
rm -r ./foo/
```

delete `foo` recursively

```
ln -s source target
```

create symbolic link

```
touch foo.txt
```

modify / create file

SYMBOLIC LINKS

- Hard links share inode; point to a specific part of the file system.
- Soft (symbolic) links are separate inodes referring to another path.
- Example use: if you install Java on Debian (package `default-jdk`), you'll have symbolic links

```
/usr/bin/java → /etc/alternatives/java →  
/usr/lib/jvm/java-11-openjdk-amd64/bin/java
```

DANGLING LINKS



```
user@cosmos:~$ touch foo.txt
user@cosmos:~$ ln -s foo.txt bar.txt
user@cosmos:~$ ls -ll
total 4
lrwxrwxrwx 1 user user 7 Aug  8 20:34 bar.txt -> foo.txt
-rw-r--r-- 1 user user 0 Aug  8 20:34 foo.txt
user@cosmos:~$ rm foo.txt
user@cosmos:~$ ls -ll
total 4
lrwxrwxrwx 1 user user 7 Aug  8 20:34 bar.txt -> foo.txt
user@cosmos:~$ cat bar.txt
```

VIEWING FILES 🐼

Command	Action
<code>cat file ...</code>	concatenate files
<code>less foo.txt</code>	view interactively
<code>head foo.txt</code>	view first lines
<code>tail foo.txt</code>	view last lines
<code>tail -f log.txt</code>	watch for updates

* `sudo apt install less`

FINDING FILES

- `find` : finds files in a directory.
- Read `man find`.
- Finding file by name:

```
user@cosmos:~$ find /usr -name *.txt
/usr/share/doc/libdb5.3/build_signature_amd64.txt
/usr/share/doc/mount/mount.txt
/usr/share/doc/util-linux/00-about-docs.txt
/usr/share/doc/util-linux/PAM-configuration.txt
/usr/share/doc/util-linux/blkid.txt
. . . . .
```

GREPPING THROUGH CONTENTS



- Read `man grep`.
- `-r`, `--recursive` for recursive search,
- `-i`, `--ignore-case` to ignore case.

```
user@cosmos:~$ grep -r -i torvalds /usr
/usr/share/doc/bsdutils/copyright:          1991, 1992 Linus Torvalds
/usr/share/doc/bsdutils/copyright:Copyright: 1991, 1992 Linus Torvalds
/usr/share/doc/libblkid1/copyright:          1991, 1992 Linus Torvalds
/usr/share/doc/libblkid1/copyright:Copyright: 1991, 1992 Linus Torvalds
/usr/share/doc/libmount1/copyright:          1991, 1992 Linus Torvalds
```

DU: DISK USAGE

- Directory size reported by `ls` and `stat`: possibly not what you want.
- Actual recursive size of contents: `du` (disk usage).
 - `-h`, `--human-readable`,
 - `-s`, `--summarize` : display total size.

```
user@cosmos:~$ stat --format=%s /usr
```

```
84
```

```
user@cosmos:~$ du -sh /usr
```

```
136M    /usr
```


DF: DISK SPACE



```
user@cosmos:~$ df -h
```

Filesystem	Size	Used	Avail	Use%	Mounted on
/dev/vdb	50G	46G	4.2G	92%	/
tmpfs	64M	0	64M	0%	/dev
tmpfs	3.3G	0	3.3G	0%	/sys/fs/cgroup
shm	64M	0	64M	0%	/dev/shm
/dev/vdb	50G	46G	4.2G	92%	/etc/hosts
devtmpfs	3.3G	0	3.3G	0%	/dev/tty
tmpfs	3.3G	0	3.3G	0%	/proc/asound
tmpfs	3.3G	0	3.3G	0%	/proc/acpi
tmpfs	3.3G	0	3.3G	0%	/proc/scsi

WORKING WITH ARCHIVES



Command

Action

```
zip -r files.zip files
```

create `.zip` archive

```
unzip files.zip -d  
/path/to/unzip
```

extract `.zip` archive

```
unzip -l files.zip
```

list contents

```
tar -czvf files.tar.gz files
```

create `.tar.gz`
archive

```
tar -xvf files.tar.gz
```

extract archive

```
tar -tvf files.tar.gz
```

list contents

WHAT IS `.tar.gz`?



- An **archive** packages files together.

`.tar` = "tape archive" format.

- **Compression** is applied to an archive.

`.gz`, `.bz2`, `.lz`, `.xz`, ... = different compression algorithms.

- `tar` command is already aware of those formats applied to `.tar`.

OTHER USEFUL COMMANDS



Command	Action
<code>sort foo.txt</code>	sort lines and print to stdout
<code>sort -u foo.txt</code>	sort lines, print unique entries
<code>diff foo.txt bar.txt</code>	compare files
<code>sed s/foo/bar/g text.txt</code>	pattern substitution
<code>wc</code>	count words, lines, bytes

Count installed packages:

```
user@cosmos:~$ apt list --installed | wc -l
```

UNIX Power Tools

2nd Edition



sed & awk

O'REILLY

Dale Dougherty & Arnold Robbins

NEXT TIME

- Users, groups, permissions.
- Processes (`ps`, `top`, `kill`, `killall`).
- Linux file system: what is `/bin`, `/dev`, `/boot`, `/etc`, and so on.
- IO redirection and pipelines.
- Pattern matching, aliases, variables, `$PATH`.
- Anatomy of scripts: shebang.