Introduction to Cyber Security

Brandenburg
University of Technology
Cottbus - Senftenberg

2024-10-24

Introduction to Linux

Prof. Dr.-Ing. Andriy Panchenko

Fabian Mildner

Chair of IT Security
BTU Cottbus-Senftenberg

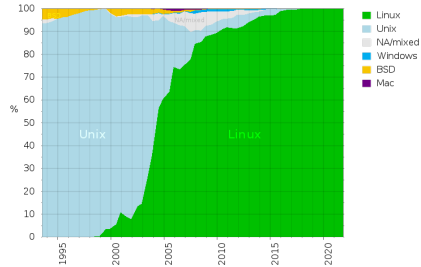
1. Why Linux?
2. The Way from Unix to GNU/Linux
3. Understanding GNU/Linux
4. Important Unix Commands and Tools
5. Some Practical Exercises

Why Linux?

- Real world importance
- Openness
- Free of charge

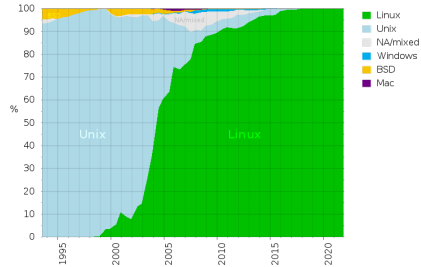
Why Linux?

- **Real world importance**:
  - Smartphones: $\sim 74\ \%$
  - Servers: probably significantly more than $50\ \%$
  - Supercomputers: $100\ \%$ (top 500)
  - PC only $\sim 5\ \%$
- **Openness**



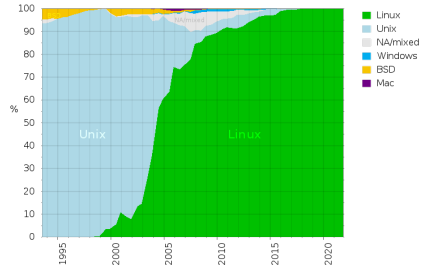
- **Free of charge**

Why Linux?



- **Real world importance**:
  - Smartphones: $\sim 74\ \%$
  - Servers: probably significantly more than $50\ \%$
  - Supercomputers: $100\ \%$ (top 500)
  - PC only $\sim 5\ \%$
- **Openness**:
  - *Examine, understand and verify* (if necessary or beneficial) what is going on
  - *Control* what your device is doing *yourself* (as opposed to the company providing the OS)
  - *Trust* is distributed over many people with different goals involved in creating Linux (as opposed to a single company with a single goal)
- **Free of charge**

Why Linux?



- **Real world importance**:
  - Smartphones: $\sim 74\ \%$
  - Servers: probably significantly more than $50\ \%$
  - Supercomputers: $100\ \%$ (top 500)
  - PC only $\sim 5\ \%$
- **Openness**:
  - *Examine, understand and verify* (if necessary or beneficial) what is going on
  - *Control* what your device is doing *yourself* (as opposed to the company providing the OS)
  - *Trust* is distributed over many people with different goals involved in creating Linux (as opposed to a single company with a single goal)
- **Free of charge**: available to everyone

UNIX

History

1969, Bell Laboratories: development of Unix to support software developer

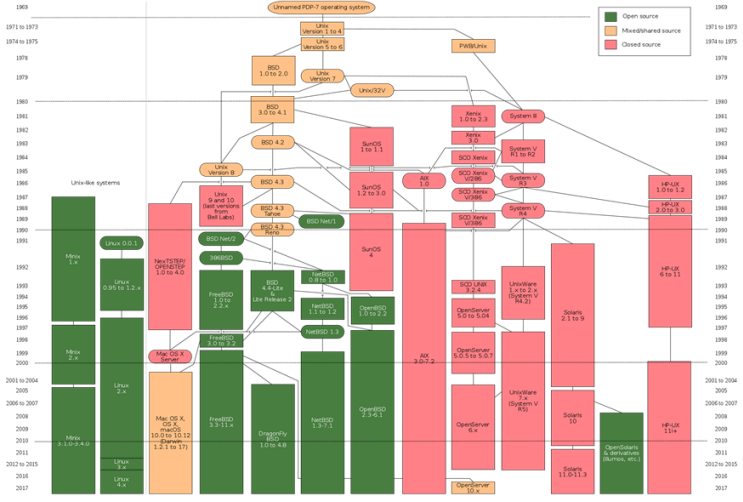
What is Unix?

Today, Unix is a term to denote any operating system which either is an descendant of UNIX or implements it concepts.

Properties

- Multi-user system
- Multi-tasking capabilities
- Multithreading
- Memory protection / virtual memory

Ancestral Chart of UNIX



From UNIX to Linux

The GNU Project (GNU: GNU is not Unix)

1983: Richard Stallman started the GNU project to develop a free equivalent of the Unix operation system

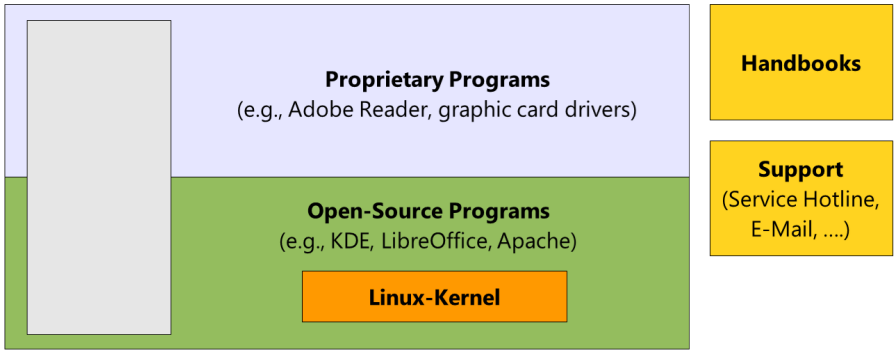
Development of Linux

- 1991: Linus Torvalds developed the OS kernel Linux[1] (open source)
- 1992: Kernel was distributed under the GNU General Public License (GPL)
- Linux[2]: similar OS to UNIX, based on Linux kernel and GNU software
- Today, Linux is the most widely used open source version of Unix
- Discussion: “Linux” or “GNU/Linux”

[1] “Linux” in the more restricted sense
[2] “Linux” in the broader sense, also called “distribution”.

Linux Distributions

Content of a distribution:



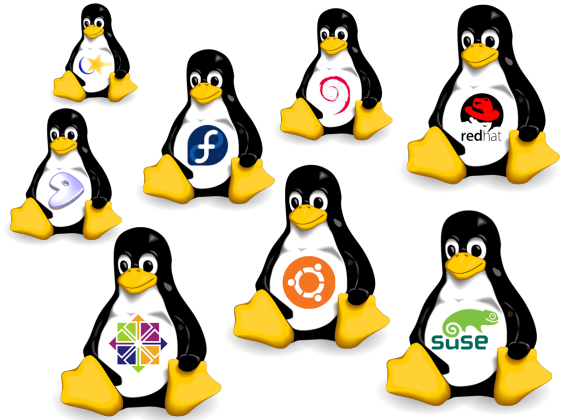
Linux Distributions - Examples

Examples:

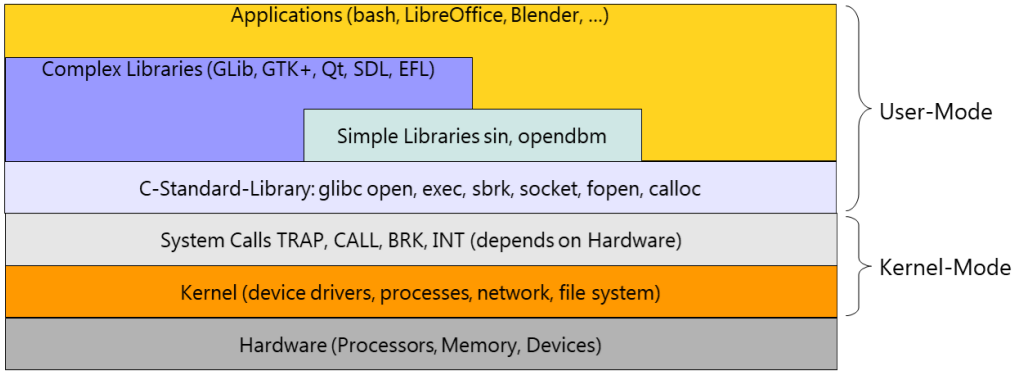
Debian, Ubuntu, Red Hat (RHEL), Fedora, Gentoo, Android, . . .

Also many special-purpose Distributions. E.g. for Security:

TAILS, Qubes OS, Parrot Security, Kali Linux, Black Arch Linux, . . .

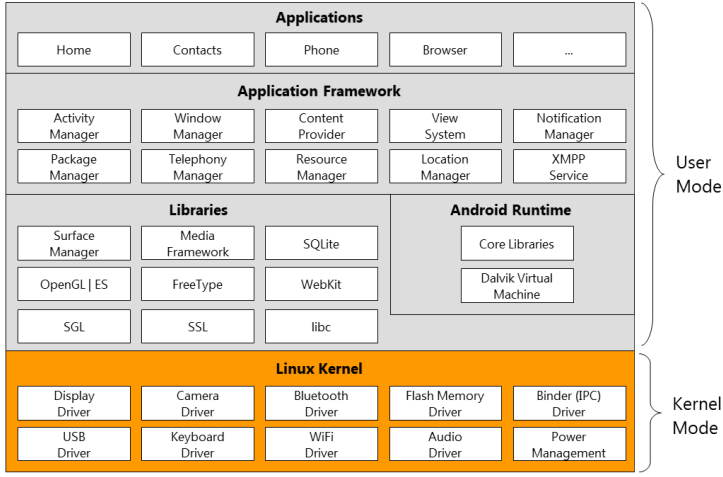


Linux Architecture

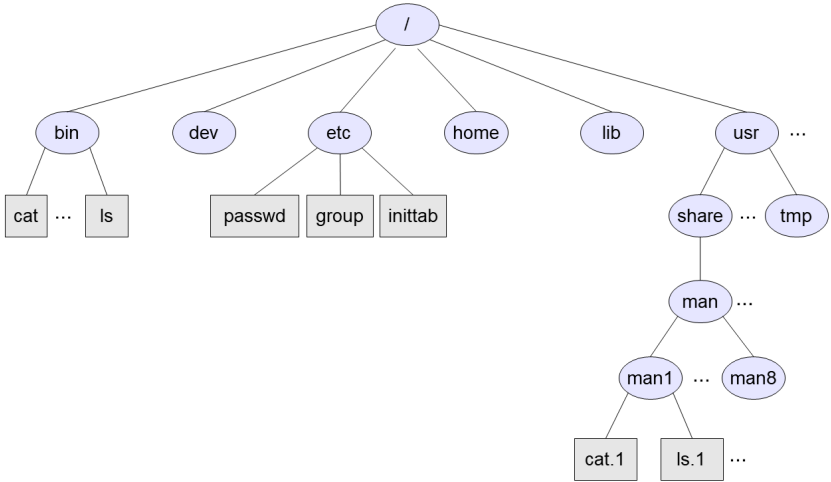


- System calls as interface between user mode and kernel mode

Android – An Example

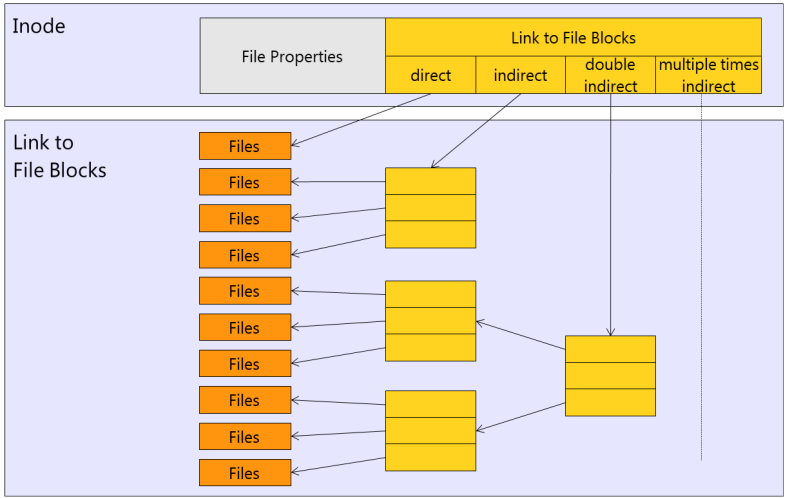


Structure of the File System



Structure of the File System

File System (ext4): Inode



GNU/Linux File Types

Regular file

text and binary files like programs, scripts, configuration files, . . .

Directory

contains inode numbers of any files in the directory

Device file

- interface to hardware;
- distinction between block based (buffered) and character devices (non buffered)

File System: Access Control

User Domains

- **User:** usually the creator of the file
- **Group:** all users in the same group
- **Others:** all remaining users

File Operations

- read (r)
- write (w)
- execute (e)

→ some practical examples

File System: Access Control

File access permissions in the case of directories:

dir permissions	Octal	del rename create files	dir list	read file contents	write file contents	cd dir	cd subdir	subdir list	access subdir files
---	0								
-W-	2								
R--	4		only file names (*)						
RW-	6		only file names (*)						
--X	1			X	X	X	X	X	X
-WX	3	X		X	X	X	X	X	X
R-X	5		X	X	X	X	X	X	X
RWX	7	X	X	X	X	X	X	X	X

https://unix.stackexchange.com/questions/21251/execute-vs-read-bit-how-do-directory-permissions-in-linux-work

Working with Files

Working with Files

Open

- Open file by absolute or relative path
- Check file access for execution
- Return file descriptor on success

Working with Files

Edit

- Reference file with its file descriptor
- Read or write file

Working with Files

Close

- Release file descriptor

The Shell

Command Line Interpreter

- Started by login service after successful authentication of user
- Interprets and executes user commands with the access rights of the caller
- Provides:
  - script language for automation
  - wild cards (e.g., $*$)
  - environment variables (e.g., $HOME)
  - input/output piping
  - command history

Important Bash Commands

File Commands

ls – directory listing
ls -al – formatted listing with hidden files
cd *dir* - change directory to *dir*
cd – change to home
pwd – show current directory
mkdir *dir* – create a directory *dir*
rm *file* – delete *file*
rm -r *dir* – delete directory *dir*
rm -f *file* – force remove *file*
rm -rf *dir* – force remove directory *dir* *
cp *file1 file2* – copy *file1* to *file2*
cp -r *dir1 dir2* – copy *dir1* to *dir2*; create *dir2* if it doesn't exist
mv *file1 file2* – rename or move *file1* to *file2*
if *file2* is an existing directory, moves *file1* into directory *file2*
ln -s *file link* – create symbolic link *link* to *file*
touch *file* – create or update *file*
cat > *file* – places standard input into *file*
more *file* – output the contents of *file*
head *file* – output the first 10 lines of *file*
tail *file* – output the last 10 lines of *file*
tail -f *file* – output the contents of *file* as it

File Permissions

chmod *octal file* – change the permissions of *file* to *octal*, which can be found separately for user, group, and world by adding:

- 4 – read (r)
- 2 – write (w)
- 1 – execute (x)

Examples:
chmod 777 – read, write, execute for all
chmod 755 – rwx for owner, rx for group and world
For more options, see **man chmod**.

SSH

ssh *user@host* – connect to *host* as *user*
ssh -p *port user@host* – connect to *host* on port *port* as *user*
ssh-copy-id *user@host* – add your key to *host* for *user* to enable a keyed or passwordless login

Searching

grep *pattern files* – search for *pattern* in *files*
grep -r *pattern dir* – search recursively for *pattern* in *dir*
command* | grep *pattern – search for *pattern* in the output of *command*
locate *file* – find all instances of *file*

Additional References

- https://missing.csail.mit.edu/
- Linux Command line Reference
  https://ss64.com/bash/
- Linux Shell Scripting Tutorial: A Beginners Handbook
  http://www.freeos.com/guides/lsst/
- Linux Services: A list of UNIX and GNU/Linux services
  http://www.linux-services.org/shell/
- Galileo Computing: Shell Programming
  http://openbook.galileocomputing.de/shell_programmierung/

Linux Challenge

Basic Tasks

- Create a file `/linux-fun/test.dat` with the contents “I like GNU/Linux. Do you like it as well?”
- Use “cat” to show the contents of the file.
- Make two copies of the file. Rename one.
- Write the string “Yes!” to the file, one time using “>” and one time “>>”. What is the difference?
- Count the number of occurrences of the word “like” in “test.dat”
- Remove the file and folder you just created.

Learning Linux with Challenges

- https://overthewire.org/wargames/bandit/
- https://cmdchallenge.com/
- Linux Challenge based on Bochs:
  https://www.b-tu.de/owncloud/s/JJdzk59ennyoeyc

Thank you for your attention.