

Introduction

Lecture 1

Welcome

to the *Internal Strcuture of Operating* class

You will learn, understand and experiment

- how operating systems work
- how basic POSIX (Linux) commands work
- how to use the Linux **command line**
- the history of Linux
- how use the development tools

We expect

- to come to class
- ask a lot of questions

Our team

Lectures

- Alexandru Radovici (CD)
- Ioana Culic (AC)

Labs

- Ana Gabriela Almăjanu (CD)
- Irina Bradu (AC)
- Andrei Dragtă (AC)
- Andreea Ocănoaia (CD)
- Irina Niță (CD)
- Cristian Paris (CD)
- Alexandra Văduva (CD)
- Alexandru Radovici (CD)

Outline

Lectures

- 12 lectures

Labs

- 12 labs

Homework

- 4 homework assignments

that will help you prepare for the hands on tests

Grading

Part	Description	Points
<u>Lecture tests</u>	You will have a test at every class with subjects from the previous class.	0.5p
Homework	You will have 4 homework assignments to exercise you for the hands-on tests	1p
<u>Lab activity</u>	Your lab activity will be graded.	0.5p
Hands-on midterm	You will have a practical test covering the first 9 weeks (a Saturday in December)	4p
Theoretical exam	You will have a quiz on Moodle with French grading (in the exams period).	2.5p
Hands-on exam	You will have a practical test covering all the subjects (in the exams period).	2.5p
Total	<i>You will need at least 2.5p during the semester and 5 points in total to pass</i>	11p

Notation

- *name* - this is a name
- **important** - this is important
- `command` - this is a command or a part of a source code

Code

```
1  $ command arguments
2  this is what the command wrote
3  $ command2 arguments
```

\$ signifies that the user can write a command, but is **not part of the command**.

*If you can't explain it simply, you
don't understand it well enough*

Albert Einstein

Bibliography

1. **Brian Ward**, *How LINUX Works*, 3rd Edition, No Starch Press, 2021
2. **Razvan Deaconescu, Razbvan Rughinis, Mihai Carabas, Alexandru Radovici**, *Utilizarea Sistemelor de Operare*, Printech 2021, [download](#)

Operating System

the purpose of an OS

Operating System

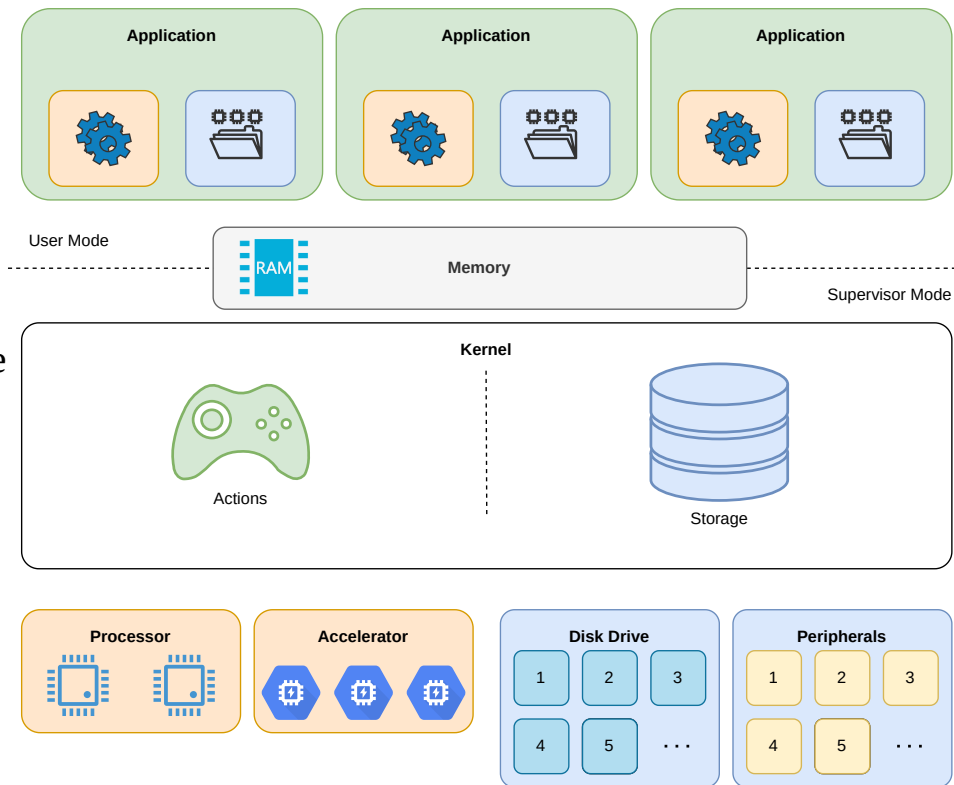
the main role

Allow Portability

- provides a hardware independent API
- applications should run on any hardware

Resources Management and Isolation

- allow applications to access resources
- prevent applications from accessing hardware directly
- isolate applications



Desktop and Server Operating Systems

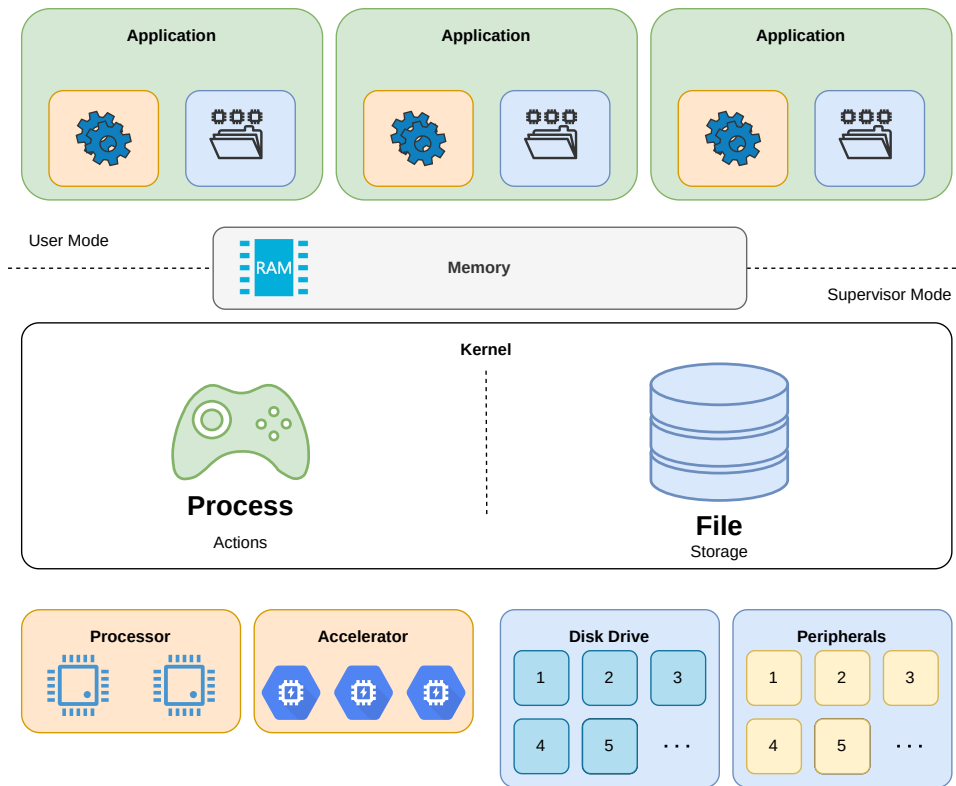
abstractions

Actions

- *Applications*
- use the *Processor* and *Accelerators* (GPU, Neural Engine, etc)

Data

- everything is a file
- peripherals are viewed as files (*POSIX*)
 - `/dev/input/keyboard` - keyboard
 - `/dev/fb` - screen (framebuffer)
 - `/dev/sda` - Disk Drive A (first)



Embedded Operating Systems

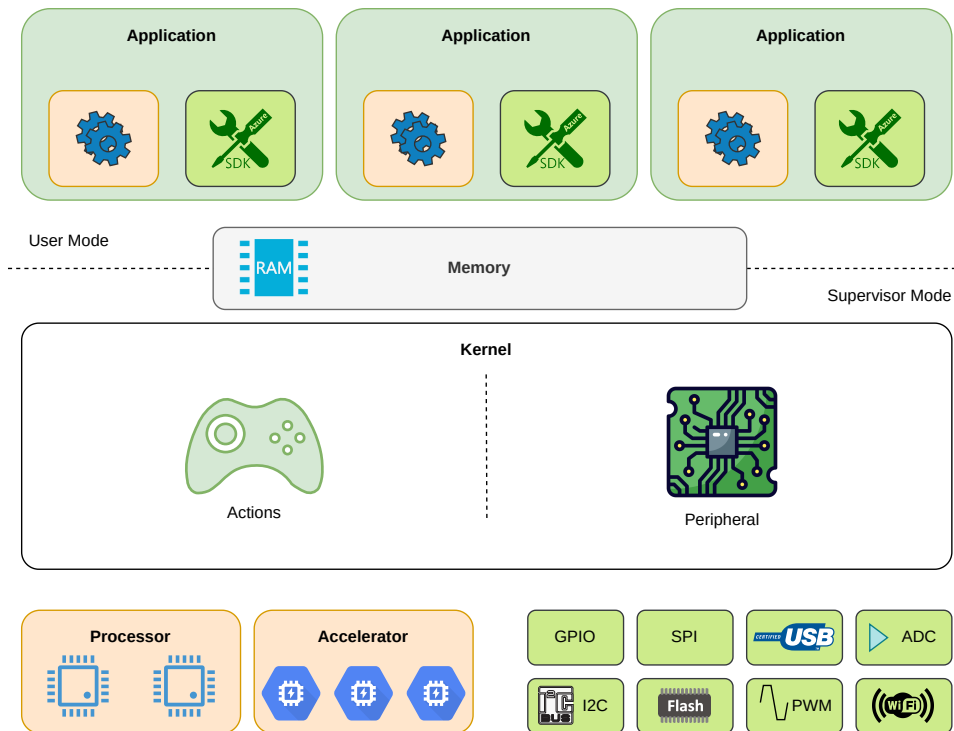
Actions

- Simple *applications*
- use the *Processor* and *Accelerators*
(Crypto Engines, Neural Engine, etc)

Peripheral

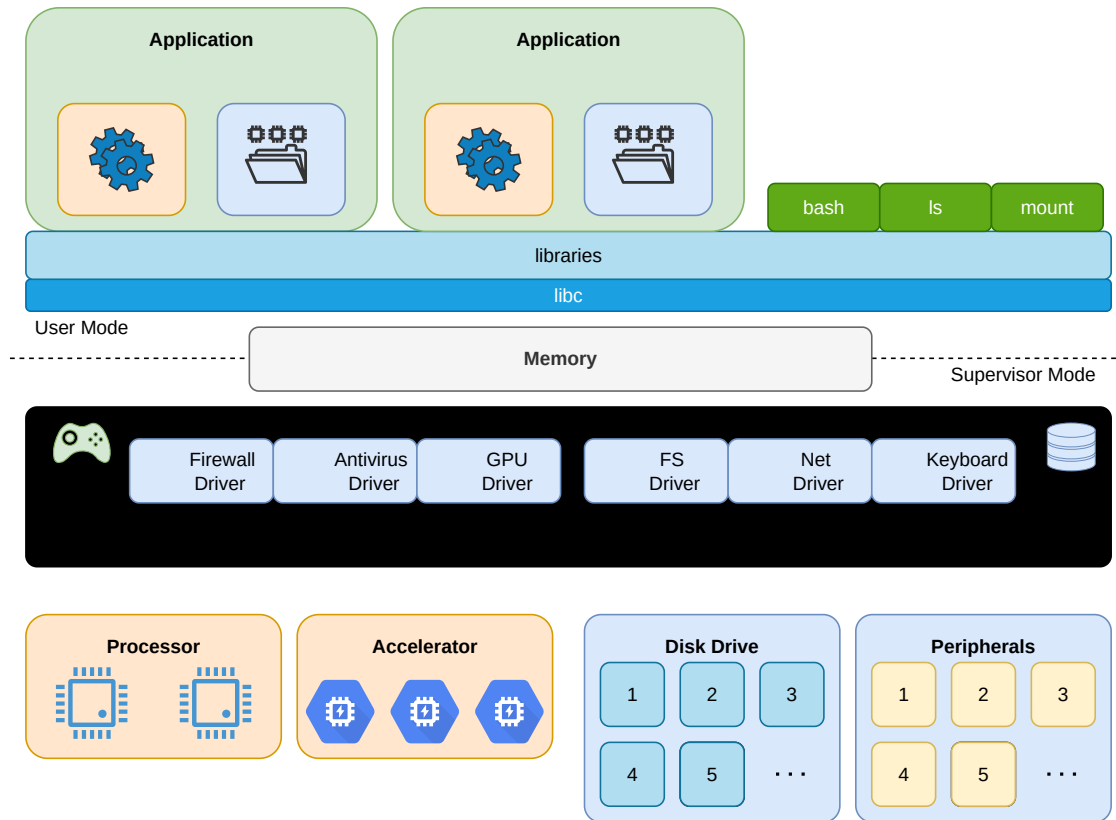
- provide a hardware independent API
- prevent processes from accessing the peripheral

usually the applications and the kernel are compiled together into a **single binary**



The OS Stack

kernel / libc / libraries / basic tools (commands) / applications



Application Examples

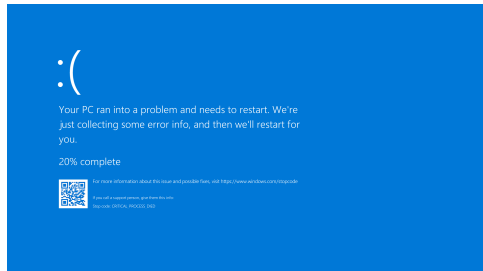
What it does	Windows	Linux	macOS
Desktop	<code>explorer.exe</code>	<code>nautilus</code>	<code>Finder</code>
Applications	<code>taskbar.exe</code>	<code>gnome-shell</code>	<code>Dock</code>
Files	<code>explorer.exe</code>	<code>nautilus</code>	<code>Finder</code>
Settings	<code>start ms-settings:</code>	<code>gnome-control-center</code>	<code>System Settings</code>
Commands	<code>cmd.exe</code> or <code>powershell.exe</code>	<code>bash</code> or <code>zsh</code>	<code>bash</code> or <code>zsh</code>
Documents	<code>powerpoint.exe</code>	<code>libreoffice</code>	<code>Pages</code>
Edit Code	<code>code.exe</code>	<code>code</code>	<code>code</code>

Where can we see the OS

During Boot

```
Booting Linux on physical CPU 0x0
Linux version 4.1.0-rc1-00016-g5704846-dirty (age@truchilidae) (gcc version 4.9.3 20141031 [prerelease] (Linaro GCC 4.9-2014.11) ) #68 Sat May 16 10:51:07 CEST 2015
CPU: ARMv7-M [410fc241] revision 1 (ARMv7M), cr=00000000
CPU: unknown data cache, unknown instruction cache
Machine model: VF610 Cortex-M4
Built 1 zonelists in Zone order, mobility grouping on. Total pages: 12192
Kernel command line: console=ttyLP2,115200 clk_ignore_unused init=/linuxrc rw
PID hash table entries: 256 (order: -2, 1024 bytes)
Dentry cache hash table entries: 8192 (order: 3, 32768 bytes)
Inode-cache hash table entries: 4096 (order: 2, 16384 bytes)
Memory: 46880K/49152K available (895K kernel code, 52K rdata, 312K rodata, 64K init, 156K bss, 2952K reserved, 0K cma-reserved)
Virtual kernel memory layout:
 vector : 0x00000000 - 0x00001000 ( 4 kB)
 fixmap : 0xffff0000 - 0xffff0000 (3872 kB)
 vmalloc : 0x00000000 - 0xffffffffff (4095 MB)
 lowmem  : 0x00000000 - 0x8f000000 ( 48 MB)
 .text : 0x0f000000 - 0x0f12e050 (1208 kB)
 .init : 0x0c000000 - 0x0c000000 ( 12 kB)
 .data : 0x0c000000 - 0x0c010000 ( 65 kB)
 .bss : 0x0c010000 - 0x0c041c30 ( 166 kB)
```

Blue Screen (BSoD)



Kernel Panic

```
Halting for driver initialization.
Scanning and configuring dmraid supported devices
Reading all physical volumes. This may take a while...
Found volume group "VolGroup00" using metadata type lvm2
Activating logical volumes
  2 logical volume(s) in volume group "VolGroup00" now active
Trying to resume from /dev/VolGroup00/LogVol01
No suspend signature on swap, not resuming.
Creating root device.
Mounting root filesystem.
x_journald starting. Commit interval 5 seconds
EXT3-fs: mounted filesystem with ordered data mode.
Setting up other filesystems.
Setting up new root fs
no fsck.sys, mounting internal defaults
Switching to new root and running init.
Unmounting old /dev
Unmounting old /proc
Unmounting old /sys
type=1404 audit(1301044547.109:2): enforcing=1 old_enforcing=0 auid=4294967295 s
es=4294967295
Unable to load SELinux Policy. Machine is in enforcing mode. Halting now.
Kernel panic - not syncing: Attempted to kill init!

```

You need to restart your computer. Hold down the Power button for several seconds or press the Restart button.

Veillez redémarrer votre ordinateur. Maintenez la touche de démarrage enfoncée pendant plusieurs secondes ou bien appuyez sur le bouton de réinitialisation.

Sie müssen Ihren Computer neu starten. Halten Sie dazu die Einschalttaste einige Sekunden gedrückt oder drücken Sie die Neustart-Taste.

コンピュータを再起動する必要があります。パワーボタンを数秒間押し続けるか、リセットボタンを押してください。

The Kernel

- is a collection of applications that run in privileged mode
- one *main application* and several *utilities apps*

```
htop - htop

0% 1.1% 4% 1.4% 6% 5.7% 12% 1.0%
1% 1.0% 5% 1.3% 6% 1.0% 12% 0.8%
2% 4.7% 6% 9.5% 10% 1.3% 14% 2.1%
3% 1.6% 7% 18.0% 11% 0.5% 15% 0.5%
Mem[|||||] 2.8% 60.4% Tasks: 154, 900 thr, 324 kthr, 1 running
Swap[|||||] 0K/0.00% Load average: 0.43 0.20 0.09
Uptime: 00:05:24

Main PID PPID NI VIRT RES SHR S CPU% MEM% TIME+ Command
PID ASPP
4257 alexandru 35 15 229M 7212 7212 0 0.0 0.0 0:00.02 /usr/bin/zsh
5208 alexandru 20 0 229M 7396 4652 0 4.2 0.0 0:10.65 htop
3910 alexandru 20 0 1056 704 652 0 0.0 0.0 0:00.00 catatonit -P
5321 alexandru 20 0 551M 34564 20192 0 0.0 0.1 0:00.11 /usr/libexec/gnome-control-center-search-provider
5432 alexandru 20 0 20724 7792 2360 5 0.0 0.0 0:00.00 /usr/lib64/firefox/crashhelper 5410 9 /tmp/ 10 12
6112 alexandru 20 0 4838M 211M 179M 5 2.6 0.3 0:00.67 /usr/bin/loope --gaplication-service
3241 ssd 20 0 241M 8840 7488 0 0.0 0.0 0:00.04 /usr/libexec/sss/sss4.xcm --logger-files
3268 root 20 0 540M 14896 17680 5 0.0 0.0 0:00.03 /usr/sbin/abrt-dbus -t133
3570 root 20 0 577M 44560 10888 0 0.0 0.1 0:00.58 /usr/libexec/fwupd/fwupd

1 root 20 0 0 0 0 0.0 0.0 0:00.00 kthreadd
3 root 20 0 0 0 0 0.0 0.0 0:00.00 pool_workqueue_release
4 root 0 -20 0 0 0 0 0.0 0.0 0:00.00 kworker/R-rcu_gp
5 root 0 -20 0 0 0 0 0.0 0.0 0:00.00 kworker/R-sysrq_wq
6 root 0 -20 0 0 0 0 0.0 0.0 0:00.00 kworker/R-kvfree_rcu_reclaim
7 root 0 -20 0 0 0 0 0.0 0.0 0:00.00 kworker/R-slub_flushwq
8 root 0 -20 0 0 0 0 0.0 0.0 0:00.00 kworker/R-netns
9 root 20 0 0 0 0 0 0.0 0.0 0:00.00 kworker/0-0-rcu_gp
10 root 0 -20 0 0 0 0 0.0 0.0 0:00.00 kworker/0-WH-events_highpri
11 root 20 0 0 0 0 0 0.0 0.0 0:00.00 kworker/0-1-rcu_gp
12 root 20 0 0 0 0 0 0.0 0.0 0:00.42 kworker/u64-0-kcryptd-252:0-1
13 root 0 -20 0 0 0 0 0.0 0.0 0:00.00 kworker/R-mm_percpu_wq
14 root 20 0 0 0 0 0 0.5 0.0 0:00.57 kworker/u64-1-events_unbound
15 root 20 0 0 0 0 0 0.5 0.0 0:00.00 ksoftirqd/0
16 root 20 0 0 0 0 0 0.0 0.0 0:00.34 rcu_preempt
17 root 20 0 0 0 0 0 0.0 0.0 0:00.00 rcu_exp_path_gp_kthread_worker/0
18 root 20 0 0 0 0 0 0.0 0.0 0:00.00 rcu_exp_gp_kthread_worker
19 root RT 0 0 0 0 0 0.0 0.0 0:00.03 migration/0
F1 help F2 setup F3 search F4 filter F5 list F6 sortby F7 nice F8 nice F9 kill F10 quit
```



htop

Virtual Machine

run an OS within an application

The Idea

run an operating system within an application

Why?

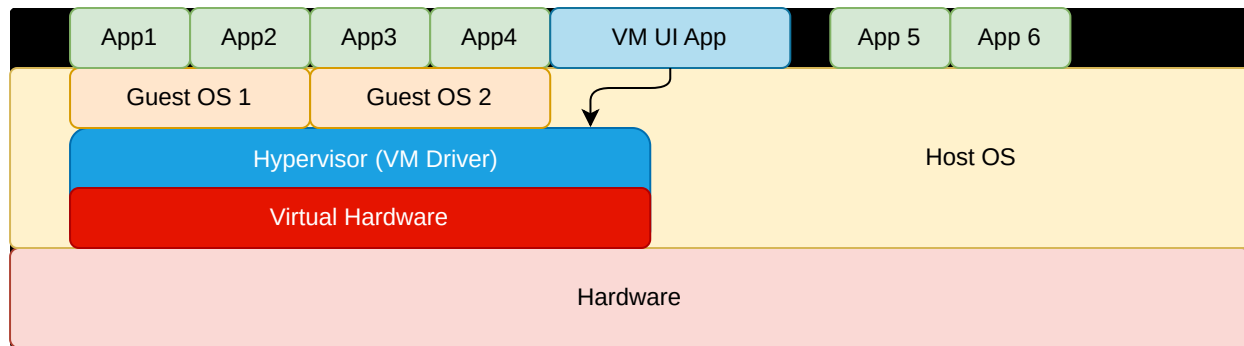
- Debug operating systems
- Run software that is not compatible with the OS your computer runs
- Securely share a server

Virtualization Software

- VirtualBox
- VMWare
- QEMU

Simulation / Emulation

simulate all the hardware - very slow

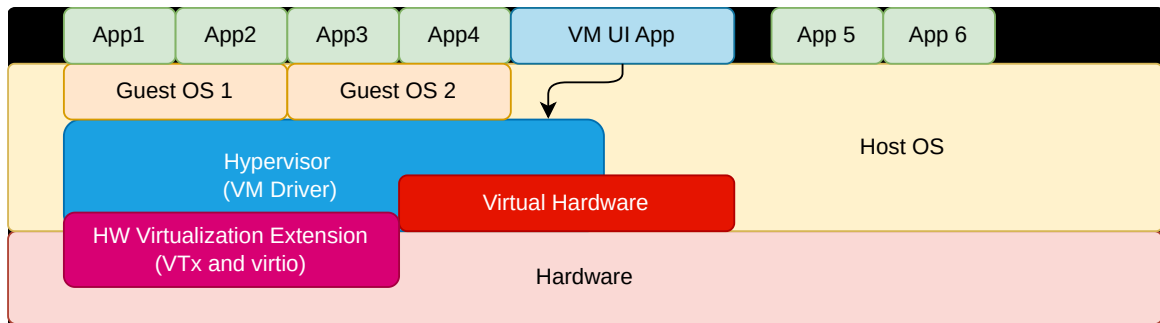


QEMU

- is able to simulate most of the available architectures
 - most used `x86` , `AMD64` , `arm` , `aarch64` , `powerpc` , `risc-v`

Virtualization

use a part of the available hardware, emulates as if the system was running alone on the hardware



- Requires VT-x (Intel),... (AMD) and ... (ARM) enabled
- The guest OS has to be built for the same CPU architecture as the host OS
 - *Apple Silicon* is ARM64 (`aarch64`)

Make sure you download a Linux version for `arm64` or `aarch64`

Hypervisors

VMWare



VirtualBox



QEMU

via `hyper-v` , `KVM` or `TODO`
macOS

Virtual Box

A Virtual Box Machine Settings Example

Distribution

what poeple think an OS is

UNIX

- allows several users
 - connect to the mainframe using several terminals
 - run several programs in paralel
- is open source up to version *UNIX System III*

Ken Thompson & Dennis Ritchie (Bell Labs, AT&T)



- mainframes provide a lot of hardware
 - needs to be shared by several users
 - each user has a terminal connected to the mainframe
- writes UNIX

The POSIX Standard

Portable Operating System Interface

Defines:

- how an OS should work (*everything is a file*)
- how the C API looks like (ex `stdio.h` ,
`unistd.h`)
- what commands (basic software) should be available

Allows:

- applications to be portable
 - *in C/C++ source code format only ->*

When UNIX became closed source ->

UC Berkely Software Distribution (BSD)

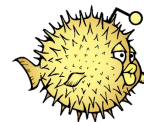
FreeBSD



Darwin



OpenBSD



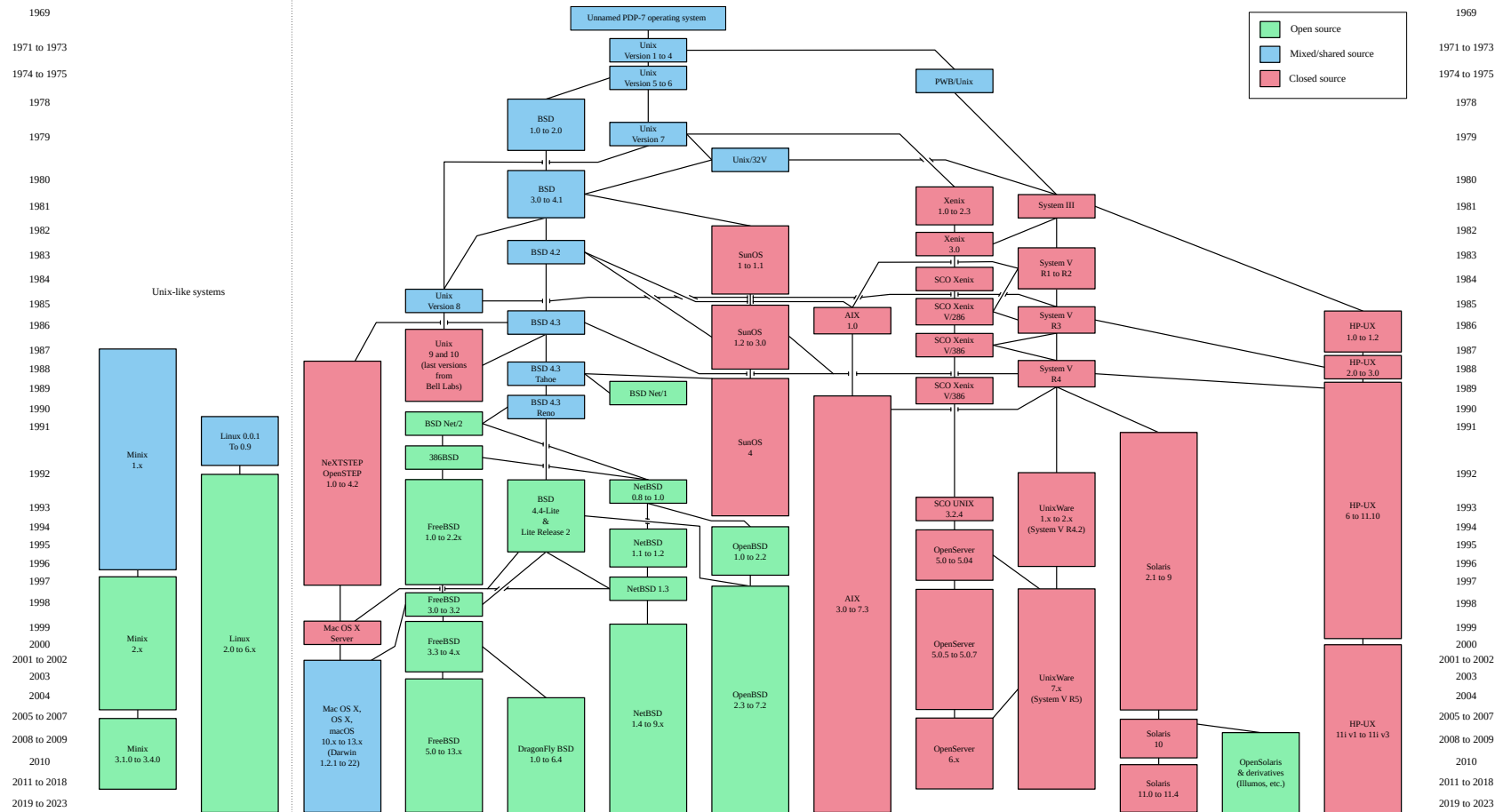
NetBSD



New kernels

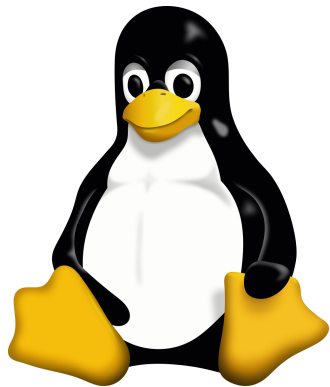
Minix

Linux



The Linux Kernel

- is the actual operating system
- invisible to the user
 - except when it boots 🏁 and panics 🤖
- 25 mil line of code (LOC)
- manages all the system resources



Meet *Tux*

Linus Torvalds (University of Helsinki)



- did not agree with A Tannenbaum about how Minix should work
- wrote this own 🧑🏻💻 *POSIX compliant* OS, called it *Linux*

GNU Software

Basic *UNIX* commands and software **rewritten**

- startup software (`init`)
- most of what we call today coreutils
- C Compiler -> GNU Compiler Collection (`gcc`)
 - C and C++ standard libraries
 - `flex` / `yacc`
- X Windows System (`x11`)

Modern Software

- *coreutils* are being rewritten in Rust (*utils*)
- `gcc` is being slowly replaced by LLVM
- `x11` is being replaced by Wayland

Richard Stallman (MIT)

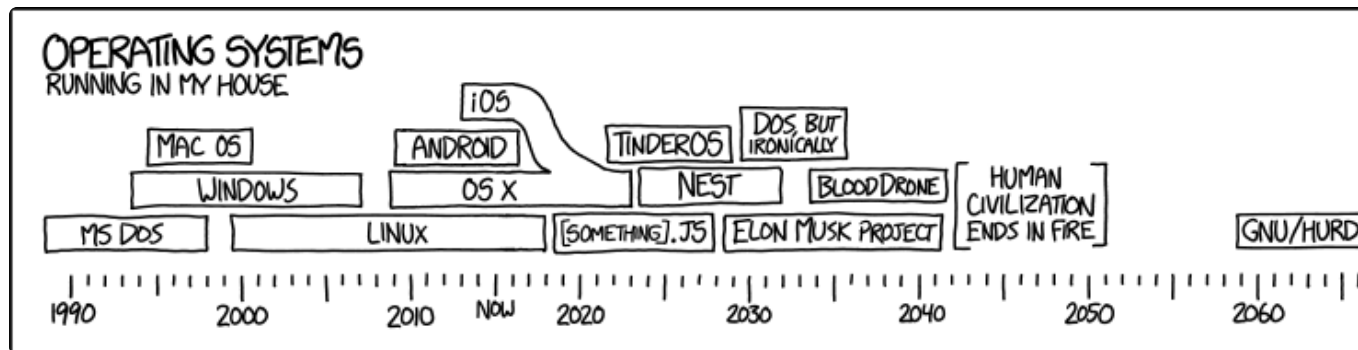


His vision is that software should be:

- free to share and modify
- GPL and LGPL licenses
 - allows selling and modifying
 - modified code must be relicensed under GPL

GNU/Linux = ❤️

GNU Software on top of the Linux kernel



- GNU Hurd 🧑🏫🚧 (work in progress for many many years)
- The Linux kernel 🐧 still is the *temporary replacement*

Base Distributions

The main Linux distributions that started it

Took the Linux kernel, added the GNU libraries and tools and wrote a package manager to install software

	Name	Tagline	Package Format	Package Manager	Release Year
	<u>Slackware</u>	<i>Oldest distribution</i>	tgz	pkgtool	1993
	<u>Debian</u>	<i>Free Software</i>	deb	apt / dpkg	1993
	<u>Red Hat Linux</u>	<i>Enterprise</i>	rpm	dnf / rpm	1995

These are the base for most of the modern distributions that we have today.

Modern Distributions

Name	Tagline	Parent	Package Format	Package Manager	Release Year
 <u>SUSE Linux</u>	<i>Enterprise-grade Linux</i>	<i>Originally Slackware, later RPM-based</i>	rpm	zypper , yast	1994
 <u>ArchLinux</u>	<i>Minimal Linux</i>	N/A	pkg.tar.zst	pacman	2002
 <u>Fedora</u>	<i>Desktop Linux</i>	<i>Red Hat Linux renamed</i>	rpm	dnf / rpm	2003
 <u>Ubuntu</u>	<i>Linux for Humans</i>	Debian	deb and snap	apt / dpkg and snap	2004
 <u>openSUSE</u>	<i>Stable, usable Linux for everyone</i>	SUSE Linux	rpm	zypper , rpm	2005

Which is the most used Linux distribution?



Non GNU Distributions

They use Linux, but most of the software is not from GNU

Distribution	Tagline	Parent	Package Format	Package Manager	Release Year
 <u>Android</u>	<i>Mobile Linux platform</i>	Linux kernel (AOSP)	<code>.apk</code> (only Android apps)	AOSP tools (not typical package manager)	2008
 <u>ChromeOS</u>	<i>The cloud-first OS</i>	Gentoo Linux	Custom (<code>.crx</code> , <code>.apk</code> , others)	<code>portage</code> , <code>cros_sdk</code> , Flatpak (via Crostini)	2011

Conclusion

we talked about

- What is Operating System is
- The Operating System Stack
- Virtual Machines
- History of Linux
- Distributions