Introduction to Operating Systems

1. Introduction

2026-11-03

USO Team

Welcome

to the *Introduction to Operating Systems* 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

 Copyright (C) 2026 Wyliodrin & Politehnica Bucharest, CC-BY-SA 4.0

Our team - Leads

Lectures

- Răzvan Rughiniș (CA)
- Mihai Carabaș (CB)
- Sergiu Weisz (CC)
- Alexandru Radovici (CD)
- Ioana Culic (AC)

Team Leads

- Alice Șuiu - Course tests
- Ionuț Mihalache - Homeworks
- Ștefan Jumărea - Practical Tests and Infra
- Andreia Ocănoaia - Labs

 Copyright (C) 2026 Wyliodrin & Politehnica Bucharest, CC-BY-SA 4.0

Team members

- Ana Gabriela Almăjanu
- Alexandru Răduță
- Alexandru-Nicolae Ocănoaia
- Vlad Cătănoiu
- Adelin-Costin Popa
- Diana Vulpoi
- Gabriel Pitic
- Rareș Sarmasag
- Maria Irina Gherman
- Catalin Cojocaru
- Rareș-Constantin Rîjniță
- Robert-Paul Pașca
- Andrei-Costin Carp
- Eduard-Ștefan Pană
- Amir Fallah-mirzaei
- Maria Sfîrăială
- Ştefăniţă-Alexandru Stoica
- Florin Postolache
- Andrei-George Nicola
- Elena-Luiza Olaru
- Maria-Teodora Popescu
- Aureliu Valentin Antonie
- Răzvan Vîrtan
- Răzvan Deaconescu
- Cătălin Pușcoci
- Daniel-Ioan Dinu
- Vlad-Iulius Năstase
- Ionuț Mihalache
- Vaduva Jan Alexandru
- Alice Suiu
- Resul Ebru
- Andreia Ocănoaia
- Alexandru Razvan Căciulescu
- Ștefan Jumărea
- Eric-Emanuel Tudorică
- Răzvan Voica
- Alex Cioloca
- Ana-Florentina Neacșu
- Darius-Florin Epure
- Maria-Alexandra Neacşu
- Cristian-Andrei Pomenea
- Alex-Ciprian Fechet
- Ştefan-Rareş Cazan
- Darius-Octavian Chifor
- Silviu-Adelin Stoican
- Robert-Alexandru Ghiță
- Ionuţ-Valentin Anghel
- Izabela-Maria Stoica
- Patrick David Covaci
- Ioana-Ruxandra Banu
- Ilinca Savu
- Victor Lișman
- Eva Cosma
- Irina Bradu
- Alexandru Iova
- Genan Omer

 Copyright (C) 2026 Wyliodrin & Politehnica Bucharest, CC-BY-SA 4.0

Outline

Activities

- 12 Lectures
- 12 Labs

Subjects

1. Intro to Virtual Machines
2. File, Folders and Path
3. Text Editors (vim, nvim or helix)
4. Users & Groups
5. Process Management
6. Linux Command Line (text processing)
7. Software Development
8. Source Code Repositories (git)
9. Shell Scripts (bash)
10. Networking

Rules

- No AI access
- No *stackoverflow.com*
- DuckDuck GO search engine

4 homework assignments

> *that will help you prepare for the hands on tests*

 Copyright (C) 2026 Wyliodrin & Politehnica Bucharest, CC-BY-SA 4.0

Grading

Part	Description	Points
Lecture tests	You will have a four lecture tests.	0.5p
Homework	You will have 4 homework assignments to exercise you for the hands-on tests	1p
Lab activity	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	*You will need at least **2.5p during the semester and 5 points in total** to pass the subject.*	**11p**

 Copyright (C) 2026 Wyliodrin & Politehnica Bucharest, CC-BY-SA 4.0

Notation

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

Code

```
$ command arguments
```

 Copyright (C) 2026 Wyliodrin & Politehnica Bucharest, CC-BY-SA 4.0

Notation

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

Code

```
$ command arguments
this is what the command wrote
```

 Copyright (C) 2026 Wyliodrin & Politehnica Bucharest, CC-BY-SA 4.0

Notation

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

Code

```
$ command arguments
this is what the command wrote
$ command2 arguments placed \ # mind the \ that splits the command over multiple lines
           using multiple \
           lines
```

 Copyright (C) 2026 Wyliodrin & Politehnica Bucharest, CC-BY-SA 4.0

Notation

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

Code

```
$ command arguments
this is what the command wrote
$ command2 arguments placed \ # mind the \ that splits the command over multiple lines
           using multiple \
           lines
```

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

 Copyright (C) 2026 Wyliodrin & Politehnica Bucharest, CC-BY-SA 4.0

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

– Albert Einstein

 Copyright (C) 2026 Wyliodrin & Politehnica Bucharest, CC-BY-SA 4.0

Bibliography

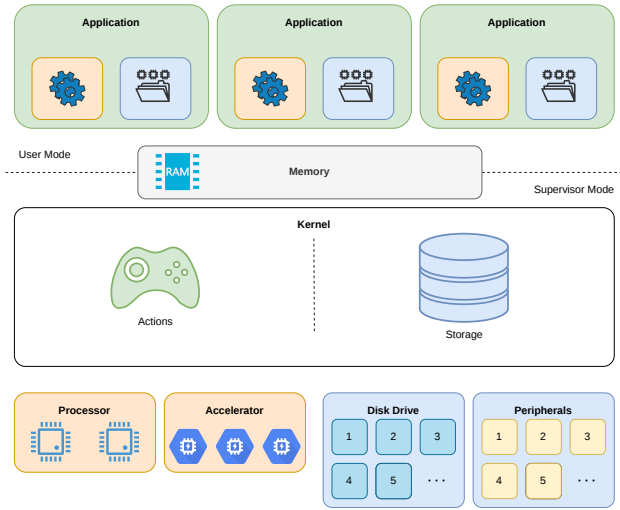
1. **Brian Ward**, *How LINUX Works*, 3rd Edition, No Starch Press, 2021
2. **Razvan Deaconescu, Razvan Rughinis, Mihai Carabas, Alexandru Radovici**, *Utilizarea Sistemelor de Operare*, Printech 2021, download

 Copyright (C) 2026 Wyliodrin & Politehnica Bucharest, CC-BY-SA 4.0

Operating System

the purpose of an OS

The *Core Role* of an Operating System

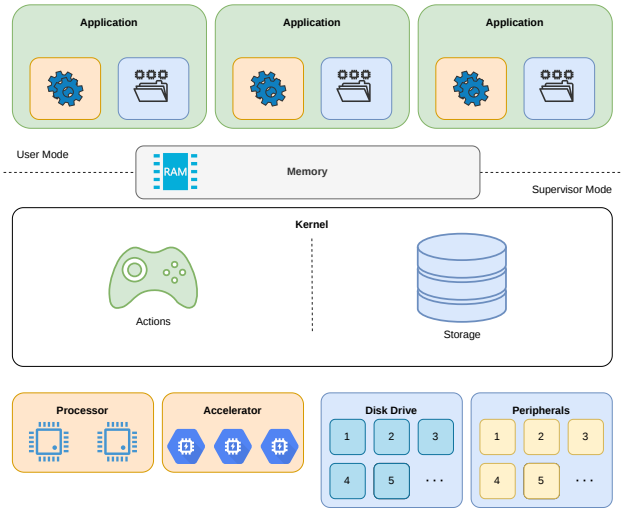


 Copyright (C) 2026 Wyliodrin & Politehnica Bucharest, CC-BY-SA 4.0

The *Core Role* of an Operating System

Allow Portability

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



 Copyright (C) 2026 Wyliodrin & Politehnica Bucharest, CC-BY-SA 4.0

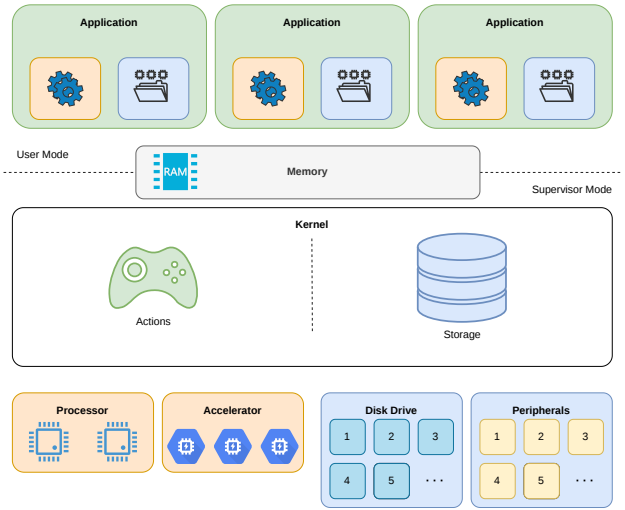
The *Core Role* of an Operating System

Allow Portability

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

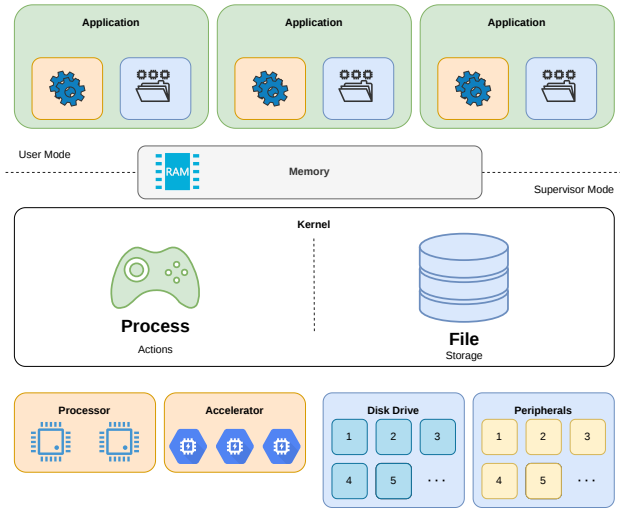
Resources Management and Isolation

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



 Copyright (C) 2026 Wyliodrin & Politehnica Bucharest, CC-BY-SA 4.0

Desktop and Server Operating Systems - abstractions

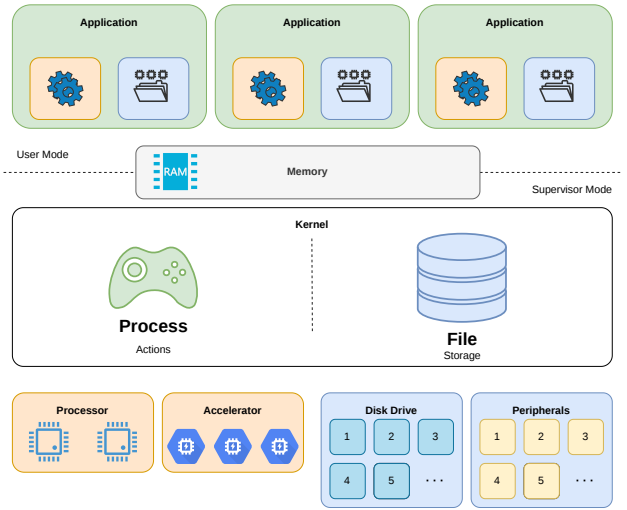


 Copyright (C) 2026 Wyliodrin & Politehnica Bucharest, CC-BY-SA 4.0

Desktop and Server Operating Systems - abstractions

Actions

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



 Copyright (C) 2026 Wyliodrin & Politehnica Bucharest, CC-BY-SA 4.0

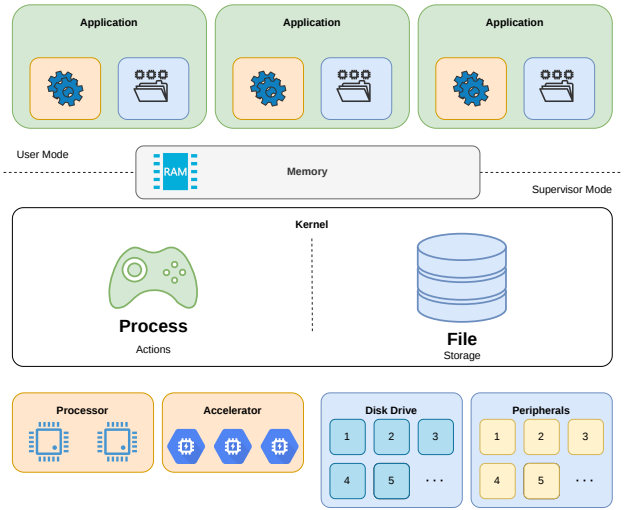
Desktop and Server Operating Systems - abstractions

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)



 Copyright (C) 2026 Wyliodrin & Politehnica Bucharest, CC-BY-SA 4.0

Embedded Operating Systems

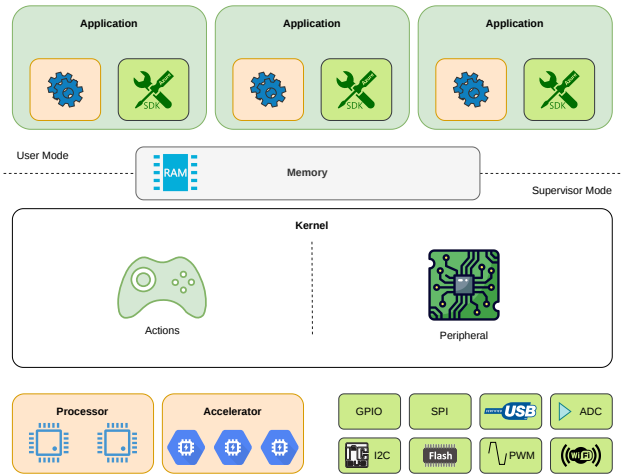
Actions

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

Peripheral

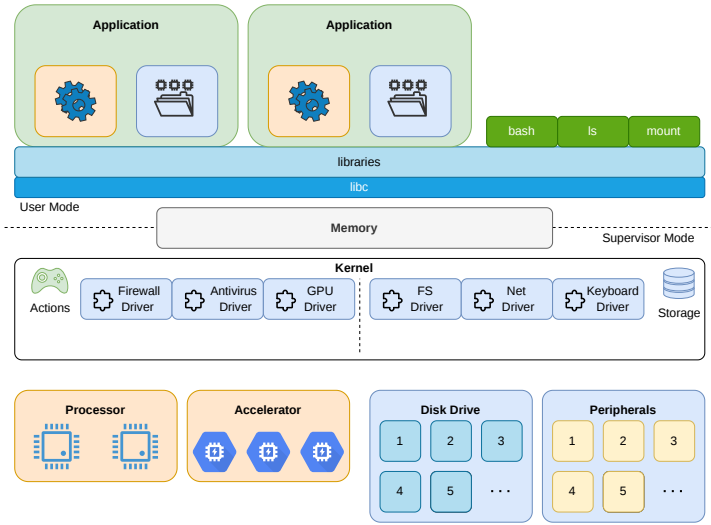
- provide a hardware independent API
- prevent processes from accessing the peripheral

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



 Copyright (C) 2026 Wyliodrin & Politehnica Bucharest, CC-BY-SA 4.0

The OS Stack



 Copyright (C) 2026 Wyliodrin & Politehnica Bucharest, CC-BY-SA 4.0

Application Examples

What it does	Windows	Linux	macOS
Desktop	`explorer.exe`	`nautilus`	Finder
Applications	`taskbar.exe`	`gnome-shell`	Dock
Files	`explorer.exe`	`nautilus`	Finder
Settings	`start ms-settings:`	`gnome-control-center`	System Settings
Commands	`cmd.exe` or `powershell.exe`	`bash` or `zsh`	`bash` or `zsh`
Documents	`powerpoint.exe`	`libreoffice`	Pages
Edit Code	`code.exe`	`code`	`code`
Terminal[1]	*handled by the OS*	`ptyxis`	Terminal

[1]Third party recommendations for Linux and macOS are Kitty, Alacritty and Ghostty

 Copyright (C) 2026 Wyliodrin & Politehnica Bucharest, CC-BY-SA 4.0

Where can we see the core of the OS

Where can we see the core of the OS

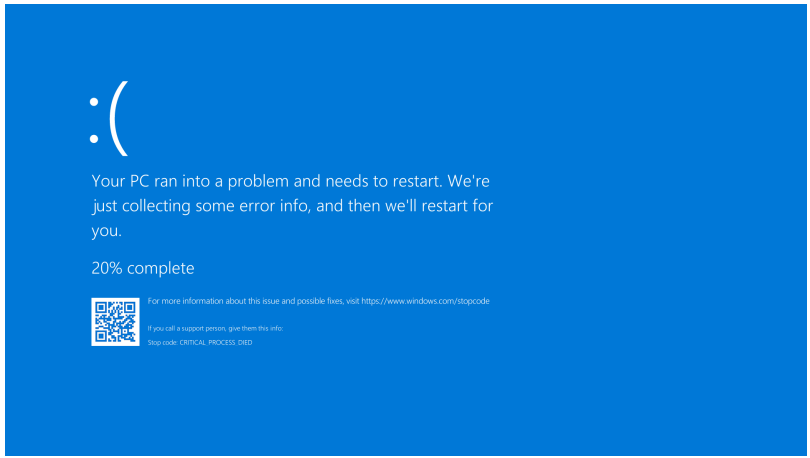
```
Booting Linux on physical CPU 0x0
Linux version 4.1.0-rc1-00016-g5704046-dirty (ags@trochilidae) (gcc version 4.9.
3 20141031 (prerelease) (Linaro GCC 4.9-2014.11) ) #68 Sat May 16 10:51:07 CEST
2015
CPU: ARMv7-M [410fc241] revision 1 (ARMv7M), cr=00000000
CPU: unknown data cache, unknown instruction 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: 46800K/49152K available (895K kernel code, 52K rwdata, 312K rodata, 64K
init, 165K bss, 2352K reserved, 0K cma-reserved)
Virtual kernel memory layout:
    vector  : 0x00000000 - 0x00001000   (   4 kB)
    fixmap  : 0xffc00000 - 0xfff00000   (3072 kB)
    vmalloc : 0x00000000 - 0xffffffff   (4095 MB)
    lowmem  : 0x8c000000 - 0x8f000000   (  48 MB)
      .text : 0x0f000080 - 0x0f12e050   (1208 kB)
      .init : 0x8c00a000 - 0x8c00d000   (  12 kB)
      .data : 0x8c008000 - 0x8c018060   (  65 kB)
       .bss : 0x8c018060 - 0x8c041630   ( 166 kB)
```



During Boot

 Copyright (C) 2026 Wyliodrin & Politehnica Bucharest, CC-BY-SA 4.0

Where can we see the core of the OS



Blue Screen (BSoD)

 Copyright (C) 2026 Wyliodrin & Politehnica Bucharest, CC-BY-SA 4.0

Where can we see the core of the OS

```
Waiting for driver initialization.
Scanning and configuring dmraid supported devices
Scanning logical volumes
  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.
kjournald starting.  Commit interval 5 seconds
EXT3-fs: mounted filesystem with ordered data mode.
Setting up other filesystems.
Setting up new root fs
no fstab.sys, mounting internal defaults
Switching to new root and running init.
unmounting old /dev
unmounting old /proc
unmounting old /sys
type=1404 audit(1381844547.109:2): 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!
_
```



Linux Kernel Panic

 Copyright (C) 2026 Wyliodrin & Politehnica Bucharest, CC-BY-SA 4.0

Where can we see the core of the OS



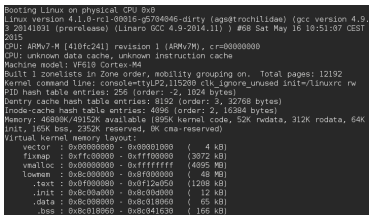


macOS Kernel Panic

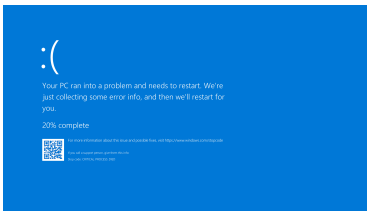
 Copyright (C) 2026 Wyliodrin & Politehnica Bucharest, CC-BY-SA 4.0

Where can we see the core of the OS

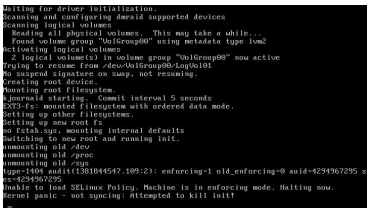
During Boot

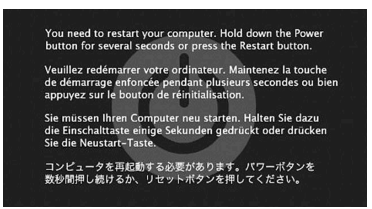


Blue Screen (BSoD)



Kernel Panic





 Copyright (C) 2026 Wyliodrin & Politehnica Bucharest, CC-BY-SA 4.0

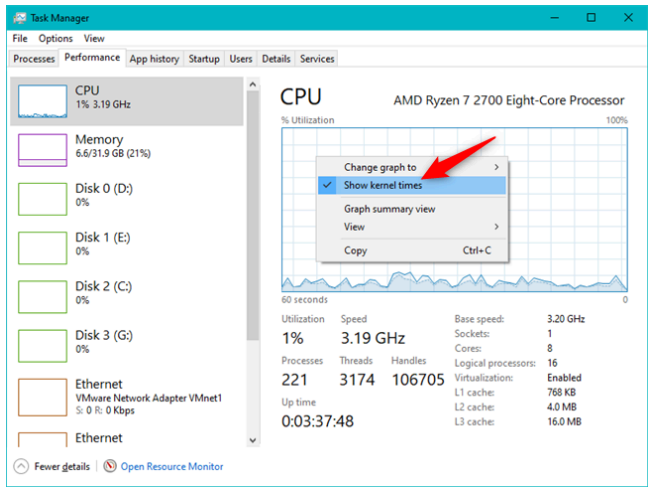
The Kernel

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

 Copyright (C) 2026 Wyliodrin & Politehnica Bucharest, CC-BY-SA 4.0

The Kernel

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

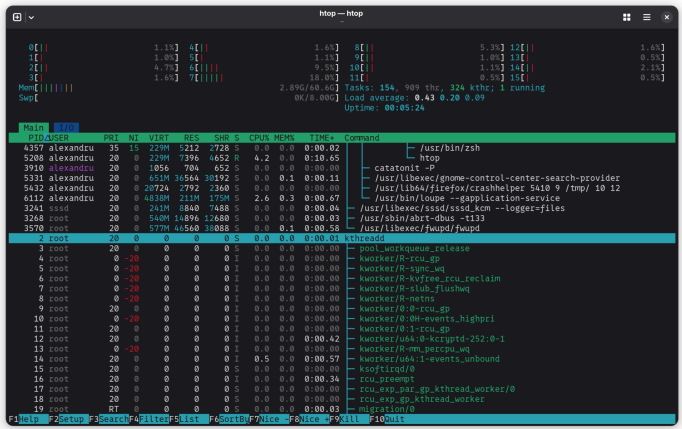


Windows Task Manager

 Copyright (C) 2026 Wyliodrin & Politehnica Bucharest, CC-BY-SA 4.0

The Kernel

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





htop

 Copyright (C) 2026 Wyliodrin & Politehnica Bucharest, CC-BY-SA 4.0

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

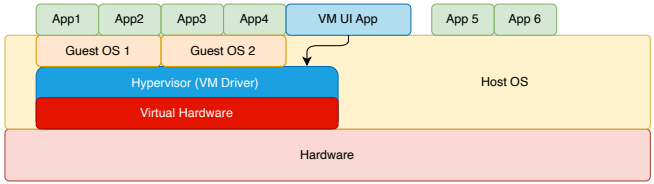
Virtualization Software

- VirtualBox
- VMWare
- QEMU

 Copyright (C) 2026 Wyliodrin & Politehnica Bucharest, CC-BY-SA 4.0

Simulation / Emulation

simulate all the hardware - very slow



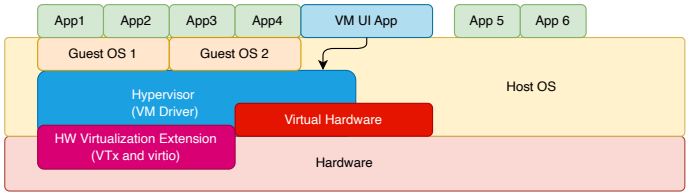
QEMU

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

 Copyright (C) 2026 Wyliodrin & Politehnica Bucharest, CC-BY-SA 4.0

Virtualization

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

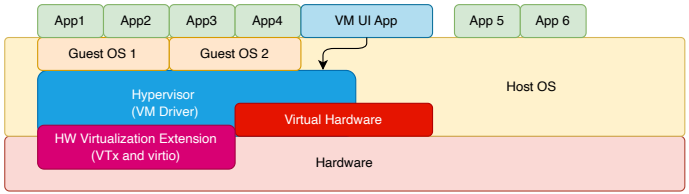


-
-

 Copyright (C) 2026 Wyliodrin & Politehnica Bucharest, CC-BY-SA 4.0

Virtualization

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

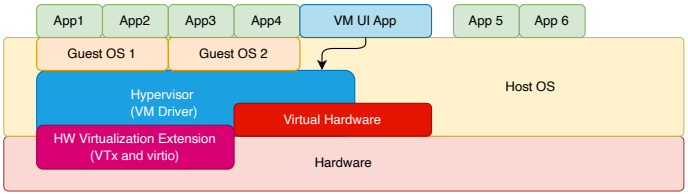


- Requires VT-x (Intel), AMD-V (AMD) or VHE (ARM)
-

 Copyright (C) 2026 Wyliodrin & Politehnica Bucharest, CC-BY-SA 4.0

Virtualization

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

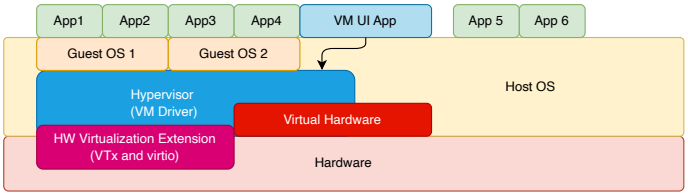


- Requires VT-x (Intel), AMD-V (AMD) or VHE (ARM)
- The guest OS has to be built for the same CPU architecture as the host OS
  - *Intel / AMD* is x86_64 (`amd64`)
  - *Apple Silicon* is ARM64 (`aarch64`)

 Copyright (C) 2026 Wyliodrin & Politehnica Bucharest, CC-BY-SA 4.0

Virtualization

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



- Requires VT-x (Intel), AMD-V (AMD) or VHE (ARM)
- The guest OS has to be built for the same CPU architecture as the host OS
  - *Intel / AMD* is x86_64 (`amd64`)
  - *Apple Silicon* is ARM64 (`aarch64`)

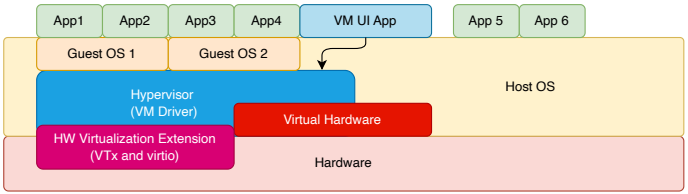
Hypervisors

- VMWare
- VirtualBox
- QEMU using `hyper-v`, `KVM` or *Hypervisor.framework*

 Copyright (C) 2026 Wyliodrin & Politehnica Bucharest, CC-BY-SA 4.0

Virtualization

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



Hypervisors

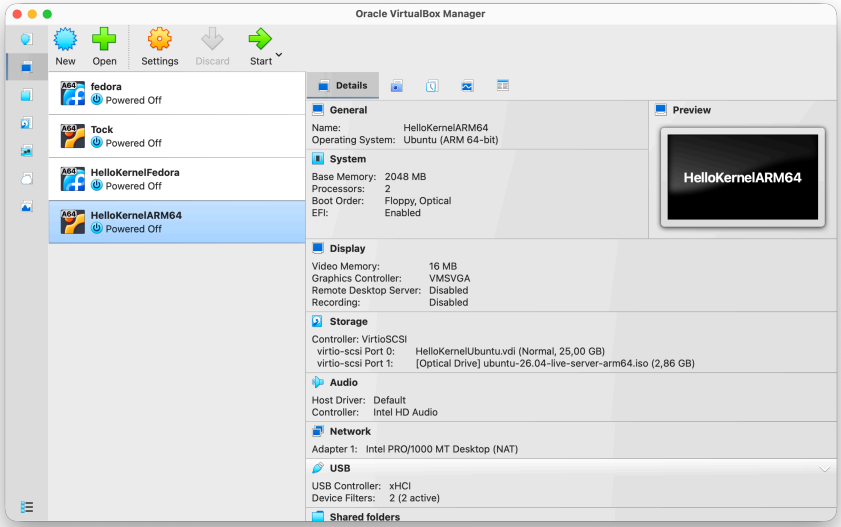
- VMWare
- VirtualBox
- QEMU using `hyper-v`, `KVM` or *Hypervisor.framework*

- Requires VT-x (Intel), AMD-V (AMD) or VHE (ARM)
- The guest OS has to be built for the same CPU architecture as the host OS
  - *Intel / AMD* is x86_64 (`amd64`)
  - *Apple Silicon* is ARM64 (`aarch64`)

> Make sure you download a Linux version for `amd64` or `aarch64`

 Copyright (C) 2026 Wyliodrin & Politehnica Bucharest, CC-BY-SA 4.0

VirtualBox Machine Settings

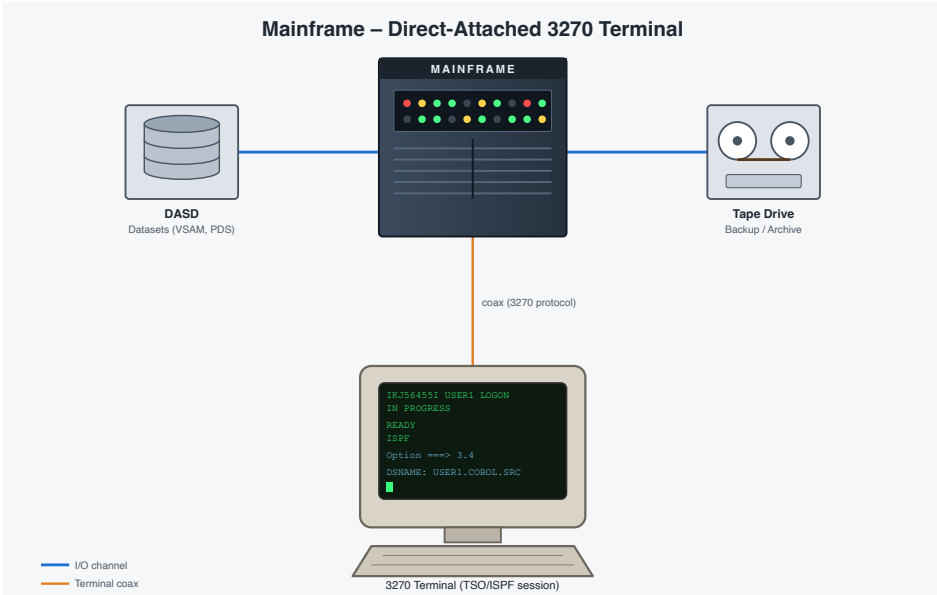


 Copyright (C) 2026 Wyliodrin & Politehnica Bucharest, CC-BY-SA 4.0

Distribution

what an OS is from the user's point of view

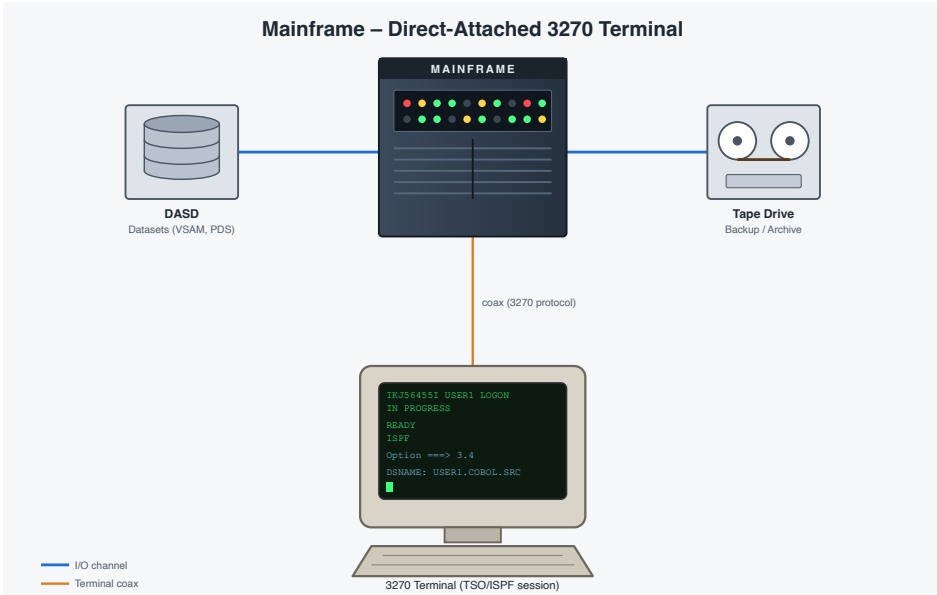
Meet the *Mainframe*



Size: a full room
IO: remote terminal
Users: single
Programs: one batch job

 Copyright (C) 2026 Wyliodrin & Politehnica Bucharest, CC-BY-SA 4.0

Meet the *Mainframe*



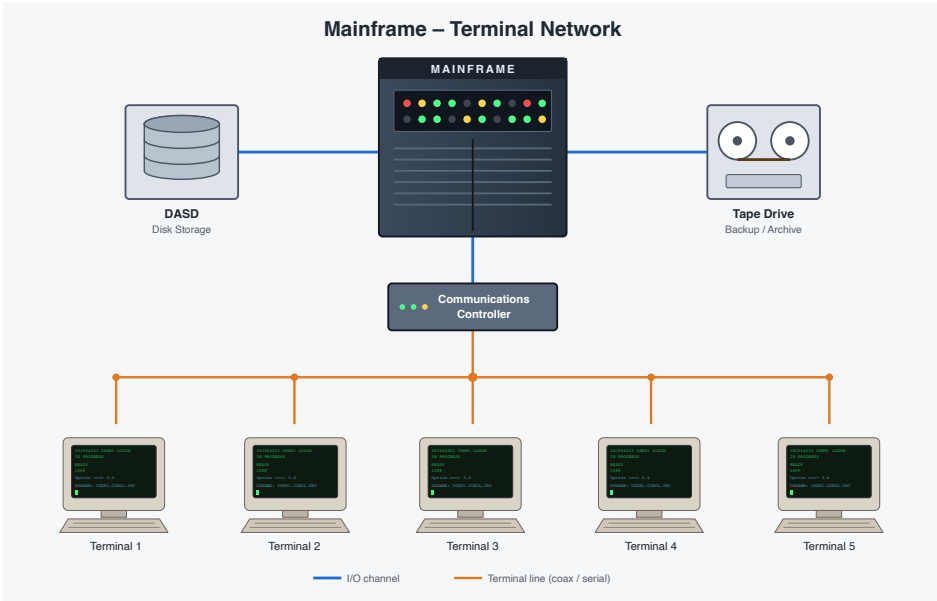
Size: a full room
IO: remote terminal
Users: single
Programs: one batch job



Waste of compute power

 Copyright (C) 2026 Wyliodrin & Politehnica Bucharest, CC-BY-SA 4.0

Meet the *Mainframe*



Size: a full room
IO: remote terminals
Users: multiple
Programs: multiple

Waste of compute power

The need for an Operating System

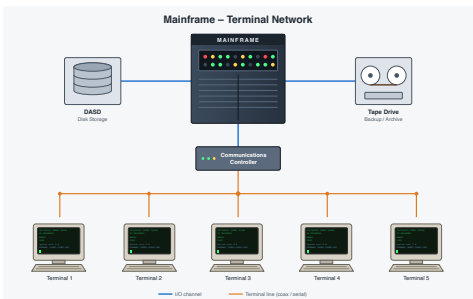
UNIX

 Copyright (C) 2026 Wyliodrin & Politehnica Bucharest, CC-BY-SA 4.0

UNIX

Mainframes provide a lot of hardware

- need to be shared by several users
- each user has a terminal connected to the mainframe
- run several programs in parallel



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



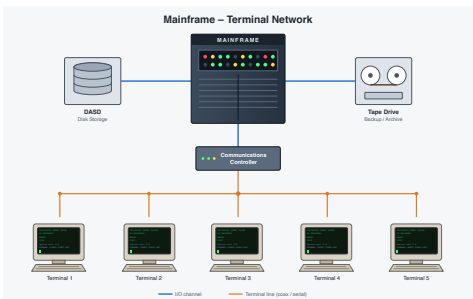
-
-

 Copyright (C) 2026 Wyliodrin & Politehnica Bucharest, CC-BY-SA 4.0

UNIX

Mainframes provide a lot of hardware

- need to be shared by several users
- each user has a terminal connected to the mainframe
- run several programs in parallel



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



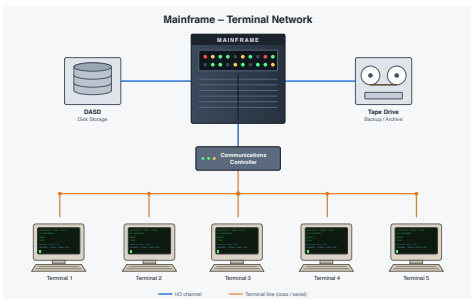
- Denis Ritchie writes C
-

 Copyright (C) 2026 Wyliodrin & Politehnica Bucharest, CC-BY-SA 4.0

UNIX

Mainframes provide a lot of hardware

- need to be shared by several users
- each user has a terminal connected to the mainframe
- run several programs in parallel



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



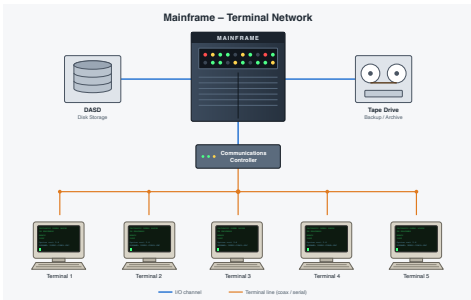
- Denis Ritchie writes C
- together with Ken Thompson use C to write UNIX.

 Copyright (C) 2026 Wyliodrin & Politehnica Bucharest, CC-BY-SA 4.0

UNIX

Mainframes provide a lot of hardware

- need to be shared by several users
- each user has a terminal connected to the mainframe
- run several programs in parallel



🔒 Is open source up to version *UNIX System III*

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



- Denis Ritchie writes C
- together with Ken Thompson use C to write UNIX.

 Copyright (C) 2026 Wyliodrin & Politehnica Bucharest, CC-BY-SA 4.0

The POSIX Standard

Portable Operating System Interface (ISO 9945)

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

 Copyright (C) 2026 Wyliodrin & Politehnica Bucharest, CC-BY-SA 4.0

The POSIX Standard

Portable Operating System Interface (ISO 9945)

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*

 Copyright (C) 2026 Wyliodrin & Politehnica Bucharest, CC-BY-SA 4.0

The POSIX Standard

Portable Operating System Interface (ISO 9945)

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 **Berkeley Software Distribution (BSD)**

FreeBSD	Darwin	OpenBSD	NetBSD
			

New kernels

Minix	Linux	RedoxOS[1]
		

[1]Partially POSIX compliant

 Copyright (C) 2026 Wyliodrin & Politehnica Bucharest, CC-BY-SA 4.0

GNU Software

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

-
-
-

-

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

 Copyright (C) 2026 Wyliodrin & Politehnica Bucharest, CC-BY-SA 4.0

GNU Software

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

- startup software (`init`)
-
-
-

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

 Copyright (C) 2026 Wyliodrin & Politehnica Bucharest, CC-BY-SA 4.0

GNU Software

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

- startup software (`init`)
- most of what we call today *coreutils*
-
-

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

 Copyright (C) 2026 Wyliodrin & Politehnica Bucharest, CC-BY-SA 4.0

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`
-

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

 Copyright (C) 2026 Wyliodrin & Politehnica Bucharest, CC-BY-SA 4.0

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`)

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

 Copyright (C) 2026 Wyliodrin & Politehnica Bucharest, CC-BY-SA 4.0

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 (*uutils*)[1]
- `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

[1] Used by Ubuntu starting with 26.04

 Copyright (C) 2026 Wyliodrin & Politehnica Bucharest, CC-BY-SA 4.0

The Linux Kernel



***Linus Torvalds* (University of Helsinki)**

- did not agree with A. Tannenbaum about how Minix should work
-

 Copyright (C) 2026 Wyliodrin & Politehnica Bucharest, CC-BY-SA 4.0

The Linux Kernel



***Linus Torvalds* (University of Helsinki)**

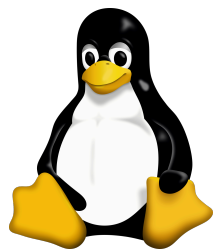
- did not agree with A. Tannenbaum about how Minix should work
- wrote his own 👩‍💻 *POSIX compliant* OS, called it Linu**x**

 Copyright (C) 2026 Wyliodrin & Politehnica Bucharest, CC-BY-SA 4.0

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*

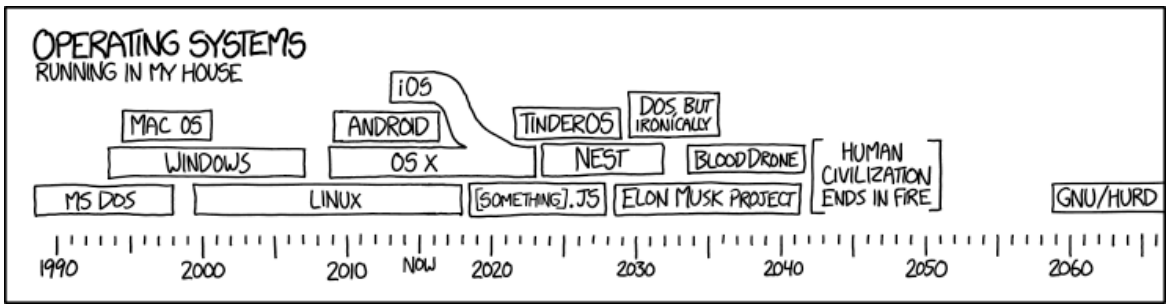
Linus Torvalds (University of Helsinki)

- did not agree with A. Tannenbaum about how Minix should work
- wrote his own 👨‍💻 *POSIX compliant* OS, called it Linu**x**

 Copyright (C) 2026 Wyliodrin & Politehnica Bucharest, CC-BY-SA 4.0

GNU/Linux = ❤️

GNU Software on top of the Linux kernel

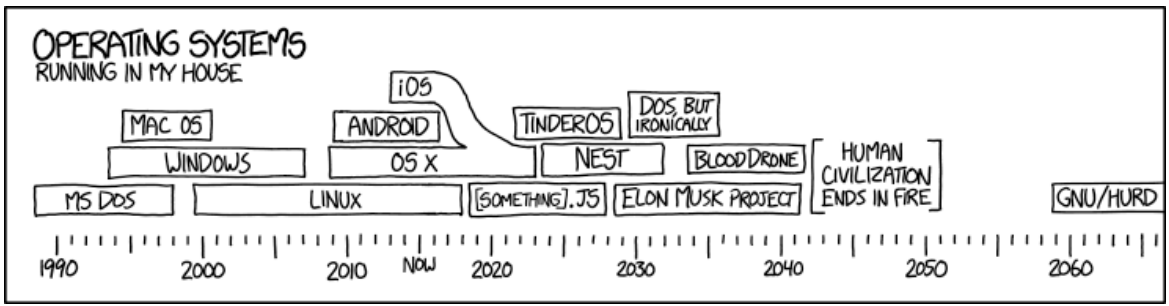


-
-

 Copyright (C) 2026 Wyliodrin & Politehnica Bucharest, CC-BY-SA 4.0

GNU/Linux = ❤️

GNU Software on top of the Linux kernel

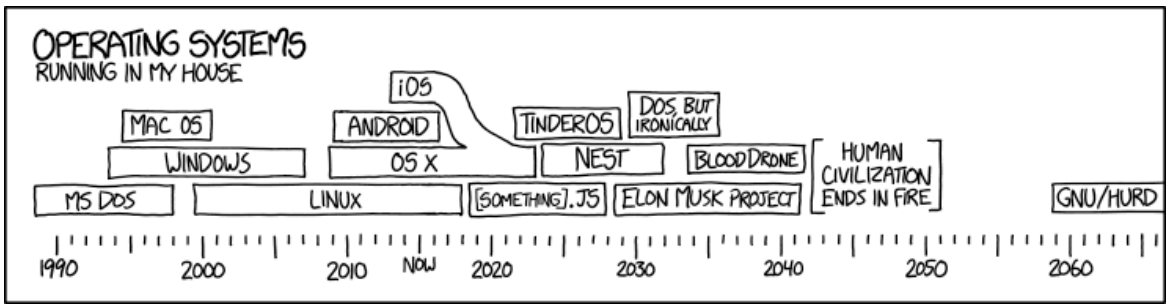


- GNU Hurd 👷🚧🏗️ (*work in progress for many many years*)
-

 Copyright (C) 2026 Wyliodrin & Politehnica Bucharest, CC-BY-SA 4.0

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*

 Copyright (C) 2026 Wyliodrin & Politehnica Bucharest, CC-BY-SA 4.0

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
	Slackware	*Oldest distribution*	`tgz`	`pkgtool`	1993
	Debian	*Free Software*	`deb`	`apt` / `dpkg`	1993
	Red Hat Linux	*Enterprise*	`rpm`	`dnf` / `rpm`	1995

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

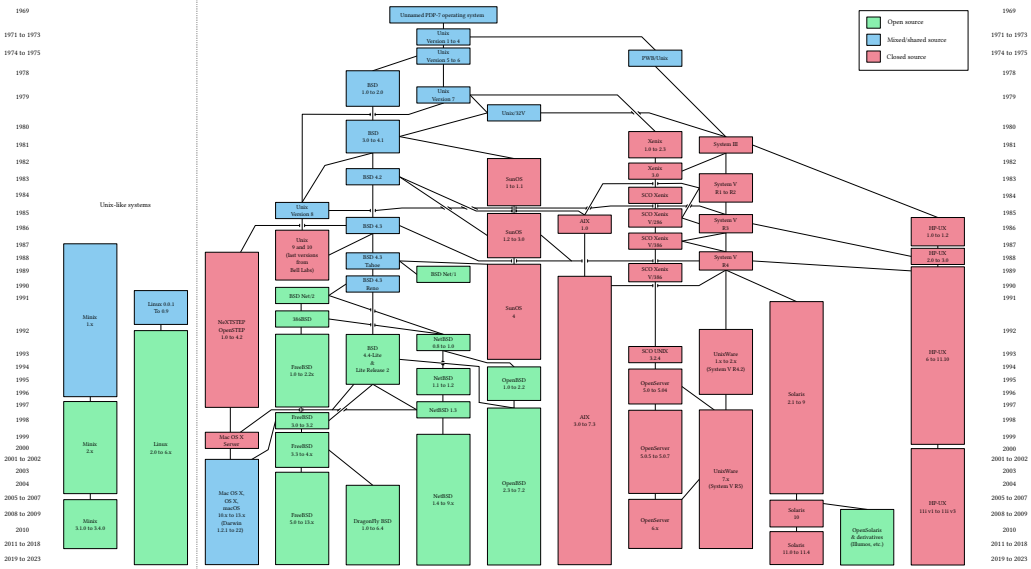
 Copyright (C) 2026 Wyliodrin & Politehnica Bucharest, CC-BY-SA 4.0

Modern Distributions

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

 Copyright (C) 2026 Wyliodrin & Politehnica Bucharest, CC-BY-SA 4.0

POSIX Distributions *Tree*



 Copyright (C) 2026 Wyliodrin & Politehnica Bucharest, CC-BY-SA 4.0

Most Used Linux Distribution?

 Copyright (C) 2026 Wyliodrin & Politehnica Bucharest, CC-BY-SA 4.0

Most Used Linux Distribution?



 Copyright (C) 2026 Wyliodrin & Politehnica Bucharest, CC-BY-SA 4.0

Non GNU Distributions

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

	Distribution	Tagline	Parent	Package Format	Package Manager	Release Year
	Android	*Mobile Linux platform*	Linux kernel (AOSP)	`.apk` (only Android apps) `.aab` (only Android apps) `.apex` (libraries)	AOSP tools (not typical package manager)	2008
	ChromeOS	*The cloud-first OS*	Gentoo Linux	Custom (`.crx`, `.apk`, others)	`portage`, `cros_sdk`, Flatpak (via Crostini)	2011

 Copyright (C) 2026 Wyliodrin & Politehnica Bucharest, CC-BY-SA 4.0

We talked about

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

 Copyright (C) 2026 Wyliodrin & Politehnica Bucharest, CC-BY-SA 4.0