

Introduction to Operating Systems

1. Introduction

2026-09-29

USO Team

Welcome



to the *Internal Structure of 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

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

Team members



- Ana Gabriela Almăjanu
- Alexandru Răduță
- Alexandru-Nicolae Ocănoaia
- Vlad Cătănoiu
- Adelin-Costin Popa
- Diana Vulpoi
- Gabriel Pitic
- Rareș Sarماسag
- Maria Irina Gherman
- Catalin Cojocar
- Rareș-Constantin Rîjniță
- Robert-Paul Pașca
- Andrei-Costin Carp
- Eduard-Ștefan Pană
- Amir Fallah-mirzaei
- Maria Sfîrăială
- Ștefăniță-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 Neacșu
- Cristian-Andrei Pomenea
- Alex-Ciprian Fechet
- Ștefan-Rareș Cazan
- Darius-Octavian Chifor
- Silviu-Adelin Stoican
- Robert-Alexandru Ghiță
- Ionuț-Valentin Anghel
- Izabela-Maria Stoica
- Patrick David Covaci
- Ioana-Ruxandra Banu
- Ilinca Savu
- Victor Lișman
- Eva Cosma
- Irina Bradu
- Alexandru Iova
- Genan Omer

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

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	<i>You will need at least 2.5p during the semester and 5 points in total to pass the subject.</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

```
$ command arguments
```

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

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

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**.

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

– Albert Einstein

Bibliography

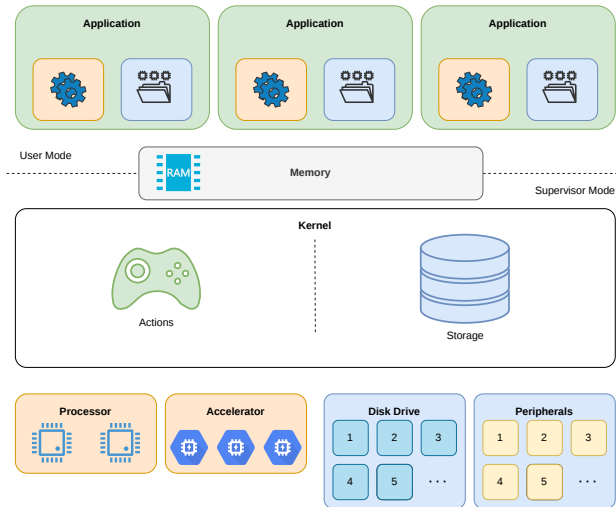


1. **Brian Ward**, *How LINUX Works*, 3rd Edition, No Starch Press, 2021
2. **Razvan Deaconescu, Razvan Rughinis, Mihai Carabas, Alexandru Radovici**, *Utilizarea Sistemelor de Operare*, Printech 2021, [download](#)

Operating System

the purpose of an OS

The *Core Role* of an Operating System

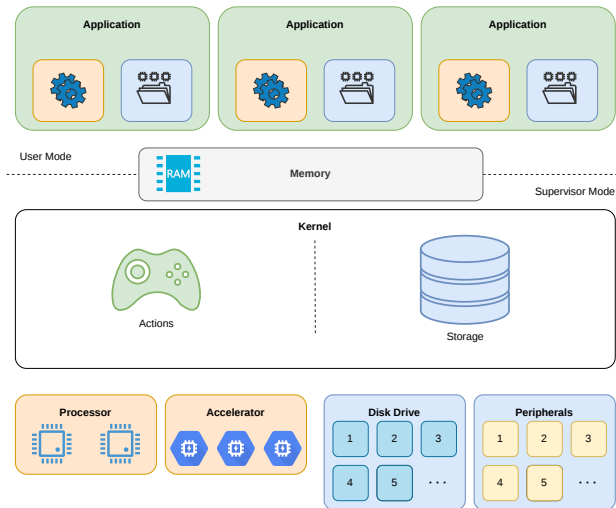


The *Core Role* of an Operating System



Allow Portability

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



The *Core* Role of an Operating System

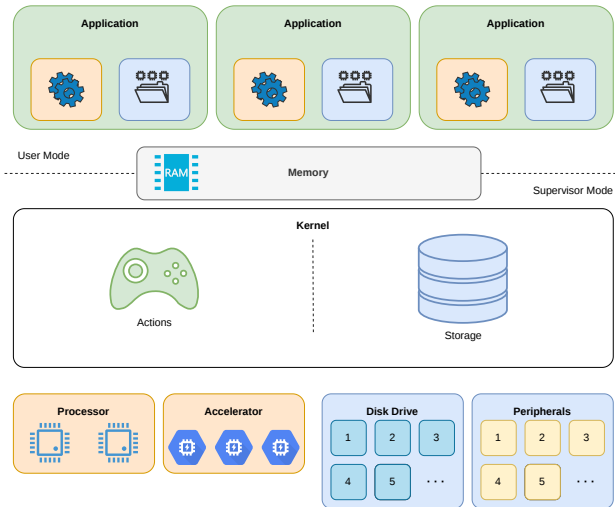


Allow Portability

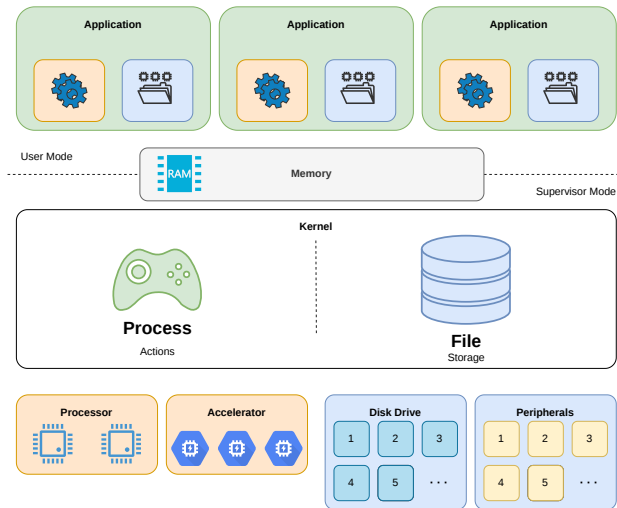
- provides a hardware independent API
- applications should run on any hardware

Resources Management and Isolation

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



Desktop and Server Operating Systems — abstractions

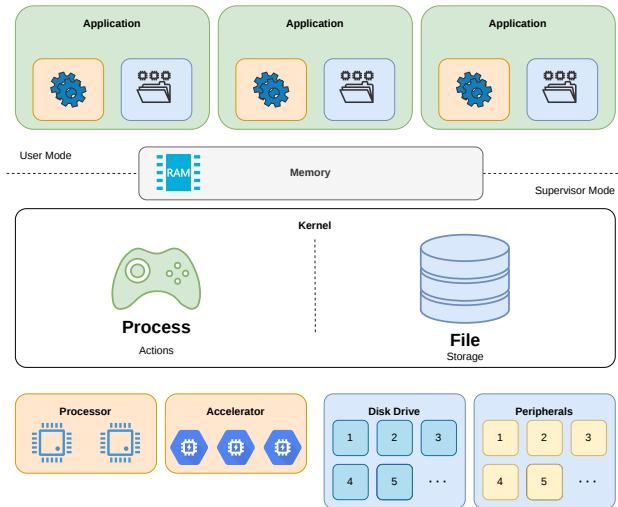


Desktop and Server Operating Systems — abstractions



Actions

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



Desktop and Server Operating Systems — abstractions

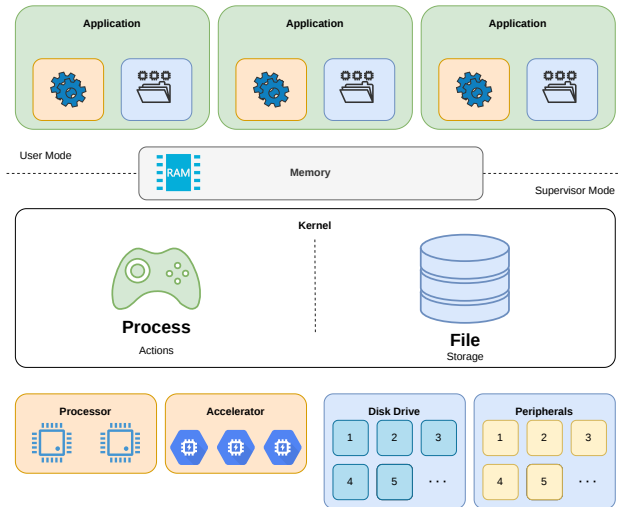


Actions

- *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



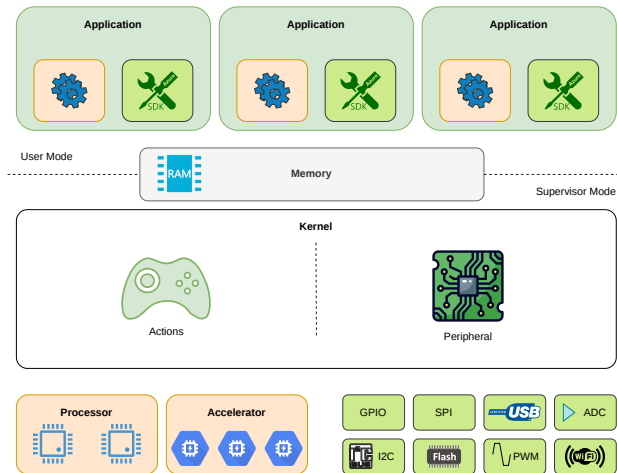
Actions

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

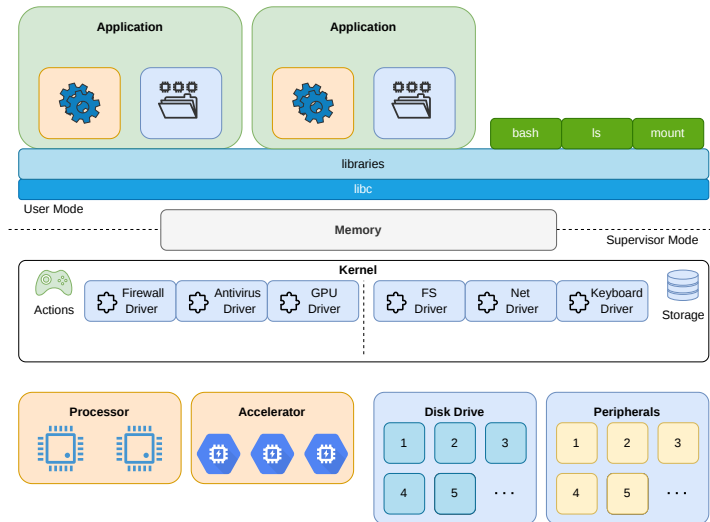
Peripheral

- 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



Application Examples



What it does	Windows	Linux	macOS
Desktop	<code>explorer.exe</code>	<code>nautilus</code>	Finder
Applications	<code>taskbar.exe</code>	<code>gnome-shell</code>	Dock
Files	<code>explorer.exe</code>	<code>nautilus</code>	Finder
Settings	<code>start ms-settings:</code>	<code>gnome-control-center</code>	System Settings
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>	Pages
Edit Code	<code>code.exe</code>	<code>code</code>	<code>code</code>
Terminal ¹	<i>handled by the OS</i>	<code>ptyxis</code>	Terminal

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

Where can we see the core of the OS

Where can we see the core of the OS

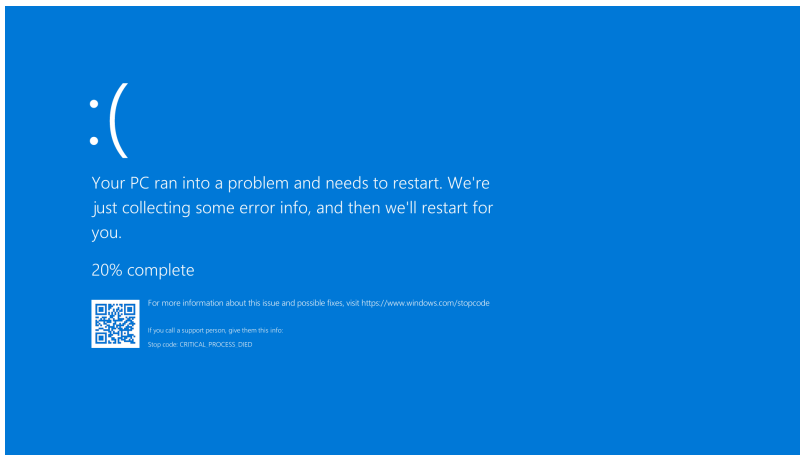


```
Booting Linux on physical CPU 0x0
Linux version 4.1.0-rc1-00016-g5704046-dirty (ags@trochilidae) (gcc version 4.9.
3 20141031 (prerelease) (Linaro GCC 4.9-2014.11) ) #68 Sat May 16 10:51:07 CEST
2015
CPU: ARMv7-M [410fc241] revision 1 (ARMv7M), cr=00000000
CPU: unknown data cache, unknown instruction 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 rdata, 312K rodata, 64K
init, 165K bss, 2352K reserved, 0K cma-reserved)
Virtual kernel memory layout:
vector : 0x00000000 - 0x00001000 ( 4 kB)
fixmap : 0xffc00000 - 0xffff0000 (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

Where can we see the core of the OS



Blue Screen (BSOD)

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

Where can we see the core of the OS



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

Veuillez 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。



macOS Kernel Panic

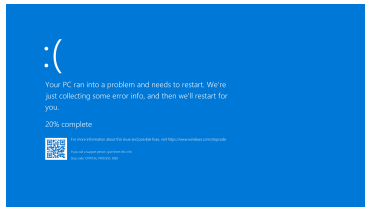
Where can we see the core of the OS



During Boot

```
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 (1895K kernel code, 52K rdata, 312K rodata, 64K init, 165K bss, 2352K reserved, 0K cma-reserved)
Virtual kernel memory layout:
   vector : 0x00000000 - 0x00001000   ( 4 kB)
   fixmap : 0xffc00000 - 0xffff0000   (3072 kB)
   vmalloc : 0x00000000 - 0xfffffff0   (4095 MB)
   lowmem  : 0x0c000000 - 0x0f000000   ( 40 MB)
   .text   : 0x0f000000 - 0x0f12e050   (1208 kB)
   .init   : 0x0c000000 - 0x0c00d000   ( 12 kB)
   .data   : 0x0c000000 - 0x0c018060   ( 65 kB)
   .bss    : 0x0c018060 - 0x0c041630   (165 kB)
```

Blue Screen (BSOD)



Kernel Panic

```
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 "Vol1Group00" using metadata type lvm2
Activating logical volumes
  2 logical volume(s) in volume group "Vol1Group00" now active
Trying to resume from /dev/Vol1Group00/LogVol01
To suspend signatures on swap, not resuming.
Creating root device.
Mounting root filesystem.
  Journaling 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(1338184454.169:12): enforcing=1 old_enforcing=0 auid=4294967295 s
ex=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

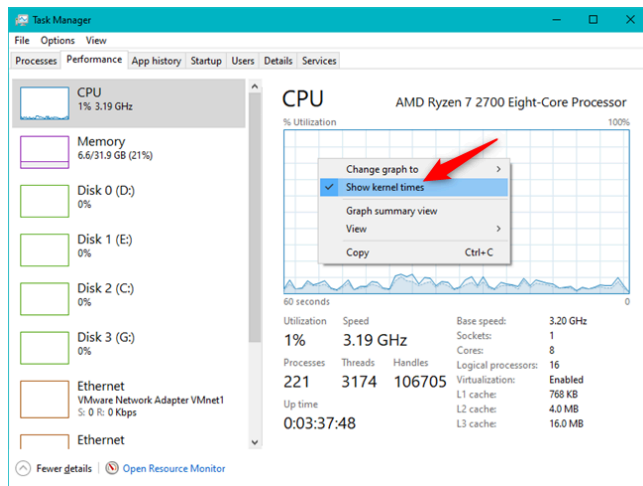


- is a collection of applications that *usually* run in privileged mode
- one *main application* and several *utilities apps*

The Kernel



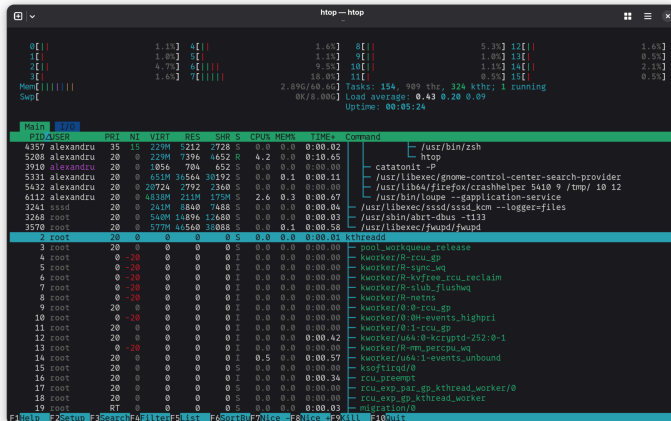
- is a collection of applications that *usually* run in privileged mode
- one *main application* and several *utilities apps*



Windows Task Manager

The Kernel

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



The screenshot shows the htop terminal window. At the top, system statistics are displayed: CPU usage (0% for 1s, 1.0% for 5s, 4.7% for 10s), memory usage (2.89G/60.4G), and tasks (154, 909 thr, 324 kthr, 1 running). The main table lists processes with columns for PID, USER, PRI, NI, VIRT, RES, SHR, S, CPU%, MEM%, TIME+, and Command. The processes are sorted by CPU usage, with 'catatonit -P' at the top. The bottom of the window shows a menu with options like F1 Help, F2 Setup, F3 Search, F4 Filter, F5 List, F6 SortBy, F7 Nice, F8 Kill, and 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

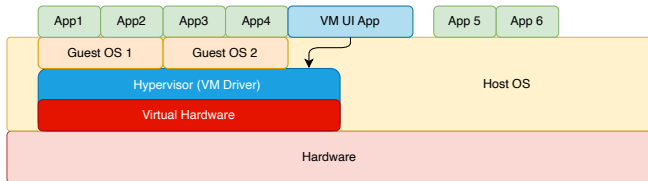
Virtualization Software

- VirtualBox
- VMWare
- QEMU

Simulation / Emulation



simulate all the hardware - very slow



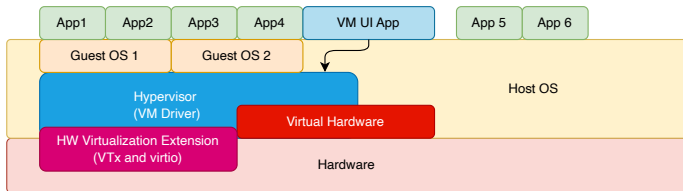
QEMU

- 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

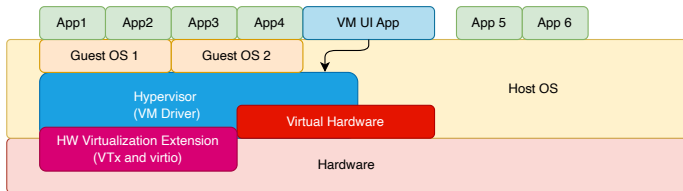


-
-

Virtualization



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



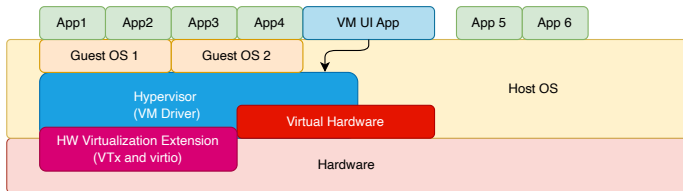
- Requires VT-x (Intel), AMD-V (AMD) or VHE (ARM)

-

Virtualization



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

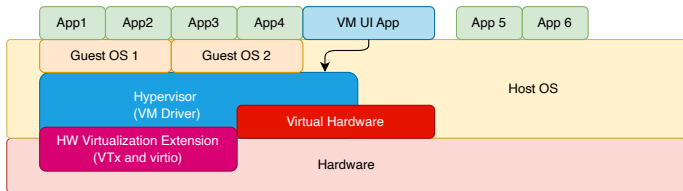


- Requires VT-x (Intel), AMD-V (AMD) or VHE (ARM)
- The guest OS has to be built for the same CPU architecture as the host OS
 - ▶ *Intel / AMD* is x86_64 (amd64)
 - ▶ *Apple Silicon* is ARM64 (aarch64)

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)

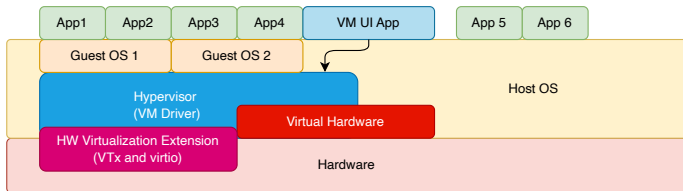
Hypervisors

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

Virtualization



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



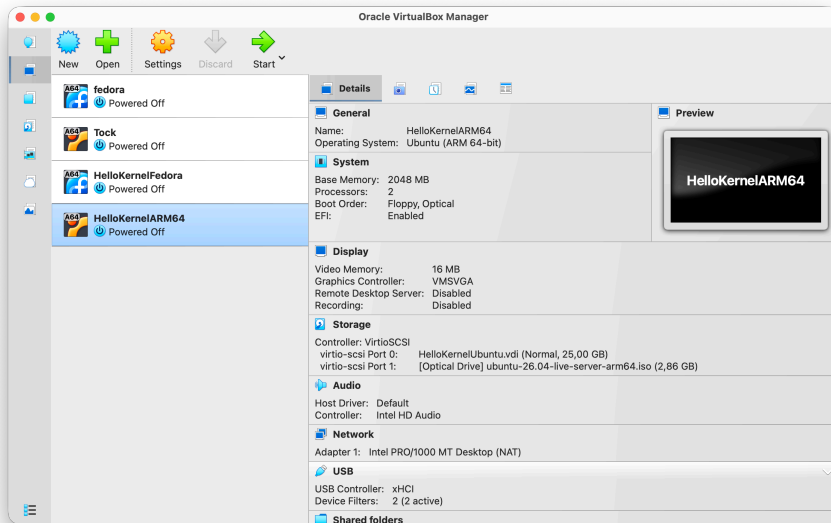
- Requires VT-x (Intel), AMD-V (AMD) or VHE (ARM)
- The guest OS has to be built for the same CPU architecture as the host OS
 - ▶ *Intel / AMD* is x86_64 (amd64)
 - ▶ *Apple Silicon* is ARM64 (aarch64)

Hypervisors

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

Make sure you download a Linux version for amd64 or aarch64

VirtualBox Machine Settings



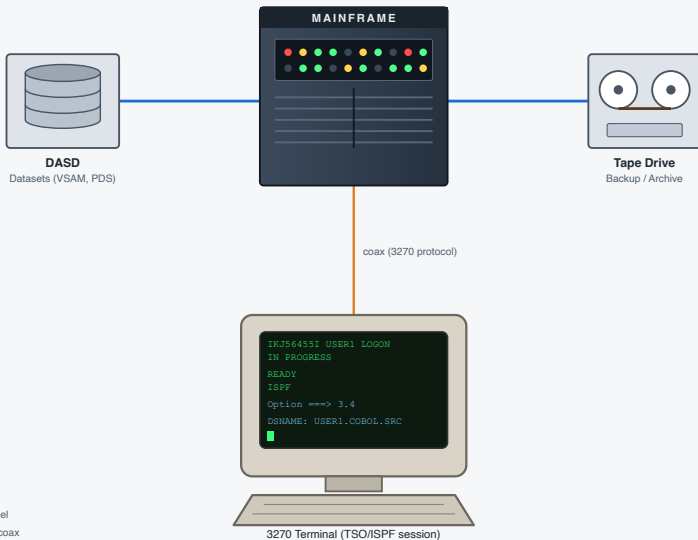
Distribution

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

Meet the *Mainframe*



Mainframe – Direct-Attached 3270 Terminal



Size: a full room

IO: remote terminal

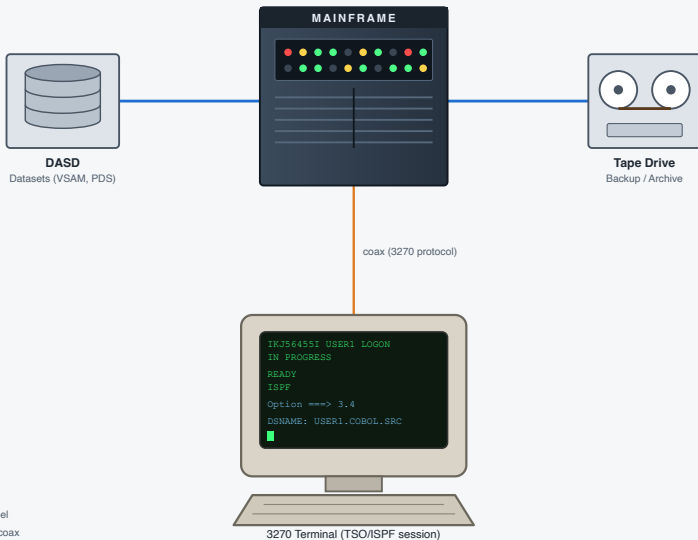
Users: single

Programs: one batch job

Meet the *Mainframe*



Mainframe – Direct-Attached 3270 Terminal



Size: a full room

IO: remote terminal

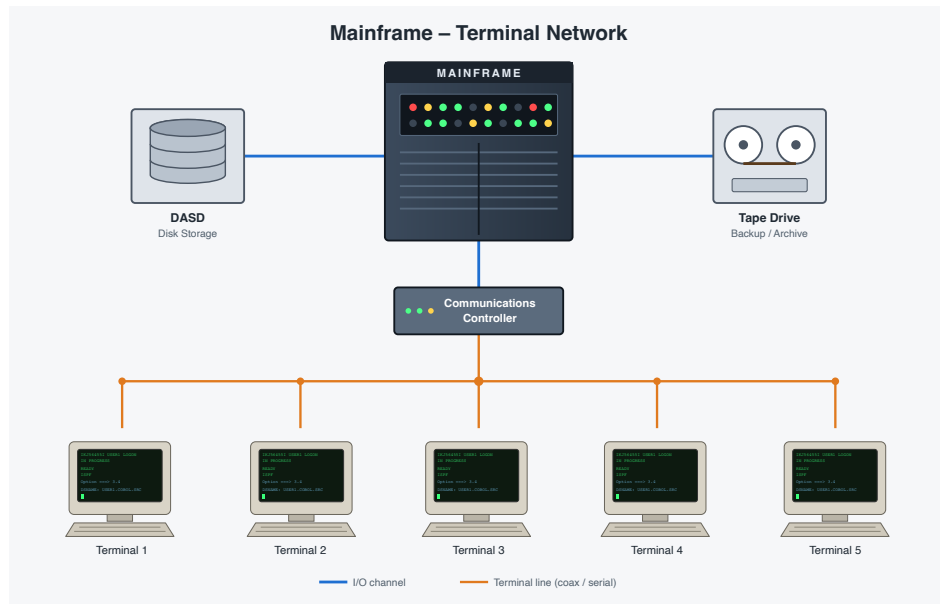
Users: single

Programs: one batch job



Waste of compute power

Meet the *Mainframe*



Size: a full room

IO: remote terminals

Users: multiple

Programs: multiple



Waste of compute power



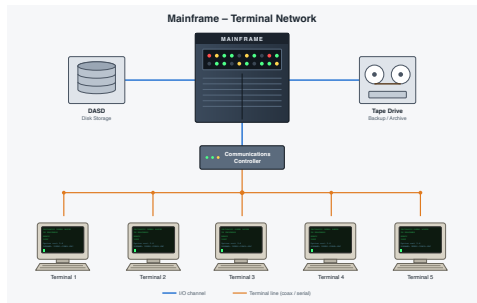
The need for an
Operating System

UNIX

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)

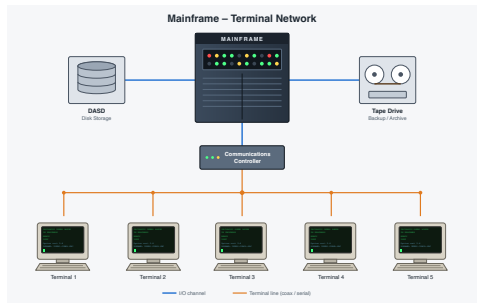


-
-

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)



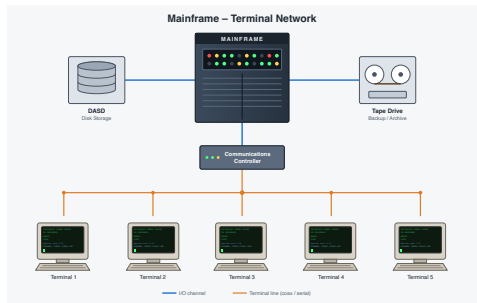
- Denis Ritchie writes C



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)

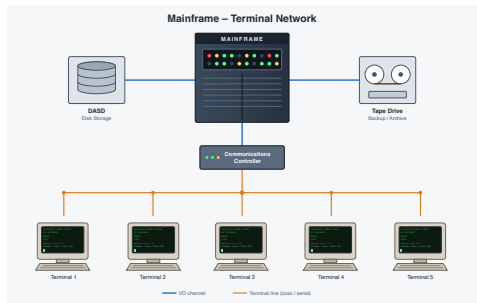


- Denis Ritchie writes C
- together with Ken Thompson use C to write UNIX.

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.

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

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*

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

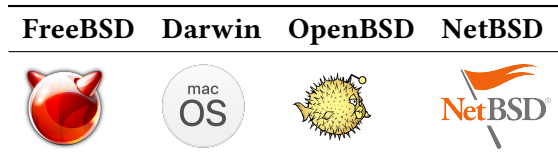
Allows:

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

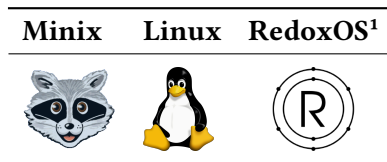
When UNIX became closed source ->



UC Berkeley Software Distribution (BSD)



New kernels



¹Partially POSIX compliant

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

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

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

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

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

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)

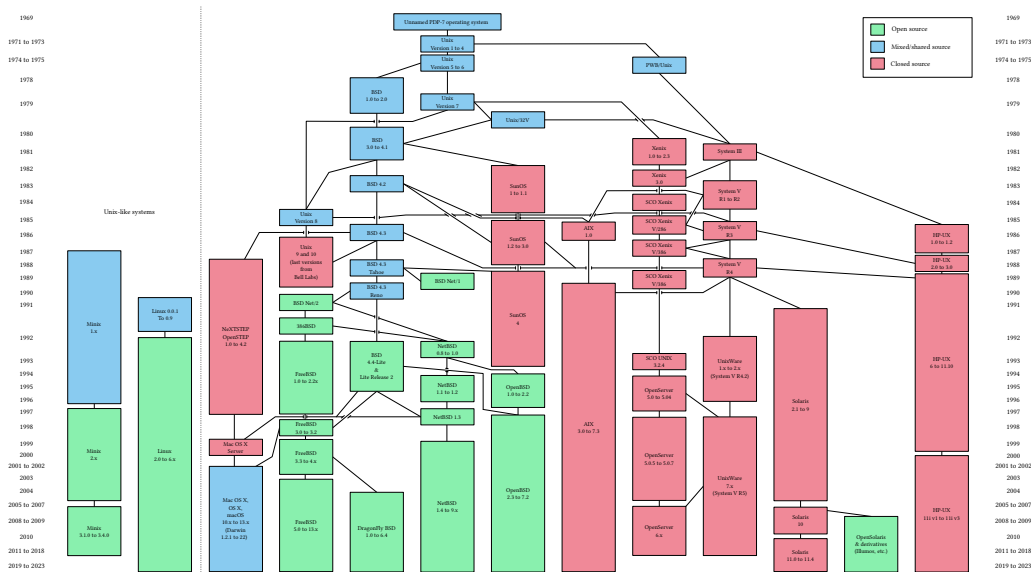


His vision is that software should be:

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

¹Used by Ubuntu starting with 26.04

POSIX Distributions *Tree*



The Linux Kernel



Linus Torvalds (University of Helsinki)

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

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 Linux

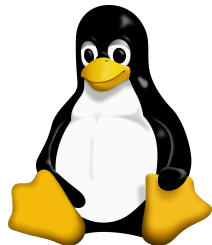
The Linux Kernel



Linus Torvalds (University of Helsinki)

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

- 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

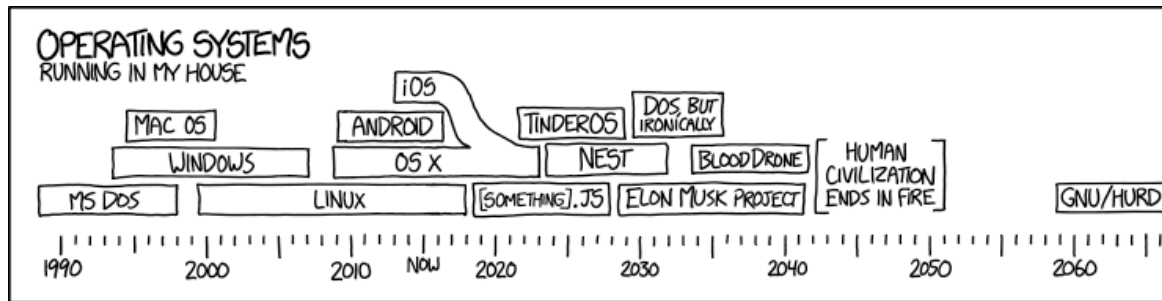


Meet Tux

GNU/Linux = ❤️



GNU Software on top of the Linux kernel

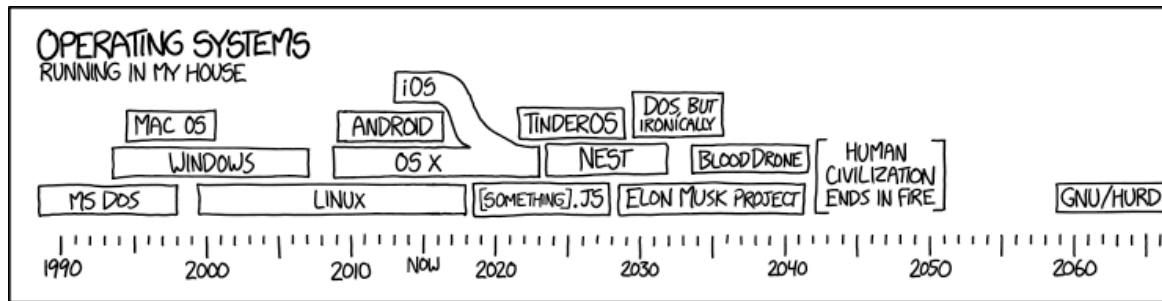


-
-

GNU/Linux = ❤️



GNU Software on top of the Linux kernel



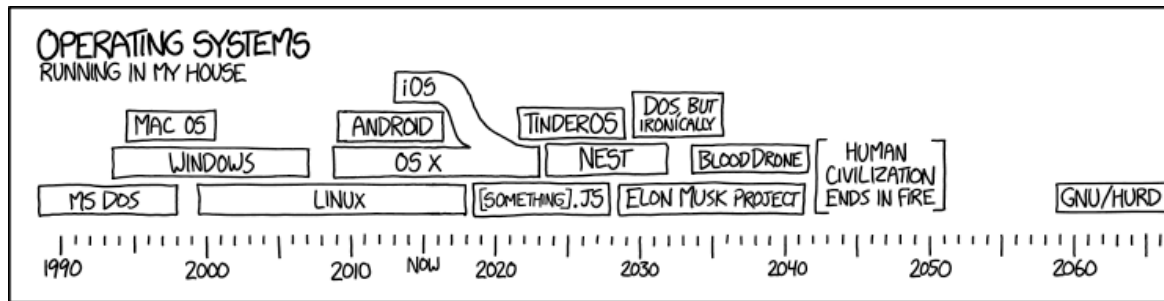
- GNU Hurd 🧑🏫🔧🔨 (work in progress for many many years)

•

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
	<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

Most Used Linux Distribution?



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)	.apk (only Android apps) .aab (only Android apps) .apex (libraries)	AOSP tools (not typical package manager)	2008
 <u>ChromeOS</u>	<i>The cloud-first OS</i>	Gentoo Linux	Custom (.crx, .apk, others)	portage, cros_sdk, Flatpak (via Crostini)	2011

We talked about

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

