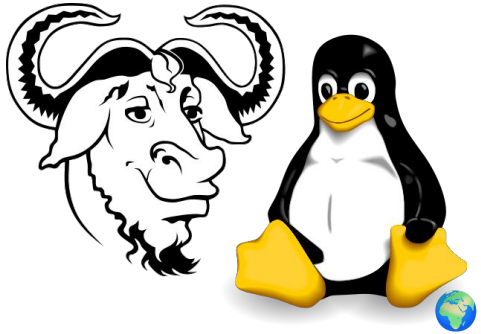
Linux

Open Source Operating System

Why Linux ?



✅ Open Source & Free

- Linux is free to use, modify, and distribute.
- Backed by a large open-source community.

🛡️ Secure by Design

- Multi-user system with strong permissions.
- Regular security patches and updates.

⚙️ Highly Customizable

- From lightweight systems to enterprise-grade servers.
- Choose your desktop environment, shell, and tools.

💻 Stability & Performance

- Powers over 90% of cloud infrastructure.
- Rarely needs reboots — ideal for servers.

🌍 Widely Used in Industry

- Android is based on Linux.
- Used in servers, supercomputers, IoT, and embedded systems.

🧠 Great for Learning

- Teaches OS internals, networking, scripting, and system architecture.
- Rich set of command-line tools.

💼 In-Demand Skill

- Essential for system administrators, DevOps, cloud engineers, and developers.
- Strong presence in job markets.

Linux = Transparency + Control + Community Defense

🧩 Open Source Transparency

- Code is visible — bugs and vulnerabilities are found and patched quickly.

🔒 Strong Permission System

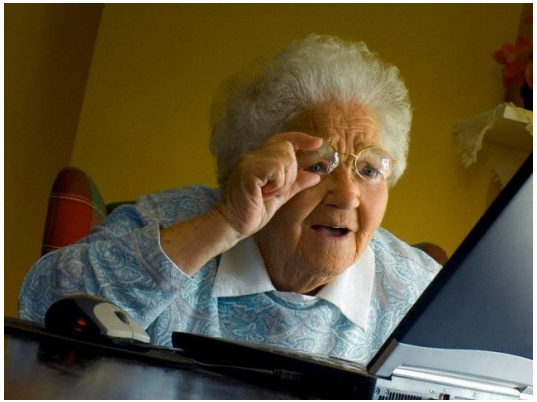
- Default non-root user access; limits damage from malware.

🔁 Fast Security Updates

- Frequent patches — security vulnerabilities are fixed much faster than in closed systems.

👨‍👩‍👧‍👦 Massive Community Monitoring

- Thousands of developers constantly reviewing and improving security.



📦 Minimal Default Installations

- Linux distros install only essential software — smaller attack surface.

🚪 Built-in Security Tools

- Built-in firewalls (iptables, nftables) and security modules (SELinux, AppArmor).

Feature	Linux	Windows	macOS
License	Free, Open Source	Paid License	Paid (Apple only)
Customization	Full	Limited	Very Limited
Security	Very High	Medium	High
Performance	Excellent (lightweight)	Good	Excellent (optimized)
Software	Good (less games)	Best (games/business)	Great (creative apps)
Ideal For	Servers, Devs, Hackers	Gaming, Office	Creatives, Designers

Using Linux for the first time:

After mastering Linux :

Using Windows for the first time:

After mastering Windows :

Linux

Open Source -Free

Windows

Closed License

- **Linux = Freedom + Control**
- **Windows = Office + Gaming**
- **macOS = Creative + Premium**

Distro	Purpose
Ubuntu	Beginner-friendly, desktops, servers
Debian	Stability-focused servers, base for others
Fedora	Latest features, developers, Red Hat ecosystem
CentOS / RedHat	Enterprise servers (stable, RHEL clone)
Arch Linux	Advanced users, full control, rolling release
Kali Linux	Penetration testing, cybersecurity
openSUSE	Developers, sysadmins, stable or rolling
Raspberry Pi OS	Lightweight for ARM devices and education

How is Linux evolved ?

🧱 1983 – GNU Project

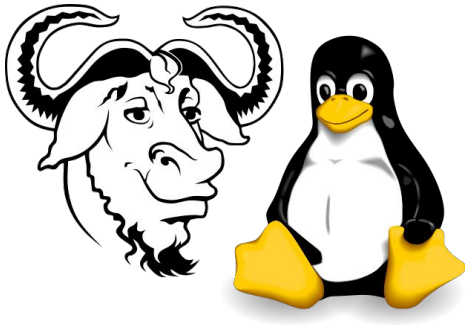
- Richard Stallman launched the **GNU Project** to build a free UNIX-like OS.
- Developed tools like GCC, bash, coreutils — but **no kernel** yet.

🧠 1991 – Linux Kernel

- Linus Torvalds developed the **Linux Kerne**
  as a free replacement for MINIX.
- Released under the GPL license.

🔗 1992 – GNU + Linux = GNU/Linux

- GNU tools + Linux kernel = complete OS.
- Birth of the **GNU/Linux operating system**.



https://upload.wikimedia.org/wikipedia/commons/1/1b/Linux_Distribution_Timeline.svg

🖥️ Linux Kernel Types

1. Monolithic Kernel

- All processes run in the same memory space.
- High speed but less modular.

2. Microkernel

- User and kernel services in separate address spaces.
- More modular but slower.

3. Exokernel - Re

- Manages hardware resources at the application level.
- Provides low-level access to hardware.

4. Hybrid Kernel

- Combines Monolithic and Microkernel features.
- Balances speed with modularity and stability.

- **Monolithic = Speed**
- **Microkernel = Modularity**
- **Exokernel = Low-level access**
- **Hybrid = Best of both worlds**

kernel.org

The Linux Kernel Archives

About Contact us FAQ Releases Signatures Site news

Protocol	Location
HTTP	https://www.kernel.org/pub/
GIT	https://git.kernel.org/
RSYNC	rsync://rsync.kernel.org/pub/

Latest Release
6.14.4

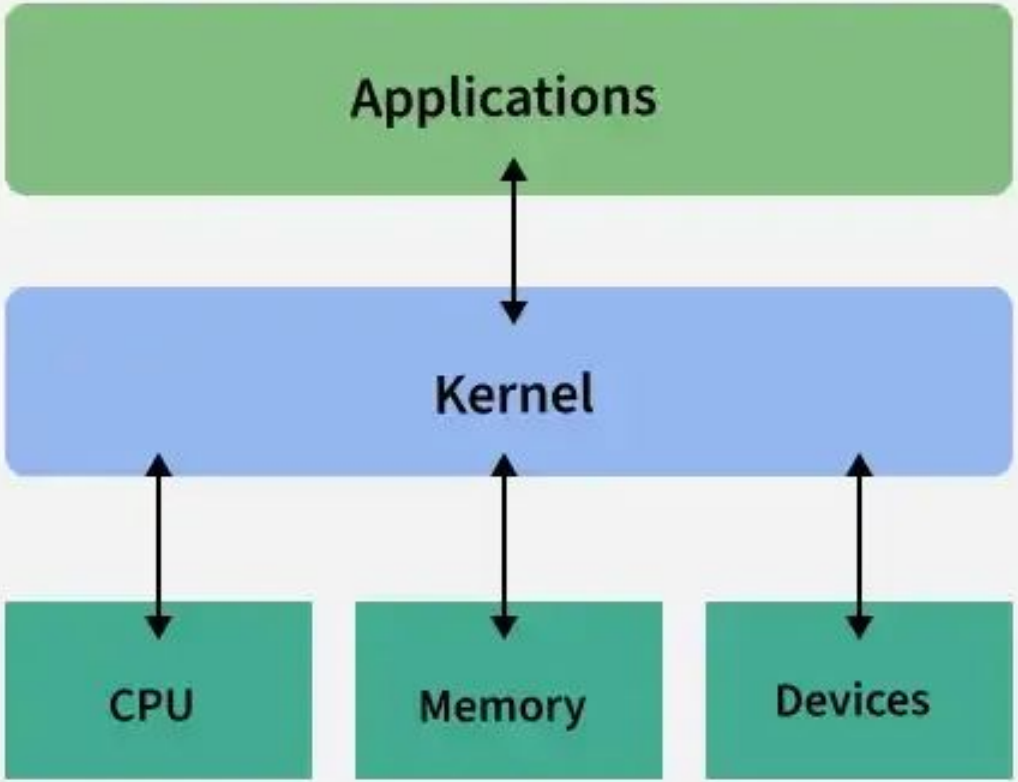
mainline:	6.15-rc4	2025-04-27	[tarball]		[patch]	[inc. patch]	[view diff]	[browse]
stable:	6.14.4	2025-04-25	[tarball]	[pgp]	[patch]	[inc. patch]	[view diff]	[browse] [changelog]
stable:	6.13.12 [EOL]	2025-04-20	[tarball]	[pgp]	[patch]	[inc. patch]	[view diff]	[browse] [changelog]
longterm:	6.12.25	2025-04-25	[tarball]	[pgp]	[patch]	[inc. patch]	[view diff]	[browse] [changelog]
longterm:	6.6.88	2025-04-25	[tarball]	[pgp]	[patch]	[inc. patch]	[view diff]	[browse] [changelog]
longterm:	6.1.135	2025-04-25	[tarball]	[pgp]	[patch]	[inc. patch]	[view diff]	[browse] [changelog]
longterm:	5.15.180	2025-04-10	[tarball]	[pgp]	[patch]	[inc. patch]	[view diff]	[browse] [changelog]
longterm:	5.10.236	2025-04-10	[tarball]	[pgp]	[patch]	[inc. patch]	[view diff]	[browse] [changelog]
longterm:	5.4.292	2025-04-10	[tarball]	[pgp]	[patch]	[inc. patch]	[view diff]	[browse] [changelog]
linux-next:	next-20250429	2025-04-29						[browse]

Other resources

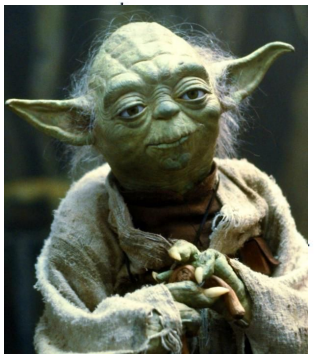
Git Trees
Patchwork
Mirrors
Documentation
Wikis
Linux.com
Kernel Mailing Lists
Bugzilla
Linux Foundation

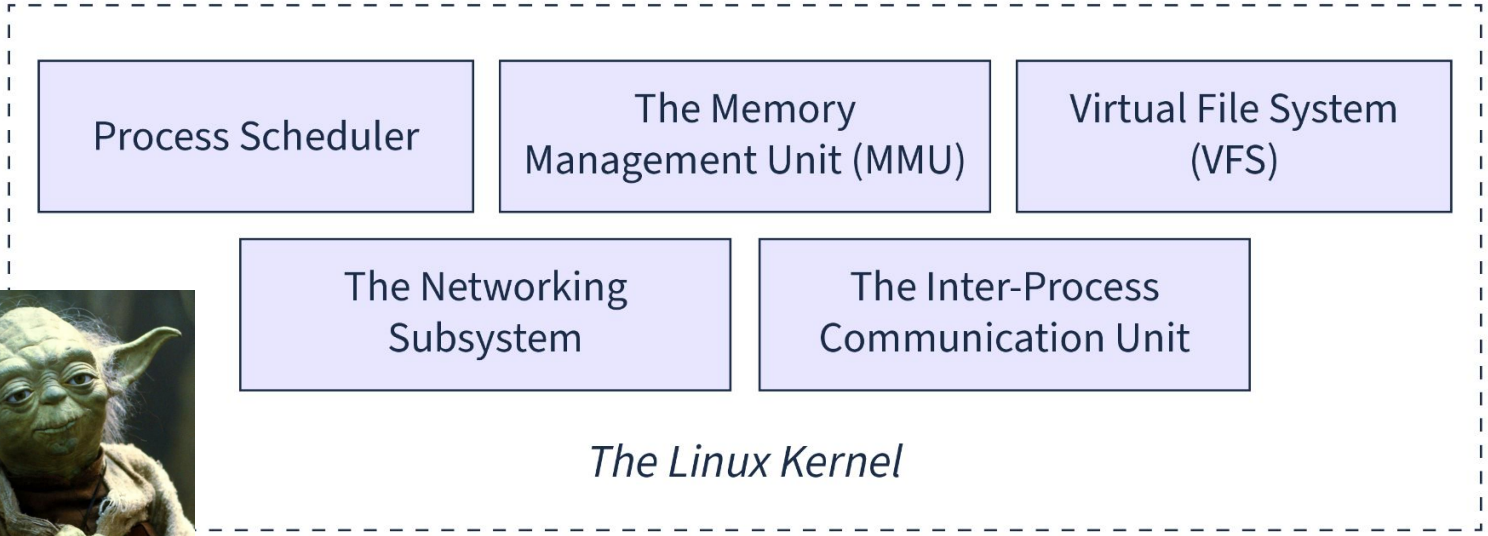
Social

Site Atom feed
Releases Atom Feed
Kernel Planet

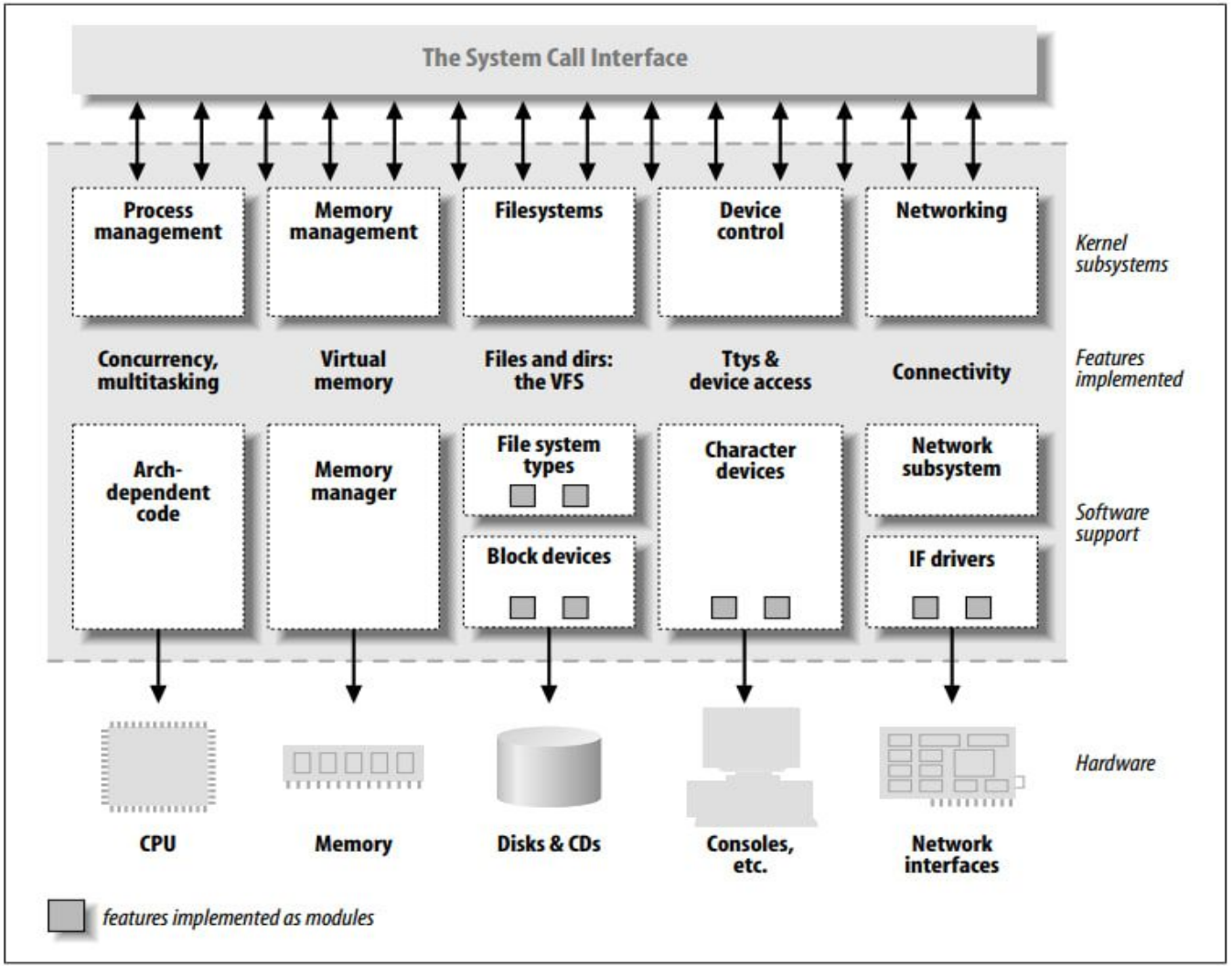

Applications
Kernel
CPU
Memory
Devices

The Linux Kernel Subsystems

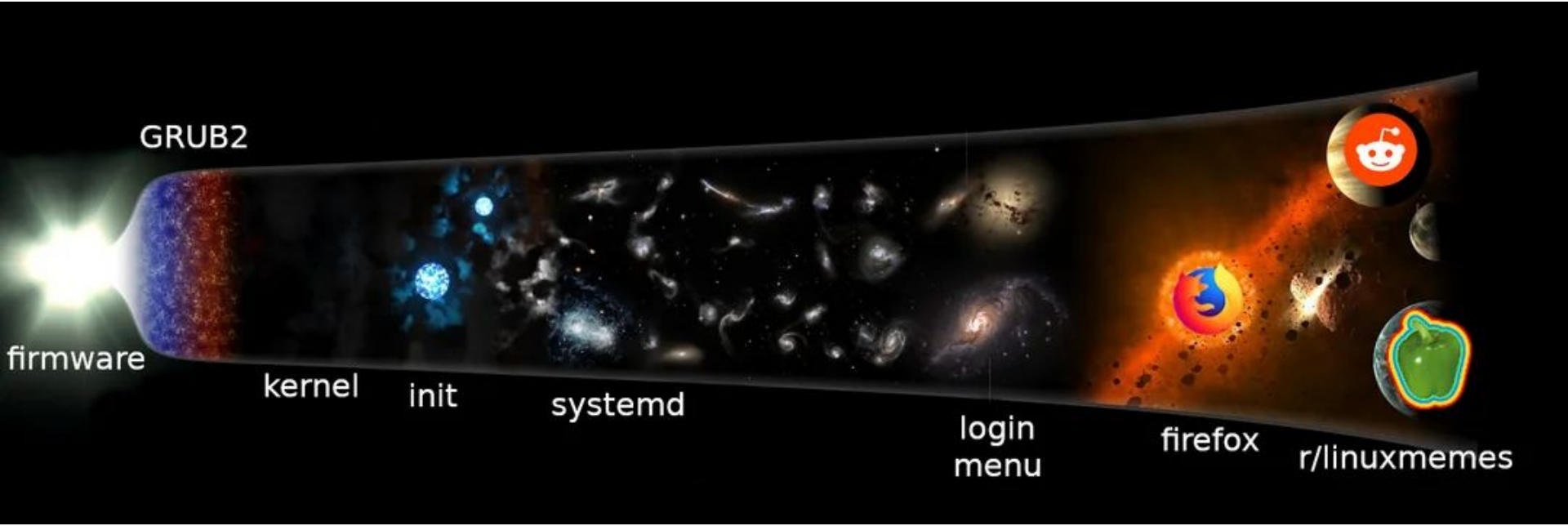




SCALER Topics


The System Call Interface
Process management
Memory management
Filesystems
Device control
Networking
Kernel subsystems
Concurrency, multitasking
Virtual memory
Files and dirs: the VFS
Ttys & device access
Connectivity
Features implemented
Arch-dependent code
Memory manager
File system types
Block devices
Character devices
Network subsystem
IF drivers
Software support
CPU
Memory
Disks & CDs
Consoles, etc.
Network interfaces
Hardware
features implemented as modules

Boot Process



The file system hierarchy

All files on a Linux system are stored on file systems which are organized into a single *inverted* tree of directories, known as a *file system hierarchy*. This tree is inverted because the root of the tree is said to be at the *top* of the hierarchy, and the branches of directories and subdirectories stretch *below* the root.

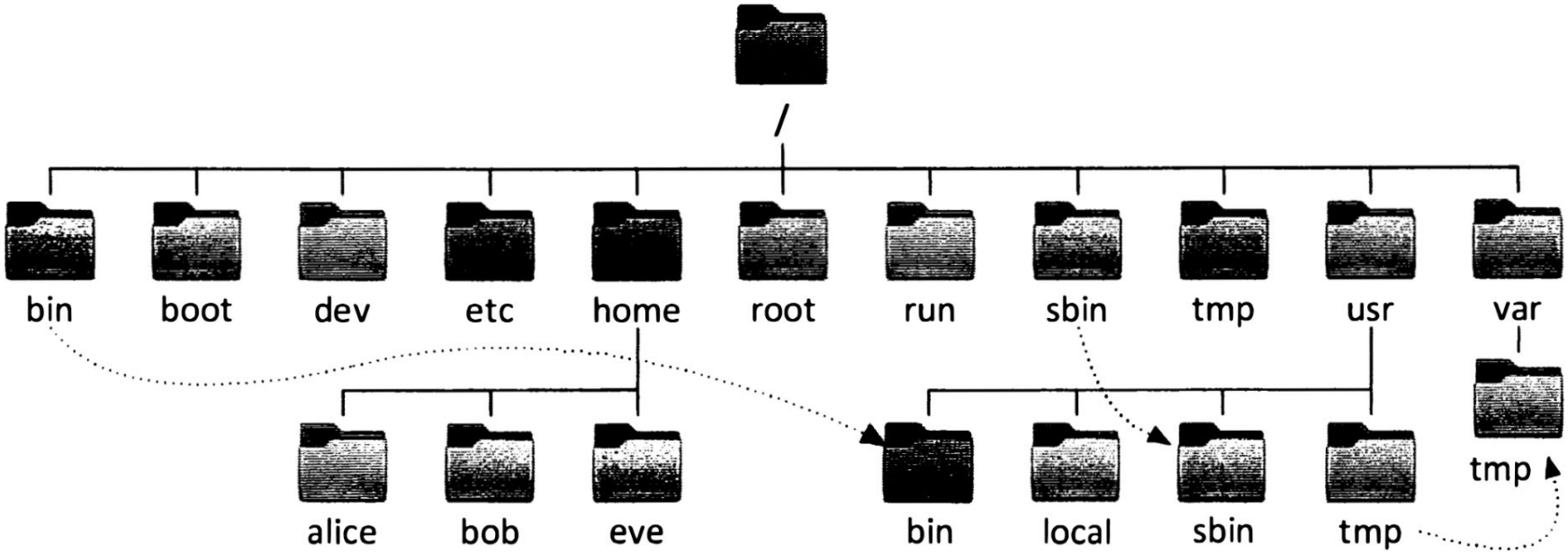
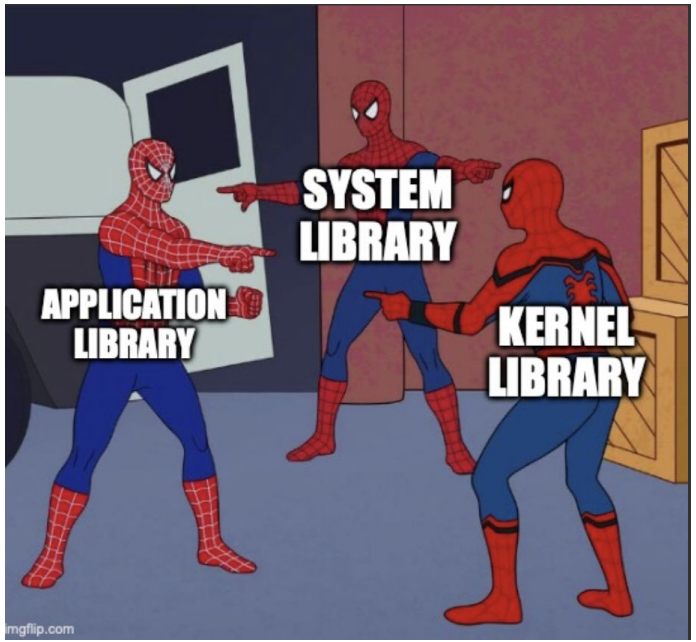


Figure 2.1: Significant file system directories in Red Hat Enterprise Linux 7

System Library:

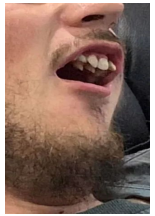
System libraries are some predefined functions by using which any application programs or system utilities can access kernel's features. These libraries are **the foundation upon which any software can be built.**

1. **GNU C library**: This is the C library that provides the most fundamental system for the interface and execution of C programs. This provides may in-built functions for the execution.
2. **libpthread (POSIX Threads):** This library plays important role for multithreading in Linux, it allows users for creating and managing multiple threads.
3. **libdl (Dynamic Linker):** This library is responsible for the loading and linking file at the runtime.
4. **libm (Math Library):** This library provides user with all kind of mathematical function and their execution.



Some other system libraries are: librt (Realtime Library), libcrypt (Cryptographic Library), libnss (Name Service Switch Library), libstdc++ (C++ Standard Library)

Shell:



can be determined **as the interface to the kernel,** which hides the internal execution of functions of kernel from the user. Users can just enter the commend and using the kernel's function that specific task is performed accordingly.

Different types of shell:

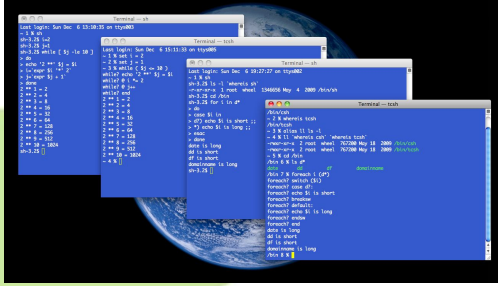
3.1. Command Line shell:

command-line shell is a text-based interface that allows users to execute commands by typing them. The shell interprets and processes these commands, then displays the output in the terminal. Examples include bash and zsh.

3.2. Graphical User Interface: (XFCE, X11, WAYLAND, GNOME, KDE)

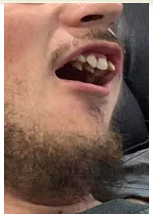
Executes the process provided by user in graphical way and output is displayed in the graphical window.

Different Shells in Linux



Bourne Shell

The Bourne Shell, created by Stephen Bourne in the 1970s, is the original Unix shell and the foundation for many modern shells.



Bash Shell

Bash is the most commonly used shell and the default on most Linux distributions.

C Shell

The C Shell, developed by Bill Joy in the late 1970s, introduced several interactive features.

KornShell

KornShell combines features from the Bourne Shell and C Shell, offering advanced scripting capabilities.

Absolute paths and relative paths

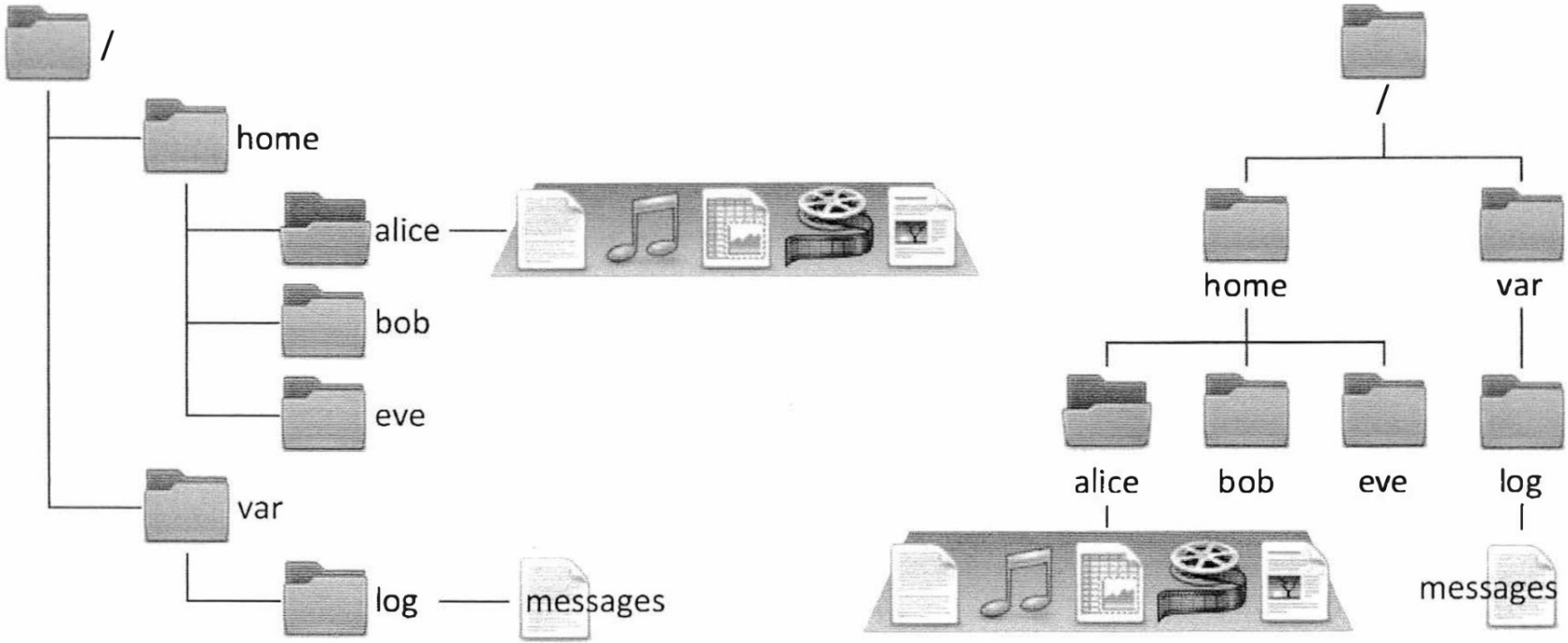
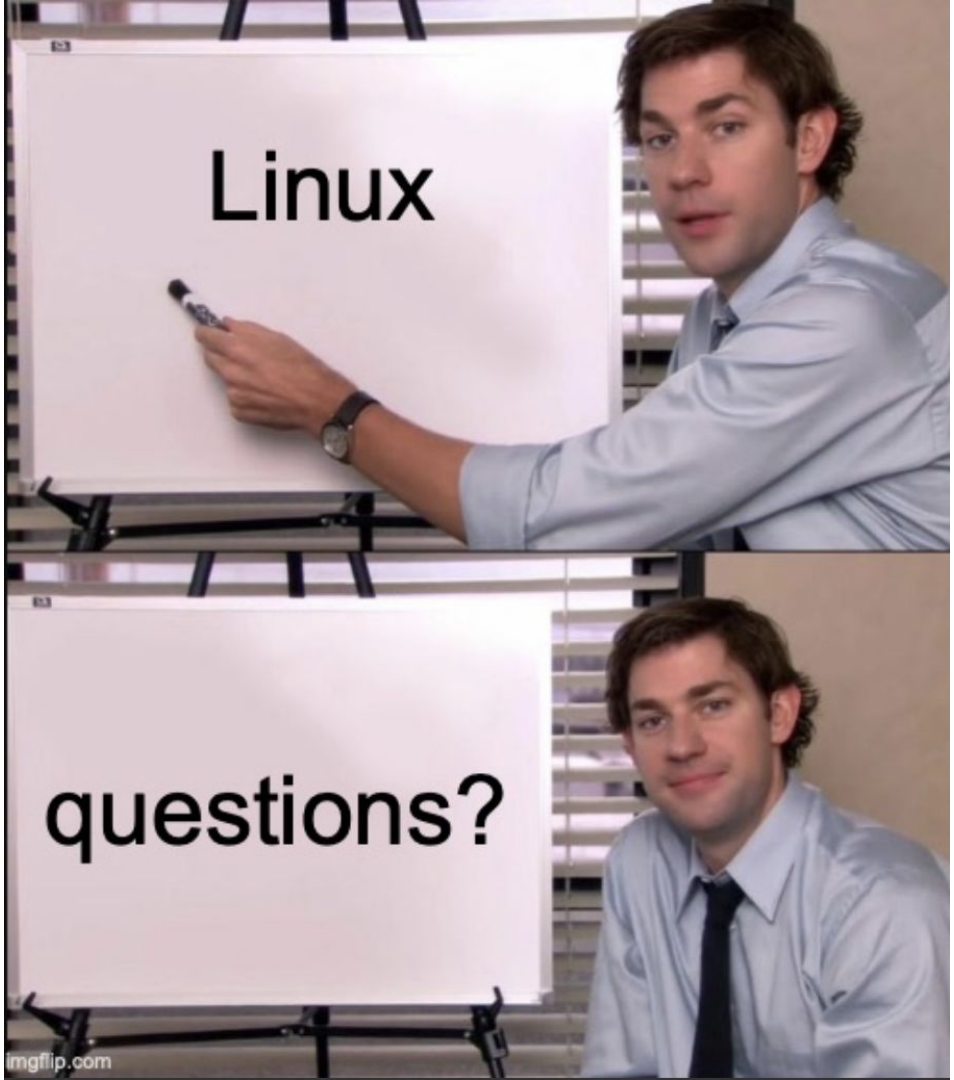


Figure 2.2: The common file browser view (left) is equivalent to the top-down view (right)


Linux
questions?
imgflip.com