History of Linux and the command line

Zubin

January 25, 2026

Contents

1 Operating Systems: Definition and Services **3**
1.1 What is an Operating System? . . . 3
1.2 Examples of Operating Systems . . . 3
1.3 Unix and Linux . . . 3
1.4 Core Services Provided by an Operating System . . . 3
1.4.1 File Management . . . 3
1.4.2 Memory Management . . . 3
1.4.3 Process Management . . . 3
1.4.4 Input/Output Management . . . 4

2 Genesis of Operating Systems **4**
2.1 Early Computers (Mid-1940s) . . . 4
2.2 Programming and Operation . . . 4
2.3 Invention of the Transistor . . . 4
2.4 Punch Cards and Role Separation . . . 4
2.5 Birth of Operating Systems . . . 4

3 UNIX Genesis **5**
3.1 Technological Context . . . 5
3.2 Project MAC at MIT . . . 5
3.3 MULTICS . . . 5
3.4 Challenges of MULTICS . . . 5
3.5 Birth of UNIX . . . 5
3.6 Evolution of UNIX . . . 6
3.7 The C Programming Language . . . 6
3.8 Spread of UNIX . . . 6
3.9 Key Milestones . . . 6
3.10 Legacy of UNIX . . . 6

4 Linux genesis and history: GNU, Stallman, GPL, Linus Torvalds, Linux **7**
4.1 The GNU Project . . . 7

4.2 Richard Stallman and the GPL 7
4.3 Linus Torvalds and the Linux Kernel 7
4.4 Integration and Growth 7
4.5 Widespread Adoption 8

5 Command line interface, prompt, command options and files data, command cal as example 8
5.1 Definition and Interaction 8
5.2 The Command Prompt 8
5.3 Command Structure 8
5.4 Example: The `cal` Command 9

1 Operating Systems: Definition and Services

1.1 What is an Operating System?

An **operating system** (**OS**) is an intermediary between computer hardware (memory, processor, network cards, etc.) and the applications that users interact with.

- Users interact with applications.
- Applications request services from the operating system.
- The operating system manages and exploits hardware resources to provide services to the applications.

1.2 Examples of Operating Systems

- Microsoft Windows
- Apple macOS and iOS
- Google Android
- Unix and Unix-like systems such as Linux

1.3 Unix and Linux

- Unix has been around longer than Linux.
- Linux is a **Unix-like, open-source operating system**. The Linux kernel, created by **Linus Torvalds** and expanded upon by thousands of programmers, is available to the world for free.

1.4 Core Services Provided by an Operating System

1.4.1 File Management

- Managing the logical tree structure of files and their physical layout on storage devices (hard drives).

1.4.2 Memory Management

- Allocation, deallocation, and sharing of memory among multiple running processes.

1.4.3 Process Management

- Creation, execution, and termination of running applications (processes).

1.4.4 Input/Output Management

- Managing hardware like network interfaces, sound cards, video cards, printers, and other peripherals.

---

2 Genesis of Operating Systems

2.1 Early Computers (Mid-1940s)

- The first computers were built using **vacuum tubes** (evacuated glass containers that control electric current).
- These were huge machines that filled entire rooms but performed more slowly than a modern hand-held calculator.

2.2 Programming and Operation

- Programming was done manually by rearranging hardware components.
- Input/output capabilities were very limited.
- A single individual often acted as the designer, builder, programmer, and operator.

2.3 Invention of the Transistor

- The invention of the transistor led to smaller, more reliable computers.
- This innovation marked the beginning of operating systems through the appearance of **punch cards**.

2.4 Punch Cards and Role Separation

- Punch cards are cards with holes in specific locations to encode computer programs and data.
- This led to a separation of roles: programmers prepared the punch cards, and operators physically loaded them into the computer and handled the output.

2.5 Birth of Operating Systems

- Operating systems were invented to manage memory, processes (running programs), and input/output operations like reading punch cards.
- We can date the invention of operating systems to the **mid-1960s**.

---

3 UNIX Genesis

3.1 Technological Context

- The era of modern computers emerged with the appearance of **integrated circuits** and magnetic disks.
- This period also saw the development of compatible computer families, such as the **IBM System/360** (1964), which made a clear distinction between architecture and implementation.

3.2 Project MAC at MIT

- It all started with **Project MAC** (Mathematics and Computation), founded at MIT.
- It was funded by the US military's research agency (ARPA) and the National Science Foundation.
- The main goal was to develop a **timesharing system** that would allow a large community of users to access a single computer from multiple locations simultaneously.

3.3 MULTICS

- Developed by MIT, Bell Labs, and General Electric
- Stands for **Multiplexed Information and Computing Service**
- It evolved beyond timesharing to incorporate features like file sharing, file management, and system security.

3.4 Challenges of MULTICS

- The project proved much more difficult than expected.
- The system became operational in 1969 on the GE-645 computer, but its performance was far below the original targets.
- As a result, Bell Labs withdrew from the project in 1969.

3.5 Birth of UNIX

- Following the withdrawal, Bell Labs engineers **Ken Thompson** and **Dennis Ritchie** decided to create a simpler, minimal system.
- Using a little-used DEC PDP-7 machine, they began developing a single-user operating system.
- As a pun on the complexity of MULTICS, they called their system **UNICS**.

3.6 Evolution of UNIX

- In 1970, the system was enhanced to support multiple users, and its name morphed to **Unix**.
- At the time, the system was written in the **B programming language**, which was invented by Ken Thompson.

3.7 The C Programming Language

- In 1971, Dennis Ritchie improved upon B and called it **New B**.
- By 1972, the changes were so significant that Ritchie renamed his new language the **C programming language**.
- Ken Thompson then rewrote the entire Unix operating system in C.

3.8 Spread of UNIX

- The C source code for Unix was distributed to universities and research centers for educational purposes.
- From 1975 onward, a very active community emerged around Unix and C.
- Other notable developers included:
  - Douglas McIlroy (McElroy)
  - Joseph Ossanna
  - Rudd Canaday

3.9 Key Milestones

- 1978: Brian Kernighan and Dennis Ritchie published the book *The C Programming Language*.
- 1983: Thompson and Ritchie received the **Turing Award**, the highest distinction in computer science, for their invention.

3.10 Legacy of UNIX

- The concepts introduced by Unix are ubiquitous today and form the foundation for many modern operating systems.
- Derivatives of Unix include:
  - macOS
  - iOS
  - Android

– Linux, which is installed on the vast majority of today's servers and connected objects.

---

4 Linux genesis and history: GNU, Stallman, GPL, Linus Torvalds, Linux

4.1 The GNU Project

- In 1983, the same year Ritchie and Thompson received the Turing Award, **Richard Stallman** of MIT launched the **GNU Project**.
- GNU is a recursive acronym standing for **GNU is Not Unix**.
- The project aimed to develop a free, open, and collaborative software system compatible with Unix, contrasting with the proprietary nature of Unix owned by Bell Labs.

4.2 Richard Stallman and the GPL

- Richard Stallman is a strong advocate for free and open-source software.
- In 1989, he conceived the **GNU General Public License** (**GPL**), designed to preserve the freedom to use, study, modify, and distribute software.
- By 1990, the GNU Project had created many tools (text editors, GUI, libraries, and the **GNU C Compiler** (**GCC**)), but it lacked a free operating system kernel to run them on.

4.3 Linus Torvalds and the Linux Kernel

- **Linus Torvalds**, a student at the University of Helsinki, was frustrated by proprietary OS licenses.
- On August 25, 1991, he announced a "free operating system" project (just a hobby) to the community.
- This became the **Linux kernel**, developed on an 80386 processor using the GNU C Compiler.

4.4 Integration and Growth

- A community formed quickly, integrating GNU software with the Linux kernel.
- In 1992, the first **Linux distributions** were released (Linux kernel + GNU tools).

- By 1993, there were over 100 developers, and popular distributions like **Debian** emerged.

4.5 Widespread Adoption

- **Late 1990s:** Major manufacturers (Dell, IBM, HP) announced Linux compatibility.
- **2000s:** Increasing deployment on web servers.
- **2010s:** Linux and Unix-based systems dominated:
  - Internet servers (70%)
  - Smartphones (90%, via Android and iOS)
  - Supercomputers (99%)
- Linux is also prevalent in game consoles, routers, and IoT devices.

---

5 Command line interface, prompt, command options and files data, command cal as example

5.1 Definition and Interaction

- A **Command Line Interface** (**CLI**) is a human-machine interface where communication takes place in text mode.
- The user types a command to request an operation, and the computer displays the result or further questions in text.
- It is central to the interaction between users and computing equipment.

5.2 The Command Prompt

- When ready to receive input, the system displays a **command prompt**.
- This prompt usually contains information like the user's name, computer name, current directory, or date.
- It ends with a character such as `$`, `#`, or `>`.

5.3 Command Structure

- Unix and Linux systems include hundreds of simple applications usable from the CLI.
- The basic structure of a command is: `command options files_or_data`

- **Command:** The name of the application (often written in C).
- **Options:** Modify the command's execution (separated by spaces).
- **Files or Data:** The inputs for the program.

5.4 Example: The `cal` Command

- Typing `cal` and pressing Enter runs the application that displays a calendar.
- Adding the `-j` option (e.g., `cal -j`) displays the calendar in Julian days (number of days elapsed since January 1st).
