

OPERATING SYSTEMS

Free and Open-Source Operating Systems

أنظمة التشغيل الحرة ومفتوحة المصدر

From a closed black box to an open teaching laboratory

PRESENTED BY

Prof. Mostafa Elhosseini

Professor of Smart Systems Engineering

What If We Get the Source?

What changes when an OS comes not only as a finished program, but as **source code** we can...

Read

Study

Modify

Compile

Redistribute

THE STORY

Source Code



Free Software



GNU



Linux



BSD



Solaris /
Darwin

Open systems as a **laboratory** for learning Operating Systems

PART 1

Core Concepts

Source, open, free, closed, and hybrid

Source Code vs Binary Code

Source Code

الشفرة المصدرية

Human-readable — statements, names, structure, comments

Binary Code

الشفرة الثنائية

Machine-readable — produced by compilation

Source Code



Compiler



Binary Code



CPU executes

Binary → Source = **Reverse Engineering** · logic approximated, **comments lost**

الهندسة العكسية

Source = the **recipe**

Binary = the finished **cake**

What Does “Open Source” Mean?

The **source code** is available to inspect.

البرمجيات مفتوحة المصدر

NOT ONLY

`program.exe`

BUT ALSO

the **actual code** it was built from

IN AN OS, YOU CAN SEE HOW

Scheduling

Memory

Files

Devices

Sync

REMEMBER **Black box** صندوق أسود → something we can not open ·
Open Source = I can see inside.

Free Software vs Open-Source

Free Software

البرمجيات الحرة

Free = **Freedom** الحرية, not zero price

Focus: **legal freedoms** granted by specific licenses — “Free/Libre”

Open Source

The **source code is available**

Focus: code **available for inspection** — not necessarily the same licensing philosophy

ANALOGY · A RESTAURANT

Closed

Eat the meal — **recipe secret**

Open

Chef **gives the recipe**

Free

Recipe + permission to **use, study, modify, share**

REMEMBER

Open = see the code · **Free** = freedom with the code

Open, Closed, and Hybrid

Open

Source available

GNU/Linux

Closed

مغلق المصدر

Proprietary برمجيات مملوكة vendor
keeps source private

Binary only

Hybrid

هجين

Open foundation + proprietary
parts

macOS (Darwin)

ANALOGY · HOUSES

Open → inspect the plans (المخططات)

Closed → plans locked

Hybrid → some plans public

REMEMBER Open = visible code · Closed = protected code · Hybrid = mixture

Why Open Source Matters for Learning

BEFORE

“How does the scheduler choose a process?”
→ documentation + observation

WITH OPEN SOURCE

“Show me the scheduler code.”

Read**Modify****Compile****Run****Observe**

+ **Virtualization:** several OSs on one machine — no separate computer each

Without source: drive the car, observe

With source: open the engine, replace, test

REMEMBER

Open-source OS = an OS you can **learn by opening it**

Benefits of Open-Source Development

Community

المجتمع

Write, debug, analyze, support,
suggest — **thousands of eyes**

Security argument

حجة الأمان

More eyes → bugs **may** be
found & fixed faster

Commercial viability

الجدوى التجارية

Revenue from **support,**
services, hardware

Not: “Open source has **no bugs.**” ✗

But: more inspectors → bugs fixed **sooner** ✓

REMEMBER Free to inspect — and still **commercially valuable**

PART 2

History, GNU & Linux

How software closed — and how it reopened

How Did Software Become Closed?

1950s

Software came **with source** — sharing normal (MIT hackers)

1960s

User groups (**DECUS**) collect programs, share them on tapes

1970s

Digital OS shipped as source — no copyright limits

1980s

Vendors ship **compiled binaries** only — protect IP

Shared source



Proprietary, binary-only

Early: engineers sharing notebooks

مثلما يتشاركون دفاترهم وملاحظاتهم

Later: use my machine — **no drawings**

لن أعطيك التصميمات أو التفاصيل الداخلية

GNU — “GNU’s Not Unix!”

1984

Richard Stallman starts GNU — a free, Unix-compatible OS

Recursive acronym `GNU's Not Unix!`
اختصار متكرر ذاتيًا

1985

GNU Manifesto

Free Software Foundation (FSF)

مؤسسة البرمجيات الحرة

GNU is a project started by Richard Stallman in 1984 to build a free, Unix-compatible operating system.

Free ≠ zero price

Free = **Freedom** — you may even sell a copy

REMEMBER GNU = a Unix-like system built around **software freedom**

The Four Essential Freedoms

FREEDOM 0

Run — for any purpose

“I can use it.”

FREEDOM 1

Study & change the source

“I can understand and modify it.”

FREEDOM 2

Share unchanged copies

“I can share the original too.”

FREEDOM 3

Modify & share copies

“I can share my improved version.”

RUN



STUDY



MODIFY



SHARE

REMEMBER Free Software = Run it, Study it, Change it, Share it

Copyleft

Copyright

حقوق النشر

“Certain rights are **reserved**.”

Copyleft

النسخ المتروك

“Use, change, redistribute — but **preserve the same freedoms** .”

FREEDOM PROPAGATES FORWARD

A gives code



B modifies



C gets the same freedoms

ANALOGY · A SPECIAL BOOK

Copy it, edit it, give it away — **whoever receives it gets the same rights.**

REMEMBER Copyleft = you get freedom → **you must pass it on**

GPL — GNU General Public License

GPL stands for: GNU General Public License
It is a software license created for free software.

رخصة تضمن استمرار حرية البرنامج.

Redistribute GPL software → **source must accompany the binary** ; modified versions stay under the GPL (**ShareAlike**).

Original GPL code

↓ modify

Your version

↓ redistribute

GPL freedoms continue

REMEMBER

Goal: no one can take free code, modify it, and **close it**

GNU Had Almost Everything...

BY ~1991, GNU HAD

Compilers ✓

Editors ✓

Utilities ✓

Libraries ✓

Games ✓

...but the **Kernel** was not ready ✗

النواة

ANALOGY · A CAR

Seats ✓ · dashboard ✓ · tools ✓ · lights ✓ — but the **engine is missing**.

REMEMBER GNU had the tools; it still **needed the kernel**

Linux Supplies the Missing Kernel

1991

Linus Torvalds , student in Finland — simple Unix-like kernel, built with GNU tools; invites contributors

1992

Re-released under the **GPL** — free software

Linux = Kernel

GNU/Linux = Linux + GNU tools + other software

Linux = **engine**

GNU tools = **car systems**

Distribution = **whole car**

REMEMBER GNU brought the **tools**; Linux brought the **kernel**

PART 3

The Linux Ecosystem

Distributions, virtual labs, and version control

Linux Distributions

Ubuntu

Debian

Fedora

Red Hat

SUSE

Slackware

A **Distribution** = Linux kernel + everything needed for a usable system

توزيعة

Utilities

Libraries

Applications

Package mgmt

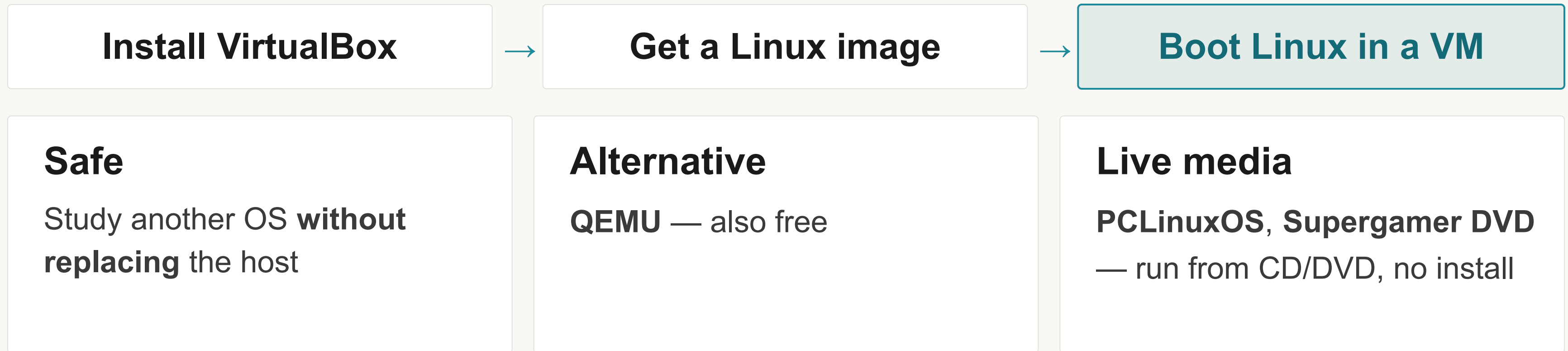
User interfaces

Hardware support

Default config

REMEMBER Linux = **kernel** · Distribution = **complete packaged system** — same core, different purpose

Learning Linux with VirtualBox



A Virtual Machine = a laboratory inside your computer

آلة افتراضية

REMEMBER Virtualization = a **safe OS laboratory** inside your computer

Version Control

Manages changes from **many developers** — history + conflict handling. نظام التحكم في الإصدارات

Pull

سحب

Repository → Developer
Bring changes to me

Push

دفع

Developer → Repository
Send my changes back

ANALOGY · A SHARED MANUSCRIPT

Download → edit → upload · it remembers **who changed what, and when.**

TOOLS

Subversion**Git**

PART 4

Other Open Lineages

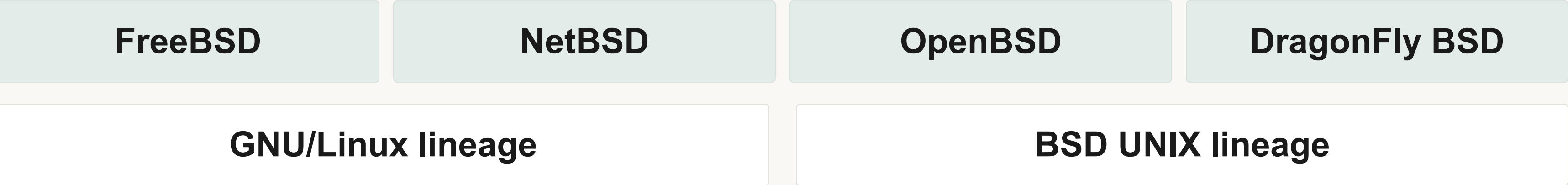
BSD, Darwin, Solaris and Illumos

BSD UNIX

Berkeley Software Distribution — derived from AT&T UNIX, late 1970s



MODERN DESCENDANTS



REMEMBER Both are open Unix-like families — with **different histories**

Darwin and macOS

Darwin

Open-source core of macOS **BSD UNIX** ancestry

Kernel: [XNU](#)

macOS = Hybrid

هجين

Open components + proprietary Apple components

ANALOGY · A BUILDING

Foundation plans are **public** — many rooms and furnishings stay **private**.

REMEMBER macOS = **open foundations + proprietary components**

Solaris, OpenSolaris, Illumos

Solaris

Sun's commercial UNIX SunOS (BSD) → **System V**

OpenSolaris

2005 — much Solaris code released

Illumos

After Oracle bought Sun — continues the **open branch**

Solaris



OpenSolaris



Illumos

No need to memorize every detail — keep the **big picture**.

REMEMBER Illumos continues an **open branch** of the Solaris family

Excellent Learning Tools

Read **real OS code** — not only *about* OS code.

Inspect

Modify

Compile

Test

Break

Debug

Fix

Historic systems

Study **Multics** from source — see how ideas evolved

Cross-learning

Projects share code & ideas when **licensing allows**

Anatomy **textbook** — useful

A real model you can **open** — far more powerful

One Story

Source Code	we see how software is built
Open Source	we can inspect the code
Free Software	freedom to use, study, modify, share
GNU + GPL	tools + copyleft philosophy
Linux Kernel	GNU/Linux becomes a complete system
BSD · Darwin · Solaris · Illumos	other open Unix-like lineages
Virtualization + Source	OS becomes something we experiment with

Four Sentences to Remember

HOW

Source code tells us **how** the software works.

SEE

Open source lets us **see** it.

FREEDOM

Free software gives us **freedom** to use, study, change, and share it.

LEARN

Open-source OSs let us **learn** OS concepts from real code.

Open source turns Operating Systems from **theory** into a **laboratory**.