k1.1 — Distributions & Families: What “Linux” Is (and Isn’t) ?

What people mean when they say “Linux”

In everyday dev/admin talk, “Linux” usually refers to an *entire operating system stack*, not just one component:

1. **Linux kernel**
   The core that handles:
   - processes and scheduling
   - memory management
   - networking
   - hardware drivers
2. **Userland tools**
   The standard command-line tools you’ll use constantly, such as:
   - `ls`, `cp`, `mv`, `rm`, `find`
   - `grep`, `sed`, `awk`
   - `tar`, `ssh`, `curl`
3. **Init / service manager**
   On most modern distros, this is **systemd** (so services behave similarly across many distributions).
4. **Package manager + repositories**
   How software is installed and updated (this is one of the biggest practical differences across distros).
5. **Defaults & policies**
   Security frameworks, filesystem defaults, preinstalled tools, and “the distro’s way” of doing common tasks.

What a “distribution” actually is

A **Linux distribution (distro)** is a curated bundle of the components above, plus:

- a release philosophy (stable vs fast-moving)
- a packaging system and official repositories
- default configuration choices
- documentation/community norms

So you’re not learning a completely different OS each time—you’re learning **the same core concepts** with different “packaging and defaults.”

The main families you’ll encounter (server/web world)

For web development and WordPress hosting, these families matter most:

1. **Debian family**
   - **Debian** (very stability-focused, conservative updates)
   - **Ubuntu** (based on Debian; very common, especially **Ubuntu LTS** on servers)
   - Typical traits:
     - uses **APT** (`apt`)
     - huge ecosystem of tutorials and packages
2. **RHEL family** (Red Hat Enterprise Linux–compatible)
   - **Rocky Linux**, **AlmaLinux** (common for enterprise-style servers and some tooling stacks)
   - Typical traits:
     - uses **DNF** (`dnf`, historically `yum`)
     - **SELinux** is commonly enabled and relevant

What’s shared across most distros (the “learn once” core)

If you learn these well, you can function on almost any mainstream Linux server:

- filesystem concepts and common directories (you’ll cover this in **k3**)
- permissions and ownership (`chmod`, `chown`)
- processes and signals (`ps`, `kill`)
- service management concepts (often via `systemctl`)
- networking basics (ports, DNS, `curl`, `ss`)
- SSH and key-based access

What this means for your course goal

Your plan—**learn cross-distro fundamentals first without committing**—works because:

- the *core Linux skills* transfer extremely well
- later, you mainly “translate”:

- `apt` ↔ `dnf`
- AppArmor ↔ SELinux
- UFW ↔ firewalld
- slightly different defaults/paths

Tiny self-check ?

Answer these without Googling (it’s fine if you can’t yet):

1. What’s the **kernel** responsible for (in one sentence)?
2. Name **two things** that are usually different between Debian-based and RHEL-based distros.
3. What’s one Linux skill that should transfer across almost all distros?

If you want, reply with your answers and I’ll correct/clarify briefly.

Revision #2
Created 2026-04-04 23:22:10 UTC by art10m
Updated 2026-04-05 00:38:28 UTC by art10m