Linux Kernel Source Tree Structure (Grouped by Function)
---
1. Core System
/init/ – Contains initialization code (boot sequence setup, kernel start).
/kernel/ – Core kernel logic: process management, scheduling, signals, timers, etc.
/mm/ – Memory management: paging, virtual memory, slab allocator, etc.
/ipc/ – Interprocess communication: semaphores, shared memory, message queues.
/certs/ – Kernel certificate handling for module signature verification.
---
2. Hardware and Architecture Support
/arch/ – Architecture-specific code (x86, ARM, RISC-V, etc.).
Each subfolder has its own startup, interrupt handling, and memory setup code.
/drivers/ – Device drivers (GPU, sound, USB, network, filesystem, etc.).
/firmware/ – Binary firmware blobs for hardware requiring preloaded firmware.
/block/ – Block layer implementation (used by filesystems, I/O scheduling).
/sound/ – Audio subsystem (ALSA).
/gpu/ – GPU subsystem (DRM, display management).
---
3. File Systems and Storage
/fs/ – All filesystem implementations (ext4, FAT, NTFS, etc.) and VFS layer.
/fs/proc/ – Implements `/proc` virtual filesystem for process/system info.
/fs/sysfs/ – Implements `/sys` interface for kernel objects and drivers.
/fs/nfs/ – Network filesystem implementation.
---
4. Networking Stack
/net/ – Entire networking stack (IPv4, IPv6, TCP, UDP, routing, etc.).
Submodules: `/net/core/`, `/net/ipv4/`, `/net/ipv6/`, `/net/sched/`, `/net/bridge/`.
/net/wireless/ – Wi-Fi and RF management layers.
/net/ethernet/ – Ethernet-specific handling.
---
5. User-space Interface and System Calls
/include/ – Header files defining interfaces for both kernel and user-space.
/uapi/ – User-space API headers exposed via `/usr/include/linux/`.
/syscalls/ – System call tables and definitions.
---
6. Security and Access Control
/security/ – LSM (Linux Security Modules) implementations: SELinux, AppArmor, etc.
/keys/ – Key management for encryption and authentication.
/crypto/ – Cryptography algorithms and frameworks (AES, SHA, RSA).
---
7. Utilities, Tools, and Documentation
/scripts/ – Build scripts, kernel configuration utilities, and automation tools.
/tools/ – User-space utilities for performance testing and tracing (perf, bpftool).
/samples/ – Example modules and BPF samples.
/Documentation/ – All developer documentation (moved to `/Documentation/` or `docs/`).
/docs/ – ReST-formatted documentation (built using Sphinx).
---
8. Build System and Configuration
/Makefile – Top-level Makefile for kernel compilation.
/Kconfig – Configuration options defining kernel features.
/scripts/kconfig/ – Logic for menuconfig and other config utilities.
---
9. Virtualization and Containers
/virt/ – Virtualization support (KVM, Xen, etc.).
/drivers/virtio/ – Virtual device drivers.
/cgroup/ – Control groups for process resource isolation (CPU, memory, I/O).
---

10. Other Subsystems
/lib/ – Generic library functions (CRC, string ops, bitmaps, etc.).
/trace/ – Kernel tracing and instrumentation support.
/bpf/ – eBPF subsystem for programmable kernel instrumentation.
/usr/ – Initramfs generation tools.
/scripts/ – Various automation utilities for building and maintenance.

■ **Summary:**
The Linux kernel is modular, with clear separation between hardware interfaces (`drivers/`, `arch/`), system core (`kernel/`, `mm/`), user-facing APIs (`fs/`, `net/`, `include/`), and utilities (`scripts/`, `tools/`).