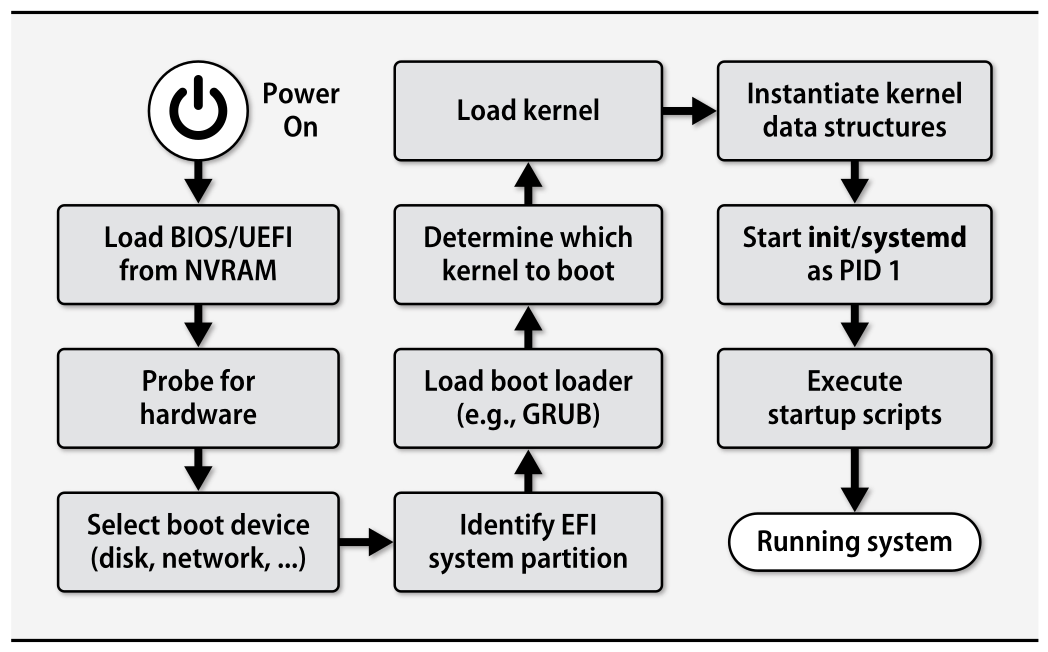
**Booting and System Management Daemons**

**Booting Process Overview**



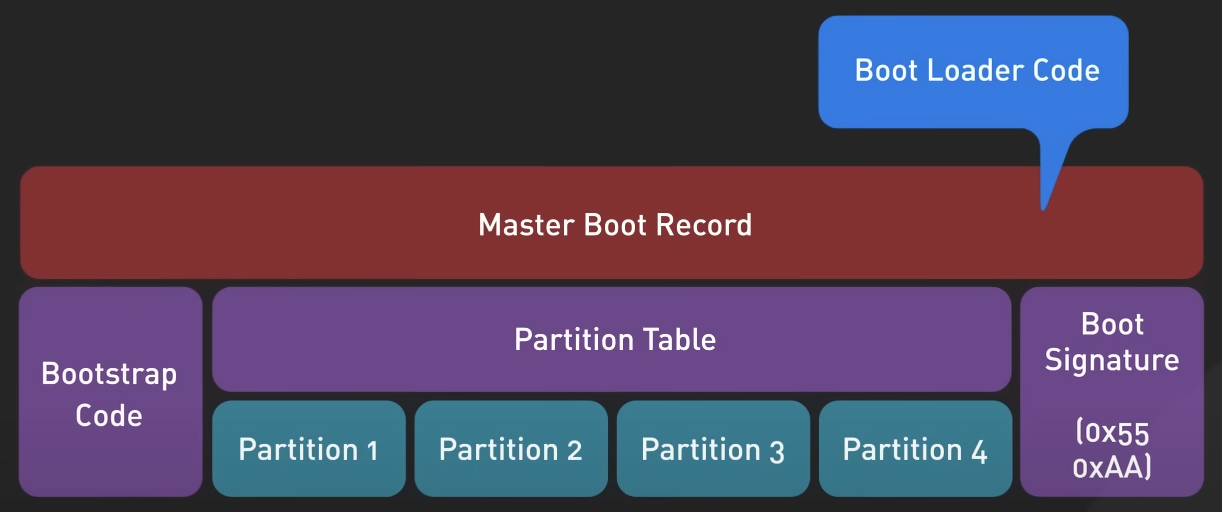
The booting process consists of a few broadly defined tasks:

1. Finding, loading, and running bootstrapping code.
2. Finding, loading, and running the OS kernel.
3. Running startup scripts and system daemons.
4. Maintaining process hygiene and managing system state transitions.

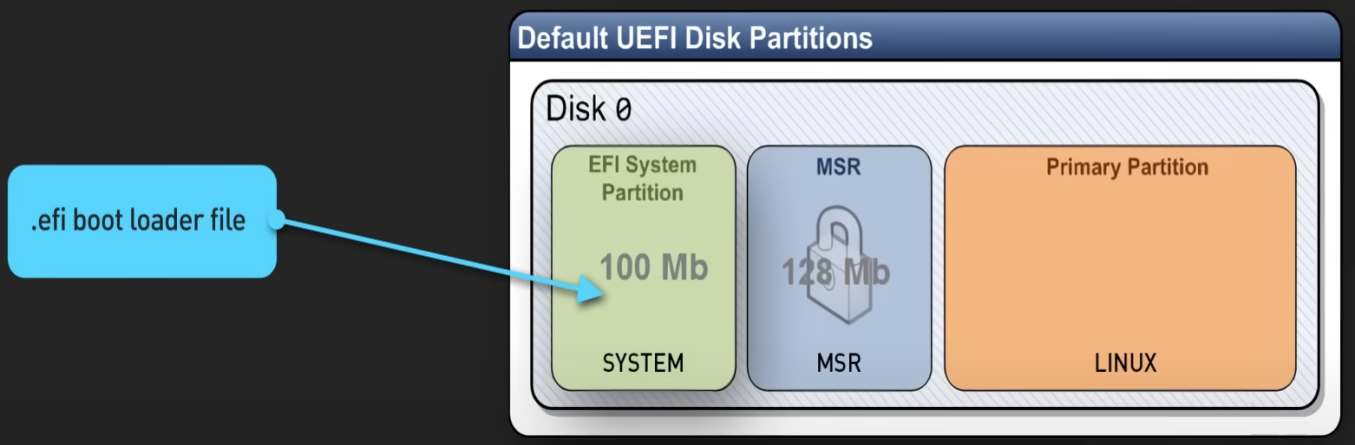
**Steps Involved in Booting**

1. The power button is hardwired with the CPU and when the power button is clicked the CPU turns on.
2. The CPU loads the BIOS/UEFI from the NVRAM.
3. The BIOS/UEFI runs a check called POST (Power On Self Test) which ensures if the pieces of the hardware connected to the system works properly and if there is any error it is displayed on the screen.
4. The BIOS/UEFI then finds the device from which it has to boot the boot loader like Hard disk, Removable drive, CD, etc.
5. The BIOS/UEFI then identifies the place where the boot loader ( /boot/efi/EFI/<distro>/grubx64.efi ) code lives, say MBR (Master Boot Record) in case of BIOS system and EFI partition system (ESP) in case of UEFI system. The grub configuration file can be found in /boot/grub/grub.cfg.

**BIOS System**

****

**UEFI System**

****

1. The boot loader determines which kernel to boot and then inserts the kernel into memory and the BIOS/UEFI hands over the control to the kernel.
2. Now the kernel decompresses itself and mounts the initramfs (Initial RAM File System) also known as the temporary root file system. It loads necessary kernel modules and device drivers necessary for the root file system to work, the root is mounted and it takes over the computer’s resources then starts the background processes & services.
3. After this the first process Systemd starts with PID 1, it checks for any remaining hardware drivers that are left out to be loaded, executes startup scripts, starts processes required to bring up the login prompt, mounts all file systems and disks and GUI environment.

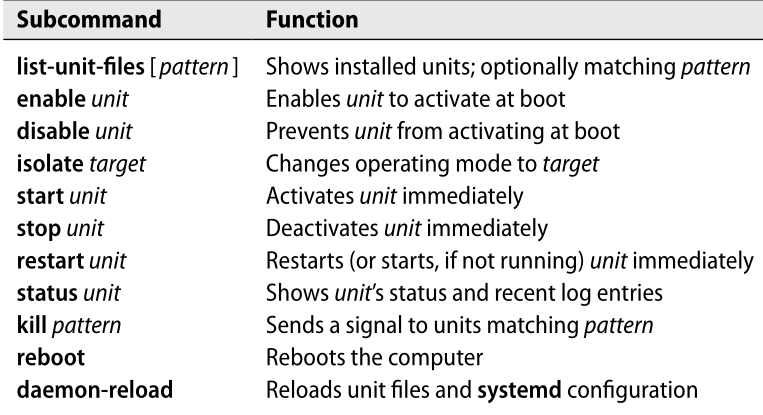
**Systemd**

systemd is a modern system and service manager for Linux, responsible for initializing the system after boot and managing all running processes and services. It organizes services using "unit files," which define how services are started, stopped, and managed, ensuring proper dependencies and order. By starting services in parallel, systemd significantly speeds up boot times compared to older init systems.

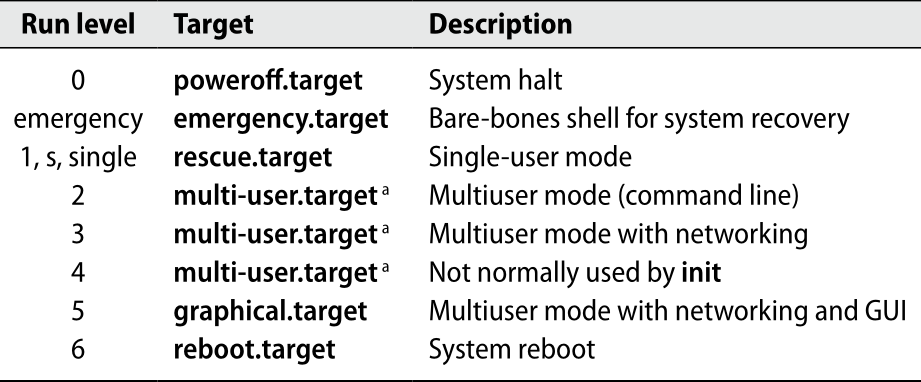
**Systemctl**

systemctl is a command-line utility used to interact with systemd, the system and service manager in Linux. It allows you to manage services, check system status, and control various systemd-related tasks such as starting, stopping, enabling, and disabling services, as well as managing system states (targets).

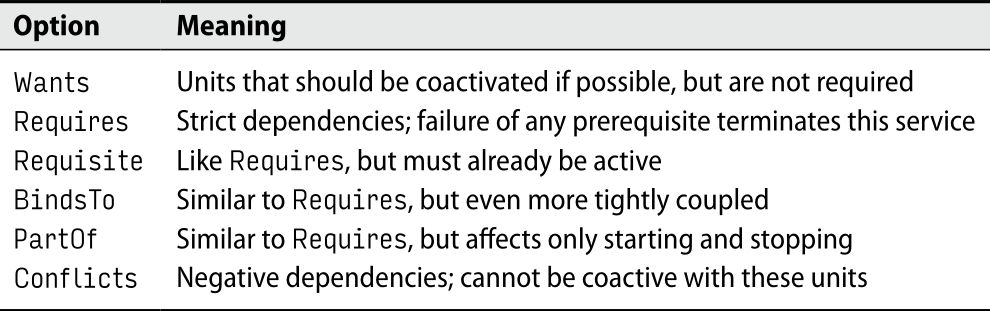
systemctl <option> <target/service name> (or) systemctl start ssh.service



**Mapping between init run levels and systemd targets**



**Explicit dependencies in the [Unit] section of unit files**

****

**Options**

start - To start a service.

stop - To stop a service.

restart - To restart the service (stop and start again).

reload - To update only changes without stopping the service.

enable - To start service during power on.

disable - To disable the enabled service.

status - To check the status of any service.