Linux Startup

Proces spuštění počítače

- Spuštění GNU/Linux

Spuštění GNU/Linux

Dovolte mi nejprve vyjasnit firmware a partition schéma.

Firmware

Firmware je to, co se spustí, když stisknete tlačítko napájení, jeho hlavním účelem je inicializovat váš hardware a poté načíst bootloader. Dva nejpoužívanější jsou:

BIOS: To je to, co najdete ve starých počítačích (téměř všechny počítače před rokem 2010), byl velmi základní až do té míry, že postrádá standardy, takže jinými slovy nepořádek. Vše, co dělá, je inicializovat základní hardware a načíst první sektor na spouštěném disku.

UEFI: Standardizováno v roce 2007, aby nahradilo BIOS, má nad hardwarem mnohem větší kontrolu, než jen jeho inicializaci (jako přetaktování, nastavení křivek ventilátoru,...). UEFI může běžet v režimu kompatibility (CMS) pro zpětnou kompatibilitu se stávajícími instalacemi MBR, kde funguje jako BIOS, který načítá pouze první sektor spouštěcího disku.

Podle specifikací UEFI:

- Musí být schopen číst souborový systém FAT (FAT12/16/32)
- Musí mít oddíl naformátovaný pomocí FAT 12/16/32 a označený ESP (EFI System Partition), který funguje jako kontejner pro bootloadery.
- Musí být schopen číst zaváděcí položky

Partition Schéma

Partition schémá je to, co organizuje celý váš disk (budu mu říkat disk, protože to znamená pevné disky, disky SSD, flash disky, optické disky...), a řídí, kolik oddílů může mít disk a dokonce jak velké jsou. Dva, které se používají hlavně:

GPT: Toto je moderní schéma rozdělení, které řeší předešlé chyby staré varianty (hlavně limit oddílů a disk větší než 2 TB), většinou je spojeno s UEFI.

MBR: Toto je tradiční schéma rozdělení dat do éry IBM, MBR je první sektor (512 bajtů), ve kterém je uložen primární zavaděč (zavaděč fáze 1, více později) a tabulka oddílů, která organizuje 4 primární oddíly disku. Nepleťte si to s tabulkou souborového systému, která organizuje vaše data uvnitř jednoho oddílu.

Nyní pojďme k těm fázím:

V čistém UEFI/GPT může UEFI číst tabulku GPT a najít oddíl ESP (podle příznaku ESP) a poté spustit GRUB (**fáze 1**), GRUB pak načíst OS (**fáze 2**) pomocí svého konfiguračního souboru (`/boot/grub/ grub.cfg`).

V systémech BIOS/MBR a BIOS/GPT BIOS načte spouštěcí kód do MBR (fáze 1), ale protože je tak malý, aby se vešel do jakéhokoli zavaděče, obsahuje pouze ukazatel na základní grub (fáze 1.5), který se nachází:

- V případě BIOSu/MBR: volné místo bez oddílů hned po MBR
- V případě BIOSu/GPT speciální oddíl bez souborového systému (obvykle po 1. oddílu)

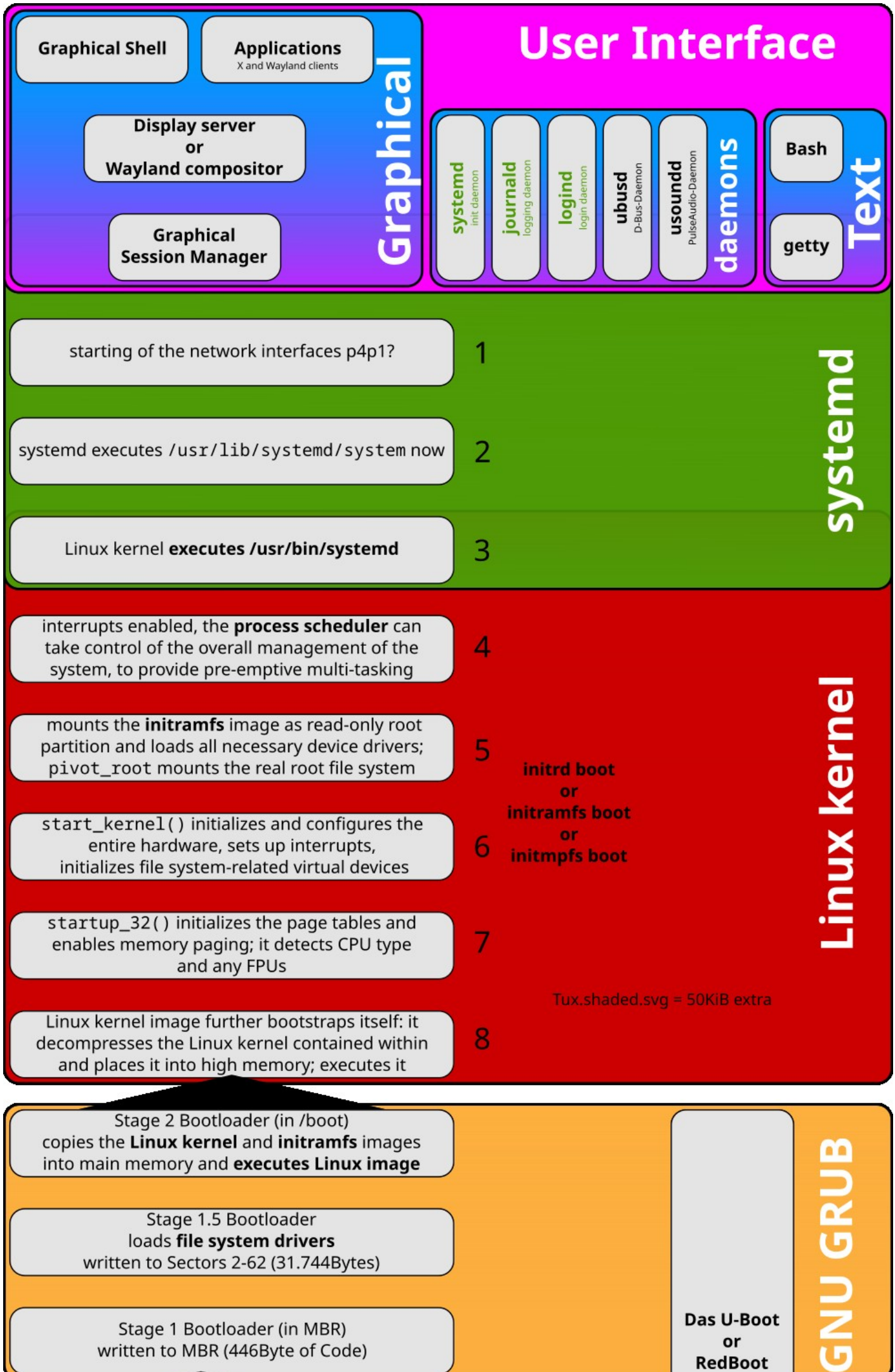
Protože BIOS je tak základní, že nezná žádnou představu o oddílech nebo souborových systémech, tento zavaděč **fáze 1.5** (`core.img`) obsahuje primární ovladače potřebné ke čtení tabulky souborového systému oddílu `/boot` a vyhledání konfiguračního souboru grub (`/boot/grub /grub.cfg`), který pak GRUB volitelně používá, nabízí uživateli zaváděcí nabídku se seznamem nainstalovaných OS/kernelů, poté načte jádro výchozího OS nebo to, co uživatel vybere v zaváděcí nabídce (**stage2**).

Co přijde po GRUB?

GRUB načte jádro a `initramfs` do paměti (`initrd` byl používán ve starších systémech, ale nahradit jej `initramfs` kvůli jeho omezením). Jádro poté připojí obsah `initramfs` (po jeho dekomprimaci) a připojí jej jako počáteční kořenový souborový systém (**který je uložen v paměti**), tento počáteční kořenový souborový systém obsahuje data potřebná pro jádro k inicializaci zařízení a spuštění programu nazvaného init, že připojení skutečný kořenový souborový systém (`/`) a převezměte kontrolu nad zaváděním zbytku systému.

Starší obrázek boot procesu

Tady ještě přikládám **starý obrázek boot procesu**, který by se mohl hodit pro pozdější úpravy. Některé části jsou pravdivé i dnes.


User Interface
Graphical
Graphical Shell
Applications
X and Wayland clients
Display server
or
Wayland compositor
Graphical
Session Manager
daemons
systemd
init daemon
journald
logging daemon
logind
login daemon
ubusd
D-Bus-Daemon
usoundd
PulseAudio-Daemon
Text
Bash
getty
systemd
starting of the network interfaces p4p1?
1
systemd executes /usr/lib/systemd/system now
2
Linux kernel executes /usr/bin/systemd
3
Linux kernel
interrupts enabled, the process scheduler can take control of the overall management of the system, to provide pre-emptive multi-tasking
4
mounts the initramfs image as read-only root partition and loads all necessary device drivers; pivot_root mounts the real root file system
5
initrd boot
or
initramfs boot
or
initmpfs boot
start_kernel() initializes and configures the entire hardware, sets up interrupts, initializes file system-related virtual devices
6
startup_32() initializes the page tables and enables memory paging; it detects CPU type and any FPUs
7
Tux.shaded.svg = 50KiB extra
Linux kernel image further bootstraps itself: it decompresses the Linux kernel contained within and places it into high memory; executes it
8
GNU GRUB
Stage 2 Bootloader (in /boot)
copies the Linux kernel and initramfs images into main memory and executes Linux image
Stage 1.5 Bootloader
loads file system drivers
written to Sectors 2-62 (31.744Bytes)
Stage 1 Bootloader (in MBR)
written to MBR (446Byte of Code)
Das U-Boot
or
RedBoot