Boot Process
UEFI + GRUB + Linux + systemd

Prof. Rossano Pablo Pinto
FATEC - Americana
04/04/2017 – v8

Agenda

- Introduction
- Operating System Initialization
- systemd

Introduction

- Bird's eye of the boot process:

BIOS/UEFI → Bootloader → kernel → init

Introduction

- Kernel space
  - Kernel
- User space
  - Init
    - All services

O.S. Initialization

- Example: Linux/IA64 with UEFI and GRUB(see pg. 86-26 UEFI - 2.3.2.1 Handoff State,pg. 90-30 - 2.3.4 x86 Platforms, Vol. 3A 2-1 book from intel, Figure 2-3 Vol 3A pg 2-11)
  - Power-on (real-mode)
  - CPU fetches first instruction <- ROM (UEFI)
  - UEFI switches processor to protected-mode
  - UEFI switches processor to Long (64 bits)
  - UEFI looks for a partition of type ESP (EFI System Partition - EF00)
  - UEFI loads an EFI application (for instance, GRUB)

O.S. Initialization

- ...cont.
  - GRUB loads the linux kernel to the memory and hands-off the control to Linux
  - Linux executes a bunch of routines to configure itself
  - The very last thing Linux does during initialization is the creation of the first process of the system:
    - the init
    - `/usr/src/linux-source-2.6.27/init/main.c` (line 823)

O.S. Initialization

// /usr/src/linux-source-2.6.27/init/main.c (line 823):

```
if (execute_command) {
        run_init_process(execute_command);
        printk(KERN_WARNING "Failed to execute %s.  Attempting "
                                        "defaults...\n", execute_command);
}
```

run_init_process("/sbin/init");

```
run_init_process("/etc/init");
run_init_process("/bin/init");
run_init_process("/bin/sh");

panic("No init found.  Try passing init= option to kernel.");
```

O.S. Initialization

- Both LILO and GRUB allows to inform which program is to act as an init:
  - init=XXXXXXX
  - This makes that the code block that starts at line 823 in main.c be executed:

```
if (execute_command) {
        run_init_process(execute_command);
        printk(KERN_WARNING "Failed to execute %s.  Attempting "
                                        "defaults...\n", execute_command);
}
```

O.S. Initialization

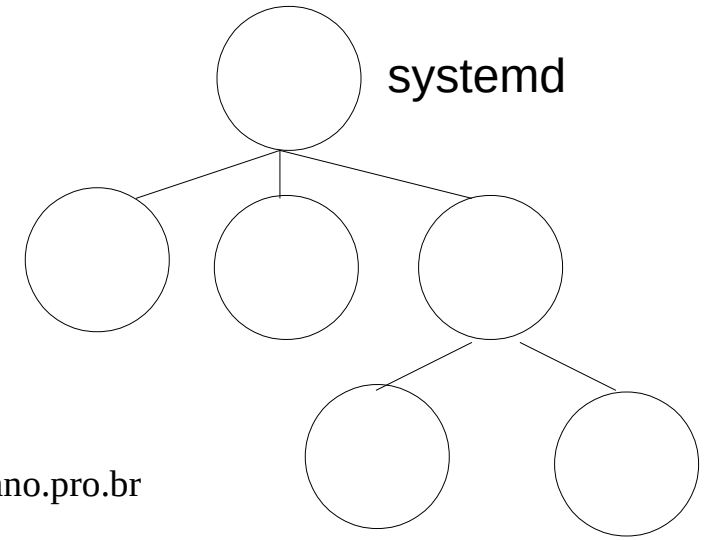
- To boot the system into a shell, and complete ignore an init system:
  - Provide `init=SHELL_PATH` at LILO or GRUB
  - It even works for very simple programs that just read the STDIN (`scanf`) and print to STDOUT (`printf`)

O.S. Initialization

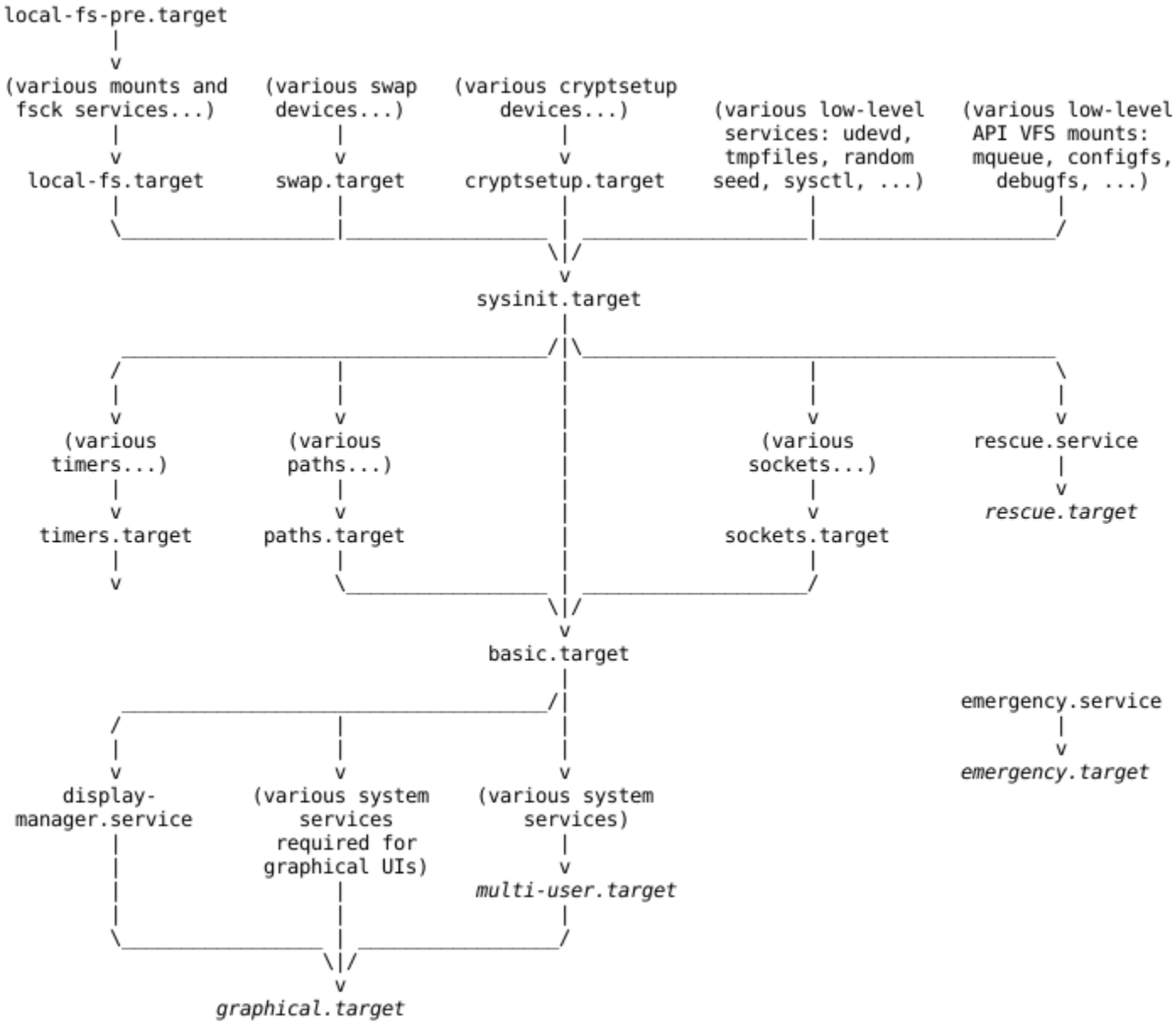
- Now, the role of the init is to load all services of the system.
- There are 3 well known init “versions”:
  - System V / BSD - config file: `/etc/inittab`
  - Upstart (used to be the ubuntu init system)
  - systemd (most of the distributions)
- **This material is based on systemd**

O.S. Initialization

- systemd reads the directories `/usr/lib/systemd/system`, `/etc/systemd/system` and `/etc/systemd/system/[name.type].d/*.conf`, and loads all services that must run at boot time
- systemd becomes the father (grandfather/great-grandfather..) of all the processes of the system:

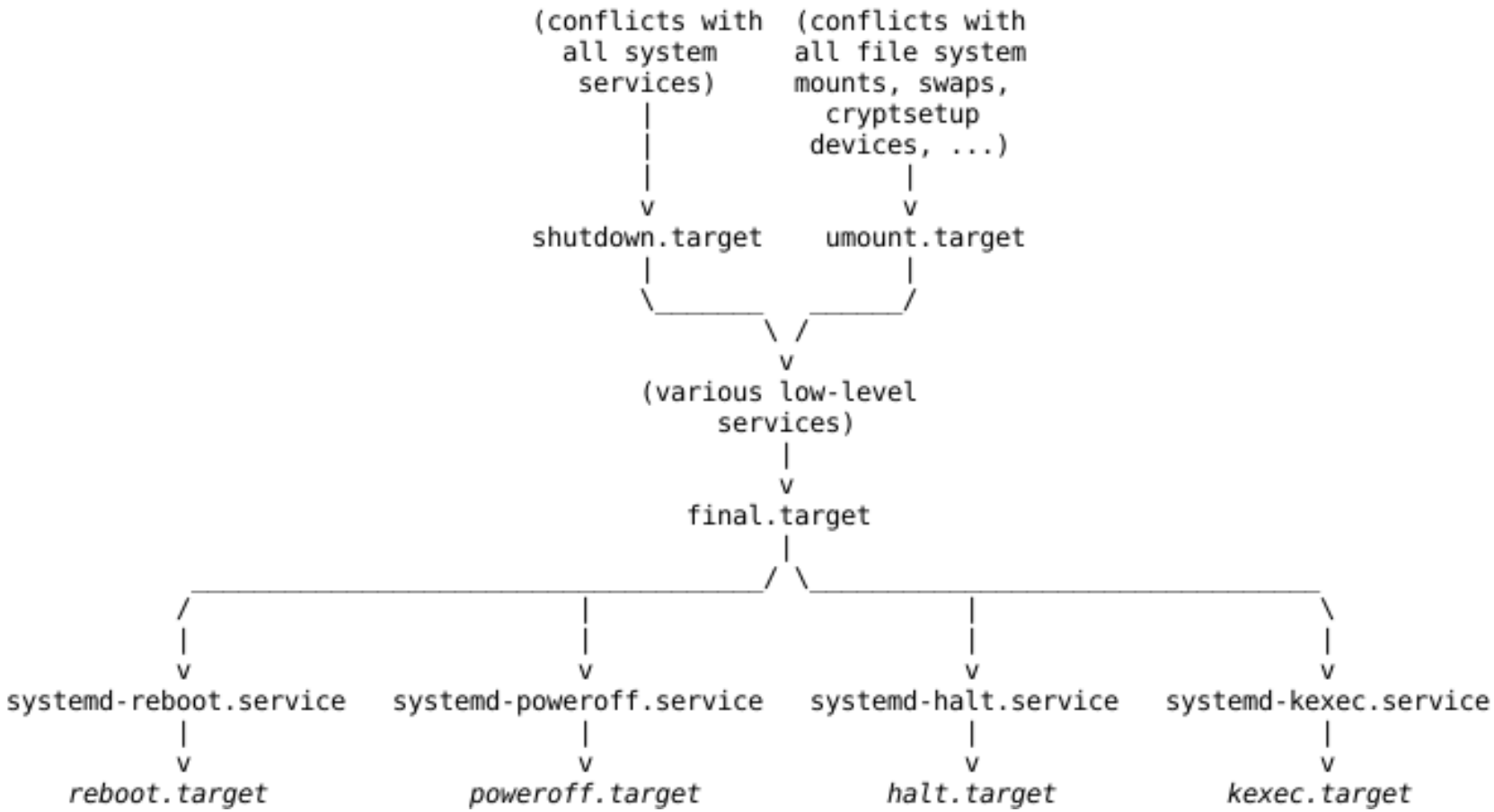


O.S. Initialization



https://www.freedesktop.org/software/systemd/man/bootup.html / man bootup

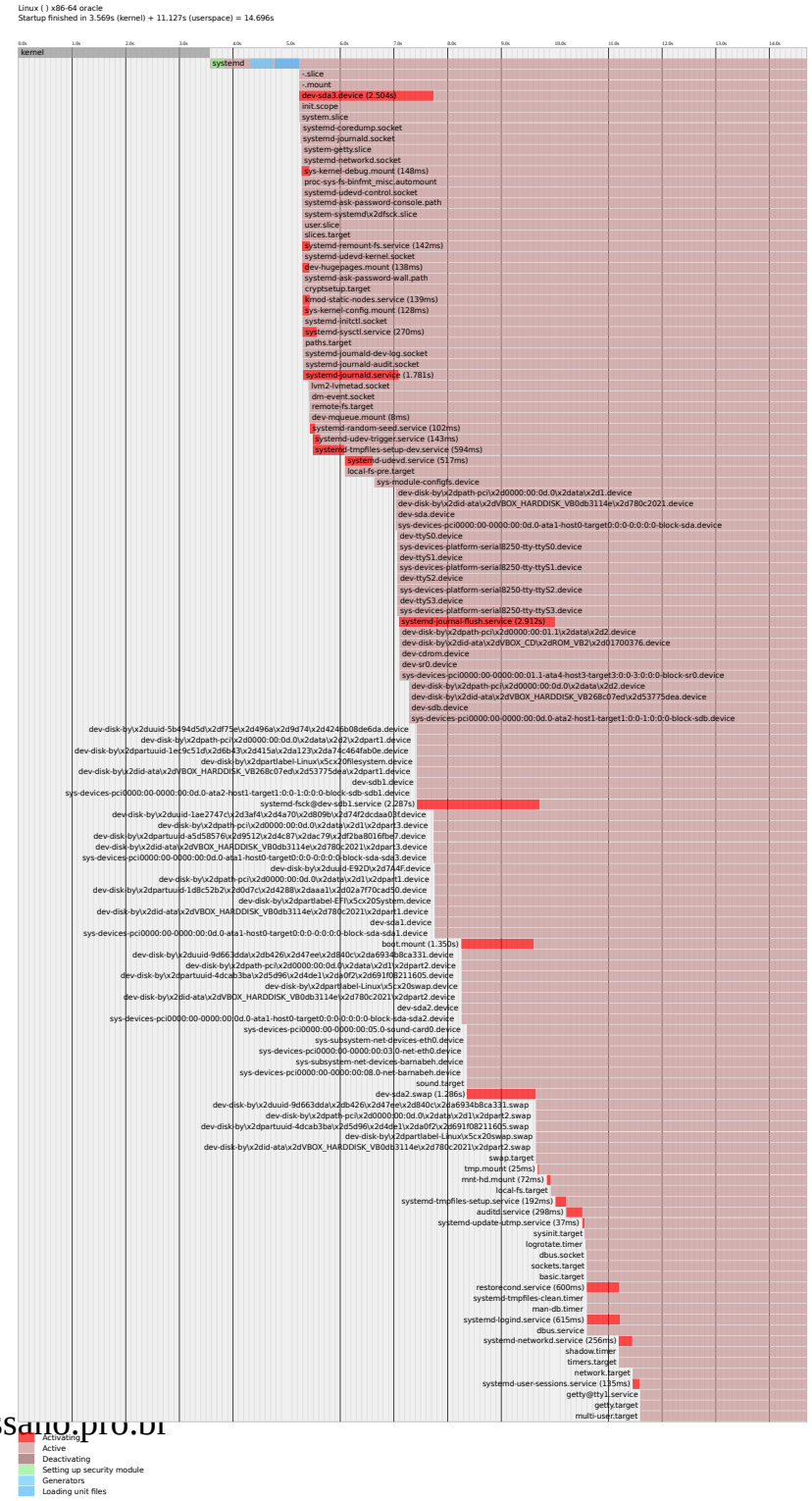
O.S. Initialization (shutdown)



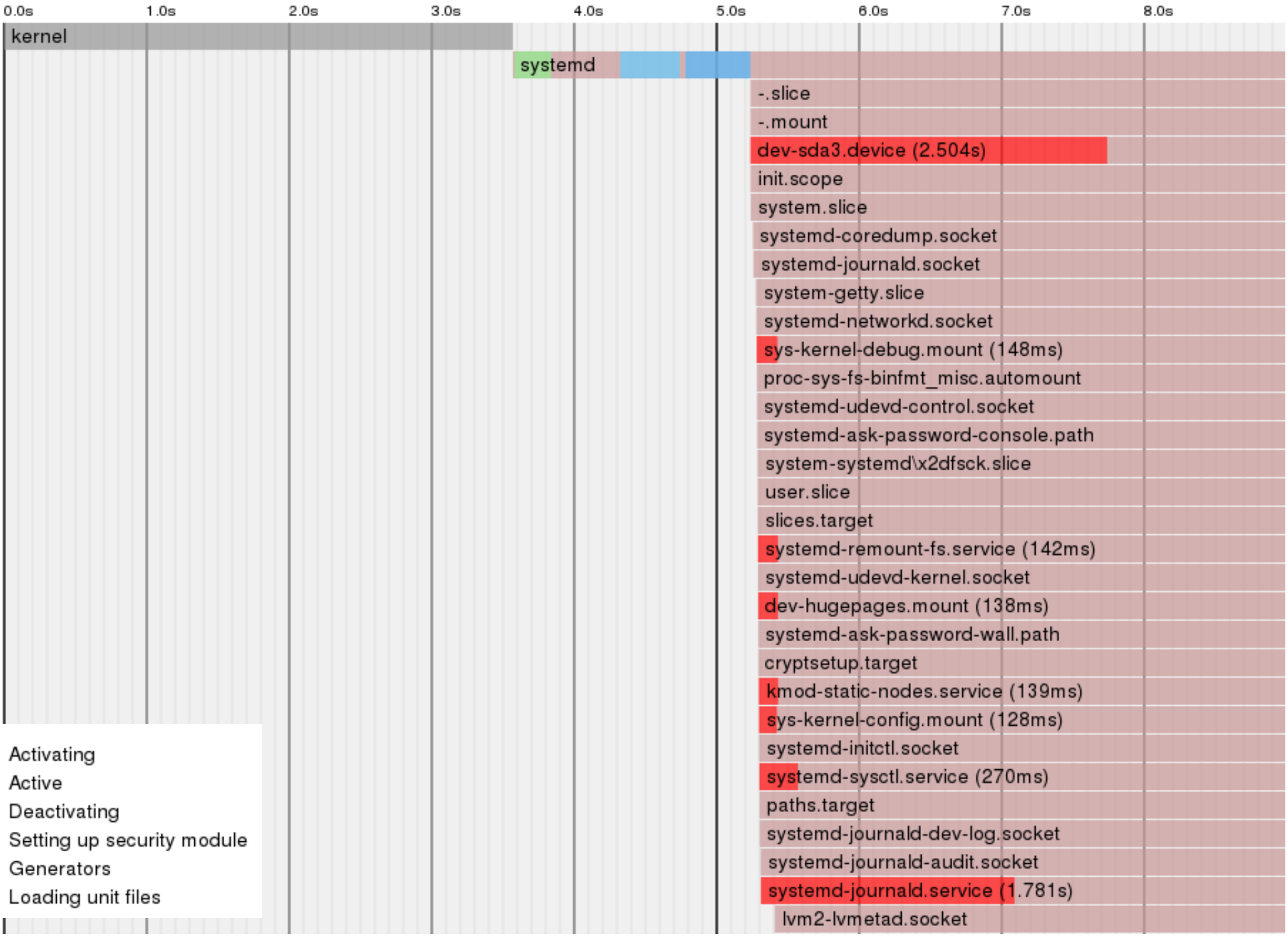
https://www.freedesktop.org/software/systemd/man/bootup.html / man bootup

`systemd-analyze plot`

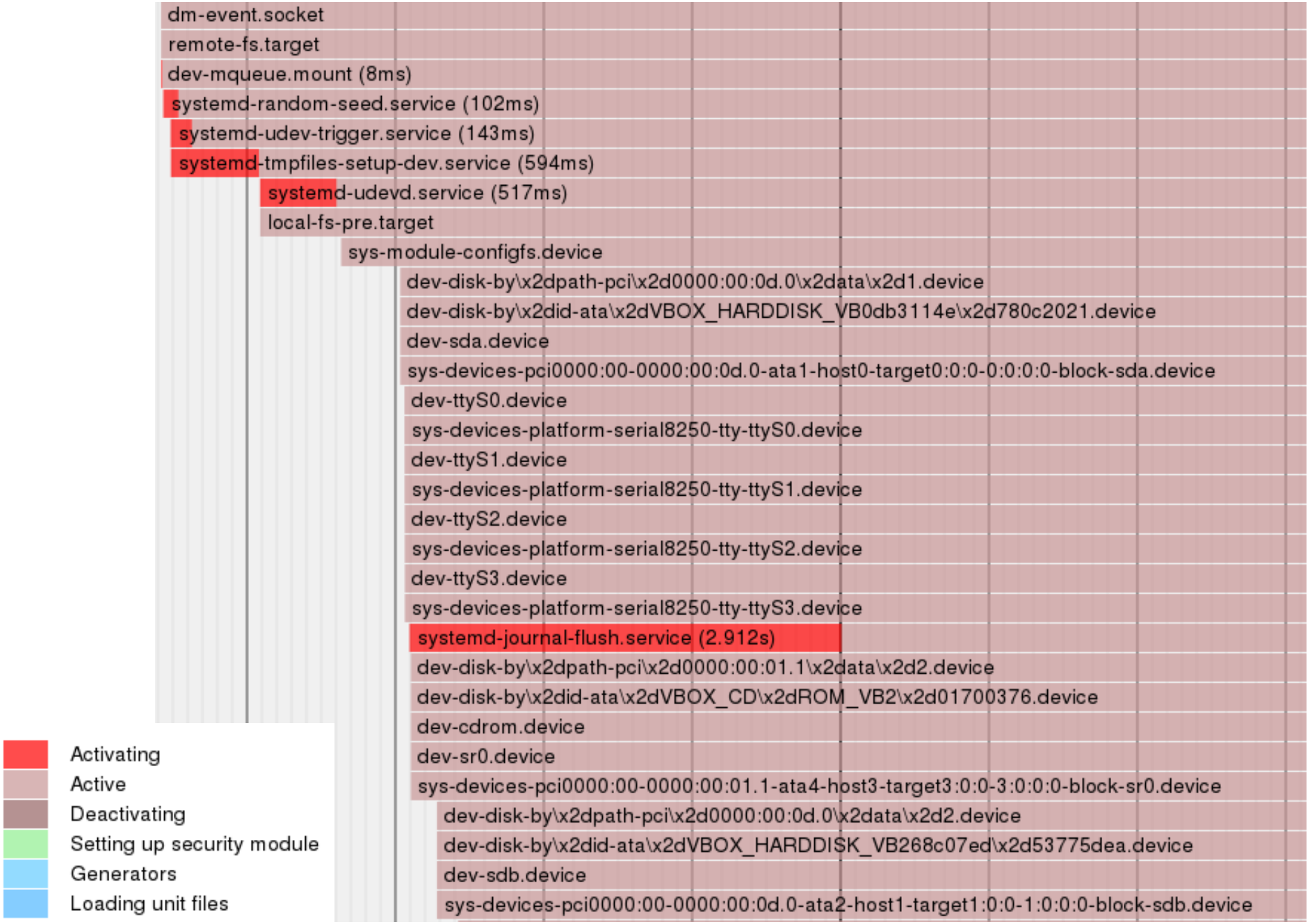
This command generates an SVG file with initialization info. Every column represents 1 second



Linux ( ) x86-64 oracle
Startup finished in 3.569s (kernel) + 11.127s (userspace) = 14.696s



O.S. Initialization




dev-disk-by\x2duuid-5b494d5d\x2df75e\x2d496a\x2d9d74\x2d4246b08de6da.device
dev-disk-by\x2dpath-pci\x2d0000:00:0d.0\x2data\x2d2\x2dpart1.device
dev-disk-by\x2dpartuuid-1ec9c51d\x2d6b43\x2d415a\x2da123\x2da74c464fab0e.device
dev-disk-by\x2dpartlabel-Linux\x5cx20filesystem.device
dev-disk-by\x2did-ata\x2dVBOX_HARDDISK_VB268c07ed\x2d53775dea\x2dpart1.device
dev-sdb1.device
sys-devices-pci0000:00-0000:00:0d.0-ata2-host1-target1:0:0-1:0:0:0-block-sdb-sdb1.device
systemd-fsck@dev-sdb1.service (2.287s)
dev-disk-by\x2duuid-1ae2747c\x2d3af4\x2d4a70\x2d809b\x2d74f2dcdaa03f.device
dev-disk-by\x2dpath-pci\x2d0000:00:0d.0\x2data\x2d1\x2dpart3.device
dev-disk-by\x2dpartuuid-a5d58576\x2d9512\x2d4c87\x2dac79\x2df2ba8016fbe7.device
dev-disk-by\x2did-ata\x2dVBOX_HARDDISK_VB0db3114e\x2d780c2021\x2dpart3.device
sys-devices-pci0000:00-0000:00:0d.0-ata1-host0-target0:0:0-0:0:0:0-block-sda-sda3.device
dev-disk-by\x2duuid-E92D\x2d7A4F.device
dev-disk-by\x2dpath-pci\x2d0000:00:0d.0\x2data\x2d1\x2dpart1.device
dev-disk-by\x2dpartuuid-1d8c52b2\x2d0d7c\x2d4288\x2daaa1\x2d02a7f70cad50.device
dev-disk-by\x2dpartlabel-EFI\x5cx20System.device
dev-disk-by\x2did-ata\x2dVBOX_HARDDISK_VB0db3114e\x2d780c2021\x2dpart1.device
dev-sda1.device
sys-devices-pci0000:00-0000:00:0d.0-ata1-host0-target0:0:0-0:0:0:0-block-sda-sda1.device
boot.mount (1.350s)
dev-disk-by\x2duuid-9d663dda\x2db426\x2d47ee\x2d840c\x2da6934b8ca331.device
dev-disk-by\x2dpath-pci\x2d0000:00:0d.0\x2data\x2d1\x2dpart2.device
dev-disk-by\x2dpartuuid-4dcab3ba\x2d5d96\x2d4de1\x2da0f2\x2d691f08211605.device
dev-disk-by\x2dpartlabel-Linux\x5cx20swap.device
dev-disk-by\x2did-ata\x2dVBOX_HARDDISK_VB0db3114e\x2d780c2021\x2dpart2.device
dev-sda2.device
ys-devices-pci0000:00-0000:00:0d.0-ata1-host0-target0:0:0-0:0:0:0-block-sda-sda2.device
sys-devices-pci0000:00-0000:00:05.0-sound-card0.device
sys-subsystem-net-devices-eth0.device
sys-devices-pci0000:00-0000:00:03.0-net-eth0.device
sys-subsystem-net-devices-barnabeh.device
sys-devices-pci0000:00-0000:00:08.0-net-barnabeh.device
sound.target
Activating
Active
Deactivating
Setting up security module
Generators
Loading unit files

O.S. Initialization

O.S. Initialization

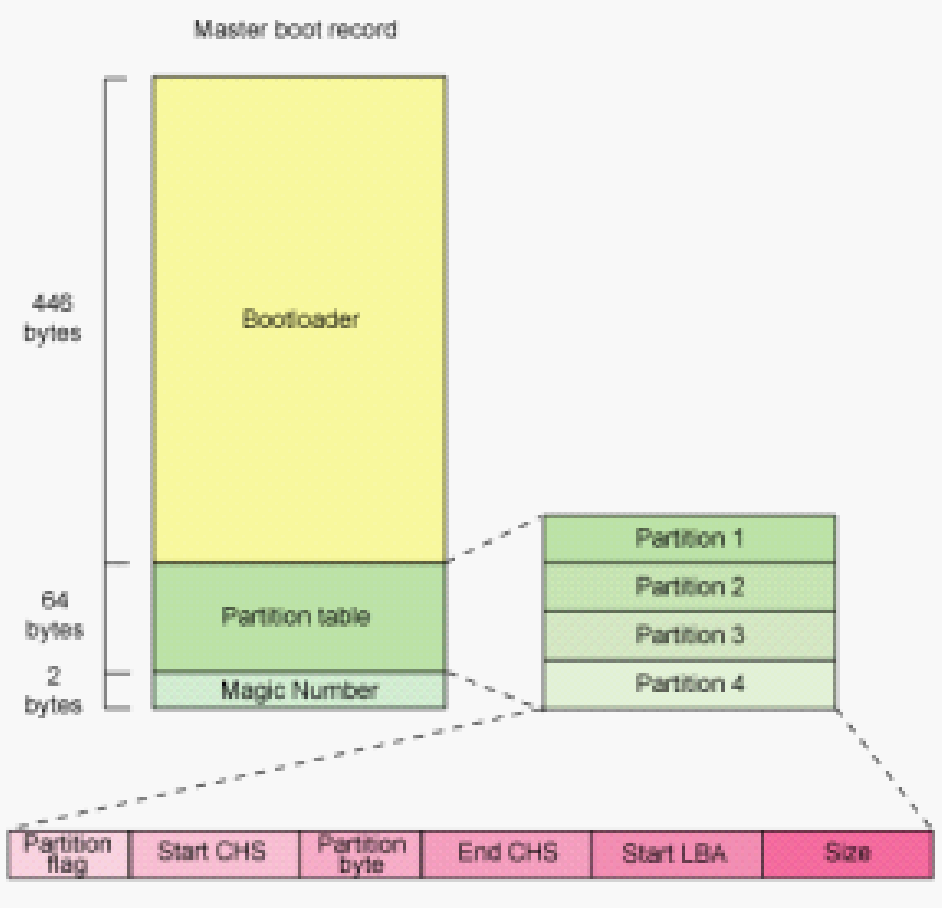
Test the following commands:

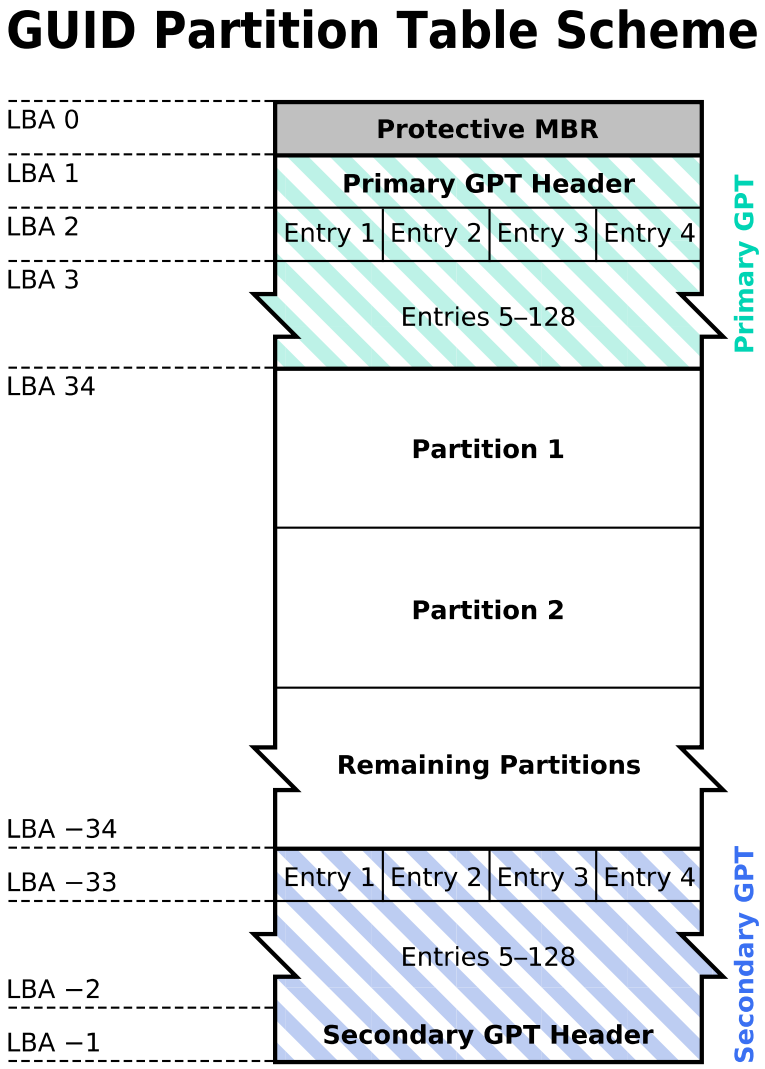
```
ps -ejH
pstree
```

O.S. Initialization

- Rescue mode at boot:
  - GRUB, press e
  - locate line “vmlinuz ….. rw”
  - type at the end of the line: `systemd.unit=rescue.target`
  - ctrl-x
- Changing target after boot
  - `systemctl isolate [TARGET]`

GPT x MBR partitions





GPT partition entry

Offset	Length	Contents
0	8 bytes	Signature ("EFI PART", 45 46 49 20 50 41 52 54)
8	4 bytes	Revision (For GPT version 1.0 (through at least UEFI version 2.3.1), the value is 00 00 01 00)
12	4 bytes	Header size in little endian (in bytes, usually 5C 00 00 00 meaning 92 bytes)
16	4 bytes	CRC32 of header (0 to header size), with this field zeroes during calculation
20	4 bytes	Reserved; must be zero
24	8 bytes	Current LBA (location of this header copy)
32	8 bytes	Backup LBA (location of the other header copy)
40	8 bytes	First usable LBA for partitions (primary partition table last LBA + 1)
48	8 bytes	Last usable LBA (secondary partition table first LBA – 1)
56	16 bytes	Disk GUID (also referred to as UUID on UNIXes)
72	8 bytes	Partition entries starting LBA (always 2 in primary copy)
80	4 bytes	Number of partition entries
84	4 bytes	Size of partition entry (usually 128)
88	4 bytes	CRC32 of partition array
92	*	Reserved; must be zeroes for the rest of the block (420 bytes for a 512-byte LBA)
LBA size	**Total**	

Conclusion

- Each O.S. offers it's own way of starting services during boot
- Some services can be offered by the kernel itself (Ex.: Linux HTTP/NFS/Firewall) or run in user space
- Traditional SysV init was replaced by systemd

systemd

- Change slides to
  - aulas-systemd-archlinux**.odp