# Booting and Services

Linux System Administration
Fall 2022

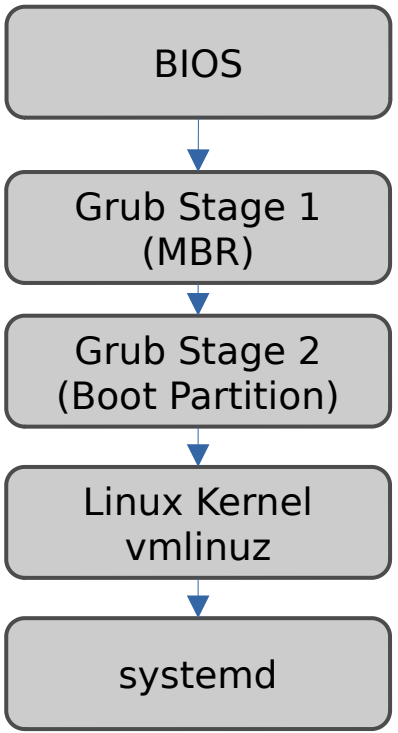
# Bootstrapping

- Loading initial software onto the hardware

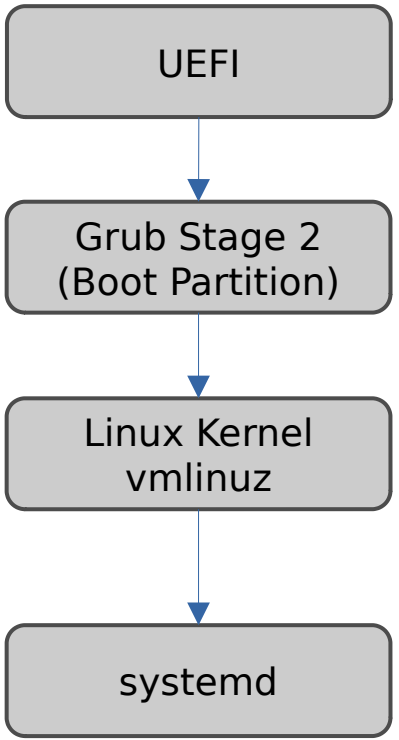


# Modern Bootstrap

- Legacy BIOS




- UEFI Boot



# Boot Methods

- BIOS
  - **B**asic **I**nput/**O**utput **Sy**stem
- UEFI
  - **U**nified **E**xtensible **F**irmware **I**terface
- PXE
  - **P**reboot **E**xecution **E**nvironment
  - Part of UEFI

# Bootloaders

- loadlin
  - Original linux bootloader (obsolete)
- lilo and elilo
  - **li**nux **lo**ader (legacy)
- GRUB
  - **GR**and **U**nified **B**ootloader
- syslinux
  - group of loaders for CD, USB, etc.

# GRUB

- BIOS started from ROM
  - Stage 1 loaded from Master Boot Record
  - Stage 2 loaded from /boot
- UEFI started from ROM
  - Stage 2 loaded from /boot
- Reads `/boot/grub/grub.cfg` (made from `/etc/grub.d`)
  - Which kernel to load
  - Parameters to pass to the kernel
  - Runs `vmlinuz`

# Loading the kernel

- `vmlinuz` compressed Linux kernel
- `initramfs` or `initrd.img`
  - Initial RAM disk file system for kernel
- `config` kernel configuration stuff
- `System.map` kernel crash symbol maps

# Kernel parameters

- Check running kernel `cat /proc/cmdline`
- Edit by `e` on GRUB boot prompt
  - `root=` root file system (`UUID=` or `/dev/????`)
  - `ro` mount read only
  - `quiet` quiet boot
  - `systemd.unit=` set systemd target
  - `rd.break` start shell instead of systemd
  - `init=` process to launch

# Managing processes

- init
  - Traditional UNIX mother of all processes
    - PID=1
    - Used by BSD and legacy and rebel Linux distros
  - Runlevels 0-6
  - `rc` scripts
- systemd
  - New parallel process manager
    - PID=1
    - RedHat, Debian, ...
  - targets
  - units
  - too many other things

# init runlevels

- Runlevel use
  - 0 system halt
  - 1 single user
  - 2 multi-user
  - 3 multi-user with networking
  - 4 no standard
  - 5 graphical login
  - 6 reboot
- Set as `init N`
- Legacy Linux/System V
  - Scripts in `/etc/init.d`
    - start, stop, restart, reload
  - Linked to `/etc/rcN.d`
    - `/etc/rc2.d/K80apache2`
      - Kills apache2 at runlevel 2
    - `/etc/rc3.d/S20apache2`
      - Start apache2 at runlevel 3
    - Sort determines order
- BSD somewhat different

# systemd concepts

- target
  - More flexible than runlevels
  - Names have meaning
    - `rescue`
    - `multi-user`
    - `graphical`
  - Dependency tree
  - Can be user defined
- unit
  - Defines a service, device,...
  - Explicit dependencies
    - `Wants` start in parallel
    - `Requires` start after
    - Also `Before` and `After`
  - `WantedBy` when to start

# systemd files

- `/lib/systemd` systemd code
- `/lib/systemd/system` unit and target files
- `/etc/systemd` configuration files
- `/etc/systemd/system/*.wants` enabled units
  - symbolic links to `/lib/systemd/system`
- `/bin/systemctl` control program

# Example unit: apache2.service

```
[Unit]
Description=The Apache HTTP Server
After=network.target remote-fs.target nss-lookup.target
Documentation=https://httpd.apache.org/docs/2.4/

[Service]
Type=forking
Environment=APACHE_STARTED_BY_SYSTEMD=true
ExecStart=/usr/sbin/apachectl start
ExecStop=/usr/sbin/apachectl graceful-stop
ExecReload=/usr/sbin/apachectl graceful
KillMode=mixed
PrivateTmp=true
Restart=on-abort

[Install]
WantedBy=multi-user.target
```

# Example Unit: ssh.service

```
[Unit]
Description=OpenBSD Secure Shell server
Documentation=man:sshd(8) man:sshd_config(5)
After=network.target auditd.service
ConditionPathExists=!/etc/ssh/sshd_not_to_be_run

[Service]
EnvironmentFile=-/etc/default/ssh
ExecStartPre=/usr/sbin/sshd -t
ExecStart=/usr/sbin/sshd -D $SSHD_OPTS
ExecReload=/usr/sbin/sshd -t
ExecReload=/bin/kill -HUP $MAINPID
KillMode=process
Restart=on-failure
RestartPreventExitStatus=255
Type=notify
RuntimeDirectory=sshd
RuntimeDirectoryMode=0755

[Install]
WantedBy=multi-user.target
Alias=sshd.service
```

# Managing systemd

- `systemctl` action unit
  - `start` start immediately
  - `stop` stop immediately
  - `restart` stop&start immediately
  - `enable` set to start on boot
  - `disable` do not start on boot
  - `status` show status
  - `is-enabled` will it start on boot?
  - `is-active` is it running?
  - `daemon-reload` restart systemd
  - `isolate` activate target
  - `show` units
  - `list-dependencies` show tree
  - `mask` prohibit unit from staring
  - `default` = `isolate` *`default`*
  - `set-default` set default target
  - `get-default` show default target

# Troubleshooting systemd

- Check if a service is running and enabled
- Stop, start, restart, enable, disable service
- Recognize, get and set the default target
- Isolate a target