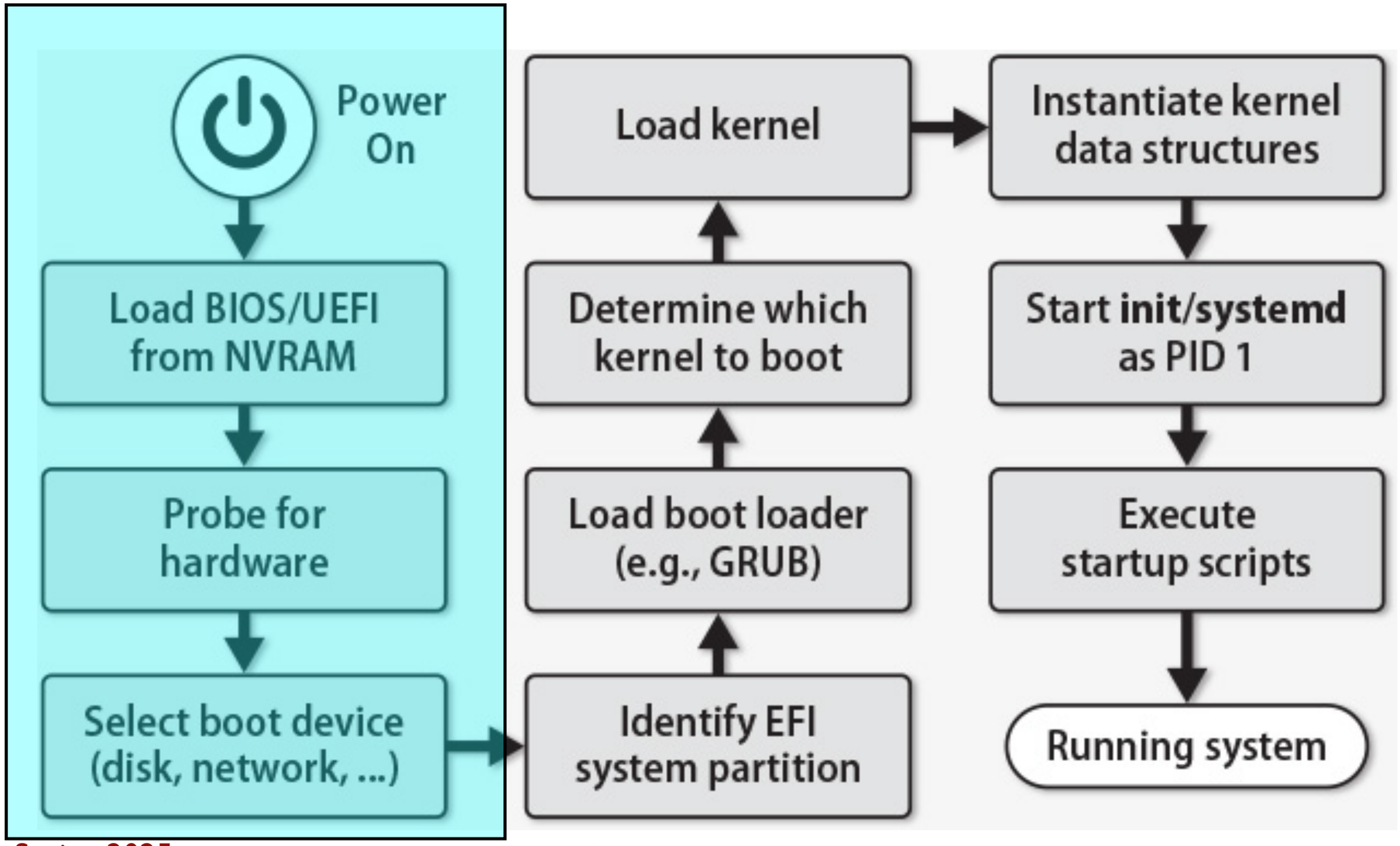
Booting and System Management Daemons

CIS 2235 Linux System Administration

Agenda

- The boot process
- The `init` program
- Service Management
  - System V initialization
  - Upstart
  - Systemd

The Boot Process



Firmware

BIOS/UEFI are located in firmware (NVRAM)
It is the starting point of the boot process.
It probes for hardware and disks.
  It can make an ordered list of devices to try for boot:
    CD-ROM first (if something in drive), then disk /dev/XXXX, for example
It does simple health checks.
It looks for the next stage of bootstrapping code.
  The secondary boot loader, which allows specification of parameters when booting the system.

BIOS

Legacy BIOS (Basic Input/Output System)

An older form of boot firmware, but still exists on some systems.

It uses the master boot record (MBR).

… has a boot block with the first stage boot loader

… primitive disk partitioning table (i.e., **not** GPT)

… very small < 512 bytes

… can only look for the secondary boot loader in specific places.

Volume Boot Record (beginning of partition on boot disk)

Between MBR and first partition (GRUB)

UEFI

Unified Extensible Firmware Interface

… more sophisticated than legacy BIOS

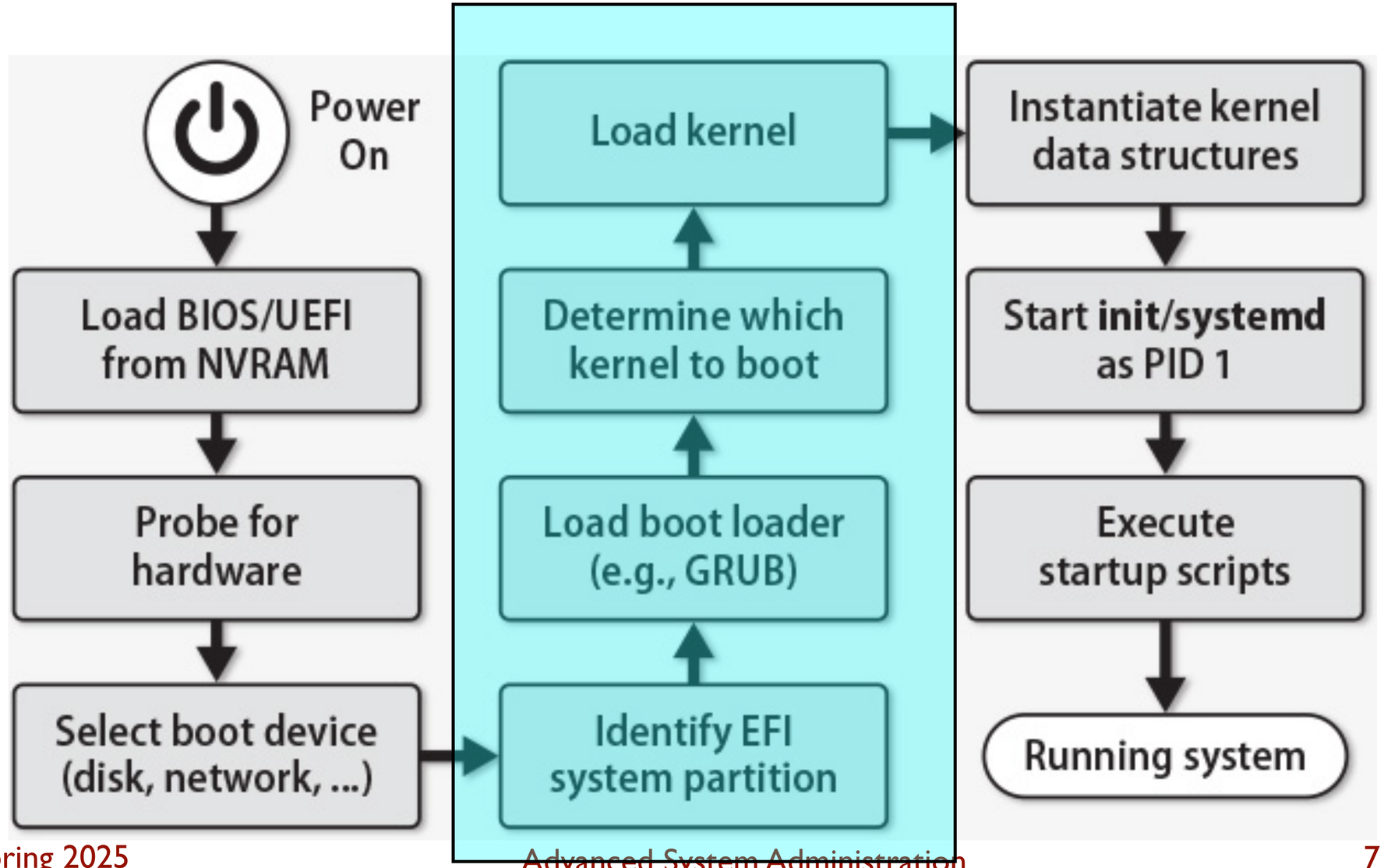
… includes a GUID partition table (GPT and large partitions > 2.2 TB)

… also understands FAT

Can do more complex and flexible things.

Has a formalized API (efibootmanager).

The Boot Process



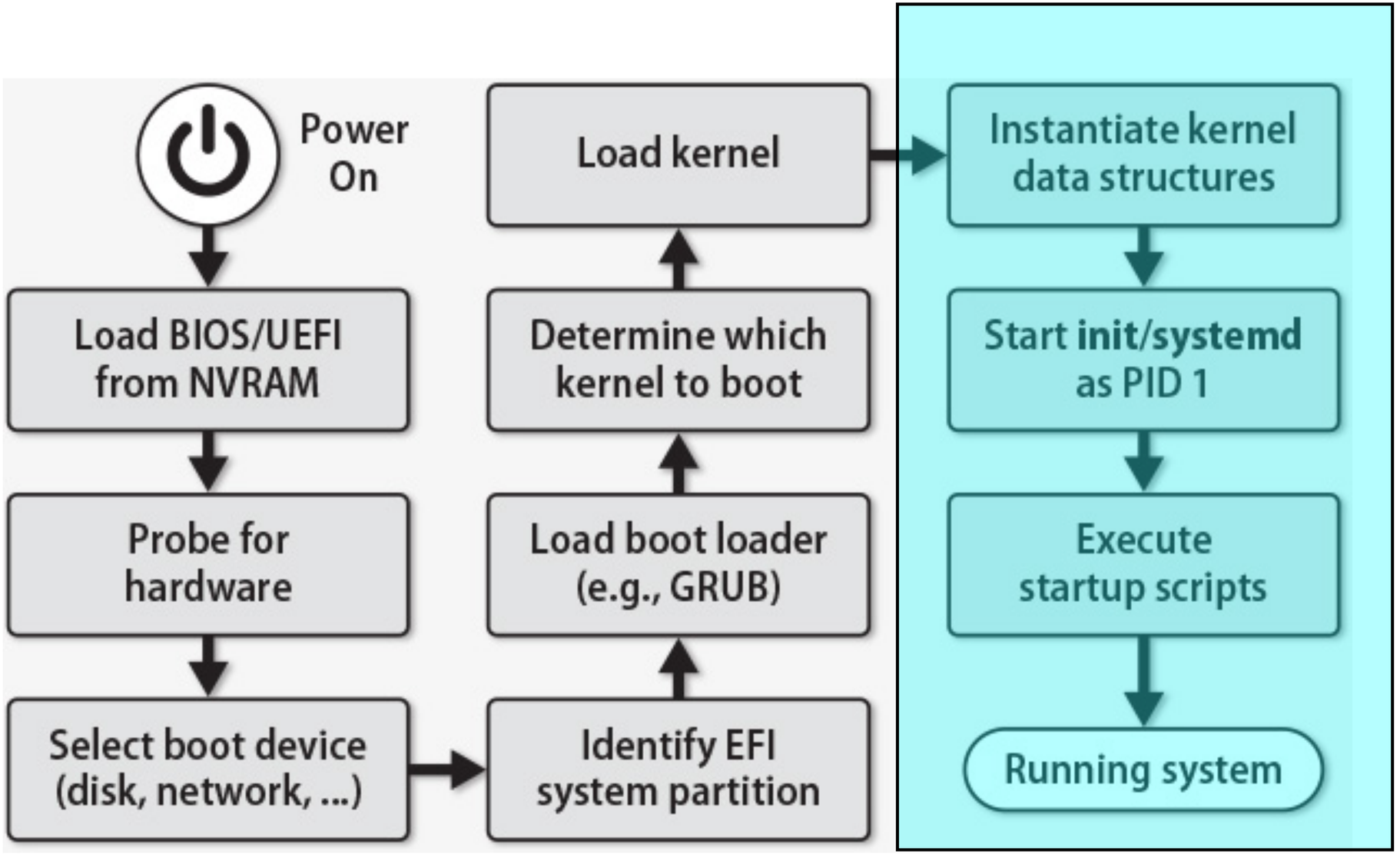
GRUB

Both forms of BIOS look for a secondary boot loader
… generally GRUB (Grand Unified Boot Loader).

GRUB identifies and loads the appropriate OS kernel
… can specify parameters for boot.
… `grub.cfg` in `/boot/grub` (or `/boot/grub2` on RH, CentOS).
… Linux distros give tooling to edit the config file.

See textbook for more details about grub and kernel configurations, or the official grub manual at https://www.gnu.org/software/grub/manual/grub/

The Boot Process



Kernel Processes

After the kernel starts:

- "Spontaneous" processes start
- … part of the kernel implementation ("kernel threads")
- … recognizable by brackets in a ps output

PID	TTY	STAT	TIME	COMMAND
1	?	Ss	0:01	/sbin/init maybe-ubiquity
2	?	S	0:00	[kthreadd]
3	?	I	0:00	[kworker/0:0]
4	?	I<	0:00	[kworker/0:0H]
5	?	I	0:00	[kworker/u2:0]
6	?	I<	0:00	[mm_percpu_wq]
7	?	S	0:00	[ksoftirqd/0]

init/kthreadd

Process ID 1 and 2.

… ensures the system is running the right stuff at any given time.

… has an idea of mode:

single user: minimal file system, few services, root shell at console

multi-user: normal running state, all fs, networking, plus GUI stuff

server mode: same as multi-user minus GUI

init jobs

`init` -> link to `systemd`

- sets the name of the computer
- sets the timezone
- checks disks with fsck
- mounts filesystems
- removes old files from tmp
- configures network interfaces
- configures packet filters
- starts up daemons and services

A lot of this work is done via scripts or services.

init History

Originally, sys V init "traditional init"

early '80s

```
/etc/init.d/apache2 start
```

replaced with upstart on Ubuntu for 6 years

```
service apache2 start
```

Ubuntu 15.04+ — now systemd

```
systemctl start apache2
```

On macOS, launchd

All of these serve the same role — start-up necessary processes and services by running commands/scripts.

systemd Design Philosophy

sys V init, all services were controlled by independent shell scripts

- scripts each ran independently
- no concurrency

Upstart allowed some parallelism, but the pieces are still relatively independent

systemd unifies many pieces of the system

- … violates the Unix principle of small modular design
- … controversial with many old-time Unix people

systemd

based on units, where a unit could be

- service
- socket
- device
- mount point
- automount point
- swap file or partition
- startup target
- + others

systemd

- behaviors defined and configured by unit files
  - look a lot like MS/DOS .ini files
  - can be found in
    - /usr/lib/systemd/system, or
    - /lib/systemd/system (depending on system)
    - (user files) /etc/systemd/system

systemd unit file

rsync unit file:

```
[Unit]
Description=fast remote file copy program daemon
ConditionPathExists=/etc/rsyncd.conf

[Service]
ExecStart=/usr/bin/rsync --daemon --no-detach

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

systemctl allows control of systemd

Subcommand	Function
list-unit-files [*pattern*]	Shows installed units; optionally matching *pattern*
enable *unit*	Enables *unit* to activate at boot
disable *unit*	Prevents *unit* from activating at boot
isolate *target*	Changes operating mode to *target*
start *unit*	Activates *unit* immediately
stop *unit*	Deactivates *unit* immediately
restart *unit*	Restarts (or starts, if not running) *unit* immediately
status *unit*	Shows *unit*'s status and recent log entries
kill *pattern*	Sends a signal to units matching *pattern*
reboot	Reboots the computer
daemon-reload	Reloads unit files and **systemd** configuration

listing services

```
$ systemctl list-units --type=service
UNIT                                LOAD   ACTIVE SUB     DESCRIPTION
accounts-daemon.service             loaded active running Accounts Service
apparmor.service                    loaded active exited  AppArmor initialization
apport.service                      loaded active exited  LSB: automatic crash report generation
atd.service                         loaded active running Deferred execution scheduler
blk-availability.service            loaded active exited  Availability of block devices
cloud-config.service                loaded active exited  Apply the settings specified in cloud-
config
cloud-final.service                 loaded active exited  Execute cloud user/final scripts
cloud-init-local.service            loaded active exited  Initial cloud-init job (pre-networking)
cloud-init.service                  loaded active exited  Initial cloud-init job (metadata
service crawler)
console-setup.service               loaded active exited  Set console font and keymap
cron.service                        loaded active running Regular background program processing
daemon
```

status of a service

```
$ systemctl status cron
● cron.service - Regular background program processing daemon
   Loaded: loaded (/lib/systemd/system/cron.service; enabled; vendor preset: enabled)
   Active: active (running) since Wed 2019-02-13 02:25:10 UTC; 1h 0min ago
     Docs: man:cron(8)
 Main PID: 699 (cron)
    Tasks: 1 (limit: 1110)
   CGroup: /system.slice/cron.service
           └─699 /usr/sbin/cron -f

Feb 13 02:25:10 ubuntu_lts systemd[1]: Started Regular background program processing daemon.
Feb 13 02:25:10 ubuntu_lts cron[699]: (CRON) INFO (pidfile fd = 3)
Feb 13 02:25:10 ubuntu_lts cron[699]: (CRON) INFO (Running @reboot jobs)
Feb 13 03:17:01 ubuntu_lts CRON[1616]: pam_unix(cron:session): session opened for user root by
(uid=0)
Feb 13 03:17:01 ubuntu_lts CRON[1616]: pam_unix(cron:session): session closed for user root
```

systemctl status

State	Meaning
`bad`	Some kind of problem within **systemd**; usually a bad unit file
`disabled`	Present, but not configured to start autonomously
`enabled`	Installed and runnable; will start autonomously
`indirect`	Disabled, but has peers in `Also` clauses that may be enabled
`linked`	Unit file available through a symlink
`masked`	Banished from the **systemd** world from a logical perspective
`static`	Depended upon by another unit; has no install requirements

stopping a service

```
$ sudo systemctl stop cron
[sudo] password for ldamon:
$ systemctl status cron
● cron.service - Regular background program processing daemon
   Loaded: loaded (/lib/systemd/system/cron.service; enabled; vendor preset: enabled)
   Active: inactive (dead) since Wed 2019-02-13 03:29:22 UTC; 6s ago
     Docs: man:cron(8)
  Process: 699 ExecStart=/usr/sbin/cron -f $EXTRA_OPTS (code=killed, signal=TERM)
 Main PID: 699 (code=killed, signal=TERM)

Feb 13 02:25:10 ubuntu_lts systemd[1]: Started Regular background program processing
daemon.
Feb 13 02:25:10 ubuntu_lts cron[699]: (CRON) INFO (pidfile fd = 3)
Feb 13 02:25:10 ubuntu_lts cron[699]: (CRON) INFO (Running @reboot jobs)
Feb 13 03:17:01 ubuntu_lts CRON[1616]: pam_unix(cron:session): session opened for user
root by (uid=0)
Feb 13 03:17:01 ubuntu_lts CRON[1616]: pam_unix(cron:session): session closed for user
root
Feb 13 03:29:22 ubuntu_lts systemd[1]: Stopping Regular background program processing
daemon...
```

starting a service

```
$ sudo systemctl start cron
$ systemctl status cron
● cron.service - Regular background program processing daemon
   Loaded: loaded (/lib/systemd/system/cron.service; enabled; vendor preset: enabled)
   Active: active (running) since Wed 2019-02-13 03:29:36 UTC; 6s ago
     Docs: man:cron(8)
 Main PID: 1660 (cron)
    Tasks: 1 (limit: 1110)
   CGroup: /system.slice/cron.service
           └─1660 /usr/sbin/cron -f

Feb 13 03:29:36 ubuntu_lts systemd[1]: Started Regular background program processing daemon.
Feb 13 03:29:36 ubuntu_lts cron[1660]: (CRON) INFO (pidfile fd = 3)
Feb 13 03:29:36 ubuntu_lts cron[1660]: (CRON) INFO (Skipping @reboot jobs -- not system
startup)
```

journald/journalctl

one unit is journald

- integrated logging for all units
- journalctl used to control it

examples:

- journalctl — shows journal oldest->newest
- journalctl -r — reverses order
- journalctl -u ssh — -u == unit
- journalctl —list-boots

By default only retains messages from current boot

/etc/systemd/journald.conf to change

- #Storage=auto to Storage=persistant