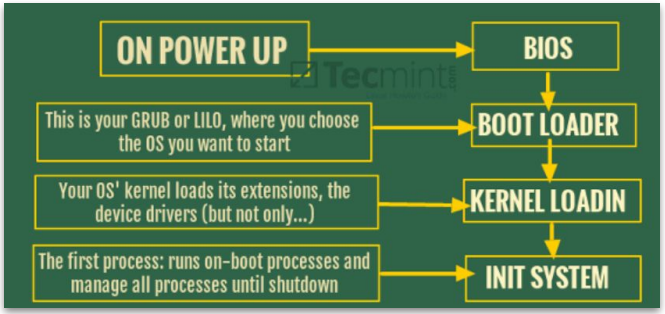
System and Network Engineering - Lecture 9

$ Services and Daemons

System boot

- System boot order



- Init system (PID 1)

```
saltanov@UbuntuPC:~$ ps -ef | head -5
UID          PID    PPID  C STIME TTY          TIME CMD
root           1       0  1 21:21 ?        00:00:01 /sbin/init splash
```

- Different types of **Init** (launchd, SysV, systemd etc)

Systemd

Systemd - is a software suite that provides an array of system components for Linux operating systems. In particular, it provides parallelization capabilities, activation for starting services, offers on-demand starting of daemons, keeps track of processes using Linux control groups, maintains mount and automount points, and implements dependency-based service control logic.

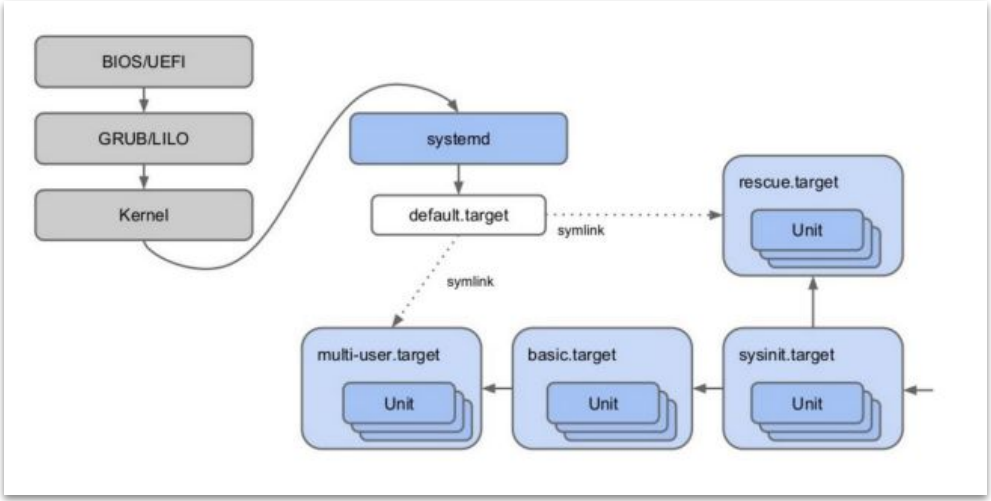
```
saltanov@UbuntuPC:~$ ls -al /sbin/init
lrwxrwxrwx 1 root root 20 Sep  9 22:47 /sbin/init -> /lib/systemd/systemd
```

Systemd's core components include the following:

- **systemd** is a system and service manager for Linux operating systems.
- **systemctl** is a command to introspect and control the state of the systemd system and service manager
- **systemd-analyze** may be used to determine system boot-up performance statistics and retrieve other state and tracing information from the system and service manager.

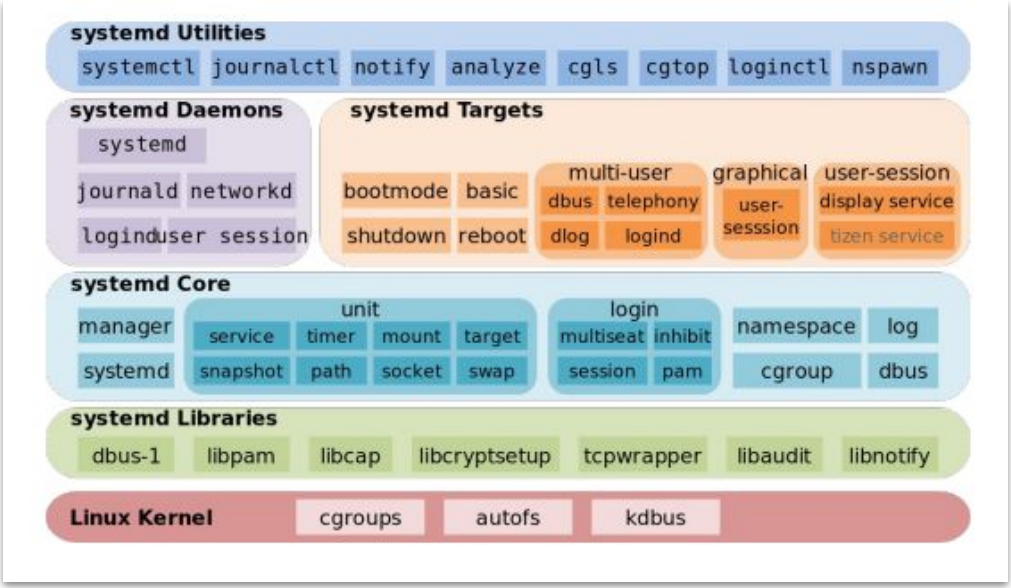
Systemd

- Prepares system
  - start services
  - mount filesystem
- Controlling resources with cgroups
- Adopts orphaned processes
- Provides powerful management tools (systemctl, etc.)



Systemd components

- `/lib/systemd` - place for files which are default and pre-installed
- `/etc/systemd` - place for user-defined custom configurations
- `man systemd.unit` - main helper for systemd units configuration



Systemd: main abstractions

Units - basic object that systemd manages and acts upon it.
Can be as following:

- ❏ Targets (.target)
- ❏ Services (.service)
- ❏ Slices (.slice)
- ❏ Mount points (.mount)
- ❏ others

```
NAME
       systemd.unit - Unit configuration

SYNOPSIS
       service.service, socket.socket, device.device, mount.mount, automount.automount,
       swap.swap, target.target, path.path, timer.timer, slice.slice, scope.scope
```

```
A unit file is a plain text ini-style file that encodes information about a service, a
socket, a device, a mount point, an automount point, a swap file or partition, a start-up
target, a watched file system path, a timer controlled and supervised by systemd(1), a
resource management slice or a group of externally created processes. See
systemd.syntax(7) for a general description of the syntax.
```

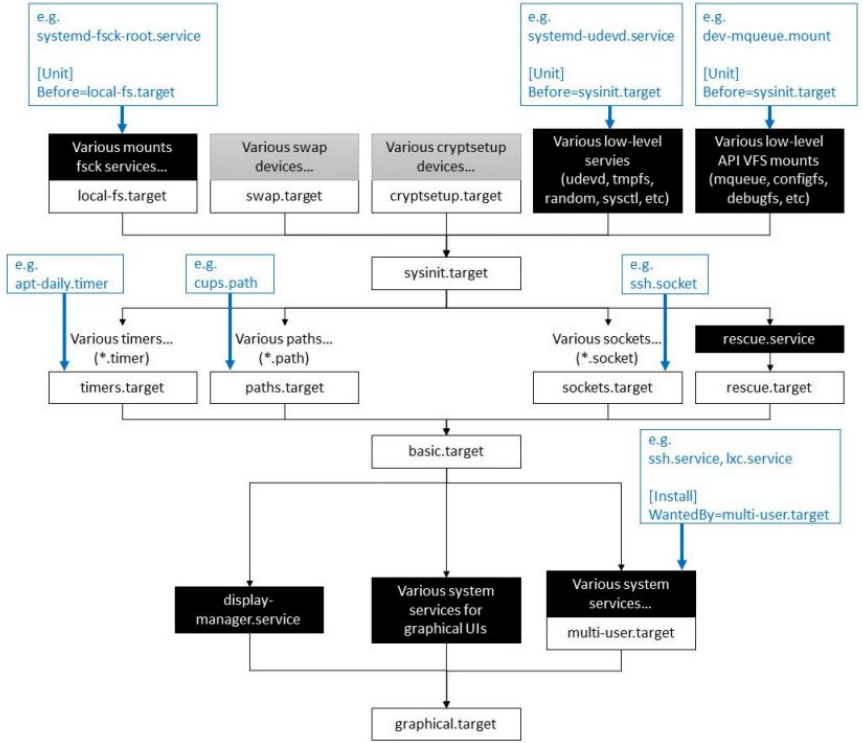
Systemd: targets

- Unit configuration files are located in Debian-based system:
  - `/usr/lib/systemd/system/` - units of installed packages
  - `/run/systemd/system/` - runtime units
  - `/etc/systemd/system/` - local configuration by administrator

Systemd: targets

.target - a unit configuration file which is used for grouping units via dependencies and as standardized synchronization points during start-up.

- ❏ targets serve a similar purpose as runlevels but much more fine grained.
- ❏ some targets are implemented by inheriting all of the services of another target and adding additional services to it.



Understanding runlevels vs systemd targets

runlevel:

- A runlevel is essentially a set of capabilities or running services that you can pre-define and set the system to boot to so you have a predictable set of services.
- It defines the number and type of daemons that are loaded into memory and executed by the kernel on a particular system
- Only one runlevel can be "active" at a time
- It was used before systemd appeared

targets:

- In systemd, targets are the new runlevels.
- systemd provides a compatibility layer that maps runlevels to targets
- systemd can activate multiple targets concurrently not as one runlevel at a time

```
saltanov@UbuntuPC:/sys/fs/cgroup/system.slice/cron.service$ ls -al /lib/systemd/system/runlevel*
lrwxrwxrwx 1 root root   15 Sep  9 22:47 /lib/systemd/system/runlevel0.target -> poweroff.target
lrwxrwxrwx 1 root root   13 Sep  9 22:47 /lib/systemd/system/runlevel1.target -> rescue.target
lrwxrwxrwx 1 root root   17 Sep  9 22:47 /lib/systemd/system/runlevel2.target -> multi-user.target
lrwxrwxrwx 1 root root   17 Sep  9 22:47 /lib/systemd/system/runlevel3.target -> multi-user.target
lrwxrwxrwx 1 root root   17 Sep  9 22:47 /lib/systemd/system/runlevel4.target -> multi-user.target
lrwxrwxrwx 1 root root   16 Sep  9 22:47 /lib/systemd/system/runlevel5.target -> graphical.target
lrwxrwxrwx 1 root root   13 Sep  9 22:47 /lib/systemd/system/runlevel6.target -> reboot.target
```

Understanding runlevels vs systemd targets

SysV Runlevel	systemd Target	Notes
0	runlevel0.target, poweroff.target	Halt the system.
1, s, single	runlevel1.target, rescue.target	Single user mode.
2, 4	runlevel2.target, runlevel4.target, multi-user.target	User-defined/Site-specific runlevels. By default, identical to 3.
3	runlevel3.target, multi-user.target	Multi-user, non-graphical. Users can usually login via multiple consoles or via the network.
5	runlevel5.target, graphical.target	Multi-user, graphical. Usually has all the services of runlevel 3 plus a graphical login.
6	runlevel6.target, reboot.target	Reboot
emergency	emergency.target	Emergency shell

```
saltanov@UbuntuPC:~$ runlevel
N 5
```

Systemd targets usage:

- View the current target configuration:
  - `$systemctl get-default`
- Change default target to another on boot
  - `$systemctl get-default multi-user.target`

```
saltanov@UbuntuPC:/sys/fs/cgroup/system.slice/cron.service$ systemctl set-default multi-user.target
Created symlink /etc/systemd/system/default.target → /lib/systemd/system/multi-user.target.
```

- List dependencies for the target/unit:
  - `$systemctl list-dependencies graphical.target`

```
saltanov@UbuntuPC:~$ systemctl list-dependencies graphical.target
graphical.target
● ├─accounts-daemon.service
● ├─apport.service
● ├─gdm.service
● ├─power-profiles-daemon.service
● ├─switcheroo-control.service
○ ├─systemd-update-utmp-runlevel.service
● ├─udisks2.service
● └─multi-user.target
○   ├─anacron.service
●   ├─apport.service
●   ├─avahi-daemon.service
●   ├─console-setup.service
●   ├─containerd.service
●   ├─cron.service
●   ├─cups-browsed.service
```

Systemd targets usage:

- **View enabled services for targets:**

```
saltanov@UbuntuPC:/sys/fs/cgroup/system.slice/cron.service$ ls -laR /lib/systemd/system/multi-user.target.wants/
/lib/systemd/system/multi-user.target.wants/:
total 44
drwxr-xr-x  2 root root  4096 Oct  5 13:51 .
drwxr-xr-x 29 root root 36864 Oct 13 12:39 ..
lrwxrwxrwx  1 root root    15 Aug 30 13:46 dbus.service -> ../dbus.service
lrwxrwxrwx  1 root root    15 Sep  9 22:47 getty.target -> ../getty.target
lrwxrwxrwx  1 root root    24 Aug 30 13:46 plymouth-quit.service -> ../plymouth-quit.service
lrwxrwxrwx  1 root root    29 Aug 30 13:46 plymouth-quit-wait.service -> ../plymouth-quit-wait.service
lrwxrwxrwx  1 root root    33 Sep  9 22:47 systemd-ask-password-wall.path -> ../systemd-ask-password-wall.path
lrwxrwxrwx  1 root root    25 Sep  9 22:47 systemd-logind.service -> ../systemd-logind.service
lrwxrwxrwx  1 root root    39 Sep  9 22:47 systemd-update-utmp-runlevel.service -> ../systemd-update-utmp-runleve
l.service
lrwxrwxrwx  1 root root    32 Sep  9 22:47 systemd-user-sessions.service -> ../systemd-user-sessions.service
```

- **Create custom target:**
  - Describe the target and create the file in the `/etc/systemd/system/<your_target>.target`
  - Create directory `/etc/systemd/system/<your_target>.wants`
  - Symlink with the additional services from `/lib/systemd/system/` that you wish to enable.

Systemd: cgroups

- Control groups
- “Box” for processes
- Resource accounting and limitation
- Tracking of mutli-process services
  - `systemd-cgls`
  - `systemd-cgtop`
  - `/sys/fs/cgroup`
  - `/proc/PID/cgroup`

```
saltanov@UbuntuPC:/sys/fs/cgroup/system.slice$ cd cron.service/
saltanov@UbuntuPC:/sys/fs/cgroup/system.slice/cron.service$ ll
total 0
drwxr-xr-x  2 root root 0 Oct 22 21:21 ./
drwxr-xr-x 59 root root 0 Oct 23 02:41 ../
-r--r--r--  1 root root 0 Oct 22 21:21 cgroup.controllers
-r--r--r--  1 root root 0 Oct 22 21:21 cgroup.events
-rw-r--r--  1 root root 0 Oct 22 21:21 cgroup.freeze
--w-------  1 root root 0 Oct 22 21:21 cgroup.kill
-rw-r--r--  1 root root 0 Oct 22 21:21 cgroup.max.depth
-rw-r--r--  1 root root 0 Oct 22 21:21 cgroup.max.descendants
-rw-r--r--  1 root root 0 Oct 22 21:21 cgroup.procs
-r--r--r--  1 root root 0 Oct 22 21:21 cgroup.stat
-rw-r--r--  1 root root 0 Oct 22 21:21 cgroup.subtree_control
-rw-r--r--  1 root root 0 Oct 22 21:21 cgroup.threads
-rw-r--r--  1 root root 0 Oct 22 21:21 cgroup.type
-rw-r--r--  1 root root 0 Oct 22 21:21 cpu.idle
-rw-r--r--  1 root root 0 Oct 22 21:21 cpu.max
-rw-r--r--  1 root root 0 Oct 22 21:21 cpu.max.burst
-rw-r--r--  1 root root 0 Oct 22 21:21 cpu.pressure
```

Systemd: services

- **.service** - a unit configuration file whose name ends in .service and encodes information about a process controlled and supervised by systemd.
  - Service files must include a "[Service]" section, which carries information about the service and the process it supervises.
  - `$man systemd.service`

```
lrwxrwxrwx  1 root root   31 Sep  1 18:38  sshd.service -> /lib/systemd/system/ssh.service
lrwxrwxrwx  1 root root    9 Aug 30 13:44  sudo.service -> /dev/null
drwxr-xr-x  2 root root 4096 Aug 30 13:51  sysinit.target.wants/
lrwxrwxrwx  1 root root   35 Aug 30 13:44  syslog.service -> /lib/systemd/system/rsyslog.ser
vice
drwxr-xr-x  2 root root 4096 Aug 30 13:51  timers.target.wants/
-rw-r--r--  1 root root  333 Sep 20 19:25 'var-snap-firefox-common-host\x2dhunspell.mount'
lrwxrwxrwx  1 root root   41 Aug 30 13:52  vmtoolsd.service -> /lib/systemd/system/open-vm-t
ools.service
saltanov@UbuntuPC:/etc/systemd/system$ cat sshd.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
```

.service

Handling dependencies - with systemd, dependencies can be resolved by designing the unit files correctly:

- The most typical case is that the `unit A` requires the `unit B` to be running before A is started.
  - In that case add `Requires=B` and `After=B` to the [Unit] section of A.
  - If the dependency is optional, add `Wants=B` and `After=B` instead.
- Note that `Wants=` and `Requires=` do not imply `After=`, meaning that if `After=` is not specified, the two units will be started in parallel.
- Dependencies are typically placed on services and not on targets. E.g., `network.target` is pulled in by whatever service configures your network interfaces, therefore ordering your custom unit after it is sufficient since `network.target` is started anyway

```
[Install]
WantedBy=default.target
```

```
ubuntu@kirill-saltanov:/etc/systemd/system$ sudo systemctl enable greetings.service
Created symlink /etc/systemd/system/default.target.wants/greetings.service → /etc/systemd/system/greetings.service.
```

Managing unit files

View the unit:

- `$systemctl cat`

```
saltanov@UbuntuPC:/etc/systemd/system$ systemctl cat sshd
# /lib/systemd/system/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
```

Replacement current unit can be done in the following ways:

- To replace the unit file `/usr/lib/systemd/system/<unit>`, create the file `/etc/systemd/system/<unit>` and reenable the unit to update the symlink
- `$systemctl edit` - this opens the file `/etc/systemd/system/unit.d/override.conf` in your text editor (creating it if necessary) and automatically reloads the unit when you are done editing.
- `$systemctl edit -full` - opens file directly and overwrites it
- `$systemctl daemon-reload` - reloads current configuration of units for systemd

Interacting with Systemd

- `$systemctl`
  - `start/stop`
  - `enable/disable`
  - `reload`
  - `status`
  - `list-unit-files` - check status for all of them
- `Journalctl` - systemd journal
  - `-u` - for specific unit
  - `-b` - from the last boot

```
saltanov@UbuntuPC:/etc/systemd/system$ systemctl list-unit-files
UNIT FILE                                                  STATE           VENDOR PRESET
proc-sys-fs-binfmt_misc.automount                          static          -
-.mount                                                    generated       -
boot-efi.mount                                             generated       -
dev-hugepages.mount                                        static          -
dev-mqueue.mount                                           static          -
proc-sys-fs-binfmt_misc.mount                              disabled        disabled
run-vmblock\x2dfuse.mount                                  enabled         enabled
snap-bare-5.mount                                          enabled         enabled
snap-core20-1611.mount                                     enabled         enabled
snap-core20-1623.mount                                     enabled         enabled
snap-firefox-1943.mount                                    enabled         enabled
snap-firefox-1993.mount                                    enabled         enabled
snap-gnome\x2d3\x2d38\x2d2004-115.mount                    enabled         enabled
snap-gnome\x2d3\x2d38\x2d2004-119.mount                    enabled         enabled
snap-gtk\x2dcommon\x2dthemes-1535.mount                    enabled         enabled
snap-hunspell\x2ddictionaries\x2d1\x2d7\x2d2004-2.mount    enabled         enabled
snap-snap\x2dstore-592.mount                               enabled         enabled
snap-snap\x2dstore-599.mount                               enabled         enabled
snap-snapd-17029.mount                                     enabled         enabled
snap-snapd-17336.mount                                     enabled         enabled
snap-snapd\x2ddesktop\x2dintegration-14.mount              enabled         enabled
sys-fs-fuse-connections.mount                              static          -
sys-kernel-config.mount                                    static          -
sys-kernel-debug.mount                                     static          -
sys-kernel-tracing.mount                                   static          -
var-snap-firefox-common-host\x2dhunspell.mount             enabled         enabled
```