

Systems 3

OS Design

Marcel Waldvogel

(Handout)

Department of Computer and Information Science
University of Konstanz

Winter 2019/2020

Who is individual process #1?



Photo by The Circus (CC BY 3.0)

Chapter Goals



Boot process

Basic overview of the Linux boot process.

Actor	Actions
BIOS/UEFI	test hardware, execute Master Boot Record
MBR	load and execute Grand Unified Bootloader
GRUB	load kernel and initrd
Kernel	mount root file system and start init
Init	determine run level and start
Runlevel	start various services

Daemons

A daemon is a process that

- runs in background (detached from user session)
- ends traditionally with **d** (e.g. `syslogd`, `sshd`)
- is often a child of the `init` process

Daemons in C

To create a daemon in C on a Linux system the following steps have to be executed:

- 1 Fork and exit parent to run in background
- 2 Create a new session to detach from controlling terminal (`Ctrl-C`)
- 3 Handle signals
- 4 Fork and exit session leader (never have a controlling terminal again)
- 5 Change file mode mask (optional)
- 6 Change working directory (optional)
- 7 Close all open file descriptors
- 8 If privileged, do privileged operations now and then drop privileges
- 9 Prepare logging (see `syslog(3)`)

```

1 pid_t pid;
2
3 // Fork off the parent process
4 pid = fork();
5
6 // An error occurred
7 if (pid < 0)
8     exit(EXIT_FAILURE);
9
10 // Success: Let the parent terminate
11 if (pid > 0)
12     exit(EXIT_SUCCESS);
13
14 // Success: The child process becomes
15 // the session leader
16 if (setsid() < 0)
17     exit(EXIT_FAILURE);
18
19 // Catch, ignore and handle signals
20 signal(SIGCHLD, SIG_IGN);
21 signal(SIGHUP, SIG_IGN);

```

```

22 // Fork again
23 pid = fork();
24
25 // An error occurred
26 if (pid < 0)
27     exit(EXIT_FAILURE);
28
29 // Success: Let the parent terminate
30 if (pid > 0)
31     exit(EXIT_SUCCESS);
32
33 // Set new file permissions
34 umask(0);
35
36 // (Change the working directory)
37 chdir("/");
38
39 // Close all open file descriptors
40 for (int x = sysconf(_SC_OPEN_MAX);
41      x >= 0; x--)
42     close(x);

```

Example taken from <https://github.com/pasce/daemon-skeleton-linux-c>

Daemons in C

Of course you could also just use `daemon(3)` ;-)

Init Systems

Daemons are rarely started manually, but instead by an init system, e.g.

- System V
- SystemD
- Upstart
- OpenRC
- runit

History

- 1 `/etc/rc`
- 2 `/etc/rc?.d/[SK]*`
- 3 Dependency graphs

Controlling with systemd

Systemd can be easily controlled by `systemctl(1)`. For example to get the status of a service just call:

```
1 $ systemctl status sshd
2 -> ssh.service - OpenBSD Secure Shell server
3 Loaded: loaded (/lib/systemd/system/ssh.service; enabled; vendor preset: enabled)
4 Active: active (running) since Mon 2019-12-09 11:19:22 CET; 3 days ago
5 Docs: man:sshd(8)
6       man:sshd_config(5)
7 Process: 1505 ExecStartPre=/usr/sbin/sshd -t (code=exited, status=0/SUCCESS)
8 Main PID: 1542 (sshd)
9 Tasks: 1 (limit: 4915)
10 Memory: 4.1M
11 CGroup: /system.slice/ssh.service
12        1542 /usr/sbin/sshd -D
```

Creating a systemd service

To create your own service, just add a file like `webserver.service` to `/etc/systemd/system/` with the following content:

```
1 [Unit]
2 Description=Awesome webserver
3 After=network.target
4 StartLimitIntervalSec=0
5
6 [Service]
7 Type=simple
8 Restart=always
9 RestartSec=1
10 User=www
11 ExecStart=/opt/webserver/server
12
13 [Install]
14 WantedBy=multi-user.target
```

Question: Why is the file located in `/etc`?

Filesystem Hierarchy Standard (FHS)¹

The Filesystem Hierarchy Standard (FHS) defines the directory structure and directory contents in Linux distributions.

/bin binaries for all users

/boot boot loader files

/dev device files

/etc System-wide configuration files

/lib Libraries for binaries in /bin and /sbin

/media Mount point for removable media

/srv Data served by the system

/tmp Temporary files

/usr User System Resources

/var Variable data

¹https://refspecs.linuxfoundation.org/FHS_3.0/fhs/index.html

mount(8) and umount(8)

The mount command serves to attach the filesystem found on some device to the big file tree.

- 1 Find your drive

```
lsblk
```

- 2 Create your mount point

```
sudo mkdir /media/my-usb-drive
```

- 3 Mount your device

```
sudo mount /dev/sdb1 /media/my-usb-drive
```

- 4 Eject your device

```
sudo umount /media/my-usb-drive (or sudo umount  
/dev/sdb1)
```

- 5 Delete your mount point

```
sudo rmdir /media/my-usb-drive
```

Process information pseudo-filesystem

The `/proc(5)` filesystem is a pseudo-filesystem which provides an interface to kernel data structures.

Some examples are:

`/proc/uptime` seconds since kernel was booted

`/proc/meminfo` summary of memory usage

`/proc/cpuinfo` information about the CPU

`/proc/cmdline` kernel boot options

`/proc/PID/cmdline` command that originally started the process

`/proc/PID/fd` containing all file descriptors

Monitoring

For simple open source tools for Linux system monitoring:

- top
- atop²
- htop³
- glances⁴

When managing multiple systems, use tools such as Nagios.

²<https://www.atoptool.nl>

³<https://hisham.hm/htop/>

⁴<https://nicolargo.github.io/glances/>

Linux Performance Tools

