Lancaster University

A Linux Introduction

Week(21) L2
Ibrahim Aref
2024/2025

Based on Debian Reference by Osamu Aoki

https://www.debian.org/doc/manuals/debian-reference/index.en.html

Quick Revision – Last Lecture(I)

Lancaster University

- In operating systems (OS), process scheduling is the process of deciding which process to run on the CPU and for how long, ensuring efficient and fair resource allocation among multiple processes.
- Process States: new, ready, running, waiting, Terminated, Swapped out and waiting: (suspended & waiting), Swapped out and blocked(suspended & blocked) .
- Scheduling Algorithms:
  - First-Come, First-Served (FCFS) – Non-preemptive (Cooperative), runs in order of arrival.
  - Round Robin (RR) – Each process gets a fixed time slice (time quantum).

Example – RR (I)

Lancaster University

We are working with a Round Robin (RR) scheduling algorithm with a **time quantum of 4 ms , (all numbers above in ms)**. Determine the execution order, completion time, turnaround time, and waiting time for each process.

Process	Arrival Time	Burst Time
P1	0	5
P2	1	3
P3	2	8
P4	3	6

Example – RR (II)

Lancaster University

- Execution Order:
  - P1 **starts at** 0 ms and **runs for** 4 ms (**remaining**: 1 ms).
  - P2 **starts at** 4 ms and **runs for** 3 ms (**finishes**).
  - P3 **starts at** 7 ms and **runs for** 4 ms (**remaining**: 4 ms).
  - P4 **starts at** 11 ms and **runs for** 4 ms (**remaining**: 2 ms).
  - P1 **resumes at** 15 ms and **runs for** 1 ms (**finishes**).
  - P3 **resumes at** 16 ms and **runs for** 4 ms (**finishes**).
  - P4 **resumes at** 20 ms and **runs for** 2 ms (**finishes**).

P1	P2	P3	P4	P1	P3	P4
0	4	7	11	15	16	20	22

Example – RR (III)

Lancaster University

- Completion, Turnaround, and Waiting Times:

Process	Arrival Time	Burst Time	Completion Time	Turnaround Time (CT - AT)	Waiting Time (TAT - BT)
P1	0 ms	5 ms	**15 ms**	15 - 0 = 15 ms	15 - 5 = 10 ms
P2	1 ms	3 ms	**7 ms**	7 - 1 = 6 ms	6 - 3 = 3 ms
P3	2 ms	8 ms	**20 ms**	20 - 2 = 18 ms	18 - 8 = 10 ms
P4	3 ms	6 ms	**22 ms**	22 - 3 = 19 ms	19 - 6 = 13 ms

- **Average Turnaround Time** = (15+6+18+19)/4 = 14.5 ms
- **Average Waiting Time** = (10+3+10+13)/4 = 9 ms

The Plan

Weeks	Lecture	Topic	
19	L1	Signal digitisation - Sound	Done
	L2	Introduction to OS (1)	Done
20	L1	Introduction to OS (2)	Done
	L2	Memory management and virtualization	Done
Summer Term			
21	L1	Process scheduling	Done
	L2	Linux Introduction	**This Lecture**
22	L1	Introduction to cloud virtualization	
	L2	Introduction to graphics	
23	L1	GPU -shaders	
	L2	Web Assembly	
24	L1	Introduction to Networking (1)	
	l2	Introduction to Networking (2)	

In this lecture

Lancaster University

- Linux.
- Explore how an operating system works (Debian):
  - Boot Process
  - Initialization
  - Filesystem
  - Shell
  - Running processes

Linux

Lancaster University

- A UNIX operating system clone for a number of platforms
- A free UNIX version developed primarily by Linus Torvalds
- Developed on MINIX using the GNU C compiler
- An OS is more than just the kernel —> Linux Distribution
  - Linux kernel
  - GNU tools (e.g. gcc, gdb) and libraries
  - Additional software, documentation,
  - Window system, desktop environment
  - Package management system
- Example distribution: Debian

Boot Process

Lancaster University

- Sequence of events that occurs when a computer starts or restarts, loading the OS from storage into memory and preparing it for user interaction.
  1. Power-On Self-Test (POST)
  2. Boot Device Selection
  3. Bootloader
  4. Kernel Loading
  5. System Initialization
  6. User Authentication

Boot Process in Debian (I)

Lancaster University

- **1- BIOS/UEFI Initialization**
  - When the system is powered on, the BIOS or UEFI (modern systems) initializes hardware components (CPU, RAM, disk controllers, etc.).
  - The firmware looks for a bootable disk and hands over control to the bootloader.
- **2- Bootloader (GRUB)**
  - Debian typically uses GRUB (GRand Unified Bootloader).
  - There are many boot loaders (e.g. GRUB2, lilio, syslinux, …)
  - Configured via /boot/grub/grub.cfg (simplified)
  - GRUB loads the kernel and an initial RAM disk (initrd/initramfs).
  - If multiple OSes are installed, GRUB presents a boot menu.

Boot Process in Debian (II)

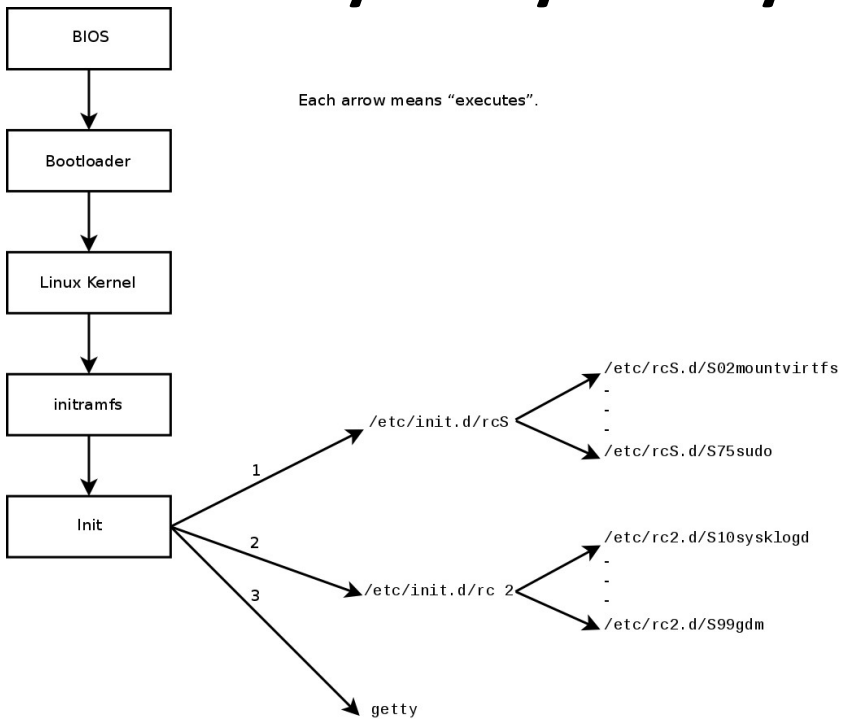
Lancaster University

- **3- Linux Kernel Initialization**
  - The kernel is loaded into memory and starts executing.
  - It initializes CPU scheduling, memory management, and device drivers.
  - The **initramfs** (temporary root filesystem) is used to mount essential system components before the main filesystem is available.
- **4- Systemd Initialization (or SysV init)**
  - Once the kernel is running, it starts Systemd (or SysV init in older Debian versions).
  - Systemd manages system services using unit files (located in /etc/systemd/system/).
  - It determines the default target (runlevel) (e.g., graphical mode or multi-user mode).

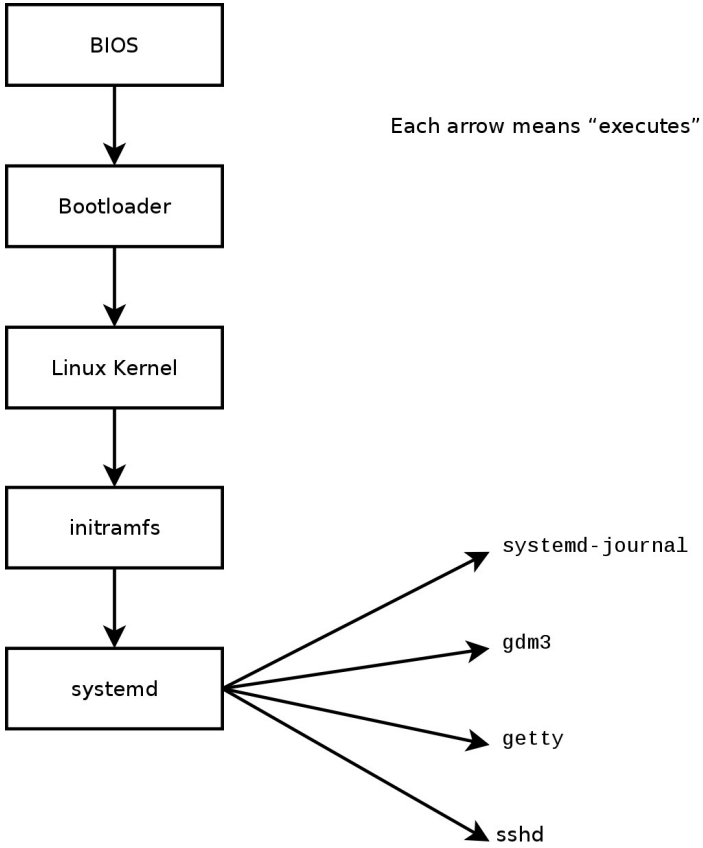
Boot Process in Debian (III)

Lancaster University

SysV-style vs Systemd



Each arrow means “executes”.



Each arrow means “executes”.

Boot Process in Debian (IV)

Lancaster University

- **5- Mounting Filesystems**
  - The root filesystem (/) is mounted.
  - Additional filesystems (e.g., /home, /var, /boot, etc.) are mounted as per /etc/fstab.
- **6- Starting System Services**
  - Systemd (or SysV init) starts services like:
    - Networking (NetworkManager, systemd-networkd)
    - Login Services (getty, SSH daemon)
    - Display Manager (if GUI is enabled, e.g., GDM, LightDM)

Boot Process in Debian (V)

Lancaster University

- **7- User Login and Shell/GUI**
  - The system reaches the target run level:
    - CLI mode: Shows a login prompt (for non-GUI setups).
    - GUI mode: Loads a Display Manager (e.g., GDM, LightDM).
  - The user logs in and starts using the system.

Filesystem

Lancaster University

- filesystem is a method used by OS to organize, store, and manage files on a storage device.
- In UNIX operating systems, files are organised into directories.
- All files and directories are arranged in one big tree rooted at `/`
- Files and directories can be spread out over several devices
    - `mount` attaches the filesystem found on a device to the big file tree
    - `umount` detaches filesystems again

- Key directories
    - `/` - the root directory
    - `/etc/` - system wide configuration files
    - `/var/log/` - system log files
    - `/home/` - all the home directories for all non-privileged users
    - `/dev` - device abstractions (every device is represented as file in unix!)

Linux Filesystem Structure (Hierarchy)

Lancaster University

- Linux follows a hierarchical directory structure, starting from the root (/) directory. Some important directories include:

Directory	Description
/	Root directory (top-level)
/home	User home directories (e.g., /home/user)
/bin	Essential binaries (e.g., ls, cp, rm)
/etc	System configuration files
/var	Logs, databases, and variable data
/tmp	Temporary files (cleared on reboot)
/boot	Bootloader files (GRUB, kernel)
/dev	Device files (e.g., /dev/sda1 for a hard disk)
/proc	Virtual filesystem for system processes
/sys	Virtual filesystem for kernel-related information
/mnt	Temporary mount point for external devices
/media	Auto-mount point for USB, CDs, etc.

Filesystem Operations

Lancaster University

- Mounting and Unmounting Filesystems
- Mount a filesystem:
  - **`sudo mount /dev/sdb1 /mnt`**

- Unmount a filesystem
  - **`sudo umount /mnt`**

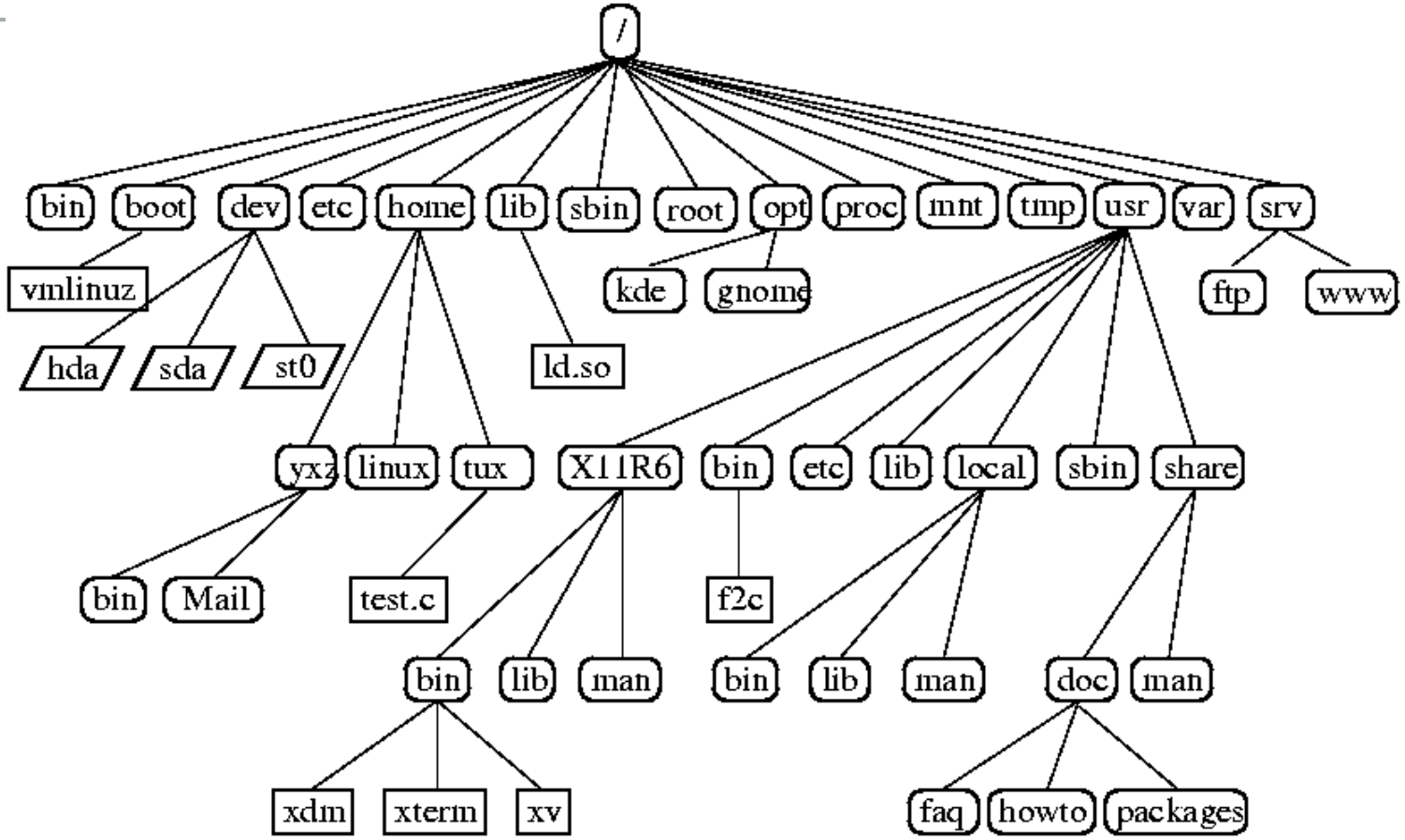
Checking and Formatting Filesystems

Lancaster University

- Check disk space usage:
  - **`df -h`**
- Check inode usage
  - **`df -iList`**
- mounted filesystems:
  - **`Mount`**
- Format a disk as ext4:
  - **`sudo mkfs.ext4 /dev/sdb1`**
- Check and repair a filesystem:
  - **`sudo fsck /dev/sda1`**

Filesystem – Linux structure

Lancaster University



Filesystem Permissions

Lancaster University

- Linux filesystem permissions control how users and groups can access and modify files and directories. These permissions ensure security and proper access control.
- Every file and directory in Linux has three permission sets:

Permission Type	Symbol	Numeric Value	Description
Read	r	4	Allows viewing or reading the file
Write	w	2	Allows modifying or deleting the file
Execute	x	1	Allows running a file (script/program) or accessing a directory

Permissions are assigned to three categories:

- Owner (User - u): The user who owns the file.
- Group (g): Users in the same group as the file owner.
- Others (o): All other users.

Viewing File Permissions

Lancaster University

To check file permissions, use the ls -l command:

```
ls -l
```

Example

```
-rwxr-xr-- 1 user group  1234 Mar 26 12:00 example.sh
```

Breakdown:

`-rwxr-xr---` → Regular file (d for directory, l for symbolic link).
rwx → Owner has read (r), write (w), execute (x) permissions.
r-x → Group has read (r) and execute (x) permissions.
r-- → Others have read (r) permission only.
user → Owner of the file.
group → Group of the file.
1234 → File size in bytes.
Mar 26 12:00 → Last modified date.
example.sh → File name.

Shell

Lancaster University

- A shell is any program that users employ to type commands
- There are many different shells available: `bash, tcsh, dash, csh, …`
- The shell is …
  - an interactive command language
  - a scripting programming language,
  - is used by the operating system to control (shell script) the system

The shell is used to start/stop programs:

```
$ echo "hello" &
[1] 18908
$ hello
[1]+  Done echo “hello”
$
```

Repetitive tasks can be implemented as script:

```
#!/bin/bash
PROGDIR=/opt/myprog
BACKUPDIR=/opt/backup
mkdir -p $BACKUPDIR
cp $PROGDIR/*.c $BACKUPDIR/
```

Processes in Debian

Lancaster University

- In Debian (or any Linux-based operating system), processes refer to Instances of executing programs.
- Each process has a unique Process ID (PID) and can have different states such as **running, sleeping, stopped, or zombie**.
- Processes can be managed and monitored using commands like **`ps`, `top`, `htop`, and `kill`**.
- Example:
  - Opening a text editor like **gedit** creates a process, and running a command like **ls** also creates a process.

Types of Processes in Debian

Lancaster University

1. **Foreground Processes:** Started by users in the terminal and interact directly.
   - E.g: Running nano file.txt keeps the process in the foreground.
2. **Background Processes:** Run in the background without user interaction.
   - E.g: Running ./script.sh runs the script in the background.
3. **Daemon Processes:** System processes that run in the background, usually started at boot.
   - E.g: cron, sshd, apache2.
4. **Zombie Processes:** Completed but still occupying process table entries.
   - These should be automatically removed by the system.
5. **Orphan Processes:** Parent processes that have terminated, but their child processes continue running under init (PID 1).

Processes – tree view

Lancaster University

- In Debian, the running processes can be viewed in a tree structure using **`pstree`** command
- It displays processes in a hierarchical tree format.

```
$ pstree
systemd─┬─acpid
        ├─agetty
        ├─collectd───11*[{collectd}]
        ├─cron
        ├─helpermonkeys───20*[{helpermonkeys}]
        ├─mdadm
        ├─monit───{monit}
        ├─netatalk─┬─afpd
        │          ├─cnid_metad
        │          └─{netatalk}
        ├─rpc.idmapd
        ├─rpc.statd
        ├─rpcbind
        ├─rrdcached───6*[{rrdcached}]
        ├─sshd───sshd───sshd───bash───pstree
        ├─systemd-journal
        ├─systemd-logind
        └─systemd-udevd
```

Processes - details

Lancaster University

```
$ps -u --ppid 2 -p 2 --deselect
USER        PID %CPU %MEM    VSZ   RSS TTY      STAT START   TIME COMMAND
root          1  0.0  0.2  28804  5212 ?        Ss   14:27   0:04 /sbin/init
root        198  0.0  0.1  29924  4052 ?        Ss   14:27   0:01 /lib/systemd/systemd-journald
root        209  0.0  0.1  40956  3248 ?        Ss   14:27   0:00 /lib/systemd/systemd-udevd
root        411  0.0  0.1  13432  2216 ?        Ss   14:27   0:00 /sbin/mdadm --monitor --scan
systemd+    507  0.0  0.1 102104  2344 ?        Ssl  14:27   0:00 /lib/systemd/systemd-
timesyncd
root        667  0.0  0.1  37080  2696 ?        Ss   14:27   0:00 /sbin/rpcbind -w
statd       680  0.0  0.1  37280  2732 ?        Ss   14:27   0:00 /sbin/rpc.statd
root        694  0.0  0.0  23356   208 ?        Ss   14:27   0:00 /usr/sbin/rpc.idmapd
…
```

Process ID - Used to identify a process in the system.
User - Identifies the owner of a process.

Summary (I)

Lancaster University

Debian Boot Process

1. BIOS/UEFI → Initializes hardware.
2. Bootloader (GRUB) → Loads kernel and initramfs.
3. Kernel Initialization → Starts system core components.
4. Init System (Systemd) → Manages services and targets.
5. Mount Filesystems → Loads root and other partitions.
6. Start Services → Networking, logins, and system daemons.
7. User Login → CLI or GUI session starts.

Summary (II)

Lancaster University

- Linux uses a hierarchical filesystem structure starting from /.
- **ext4** is the most common filesystem, but others like XFS, Btrfs, and ZFS exist.
- **Filesystems** can be mounted, unmounted, checked, and formatted using CLI commands.
- Processes is an instances of executing programs. It has a unique Process ID (PID) and can have different states such as running, sleeping, stopped, or zombie.