

Linux Theorie

Sessie 2



File System
File Permissions
Directory Structure
“In Linux everything is a file”



File System

Wat is een file system?

In computers wordt een file-system gebruikt om te controleren hoe data wordt opgeslagen en opgeroepen.

Zonder file-system zou de informatie die opgeslagen wordt op een opslag-medium, zoals bvb een hard-disk, één grote ononderbroken hoop data zijn zonder manier om te weten waar een bestand stopt en waar een volgende begint.

.... ?!?



Opslag-medium

“HDD” - hard disk drive



Opslag-medium

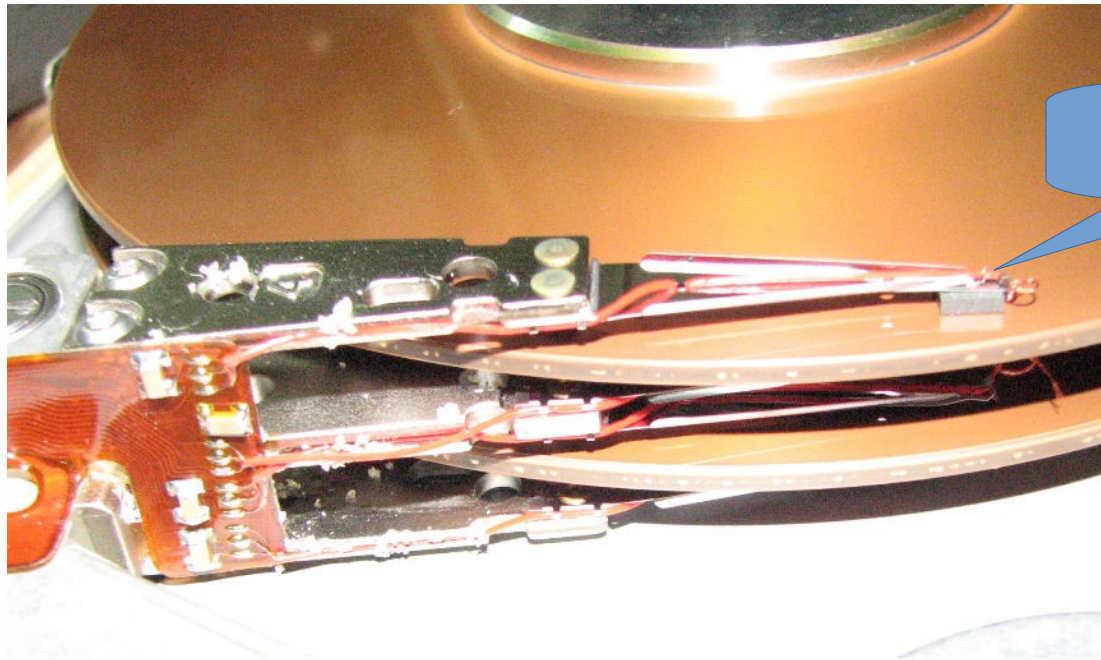
Hoe werkt een HDD?



Schijfjes met magnetische laag

Opslag-medium

Hoe werkt een HDD?

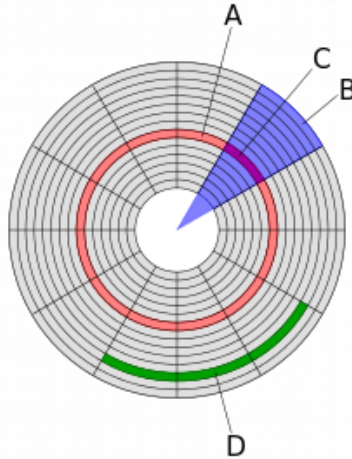


Lees-schrijf kop

Opslag-medium

HDD als block-device

Hoe terug vinden wat waar staat op deze schijfjes?



- Schijfjes opdelen in 'blokjes'

→ dit is wat we noemen 'formatteren'

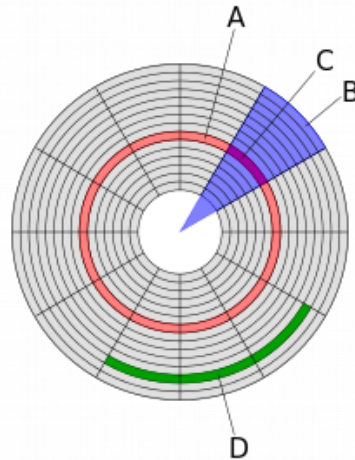
schrijven van markeringen
op de schijfjes waar de blokjes
starten en waar ze stoppen

→ linux commando **dd**

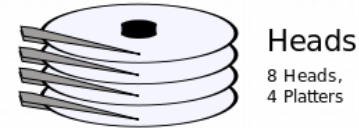
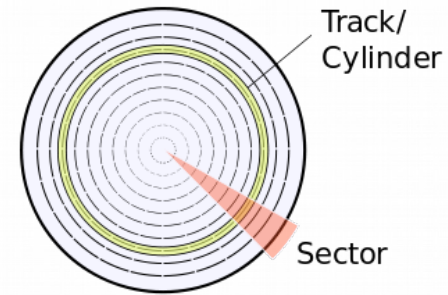
```
sudo dd if=/dev/zero of=/dev/sdc bs=512  
(count=x)
```

Opslag-medium

HDD als block-device



A → Track of Cylinder
B → Sector
C → Block



Opslag-medium → File System

'Read' - 'Write' voorbeeld

- File1 staat op cylinder1, block 1 tot 10
- File2 staat op cylinder1, block 11 tot 33

Wat nu als er files bijkomen of gewist worden.. ?

- File1 wordt gewist, block 1 tot 10 is nu leeg
- File3 is een stuk groter en komt op...
cylinder1, block 1 tot 10 en 34 tot 90
én nog een stukje op cylinder2...

Veronderstel nu honderden lees-schrijf operaties...



Opslag-medium → File System

'Read' - 'Write' voorbeeld

=> nood aan een systeem die kan bijhouden:

- Welke files waar staan
- Volgorde van de stukjes...
- Waar er lege plaatsten zijn op disk
- Waar eventueel onbruikbare plekken zijn
- Taken als 'Defragmenting' kan uitvoeren (automatisch)



File Systems

**De verzameling van alle voorgaande taken
is wat men een file system noemt**

- Verschillende types mogelijk
- Bij Microsoft:
 - MSDOS-FAT
 - NTFS
- Bij linux zijn er een hele reeks:
 - EXT2, EXT3, EXT4
 - REISER-FS
 - XFS, ZFS, BTRFS
 - JFS, ...

File Systems

Waarom meerdere?

EXT2:

- Is een goed en snel file-system...
- Maar wat als de stroom uitvalt?
- Schrijven naar disk kan tijdens schrijven onderbroken worden.
- Wat is er nu wel en niet correct geschreven?
- EXT2 gaat na stroomuitval ALLES van de hele disk controleren
→ dit kan héél lang duren...

File Systems

Waarom meerdere?

EXT3:

- Lange hersteltijd is onaanvaardbaar (bvb webserver → alle site's onbereikbaar!)
- Ext3 lost dit op door gebruik van een 'journal'
- Na power out enkel controle nodig v/h journal

File Systems

Waarom meerdere?

EXT3:

- Hoe werkt zo'n journal nu ?
 - Telkens een file bewaard moet worden wordt éérs in het journal geschreven welke file en waar ze moet komen
 - Als het bewaren gelukt is wordt opnieuw in het journal geschreven dat de file ok is.



File Systems

Waarom meerdere?

EXT3:

- Nadelen:
 - 3 schrijfoperaties dus trager...
 - 32 TBytes = max volume v/h File-system



File Systems

Waarom meerdere?

EXT4:

- Deze nadelen lost ext4 merendeels op
- Performantie is iets hoger dan ext3
- Max formaat file-system is 1 Exabytes

1 ExaBytes = 1000 PetaBytes
= 1000000 TerraBytes
= 1000000000 GigaBytes

File Systems

Waarom meerdere?

REISER-FS:

- Online resizing → enkel groeien (zie partities)
- ReiserFS is sneller dan ext2 & ext3
- Nadelen
 - Geen defragmentatie tools voor het ReiserFS file system

File Systems

Waarom meerdere?

BTRFS:

- *“butter fs”*
- Is een FS gebaseerd op copy-on-write (COW)
- Pooling (meerdere disks → hotswap hotadd)
- Snapshots
- Journal met checksums

File Systems

Waarom meerdere?

BTRFS:

- Wat is copy-on-write ?
 - kopies van bestanden
 - enkel referentie gemaakt naar origineel
 - Bij modificaties
 - enkel wijzigingen worden geschreven op de plaats waar schrijfkop nu is



File Systems

Waarom meerdere?

BTRFS:

- Nadelen van 'COW'
 - Extra overhead → bijhouden wat waar is
 - Extra fragmentatie

File Systems

Waarom meerdere?

ZFS:

- Sun Micro Systems
- Efficiënte data compressie
- Mogelijkheid om “Snapshots” te nemen
- Permanente controle op data integriteit
- Automatic repair
- Nadelen:
 - Wel controle maar weinig preventie tegen data corruption
 - Sommige RAID configuraties zijn niet mogelijk met ZFS

File Systems

Waarom meerdere?

Intermezzo.. Wat is RAID ?

- Redundant Array of Inexpensive Disks
(nu ook wel *Independant i.p.v Inexpensive*)
- Een RAID is een verzameling van disks die samen een logisch volume vormen
- Het is een manier om dezelfde data te bewaren op verschillende disks tegelijk
= bescherming tegen problemen met een disk zoals bad sector's of crash

File Systems

Waarom meerdere?

Intermezzo.. Wat is RAID ?

- Verschillende configuratie zijn mogelijk:
RAID0 ... RAID6
- De configuraties verschillen in hoe de data over verschillende disk verdeeld wordt.

Bvb: RAID0 = verdeelt de data over de disks heen
zonder extra kopieën of informatie

RAID1 = exacte kopie op alle disks

...



File Systems

Waarom meerdere?

XFS:

- Silicon Graphics – (SGI)
- Blinkt uit in het uitvoeren van parallele input/output (I/O) operations
- Ontworpen voor 3D graphics
- Nadelen:
 - XFS FS kan enkel van formaat veranderen door backup, reinstall + restore (zie partities)
 - geen checksum protectie

File Systems

Waarom meerdere?

Zo heeft elk File system
z'n voor & nadelen..

→ Momenteel **EXT4** meest gebruikte file system op linux



File Systems

Hoe meerdere FS gebruiken op één PC ?

- Opslag-media opdelen in Partities
- Elke partitie kan een ander FS bevatten
- Partition-table → MBR – master-boot-record
- Terminal commando **fdisk**
- In de GUI is dit **gparted**



File Systems

Hoe meerdere FS gebruiken op één PC ?

- **fdisk -l** → geeft lijst van partities

Disk /dev/sda: 250.0 GB, 250059350016 bytes 255 heads,
63 sectors/track, **30401 cylinders**

Units = cylinders of 16065 * 512 = 8225280 bytes

Device	Boot	Start	End	Blocks	Id	System
/dev/sda1	*	<u>1</u>	191	1534176	83	Linux
/dev/sda2		192	2231	16386300	83	Linux
/dev/sda3		2232	3506	10241437+	83	Linux
/dev/sda4		3507	<u>30401</u>	216034087+	5	Extended
/dev/sda5		3507	3767	2096451	82	Linux swap / Solaris



File Systems

Hoe meerdere FS gebruiken op één PC ?

- **fdisk -l** → geeft lijst van partities

Disk /dev/sda: 250.0 GB, 250059350016 bytes 255 heads,
63 sectors/track, 30401 cylinders

Units = cylinders of 16065 * 512 = 8225280 bytes

Device	Boot	Start	End	Blocks	Id	System
--------	------	-------	-----	--------	----	--------

/dev/sda1	*	1	191	1534176	83	Linux
------------------	----------	----------	------------	----------------	-----------	--------------

/dev/sda2		192	2231	16386300	83	Linux
-----------	--	-----	------	----------	----	-------

/dev/sda3		2232	3506	10241437+	83	Linux
-----------	--	------	------	-----------	----	-------

/dev/sda4		3507	30401	216034087+	5	Extended
-----------	--	------	-------	------------	---	----------

/dev/sda5		3507	3767	2096451	82	Linux swap / Solaris
-----------	--	------	------	---------	----	----------------------



File Systems

Hoe meerdere FS gebruiken op één PC ?

- **fdisk -l** → geeft lijst van partities

Disk /dev/sda: 250.0 GB, 250059350016 bytes 255 heads,
63 sectors/track, 30401 cylinders

Units = cylinders of 16065 * 512 = 8225280 bytes

Device	Boot	Start	End	Blocks	Id	System
--------	------	-------	-----	--------	----	--------

/dev/sda1	*	1	191	1534176	83	Linux
-----------	---	---	-----	---------	----	-------

/dev/sda2		192	2231	16386300	83	Linux
-----------	--	-----	------	----------	----	-------

/dev/sda3		2232	3506	10241437+	83	Linux
-----------	--	------	------	-----------	----	-------

/dev/sda4		3507	30401	216034087+	5	Extended
-----------	--	------	-------	------------	---	----------

/dev/sda5		3507	3767	2096451	82	Linux swap / Solaris
------------------	--	-------------	-------------	----------------	-----------	-----------------------------



User omgeving → File System

→ *Tot nu toe hadden we het over de kant naar disk toe*

In de eerste labo's maakten we reeds kennis
met de gebruikers kant v/h FS

De mapjes in de grafische omgeving
gelijken op die van MS Windows

Het linux FS is echter 'iets' geavanceerder...



User omgeving → File System

MS Windows

- In windows heeft elke disk zijn letter
- Je windows systeem staat bijvoorbeeld op C:\
- DVD-drive is vaak D:\
- Als je op een netwerk zit dan zie je wel vaker G:\, H:\ of X:\ of nog andere letters.
- Partities hebben ook opnieuw een eigen letter

User omgeving → File System

MS Windows

- En als je C:\ vol staat dan is het gedaan met spelen...
- Windows zal een error geven en geen programma's meer kunnen installeren

“please remove unused program's to create extra disk space”



User omgeving → File System

Linux

- Linux kent geen drive letters.
- Bij linux worden disk's of andere media ge-**mount**.
→ commando's '**mount**' en '**umount**'
- Kan manueel of automatisch
- Linux mount automatisch in de directory
/media/jou-username
- Mounten kan om het even waar in de boomstructuur

User omgeving → File System

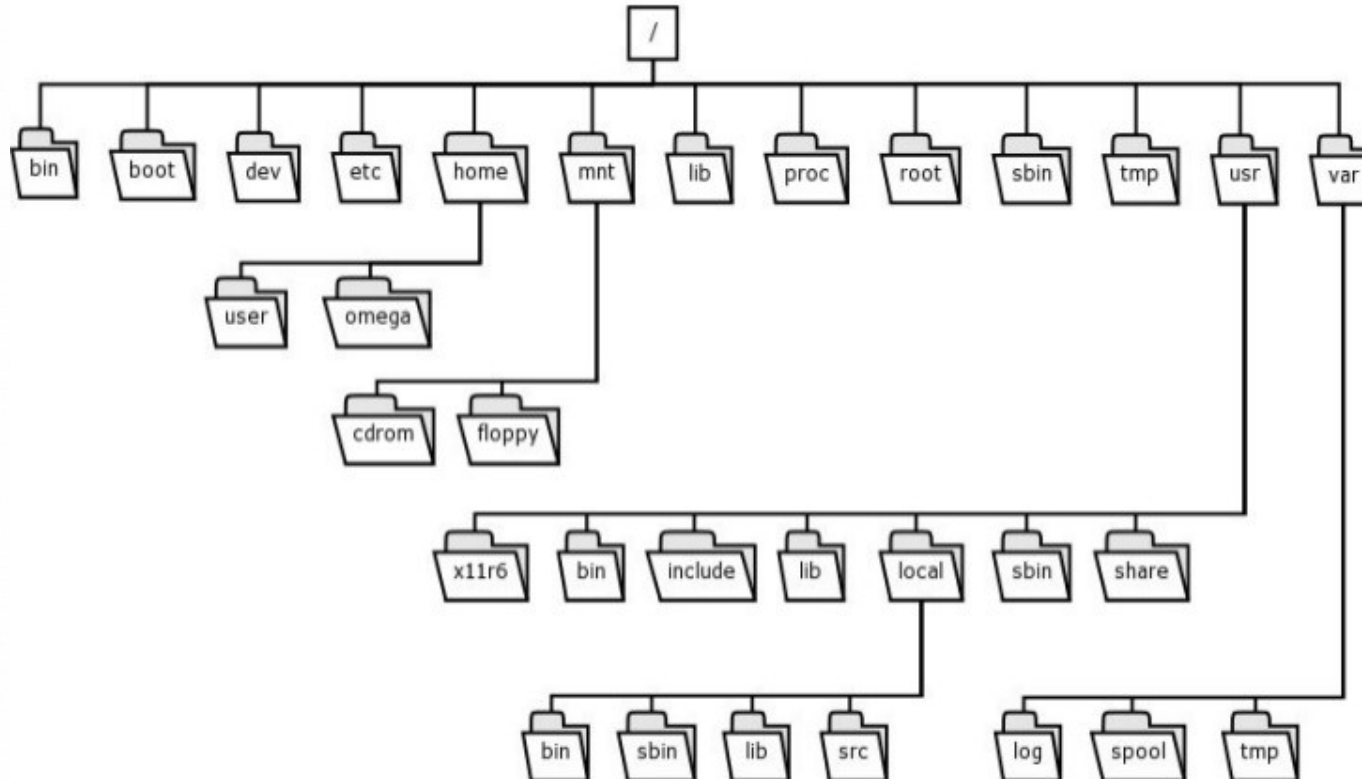
Linux

- Dit wil zeggen dat:
andere directory = mogelijks andere disk...
- Dit kan zelfs een disk zijn op een andere locatie!
- De plaats waar het medium ge-mount wordt =
mount-point
- éénzelfde medium kan op meerdere plaatsen
in de boomstructuur gemount worden

User omgeving → File System

Linux Directory Structure

*boom
structuur*



User omgeving → File System

voorbeeld:

/ → root is hd1

/usr → /usr stond eerst ook op hd1
maar hd1 was vol en /usr staat nu op hd2...

/mnt/ny → is een data-storage in New-York..

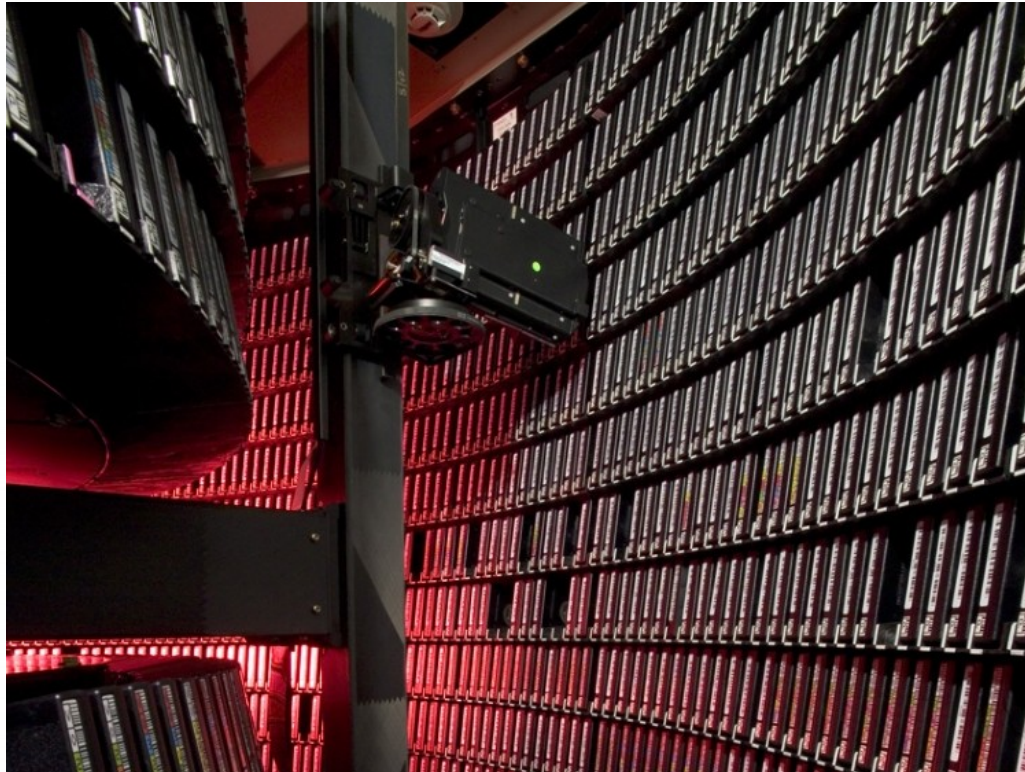
User omgeving → File System

Oorsprong commando **'mount'** & **'umount'**



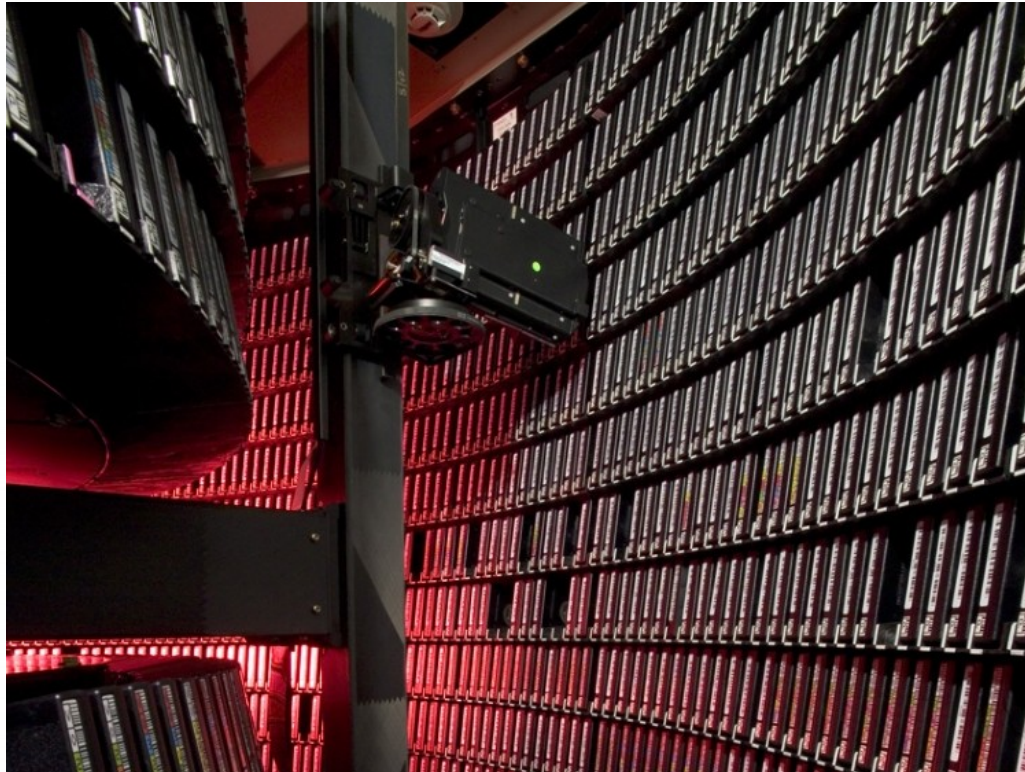
User omgeving → File System

moderne tape library backup



User omgeving → File System

480 TerraBytes per tape ...



User omgeving → File System

Directory systeem

- Komt voort uit nood aan orde & structuur
- Rechten systeem:
 - welke gebruiker heeft toegang ?
Linux = multi-user omgeving
3 niveau's → all, group, user
- Groepering van bestanden:
 - mapjes / directories



User omgeving → File System

Directory systeem

- Linux volgt hiervoor de 'FHS' norm
- FHS = “Efilesystem Hierarchy Standard”
- FHS is een POSIX norm:

POSIX = “Portable Operating-System Interface”
= Set van normen voor Unix-achtige OS



FHS

/ = root

- /bin : basic command binaries
- /boot : files and commands required for boot
- /dev : device files
- /sys : system files
- /etc : basic configuration files
- /home : home directory of all users except root
- /lib : program libraries (+ /lib64 voor 64 bit pc's)
- /media : mount point for removable media (automount)
- /mnt : mount point for temporary file system
- /opt : applications
- /proc : kernel processes and resource allocations
- /root : root user directory
- /sbin : system administration tools
- /tmp : temporary storage
- /usr : programs and data for users
- /var : variable data like log files, print spools



FHS

/dev = device files

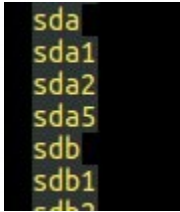
We nemen even een kijkje in deze directory:

```
autofs      fuse      loop3      parport0    sdd         tty1        tty29      tty48      ttyprintk   ttyS27      vcs
block       hidraw0    loop4      port         sde         tty10      tty3       tty49      ttyS0        ttyS28      vcs1
bsg         hidraw1    loop5      ppp          sdf         tty11      tty30      tty5       ttyS1        ttyS29      vcs2
btrfs-control hidraw2    loop6      psaux        sdg         tty12      tty31      tty50      ttyS10       ttyS3       vcs3
bus         hpet       loop7      ptmx         sg0         tty13      tty32      tty51      ttyS11       ttyS30      vcs4
cdrom       hugepages  loop-control pts          sg1         tty14      tty33      tty52      ttyS12       ttyS31      vcs5
cdrw        hwrng      lp0        random       sg2         tty15      tty34      tty53      ttyS13       ttyS4       vcs6
char        i2c-0      mapper     rfc2515     sg3         tty16      tty35      tty54      ttyS14       ttyS5       vcsa
console     i2c-1      mcelog     rtc          sg4         tty17      tty36      tty55      ttyS15       ttyS6       vcsa1
core        i2c-2      mem        rtc0         sg5         tty18      tty37      tty56      ttyS16       ttyS7       vcsa2
cpu         i2c-3      memory_bandwidth sda          sg6         tty19      tty38      tty57      ttyS17       ttyS8       vcsa3
cpu_dma_latency i2c-4      mqueue     sda1         sg7         tty2        tty39      tty58      ttyS18       ttyS9       vcsa4
cuse        initctl    net         sda2         shm          tty20      tty4       tty59      ttyS19       uhid        vcsa5
disk        input     network_latency sda5         snapshot    tty21      tty40      tty6       ttyS2        uinput     vcsa6
dri         kmsg      network_throughput sdb          snd          tty22      tty41      tty60      ttyS20       urandom     vfio
dvd         kvm        null        sdb1         sr0          tty23      tty42      tty61      ttyS21       usb         vga_arbiter
dvdrw       lightnvm   nvidia0     sdb2         stderr       tty24      tty43      tty62      ttyS22       userio      vhost-net
ecryptfs    log        nvidiaactl  sdc          stdin        tty25      tty44      tty63      ttyS23       vboxdrv     zero
fb0         loop0      nvidia-modeset sdc1         stdout       tty26      tty45      tty7       ttyS24       vboxdrv     zero
fd          loop1      nvidia-uvm  sdc2         tty          tty27      tty46      tty8       ttyS25       vboxnetctl
full        loop2      nvram       sdc5         tty0         tty28      tty47      tty9       ttyS26       vboxusb
```


FHS

/dev = device files

We nemen even een kijkje in deze directory:

A terminal window showing a list of device files in the /dev directory. The text is yellow on a black background. The visible entries are: sda, sda1, sda2, sda5, sdb, sdb1, and sdb2.

```
sda  
sda1  
sda2  
sda5  
sdb  
sdb1  
sdb2
```

- Hier vinden we bvb “sda” → dit is harddisk “a” !
- Hieronder eventueel “sda1”, “sda2”, ..
Dit zijn de partities op deze disk

*Dit wil zeggen dat deze harddisk ook zélf
wordt gepresenteerd door een bestandsnaam!*

FHS

/dev = device files

Kijken we verder dan zien we tty1, tty2, ...



```
tty1  tty29  t
tty10 tty3    t
tty11 tty30   t
tty12 tty31   t
tty13 tty32   t
tty14 tty33   t
tty15 tty34   t
tty16 tty35   t
tty17 tty36   t
tty18 tty37   t
tty19 tty38   t
```

- 'tty' komt van 'teletype' → 'terminal'

*De terminal vensters zelf worden dus opnieuw
gerepresenteerd door een bestandsnaam*

FHS

Cray mainframe & terminal



FHS

/dev = device files

We zien ook de directory 'cpu'!

- Als we 'cd' doen naar deze cpu directory zien we opnieuw enkele directories: 0,1,2, etc.

```
/dev/cpu$ ll
root root    60 0kt  2 11:37 0
root root    60 0kt  2 11:37 1
root root    60 0kt  2 11:37 2
root root    60 0kt  2 11:37 3
root root 108 0kt  2 11:37 mi
```

Elk van deze directories = core van de cpu...

FHS

/dev = device files

In /dev staat ook nog het bestand 'mem'

- 'mem' ... van memory



Dus zelfs de cpu of het geheugen krijgt een bestands- of directory representatie

“ In Linux everything is a file “



“ In Linux everything is a file “

of toch ongeveer...

- Men kan niet schrijven naar een keyboard
- Niet lezen van een scherm
- Geen files als dusdanig maar wel ‘file achtig’

Doel = uniformisering

- File achtig → *Not files but treated as files*
- Data van/naar devices noemt men ‘streams’
- Mogelijkheid tot stream redirection (zie labo’s)



User omgeving → File System ← Opslag medium

inodes

- Ruimere definitie van file system is het geheel v/d twee vorige delen samen: *user + hardware kant*
- Dit wordt duidelijk door het gebruik van 'inodes' om alle informatie bij te houden
- **inodes** verbinden alles met elkaar



User omgeving → File System ← Opslag medium

inodes

Een inode-object bevat volgende informatie:

- waar de block's op het opslag medium staan
- In welke directory de file komt
- De tijd en datum van de laatste modificatie
- Wie de eigenaar is van het bestand
- Tot welke groep
- Wie er welke toegangsrechten heeft



User omgeving → File System ← Opslag medium

inodes

inodes bevatten niet de bestandsnaam !

- Bestandsnamen refereren naar een inode
- Elke inode heeft namelijk een uniek nummer
- Dit laat *meerdere bestandsnamen* toe voor hetzelfde achterliggende bestand

User omgeving → File System ← Opslag medium

inodes

Commando **stat [filename]**

```
File: 'Cylinder.png'
Size: 57784          Blocks: 128          IO Block: 4096   regular file
Device: 821h/2081d   Inode: 2884830    Links: 1
Access: (0664/-rw-rw-r--)  Uid: (  0/   root)   Gid: (  0/   root)
Access: 2017-10-02 19:20:15.389853713 +0200
Modify: 2017-10-02 19:19:36.509789262 +0200
Change: 2017-10-02 19:43:35.192589111 +0200
Birth: -
```



User omgeving → File System ← Opslag medium

inodes

Merk op:

Uid → user ID

Gid → group ID



User omgeving → File System ← Opslag medium

inodes

ls -i

'ls' met de optie -i
toont naast elk bestand
het inode nummer

