

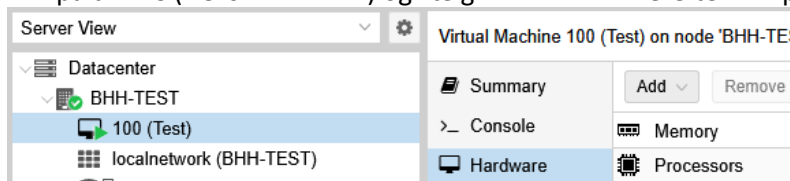
Installation af ekstra HDD på Ubuntu 24.04.

Hvis du ikke sidder på en Proxmox VM, kan næste afsnit springes over, og gå til: Installation af ekstra HDD under Ubuntu. Det kan være forskellige måder at tilføje en ekstra harddisk alt efter hvilken hypervisor du bruger. Se i dokumentationen for din hypervisor hvordan du tilføjer en ekstra harddisk.

Tilføj Ekstra Harddisk til VM i Proxmox

Start med at logge ind på din Proxmox. (Virker også under Cluster opsætning) Vær også sikker på at der er plads på den FYSISKE harddisk for at kunne tilføje en virtuel harddisk. Hvis det er en Virtuel behøver man ikke slukke PC for at tilføjer harddisk.

Klik på din PC (F.eks. 100 (Test)) og vælg Hardware Derefter klik på Add



Herefter vælg Hard Disk

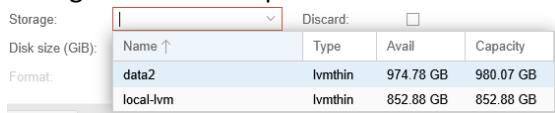


Jeg vil lige gennemgå de forskellige sektioner, inden jeg viser der samlede billede:

Bus/Device: Det er hvilken "controler" harddisk skal sidde på hvilken type. (Her VirtIO Block, og busdevice 1)

Bus/Device: VirtIO Block 1

Storage: Hvor skal du placere din viretuelle harddisk. (Her på data2)



Disk size (GiB): Størrelsen på harddisk i Giga Byte (Her 50 Gbyte)

Disk size (GiB): 50

Resten forbliver standard, når alt er udfyldt tryk på Add

Som man kan se er der tilføjet en ekstra harddisk (Ved pilen)

Add	Remove	Edit	Disk Action	Revert
Memory	8.00 GiB			
Processors	4 (2 sockets, 2 cores) [x86-64-v2-AES]			
BIOS	Default (SeaBIOS)			
Display	Default			
Machine	Default (i440fx)			
SCSI Controller	VirtIO SCSI single			
CD/DVD Drive (ide2)	none,media=cdrom			
Hard Disk (virtio0)	data2:vm-100-disk-0,iothread=1,size=50G			
Hard Disk (virtio1)	data2:vm-100-disk-1,iothread=1,size=50G			
Network Device (net0)	virtio=BC:24:11:2E:95:4D,bridge=vbr0,firewall=1,tag=500			

Slut på Tilføj Ekstra Harddisk til VM i Proxmox

Installation af ekstra HDD under Ubuntu.

Sørg for at harddisk er klar, tilføjet til din Ubuntu Server, og log på din Ubuntu med SSH.

Det første vi skal er at se om vi overhovedet kan se vores nye harddisk. Som man kan se er der kommet en harddisk til ved pilen. Vi bruger denne kommando:

```
lsblk
```

Resultat:

```
dtmek@test:~$ lsblk
NAME        MAJ:MIN RM  SIZE RO TYPE MOUNTPOINTS
sr0         11:0    1 1024M  0 rom
vda         253:0    0   50G  0 disk
├─vda1      253:1    0    1M  0 part
└─vda2      253:2    0   50G  0 part /
vdb         253:16   0   50G  0 disk
```

Som vi kan se er harddisken ikke Partitioneret, eller formateret. Det er god ide at gøre, ellers kan vi jo ikke bruge den. Så det gør vi nu med kommandoen "sudo fdisk /dev/(det der står under NAME ved lsblk)". Ved mig er det:

```
sudo fdisk /dev/vdb
```

Resultat:

```
dtmek@test:~$ sudo fdisk /dev/vdb
[sudo] password for dtmek:

Welcome to fdisk (util-linux 2.39.3).
Changes will remain in memory only, until you decide to write them.
Be careful before using the write command.

Device does not contain a recognized partition table.
Created a new DOS (MBR) disklabel with disk identifier 0xbbl60eb3.

Command (m for help):
```

Vi trykker "m" for at se hvilke kommandoer vi har til rådighed:

```
Command (m for help): m
Help:
DOS (MBR)
a toggle a bootable flag
b edit nested BSD disklabel
c toggle the dos compatibility flag

Generic
d delete a partition
f list free unpartitioned space
l list known partition types
n add a new partition
p print the partition table
t change a partition type
v verify the partition table
i print information about a partition

Misc
m print this menu
u change display/entry units
x extra functionality (experts only)

Script
l load disk layout from sfdisk script file
O dump disk layout to sfdisk script file

Save & Exit
w write table to disk and exit
q quit without saving changes

Create a new label
g create a new empty GPT partition table
G create a new empty SGI (IRIX) partition table
o create a new empty MBR (DOS) partition table
n create a new empty Sun partition table
```

Ud fra listen over kommandoer, kan vi se at vi skal starte med at trykke g (create a new empty GPT partition table):

```
Command (m for help): g
Created a new GPT disklabel (GUID: 8117F0DA-F90E-4567-A48D-8C64AE090909).

Command (m for help):
```

Derefter skal vi trykke n (add a new partition). Brug default værdier til alt input.

```
Command (m for help): n
Partition number (1-128, default 1):
First sector (2048-104857566, default 2048):
Last sector, +/-sectors or +/-size(K,M,G,T,P) (2048-104857566, default 104855551):

Created a new partition 1 of type 'Linux filesystem' and of size 50 GiB.

Command (m for help):
```

Så prøver vi at trykke p (print the partition table) Her kan vi se en partition /dev/vdb1. Det er vores data partition der nu er oprettet.

```
Command (m for help): p
Disk /dev/vdb: 50 GiB, 53687091200 bytes, 104857600 sectors
Units: sectors of 1 * 512 = 512 bytes
Sector size (logical/physical): 512 bytes / 512 bytes
I/O size (minimum/optimal): 512 bytes / 512 bytes
Disklabel type: gpt
Disk identifier: 8117F0DA-F90E-4567-A48D-8C64AE090909

Device      Start      End      Sectors  Size Type
/dev/vdb1   2048 104855551 104853504 50G Linux filesystem

Command (m for help):
```

Vi afslutter vores partitionering af harddisk ved at trykke w (write table to disk and exit)

```
Command (m for help): w
The partition table has been altered.
Calling ioctl() to re-read partition table.
Syncing disks.


dtmek@test:~$
```

Vi kan prøve at skrive lsblk igen. Læg mærke til at vores nyoprettede partition er synlig ved pilen.

lsblk

Resultat:

```
dtmek@test:~$ lsblk
NAME        MAJ:MIN RM  SIZE RO TYPE MOUNTPOINTS
sr0         11:0    1 1024M  0  rom
vda         253:0    0   50G  0  disk
├─vda1      253:1    0    1M  0  part
├─vda2      253:2    0   50G  0  part /
└─vdb        253:16   0   50G  0  disk
   └─vdb1    253:17   0   50G  0  part
```



Nu har vi oprettet partitionen. Men, og ja, der er også et men, harddisken kan ikke bruges endnu. Harddisken er jo ikke formateret. Det må vi hellere gøre med en kommando der hedder: "mkfs.ext4 -E lazy_itable_init=0,lazy_journal_init=0 /dev/partition". Den kommando formaterer en harddisk i formatet Ext4 der er standard for Linux/Ubuntu. For mig vil kommandoen være (læg mærke til det er partitionen, ikke harddisken vi formaterer) Hvis der ikke bruges de to flag der hedder noget med "lazy" så formateres harddisken efterfølgende stille og rolig medens CPU ikke har noget at lave i baggrunden. Jeg anbefaler helt klart følgende kommando, der formaterer HELE harddisken med det samme. (Det kan tage LANG tid med store meget Harddiske f.eks. flere Terra byte):

```
sudo mkfs.ext4 -E lazy_itable_init=0,lazy_journal_init=0 /dev/vdb1
```

Resultat:

```
dtmek@test:~$ sudo mkfs.ext4 -E lazy_itable_init=0,lazy_journal_init=0 /dev/vdb1
[sudo] password for dtmek:
mke2fs 1.47.0 (5-Feb-2023)
Discarding device blocks: done
Creating filesystem with 13106688 4k blocks and 3276800 inodes
Filesystem UUID: ec8e5eed-ebd2-4b53-a112-dc40db14c6d0
Superblock backups stored on blocks:
    32768, 98304, 163840, 229376, 294912, 819200, 884736, 1605632, 2654208,
    4096000, 7962624, 11239424

Allocating group tables: done
Writing inode tables: done
Creating journal (65536 blocks): done
Writing superblocks and filesystem accounting information: done

dtmek@test:~$
```

Nu har vi formateret harddisken, og det har tildelt vores partition et unikt ID. Dette skal vi lige finde for at kunne tilføje harddisken til vores system. Det finder vi med følgende kommando:

```
sudo blkid
```

Resultat:

```
dtmek@test:~$ sudo blkid
/dev/vda2: UUID="c6a53cal-6935-4871-962e-cdbbeb99f4e7" BLOCK_SIZE="4096" TYPE="ext4" PARTUUID="c4907993-8b1f-4220-b7a0-07490c666f94"
/dev/vdb1: UUID="ec8e5eed-ebd2-4b53-a112-dc40db14c6d0" BLOCK_SIZE="4096" TYPE="ext4" PARTUUID="e06e5ecc-11e9-46da-9577-3ba5e9c8fcab"
/dev/vda1: PARTUUID="e91c262e-501e-4285-960c-9e113f4b05e0"
dtmek@test:~$
```

For at kunne tilføje en harddisk skulle vi jo have dens unikke ID (kaldet UUID). Her kan vi se at vores harddisk har UUID nummer: UUID="ec8e5eed-ebd2-4b53-a112-dc40db14c6d0". Det nummer vil selvfølgelig være forskellig for dig. Det UUID nummer skal du bruge lige om lidt.

Nu er vi klar til at sætte den ekstra harddisk ind i vores filsystem. Så det går vi bare i gang med... HOV... Vent lige lidt! Vores harddisk skal jo tilføjes vores /sambashare bibliotek. Men hvad så med de data der evt. ligger der? Lad os se om der ligger noget. Det gør vi ved at skrive:

```
ls -l /sambashare
```

Resultat:

```
dtmek@test:~$ ls -l /sambashare
total 12
-rw----- 1 elev elev 7452 Mar 20 08:44  HelloWindows.asm
drwx----- 2 elev elev 4096 Mar 20 09:55  'Ny mappe'
dtmek@test:~$
```

Det var de vist meget godt at jeg lige kontrollerede det. Der ligger jo noget i biblioteket ved mig! Hvordan kommer vi videre? Der findes et par løsninger. Jeg vil nævne 2:

1. Tage backup af data. Tilslutte den ekstra harddisk. Også restore data efter vi har tilsluttet harddisk.
2. Mounete den nye harddisk et andet sted. Flytte data til denne harddisk. Derefter flytte harddisk til det rigtig mount punkt.

Der er selvfølgelig både fordele og ulemper på hver af måderne. Jeg vil kort forklare lidt om de 2 muligheder. Begge muligheder vil medføre downtime på server.

1. Denne mulighed kan tage lang tid. Man skal jo flytte dataene 2 gange. Desuden vil den storage plads som dataene optager på den første harddisk ikke blive frigivet, medmindre, man sletter data inde man mouneter sin nye harddisk.
2. Dette er den udgave jeg vil gennemgå, da det tager kortest nede tid. (data flyttes kun en gang). Denne medfører også at pladsen som data optog på den gamle harddiske bliver frigivet med det samme.

Først skal vi mounete (forbinde) den nye harddisk til et bibliotek i vores bibliotekststruktur, så vi kan flytte data over på den. Til det skal vi redigere i edn fil der sørger for at tilslutte harddiske under boot af Ubuntu. Fileen hedder fstab, og det gør vi med kommandoen:

```
sudo nano /etc/fstab
```

Resusltat:

```
GNU nano 7.2
/etc/fstab: static file system information.
#
# Use 'blkid' to print the universally unique identifier for a
# device; this may be used with UUID= as a more robust way to name devices
# that works even if disks are added and removed. See fstab(5).
#
# <file system> <mount point> <type> <options> <dump> <pass>
# / was on /dev/vda2 during curtin installation
/dev/disk/by-uuid/c6a53cal-6935-4871-962e-cdbbeb99f4e7 / ext4 defaults 0 1
/swap.img none swap sw 0 0
```

Tilføj en linje nederst der hedder `UUID=(Harddisks UUID) /Sti/Til/Mountpunkt ext4 errors=remount-ro 0`. Jeg placerer min nye harddisk under det standard mountpunkt der er i Ubuntu, og det hedder /mnt. For at gøre dette, tilføj følgende linjer til sidst i filen. (Man kan tilføje kommentar bag ved et # tegn. Her tilføjer jeg kommentaren: #Mount Ekstra Harddisk)

```
#Mount Ekstra Harddisk
```

```
UUID=ec8e5eed-ebd2-4b53-a112-dc40db14c6d0 /mnt ext4 errors=remount-ro 0
```

Resultat efter tilføjelse:

```
GNU nano 7.2
# /etc/fstab: static file system information.
#
# Use 'blkid' to print the universally unique identifier for a
# device; this may be used with UUID= as a more robust way to name devices
# that works even if disks are added and removed. See fstab(5).
#
# <file system> <mount point> <type> <options> <dump> <pass>
# / was on /dev/vda2 during curtin installation
/dev/disk/by-uuid/c6a53cal-6935-4871-962e-cdbbeb99f4e7 / ext4 defaults 0 1
/swap.img none swap sw 0 0

#Mount Ekstra Harddisk
UUID=ec8e5eed-ebd2-4b53-all2-dc40db14c6d0 /mnt ext4 errors=remount-ro 0
```

For at gemme ændringer trykkes der CTRL+S og så CTRL+X, og man er tilbage i prompten.

Nu skal vi bare have vores server til at tilføje harddisken så vi har adgang til den. Vi får vores server til at genmounte alle vores harddiske ved at udføre følgende kommando:

```
sudo mount -a
```

Resultat (Den kan komme med en besked der siger noget med "(Hint):") Man kan godt køre denne kommando den foreslår, men det er normalt ikke nødvendigt. Hvis man er i tvivl, kørs kommandoen igen, så får man resultatet:

```
dtmek@test:~$ sudo mount -a
mount: (hint) your fstab has been modified, but systemd still uses
the old version; use 'systemctl daemon-reload' to reload.
dtmek@test:~$ sudo mount -a
dtmek@test:~$
```

For at se om vi reelt har fået tilsluttet harddisken, kan man udføre følgende kommando. Hvis man ser et underbibliotek der hedder "lost+found" så er der mountet en "fysisk" harddisk på den plads.

```
ls -l /mnt
```

Resultat:

```
dtmek@test:~$ ls -l /mnt
total 16
drwx----- 2 root root 16384 Mar 20 09:41 lost+found
dtmek@test:~$
```

OK. Nu har vi fået tilføjet vores harddisk. Så nu skal vi have flyttet vores data. Men... hvad nu hvis der er en eller anden der har en fil åben allerede, medens vi flytter? Så kan det jo gå galt? Man skal selvfølgelig give besked om at systemet ikke kan bruges mellem kl. 17:00 og 18:00 eller hvornår man vil flytte data. Men hvordan sikrer vi os at der ikke er nogen der kan gå på vores server medens vi flytter data?

Det er jo nemt. Vi stopper vores Samba og FTP service. Hvis de services ikke kører, kan man ikke tilgå vores server... Smart! Det gør vi med følgende kommandoer:

```
sudo service smbd stop
sudo service vsftpd stop
```

Resultat:

```
dtmek@test:~$ sudo service smbd stop
dtmek@test:~$ sudo service vsftpd stop
dtmek@test:~$
```

Så er servicerne stoppet, og vi kan flytte vores data. Det gør jeg med følgende kommando (sudo er nødvendig, da vi ikke er logget ind som elev) Bag efter ser vi indholdet af de to biblioteker, og kan se at indholdet er flyttet):

```
sudo mv /sambashare/* /mnt
ls -l /sambashare
ls -l /mnt
```

Resultat

```
dtmek@test:~$ sudo mv /sambashare/* /mnt
dtmek@test:~$ ls -l /sambashare/
total 0
dtmek@test:~$ ls -l /mnt
total 28
-rw----- 1 elev elev 7452 Mar 20 08:44 HelloWorlds.asm
drwx----- 2 root root 16384 Mar 20 09:41 lost+found
drwx----- 2 elev elev 4096 Mar 20 09:55 'Ny mappe'
dtmek@test:~$
```

Nu har vi flyttet data. Så skal vi have vores filer tilbage til /sambashare igen. Vi starter med at redigere vores /etc/fstab fil igen. Og på linjen med vores ekstra hardiskmount ændres /mnt til /sambashare

```
sudo nano /etc/fstab
```

Resultat:

```
#Mount Ekstra Harddisk
UUID=ec8e5eed-ebd2-4b53-all2-dc40db14c6d0 /sambashare ext4 errors=remount-ro 0
```

For at gemme ændringer trykkes der CTRL+S og så CTRL+X, og man er tilbage i prompten.

Nu skal vi igen have mountet vores nye opsætning, det gør vi igen med "sudo mount -a", og så ser jeg straks at der er data tilbage der hvor vi forventer det er, med "ls -l /sambashare":

```
sudo mount -a
ls -l /sambashare
```

Resultat (Der kan komme med en besked der siger noget med "(Hint):" Man kan godt køre denne kommando den foreslår, men det er normalt ikke nødvendigt. Hvis man er i tvivl, kør kommandoen igen, så får man resultatet:

```
dtmek@test:~$ sudo mount -a
mount: (hint) your fstab has been modified, but systemd still uses
the old version; use 'systemctl daemon-reload' to reload.
dtmek@test:~$ sudo mount -a
dtmek@test:~$ ls -l /sambashare
total 28
-rw----- 1 elev elev 7452 Mar 20 08:44 HelloWorlds.asm
drwx----- 2 root root 16384 Mar 20 09:41 lost+found
drwx----- 2 elev elev 4096 Mar 20 09:55 'Ny mappe'
dtmek@test:~$
```

Vi skal ikke glemme at starte de services vi stoppede inden vi gik i gang med at flytte data. Ellers kan folk jo ikke tilgå server igen. Det gør vi sådan her:

```
sudo service smbd start  
sudo service vsftpd start
```

Resultat:

```
dtmek@test:~$ sudo service smbd start  
dtmek@test:~$ sudo service vsftpd start  
dtmek@test:~$
```

For at være sikker på at elev har adgang til alt under sambashare kører vi lige en kommando, der giver fuld adgang til elev, da dette kan have ændret sig da vi mountede en harddisk under "/sambashare".

```
sudo chown elev:elev -R /sambashare
```

Resultat:

```
dtmek@test:~$ sudo chown elev:elev -R /sambashare  
dtmek@test:~$
```

Den ekstra harddisk er nu tilføjet.