

Linux 2

DEVOPS 2021

Lektion 2

Uppvärmning

- Kommentarer / önskemål efter första dagen?
- Skicka förslag på grupper till grupparbetet denna veckan

Idag

- Allmänt om server, uppbyggnad
- Miljöer för utveckling, test och drift
- Repetition: Filer och filrättigheter
- Filöverföringar – om ftp, scp etc
- NFS, olika typer av filsystem

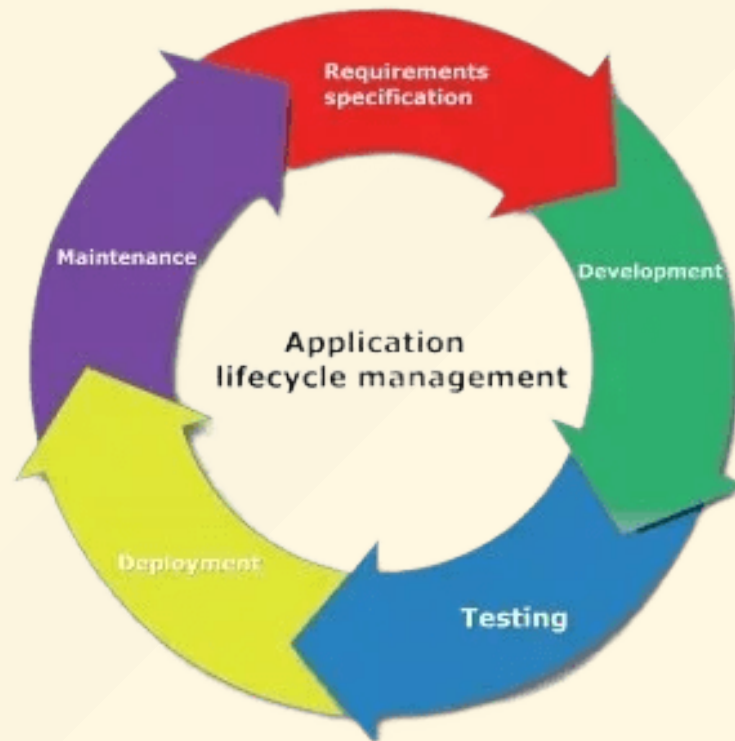
Miljöer och uppbyggnad

Hur passar en server in

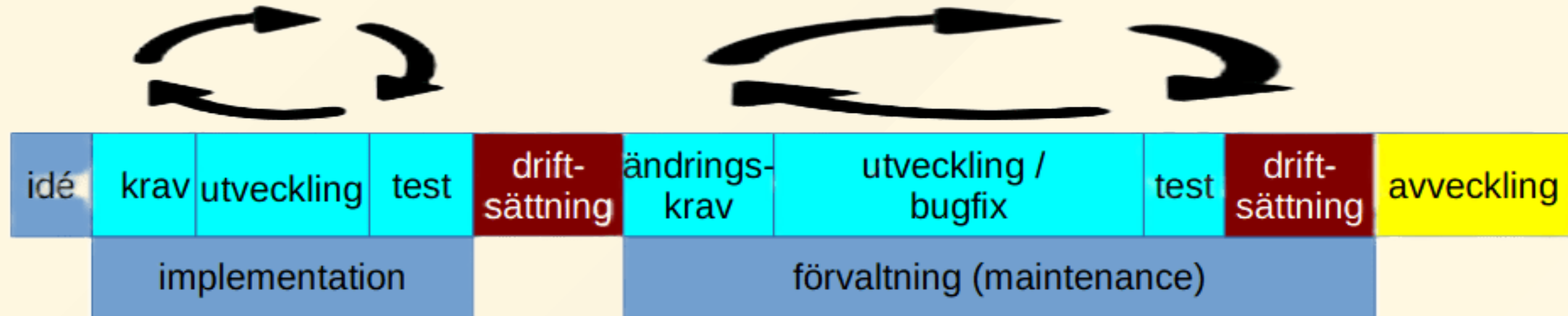
Linux-server

- Behöver först och främst veta vad just denna server är till för
- Använder sällan annat än kommandorad
- Fysiska och virtuella servrar

Applikationens livscykel

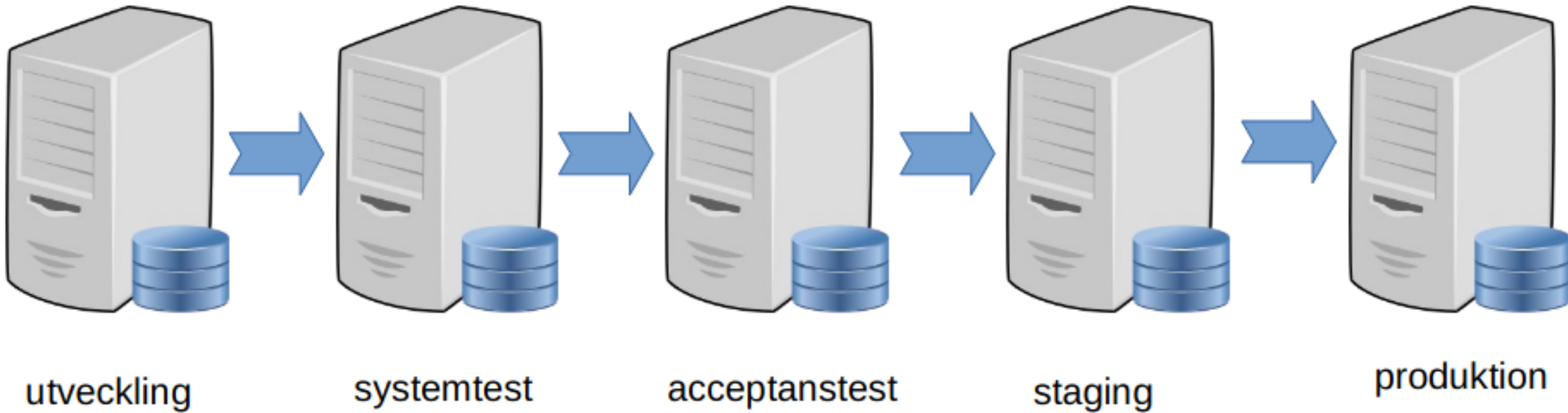


Tidslinje för ett system

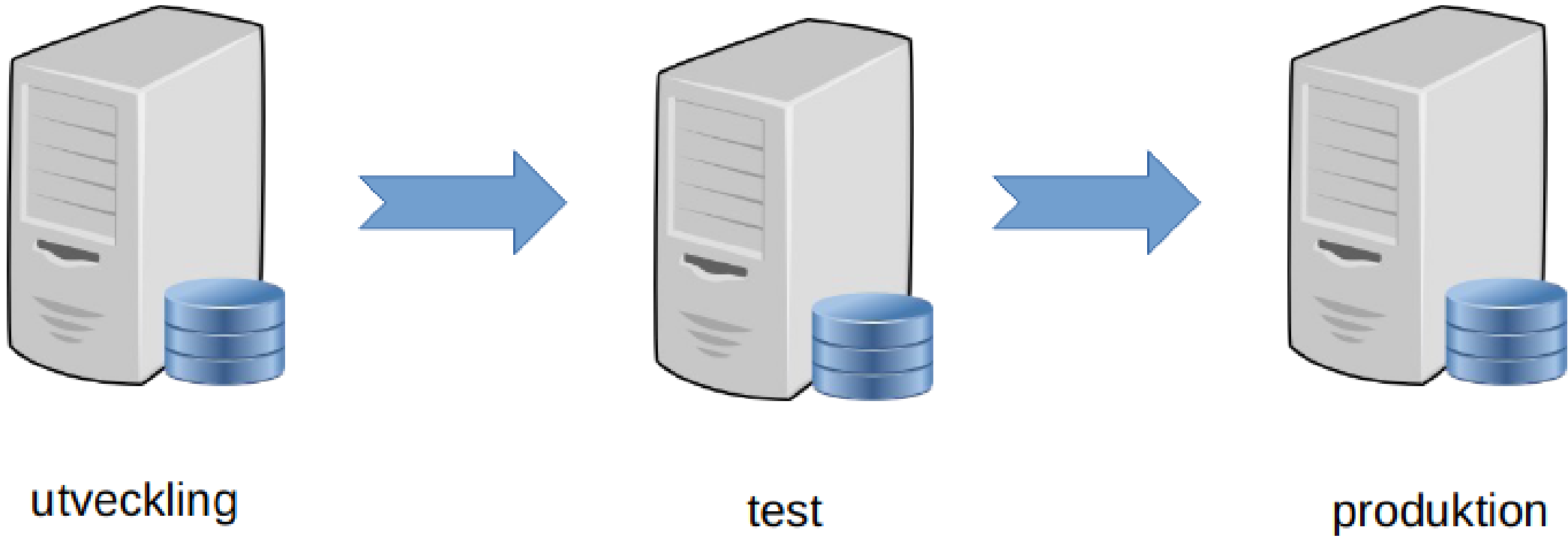


Milijööri

Traditionell uppsättning miljöer



Mer minimalistisk (och vanligare)



Dev-miljö ("utveckling")

- Har utvecklingsverktyg (debugger, profilering, etc)
- Koppling till versionshantering (commit -> deploy)
- Test-data i databas
- Oftast på privat internt nätverk

Test-miljö

- Har testverktyg (profilering, automat-testning, UI-tester, etc)
- Har ofta även utvecklingsverktyg
- Också test-data
- Oftat också på privat internt nätverk

Prod-miljö ("drift", "produktion")

- Bara drift-verktyg, dvs inga utvecklings- eller testverktyg
- Äkta data ("prod-databas")
- Regelbunden backup
- Redundans: flera servrar för ökad driftssäkerhet
- Tillgång endast för administratörer (eller helst ingen access alls!)

Regler och konfiguration

Allmänna regler

- Ha inte saker installerade som inte behövs på Servrar – ha mindsetet “less is more”
- Koll på användare / vem som har tillgång till miljöerna
- Resursplanering utifrån serverns funktion

Att ha koll på

- Vilka program / paket behövs för det som skall finnas på servern?
- Vilka portar behöver vara öppna?
- Var i nätverket skall servern vara?
- Vilka skall ha tillgång till den?
- Vad för övervakning behövs?
- Vilka backuper behövs?

Övning 1

Antag att ditt team utvecklar en Java- applikation med webbinterface, användardata i en MySQL-databas, samt ett antal informationsfiler som kan laddas ned från webben.

Vad behöver ni på en utvecklingsserver respektive på en driftsserver? Fundera på alla verktyg som kan behövas.

Filer och filsystem

Filer

- En Unix-princip är att allt är filer
- Filer har en ägare och rättigheter sätts på nivåerna ägare, grupp, övriga ("världen")
- Filen har förstås en typ / ett format – till skillnad från i en del andra miljöer är det inte hårt kopplat till någonting i filens namn
 - `file <filename>` gissar vad det är för sorts fil
- Verktyg för att hitta och hantera filer

Filrättigheter

Filrättigheter

- Tre set av rättigheter: `|user|group|world|`
- Tre typer av rättigheter: `rwX` (read, write, execute)
- Exempel: `-rwxrwxr-x 1 nevyn nevyn 0 aug 9 19:53 minfil`
- Kan även uttryckas som siffror
 - $r = 4, w = 2, x = 1$
 - $rwX = 4 + 2 + 1 = 7$
 - $rw- = 4 + 2 = 6$
 - `-rwxrw-r--` blir alltså `764`. "allt" blir `777`.

Ändra filrättigheter

- `chown` - byt ägare
 - `chown nevyn minfil`
- `chgrp` - byt grupp
 - `chgrp wheel minfil`
- båda samtidigt
 - `chown nevyn:wheel minfil`

Ändra filrättigheter

- `chmod` - ändra rättigheter
 - `chmod u+rwx minfil`
 - `chmod g-wx minfil`
 - `chmod o+x minfil`
 - `chmod 777 minfil`

Filrättigheter

- Första tecknet då?

- `-` betyder vanlig fil, e g `-rwxrwxr-x`
- `d` betyder directory, e g `drwxrwxr-x`
- `c` betyder "character special", en device, e g `crwxrwxr-x`

- "Sticky bit"

- Hindrar andra än ägaren från att manipulera filen
- Exempel: `drwxrwxr-t 2 nevyn nevyn 4096 aug 9 19:58 foobar`
- `chmod +t <fil>`

Repetition: Kopiera, flytta, etc

- `cp` – kopiera filer
 - `cp fil1 fil2`
 - `cp -r dir1 dir2`
- `mv` – flytta filer
 - `mv fil1 fil2`
- `rm` – ta bort filer
 - `rm fil1`
 - `rm -r dir1`
- `touch` – markera en fil som läst
 - `touch fil1`

Övning 2

- Gör ett directory filexempel, lägg in filer med namnen test1 ... test10 i det (använd gärna en loop).
- Gör ett script som hittar alla filer i det directory där det körs som har namn som börjar med test. Låt det kopiera varje sådan fil till en som har samma namn med tillägget ".bak".
- Testa att köra scriptet som olika användare. Testa sedan att sätta sticky bit på directory filexempel och se hur det blir när ni försöker köra scriptet som olika användare.

Övning 2

```
for i in {1..10}
do
    touch test$i
done
```

Övning 2

```
#!/bin/bash
for i in `ls test*`
do
    cp $i $i.bak
done
```

Skriiva, äändra och hitta filer

Skriv till filer

- Skriv över fil1:
 - `echo "hej" > fil1`
- Lägg till i slutet av fil1 (append):
 - `echo "hej" >> fil1`
- Skicka stderr till en fil med `2>`
 - `echo "hej" >> fil1 2> &1`
- Input från fil1:
 - `cat < fil1`

Regular expressions-repetition (söka/ersätta)

- Användbart för att hitta strängar och för att byta ut strängar
- Exempel (från Linux 1):
 - `${arr[*]}//rad//foo}`
- Ytterligare verktyg: `sed`
 - Exempel: `cat fil1 | sed 's/abc/ABC/g'`

Övning 3

- Skapa en fil fil1 med ett antal namn:
 - Tintin
 - Milou
 - Haddock
 - Kalkyl
 - Dupond
 - Dupont
- Gör ett skript som går igenom filen, byter ut "Milou" mot "Milou Hund" och skriver resultatet till en fil fil2.
- Extraövning: Gör nu samma sak på (minst) ett annat sätt.

Övning 3

```
#!/bin/bash
while read a
do
    echo ${a//Milou/Milou Hund}
done < fil1 > fil2
```

Använda **sed** istället!

- Som scriptet nyss:
 - ``sed 's/Milou/Milou Hund/g' < fil1 > fil2`
- Eller gör ersättningen i samma fil:
 - `sed -i -e 's/Milou/Milou Hund/g' fil1`

Hitta filer

- Find

- `find {sök i vilken mapp} {sökpredikat och andra uttryck}`

- `find . -name "*.sh"`

- `find . -newermt "2022-08-10"`

- Och till och med saker som

- `find . -name "*.png" -exec echo convert {} -resize 64x64
thumbs/{} \;`

- Oändligt med varianter! Se `man find`.

Övning 4

- Gör ett script som hittar alla filer som heter något som slutar på "sh" (i det directory där scriptet körs plus underdirectories) och som skriver ut resultatet av ett `ls -l` för de filerna.

Övning 4

```
#!/bin/bash
for i in `find . -name "*sh"`
do
    ls -l $i;
done
```

Överföra filer

Överföra filer: ftp, scp

- `ftp` - Osäkert (allt i klartext) och krångligt (kontrollkanal och datakanal)
- `scp` - tänk `cp` över `ssh`
 - `scp file remote_user@remote_ip:/remote/directory`
 - `scp remote_user@remote_ip:/remote/directory/file file`

Överföra filer: sftp

- `sftp` - tänk `SSL/TLS-säkrad ftp`
 - `sftp remote_user@remote_ip`
- några sftp-kommandon
 - `ls`
 - `cd <dir>`
 - `get <file>`
 - `put <file>`

Övning 5

- Testa att föra över en fil med scp respektive med sftp. Det går att köra mot localhost.
 - Vad behöver du ha igång på din dator för att det skall fungera?

Överföra filer över http och dylikt

- `wget` -- enkel hämtning från främst `http(s)`
 - `wget https://www.dn.se/`
 - `wget -O test.html https://www.dn.se/`
- `curl` -- i princip samma som wget men mycket mer kraftfullt, många olika protokoll och funktioner
 - `curl https://www.dn.se/`
 - `curl -X POST http://www.yourwebsite.com/login/ -d 'username=yourusername&password=yourpassword'`

Övning 6

- Hämta data från `https://www.nackademin.se` med `wget`
- Hämta data från `https://www.nackademin.se` med `curl`
- Kan du logga in i studentportalen med `curl`?

Fiilträd

- tree – ett verktyg för att enkelt visualisera filträd

- apt install tree

```
nevy@linmishi ~/D/n/nackademin-linux2 (main)> tree .
.
├── 00-gruppuppgift.md
├── 00-lektionsplan.md
├── 01-extralabb.md
├── 01-lektion.md
├── 02-lektion.md
├── img
│   ├── alm.png
│   ├── companies.png
│   ├── gnu-linux-distribution-timeline.png
│   ├── minimal.png
│   ├── passwd-file.png
│   ├── pki.png
│   ├── tidslinje.png
│   ├── traditional.png
│   └── unix-family.jpg
├── README.md
└── renders
    ├── 00-gruppuppgift.pdf
    ├── 00-lektionsplan.pdf
    └── 01-lektion.pdf

2 directories, 18 files
```

tree

- begränsa hur många nivåer ner den söker sig:
 - `-L <antal>`
- Enbart directories:
 - `-d`
- Skriv ut hela sökvägen
 - `-f`
- Skriv ut filrättigheterna
 - `-p`

Övning 7

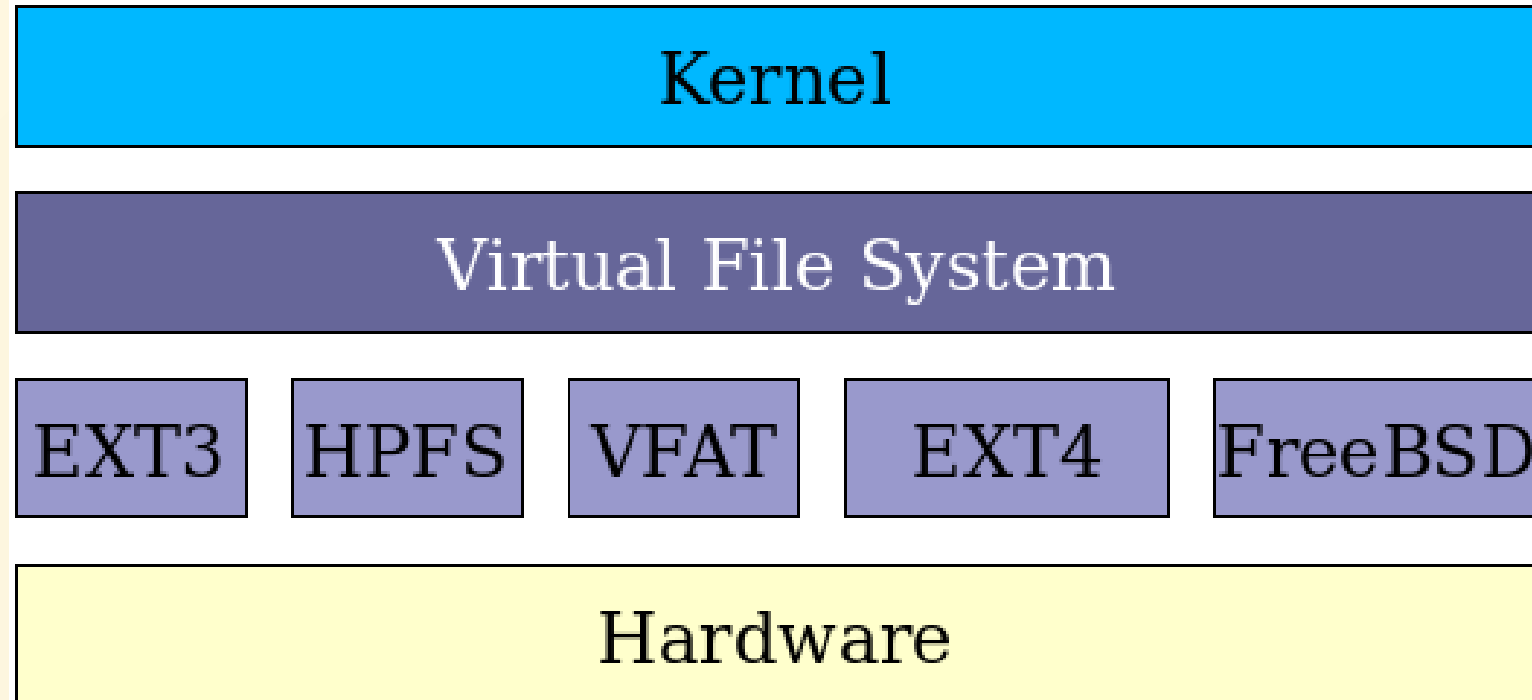
- Installera tree (om du inte redan har den).
- Titta på vad som ligger på din Linux-burk genom att göra ett "träd" från /
 - Tips: Det blir betydligt enklare och tar mindre lång tid om du nöjer dig med att titta på första nivån
- Gör nu samma sak som nyss men med alla filrättigheter utskrivna

Övning 7

- `tree -L 1 /`
- `tree -L 1 -p /`

Filsystem

- Olika sätt att lagra data. Linux har stöd för över 100 olika filsystemstyper



mount

- `mount` gör ett filsystem tillgängligt
 - Tar en "device", e g `/dev/sdb2`
 - Och gör den tillgänglig vid "mount point", dvs ett ställe i ditt existerande filträd (nånstans under `/`)
 - `/mnt` används för temporära mountningar
 - `/media/{nånting}` vanligt för extra hårddiskar
 - kan vara precis varsomhelst, e g `/opt/data`
- `/etc/fstab`

Filsystemstyper: Journaling FS

“ A journaling file system is a file system that keeps track of changes not yet committed to the file system's main part by recording the intentions of such changes in a data structure known as a "journal", which is usually a circular log. ”

(Wikipedia)

- Exempel: Ext4, ZFS, ReiserFS

Filsystemstyper: Versioning FS

“ A versioning file system is any computer file system which allows a computer file to exist in several versions at the same time. Thus it is a form of revision control. ”

(Wikipedia)

- Exempel: **NILFS**

Filsystemstyper: Distributed FS

“ Distributed file systems do not share block level access to the same storage but use a network protocol. These are commonly known as network file systems, even though they are not the only file systems that use the network to send data ”

(Wikipedia)

- Exempel: **NFS**

Filsystemstyper

- Standard i de flesta moderna Linux-system är `ext4`
- macOS använder `apfs` (tidigare `hfs+`)
- Windows använder `ntfs` (tidigare `ext32`, ibland `exfat`)

Mount external filesystem: NFS

- `mount -t nfs <ip>:<path> <dir>`
- E g: `mount -t nfs 10.0.1.10:/backups /var/backups`

Tillbakablick, reflektion, kommentarer ...