Unix

Henrik Lindgren, 2000

Uppdaterat och reviderat, Henrik Lindgren, 2002, 2004

Idag används det på Chalmers i princip två operativsystem Unix/Linux och Windows. Operativsystemet är den programvara som är länken mellan hårdvaran i datorn (cdrom, diskettenhet, högtalare etc) och de program du som användare kör. Man kan i alla fall uttrycka den förenklade versionen av vad ett operativsystem är lite slarvigt på det sättet, i verkligheten är det ett oerhört komplext djur. De vanligaste operativsystemen på marknaden är idag Windows av diverse utgåvor såsom 2000, XP, 98… Unix/Linux och MacOS, men betydligt fler existerar. Det som skiljer dem åt är att en del är kostnadsfria andra inte och naturligvis tekniska detaljer såsom deras uppbyggnad och prestanda. Skillnaden mellan Unix och Linux är främst att Linux är kostnadsfritt medans det som går under samlingsnamnet Unix oftast inte är det. Under kategorin Unix faller Solaris från SUN, HPUX från Hewlett Packard. Man kan säga att Unix är ”basstrukturen” för hur operativsystemet är uppbyggt, sedan har varje tillverkare gjort sina egna nischer på den strukturen.

System i användning

Matematik/Datavetenskap/Bioinformatik

För en systembeskrivning se: www.medic.chalmers.se

Elektroteknik/Datorteknik/IT-linjen

För en systembeskrivning se: www.medic.chalmers.se

Teknisk Fysik/GU-Fysik

För en systembeskrivning se: www.dd.chalmers.se

Maskinteknik/Automatiseringsteknik

För en systembeskrivning se: http://mdc.chalmers.se

Som student här är det av enorm vikt att man förstår datorsystemet då man skall ägna sina nästa 4 år i symbios med det. Bästa sättet att förstå Unix är att använda det. Man kan naturligtvis läsa allt om Unix först och sedan börja använda det för att känna sig säkrare innan man sätter sig framför datorn. Problemet med det är att inget system är det andra likt. Varje grupp av administratörer har sin åsikt om hur ett system skall se ut för att fungera så bra som möjligt, så fullärd blir man först efter att ha testat. Förhoppningen med dessa anteckningar är att

åskådliggöra hur Unix i sig ”basen” fungerar samt påvisa hur just de unika detaljerna med varje system är tänkta att användas. Detta är en introduktion , ni lär er först när ni tar steget ut och testar och tänk på att det värsta som kan hända är att ert konto slutar fungera – inget annat! Händer det så kontaktar man Helpdesk. Så lek och lär!

Att logga in

Lärdomen från min tid i Helpdesk är att detta inte är en helt enkel uppgift det finns faktiskt många icke triviala problem med en till synes så enkel procedur!

Det första man möts av när man sätter sig ned vid sin arbetsstation är en svart skärm. Troligtvis är det skärmsläckaren som är igång men det kan också vara strömproblem. För att inte skärmen skall förstöras (bilden skall brännas fast i ”glaset”) finns det en skärmsläckare som går igång ett tag efter att ingen använt datorn, alternativt har någon lämnat sin dator och låst skärmen. Man Man väcker skärmsläckaren genom att röra på musen eller trycka på någon tangent på tangentbordet. Är det någon annan student som låst skärmen dyker det upp en ruta som talar om vem och har man tur kan man kasta ut den personen genom ett enkelt musklick på angiven plats.

När man väckt släckaren dyker det upp en ruta som ber en ange sitt användarnamn samt lösenord - detta är den så kallade login-bilden.

Användarnamnet är ens identitet i datorvärlden. Det finns många synonymer till användarnamn, ett par är användare, login-namn, username, user. Mitt användarnamn är `henli`. Användarnamnet är unikt inom Chalmers om det är ett personkonto. På Chalmers kan det finnas beroende på vilket datorsystem man sitter på (mdstud – Datavetenskap/Matematik, dd – fysik…) två typer av konton. Personkonton (personliga konton) eller laborationskonton (ett konto delat av flera individer). Skillnaden mellan konto och användare är att ett konto har ett tillhörande användarnamn men även fler saker såsom diskutrymme, utskriftsmöjlighet etc. Man kan alltså existera som användare utan att ha någon lagringsplats för sina dokument etc. Detta är extremt ovanligt men Unix använder sig av det för speciella egen ändamål.

Eftersom man inte vill att obehöriga skall kunna utnyttja systemet har varje användarnamn ett tillhörande lösenord. Lösenordet kan vara samma för flera olika användare. Det är kombinationen användarnamn och lösenord som tillsammans står för säkerheten att ingen kan utföra olagligheter utan att ställas ansvarig för det. Användarnamnet är offentligt, lösenordet känner man endast själv till, om man nu inte slarvar och avslöjar det för någon annan!

Lösenord skall ni skydda på samma sätt som ni skyddar era kontokortskoder, annars kan ni ställas ansvariga för vad andra gjort genom att logga in som er. Tappar ni bort ert lösenord eller tror att någon annan kan det skall ni genast byta lösenord och gärna kontakta den lokala driftavdelningen snarast möjligt!

Det finns tre mycket vanliga fel väl värda att nämna när det gäller att logga in:

1. När man fyller i sitt användarnamn och lösenord skall man vara mycket försiktig så att man av misstag inte skriver lösenordet där användarnamnet skulle stått för då kan alla som sitter i närheten läsa lösenordet i klartext. Att sedan ta reda på användarnamnet är inte så svårt. Skulle detta ske kan de använda ens identitet för att skicka kärleksbrev från en själv till andra, surfa efter snusk eller andra (o)trevligheter.

2. Ett annat vanligt fel man gör är att man väcker skärmsläckaren genom att trycka på mellanslagstangenten, vilket medför att man får ett felmeddelande om att användaren inte existerar. Det är inte så konstigt, användaren: `<space>henli` är inte samma som `henli`.
3. Om de som generarat era konton inte riktigt tänkt sig för kan de ha använt ett typsnitt på text som gör att det är omöjligt att skilja en etta ”1” från ett ”l”. Som i denna text. Det behöver alltså inte vara något fel på er eller ert lösenord utan snarare på rutinera för att generera och skriva ut lösenord. Lyckas ni inte med första login försöket så testa igen och byt ut tveksamma tecken som ett litet L mot en etta istället.

Lösenord och användarnamn används för att ge tillgång till personkontot. Alla som studerar vid Chalmers får ett så kallat personkonto vid den sektion han / hon studerar vid. Ett personkonto består av (med vissa variationer):

- Utrymme att spara information på
- Utskriftsmöjlighet
- Hemsideutrymme
- Möjlighet att skicka och ta emot epost.

Användarnamnet och lösenordet fungerar endast att använda på det datorsystemen som den sektion man tillhör tillhandahåller. Som fysiker kan man alltså endast använda datorerna som finns vid fysik, det skulle inte fungera att gå bort till Datavetenskap och försöka använda deras datorer med samma användarnamn och lösenord som vid fysik. Detta gäller under förutsättning att de som är systemansvariga vid Datavetenskap inte ordnat så att det fungerar där också (det är dock undantag).

Filer och filsystemet

Vad är en fil?

Datorn byggdes ursprungligen för att lösa komplexa matematiska problem, sådana problem genererar mycket data (information). För att man senare skall kunna titta på dessa data måste de sparas, detta görs i vad som benämns filer på fackspråk. Då det enda språk datorn egentligen förstår är ettor och nollor så använder datorn sig av dessa för att spara data i filer. Så i grund och botten ser all information ut på samma sätt för datorn - som en lång rad av ettor och nollor. En bild är en rad av ettor och nollor och så även en text.

Säg nu att man vill spara en färgbild. En bild består av massor av små punkter i olika färger. Lämpligt vore alltså att spara vilken färg varje punkt har. Med en bild på 300*200 punkter blir det alltså 60 000 färgkoder som skall sparas. Varje färgkod tar upp en byte det vill säga åtta ettor/nollor i rad. Det skulle alltså bli totalt 480 000 ettor/nollor i en följd.

Är det då lika lämpligt att spara en svart och vit bild på 300*200 punkter där endast 20 punkter används med samma metod? Nej, det skulle kräva en fil med samma storlek som den ovan. Här kan man spara bytes genom att välja ut färgen med minst punkter använda i bilden och spara koordinaterna för vart de punkterna är i bilden. Man sparar alltså x,y-koordinaterna. Säg att de tar 2 byte per par. Detta ger en fil som är 40 byte (320 ettor/nollor i följd) stor. Är man riktigt slug inser man att man inte behöver spara varje färg med en en bytes kod som ovan, det finns ju bara 2 färger svart och vit.

Som man ser ovan så används alltså två helt olika metoder för att spara bilder beroende på dess struktur och antal färger. Man pratar om olika format på bilderna det vill säga olika strukturer på hur de sparats.

Två mycket vanliga bildformat är `gif` respektive `jpg`. Skillnaden mellan formaten är hur datan sparas beroende på vilken bildkvalitet man vill ha. `gif`-formatet drar ner lite på bildkvaliteten (kan ej hantera så många färger), och tar därmed mindre plats eftersom mindre data måste sparas, medans `jpg` är mer inriktat på att verkligen återge original bilden med alla kontraster och färger.

För att spara och läsa en fil i ett visst format använder man sig av ett datorprogram. Programmet tolkar det man ritat och sparar ner det enligt givet format (`jpg`, `gif`). Likaså läser det en fil, tolkar innehållet, och visar bilden på skärmen eller ger dig möjlighet att skriva ut bilden. Alla program klarar inte alla format till exempel kanske inte ett ritprogram kan tolka alla olika bildformat utan endast ett fåtal. Det gäller alltså att hitta rätt program för rätt fil givet att man vet vad det är för typ av fil till exempel om det är en text eller en bild.

Filändelser

För att lätt kunna avgöra vad det är för typ av fil - alltså vilket format filen är sparad i och därmed vad för typ av program som skall användas för att tolka om datan och presentera den - använder man sig av filändelser. Det finns inget tvång på att använda filändelser egentligen men det är att rekommendera! Noteras bör att en del program vägrar tolka filer som inte har rätt filändelse.

`filnamn.filändelse` exempel *`recept.txt`*

Det finns otaliga "standardiserade" filändelser att använda sig av, nedan ses ett urval. Lägg märke till att det ibland finns två förkortningar för samma format eftersom olika tillverkare av program beslutat sig för olika "standardnamn" på filändelsen.

Ändelse	*Förkortning för*	*Fil-innehåll*
`txt`	**Te**x**t**	Vanlig text
`jpg` eller `jpeg`	**J**oint **P**hotographics **E**xperts **G**roup	Bild i `jpg`-formatet
`gif`	**G**raphics **I**nterchange **F**ormat	Bild i `gif`-formatet
`bmp`	**B**i**t**m**a**p	Bild i `bmp`-formatet
`html` eller `htm`	**H**yper**t**ext **m**arku**p**-**l**anguage	Kod för hemsidor (sparas som vanlig text)
`ps`	**P**ost**S**cript	Dokument sparat i PostScript-formatet.
`pdf`	**P**ortable **D**ocument **F**ormat	Dokument sparat i `pdf`-formatet.
`mp3`	**M**oving **P**ictures Expert Group-1 Layer **III** audio	Ljud komprimerat enligt `MPEG-1 Layer III`-

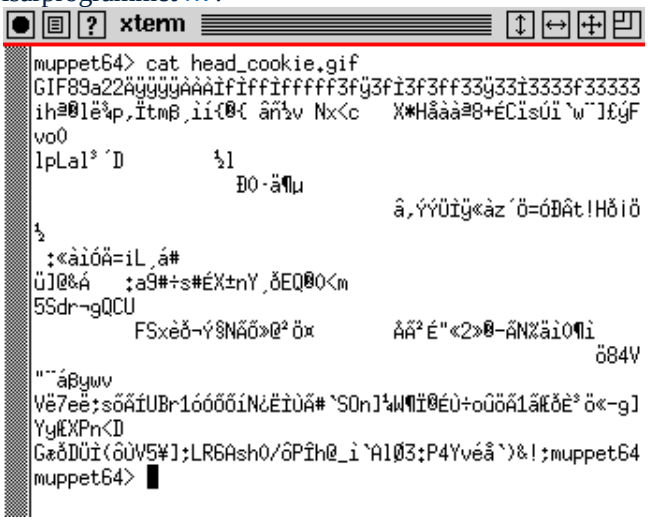
		formatet

När man vet vad det är för typ av data som sparats kan det vara av intresse att veta vilket program man kan använda för att återläsa datan och presentera den. För att ta reda på om ett program klara av att tolka ett format kan man antingen läsa dess manualblad (se avsnittet om att söka hjälp) eller starta programmet och läsa den ”oftast inbyggda hjälpen”. Programmen nedan är förslag på lämpliga program att använda.

Ändelse	***Lämpligt Program***	***Kommentar***
`txt`	`cat, emacs, more, less, netscape`	
`jpg` eller `jpeg`	`xv, gimp, netscape`	Jpg formatet används ofta på webben. Det är ett format där man gör en avvägning mellan storleken på filen och bildkvaliteten. Man kan alltså få en ganska liten fil till priset av bildkvaliteten.
`gif`	`xv,gimp, netscape`	Ett annat väl använt lagringsformat för bilder på webben. Klarar inte lika många färger som `jpg`-formatet.
`bmp`	`xv, netscape`	Ett format framtaget av Microsoft, var tidigare populärt på webben men har på senare tid nästan försvunnit.
`png`	`netscape`	Är ett nytt bildformat som snabbt vuxit sig stort på webben. Klarar av det mesta som jpg och gif gör och dessutom betydligt bättre.
`html` eller `htm`	`cat, more, emacs, netscape, lynx`	HTML är ett språk som används för att bygga hemsidor genom att man skriver korta koder som beskriver hur texten skall visas på sidan. `more` kan man använda om man vill se vilka koder som använts och `emacs` om man vill titta/ändra på något. Av de förslagna programmen är det endast `netscape` som klara av att visa bilder och liknande som man med koder sagt skall finnas med. `lynx` klarar av att visa sidan uppbyggd med text istället för grafik. *Alltså `cat`,`more` visar koden medans `lynx` och `netscape` visar resultatet.*
`ps`	`gv`	PostScript är vad som kallas för ett sidbeskrivningsspråk. I mångt och mycket liknar det HTML. Man anger små koder som berättar hur texten som följer efter koden skall presenteras. Skall den vara understruken, tjock etc.
`pdf`	`gv, acroread`	Pdf är en utveckling av PostScript som är mer lämpad för webben. Det komprimeras (tar mindre plats) bättre än PostScript och har bättre layout möjligheter.
`mp3`	`mpg123`	Mp3 är ett format som bygger på att man från orginalljudet plockar bort våglängder som av människan inte är hörbara.

		Detta minskar mängden data som måste sparas. Man kan med mp3-formatet få ned storleken på orginalljud-filen med faktor 12:1. En fil på 40MB blir ungefär 4MB stor, och ljudet kommer fortfarande för det mänskliga örat vara av CD-kvalitet.

Vad skulle nu hända om man valde fel program för tolkning av en fils data till exempel textvisar kommandot `cat` används på en fil som egentligen representerar en bild, istället för bildvisarprogrammet `xv`.



Resultatet av *cat* Resultatet av *xv*

När man namnger en fil finns det en del regler att följa och en del tecken man inte bör använda. Det kan skapa problem då de har speciell betydelse för operativsystemet (egentligen skalet – shellet). Betydelsen av en del tecken kommer att tas upp senare (under avsnittet om shellet).

Man skall undvika följande tecken i filnamn annars har man snart skapat mer problem än man kan hantera.	, (blanktecken) * < > | & ~ ? ! [ ] \ / ' " ` : ;
Undvikas bör även de svenska tecknen, då de ibland tolkas som svenska tecken och ibland inte. `Ö` kan ibland tolkas som `\` vilket har en lite speciell betydelse för skalet (se ovan).	å ä ö Å Ä Ö

Kommentar

I windows-världen lade man förr till ändelsen `.exe` till filer som var program/kommandon för att markera att de var körbara. I Unix finns det ingen ändelse för körbara program, men det finns två andra sätt att ta reda på om en fil är körbar istället som kommer att diskuteras i stycket om filrättigheter (rättigheten `x` satt eller med kommandot `file`).

Vad är ett filträd?

I och med att man producerar mycket information måste man organisera den på ett enkelt sätt så man själv och andra hittar den. En metod var med filändelser men den räcker inte om man har flera hundra filer därför finns det också filträd.

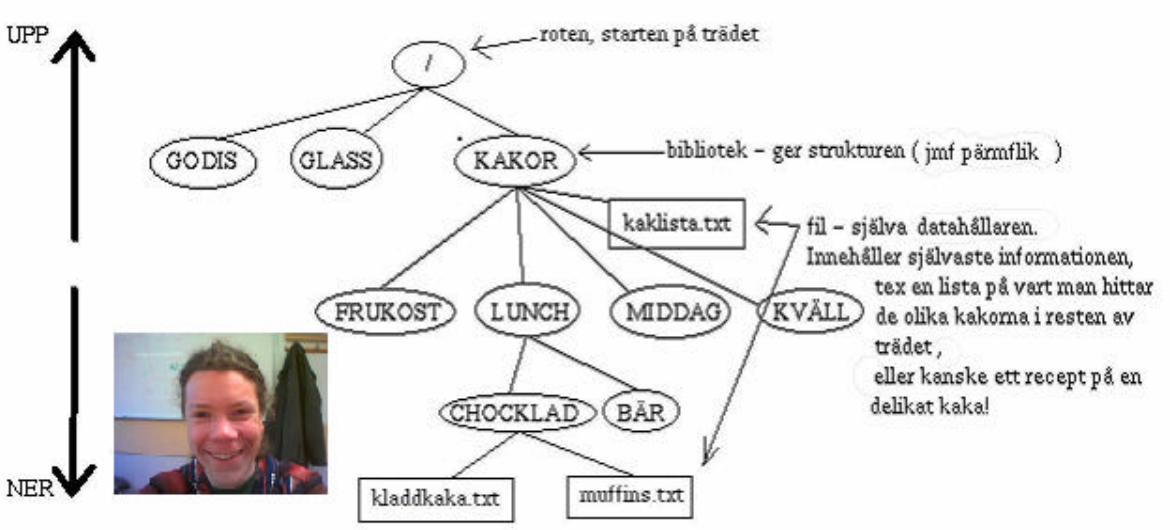
Ett vanligt sätt att organisera sitt kursmaterial är med hjälp av en pärm och tillhörande pärmflikar med siffror eller bokstäver på i ytterkanten. I Unix finns inte pärmflikar utan metoden i Unix för att skaffa ordning och reda är kallas bibliotek (eng. directory), i windows finns samma funktionalitet men där kallas den mappar på grund av den grafiska strukturen de representeras av.

Strukturen för ett filträd efterliknar den som ett träd har med förgreningar och ytterpunkter (noder). Starten på filträdet är roten (/) och kan liknas vid punkten på ett vanligt träd där kronan börjar förgrena sig.

Ett filträd består av filer och bibliotek i en salig "röra". Filerna är datahållarna och biblioteken ger strukturen, som med pärmen där flikarna ger struktur och bakom flikarna finns papperna med information.

Cirklarna markerar här bibliotek och rektanglarna är filer. Man brukar prata om att röra sig uppåt och neråt i filträdet, när man rör sig uppåt menas i riktning mot roten.

Noteras bör att chocklad stavas choklad inte chocklad, i bildserien som följer!



Orienteringskommandon

För att man skall kunna ha någon nytta av strukturen i trädet måste man kunna röra sig i det och titta på vad som finns i varje bibliotek. Till ens hjälp finns det en del kommandon gjorda för just detta. Vad ett kommando är kommer att förklaras tydligare i efterföljande stycke, man kan se det som en informationshanterare. Man skriver kommandots namn och tillbaka får man en del information eller också ber man det utföra en uppgift åt en som till exempel att kopiera data. Så fort man loggat in på sin arbetsstation så befinner man sig någonstans i ett enormt filträd (oftast i sitt hembibliotek). *Man benämner det biblioteket man står i för tillfället för aktuellt bibliotek.*

Kommando	*Förkortning för*	*Syfte*
`cd`	Change working Directory	Byter aktuellt bibliotek, används för förflyttning i trädet
`pwd`	Present Working Directory	Vart befinner jag mig nu i trädet? Sökvägen anges från roten.
`ls`	List	Listar innehållet i ett bibliotek.

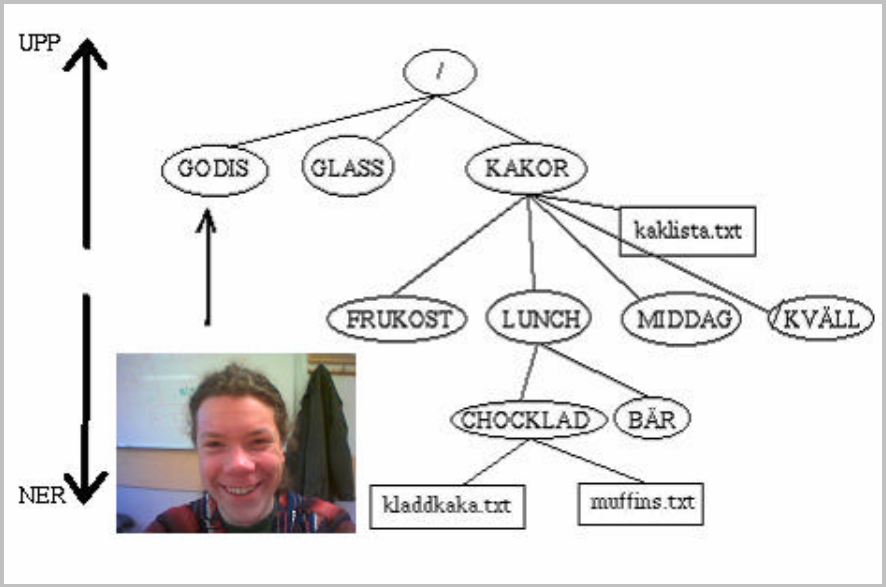
Exempel (Kakmonster-spelet)

Glade Handledaren har lyckats irra bort sig i filträdet. Det är lunch och han bara längtar efter att få sätta i sig en chokladkaka. Men först måste den bakas, och då måste man ha receptet. Han behöver hjälp. Glade Handledarens aktuella bibliotek är `/GODIS/`.

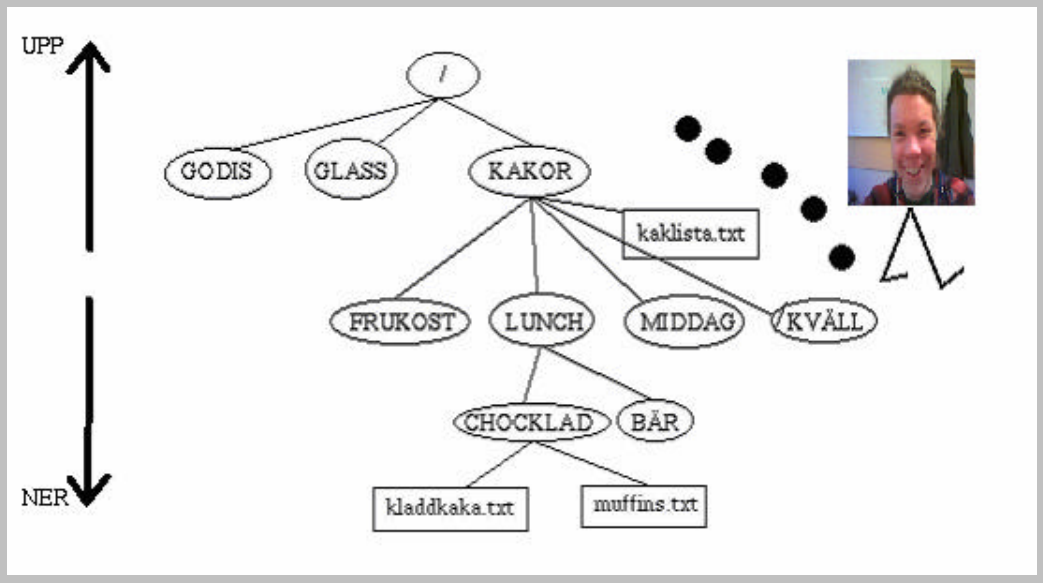
```
> Kommando
Utskrift från kommandot
```

Effekt för Glade Handledaren

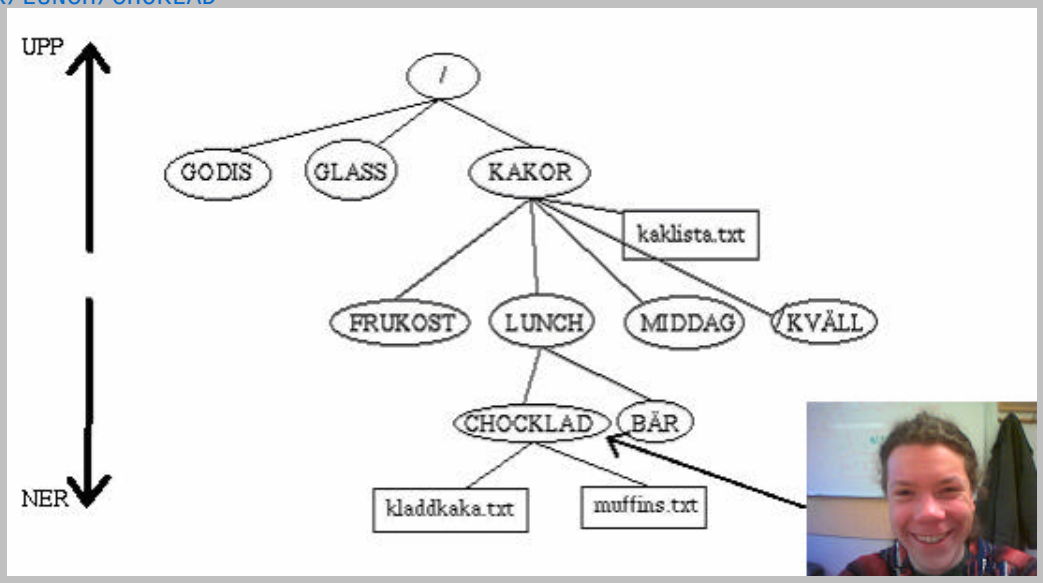
```
> pwd
/GODIS
```



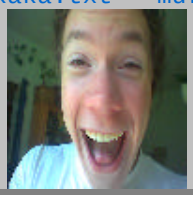
```
> cd /KAKOR/LUNCH/CHOKLAD/
```




```
> pwd
/KAKOR/LUNCH/CHOKLAD
```




```
> ls
kladdkaka.txt   muffins.txt
```



Efter förflyttningen har Glade Handledarens aktuella bibliotek bytts till `/KAKOR/LUNCH/CHOKLAD`

Sökväg

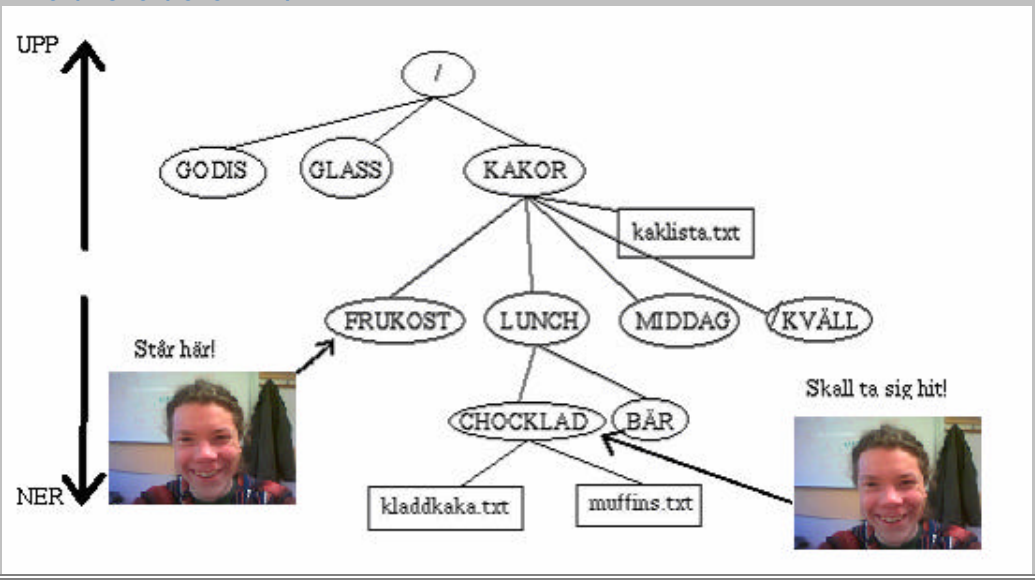
Det finns två sätt att använda sig av när man anger sökvägen till en fil eller vart man står i filträdet. Man kan använda den absoluta sökvägen eller den relativa sökvägen.

Om man skulle flytta Glade Handledaren med hjälp av den absoluta sökvägen från där han står nu `/KAKOR/FRUKOST/` (enligt bild nedan) till dit han vill skulle det se ut som:

```
>                                                    Kommando
Utskrift från kommandot
```

Effekt för Glade Handledaren

```
> cd /KAKOR/LUNCH/CHOKLAD/
```

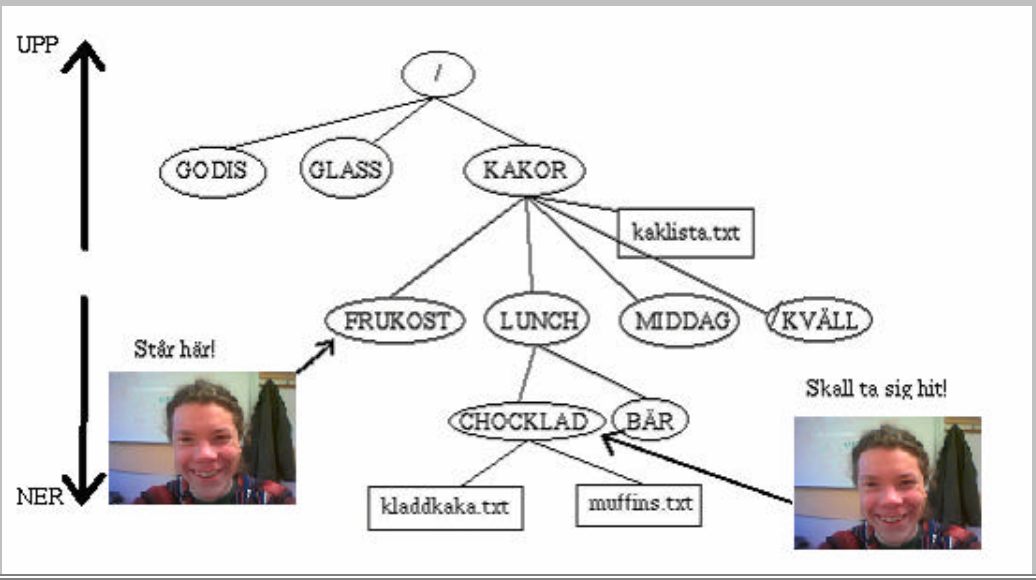


Först tar man alltså sig upp till roten (`/`) och därifrån tar man sig nedåt igen (`KAKOR/LUNCH/CHOKLAD/`). Använder man istället den relativa sökvägen behöver man inte gå till roten (`/`) först utan man kan ta en genväg.

```
> Kommando
Utskrift från kommandot
```

Effekt för Glade Handledaren

```
> cd ../LUNCH/CHOKLAD/
```



Nu går han istället endast ett steg (istället för två) bakåt till biblioteket kakor. Efter ett tag när man lärt sig hur filträdet ser ut - vad biblioteken heter, så kan man spara en hel del tid och tangenttryckningar på att använda den relativa sökvägen. Ett filträd kan vara ganska djupt och innehålla minst 15-20 bibliotek på djupet (eller nivåer om man så vill).

```
.
Nuvarande bibliotek
..
Biblioteket ett steg uppåt i filträdet
```

Följande stycke bör läsas sakta och flera gånger under nyktra omständigheter!

Till ens hjälp att att röra sig relativt sin nuvarande position finns de mycket speciella biblioteken (`.`) och (`..`). (`.`)-biblioteket är en beskrivning av det nuvarande biblioteket, det vill säga där man står just nu. (`..`)-biblioteket är ens hjälp att röra sig ett steg bakåt (eller uppåt om man nu så vill), från den nuvarande positionen i filträdet.

Absolut sökväg
Man anger alltid positionen man skall till/är på utgående från roten (`/`).

Relativ sökväg
Man anger positionen man skall till relativt sin nuvarande position.

Kommentar

Säg att man står i `/KAKOR/LUNCH/CHOKLAD/`-biblioteket och skriver `cd /KAKOR/`. Det som operativsystemet nu gör för att byta bibliotek är just att använda sig av (`..`)-biblioteket. Genom att gå ett steg uppåt i filträdet, jämföra nuvarande position med dit den skall, och sedan gå uppåt i filträdet igen eller neråt beroende på vart man skall. (..-biblioteket visas normalt inte

om man skriver `ls` utan det krävs att man skriver `ls -al` för att se det med alla bibliotek och filer, förklaring kommer senare.)

Att bygga om filträdet

Allt eftersom man arbetar vill man kunna förändra filträdsstrukturen, bygga ut den med nya grenar eller ta bort en del av trädet. För att åstadkomma detta finns det naturligtvis en del färdiga verktyg - kommandon:

Kommandonamn	*Förkortning för*	*Effekt*
`cp`	**cop**y	Kopiera en eller flera filer
`rm`	**rem**ove	Ta bort en eller flera filer
`mkdir`	**m**a**k**e **dir**ectory	Skapa ett bibliotek
`rmdir`	**rem**ove **dir**ectory	Plocka bort ett bibliotek
`mv`	**m**o**v**e	Flytta en eller flera filer (effekten blir samma som `cp` följt av `rm`)

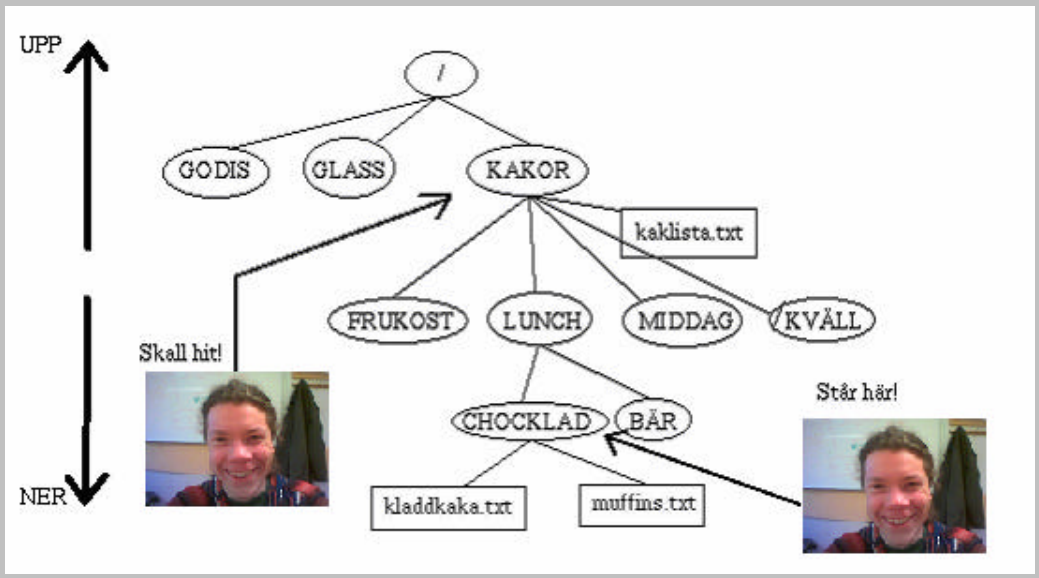
Tillbaka till Glade Handledaren

Glade Handledaren blir lätt trött på att behöva springa runt vid varje måltid i de olika lunch-biblioteken och leta kakrecept. Han beslutar sig för att bygga ut filträdet med ett bibliotek `smaskens` som innehåller hans favoriter.

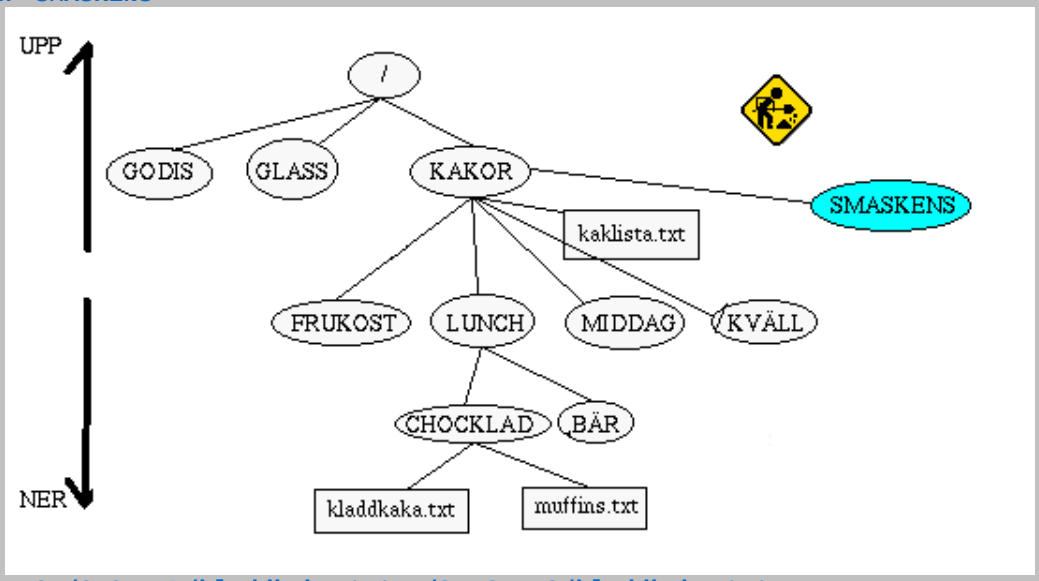
```
> Kommando
Utskrift från kommandot
```

Effekt för Glade Handledaren

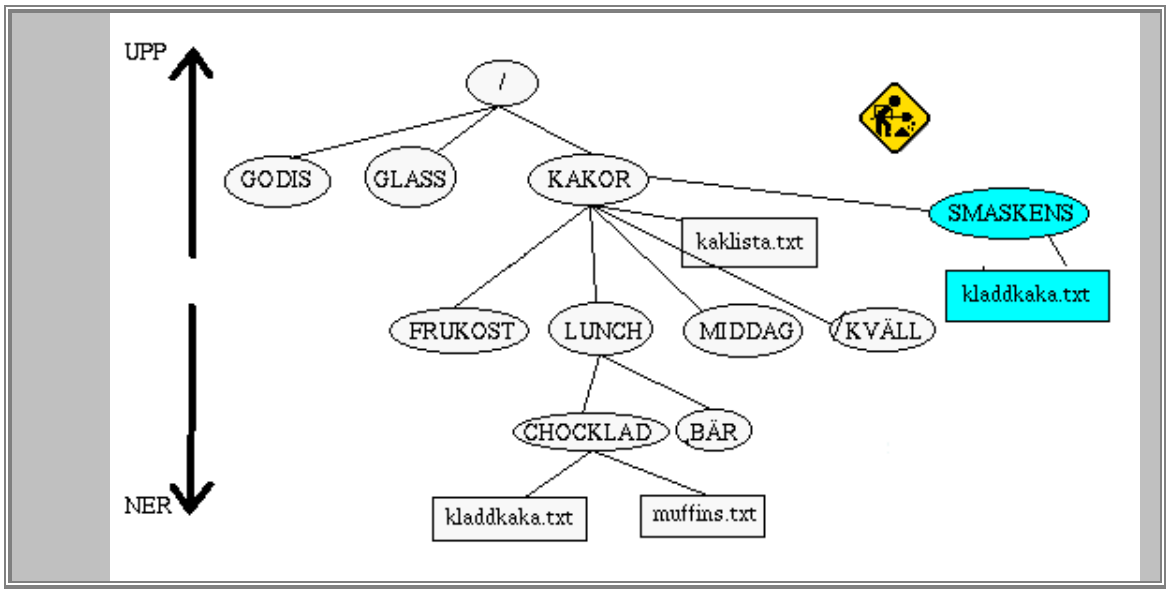
```
> cd ../../
```




```
> mkdir SMASKENS
```




```
> cp LUNCH/CHOKLAD/kladdkaka.txt ./SMASKENS/kladdkaka.txt
```



Gömda filer

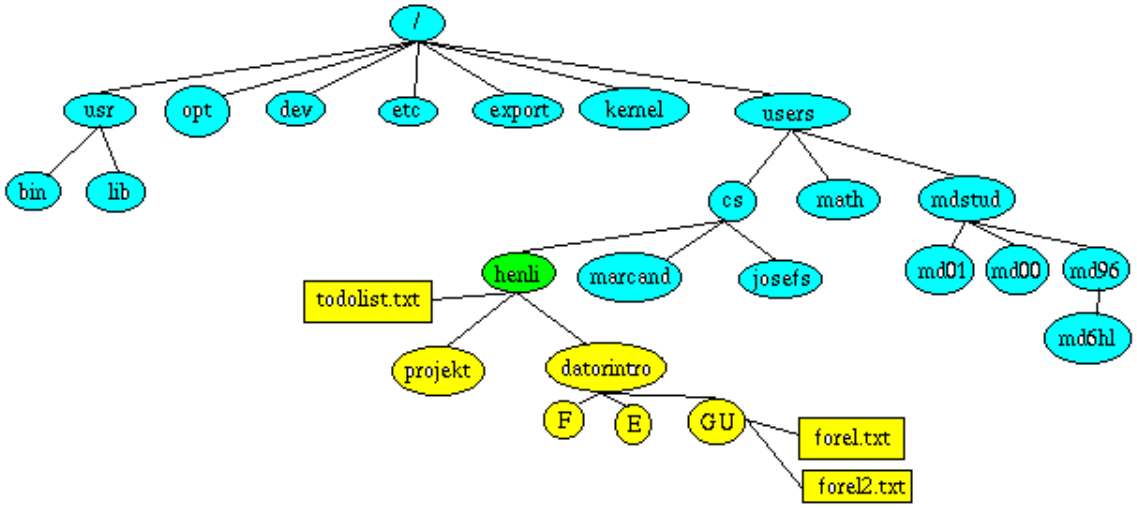
När man skriver `ls` och får en listning på innehållet i det bibliotek man står i får man inte hela sanningen om bibliotekets innehåll. Ibland finns det i bibliotek andra bibliotek eller filer som startar med en (`.`). Detta är filer och bibliotek som oftast berör inställningar till olika program eller systemspecifika filer/bibliotek. Saker som rör hur det ser ut när man loggat in, vilka rättigheter man har och en del annat. De flesta filerna skapas av olika program eller er lokala systemadministratör och skall icke manipuleras med - eller rättare sagt, man gör det på egen risk. Det värsta som kan hända är att man inte kan logga in igen men då kan driften hjälpa! Det finns naturligtvis (`.`)-filer som man som användare kan ändra på utan att komma i problem med systemansvarig också, lite kul måste man få ha.

Fil	***Användning***
`.signature`	Används av ens epostprogram, om man vill slippa skriva sitt namn och adress vid varje brev kan man lagra det i denna fil. Varje gång man skickar ett brev fylls slutet av brevet på med det som står i denna fil. Man måste själv skapa denna fil.
`.logout`	Varje gång man loggar ut läses denna filen och utför det som står i den, man kanske vill plocka bort vissa filer varje gång man loggar ut. Då kan man ange det i denna fil. Man måste själv skapa denna filen.
`.cshrc`	En fil som används för att sätta grundinställningarna för ens konto. *Ändrar man på denna skall man inte förvänta sig någon hjälp av de systemansvariga vid problem.* Filen finns när man får sitt konto. Gör den inte det har något gått snett vid skapadet av kontot.

För att få `ls` att ge mera information såsom eventuella (`.`)-filer/bibliotek kan man skriva `ls` följt av en flagga `-a`, det vill säga `ls -a`. En flagga kan man se som en växel. Den kontrollerar vilken och hur mycket information som skall visas, eller vad som skall göras.

Hembibliotek

När man loggar in i systemet hamnar man i sitt hembibliotek. Detta är ens egna lilla vrå i datorns universum. Precis som det faktum att hyresvärden blir aggressiv om man ställer till det i sin bostad kan systemansvariga bli aggressiva om man huserar alldeles för vilt och ofta i sitt hembibliotek så att saker slutar fungera och ger dem extra arbete. Extra arbete betyder mindre fritid och fritid är trevligt, så tänka innan göra är en god parol att leva efter. Hembiblioteket har sin bestämda plats i filträdet och sökvägen till det startar ofta med `/users/` men inte alltid. Med `pwd` kan man enkelt ta reda på var i filträdet just det kontot man använder sig av är placerat, givet att man precis loggat in och inte gått iväg någonstans.



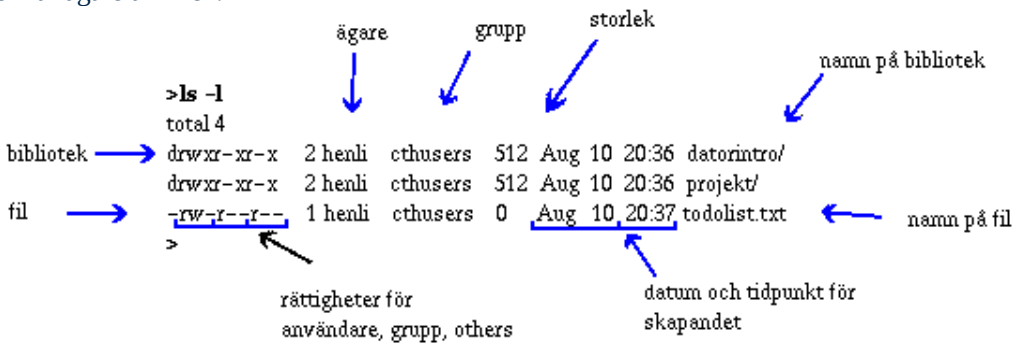
Då man ofta rör sig i sitt eget eller andras hembibliotek finns det ett förkortat skrivsätt för att ta sig till dessa förkortningen man använder sig av är `~`, det så kallade *tilde*-tecknet. Utgående ifrån att man är inloggad som användare `henli` kommer följande hända om man skriver:

Kommando	***Resultat***
`cd ~`	Man hamnar i sitt hembibliotek
`cd ~henli`	Man hamnar i sitt hembibliotek
`cd ~josefs`	Man hamnar i `josefs` hembibliotek
`cd`	Man hamnar i sitt hembibliotek

Filrättigheter (och skyldigheter!)

För att kunna skydda information som till exempel en lista med bankkonton och koder från att läsas av andra finns det i Unix metoder för att sätta rättigheter på filer och bibliotek. Detta ger en möjlighet att reglera vem som skall ha tillgång till ens filer/bibliotek.

Genom att skriva `ls -l` istället för `ls` kan man få reda på en del extra information om filerna/biblioteken i det nuvarande biblioteket såsom datum när filen senast ändrades och vem som är ägare till filen.



Gängbildning är något som är en populär företeelse i vanliga livet och naturligtvis har man inkoorporerat detta begrepp i Unix. I Unix har en fil 3 (del)ägare:

1. En användare
2. En bestämd grupp
3. `others – alla andra`

Den enskilda användaren är den som är ytterst ansvarig för en fil eller ett bibliotek. När man som användare skapar en fil sätts man automatiskt själv som ägare till filen, men inte som ensam ägare. Även en grupp sätts som ägare, eller delägare kanske är mer korrekt. Ägarskapet styr vem som får göra vad med en fil eller ett bibliotek.

Eftersom man som användaren är ytterst ansvarig för en fil så kan man begränsa gruppens rättigheter att röra filen. Vilken grupp som sätts som delägare styrs automatiskt av vilken grupp som äger biblioteket filen skapas i, detsamma gäller bibliotek. Skapar jag som användare `henli` en fil i biblioteket projekt (ovan) så sätts ägaren av filen till mig, den delägande gruppen till samma grupp som biblioteket projekt har, alltså `cthusers`. Endast systemadministratörerna kan ändra ägare och grupp på en fil/bibliotek. Som användare/gruppmedlem har man alltså inga möjligheter att sätta sin kompis som ägare till en fil man själv skapat.

En grupp är en samling användare. Alla användare tillhör minst en grupp och det finns oftast mängder med grupper att vara med i. Som med gäng i verkligheten så har alla grupper i Unix en ledare, ledaren styr över vem som skall få vara med i gruppen. I Unix kallas ledaren för en grupp för gruppens administratör. Det är gruppens administratör man kontaktar om man vill vara med i en grupp. Över administratören står sedan de systemansvariga, de allsmäktiga, gudarna!

Den sista delägaren till en fil/ett bibliotek är `others`, vilket betyder alla användare i systemet. Nu uppstår då frågan: Om jag är med i gruppen `cthusers` och de inte har några rättigheter alls på en fil (de får varken skriva i den eller läsa från den) `doktorander.txt` men `others` har läsrättigheter på filen, vilka rättigheter har jag då? Svaret är de som är satta för `others`.

Som sagt ovan så kan man inte som enskild användare styra över vem som är ägare till en fil/bibliotek eller vilken grupp som sätts som delägare. Men som ägare till filen kan man styra över vilka rättigheter man själv och andra skall ha på filen/biblioteket. Skall medlemmarna i gruppen kunna läsa filen? Skall de kunna ändra dess innehåll? Precis som det sätts en default (förbestämd) grupp som delägare när man skapar en fil så sätts det vissa förinställda rättigheter. Trivs man inte med dem får man ändra dem.

De rättigheter man som användare kan sätta på sina filer och bibliotek är läs-, skriv- respektive exekverings- (körbarhets-) -rättigheter. Respektive rättighet kan sättas för användare, grupp samt `others`. Beroende på om det rör sig om en fil eller ett bibliotek så ger rättigheterna lite olika effekt.

Rättighet	*Fil*	*Bibliotek*
`r`	Titta på innehållet	Söka i biblioteksinnehållet
`w`	Ändra på filens innehåll	Ändra på biblioteksinnehållet
`x`	Exekvera (köra) filen	Göra biblioteket till nuvarande arbetsbibliotek

För att se om en fil är exekverbar kan man titta om dess exekveringsrättighet `x` är satt, detta ser man genom att göra en listning på filen med `ls -l`. En annan metod är att kolla var filen ligger. Ligger den i ett bibliotek `???/bin` är den med största sannolikhet exekverbar. Annars har någon placerat den fel i filträdet. `bin` står för binary, och betyder kort och gott exekverbar. Ytterligare en metod är att använda kommandot `file`, exempel `file netscape`

Ett par exempel på effekten av vad de satta rättigheterna ger för begränsningar är de som ges nedan. Exemplen gäller under förutsättning att gruppen och `others` inte har några rättigheter alls. Med (`-`) menas att ingen rättighet är satt.

Rättighet	*Fil*	*Bibliotek*
`---`	Kan inte göra något	Kan inte göra något
`r--`	Kan titta på dess innehåll	Kan lista bibliotekets innehåll
`rw-`	Kan se och ändra innehållet i filen	Kan lista innehållet i biblioteket och ändra på det (skapa/ta bort bibliotek/filer).
`rwx`	Kan se och ändra innehållet samt exekvera filen.	Kan lista innehållet i biblioteket, skapa/ta bort filer/bibliotek samt göra det till nuvarande arbetsbibliotek.
`r-x`	Filen kan läsas och om den är exekverbar köras.	Kan göra biblioteket till nuvarande arbetsbibliotek, lista innehållet men ej skapa/ta bort filer/bibliotek.
`--x`	Kan exekvera filen om den är en binär.	Användare kan exekvera en binär som de redan känner till namnet på. Göra biblioteket till arbetsbibliotek, men ej lista innehållet. Kan öppna en fil om man kan namnet på den och rättigheterna på filen tillåter det.

Det finns två sätt att ange rättigheterna genom- symboler eller siffror. Från början fanns endast siffermetoden i Unix. Den metoden är för de flesta människor ganska icke-användarvänlig så man tillförde en symbolisk metod också. För information om hur man sätter rättigheter med hjälp av siffror hänvisas till manualbladet för kommandot `chmod (nås med: man chmod)`. Rättigheterna kan alltså sättas för användare, grupp samt alla andra (`others`).

Symbol	*Betydelse*
u	user
g	group
o	others

Kommandot som används för att sätta rättigheter är `chmod`. Säg att man har en fil där användaren från början har läs- och skrivrättigheter, gruppen har inga rättigheter och `others` har samma rättigheter som användaren. Nu vill man att gruppen skall ha läs- och skrivrättigheter och `others` endast läsrättigheter. För att ordna detta skriver man följande (`+/-` adderar respektive plockar bort given rättighet):

```
> ls -l
total 0
-rw----rw-   1 henli   10  0 Aug 10 20:36 todolist.txt
> chmod g+rw todolist.txt
> ls -l
total 0
-rw-rw-rw-   1 henli   10  0 Aug 10 20:36 todolist.txt
> chmod o-w todolist.txt
> ls -l
total 0
-rw-rw-r--   1 henli   10  0 Aug 10 20:36 todolist.txt
```

Kommentar

Som användare kan man ställa om de per default satta rättigheterna som skall ges när man skapar en ny fil. Detta görs med kommandot `umask`. Man måste dock köra `umask` varje gång man loggar in, om man inte ändrar i en fil kallad `.profile`. Naturligtvis rekommenderas det ej att ändra grundinställningen om man inte ställer rättigheterna mot det säkrare hållet, dvs mindre rättigheter åt andra. På en del system är rättigheterna på personkonton satta extremt löst, så att alla har möjlighet att läsa ens filer! För att åtgärda det använder man `chmod` direkt på hembiblioteket. Till exempel genom: `cd` , `chmod go-rwx ../henli`

Kommandon

Kommandon används främst för att kommunicera med operativsystemet, till exempel ändra rättigheter på en fil eller lista ett biblioteksinnehåll, men det finns även andra typer av kommandon för att till exempel läsa texter. Kommandon är små ganska specialiserade program. I windowsmiljön känner de flesta till Word som ett program, det har en uppgift att vara en ordbehandlare och har en grafisk omgivning. Vad skiljer då Word från ett kommando? Kommandon har en mycket mer specifik uppgift än att vara ordbehandlare, ett kommando kanske sköter sortering av textlistor. Kommandon har oftast inte något grafiskt gränssnitt, men det finns naturligtvis alltid undantag som bekräftar regeln.

`ls`, `cp` och `rm` är alla kommandon. Till en del kommandon kan man ange flaggor (växlar) som gör att det informationsflödet som man som användare får kan begränsas eller utökas. Ta

till exempel `ls` och `ls` med flaggan `-l` alltså `ls -l`. Det är oftast inte nödvändigt att få reda på all information som `ls -l` matar ut till en. Därför kan man ge `ls` utan flagga. Genom en flagga kan man också få ett kommando att göra mer saker, till exempel `cp` med flaggan `-R`, det vill säga `cp -R`. Istället för att endast kopiera en fil kan man nu kopiera allt innehåll i alla eventuella underbibliotek också. *Observera att det är ett stort `R` på flaggan och mellanslag mellan kommando och dess flagga! Unix gör stor skillnad vid stor och liten bokstav i kommandon, filer och alla andra tillämpningar. `cp` och `Cp` är inte samma sak*. Det sistnämnda existerar inte. Det är väldigt lätt att tro att en flagga alltid föregås av ett `-` just för att påvisa att det skulle vara en flagga. Så är inte fallet. Flaggor kan ges utan `–` före, det hela beror helt på vilken tomte som skapat kommandot.

Kommandon skriver man in direkt efter prompten (redotecknet) i terminalfönstret (xterm-fönstret). Genom att trycka på returtangenten utförs kommandot. Ibland kan ett kommando ta väldigt lång tid på sig att utföra en uppgift då kan det vara bra att veta hur man avbryter det. Det gör man genom att trycka `Ctrl`-`C`. X:et ovan i xterm står för den grafiska miljön. I den gamla datorvärlden var allt textbaserat, det vill säga inget pekande och klickande. Outhärdligt som det var utvecklade man en grafisk miljö. I Unix kallas basen för den X.

Beroende på vad det är för kommando så kan det ha argument eller flaggor, eller till och med både och. Ett argument är ett filnamn eller någon annan sorts information från användaren, till exempel antalet gånger en sak skall utföras. Vad ett kommando har för flaggor (om det har flaggor) och vad de gör eller vilka argument det kan ta kan man läsa om i manualbladet för kommandot. Nästan alla kommandon finns beskriva i manualbladen, de som kan saknas är de som är egentillverkade på sektionen. Man har i manualbladen byggt en syntax (språkregler) för hur ett kommando skall beskrivas mer om detta kan man läsa under ”Att söka hjälp-kapitlet”. Ett exempel på hur syntaxen kan se ut ges här för `ls`:

```
ls [ -aAbcCdfFgilLmnopqrRstux1 ] [ file... ]
```

`[ ]` - betyder att valet är frivilligt. Väljer man dock att använda både en flagga och en fil, fil kan också ha betydelsen bibliotek här, så skall flaggan komma före biblioteket. Ordningen flagga sedan bibliotek ges av hakparantesernas ordning.

`-aAbcCdfFgilLmnopqrRstux1` - uppräkning av de flaggor som finns att använda (flaggor skall ha ett (`-`) före sig)

Man kan alltså använda `ls` på lite olika sätt, vilket resulterar i att man får ut olika mängder information.

```
> ls
datorintro  projekt  todolist.txt
> ls -l
total 4
drwxr-xr-x  2 henli  cthusers  512 Aug 10 20:36 datorintro
drwxr-xr-x  2 henli  cthusers  512 Aug 12 16:38 projekt
-rw----rw-  1 henli  cthusers    0 Aug 10 20:36 todolist.txt
> ls todolist.txt
todolist.txt
> ls -l todolist.txt
-rw----rw-  1 henli  cthusers  0 Aug 10 20:36 todolist.txt
```

Som framgår av listningen ovan så kan man inte genom att endast använda `ls` urskilja vanliga filer från bibliotek. Ett alternativ är att alltid göra `ls -l` vilket resulterar i massor av (onödig) information. Det bättre alternativet kan istället vara `ls -F`.

Det finns också kommandon som inte behöver några argument men som ändå kan utföra uppgifter. Ett exempel på ett sådant kommando är `pwd` som beskrivits tidigare. Ett annat bättre exempel är `sort`. `sort` kan man använda om man vill sortera något. Det finns två sätt att tala om för `sort` vad det är den ska ll sortera. Man kan ge en fil med den datan som skall sorteras som argument till sort eller så kan man skriva `sort` följt av en tryckning på Return-tangenten. Nu kan man lätt få för sig att `sort` inte fungerar för det händer ju ingenting! Man ser bara en mar kör. Men `sort` fungerar precis som det skall. Det `sort` gör är att vänta på att man skall skriva något på tangentbordet sedan läser den detta som sin data istället för att läsa från en fil. När man skrivit klart det man vill ha sorterat trycker man på Ctrl-`d`. Detta talar om för `sort` att den skall börja sortera. Det är ingen bra ide att trycka Ctrl-`c` här, då kommer `sort` att avslutas utan att sortera datan.

Kommentar

Anledningen till att man använder det till en början krångliga systemet med kommandon i Unix är att det är otroligt flexibelt. Man kan som ni senare kommer få se kombinera olika kommandon och få dem att utföra mycket avancerade uppgifter utan att man behöver kunna ett uns programmering. Man måste dock kunna massor av kommandon :0) .

Kommandotolken (shellet - skalet)

Shellet är gränssnittet mellan dig som användare och operativsystemets kärna. Operativsystemet sköter bland annat fördelningen av minne och kontrollerar så att det man gör verkligen är OK enligt uppsatta regler. Shellets uppgift är att tolka det man skriver på kommandoraden och eventuellt tillföra information innan det skickar det hela vidare till kärnan som utför det man vill ha gjort. Eftersom shellet är ett program som vilket annat program så finns det olika shell med olika funktioner. Det shell som används här är `tcsh`, andra skal är Bourne shell, Korn shell och C shell. `tcsh` är en utveckling av `csh` (C shell). Man kan som användare byta shell men det rekommenderas inte eftersom det kan ställa till vissa problem såsom att gamla kända kommandon och annat slutar fungera.

När man skriver

```
cd ~md6hl/
```

är det shellets uppgift att tolka om `~` till `/users/mdstud/md96/`, sätta samman denna sökväg med `md6hl` och skicka kommandot med dess argument vidare till kärnan [`cd`, `/users/mdstud/md96/md6hl/`]. Kärnan utför sedan bytet av bibliotek efter det att den kollat så att man verkligen har de rättigheter som krävs för att få göra detta byte. Biblioteket som man försöker byta till måste alltså ha sin `x`-rättighet satt för antingen en grupp som man själv och biblioteket tillhör eller för gruppen `others`, om man nu inte äger biblioteket.

Annan funktionalitet som döljer sig i shellet är det förenklade skrivsättet för att kopiera alla filer i ett bibliotek genom att använda sig av `*`. `*` fungerar även för till exempel `rm` och `mv`.

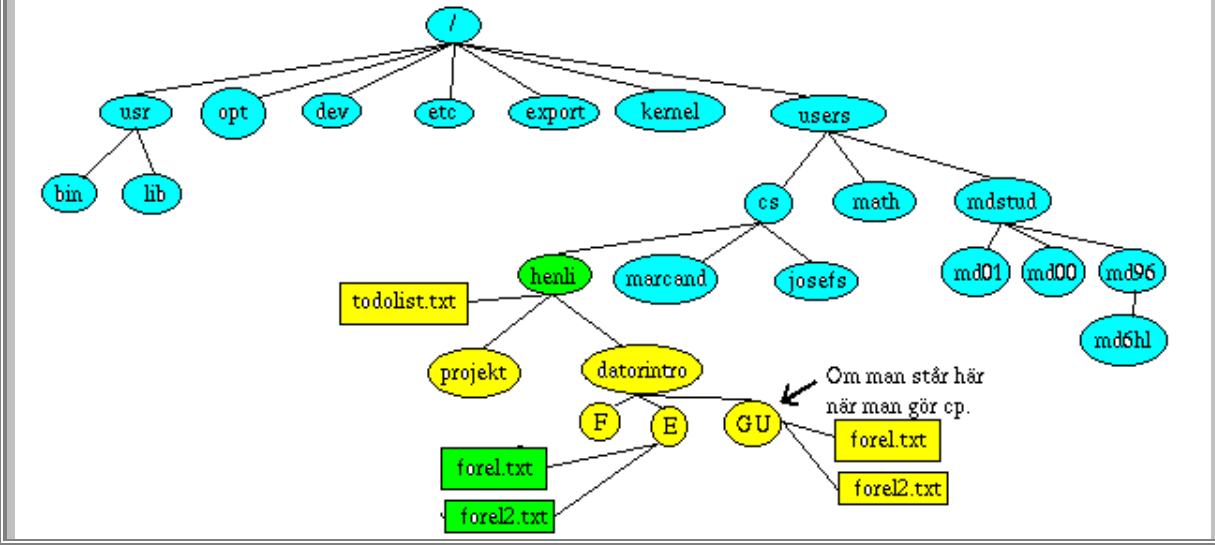
Istället för

```
cp forel.txt ../E/
cp forel2.txt ../E/
```

kan man skriva

```
cp * ../E/
```

Båda sätten ger samma resultat.



Lata som vi människor är så vill vi oftast snabba upp allt, så vi får mer tid över till annat. De som skapade shellen var också människor (ibland undrar man dock när man tittar på källkoden för ett skal) och därmed också lata. Därför införde man funktioner för att slippa skrivaom kommandon, slippa skriva hela kommandonamn och en del andra finesser.

Tangent / tangetkombination	***Effekt***
Pil upp	Bläddrar bland gamla inskrivna kommandon.
Pil ner	Bläddrar bland gamla inskrivna kommandon.

Control-D	Om man står i hembiblioteket för `henli`
	`>cd datorintro/` **[Ctrl-D]**
	`F/  E/  GU/`
	`> cd datorintro/E/`
	
	istället för
	`> cd datorintro/`
	`> ls -F`
	`F/  E/  GU/`
	`> cd E/`
	
	Control-D gör alltså en listning av biblioteketsinnehållet utan att man behöver byta aktuellt bibliotek. Kommandot inklusive eventuella argument kommer också tillbaks efter listningen så man slipper skriva om det. Control-D går även att använda om man skrivit början av ett kommando men glömt vad slutet på det är.

	> **mk**[Ctrl-**D**] mkdir mkfile mkpasswd > **mk**
Tabulatorn	Om man angett en del av en sökväg/kommando kan man få resten ifyllt genom att trycka [TAB]. Kriteriet är att det kan avgöras unikt vilken sökväg eller vilket kommando det rör sig om.

Kommentar

En `xterm` är ett förklätt shell. Ett shell med en beklädnad som gör att man kan flytta runt det på skärmen och förstora texten i det, vilket man gör genom att trycka Ctrl + höger musknapp i `xterm`-fönstret.

Omdirigering

Omdirigering är namnet på metoden för att styra vart ett kommando hämtar respektive lämnar sin information. Normalt hämtas datan som ett kommando använder från en fil eller tangentbordet och efter bearbetning lämnas resultatet ut på skärmen. En del kommandon ger sitt resultat till skärmen, i en del fall är detta kanske inte önskvärt. Man vill kanske spara resultatet i en fil.. Kommandot man använder kanske inte är gjort för att hantera utskrift till fil, man kan då använda omdirigering för att få datan i en fil istället.

Normalt skriver `ls` ut sitt resultat på skärmen

```
> ls -F
datorintro/  projekt/  todolist.txt
```

men med ">" kan man få resultatet i en fil istället. Observera att ingen utskrift på skärmen sker.

```
> ls -F > listning.txt
```

Nu ger en listning :

```
ls -F
datorintro/  projekt/  todolist.txt  listning.txt
```

Dvs det existerar en fil mer än tidigare!

Filen `listning.txt` innehåller

```
> cat listning.txt
datorintro/
projekt/
todolist.txt
```

I omdirigeringens värld brukar man prata om att man styr om standardout. Man styr om det stället kommandot normalt lämnar sin information till, i fallet ovan från skärmen till en fil. Kommandot har i sig ingen känsla för vart det presenter ar sitt resultat, om det är på skärmen

eller till en fil. En del kommandon hämtar också in information från till exempel tangentbordet eller en fil. Stället där ett kommando får sin information ifrån är standardin. Slutligen kan det hända att kommandot misslyckas med sin uppgift vilket brukar resultera i att ett felmeddelande ges. Vart detta hamnar styrs av standarderror. Normalt är att det skickas till skärmen, men även en fil eller skrivare kan vara tänkbart. Ett kommando kan bara ha en standardin, standardout respektive standarderror. Det går alltså inte att få en utskrift både på skärm och fil genom att använda omdirigering.

standardin (stdin)

Härifrån hämtar kommandot sin information (är normalt tangentbordet).

standardout (stdout)

Hit lämnas information från kommandot (är normalt skärmen)

standarderror (stderr)

Hit lämnas information från kommandot om något går snett (är normalt skärmen)

Det finns ett par olika omdirigeringsverktyg till att använda, nedan kommer en kort resume.
Fil kan i de två första exemplen bytas ut mot kommando.

`kommando < fil`

Ta det som står i "fil" och skicka det som indata till kommandot "kommando".

`kommando > fil`

Ta det som kommandot genererar och skriv det till filen "fil". Om filen finns skrivs den över, om den inte finns skapas den.

`kommando >> fil`

Ta det som kommandot genererar och skriv det till filen "fil". Om filen finns skrivs det sist i filen, om filen inte finns skapas den.

`kommandoA | kommandoB`

Ta utdatan från "kommandoA" och skicka den som indata till "kommandoB". "KommandoB" skriver sedan ut resultatet dit dess standardout är styrd - skärm alternativt fil. `|` kallas för pipe (rörledning).

Kommentar

Shellets uppgift i fallet ovan är att skapa filen `listning.txt` och se till att det som `ls` producerar skrivs ner till denna fil istället för på skärmen. Det är alltså shellet som har omdirigerings funktionaliteten inbyggd i sig.

Processer

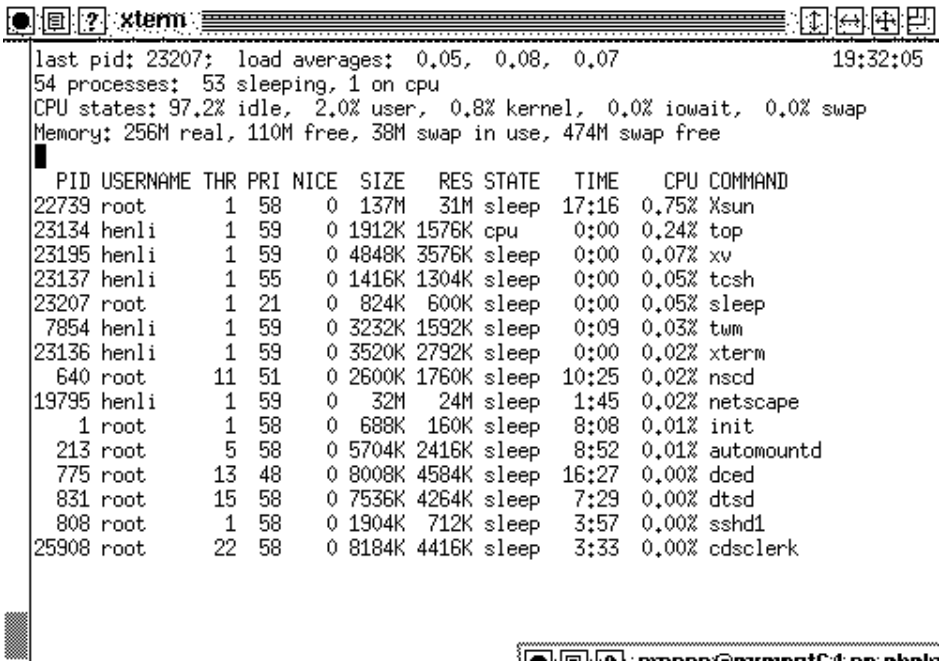
Processbegreppet är av fundamental betydelse i Unix. Alla operativsystem har processbegreppet som en central del i sig men det är bara i Unix i dagsläget som man verkligen kan se hur det fungerar. Ett steg i förståelsen av vad en process är är att lära sig skilja mellan begreppen program och process. Ett program är den exekverbara, körbara koden lagrad i en fil, dvs den kompilerade programkoden (kod som ordnats efter ett visst mönster som datorn kan tolka). En process är när denna kod körs, när ett program övergår från att vara exekverbart till att exekveras, då går vi från begreppet program till process. En process körs i en viss omgivning, speciella inställningar som gäller under körningen för det programmet. Ett program tar upp plats på hårddisken medans en process tar upp plats, minne, swap (tillfälligt utrymme) men framförallt processor-tid (kraft från datorns hjärna). I Unix körs flera program parallellt, det vill säga samtidigt, eller åtminstone ges illusionen av det. Det körs alltså flera processer under en viss tidpunkt.

Titta på processer

Varje process har ett unikt nummer som används för att identifiera den. Detta nummer kallas för processens PID, Process Identification Number. Alla processer har också en förälder, någon process som har skapat dem, och föräldern har också ett PID. För att man skall kunna spåra vilken process som givit upphov till en annan sparar varje enskild process även sin förälders PID (kallad PPID, Parent Process Identification Number). När man arbetar med ett program kan man tycka att man arbetar ensam på datorn. Detta är inte sant. Man samsas med flera olika processer (program som körs). Det är processer som ser till att man får sin epost, ser till att man kan skriva ut och så vidare. De sköter en del grovjobb åt en. Vill man se vilka varelser man delar datorn med kan man använda kommandot `ps` .

```
> ps -A
PID TTY    TIME CMD
    0 ?     0:01 sched
    1 ?     8:08 init
    2 ?     0:00 pageout
    3 ?     66:00 fsflush
...
 7694 ??     0:02 xterm
21771 ?      0:00 xterm
  7866 pts/1     0:00 xclock
...
```

Med ps kan man se vilka processer som körs, hur mycket minne de tar upp med mera, men man får endast en "snapshot". Vill man istället för att få en "snapshot" följa en process under dess arbete kan man använda top. top ger ungefär samma information som ps ger men med beräkningar på vilken process som tar mest resurser (minne, processor-tid) och en del andra godsaker.



Hängda processer

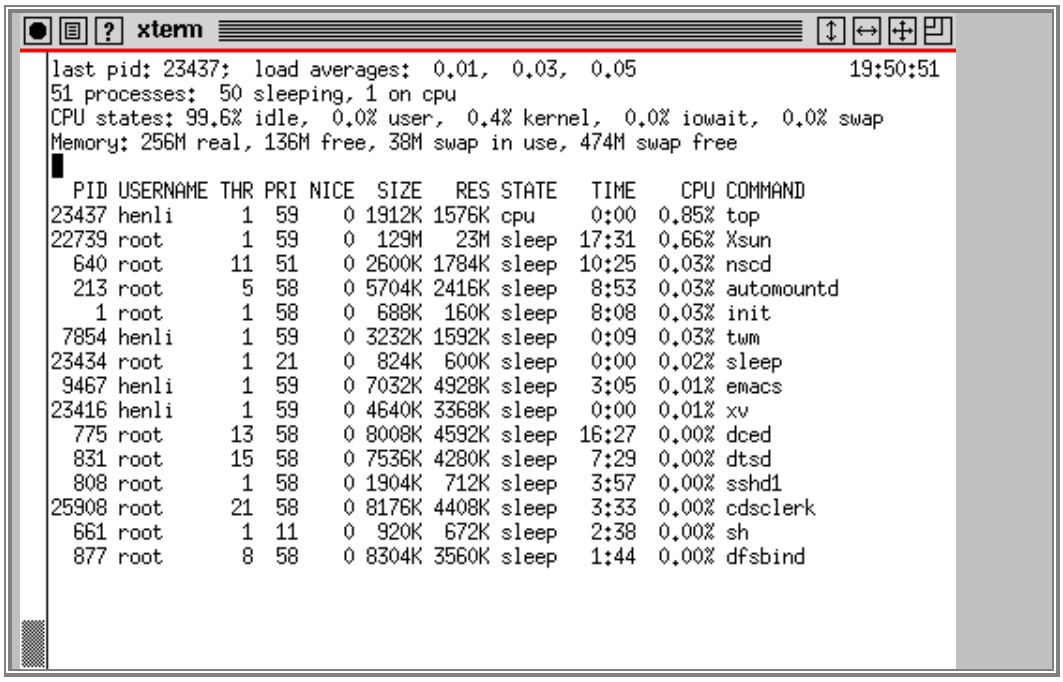
Ibland händer det att en process skenar iväg och tar all processor-tid eller allt minne som finns vilket resulterar i att inga andra processer får möjlighet att köras. Man brukar säga att processen alternativt datorn hängt sig och när detta händer brukar man inte kunna fortsätta sitt arbete. Vanliga program som brukar hänga sig är browsern `netscape` samt matematikprogrammet `matlab`. Att hitta programmet som hängt sig är inte alltid så lätt. En kvalificerad gissning är dock det program man senast startat eller använt. För att döda en hängd process, dvs få den att sluta köra, finns kommandot `kill`.

Vill man döda `netscape` processen med PID 19795 (enligt bild ovan) skriver man

```
> kill 19795
```

Top ger sedan:

```
> top
```



Ibland kan `kill` misslyckas med att döda processen. Då får man ta till våld genom att använda flaggan `-9` med `kill, exempel kill -9 19795`. En del gånger kan man också behöva döda föräldern till processen. Det är då en god idé att kolla processens PPID-nummer innan man slår ihjäl den. PPID-numret kan man få genom att använda `ps` med flaggkombinationen `-ef.`

Som användare kan man endast döda sina egna processer. Skulle man kunna döda andras processer skulle man ju kunna fördärva för andra genom att slå ihjäl deras nyligen skapade och icke sparade dokument. En ganska kul ide egentligen men inget som man får, tyvärr.

Maskinteknik/Automatiseringsteknik

För er som använder detta datorsystem beror kommandot ni skall använda på vilken typ av klient ni sitter vid. Sitter ni vid Linux baserade maskiner gäller `ps -ef`*, sitter ni vid gamla True64 baserade klienter gäller* `ps aux`.

Bakgrundsprocesser

Om man skriver `netscape` i ett `xterm`-fönster så kommer `netscape` att startas. En sidoeffekt av att göra detta är att man kommer inte få tillbaka sin prompt igen (förrän man avslutar `netscape`). Vill man skriva in ett nytt kommando får man starta upp en ny `xterm`, detta blir jobbigt i längden. Det finns därför en möjlighet att lägga en process i bakgrunden, det vill säga starta kommandot/programmet och sedan få tillbaka sin prompt igen så att man kan utnyttja samma `xterm` igen.

```
> netscape &
>
```

Om man glömmer skriva `&`-tecknet sist kan man istället trycka Ctrl-`Z` följt av kommandot `bg` (background) för att uppnå samma effekt. Observera att man inte behöver använda `&`-tecknet när man använder vanliga kommandon som `ls`, `rm`, `cp`. Dessa avslutas automatiskt när de är klara och därmed får man tillbaka sin prompt. När `netscape` avslutas kommer prompten också tillbaka, men det brukar ta betydligt längre tid att surfa än att kopiera lite filer.

Kommentar

Det är återigen shellet som är med i leken genom att det tolkar `&` som att `netscape` skall startas och sedan skall en ny prompt skrivas ut.

Låsa datorn

...bör man annars dör man! Nej, inte riktigt, men att lämna sin dator utan att låsa den är att be om trubbel. Det finns alltid studenter som ser det som sin uppgift att använda ens konto till otrevligheter. Man kan med 100% lugn vara säker på att om man inte låser sin dator kommer någon lustigkurre att använda ens konto till fuffens såsom att skicka kärleksförklaringar till lämpliga studiekamrater, eller varför inte lärare?

Låsa datorn gör man genom

- menyval xlock
- kommandot `xlock`

Efter ett tag startas en skärmsläckare och en klocka räknar ned tiden innan andra kan logga ut en så de kan utnyttja datorn. De får ingen tillgång till ens konto när de loggar ut en på detta sätt. Tiden det tar innan andra kan logga ut en varierar från system till system. *Ni skall och får inte låsa er dator för att gå på lunch!* Man stänger skärmsläckaren genom att trycka på en tangent eller röra på musen. När man gjort detta uppenbarar sig sakta en möjlighet för en att ange sitt lösenord.

Logga ut

Detta är en av de viktigaste sakerna att kunna när man använder en dator som är allmänt tillgänglig! Att endast lära sig att logga in utan att lära sig logga ut vore som att glömma berätta för den kommande bilisten att man skall frikoppla när man bromsar! Egen erfarenhet säger att det blir ett mycket abrupt slut på färden om man glömmer det. Nej, jag har inget körkort ännu.

Logga ut gör man genom

- menyval logout
- Ctrl-`C` i console-fönstret (om ett sådant finns, beror på grafisk miljö)

Skrivarsystemet

I varje sal med arbetsstationer finns det på de flesta sektioner en eller flera skrivare. En skrivare sköter utskrifter för flera datorer. Detta betyder att om flera personer skriver ut samtidigt hamnar man i en kö, en skrivare kan bara hantera en utskrift åt gången. Precis som med processer så har varje utskrift ett unikt nummer så att man kan särskilja en speciell utskrift. Vill man titta på hur kön av utskrifter ser ut kan man använda kommandot `lpq`. Eftersom alla skrivare och datorer är sammankopplade i ett nät kan man med vissa begränsningar skriva ut på en annan skrivare än i den sal man sitter, om skrivaren där är sönder. Varje dator och skrivare har ett unikt namn. En skrivares namn byggs enligt:

- *`skrivarnamn@domän`*
- `labB@mdstud.chalmers.se`

Varje studentdatorsystem på Chalmers är sin egen utskriftsdomän. Det vill säga man måste ha konto på det systemet och sitta vid en dator som tillhör det systemet för att kunna skriva ut. Det är därför onödigt att ange domän på sitt skrivarnamn det räcker med skrivarensnamn utan delen efter @ . Domänerna har tidigare angivits i texten som specificerat system skillnader:

Titta på skrivarkön/vilka skrivare finns

Vill man se en specifik skrivares kö kan man skriva

```
lpq -Pskrivarnamn
```

Observera att flaggan -P används för att man skall kunna specificera vilken skrivare man menar. Utan den skulle den förinställda skrivarens kö visas vilket oftast är den som finns i salen man sitter, med lite tur ☺.

Hur kan man då ta reda på vilka skrivare som finns och var de finns?

Ändra på skrivarkön

Ibland händer det att man råkar skriva ut något av misstag eller att man inte orkar vänta på sin utskrift. Då är det bra att veta hur man plockar bort sig från kön. Har utskriften väl startat (papper matas ut) är det enda sättet att avbryta den genom att först plocka bort den ur kön och sedan tömma skrivarens minne genom att stänga av strömmen till skrivaren. Endast ens egen utskrift påverkas av detta, alla andra utskrifter finns kvar i kön. Om utskriften inte har startat räcker det med att plocka bort den ur kön. För att plocka bort en uskrift ur en kö använder man `lprm`. `lpq`-listningen ger ett jobbnummer för ens utskrift. Detta nummer och `lprm` kan användas för att plocka bort utskriften ur kön.

- `lprm` *`JOBNUMBER`* (om det finns en default-skrivare satt)
- `lprm -PB JOBNUMBER (Obs! -Pskrivarnamn. Om ingen default-skrivare är satt)`

```
piggy> lpr -PB lecture.txt
piggy> lpq -PB
Printer: B@piggy  'HP LaserJet 4050 i labsal B'
 Queue: 2 printable jobs
 Server: pid 27954 active
 Unspooler: pid 27955 active
 Status: opening 'labbps.mdstud.chalmers.se' at 18:04:58, attempt 1 at 18:04:58
 Filter_status: Unknown status: "BEARBETAR UTSKR FRÅN FACK 2" (10023)  ("BEARBET
AR UTSKR FRÅN FACK 2")
 Rank   Owner/ID                  Class Job  Files              Size Time
active  henli@piggy+948              A  948 lecture.txt        14 18:04:58
2       henli@piggy+972              A  972 lecture.txt        14 18:05:12
piggy> lprm -PB 972
Printer B@piggy:
checking 'henli@piggy+948'
  checking perms 'henli@piggy+948'
  not selecting 'henli@piggy+948'
checking 'henli@piggy+972'
  checking perms 'henli@piggy+972'
  dequeued 'henli@piggy+972'
piggy>
```

Att skriva ut

Normalt när man loggar in ställs närmsta skrivare in som default-skrivare. Då behöver man oftast inte ange vart utskriften skall gå. Det finns två sätt att göra utskrifter på.

- Via ett program som t ex ordbehandlingsprogrammet `framemaker`
- Via kommandot `lpr`

Gör man det via program som FrameMaker behöver man oftast bara välja utskrift i menyn så sköter sig allting automatiskt, om det finns en default-skrivare installerad vill säga.

Den andra metoden man kan använda är via `lpr`. För att t ex skriva ut en textfil som heter `sommaren2000.txt`, om en default skrivare är satt, går man tillväga på följande sätt:

```
> lpr sommaren2000.txt
```

Ibland uppstår dock problem som att ingen default-skrivare har satts upp, och då måste man ange en. Detta görs med flaggan `-P` till kommandot `lpr`.

```
> lpr -PlabB sommaren2000.txt
```

Det finns även program som använder sig av `lpr`. Ett sådant är `netscape`. Ibland kan det finnas anledningar till att man vill skriva ut enkelsidigt eller dubbelsidigt då är det bra att veta hur man gör, för att ta reda på det kan man läsa på hemsidorna om datorsystemen för respektive sektion (se ovan).

Skrivare ett språk som heter Postcript. Allt som är Postscript (eller Portable Document Format) går att skicka till en skrivare direkt med kommandot `lpr`. Värre är det med bilder till exempel `jpg` eller `gif`. Dessa måste omvandlas till något skrivaren förstår, vilket man gör genom att skriva ut dem från till exempel ett bildvisar program som `xv`. `xv` kommer omvandla bilden till postscript och skicka den till skrivaren. Gör man inte omvandligen kommer flera hundra sidor text skrivas ut – istället för en bild. Som om man gjorde `cat bild.jpg` istället för `xv  bild.jpg`. `lpr` klara av att själv omvandla text till Postscript därför kan man skicka text utan problem.

Kommentar

Hur vet man då om det finns en default-skrivare satt? Skriver man `echo $PRINTER` i ett xterm-fönster får man reda på det! För att sätta en skrivare, om ingen är satt eller för att man helt enkelt vill ändra, skriver man `setenv PRINTER skrivarnamn@domän`.

Maskinteknik/Automatiseringsteknik

På detta datorsystem måste samtliga dokument göras om till PS-format innan de skrivs ut. Även textdokument. För en beskrivning se: http://mdc.chalmers.se, klicka Helpdesk MDC.

Utskriftsbegränsningar

Till varje konto brukar det finnas en tillhörande begränsning på hur mycket sidor man får skriva ut - utskrifter kostar pengar. För att ta reda på hur många sidor man har kvar används kommandot:

Datavetenskap/Matematik/Bioinformatik

`pquota`

Teknisk Fysik/Gu -Fysik

`printerquota`

Datorteknik/Elektroteknik/IT-linjen

`Pquota`

Maskinteknik/Automatiseringsteknik

`http://mdc.chalmers.se (klicka P-Quota länken)`

Skulle sidorna ta slut finns det möjlighet att skaffa fler. Hur det går till beror på vilken sektion du tillhör och vad det är för konto. *Om man t ex har ett konto på fysik (dd) och ett på etek (tekno) kan man inte använd pquotan för etek för att göra utskrifter på fysik, det är vitt skilda system*.

Datavetenskap/Matematik/Bioinformatik

Personkonto på elevsystemet:

Köp i expeditionen (50 öre per sida) och besök sedan helpdesk Varje termin får man +200 sidor på sitt personkonto.
Labkonto (eller labarea)/Exjobbare:
Gå till handledare/kursansvarig. Kursansvarig bestämmer hur många sidor varje konto skall ha.

Teknisk Fysik/Gu -Fysik

Maila *fop@dd.chalmers.se*. Man får mer printerquota gratis. Men det skall finnas ypperligt goda skäl till varför man behöver den och ber man för ofta får man ingen. Utskrifter man gör skall vara relaterade till studierna. 300 sidor per termin delas det ut per personkonto med automatik.

Datorteknik/Elektroteknik/IT-linjen

700 sidor tillhandahålls vid läsårsstart (sparade sidor från föregående läsåret tas bort). När de är slut kan man genom besök i Helpdesk få 300 sidor extra kostnadsfritt. När man förbrukat dessa och alltså fått totalt 1000 sidor så måste man få godkännande från linjeledningen för att få mer alternativt så kan man köpa för 50 öre per sida i Helpdesken.

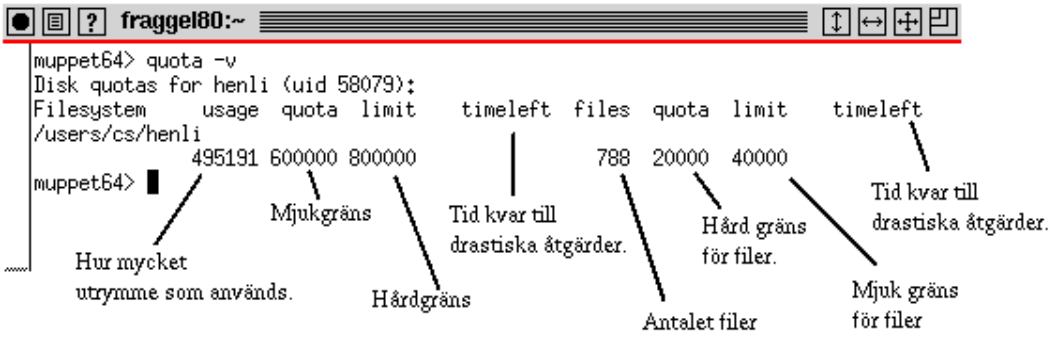
Maskinteknik/Automatiseringsteknik

http://mdc.chalmers.se (klicka P-Quota länken)

Filutrymme

Precis som det finns en begränsning på antalet utskrifter man får göra så finns det en begränsning för hur mycket information man får spara på sitt konto. För att ta reda på hur mycket plats man har att spara på och hur mycket man använt använder man kommandot `quota` med flaggan `-v`. Presentationen av denna information ser lite olika ut beroende på vilket datorsystem man använder.

Datavetenskap/Matematik/Bioinformatik



Teknisk Fysik/Gu -Fysik

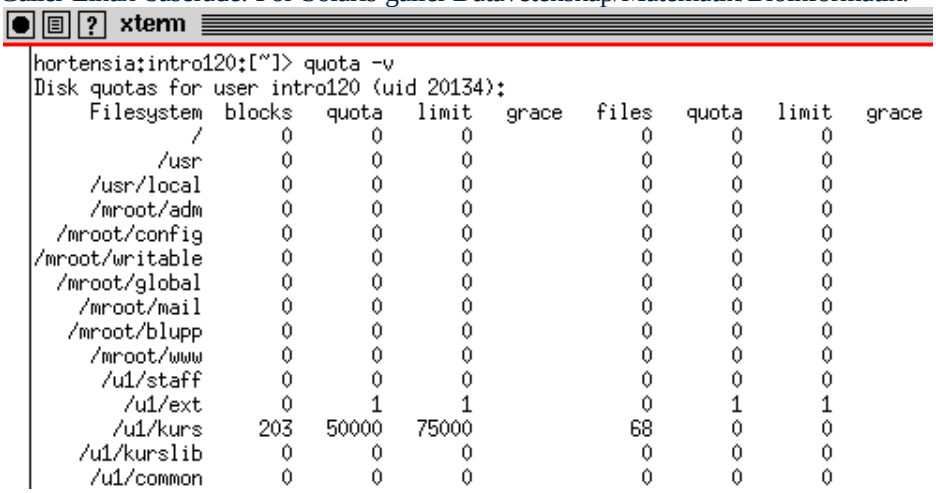
Se Datavetenskap/Matematik/Bioinformatik ovan.

Maskinteknik/Automatiseringsteknik

Se Datavetenskap/Matematik/Bioinformatik ovan.

Datorteknik/Elektroteknik/IT-linjen

Gäller Linux baserade. För Solaris gäller Datavetenskap/Matematik/Bioinformatik.



På etek gäller det att veta vad sökvägen till hembiblioteket börjar med, för att kunna ta reda på vart man skall titta i listan som ges av `quota -v`. Kontot som användes i exemplet för etek var `intro120`. Genom `pwd` ges det att sökvägen till hembiblioteket är `/u1/kurs/intro/intro120`. Alltså skall man leta efter `/u1/kurs/` i listan.

De olika rubrikernas betydelse är:

Rubrik	***Betydelse***

`usage`/`blocks`	Hur mycket diskutrymme man för tillfället ockuperar.
`quota`	Den mjuka gränsen för hur mycket diskutrymme man får ta upp. När man når denna gränsen börjar timeleft ticka ned. Man kan dock fortsätta spara tills man når den hårda gränsen eller timeleft tickat klart.
`limit`	Den hårda gränsen för hur mycket diskutrymme man får ta upp. Denna gränsen kan man inte överskrida, man kan alltså inte spara något när denna gränsen överskridits.
`timeleft`/`grace`	Tid kvar tills drastiska åtgärder tas.
`files`	Antalet filer/bibliotek man har.
`quota`	Den mjuka gränsen för hur många filer/bibliotek man får ha.
`limit`	Den hårda gränsen för hur många filer/bibliotek man får ha. Samma regler som med limit för diskutrymme.
`timeleft`/`grace`	Tid kvar tills drastiska åtgärder tas.

När man övertstigit sin quota tar det ett tag innan drastiska åtgärder tas för att man skall rensa. Tiden man har på innan man drabbas av fasa varierar och anges av timeleft/grace kolumnen. Hur vet man då att man överstigit quotan?

Bioinformatik/Datavetenskap/Matematik

Man får ett mail.

Elektroteknik/Datorteknik/IT-linjen

Ett meddelande poppar upp när man loggar in.

Teknisk Fysik/GU -Fysik

Man får ett meddelande när man loggar in.

Maskinteknik/Automatiseringsteknik

Man får ett mail.

De drastiska åtgärderna som tas när `timeleft` tickat klart varierar från system till system men går alla ut på att få en att rensa bort filer.

Datavetenskap/Matematik/Bioinformatik

Man kan endast logga in via Failsafe-login (väljs via meny i loginrutan. Session→Failsafe). Man kan också maila helpdesken: *helpdesk@medic.chalmers.se*.

Teknisk Fysik/GU-Fysik

Man kan logga in men måste rensa för att kunna spara.
Maila systemansvariga: *fop@dd.chalmers.se*

Elektroteknik/Datorteknik/IT-linjen

Använd Failsafe-login som väljs via meny. Rensa. Logga ut & sedan in igen.
Maila systemansvariga: *helpdesk@medic.chalmers.se*, eller besök dem!

Maskinteknik/Automatiseringsteknik

Använd Failsafe-login som väljs via meny. Rensa. Logga ut & sedan in igen.

Ibland finns det verkligen inga möjligheter att rensa bort något från sitt konto. Då kan man försöka få sin quota-gräns höjd. Detta brukar vara möjligt om det finns mycket bra skäl till det. Ett mycket bra skäl är *inte* att man har mp3-filer och gigantiska privata bilder från fester.

Backup

Backup på filer i ens hembibliotek, mail samt web-filer görs av datorsupporten på de olika sektionerna och instutitionerna med jämna mellanrum. Åtminstone en gång per dygn. Tappar man bort filer får man kontakta supporten (mailadresser se stycket ovan). Det finns ingen ångra funktion efter en `rm`, tyvärr. Likadant om man råkar ta bort en fil innan backupen tagits, vilken oftast tas under natten. Backupade filer arkivera. Hur länge bestäms av datorsupporten, en månad brukar man dock kunna räkna med.

Att byta lösenord

”Men jag har ingen hemlig information på mitt konto!”

En mycket vanlig kommentar från användare som tycker det är jobbigt med lösenord som gör det svårt för andra obehöriga att gissa dem. Så varför är det så viktigt med dessa tillsynes krångliga lösenord dvs lösenord som inte är namnet på husdjuret och som innehåller konstiga tecken dvs annat än a-z och 0-9?

Må så vara att du inte har hemlig information på ditt konto och det är oftast inte orsaken till att någon försöker komma in på kontot heller. Orsaken är snarare att ta sig in för att ta sig vidare. Du har ett enkelt lösenord, någon gissar det och kommer in på ditt konto. Därifrån har de sedan en utmärkt startpunkt för att ta sig djupare in i datorsystemet eller ta sig vidare till andra mål, företag, myndigheter, listan kan göras lång. Eller varför inte använda ditt konto eller din dator till att sprida olaglig information? Vems namn är det som kommer synas på andra sidan i företagets loggar, polisens? Inte är det den elaka personen som tog sig in på ditt konto i alla fall! Det här är också vanligare än man tror.

För att undvika detta scenario och att därmed bli falskt anklagad så finns det ett par enkla steg du kan vidta på egenhand:

1. Byt lösenord då och då. Du bör definitivt göra detta så snart du har ditt konto i handen.
2. När man gör detta skall man tänka efter noga så att man inte byter det till ett lösenord som någon annan kan gissa sig till. Det innebär att man skall undvika att byta det till något som innehåller personrelaterad text som namn! Ett lösenord skall vara 8 tecken långt och får innehålla tecknen `A-Z`, `a-z`, `0-9` samt diverse andra tecken som `(),=`... Man bör dock undvika `å`, `ä` och `ö` samt mellanslag. Ett bra lösenord innehåller en blandning av tecknen ovan.
3. Vara allmänt paranoid. Litar du på datorsystemet du använder? Verkar dina filer vara i sin ordning? Har det körts några kommandon du inte känner igen (pil upp i skalet)?

Effekten av att inte byta lösenord med jämna mellanrum eller av att ha ett för lätt lösenord kan bli att man får ett mail från de systemansvariga om att man skall byta lösenord. Gör man inte detta kan kontot stängas. För att öppna det igen måste man besöka de systemansvariga personligen och få lösenordet bytt. Att få en tid med dem kan vara väldigt krångligt och som student har man laborationer som skall vara inne i tid. Av säkerhetsskäl skickas inte lösenord per epost, brev eller fax. De delas inte heller ut över telefon.

Byta lösenord:

Datavetenskap/Matematik/Bioinformatik/Elektroteknik/Datorteknik/IT-linjen

Personkonton: www.cdks.chalmers.se
Labkonton: UNIX kommandot `yppasswd`

Teknisk Fysik/GU-Fysik

UNIX kommandot: `passwd`

Maskinteknik/Automatiseringsteknik

www.cdks.chalmers.se

Att söka hjälp

Detta avsnitt är troligtvis det viktigaste man som ny student kommer att stöta på under hela kursens gång. Att söka hjälp är inte fult och det är av stor vikt att man vet hur man gör. Dessutom kanske hjälpen gör att man kommer hem tidigare. Den främsta hjälpen som finns i Unix är de beryktade manualbladen. Beryktade är de för att vid en första anblick är de totalt oförståeliga. Det krävs en del träning för att lära sig att läsa dem, men det är väl spenderad tid.

Manualbladen är för Unix vad marinen är för militären - första försvaret!

Men innan vi går in på hur man tyder dessa så finns det ett par andra nyttiga kommandon att känna till, för att hitta rätt i djungeln av dem. Ett kommando/program finns oftast i flera olika versioner med olika flaggor och argument som det kan ta. Man skiljer de olika kommandona/programmen åt genom deras sökväg. En del av kommandona nedan kan därför ge flera svar på en fråga

.

`apropos` *`keyword`*	Söker igenom manualbladen efter manualblad innehållande argumentet *`keyword`*. Som resultat av sökningen visas en lista över de blad som innehöll *`keyword`*.
`locate` *`argument`*	Skriver ut sökvägar till filer i systemet som innehåller argumentet *`argument`* i sitt namn.
`whatis` *`keyword`*	Skriver ut en enrads-förklaring till vad argumentet *`keyword`* gör.
`whereis` *`program`*	Anger sökvägen till argumentet *`program`*.
`where` *`program`*	Anger sökvägen till argumentet *`program`*. I kontrast till `which` så kan where ge flera svar på vart ett program finns. `ls` är ett typiskt kommando som oftast finns i flera versioner på flera ställen.
`which` *`program`*	Anger sökvägen till argument *`program`*. Talar om sökvägen till den versionen som skulle använts om man hade skrivit *`program`* på kommandoraden. Ger endast ett svar i kontrast till `where`.

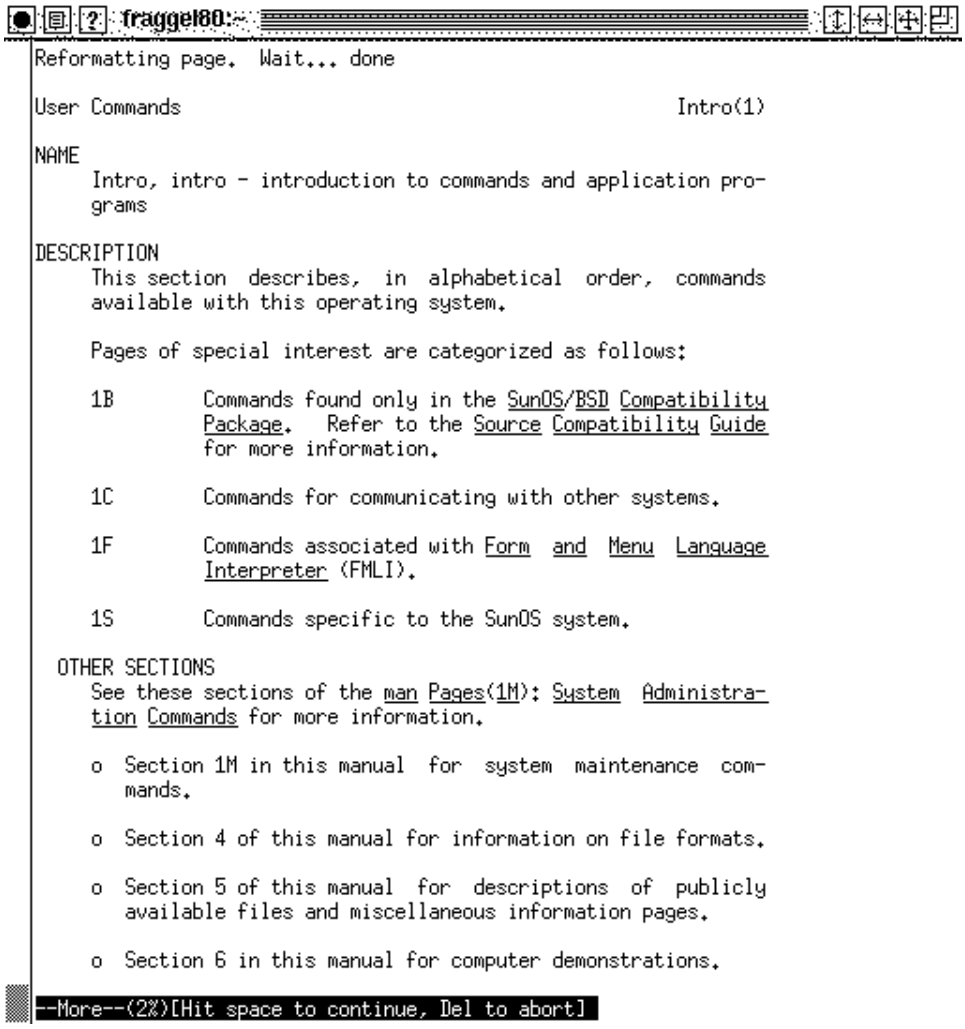
Så till de beryktade manualbladen. Manualbladen är i princip en total beskrivning av Unix vad gäller kommandon, begrepp som filsystem och shell samt hur man programmerar egna kommandon och en hel del annat. Precis som telefonboken är indelad i kapitel så är manualbladen indelade i kapitel:

Kapitel	***Datavetenskap/ Matematik/ Bioinformatik***	***Teknisk Fysik/ Gu-Fysik***	***Elektroteknik/ Datorteknik/ IT-linjen – FreeBSD systemet***	***Elektroteknik/ Datorteknik/ IT-linjen – true64 systemet***
1	User Commands	User Commands	General Commands	Commands
2	System Calls	System Calls	System Calls	System Calls
3	Library Functions	XIL Library	Library Functions	Library Functions
4	File Formats	File Formats	Kernel Interfaces	File Formats
5	Environments etc	Miscellaneous	File Formats	Miscellaneous
6	Games and Demos	Games and Demos	Games	-
7	Special Files	Special Files	Miscellaneous	Special Files
8	-	-	System Managers	System Maintenance Commands
9	Device Driver Information	Device Driver Interface	Kernel Developer	-

Varje kapitel är i sig indelat i underkapitel. Underkapitlen inleds med samma nummer som kapitlet följt av en bokstav tex 1B. Om man vet vilket kapitel man är intresserad av men inte vilket underkapitel så kan man läsa introduktionsbladet för kapitlet. Där står vilka underkapitel som finns.

För att läsa ett manualblad använder man kommandot `man` och ger kommandot (eller vad det nu kan vara som man vill ha hjälp för) som argument till `man`. För att till exempel läsa manualbladet för `man` självt skriver man `man man`. Vill man läsa introduktionsbladet för ett visst kapitel skriver man `man intro.`*`kapitelnummer`*.

```
> man intro.1
```



Varje manualblad är indelat i rubriker. Dessa rubriker är till viss del standardiserade och finns på alla manualsidor men det finns också sidor som har tillskott av rubriker. Beroende på om det är ett kommando eller ett systemanrop så varierar också vilka rubriker som finns med. Här diskuteras endast utifrån antagandet att vi läser manualblad för kommandon.

Rubrik	***Förklaring***
`Name`	Kommandonamn
`Synopsis`	Syntax (möjliga flaggor och argument listas)
`Descriptions`	Vad kommandot gör
`Options`	Vad vilken flagga gör.
`Operands`	Förklaring av argumenten som kommandot kan ta.
`See Also`	Liknande (relaterade) kommandon
`Bugs`	Kända fel som kan uppstå

Här följer ett något omarbetat (censurerat) manualblad för kommandot `cp`. Långtifrån alla manualblad är så pass välskrivna som det för `cp`. `man cp` ger i princip:

```
Reformatting page.  Wait... done

User Commands                                                cp(1)

NAME
     cp - copy files

SYNOPSIS
     /usr/bin/cp [-fip] source_file target_file
     /usr/bin/cp [-fip] source_file... target
     /usr/bin/cp -r|-R [-fip] source_dir... target

     /usr/xpg4/bin/cp [-fip] source_file target_file
     /usr/xpg4/bin/cp [-fip] source_file... target
     /usr/xpg4/bin/cp -r|-R [-fip] source_dir... target

DESCRIPTION
     In the first synopsis form, neither  source_file  nor
     target_file  are directory files, nor can they have the same
     name.  The cp utility will copy the contents of  source_file
     to  the  destination  path  named by  target_file.  If
     target_file exists, cp will overwrite its contents, but  the
     mode  (and  ACL  if applicable), owner, and group associated
     with it are not changed.  The  last  modification  time  of

OPTIONS
     The following options are supported for both /usr/bin/cp and
     /usr/xpg4/bin/cp:

     -i      Interactive.  cp will prompt for confirmation  when-
             ever the copy would overwrite an existing target.  A
             y answer means that the copy  should  proceed.  Any
             other answer prevents cp from overwriting target.

     -r      Recursive.  cp will copy the directory and  all  its
             files,  including any subdirectories and their files
             to target.

OPERANDS
     The following operands are supported:

     source_file    A path name of a regular file to be copied.

     source_dir     A path name of a directory to be copied.

     target_file    A pathname of  an  existing  or  non-existing
                    file,  used for the output when a single file
                    is copied.

EXAMPLES
     1. To copy a file:

          example% cp goodies goodies.old

     2. To copy a list of files to a destination directory:

          example% cp ~/src/*   /tmp
SEE ALSO
     chmod(1),  chown(1),  setfacl(1),  utime(2),  attributes(5),
     environ(5), largefile(5), xpg4(5)

NOTES
     The permission modes of the source file are preserved in the
     copy.
```

Raden efter Reformatting page. ger vilket manualblad vi fått tag på. Det är ett kommando för vanliga användare anges av texten ”User Command”. Det heter cp och tillhör kapitel 1 i manualbladen (alltså (User) Commands) - cp(1). Det är viktigt att man kollar siffran i högerkanten så att det verkligen rör sig om ett kommando (siffra=1). chmod finns både som

kommando och systemanrop (för programmerare). Det som skiljer de två åt är i vilket kapitel de återfinns! Söker man manualbladet för systemanropet chmod genom man chmod kommer det manualblad som först matchar chmod visas, det vill säga man får kommandots manualblad. Man kan istället skriva man chmod.2 för att få manualbladet för systemanropet chmod.

NAME
En enradsbeskrivning av kommandot, samt namn.

SYNOPSIS
Här ser man att det finns två olika versioner av `cp`. En i biblioteket `/usr/bin/` och en i `/usr/xpg4/bin/`. Här ges också en listning av de flaggor/argument som kan användas till varje version. I detta fallet är det samma namn på flaggorna, men olika funktion på vad de gör! Det gäller alltså att veta vilken `cp` man använder och läsa manualbladet extra noga när mer än en version finns. Hur tar man reda på vilken `cp` man använder? Se kommandon ovan.

DESCRIPTION
En beskrivning av vad kommandot gör och vilka regler som gäller vid kopiering. Om man kopierar en fil till ett annat ställe och en fil existerar där med samma namn så kommer den filen skrivas över.

OPTIONS
Beskrivning av flaggor. Först anges det att de flaggor som följer är identiska för båda `cp` versionerna. Sedan beskrivs varje flagga. Flaggan `-i` kan till exempel användas för att få en fråga om man vill fortsätta när man riskerar att kopiera över en fil som redan finns.

OPERANDS
Här beskrivs de argument som kommandot kan ta.

EXAMPLES
Ett par mer eller mindre illustrativa exempel på hur man kan använda `cp`.

SEE ALSO
Andra relaterade manualblad som kan vara av intresse. I detta fallet kommadon (de som har 1 i kanten) som har med filoperationer att göra.

NOTES
Viktigt-notiser. Här berättar man att när man kopierar en fil så kommer rättigheterna (se kapitlet om rättigheter) att vara samma på kopian som på orginalet.

Tycker man att det jobbigt att använda det textbaserade systemet med `man` för att läsa manualbladen finns det två alternativ som är mer grafiska (pek o klick); `xman`.

Naturligtvis finns det fler sätt att hitta information på. **W**orld **W**ide **W**eb (www) är ett utmärkt sådant sätt. En sak som inte är så utmärkt är att skicka sina frågor till systemadministratörerna. Dessa personer har oftast mycket att göra som det är. Skickar man en fråga till dem skall man vara säker på att man har gjort allt nedan innan man skickar den! Svaret från administratörerarna kan också dröja allt från omedelbums till flera veckor och ibland månader, det beror på vilken prioritet de sätter på frågan. Viktigast är ju att systemet fungerar så att alla kan använda det. I andra hand kommer att alla skall ha ett fungerande konto att arbeta i.

Sist kommer att ge användarna hjälp med frågor som de själva borde kunna ta reda på, typ "hur listar jag skrivarkön?"

1. Fråga kompis.
2. Använd manualsidorna i Unix
3. Programmets egna hjälp
4. Datorsystemets hemsidor (här återfinns beskrivningar om det mesta man behöver veta om datorsystemen inklusive hur man kontaktar de systemansvariga, öppettider för helpdesk och liknande.)
5. Läs news
6. Fråga en handledare/föreläsare
7. Fråga helpdesk om en sådan finns
8. Fråga driftavdelningen – systemansvariga

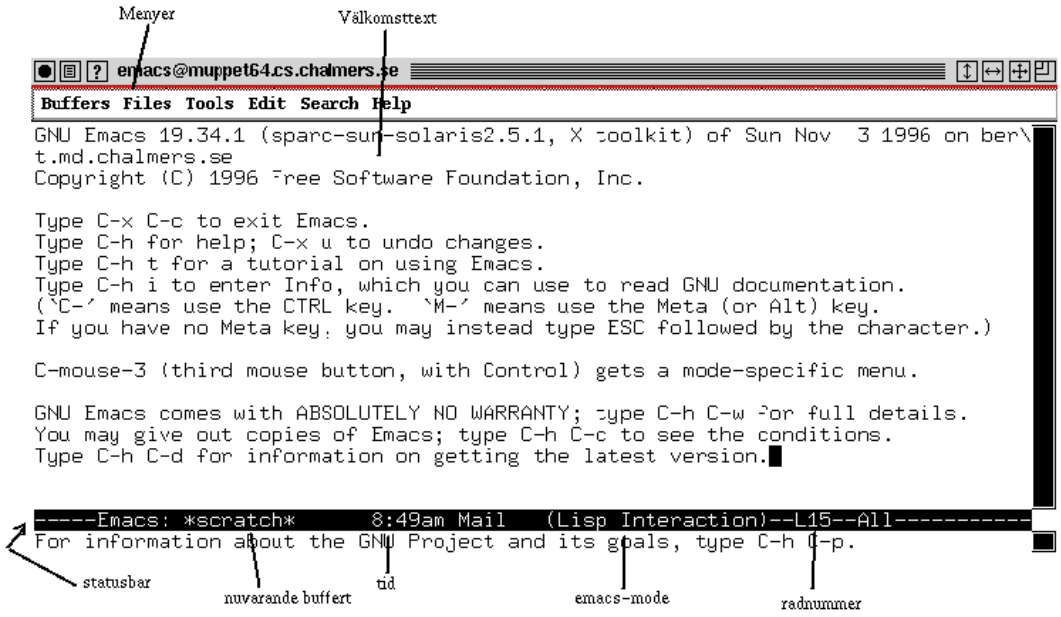
Om man mailar till de systemansvariga bör man tänka på följande:

- Var kortfattad och koncis, tänk också på att de systemansvariga inte är tankeläsare. Faktiskt.
- Ange användarnamn om ni mailar från ett annat konto än det det berör och ange varför ni mailar från det kontot.
- Klipp gärna in utdatan från följande kommandon i det ni skickar, underlättar felsökandet: `date, uname –a, whoami, hostname` enstaka fall `ps –ef` också.
- Tänk på att ett svar kan dröja från systemansvariga då de både underhåller system, utvecklar och hjälper användare. Har ni mailat och inte fått ett retur brev med ”unknown adress” eller liknande så har ert mail nått ansvariga, det finns då ingen poäng i att maila igen. Jo, kanske om det går 2 månader utan svar.

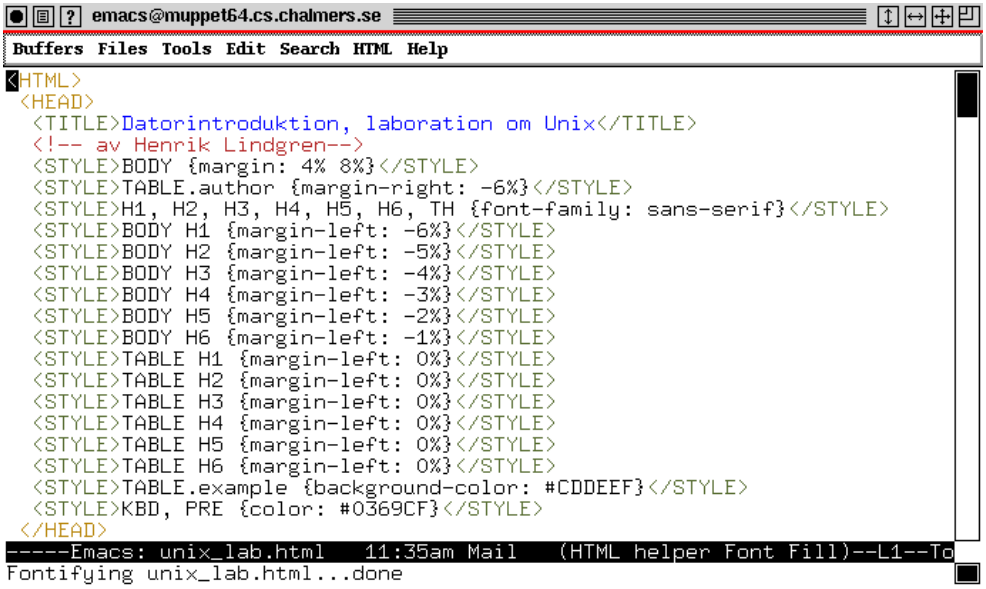
Emacs

Emacs är egentligen byggt för att vara en editor (textredigerare) men på grund av sin flexibla struktur kan man läsa epost, spela spel och mycket, mycket mera med `emacs`. Här kommer fokus att ligga mot textredigering. Det finns två versioner av `emacs`; `emacs` respektive `xemacs` Den sistnämnda är något mera grafiskt (mer pek och klick) orienterad än den första. Det som beskrivs nedan gäller för `emacs`.

Det första man möts av när man startar `emacs` är en välkomsttext. Emacs kan startas via meny eller via ett xterm fönster som man helt enkelt skriver emacs i!



Välkomsttexten ger en snabb beskrivning av hur man avslutar emacs och söker hjälp. `Type C-x C-c to exit emacs` betyder att man för att avsluta `emacs` skall trycka Control-`x` Control-`c`. Statusbaren visar de inställningar som man har under sitt arbetspass, vad den aktiva bufferten heter (förklaras nedan), vad klockan är, vilken emacs-mode som används samt vilket radnummer markören (den svarta fyrkanten) är placerad på. Under statusraden finns en minibuffert. Den används av emacs för att tillhandahålla/skaffa information till/från användaren. Vill man öppna en fil så kommer man att få ange sökväg och namn på filen där. Ett så kallat emacs-mode är ett speciellt tillstånd som emacs befinner sig i. Det finns till exempel ett html-mode. Öppnar man emacs med en html-fil som argument, t ex `emacs hemsida.html`, startar emacs automatiskt i html-mode vilket innebära att man får extra menyer för html-relaterade saker. Man kan också få färg på texten så man enkelt kan urskilja html-formateringskommandon från den vanliga texten. Det är den största fördelen med emacs.



Som med en vanlig editor så kan man öppna, stänga, spara och bläddra bland öppna dokument. För att sköta denna hantering finns det två tillvägagångssätt, via menyer eller kortkommandon. Ett kortkommando är en tangentkombination man trycker ned. Det finns inte menyer för allt som går att göra utan endast de vanligaste valen täcks av menyerna. Däremot finns det kortkommandon för allt. I en del menyval finns också kortkommandot angett i kanten. Kortkommandot för menyvalet Files->Open File är `C-x C-f` vilket är en förkortning för Control-`x` Control-`f` som betyder att man skall trycka Control och `x` sedan Control och `f` för att utföra önskad operation. Man bör satsa på att lära sig de vanligaste kortkommandona då det resulterar i en mycket mer effektiv anvädning av emacs.

Varje dokument man öppnar i emacs hamnar i en så kallad buffert. En buffert är ett temporärt lagringsutrymme. Gör man en ändring av dokumentet i bufferten och avslutar utan att spara så kommer inte ändringen att synas nästa gång man öppnar samma dokument i emacs. Alla buffertar har namn i emacs. Nuvarande buffertsnamn kan man få via statusbaren (se bild). Vill man byta buffert kan man göra detta via menyvalet Buffers genom att välja en ur listan där.

När man öppnar ett dokument kommer texten att hamna i det område där introduktionstexten är i bilden ovan. För att röra sig i texten kan man använda antingen musen eller tangentbordet. Enklaste sättet att kopiera/flytta text är med musen. För att markera text tryck ned vänster musknapp, klippa ut den tryck snabbt två gånger på högermusknapp. För att klistra in den tryck mittenknappen på musen. Tangenterna man använder för att röra sig i texten är:

Tangent	***Funktion***
Pil upp	Gå en rad uppåt i texten
Pil ner	Gå en rad neråt i texten
Page Up	En sida uppåt i texten
Page Down	En sida neråt i texten

Home	Gå till början av texten
End	Gå till slutet av texten
Del	Sudda tecknet till *vänster* om där jag står nu. Så som backspace normalt gör!
Insert	Skriv över tecknen till höger från där jag står med det jag skriver nu.
M-x goto-line	Hoppa till radnummer

Emacs sätt att beskriva kortkommandon är lite annorlunda från det som använts här. I emacs skrivs Control som ett C endast. Ibland stöter man också på M vilket står för Meta-tangenten. Meta ser ut som en snedvänd fyrkant på tangentbordet. Ibland finns ingen sådan tangent på tangentbordet och då kan man använda Esc-tangenten istället. När det gäller M-x goto-line skall man trycka Meta och x och sedan i minibufferten (under statusraden) skriva goto-line. De kortkommandon som det kan vara en bra ide att lära sig redan nu är:

Funktion	***Menyval***	***Kortkommando***
Öppna en fil	Files » Open File	Control-x Control-f
Spara en fil som	Files » Save buffer as	Control-x Control-w
Snabbspara	Files » Save Buffer	Control-x Control-s
Ångra	Edit » Undo	Control-Shift--
Sök och byt ut	Search » Query Replace	Meta-Shift-5
Avsluta (påbörjat kommando)	-	Control-g

När det gäller att söka efter ett ord är det viktigt att man placerar markören i början av texten eftersom sökning sker från och med där markören står.

Ibland kan det hända att emacs delar upp en buffert i två delar så att man helt plötsligt har två buffertar på skärmen. Vad emacs gjort då är att skapa en frame, delning av skärmen. För att återgå till det normala tillståndet med en buffert kan man välja Files » One Window.

Emacs hjälp

I emacs hjälp-meny kan man hitta allehanda hjälpmedel som FAQ (frequently asked questions), Tutorial (lektion i emacs), beskrivningar av kortkommandon med mera.

Nyttiga länkar

- www.dd.chalmers.se - Teknisk Fysiks datorsupportsidor
- www.medic.chalmers.se - Elektro/Dtek/Ma/Dv datorsupportsidor.

- www.mdc.chalmers.se – Maskin/Z