Övergripande praktisk information

Denna kurs ges inom dataingenjörsutbildningen vid KTH – STH vårterminen 2017. Allt material som behövs för att klara kursen ska finnas på KTH-Social. Dit måste ni gå.

Systemkrav för denna kurs: *50GB hårddiskutrymme samt installerad VirtualBox eller annan virtualiseringsprogramvara. Det är rekommenderat att Guest Additions (eller liknande) används men det är inte obligatoriskt.* ***Kurslitteratur:*** *Gratis på kursens webssida.*

Schema och tentamensschema: *se www.kth.se/schema*

Examination: *En skriftlig tentamen (*`TEN1`*), en datortenta och inlämningsuppgift (*`LAB1`*).*

Tentamen: *Enligt tentamensschema.* ***Planering:*** *Se* `Planering.pdf` *på KTH-Social.*

Genomförande: *Kursen har ett antal bokade tillfällen med tillhörande övningar. Kursen har mer av alternativa metoder för undervisning, bland annat så kallad Peer Instruction.*

Laborationer och övningar: *Detta är det viktigaste i kursen, om man genomför laborationerna självständigt (men gärna i dialog med kurskamrater) så är det ganska säkert att man klarar datortentan, förutsatt att man har teorin bakom laborationen hyfsat klar för sig, så satsa friskt på laborationerna. I nästa kurs "Datortekniskt projekt" är det dessutom nödvändigt att kunna mycket från operativsystemskursen, laborationerna i OS-kursen förbereder er för båda tentorna men även för genomförandet av nästa kurs. Dessutom fördjupar OS-kursen grundkursen i programmering så om ni inte klarat grundkursen i programmering ännu så kommer OS-kursen att hjälpa er mot att också fullborda grundkursen i programmering. Så, som sagt, satsa friskt på laborationerna!*

Installation och underhåll av operativsystem, inlämningsuppgiften samt lite TCP/IP

Vi kommer nu in i första delen av kursen. Man kan givetvis förbereda sig inför denna föreläsning också genom att läsa igenom materialet innan, men vi kommer att vänta med *Peer Instruction* till alla har hunnit arbeta med materialet närmare.

Självstudiematerial: Det finns två självstudiematerial som ni behöver arbeta med under denna vecka. Vi kommer att gå igenom det materialet mycket översiktligt idag men efter det är det tänkt att ni arbetar med det självständigt. **[Genomgång av självstudiematerialet.]**

Installation av operativsystem

Ett OS måste givetvis installeras innan man kan köra det. Själva installationsprocessen är gansk lärorik, tidigare om åren har vi arbetat med *Gentoo Linux* eller *Arch Linux* men i år lättar vi lite och arbetar istället endast med *Debian*. Vi ska hela tiden arbeta virtuellt så ni får skapa ytterligare en virtuell maskin (förutom *NewTinyDebian*) som ni arbetar med. Ni kanske även vill skapa flera virtuella maskiner. Vi ska även använda *NewTinyDebian* och göra en omorganisation av filsystemen för att träna på förståelsen av en oerhört central del av ett operativsystem: hur man hanterar de sekundära lagringmedierna alltså oftast helt enkelt hårddiskarna. Primärminnet är ju datorns minne som kör när datorn är på och som töms vid avstängning.

Installationen av *Debian* finns beskriven på `www.debian.org`. Efter installationsprocessen ska vi arbetar med *SystemRescueCD* för att omorganisera ett filsystem.

En installation av ett operativsystem har normalt två skeden. I det första skedet sker grundläggande aktiviteter som behövs för att överhuvudtaget kunna köra systemet på datorn. I det andra skedet körs systemet igång på datorn men systemet är inte ännu helt körbart, vissa kompletterande och avslutande installationer och konfigurationer måste utföras först. Vi kan skissera det så här:

Skede 1. *Grundläggande installation.* Systemet kör inte än, men datorn måste ju köra för att överhuvudtaget kunna göra någonting, då kör datorn på en installations-CD eller liknande.

- Partitionering och formatering: Detta betyder ofta att man bestämmer hur en hårddisk ska vara organiserad som ska innehålla systemet.
- Kopiering/hämtning: Man laddar ner från nätet eller kopierar från CD eller DVD det man behöver för att systemet ska kunna köra igång.
- Installation/kompilering: Man installerar det som behövs för att systemet ska kunna köra igång på egen hand, inte bara köras från ett installationsmedium.
- Konfigurering: De sista inställningarna görs för att få systemet att kunna köra på datorn.

Skede 2. *Första start.* Systemet kör nu, men behöver vissa kompletterande och avslutande installationer och inställningar. I detta skede kommer vi att koppla bort installationsmediet (alltså CD-skivan) och systemet kör nu själv och man kan med hjälp av systemet själv fullborda installationen och den kommande administrationen.

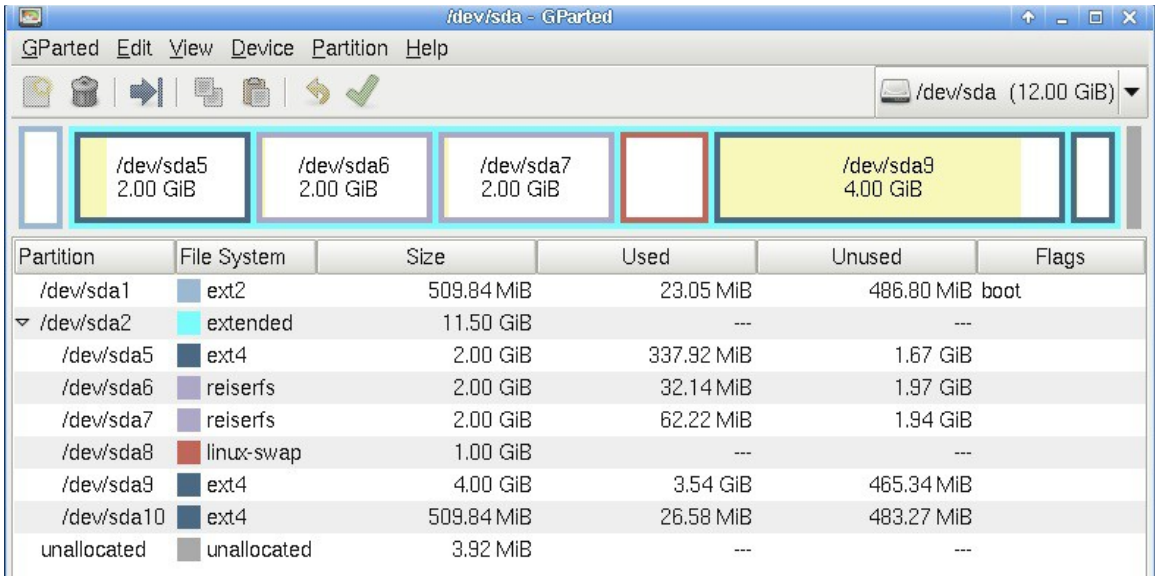
Skede 3. *Drift.* Då systemet är installerat återstår nu den dagliga driften av systemet.

Vi kommer inte att beskriva alla dessa delsteg i detalj, de finns att läsa om i på `www.debian.org` och det är mycket viktig läsning. Ni måste läsa dessa instruktioner i detalj och följa dessa instruktioner nästan precis som de är givna, men ni måste också förstå vad de innebär eftersom ni ska anpassa installationsprocessen efter våra behov. Eftersom vi kör virtuellt kan ni frysa installationsprocessen i något delsteg och fortsätta vid en senare tidpunkt, man kan också ta så kallade "Snapshots" som innebär att man slipper göra om vissa saker om vissa installationsmanövrar leder till ett fel. Då kan man backa till en tidigare snapshot. Använd gärna denna funktionalitet.

I det här läget är det alltså dags att ta fram *VirtualBox*, den dokumentation och de programvaror som behövs och sätta igång med installationen av det system du valt. Det kommer att involvera många många nya begrepp och det är rekommenderat att använda *Google* och *Wikipedia* under installationsproceduren för att **förstå i detalj vad det är för manövrar som ni utför**. Det kommer troligen att ta lång tid men vara mycket lärorikt.

Vi ska nu ge en övergripande beskrivning av hur det första punkten ovan "Partitionering och formatering". Den skulle faktiskt kunna göras utanför installationsmediet för *Debian* med verktyget *SystemRescueCD*. Och faktiskt beskriver nedanstående hur ni kan komma igång med en senare del av laboration 1 som är just en omorganisation av *NewTinyDebian* som är en av maskinerna ni ska arbeta med.

Vi ska studera screenshots hämtat från en beskrivning av en installation av ett annat system, men den teori som beskrivs gäller också installationen av *Debian*. *SystemRescueCD* innehåller *gPartEd* som gör det möjligt för oss att skapa partitioner och formatera som vi önskar. Den skiss som anges på nästa sida är gammal, vi uppdaterar informationen lättast genom att ange följande specifikationer: I skissen på nästa sida anges storlek 2.0 GB på sda6 och sda7, vi väljer istället storlekarna **2.5 GB** på dessa två partitioner. I skissen på nästa sida anges storleken 4.0 GB på partition sda9, men den ska bort helt och hållet och istället ska sda5 utökas med **5.0 GB**. De partitioner och filsystem som vi ska ha för vår virtuella maskin som ska innehålla vårt system ska se ut så här:



Partition	File System	Size	Used	Unused	Flags
/dev/sda1	ext2	509.84 MiB	23.05 MiB	486.80 MiB	boot
/dev/sda2	extended	11.50 GiB	---	---	
/dev/sda5	ext4	2.00 GiB	337.92 MiB	1.67 GiB	
/dev/sda6	reiserfs	2.00 GiB	32.14 MiB	1.97 GiB	
/dev/sda7	reiserfs	2.00 GiB	62.22 MiB	1.94 GiB	
/dev/sda8	linux-swap	1.00 GiB	---	---	
/dev/sda9	ext4	4.00 GiB	3.54 GiB	465.34 MiB	
/dev/sda10	ext4	509.84 MiB	26.58 MiB	483.27 MiB	
unallocated	unallocated	3.92 MiB	---	---	

Skissen ovan är en screenshot från ett gammalt system men det ska alltså uppdateras enligt instruktionerna ovan. Tidigare var det 10 partioner, nu ska vi alltså ha 9. När ni börjar och skapa denna uppsättning partitioner kommer givetvis mycket mer utrymme vara ledigt från början. Ni får gärna göra partionerna större om ni vill, detta representerar ett slags minimum, men då kanske ni kommer över gränsen 30GB som är det minsta som ni behöver använda av ert eget diskutrymme för att klara kursens laborationer. (Ni kanske till exempel vill att `sda9` ska vara större. Valet är ert.) Observera att verktyget också formaterar dessa partitioner, att formatera en partition betyder att införa ett så kallat filsystem på den det är först då som man kan börja lagra data på partitionen.

För att komma åt innehållet på dessa partioner behöver man också använda kommandot `mount` samt `swapon` i installationsproceduren för att åstadkomma en så kallad montering av dessa partitioner på delar i katalogstrukturen. Den övre schemat anger 10 partitioner som användes för installation av ett system som heter *Gentoo Linux*, vi ska som sagt använda *Debian*, men vid den tidigare installationen användes mount som gjorde att själva filhierariken kunde monteras ihop. Monteringen var enligt följande:

`/dev/sda1` på `/boot/`
`/dev/sda5` på `/`
`/dev/sda6` på `/tmp/`
`/dev/sda7` på `/var/`
`/dev/sda8` används som *swap* *(egentligen inte en montering, men vi anger den här ändå.)*
`/dev/sda9` på `/home/`
Lägg märke till att vi har olika filsystem på de olika partitionerna.

Vi kommer även använda *SystemRescueCD* då vi omorganiserar innehållet i *NewTinyDebian*, men detta specificeras i övningen. Då kommer vi att se liknande vyer som den ovan.

IP – Internet Protocol – Internetprotokollet

Vi ska ge en kort introduktion till IP, alltså det som *Internet* bygger på. Det kommer att hjälpa oss att klara operativsystemskursen.

Protokoll i allmänhet

Inom datortekniken betyder ordet "protokoll" ett förbestämt sätt att bete sig på. För att två olika enheter ska kunna kommunicera måste de följa samma protokoll. På sätt och vis kan man säga att

ett mänskligt språk är ett exempel på ett protokoll, jag (en enhet) talar nu svenska och då kan ni (de andra enheterna) förstå vad jag säger eftersom ni förstår svenska. Men vi kan också byta språk, byta protokoll, till engelska, om jag talar engelska så förstår ni också vad jag menar. You understand what I mean even if we shift language.

IP

De flesta datorer idag har någon form av nätverksanslutning och de kan kommuicera med varandra med hjälp av denna anslutning. *IP* är en förkortning för *Internet Protocol* och beskriver det protkoll som enheter på Internet använder för kommunikation. I korthet fungerar det så här: Varje enhet på nätet (en dator, skrivare eller router etc.) har en så kallad IP-adress. Klassiskt sett består en IP-adress av fyra bytes, alltså tal mellan 0 och 255, noterade med hjälp av punkter. Ett exempel är 130.236.45.81.

En författare av den tidigare kurslitteraturen skrev att "varje dator har en unik adress", men det är *inte datorn i sig* som har en IP-adress, det är varje *nätverkskort* som kan ha en (eller flera) adresser. Eftersom det ofta är så att varje dator har precis ett nätverkskort så stämmer det ofta att man kan ordna en IP-adress till varje dator, men den precisa formuleringen är att *ett nätverkskort har en adress*. Ett nätverkskort kan även ha fler adresser, men det blir en specialiserad situation. (Sker vid virtualisering.) Värt att notera är också att det finns två sorters adresser, en hårdvarumässig adress som (i princip) aldrig ändras, och en mjukvarumässig som kan ändras mellan varje gång man startar nätverkskortet.

Ett nätverkskort är anslutet till ett nätverk via en kabel eller så sker kommunikationen trådlöst. Den grundläggande funktionen är dock att nätverkskortet hela tiden lyssnar efter trafik med sin egen adress i, om det hör att en försändelse finns på nätet med sin adress på så tar det upp den informationen och skickar in till operativsystemet för vidare behandling. Det är detta som hela tiden sker när vi till exempel kör en webbläsare. (Alltså surfar på nätet.)

IP-protokollet bygger alltså på de så kallade IP-adresserna. För att kunna kommunicera över nätet måste alltså en dator (eller mer precist: *ett nätverkskort*) få en IP-adress. Man kan sätta en dators (ett nätverkskorts!) IP-adress manuellt men det är mycket mer vanligt att man låter datorn vid uppstart lyssna på nätet efter en så kallad DHCP-server, en DHCP-server kommunicerar också över IP men med hjälp av hårdvaruadresser (som finns i varje nätverkskort). På så sätt kan man vid en förhandling låta en server bestämma vilken IP-adress man ska ha. Det här är egentligen allt vi behöver veta just nu. Det är rekommenderat att läsa på *Wikipedia* om DHCP, där står:

"Dynamic Host Configuration Protocol automates network-parameter assignment to network devices from one or more DHCP servers. "

Texten "network-parameter assignment" syftar då bl.a. på IP-adress som tilldelas (*assign*=tilldela) klienten, alltså den som frågar efter IP-adress. Men mer än bara IP-adress tilldelas, det finns flera parametrar som *Nätmask*, *Broadcast*, *Gateway* osv. Men dessa saker täcks i en kurs i datorkommunikation. Vi lämnar detta ämne här genom att förenklat konstatera att "En DHCP-server ger oss en IP-adress med mera."

Olika applikationer som använder IP

När man har en dator (nätverkskort) ansluten till ett nätverk där andra också är anslutna och när man har fått en IP-adress kan man börja köra applikationer (alltså användarprogram) som kommunicerar över nätet. Den mest populära är förstås *webbläsaren*, men det finns ett par andra som är viktiga att känna till:

`ping` är ett kommando som testar om en viss adress är tillgänglig. Man kan skriva

```
ping www.google.com
```

för att se om man har kontakt med *Google*. Det brukar vara ett bra sätt att pröva om man har kontakt med Internet. Har man inte kontakt med *Google* så är man troligtvis inte på nätet, något fel har då uppstått vid tilldelning av IP-adress eller liknande.

`ssh` är ytterligare ett kommando och det står för "*Secure Shell*". Med hjälp av detta kan man göra en inloggning och starta ett kommandoskal på en dator över nätet. Vi kommer troligen inte att göra det så mycket, det är inte så mycket att säga om det, det kan vara mycket användbart om man behöver styra en dator från en annan plats än datorn står på. Typisk kan man vilja anluta sig via `ssh` till en dator som kör en webbserver eller dylikt. Intressant att nämna här är att laboration 1 faktiskt är ett slags shell, vi ska, som ett delsteg i laboration 2, utvidga laboration 1 så att den kan köra över IP mellan två virtuella maskiner.

`telnet` är en tidigare version av ett fjärrinloggningsprogram (som `ssh`) men det har ingen säkerhet, *Telnet* krypterar inte lösenord så när man loggar in med Telnet syns ens lösenord över nätet om någon lyssnar då. Det finns fler säkerhetsproblem med Telnet som SSH försöker lösa. Läs gärna *Wikipedia*s artiklar om *Telnet* och *SSH*.

Det finns en uppsjö av applikationer som använder sig av IP, det här var bara några stycken. Vi lämnar dock listan på exempel över IP-applikationer nu.

IP-aspekter av installationsproceduren

Det ni ska åstadkomma med installationen av Debian är att datorerna ska kunna kommunicera via ett lokalt nätverk. Detta lokala nätverk presenteras för de virtuella maskinerna av VirtualBox självt. I laboration 1 (som är det som ni ska arbeta med nu) är det alltså ett mål att de olika debianmaskinerna ska kunna pinga varandra, dvs man ska, från en maskin som har IP-adress `192.168.0.2`, kunna skriva `ping 192.168.0.1` och en positiv respons.

Vidare är det viktigt att ni väljer att installera en DHCP-klient i slutet av installationsproceduren. Det står "*optional*" (valfritt) i handboken, men det är *inte* valfritt för oss, vi *måste* ha en DHCP-klient installerad. *VirtualBox* själv kommer att fungera som DHCP-server och kommer att dela ut IP-adresser till de virtuella datorer (nätverkskort) som vi skapar.

Vi återvänder till en diskussion av installation av operativsystem. Det finns olika typer av installationssammanhang: 1. "Omblåsning" – installation av operativsystem genom avbildningskopiering – **inte övervakad**

I en stor organisation där många ska använda datorer av samma slag, som till exempel här på skolan, kan man inte sitta vid varje dator och göra enskilda installationer. Det man gör då är en installationsmall på en dator, sedan använder man verktyg för att kopiera den installationen till alla andra maskiner i alla salar. Systemgruppen gör detta och vi kan ibland se att salarna ominstalleras, då kör alla datorer i hela salen ett diskkopieringsprogram som helt enkelt bara skriver över innehållet på hela hårddisken. Populärt kallas det att man "blåser om" salen. Det finns verktyg för detta vi kommer att se ett sådant verktyg, men det finns inget planerat moment att använda det verktyget. Ni kan dock använda det om ni vill, det heter *PartImage* och finns på *SystemRescueCD*.

2. Övergripande abstrakt beskrivning av installationsproceduren då man övervakar den

"Omblåsning" innebär att man inte övervakar proceduren installationen sker från en redan gjord

mall, en avbild, så jobbet är gjort på någon annan dator. Nu ska vi se hur man skapar den här mallen som givetvis också kan användas i dagligt arbete.

3. Installationsmedium

För att kunna installera ett operativsystem måste man ha något slags installationsmedium. Flera saker behöver tillgängliggöras/möjliggöras genom detta medium:

3.1. Miljö som är körbar

Då ett OS installeras på en dator så måste datorn **köra**, det är ett självklart krav. För att tillgodose det kravet innebär det att själva installationsmediet måste innehålla ett mindre operativsystem för att datorn ska kunna köra. Installationsmediet är ofta en CD-skiva som man kan starta på, en så kallad boot-bar CD, och denna CD innehåller då ett mindre system som kör igång datorn. Detta system är då ofta mindre och även om det faktiskt ofta är en mindre variant av det system man installerar så innehåller det inte alla möjligheter som det fullständiga systemet innehåller. Detta mindre system är anpassat till installations- och serviceuppgifter. Exempel på sådana installationsmedier är *SystemRescueCD*, men också förstås en liknande installations-CD för *Debian*.

3.2. Tillgång till installationsfiler

Installation av ett operativsystem innebär i princip väldigt mycket kopiering av filer in på rätt plats. Installationsmediet måste ge den körbara miljön tillgång till de filer som ska kopieras in på rätta ställen. Installationsfilerna kan ligga på mediet men det är också mycket vanligt (och bra!) om mediet tillhandahåller kontakt med en server från vilket man kan ladda ner de senaste versionerna av installationsfilerna. När vi installerar ett operativsystem kommer vi att arbeta tätt tillsammans med flera servrar som kan tillhandahålla de senaste versionerna av installationsfilerna.

3.3. Konfigurationsmiljö

Den körbara miljön behöver utföra inställningar av systemet för att anpassa mjukvaran mot den specifika dator man kör på. Det betyder varierande krav på användaren, i *Windows XP* behöver man i princip bara svara "Ja, Ja, Jadå, OK" hela tiden och det är inte svårt alls, allt är automatiserat. *Debian* är ganska lätt också. I *Arch Linux* blir det lite mer avancerat och i *Gentoo* är installationsproceduren en riktig prövning. Miljön man arbetar i för installation av ett OS ska alltså innehålla konfigurationsmöjligheter och det tillhandahålls i *UNIX*/*GNU*-*Linux*-världen genom att det installationsmediet helt enkelt är ett minisystem där man kan utföra konfigurationerna antingen grafiskt och/eller vid kommandoprompten. I *Windows*-världen kan man ofta inte göra annat än att sitta och titta på när de fördefinierade installationsprocedurerna genomförs. (Det är då man säger "Ja, Ja, Jadå, OK".) Det enda man egentligen måste vara noggrann med är att välja rätt tangentbord. Allt annat kan ordnas när systemet är installerat om man råkat göra något fel. Välj dock DHCP-konfiguration då ni installerar nätet.

4. Varför ska vi arbeta virtuellt?

Det finns väldigt många väldigt starka skäl att arbeta virtuellt:

4.1. Säkerhet

Att installera ett operativsystem är en riskfylld procedur, om man gör fel så kan man förstöra viktig data. Om vi skulle installera operativsystem direkt på era bärbara datorer så skulle vi riskera de data som ni redan har där. Vi ska installera operativsystem som en träningsprocedur i en utbildning, vi vill inte påverka de verktyg ni har för att genomföra hela utbildningen, enkelt uttryckt, datorn ni har behöver ni för parallellkursen också och ni har väl troligen privata filer på era datorer också. Datorerna är ju era egna. Då OS-installation går fel så blir följderna ofta allvarliga; att man inte kan

starta datorn, att hela innehåller på hårddisken raderas exemepelvis.) Med virtualisering så simuleras allt det här och en krasch blir då alltså inte så allvarlig.

4.2. Man kan pausa

VirtualBox som är den plattform som vi rekommenderar, har möjligheten att frysa ett system i ett läge som man sedan kan gå tillbaka till om man skulle göra ett fel längre fram, man kan ta en "snapshot" som det heter. Det innebär att man kan genomföra installationen i delsteg, ta ett snapshot och fortsätta efteråt. Troligen behöver ni inte göra det eftersom *Debian* är ganska lätta att installera. Det här kommer att bli mest användbart om ni installerar *Gentoo* och de tidpunkter då det är lämpligt att ta ett snapshot är a) efter allt är partitionera och monterat b) efter tarballarna är nedladdade och uppackade c) efter en så kallad `chroot` är genomförd d) Efter kärnan är kompilerad e) straxt innan *Grub* är installerad. Eftersom *Arch Linux* anses lättare är det inte lika angeläget att ta snapshots vid installation av *Arch Linux*. Det här är bara ett förslag, ni kan förstås ta tätare snapshots om ni vill, allt eftersom ni själva behöver. Se manualen för *VirtualBox* om hur man tar en snapshot. Man kan också bara frysa maskinen i ett läge och spara läget utan att ta ett snapshot nästa gång man startar fortsätter installationen från den punkten, det kan vara lämpligt om man bara vill pausa installationsproceduren.

4.3. Inga (eller mindre) drivrutinsproblem

Normalt då man installerar ett OS behöver man lägga mycket omsorg kring vilka drivrutiner som ska installeras. Eftersom vi arbetar virtuellt kommer vi att att få samma drivrutiner att hålla reda på, de som finns i *VirtualBox*. Det är en sanning med modifikation dock, närverksdrivrutinen kan komma att vålla en del bekymmer, men vi får krångla oss förbi det.

4.4. Modern teknik

Virtualisering är en av de viktigaste trenderna och arbetssätten inom modern datorteknik. Istället för att ha 10 datorer som kör olika serverprocesser väljer man gärna idag att ha en dator som kör 10 virtuella datorer som kör dessa serverprocesser. Lägg märke till att jag refererar inte till server som om det vore en dator, en server är en *process*, inte en dator. Men serverprocessen måste ju köras på en dator så då brukar man förenkla det hela genom att säga "servern" och då mena en dator som är dedikerad till att köra en viss serverprocess, till exempel en webbserver eller en databasserver. Man brukar då tala om *servervirtualisering*. Ni kommer att få arbeta med virtuella servrar i nästa kurs, datortekniskt projekt. Hela virtualiseringsbegreppet ligger förstås också till grund för "Moln-begreppet" där de virtuella resurserna även flyttar ut på nätet, så att man alltså via nätet har tillgång till dokument, nätverk och datorer som då finns i något slags virtuell miljö som man interagerar med via nätet som sagt. Det dröjer väl ett tag tills molnet - "*Cloud computing*" - blir fullt utbyggt, men det kommer troligen mer och mer. Vi är ju många vana vid att ha mejl över webben.

5. Varför Linux?

Linux är ett slags *UNIX*-system och har den allmänna *UNIX*-skärpan och systematiken och den måste ni kunna som dataingenjörer. Att installera ett *UNIX*-system sker med ett så kallat installationsmedium och installationsproceduren är detaljerad och man är benägen att ofta göra fel, men då man en gång genomfört den så har man lärt sig mycket.

7. Varför inte Windows?

Windows-familjen av system är världens mest använda system på persondatorer. *Microsoft*s programvaror och standarder är också mycket utbredda i näringliv och samhälle i stort. Hur många kör *Windows* här som primärt system på er maskin? Vi kommer inte att studera så mycket *Windows* i denna kurs, även om *Windows* är ett viktigt operativsystem. Anledningen är att saker förändras så

snabbt i *Windows*-världen så det som är nytt idag är gammalt när ni tar examen om några år. Jag har inget emot *Windows* i sig men jag vill inte att det ni lär er i denna kurs ska vara omodernt när ni kommer ut som ingenjörer. Jag vet att årskurs 3 innehåller mer om *Microsoft*s produkter.

8. Varför inte Mac OS X?

OS X är redan ett *UNIX*-system! Eller, ja, det är *baserat* på en *BSD*-kärna så genom att arbeta med *Gentoo Linux* och *Arch Linux* lär vi oss saker som gäller för *Mac OS X*. Dessutom så innehåller *Mac OS X* också det avancerade kopieringsskyddet (DRM) som vi gärna undviker.

9. Allmänt arbetssätt, att använda hjälpforum och nätet, manualsidor, att låta installationsprocessen bli lärorik. Kommentarer om detaljsupport.

Vårt mål med kursen är att ni ska inhämta maximalt med egen kompetens/förmåga. Vi har ett par riktlinjer för att nå detta

9.1 Arbeta med egen drivkraft med frågeställningar

Det finns ett dokument som är bra att läsa för att beskriva attityden till hjälp och stöd, det heter "*How To Ask Questions The Smart Way*", googla på det så hittar ni det. Det finns en speciell passage där som låter så här: "you will learn more if you seek out the information than if you have it spoon-fed to you." Det vill säga, om ni finner den information ni behöver av egen självständig kraft så kommer ni att lära er mer än om någon bara matar er (med sked) med informationen. Av den anledningen består laborationerna på installationssidan väldigt mycket av uppgiften att *själv* söka den information man behöver. Dokumentet "*How To Ask Questions The Smart Way*" beskriver också hur man kan använda hjälpforum på nätet, ni kan använda hjälpforum på nätet för att lösa installationsproblem, men det är väldigt viktigt att ni samtidigt läser manualsidor och experimenterar med egen drivkraft – ni ska ju ha en självständig kompetens. Läs mycket gärna "*How To Ask Questions The Smart Way*". Det händer ibland också att om man ställer en fråga på ett hjälpforum utan att följa riktlinjerna i "*How To Ask Questions The Smart Way*" så kan man få bryska svar som RTFM vilket är ett argt sätt att säga "*Read The Manual*". Eller STFW vilket är ett argt sätt att säga "*Search The Web*". Båda dessa responser innebär att svaret man söker finns i en manualsida eller på webben.

9.2 Detaljsupport på olika värddatorer

Jag kan tyvärr inte ge detaljsupport på enskilda plattformar, med det menar jag att ni själva ska ta ansvar för att lära känna er dator som ni kör, ert operativsystem och se till att er virtualiseringsprogramvara fungerar. Det kan innebära att ni behöver ominstallera *VirtualBox* (eller vad ni nu använder) tills det fungear. Jag lägger inte upp någon plattformsberoende virtualiseringsprogramvara på kurswebben utan jag antar att ni väljer en och installerar den på er dator.

Om ni inte vet hur man startar en *VirtualBox* på er dator, gå tillbaka och repetera introduktionskursen i *Linux* från *InfoMet*. Jag antar att ni har den klar när ni går denna kurs.

9.3 Hur arbetar man med frågeställningar då?

Ovanstående betyder absolut inte att ni inte får fråga om saker, det bästa som finns är frågor, så ställ alla frågor ni kan, men var också medvetna om att det bästa sättet att arbeta med en frågeställning är att ha en eget förankrat arbetande med frågeställningen. Det innebär att ni mycket gärna kan ställa en fråga, men de svar ni får kommer ibland att låta så här:

"Vad säger manualsidan? (RTM) Vad säger *Google*? (STW) Vad säger handboken?"

Det är alltså viktigt att ni konsulterar dessa kunskapskällor och *att er fråga uttrycker det.*