Introduktion till Linux och små nätverk

Inlämningsuppgift fem

Denna inlämningsuppgift består av tre delar.

1. Ni skall kontrollera vilka tjänster/portar som är öppna på datorn.
2. Ni skall starta två servrar, dels en webbserver och dels en filserver för NFS (Network File System).
3. Ni skall installera och ställa in en brandvägg så att ni begränsar användningen av datorn.

Del ett

För att se vilka tjänster som är igång på en server som andra datorer kan använda, så kan man titta vilka TCP och UDP-portar som är öppna. Om en port är öppen, så finns det nämligen ett program som lyssnar på den porten. Det programmet kan då göra olika saker beroende på vilken data som skickas till programmet via porten. Detta är vad ett server-program gör, som exempelvis en webb-server eller datorpostserver.

Det finns många olika typer av servrar som utför olika tjänster, vi har redan använt två stycken. Dels SSH-servern för att ansluta till datorn för att utföra kommandon (`ssh(1)`) och att kopiera filer (`scp(1)`), dels NTP-servern för att se till att klockan alltid går rätt och att andra datorer kan ställa in sin klocka enligt serverns klocka.

Hur ser man då vilka portar som är igång?
Det finns många sätt att göra det på, vilket man använder beror på många olika saker. Som om man är på samma dator som servern och vill se vilket program som lyssnar på en port eller om man vill se om tjänsten är tillgänglig från en annan dator etc.

För att titta i en dator se vilka portar den har öppna samt vilka program som är anslutna till dem så använder man vanligen `netstat(8)`. Om man vill se vilka portar som är öppna från en annan dator så kan man använda kommandona `nmap(1)` och/eller `tshark(1)` eller det grafiska programmet `wireshark(1)` som använder `tshark`. Programmet `nmap` tittar på vilka portar som är öppna hos en dator och med `wireshark/tshark` kan man se all data som skickas mellan datorer.

De kända portarna, som är kopplade till speciella tjänster har även namn. De namnen finns listade i en fil som heter `/etc/services`. Om man vet namnet på den tjänst man vill se, så kan man använda programmet `grep(1)`, exempelvis `grep -e http /etc/services` söker efter strängen `http`. Det går även utmärkt att använda kommandot `getent(1)` för att slå upp information i filen `services`, exempelvis genom att skriva `getent services http` eller `getent services 22/tcp`.

Kontrollera öppna portar lokalt

Normalt använder man sig här av kommandot `netstat(8)`. Med kommandot `netstat` kan man få reda på mycket information, och vad man kan välja vilken information med växlar. Så för att titta på Internetanslutningar *till och från* den lokala datorn, så kan man använda kommandot

`netstat -A inet` (eller `–protocol=inet`). Om man bara vill ha TCP-anslutningar, så använder man växeln `-t` (eller `--tcp`) och för UDP använder man växeln `-u` (eller `--udp`). Om man bara vill ha reda på vilka servrar som är igång och lyssnar på olika portar, så kan man använda växeln `-l` (eller `--listening`).

Notera att det finns portar som inte går via nätverket som man även ser om man inte lägger till någon växel eller använder `-a` (eller `--all`). Dessa används för att kommunicera mellan olika lokala program.

Kontrollera öppna portar över internet

Man kan titta på vilka portar som är öppna från en annan maskin med programmet `nmap(1)`, eller grafiska programmet `zenmap(1)` som använder sig av `nmap`. Enklaste sättet att använda `nmap` är att skriva `nmap` *`maskinnamn`*. Då får man en lista av portar som man kommer åt från den maskin som man kör `nmap` från på *`maskinnamn`*. Detta är ett bra sätt att se vad som tillåts passera en brandvägg. Om man lägger till växeln `-A`, så får man även reda på operativsystem och vilken programvara som har porten öppen. Växeln `-Pn` brukar man även använda, så hoppar `nmap` över att använda `ping` för att se om maskinen är igång.

Vill man se vilken trafik som går via sin maskin, så kan man med fördel prova programmet `wireshark(1)`. Wireshark visar exakt vad som skickas mellan olika datorer, samt avkodar informationen så man ser vad som skickas. Det är ett avancerat verktyg, men kan hjälpa till att förstå hur saker fungerar. Så vill ni titta lite noggrannare på hur nätverket fungerar, så använd gärna det.

Uppgift 1

Kontrollera med `netstat` och `nmap` vilka portar som är öppna på er server. Jämför resultaten. Vad visar de för något? Vad är skillnaden mellan dem?

Del två

Att använda en webbserver är ganska enkelt i Debian. Det finns flera olika webbservrar, och de vanligaste är `Apache`, `nginx` och `lighttpd`. De finns paketerade och klara så de går bra att installera från Debians förråd. I laborationen kommer vi att beskriva `Apache`, men ni får gärna använda någon av de andra om ni vill. Om inget annat anges, så körs webbservern som användare `www-data` och med gruppen `www-data`. Vilket är bra att veta för rättigheter på filer och kataloger.

Apache

Apache 2, som finns i Debian är en vanlig webbserver som brukar finnas i en kombination av programvara som kallas LAMP (Linux, Apache, MySQL och PHP/Python/Perl). Vi skall här bara titta på Apache, men att installera MySQL och PHP är lika enkelt som att installera Apache. Notera att MySQL även kan anslutas via internet, så man bör se till att MySQL inte lyssnar på någon port via internet. Debian har MySQL konfigurerad att endast lyssna på `loopback` (`127.0.0.1`), så det behöver man inte ändra, eftersom den adressen bara är åtkomlig från maskinen själv. Men det kan vara bra att kontrollera detta varje gång man installerar MySQL.

Apache 2 kan hantera flera webbservrar via samma IP-adress och port-nummer (vanligtvis TCP-

port 80 eller 8080, se `getent services`). Detta görs genom att webbläsaren skickar med vilken adress som webbläsaren ansluter via. Så exempelvis skulle webbsiterna `http://www.hig.se/` och `http://schema.hig.se/` kunna använda samma webbserver. När webbservern får ett anrop från webbläsaren så kommer webbläsaren tala om vilken av `www.hig.se` och `http://schema.hig.se/` som den vill titta på, och webbservern kommer att använda inställningarna för respektive webbserver. För att tala om att dessa webbsiter finns görs i en konfigurationsfil som heter `/etc/apache2/apache2.conf`

Apache 2 kan även enkelt lägga till eller ta bort funktioner med något som kallas moduler. Dessa kan användas för att exempelvis spåra och hantera inloggning etc. Dessa moduler kan man lägga till eller ta bort genom att ändra i samma konfigurationsfil som tidigare, nämligen `/etc/apache2/apache2.conf`[1].

Alla inställningar, konfigurationer, för Apache 2 finns i filen `/e/a/apach2.conf`.
I Debian går det att konfigurera som ”vanligt” genom att ändra i den filen och starta om servern etc. Men för att förenkla konfigurationen så har Debian delat upp `/e/a/apache2.conf` i flera små filer, exempelvis för att hantera moduler. Att konfigurera Apache från `/e/a/apache2.conf` är inte helt enkelt, därför att så lång och mycket konfigureras i den, se ovan.

För att aktivera eller konfigurera moduler så finns det två kataloger `/etc/apache2/mods-available` och `/etc/apache2/mods-enabled`. I katalogen `/e/a/mods-available` så finns alla moduler som är installerade. Varje modul består av en fil vars namn slutar med `.load` och det kan finnas en konfigurationsfil vars namn slutar med `.conf`.

I katalogen `/e/a/mods-enable` finns då alla moduler och deras inställningsfiler som är aktiverade. Så finns de bara i `/e/a/mods-available`, så använder sig inte Apache 2 av dem. Vill man göra ändringar i en moduls inställningar, så ändrar man i filen som slutar med `.conf`.

För att aktivera en modul så kan man göra några steg manuellt (skapa en symbolisk länk, kopiera en file och starta om apache 2) eller så använder man kommandot `a2enmod` eller `a2dimod` för att aktivera (enable) eller stänga av (disable) en modul. Se manualsidan för kommandona.

Motsvarande finns sedan för webbsiter.

I katalogen `/etc/apache2/sites-available` så finns de inställningar som man har för olika webbsiter. I `/etc/apache2/sites-enabled` finns de som är aktiverade. Om man vill lägga till en webbsite, så kopiera en av filerna i `/e/a/sites-available` till ett nytt namn, exempelvis `www.example.com.conf`, och redigera den nya filen. Sedan kör man bara kommandot `a2ensite` *`sitenamn`* för att aktivera den. Om man behöver göra något mer, så skriver kommandot ut vad det är. För mer information läs manualsidan för `a2ensite` och `a2dissite`.

I denna site-fil så står bland annat vilket DNS-namn som webbservern skall använda, vilka moduler den skall använda och vilka kataloger som webbsidorna finns. Ni bör ändra så att sitens webbsidor lagras på något annat ställe än under katalogen `/var/www`. Detta eftersom annars så kan installation av något annat webbprogram störa redan befintliga installationer.
Dessutom blir säkerhetskopiering enklare. Data för servrar brukar finnas under `/srv/`, så lämpligtvis skapar man en katalog där för varje webbserver där, exempelvis `/srv/www/`*`www.example.com`*`/html`. Eftersom webbservern exekverar som användaren

1 Path:en `/e/a/apache2.conf` är ett sätt att förkorta `/etc/apache2/apache2.conf`, som går att använda när man av tidigare text vet vad `e` och `a` skall motsvara i den förkortade versionen.

`www-data` (se exempelvis kommandot `top(1)` och titta efter processerna som kör kommandot `httpd`, eller i katalogen `/var/www`), så måste ni komma ihåg att ändra rättigheterna på den nya katalogen så att webbservern kan läsa webbsidorna.

Uppgift 2

Installera `apache2` (eller någon av de andra befintliga webbservrarna) i er server. Ändra innehållet i webbsidan (`index.html`) så att den inte har standardvärdet. Titta sedan på sidan via lämplig URL, som exempelvis `http://localhost/` om ni kör webbläsaren på samma maskin som `apache2`. Annars så kan ni prova `http://`*`server`*`/` om ni kör webbläsaren från någon annan maskin. Eventuellt så kan ni behöva använda IP-adressen till servern istället för namnet *`server`*.

Dokumentera vilka kommandon nu kör samt vilka filer ni ändrar i. Det skall vara lätt att följa vad ni gjort.

En NFS-server

För att dela filer i ett lokalt nätverk finns det många protokoll att använda. Exempelvis så kan man använda protokollet NFS (Network File System) för att dela filer, framförallt mellan Linux/Unix-datorer. För att dela filer (och skrivare) med MS Windows-maskiner så använder man vanligen SMB/CIF (Server Message Block/Common Internet File System), även om NFS fungerar med.

Eftersom vi håller på med Linux, så kommer vi att titta på NFS. NFS finns med direkt i Linux, men man behöver installera några paket på klienten som skall använda NFS och några andra paket på NFS-servern. Då kan man göra kataloger och underkataloger tillgängliga över nätverket.

NFS kommer i två varianter, NFS version 3 (och lägre) samt version 4. NFSv4 är krypterad och förbättrad jämfört med NFSv3. Dessutom så kan den se till att samma användarnamn fungerar mellan klienten och servern utan problem, även om de har olika uid och gid i klienten och servern. Den översättningen görs då automatiskt av NFSv4.

Men om man behöver använda NFSv3, så är det mycket fördelaktigt om man se till att alla maskiner som använder NFS har samma värden för en användare/grupp i `/etc/passwd` och `/etc/group`. Det som är viktigt är att användarna med samma namn har samma användar-id (`uid`) samt att alla grupper med samma namn har samma grupp-id (`gid`) på alla maskinerna. Annars kan man bli lite förvirrad när en användare på servern plötsligt dyker upp under ett annat namn på klienten när man gör `ls -l`. Linux använder nämligen bara `uid` och `gid` för att avgöra användar- och gruppnamn. Så när man loggar in så letar Linux upp namnet i `/etc/passwd` och så använder det `uid` och `gid` som identitet på vilken användare man är och vilken grupp man tillhör. Se manualsidorna för `passwd(1)`, `passwd(5)` och `group(5)`.

NFS-klient

För att använda NFS från en klient så installerar man paketen `nfs-common` och `rpcbind` (tjänsterna kom tidigare i paketen `nfs-client` och `portmap`).

När de är installerade kan man monterar (gör tillgänglig) filsystem från servern med kommandot `mount(8)`, som med andra filsystem. Så om man vill göra alla filer och kataloger från servern *`server`* i katalogen `/srv/home/kalle` tillgängliga i katalogen `/home/kalle`, så kan man skriva `mount` *`server`*`:/srv/home/kalle /home/kalle`. Detta förutsätter att katalogen

`/home/kalle` redan existerar.

Vill man att katalogen skall monteras varje gång som maskinen startar, så kan man skriva in en ny rad i `/etc/fstab`, så kommer monteringen att ske automatiskt. Den kan se ut så här:

```
server:/srv/home/kalle  nfs  /home/kalle  nfs  default  0  0
```

För att se vilka filsystem som en server exporterar med NFS, så kan man med fördel använd kommandot `showmount -e` *`server`* från en klient (eller servern själv).

NFS-server

För att göra `/srv/home/` (eller `/srv/home/kalle/`) tillgänglig från servern, så måste man först installera paketen för NFS-server, nämligen `nfs-kernel-server`, `nfs-common` och `rpcbind`. Man kan även kontrollera att översättningen mellan användarnamn fungerar, genom att kontrollera att i filen `/etc/default/nfs-common` står `NEED_IDMAPD=YES`.

Sedan så ser man till att konfigurationsfilen `/etc/exports` innehåller en rad som gör `/srv/home/` (eller `/srv/home/kalle/`) tillgänglig över NFS. När man gör det kan man begränsa så att bara vissa maskiner eller nätverk kan montera och använda katalogen. Man kan även begränsa så att de bara kan läsa filer. Exempel på en rad kan se ut så här:

```
/srv/home 192.168.2.0/24(rw,no_root_squash,no_subtree_check,crossmnt,fsid=0)
www.first.com(ro)
```

Denna rad gör att allt under katalogen `/srv/home` är läs- och skrivbart från hela nätet men bara läsbart från maskinen `www.first.com`. Normalt har man mer specifik i början, men här går det bra, eftersom `www.first.com` inte finns i det privata nätet `192.168.2.0/24`.

Notera att maskiner som skrivs efter en annan maskin får samma inställningar som föregående maskin, förutom de förändringar som står inom parenteserna för de maskinerna. Se manualsidan för `exports(5)` samt dokumentationen för paketen som är installerade (`/usr/share/docs`).

Uppgift 3

Ni skall installera en NFS-server och NFS-klient, samt ställa in dem så att katalogen `/srv/data` görs läsbar för hela det lokala nätverket men skrivbar endast från en maskin. Om man inte har flera maskiner i LAN:et, exempelvis om man installerat på en Raspberry Pi, så kan man installera server och klient på samma maskin. Men då måste man testa genom brandväggen, d.v.s. montera NFS genom maskinens nätverksadress och inte genom `localnet` (d.v.s `127.0.0.0/8`) eller `localhost` (`127.0.0.1`).

Del tre

Här skall vi skydda servern med en brandvägg i servern.
Vi kommer att använda `ufw` (`uncomplicated firewall`), eftersom den är lite enklare att hantera än de mer kompetenta, som exempelvis `shorewall`.

Alla brandväggar i Linux är baserade på `iptables`[2] (respektiver `ip6tables` för IPv6), men

2 För närvarande så håller Linux-världen på att byta ut `iptables` mot `nftables`. Kommandot `nft` kommer då att ersätta flera olika verktyg (`iptables`, `ip6tables`, etc.). Se Wikipedia för mer information.

eftersom det är så komplicerat att sätta upp en korrekt brandvägg med `iptables`, exempelvis så används programmen `ufw` eller `shorewall` för att gör det enklare att konfigurera brandväggar. Dessa program använder enklare regler som sedan görs om till `iptables`.

Brandväggsprogrammet `ufw` har stöd för IPv6 redan från början, vilket är en fördel med `ufw` jämfört med många andra brandväggsprogram.

För att veta vilka tjänster som används på maskinen så kan man använda `nmap` eller `netstat`. Då vet man vilka tjänster som används, och vilka som skall skyddas.

Grunden när det gäller datorsäkerhet är att **begränsa tillgången till det minimala**. Det vill säga att se till att så få tjänster är igång som möjligt, samt att de som är igång har så begränsad tillgång som möjligt från nätet. Lämpligen görs det genom att blockera allt, och sedan ta bort blockeringen på tjänster allt eftersom man ser att man behöver access till tjänsterna som man blockerat.

När man ställer in en brandvägg, så avgör man först vad man skall göra om inget annat sägs. Detta görs genom att ställa in ett defaultvärde. En brandvägg kan göra tre saker på varje försök till anslutning över nätverket. Den kan godkänna anslutningen, `ACCEPT`; den kan tala om att anslutning inte är tillåten, `REJECT`; eller så kan den bara ignorera anslutningen, `DROP`.

`ACCEPT` betyder att anslutningen fortsätter som om brandväggen inte fanns, medans de andra två hindrar anslutning. Vad är då skillnaden mellan `REJECT` och `DROP`? `REJECT` skickar ett `ICMP`-meddelande tillbaka och talar om varför anslutningen nekats. `DROP` däremot slänger bara bort paketet och den andra datorn får inte veta varför den inte fick ansluta till maskinen och porten.

Att göra `REJECT` är snällare än `DROP` eftersom avsändaren inte behöver vänta så länge innan den får reda på att anslutningen inte fungerar. Men är man väldigt försiktig, så använder man kanske hellre `DROP`. Detta eftersom `ICMP`-meddelandet från `REJECT` talar om att det finns en maskin på den adressen, vilket någon som attackerar kan använda sig av.

Vilket man använder är upp till var och en, men standardinställningen för inkommande `TCP/UDP` är att neka anslutning med `REJECT` eller `DROP`.

Därefter tillåter man varje anslutning som behöver anslutning explicit. Då minimeras risken att man av misstag tillåta något man inte vill tillåta. Om man glömmer att tillåta någon port/tjänst, så märker man det normalt ganska snart. Men om man glömmer att stänga en tjänst, så är det inte säkert att det upptäcks i tid.

Om man har sin server bakom en s.k. NAT, så måste NAT:en i brandväggen konfigureras så att anslutningar mot tjänsten skickas vidare in i LAN:et. Det kan göras på flera sätt, exempelvis så kan man sätta att bara en viss port-anslutning från Internet skickas till en specifik maskin. Eller så kan man säga att alla anslutningar från Internet (som inte är speciellt specificerade ovan eller som brandväggen själv använder) skickas till en speciell maskin, som brukar kallas DMZ host (Demilitarized zone). Den maskinen måste man då vara noga med, eftersom en angripare kan komma åt hela ditt nätverk om den maskinen kompromiteras (hackas).

Detta innebär att man bara kan ha en IPv4-server, som ju har minst en tilldela port, hemma som man har åtkomst via internet. Det kan lösas genom att man kan ha vissa tjänster, som webbserver, tillgänglig på andra portar (som port 80 och 8080 för webbservrar). Allt detta p.g.a. att IPv4-adresserna har tagit slut.

Dessa begränsningar finns inte med IPv6, som vi skall titta på i projektet.

Brandväggen i servern med UFW (Uncomplicated Firewall)

Det första man gör är att installera paketet `ufw`. Kontrollera därefter i filen `/etc/default/ufw` att inställningarna ser bra ut, d.v.s. att IPv6-stödet är aktiverat.

Sedan ställer man in standardinställningen för inkommande `TCP/UDP` genom brandväggen. Man kan då ställa in `allow`, `deny` eller `reject`. Inställningen `allow` är samma som `ACCEPT`, `deny` är samma som `DROP` och `reject` är samma som `REJECT` i `iptables`.

```
sudo ufw default reject
```

Man kan även ställa in hur mycket data man vill skriva i loggen om en nekad anslutningar (vanligtvis logg-filen `/var/log/syslog`, kom ihåg `tail -f /var/log/syslog` i ett annat terminalfönster för att lätt se vad som händer i servern i realtid). Det kan vara bra att ha ganska mycket information i loggen i början, för att sedan minska ned antal saker som skrivs i loggfilen när man vet att det fungerar. Så här slår man på loggningen.

```
sudo ufw logging on
```

Så för att ställa in brandväggen så tar man först reda på vilka portar man vill tillåta, vilket görs med `allow`. Normalt är det SSH, så att man kan ansluta med programmet `ssh(1)` för att administrera datorn. Eftersom `ssh` är krypterat, så är det ok, ingen kommer att kunna avlyssna nätverket och se lösenord, som man kan med exempelvis `telnet` och `ftp`. Därför, undvik dessa protokoll. Vissa protokoll, som `ssh`, kan man lämpligen begränsa antalet anslutningar per minut från samma maskin, vilket görs med `limit`. Exempelvis så kan det se ut så här.

```
sudo ufw allow app OpenSSH
sudo ufw limit app OpenSSH
```

När det är gjort, så kan man aktivera brandväggen. Notera att man först godkänner att man ansluter till SSH innan man startar brandväggen. Om man inte gör det, så kan av misstag låsa ut sig själv från maskinen när brandväggen aktiveras. Kommer man då inte åt maskinen fysiskt, kan det bli jobbigt.

```
sudo ufw enable
```

Sedan så tillåter man var och en av de andra tjänsterna. Då kan man tala om ifall tjänsten är tillgänglig från hela Internet, från vissa nätverk, eller vissa datorer. Det gör man genom att skriva ett ufw-kommando för var och en av tjänsterna. Då kan man även specificera om man vill ha extra loggning etc. för varje rad.

När man är klar, så skriver man ut status så att man kan se hur brandväggen är inställd.

```
sudo ufw status
```

Om man vill återställa brandväggen i ursprungsläge, så kan man använda `sudo ufw reset`. Man kan även lägga till eller ta bort regler som är inlagda ett i taget genom att ange vilket radnummer som regel skall läggas till som eller vilken som skall tas bort.

Notera att reglerna används i den ordning de listas av `ufw status`. Så se till att mera specifika regler står före de mer allmänna reglerna. Så vill man tillåta en maskin i det lokala nätverket men inte de andra, så tillåt först den enskilda maskinen innan regeln för att neka alla andra läggs in.

Notera även att man vanligtvis bara skriver regler om inkommande anslutningar.

Andra möjliga inställningar med `ufw` är att man begränsar antal anslutningsförsök till 6 st under senaste 30 sekunderna. Om det blir fler, så stängs alla anslutningar från den maskinen som den regeln bevakar under en längre tid. Detta görs med `limit` och är lämpligt för `ssh`, men inte en webbservern. Läs gärna manualsidan för `ufw(8)`.

Uppgift 4

Ställ in brandväggen för servern så att alla kan komma åt SSH och HTTP, men bara maskiner i det lokala LAN:et kan komma åt NFS-tjänsterna, notera att det är flera portar som används av NFS. Ni skall ställa in så att antalet uppkopplingar mot SSH från samma adress begränsas med `limit`.

Se även till att maskinen som ni kör `nmap` från inte får komma åt HTTP. Redovisa vilka kommandon ni har använt samt resultatet. Det sista gör ni lämpligen med `ufw status`.

Demonstrera hur brandväggen fungerar med `nmap`, d.v.s. vilka anslutningar som blockeras och vilka som tillåts. Dels från servern och dels från en annan maskin.

Om ni inte har möjlighet att göra demonstrationen, visa mot någon annan maskin hur `nmap` fungerar, exempelvis maskinen `www.hig.se` och `rigel.hig.se`.

Rapporten

Ni vet ju vid det här laget hur vi vill ha rapporten skriven. Rapporten som ni skriver skall innehålla *ett försättsblad* som innehåller *laborationens namn*, *datum*, *ert namn*, *födelsedatum/personnummer* samt *datorpostadress* (på högskolan).

Rapporten skall vara skriven så att vi kan förstå att ni förstått samt ser vad ni gjort. Ingen roman behövs dock. Det borde räcka med 4-7 sidor totalt med text som den här laborationsunderlaget. Följande delar/rubriker kan vara bra att ha i rapporten.

1. Försättsblad
2. Innehållsförteckning (ej nödvändig)
3. Inledning: Ni beskriver problemet och vilka frågor som skall besvaras
4. Genomförande: Här beskriver ni hur ni har löst laborationen
5. Slutsatser: Här beskriver ni svaren på frågorna i Inledning:en
6. Övrigt: Om ni vill lägga till något som inte får plats i Slutsatser
7. Bilagor: Här lägger ni stora bilder och programlistningar

Rapporten skall vara i PDF-format och eventuellt tillhörande filer lägger ni in som bilaga till rapporten.

Nu har ni det som behövs i `laboration5-`*`mitt-namn`*`.pdf`, skicka in och ni är klart!

Om uppgiften och forum

Om ni får **problem**, så **ställ frågor i forumet** som finns **i BlackBoard**. Att lära sig att administrera datorer handlar om att i forum kunna ställa rätt frågor, så det kan ni gärna öva på här.

När ni ställer en fråga, så **skall ni beskriva** vad ni **vill göra**, vad ni **har gjort** samt **vad ni förväntat er** skall ske samt **vad som skett**. Om ni beskriver för dåligt, så kommer ni att få frågor om mer information. **Tänk på att de som läser era frågor inte har sett vad ni gjort, så det är ert ansvar att förklara så att de andra förstår ert problem och kan besvara frågan**.

Ni får även gärna svara på frågor i BlackBoard, där medstudenter förklarat vad de försökt med och vad som inte gått som de tänkt. Begär mer information om ni inte har fått tillräckligt med information så att ni förstått vad som frågats efter. Samt i den här kursen så begär att få veta vad de som ställer frågan har gjort innan ni svarar.

Lycka till!

Anders Jackson

Referenser

http://www.debian.org/

http://wiki.debian.org/

http://www.debian-administration.org/

http://www.howtoforge.com/

Apache

http://wiki.debian.org/Apache

http://httpd.apache.org/

http://www.control-escape.com/web/configuring-apache2-debian.html
(bra beskrivning av inställningar av Apache 2 i Debian/Ubuntu)

https://library.linode.com/web-servers/apache/installation/debian-6-squeeze

http://wiki.debian.org/SubversionApache2SSLHowto
http://wiki.debian.org/EtchApache2DefineSSL
(hur man kan skapa en HTTPS-server)

NFS

http://sv.wikipedia.org/wiki/Network_File_System

http://www.cyberciti.biz/faq/nfs4-server-debian-ubuntu-linux/
(hur man konfigurerar NFSv4)

UFW

http://en.wikipedia.org/wiki/GUI_for_Uncomplicated_Firewall

https://wiki.ubuntu.com/UncomplicatedFirewall

https://help.ubuntu.com/community/UFW

Program

https://www.wireshark.org/

http://nmap.org/