Studietips Inför Tentan -- Advanced Enterprise Linux Administration

Kap 1, Pre-Installation Considerations

Det är naturligtvis viktigt att ha klart för sej grundläggande krav som hur systemet skall partitioneras, vilka paket man behöver, om det behövs en swap, var skall GRUB läggas och liknande praktiska frågor. Helst innan man sätter igång. När man installerat ett större antal system brukar man lära sej vad som fungerar, men ni har ju fördelen att ha både bok och skola som vägledning. Valen styrs ofta av både fysiska krav, men även av tycke och smak; företagsstandarder och liknande.
För er som kommer att bli konsulter är det viktigt att snabbt identifiera viktiga grundbultar i installationsfilosofin hos kunden så att ni inte installerar system som verkar ”utomjordiska" och konstiga för kunden. Om kunden förväntar sej att /var skall ligga på en egen disk och att det skall swappas på samma disk där /usr ligger så kan det ju bli fel om du klumpar ihop / boot med /var och levererar ett system utan swap...

Kap 2, Installing RHEL5

Med laboration 1 avklarad så tror jag ni är ganska väl förtrogna med installationsprogrammet Anaconda och vad man kan göra. Men glöm inte möjligheterna att installera från **ovanliga lagringsmedier som exempelvis ftp, nfs, http och pxe.** Vidare är det viktigt att komma ihåg att det går att **installera ”på distans"** via ssh och vnc, en räddning för många som inte står ut med fläktbrus eller för de som känner att de vill ha nära till kaffemaskinen under en besvärlig installation...

Kap 3, Linux Hardware Discover, Interaction, and Control

Detta kan vara det mest frustrerande med hela installationsprocessen, saker vägrar att fungera fast man försökt välja rätt grejjer. För hemmanvändaren kan det vara så enkelt som att välja en annan distribution, men i serverhallen är sällan det valet aktuellt. **Kommandon som dmesg lsusb, lspci och lsscsi** är en bra början för att få veta vad operativssytemet tror om hårdvaran. Läsa loggfiler som **/var/log/messages** kan också vara intressant. Vi får inte glömma de automatiska hjälpprogrammen som **kudzu och udev**...

Kap 4, Boot Process and Sysv Init

Grundkonfiguration av startprogrammet GRUB, brukar Anaconda klara av bra, i de flesta fall. Men fortfarande kan **hiddenmenu, lösenordsskydd** och liknande vara bra saker att känna till. Ni skall känna till att systemet laddar en ram-disk, **initrd**, som en del av uppstartprocessen för att göra det enklare att konfigurera olika typer av hårdvara innan innan systemet börjar montera filsystem som sitter på exotiska diskar med mystiska kontrollerkort, ni kommer att få bekanta er närmare med den saken när vi tittar närmare på hur man bygger kärnan.
Men det viktigaste steget under uppstarten är processen Init, eller rättare sagt processen **initd**, med konfigurationsfilen **inittab**. Ni måste naturligtvis veta att olika tjänster (demoner...) hanteras genom startskript under **/etc/init.d/** och att dessa länkas till kataloger som **/etc/rc5.d/** och liknande, **chkconfig**...

Grundläggande känsla för hur och när skripten sedan körs kan vara bra, men det är ju sällan du behöver ändra den ordning som skaparen av paketen tänkt ut. Däremot kanske du vill göra en del ”fulhack” i **/etc/rc.local**...

Kap 5, software maintenance

Detta är ju mycket grundläggande saker och för många kommer det praktiska systemarbetet att vara den givna vägen till kunskap här. Läs gärna boken och fungera igenom sakerna. De vanligaste problemen, brukar för det första vara hur man hanterar situationer där det ”saknas” paket i standard repository; hur gör man när man behöver en nyare version eller något helt annat? EPEL och andra kan vara bra, ibland tankar man hem paket till andra distar och fulinstallerar med rpm. Det senare fungerar ofta, men man kan komma i lägen då beroende strukturen blir ett enda trassel och då det inte går att uppgradera systemet. Att själv lägga på saker som man kompilerat kan också verka frestande, men detta kan vara lika problemfyllt som rena ”fulinstallationer” så vägen utanför standardpaketen kan vara ganska tröstlös.

Kapitel 6, filsystem och lagring

Läs och begrunda, tror inte ni kommer att se så många nyheter från det vi redan gått igenom, åtminstone inte om ni hängt med på lektionerna. Hittar ni saker som ni inte känner till måste ni, naturligtvis, kolla upp dessa. Vad som kan vara att notera, vi ägnar oss inte åt Storage Area Networks (SAN) eller Network Attached Storage (NAS) på kursen, ni skall känna till termerna och ni bör veta (ungefär) vad sakerna används till; sedan kommer ni ut på lia där dessa saker kommer att bli självklara.
Ni bör bundera över olika backuprutiner, inte bara tekniskt (cpio och liknande) utan även administrativt, hur ofta backar man? vad backas? hur länge sparas det? var sparas det?

Kapitel 7, LVM och RAID

Det är naturligtvis kul att köra raid, konfigurationer där man kopplar ihop flera diskar för att antingen skapa bättre prestanda, säkrare lagring eller större lagringsutrymme; men inte alla tre sakerna samtidigt. För att raid skall fungera långsiktigt krävs det fungerande övervakningsfunktioner och rutiner för dessa. Samma sak med LVM, det går att bygga facinerande och avancerade lagringsstrukturer, men det är lätt att gå vilse i all kul teknik och glömma bort varför vi har lagring över huvud taget. Utan övervakning, backup och seriösa rutiner, så kan ett litet fel fortplantas till att -i princip- alla filer är borta innan du upptäcker att saker gått snett. Därför är det viktigt att tänka efter före, behöver jag verkligen detta? Vad skall jag göra för att det skall fungera långsiktigt? Hur sköts säkerhetskopieringen?

Kap 8, Remote Storage Administration

Utbildningen är *inte* avsedd att hantera moderna lagringslösningar som NAS och SAN, eller ens bandstationer. Ni kommer att lära er sådant på er LIA. **Däremot är det viktigt att kunna sätta upp NFS och att känna till de grundläggande säkerhetskonceptet bakom NFS**. Att veta att alla användare förväntas vara tillräckligt autenticerade på de system de använder för att ansluta till servern. Ni som kommer från windowsvärlden är ju vana vid att servern autenticerar varje enskilld användare innan denne tillåts hantera några filer. **Ni skall känna till att det finns mer avancerade autenticeringsmetoder till NFS**, men att dessa oftast inte används eftersom de ofta är buggiga.

Kap 9, Remote Storage Administration

Rent praktiskt, ute i arbetslivet och på LIA, kommer saker som diskarrayer i form av Networked Attached Storage (NAS) och Storage Area Networks (SAN) att vara mycket viktiga att kunna hantera. Men eftersom detta är teknik som hela tiden utvecklas så väljer vi att studera sådana saker som alltid är aktuella och användbara. Därför skall ni rikta in er på NFS och AutoFS framför iSCSI och SAN-teknik, åtminstone inför tentan. Men bli inte förvånad om jag skulle undra över vad man använder ett SAN till, eller om vad ett NAS kan vara...

Kap 10, PAM - Pluggable Authentication Modules

Mycket användbar teknik, ni skall -på det här stadiet- känna till den och veta att PAM används av de flesta program som kräver autenticering av användare (inloggning alltså). Ni skall även veta att det går att konfigurera olika parametrar om hur användaren får välja lösenord, när han får logga in och även saker som ”bäst före” datum på lösenord. En annan viktig aspekt är att PAM medger utbyte av användarnamn och lösenord till helt andra saker, allt från biometri (avläsning av fingeravtryck, näthinnor och DNA, för att nämna några saker). Men ni kan vänta till systemsäkerhetskursen nästa år med att fördjupa er i hur detta konfigureras.

Kap 11, Security Administration

Det är viktigt att ni förstår att enkelhet och ordning är två hörnpelare för systemsäkerhet. Den som lägger på alla paket han hittar och fipplar runt med systemet som om vore det en speldator, han skall inte ha någon anledning att förvåna sej den dagen servern blir crackad.

Ta bort tjänster som inte behövs, blockera konton som inte används; se över systemet så att rimliga lösenord används, bara små bokstäver, eller personnamn, könsord och populära gestalter från populärkulturen hör inte hemma som lösenord på ett säkert system. Det må vara att så väl Hitler, Voldemort, Dracula som Sauron, var och en på sitt sätt kan vara fasansväckande, men de skrämmer inga inkräktare. Använd **/etc/hosts.allow** och **/etc/hosts.deny**. **iptables** är naturligtvis lika praktiskt som det kan vara bökigt att använda, skaffa ett bra redskap att trimma **netfilter** om du inte kommer överens med kommandot **iptables**. Ta bort SUID och SGID behörigheten överallt där den inte är helt nödvändig. Ladda gärna hem redskap för att crackers, den populära distributionen BackTrack (**http:// www.backtrack-linux.org**) kan vara en bra startpunkt. När du kikat runt på vilka metoder ”kidsen" kör med begriper du bättre hur viktigt skydd kan vara.

Till specifika saker:

Behörighetslistor - ACL
Vid sidan av det gängse behörighetssystemet på UNIX/LINUX kan man lägga till ACL:er. Sådana finns inte till alla sorters filssystem och de fungerar inte om du inte slår på dem när du monterar filsystemet; /etc/fstab kan ju enkelt editeras...

setfacl -m u:voldemort:rw /ond/fil.txt	ger användaren voldemort läs och skriv på fil.txt
getfacl -r /ond> /bak.acl	backar behörighetslistorna i trädet /ond
setfacl --restore < /bak.acl	hämtar behörigheterna från backup
tar --acls -cf /ond.tar /ond	backar trädet /ond till filen /ond.tar

Glöm inte bort att ACL:erna inte bara kan verka som lite ”främmande” och att de inte är 100% kompatibla med alla saker ni vill kunna göra, de kan även ge prestandaförluster i många fall. Personligen ser jag att det främsta användningsområdet är att kunna erbjuda en behörighetsmodell som liknar den i Windows, vilket kan underlätta övergången för de som uppgraderar sina operativsystem. De som inte behöver vara kompatibla med Microsoft kan oftast hitta på bra lösningar även med UNIX-modellen utan ACL.

Private Group Scheme (UPG)
är populärt, men förleds inte att tro att det är det enda vettiga sättet att administrera en linuxkärra, med hjälp av traditionella behörigheter (utan ACL).

SELinux
är en modul som ger kärnan möjlighet att ge varje process ett säkerhetskontext där det nogrant definieras vilka filer och resurser som får utnyttjas. Detta är naturligtvis mycket kraftfullt, men kan ofta upplevas som besvärligt och krångligt. Det är lätt att skapa ett system som varken fungerar eller ger vettiga loggar som förklarar varför.
Ni skall veta *ungefär* hur det fungerar, på säkerhetskursen nästa år får ni bekanta er närmare med koncepten.

Kapitel 12, Process Administration

Vid sidan av **at** och **batch** är det huvudsakliga redskapet för automation **crond** med kommandot **crontab**. På CentOS finns både anacron som framför allt hjälper de som inte kör sina datorer dygnet runt, så detta är inte så intressant för serverdrift; vixie cron som är Paul Vixies implementation av cron-systemet är däremot ett mycket viktigt verktyg som du bör bekanta dej med.

man crontab	berättar om kommandot crontab
man 5 crontab	berättar om syntaxen på en crontab-fil

Normalt kan du ofta strunta i de finare detaljerna i crontabbar om du bara vill köra saker regelbundet: gör ett vanligt shell-skript som du lägger i **/etc/cron.daily**; inga speciella krav på syntax, bara ett vanligt skript. **crond** kommer då att köra programmet dagligen; vill du i stället köra det varje timme väljer du **/var/cron.weekly** och så vidare.
Behöver du mer precis kontroll kan du antingen editera /etc/crontab eller skapa en egen crontab-fil som du lägger till med hjälp av kommandot crontab. *Det är en viktig skillnad på syntaxen mellan **/etc/crontab** och vilken crontab-fil som helst: den första kräver att du skriver **användarnamnet efter tidsangivelsen** så att crond vet vilken användare som skall användas för att köra varje kommando. I en vanlig crontab-fil så kör crond programmen som filens ägare.* Eventuell utdata från programmen i cron, skickas på epost till crontab-filens ägare; så använd rörledningar till **/dev/null** om du vill slippa att bli spammad av crond.

Process accounting kan vara intressant, men grotta inte ned er i detta. Däremot skall ni behärska kommandon som ps, top, kill, killall, nice och renice. Likaså bör ulimit finnas i arsenalen, så att du kan begränsa resursanvändningen på användar- och process-nivå.

Kapitel 13, Basic Networking och Kapitel 14, Advanced Networking

För den rena nätverkskompetensen har vi CISCO-blocket, med detta inte sagt att det kan vara bra att titta på det här avsnittet eftersom vi ju kommer att tala en mycket om nätverkstjänster i nästa kurs. Ntp kan dock vara en viktig del för att NFS och andra saker skall fungera väl. Det är dock några saker som är helt nödvändiga för er linuxdrift, först och främst konfigurationsfiler som:

/etc/services
lista över vanliga nätverkstjänster, protokoll och portnummer

/etc/resolv.conf
konfiguration av namnservrar och sökdomän (dns), kudzu kan bråka här

/etc/sysconfig/network-scripts
den katalog där du hittar grundläggande nätverksinställningar för ipadresser, nätmasker, routing och annat.

ifconfig
grundläggande kommando för att styra nätverksgränssnitten, addressering och liknande; resultatet tar omgående, men skall du ha permanenta inställningar som håller sej efter en omstart måste du använda filerna i **/etc/sysconfig/network-scripts**.

ifconfig eth0 10.10.4.42 netmask 255.255.255.0	ställer eth0 till 10.10.4.42/24
ifconfig eth0 10.10.4.42/24	ställer eth0 till 10.10.4.42/24 (precis som ovan)
ifconfig eth0:1 10.10.4.43/24 up	ställer och aktiverar en ytterligare ip med mask på eth0

ifup och **ifdown**
är två praktiska kommandon som aktiverar eller avaktiverar nätverksanslutningen

ifup eth0	startar trafik på eth0
ifdown eth0	stänger av tra

route
konfigurerar och visar routinginställningar; hur datorn hanterar paket till andra nätverk än det lokala.

route add default gw 10.10.4.1	ställer standardadress för att vidarebefordra paket till andra nät (default gateway) till 10.10.4.1
route del default	tar bort ovanstående
route -n	listar rutter
route add -net 10.10.4.0/24 gw 172.16.52.2	lägger till rutt för ett helt nät
route add -host 10.10.4.10 ppp0	lägger till rutt för specifik dator

arp
ARP står för Address Resolution Protocol, som används för att uppdatera information om hur paket skall skickas till olika destinationer. **arp** är ett kommando för att manuellt hantera rutt-tabellen.

arp -a	listar rutt-tabellen
arp -d 10.10.4.43	tar bort rutten till 10.10.4.3
arp -s 10.10.4.97 00:01:02:03:04:05	kopplar ipadressen 10.10.4.97 till en macadress.

ip
är en modern ersättare till de flesta av ovanstående kommandon. De flesta böcker och handledningar på nätet använder fortfarande de gamla varianterna; rent praktiskt är det ingen skillnad på vilken variant du väljer, men många av dina kollegor kan känna sej främmande till nymodigheter. Speciellt om de moderna varianterna inte ger några praktiska fördelar.

netstat och **lsof**
netstat visar information om nätverksanslutningar, uppkopplingar, rutt-tabeller, statistik, vilka program som lyssnar på vilken port och annat tekniskt om nätverket. **lsof** berättar vilka program som lyssnar på olika portar, vilka program som ansluter någonting på en port; saker som netstat också visar. Men **lsof** kan även ge upplysning om vilka filer ett program använder just nu, praktiskt om du har svårt att avmontera ett filssystem för att någon process använder någon fil där.

På ditt personliga system kan du naturligtvis använda program som exempelvis **NetworkManager** eller exempelvis **wicd** för att ställa grundparametrar på olika nätverksgränssnitt. Detta kan vara helt nödvändigt om du skall du drifta servrar för mobila tillämpningar, exempelvis som de system som SL använder i bussarna för stationsutrop och skyltar. Annars torde dessa verktyg mest höra till skrivbordet, där de ju kan vara mycket användbara.

Kapitel 15, X Window

Eftersom detta är en serverorienterad utbildning finns det ingen direkt anledning att ägna tid åt detta avsnitt, ur skolans perspektiv. Däremot kanske många känner intresse av att förstå hur X fungerar och då kan det ju vara trevligt att läsa lite här...

Kapitel 16, Log File Administration

Det är inget mystiskt med loggar, men ni skall veta var de finns. Hur ni konfigurerar syslogd (/etc/syslog.conf), likaså måste ni inse att syslog kan skicka loggar över nätet till någon central server; eller kanske administratorns skrivbord...
Ni bör också veta hur och varför loggarna skall roteras.
Glöm inte **dmesg**...

Kapitel 17, Monitoring and Troubleshooting

Rent tekniskt är detta ett viktigt kapitel, jag kan fråga er om olika kommandon och vad de används till på tentan. Men i verkliga livet är det en massa andra saker ni måste veta som är mycket viktiga; tyvärr tror jag inte jag har någon bra metod att fråga om dessa på ett rättvisande sätt, men jag hoppas att ni inser att detta är ett mycket viktigt avsnitt ändå.

Övervakning och felsökning är viktigt. **top**, **free**, **swapon**, **vmstat, iostat, mpstat** är en kort lista över enkla, men viktiga kommandon. /proc/cpuinfo är en fil som berättar vad du har för processor.
Monitoreringsprogram som **sar** kan vara bra, men många kommer att kunna gå igenom en hel karriär utan att komma i närheten av dessa; men tung serverdrift och det kan vara din bästa vän...
strace är även det ett program som somliga aldrig kommer att behöva och som andra inte kan leva utan; oftast är det väl de mer programmeringsinriktade som behöver det.
Läs igenom och fundera över hur man felsöker, även här kan ordning och reda vara viktigt. Men även erfarenhet och instinkt (som man får genom erfarenhet); man blir aldrig gammal och klok om man inte varit ung och dum...
Boken tar ju upp räddningsmiljön på CentOS boot-skivor, fungerar helt OK i många fall, men de flesta brukar uppleva system som exempelvis följande:

Parted Magic	http://partedmagic.com/
Super Rescue CD	http://www.kernel.org/pub/dist/superrescue
Knoppix	http://www.knoppix.com/

som betydligt roligare. Ofta går det enkelt att klämma in sådana dist:ar på en minnepinne.

Lycka till,
Rolle.