Denne guide er oprindeligt
udgivet på Eksperten.dk

COMPUTERWORLD

Linux på kommandolinje

Kommandolinjen er der hvor den virkelige saft og kraft i Linux er. Denne artikel giver for begyndere og let viderekomne en introduktion til at bruge bash shell, og gennemgår nogle nyttige kommandoer som du næsten helt sikkert får brug for før eller siden

Skrevet den **03. Feb 2009** af **strych9** I kategorien **Server / Linux** | ★★★☆☆

Historik:
1.0.0: Første udkast
1.0.1 (24/6 2004): Rettede lille fejl angående slocate. Pakken hedder på SuSe findutils-locate. (Tak til Jan)

Linux på kommandolinje

Noget der skræmmer nye Linux brugere er alle de valg man pludselig bliver sat over for. Der er ikke bare een god måde at gøre tingene på. - Der er mange gode måder. Jeg kan levende huske de mange gange hvor jeg har siddet og kløet mig i nakken og tænkt "ok, så man kan gøre det på den her måde, men er det best practice?"
Nogle af nedenstående tips og metoder afspejler uvægerligt hvad jeg mener er best practice, og du skal være opmærksom på at der er andre måder at gøre tingene på, og der vil være ting som du med tiden vil foretrække at gøre på en helt anden måde. Fleksibilitet er en force i Linux, men også en kilde til forvirring.

Kommunikation via kommandolinje med de centrale dele af operativsystemet foregår igennem det der hedder en shell. En shell kan forstå en række kommandoer og fortolke dem således at de omsættes til nogle instruktioner som operativsystemet kan forstå.
Kært barn har mange navne, og en shell i linux refereres til med forskellige næsten synonyme navne som feks en terminal, en bash prompt, CLI, eller slet og ret bare "bash". I nogle danske oversættelser er en shell oversat med "en skal". En oversættelse som undertegnede overhovedet ikke bryder sig om, men det burde lige nævnes.

Du skal forstå at der er flere forskellige shell typer at vælge imellem i Linux. En shell er slet og ret bare en eksekverbar fil på din harddisk, og der findes Linux traditionen tro en masse af dem som hver har deres eget særpræg: zsh, Korn shell, Bourne shell og ash blot for at nævne nogle få.
De fleste desktop og server Linux distributioner bruger som standard en shell der hedder bash. Det er en forkortelse for "Bourne-Again SHell". Grunden til det underlige navn er at den er inspireret af Bourne Shell, som er en meget gammel shell som kørte på diverse UNIX typer. Du kan se den eksekverbare fil til bash på din harddisk. Den ligger for det meste i stien /bin/bash, men det skal du nu ikke bekymre dig så meget om på nuværende tidspunkt.
For en ordens skyld kan jeg nævne at bash følger med i den pakke til din distribution der slet og ret hedder "Bash". Den følger med i standard installation af næsten enhver Linux distribution, og du skal selvfølgelig ikke forsøge at afinstallere den.

Hvordan får man så en bash prompt frem så man kan begynde at skrive kommandoer? Der er flere måder at gøre det på, og nogle af dem afhænger af hvilken grafisk brugerflade du har valgt at køre. For at starte en bash shell i Gnome skal du blot højreklikke på desktoppen og vælge "new terminal". I en standard KDE installation er der en ikon der ligner en muslinge skal med en sort skærm bag du kan klikke på. Ellers kan du forsøge at lede menuerne igennem for ord som "shell" og "terminal" eller "term". Den skal nok være der et sted.

Hvis alt andet fejler kan du trykke ctrl + alt + F2, og så vil du se at du kommer ud af X, som er den service din grafiske brugerflade kører på, og ud til en sort skærm hvor du bliver bedt om at logge ind. Dette betyder ikke at din grafiske brugerflade er lukket. Den kommer du tilbage til hvis du blot trykker ctrl + alt + F7. Hvis du har brug for mere end een bash prompt kan du trykke ctrl + alt + F3, og skifte imellem dem så fremdeles. Det er dog rarest hvis man kan gøre det i et terminal vindue på sin desktop.

Hvis du åbner shell via din desktop, og ikke med genvejstasterne, vil du måske lægge mærke til at du ikke bliver bedt om at skrive brugernavn og password. Det er fordi du har åbnet bash via din desktop, og bash vil dermed køre videre med det samme brugernavn og de samme rettigheder som du kører din desktop med.
Linux er et flerbruger system. Man kan køre lige så mange bash shells som man har lyst og RAM til, og man kan køre dem som flere forskellige brugere.
Det er god smag at man som udgangspunkt kun starter shells og processer som en almindelig bruger, i det omfang det er muligt. Der er en speciel brugerkonto i Linux som hedder "root". Det svarer til administrator kontoen i Windows. Når man er logget ind som root så kan man alt på maskinen, men det er ikke altid godt. Linux er ikke idiot-sikret (undskyld udtrykket), og derfor kan man med en forkert kommando komme til at beskadige sit system. Linux er ikke så flinkt som Windows til at spørge om man nu er helt helt sikker på at man vil det man har bedt systemet om at udføre. Samtidig skal man tænke på sikkerhed. Hvis man er logget ind som root og kommer til at køre et eller andet virus eller malware, så kan denne malware det samme på maskinen som root kan. Der er altså ingen begrænsninger i den skade det kan udføre.
Derfor bør man altid logge ind som en almindelig bruger. Heldigvis kan man skifte fra almindelig bruger til root ret nemt i en bash shell når man har brug for det. Man skriver simpelthen bare "su root", hvorefter man bliver bedt om at skrive password til root kontoen. Når man er færdig med at lave de ting man skulle som root kan man blot skrive "exit", og så er man tilbage som den bruger man var før.

Det kan godt være forvirrende med alle de bash shells åbne. Sommetider har jeg selv 10 eller mere kørende, og man kan miste overblikket over hvad man er logget ind som og hvorhenne. Det er der heldigvis råd for. Kommandoen "whoami" kan fortælle hvilket brugernavn du er logget ind som. Prøv det ad.

Nu hvor du så er logget ind, og har fået læst og påskrevet om aldrig at logge ind som root med mindre der virkelig er brug for det, er det tid at nævne det måske allervigtigste i en artikel om at bruge CLI i Linux: Hvordan bruger man hjælpe funktionerne?
Til de fleste Linux pakker hører der en række manual sider. Man kan få adgang til dem med kommandoen "man". Feks kan du prøve at skrive "man whoami" og du vil se at du får noget mere eller mindre brugbar information om kommandoen. For at komme ud af man siden og tilbage til bash skal du blot trykke "q".
Mange Linux kommandoer har det der hedder switches som kan give programmet nogle hints om hvad det skal gøre og måske også hvilken form for output det skal give. En standard switch som de fleste programmer har er "--help". Prøv feks at skrive "whoami --help" og du vil se en hurtig hjælp til hvad programmet gør, og hvilke andre switches på kommandolinjen det forstår.
Man kan sige at manual siderne er nyttige til at forstå hvad et program gør, mens --help er en god hjælp til at forstå hvordan.
Det skal også nævnes at man kan søge i man siderne med den switch der hedder -k. Feks vil "man -k routing" gennemsøge alle man siderne for ordet routing, og man vil dermed have navne på en række kommandoer på sit system som er relevante i forbindelse med routing. Feks dukker netstat og route kommandoerne op på mit system, og de må jo siges at være yderst relevante hvis man skal bruge sit Linux system som router.

En anden vigtig ting som indirekte kan siges at være en hjælpefunktion er den tab completion som bash er udstyret med. En meget tidsbesparende ting som du skal vænne dig til at bruge med det samme.
Prøv engang at skrive "whoa" og trykke på tabulator tasten. Du vil se at bash skriver resten af kommandoen for dig så der står whoami. Hvordan er det muligt? Bash ved godt hvilke programmer og filer der er markeret som executable. Når du trykker på tabulator tasten kigger bash alle programmer igennem som er i din PATH, og bash kan så se at der kun er een kommando på systemet der begynder med whoa. Derfor ved den også at du må mene whoami.

Prøv nu at skrive "a" og tryk på tabulator tasten en gang. Der sker ingenting ud over at bash siger et bip. Prøv så at trykke tabulator tasten igen. Nu bliver der skrevet en masse output til din terminal. I stedet for at lave tab completion har bash konstateret at du har en masse kommandoer på dit system der starter med a. Ved at trykke på tab to gange får du alle mulighederne skrevet ud. Et tip er altså at når du konstaterer at tab completion ikke virker når du trykker tab een gang, så er det fordi bash har flere muligheder at vælge imellem. Prøv så derfor at trykke tab to gange i de tilfælde for at få vist de muligheder du har.
Tab completion virker også med stier og filnavne. Prøv at skrive "cd /u" og tryk tab. Du vil se at bash laver stien i kommandoen komplet til "cd /usr/", og her virker dobbelt tab skam også iøvrigt. Tryk tab igen for at få vist hvilke underfoldere der er til /usr/ som du kan gå ind i.
Væn dig til at bruge tab completion. Man kan ikke leve med kommandolinje linux uden.

Artiklen begynder nu at blive lidt rodet og opslagsagtig. Det er en gennemgang af nogle centrale ting som du måske ikke har brug for nu og her, men helt sikkert vil få på et tidspunkt. Gennemgå disse små tutorials og du vil hurtigt være på vej til at have bedre kontrol over din Linux maskine.

At se filer på sin harddisk
Ingen introduktion til bash ville være komplet uden et par ord om list kommandoen. Det er helt sikkert den allermest brugte kommando i Linux.
De fleste ved godt at man kan liste sine filer i den folder man står i med "ls". Men det er ikke alle der kender til de switches man kan bruge i den forbindelse. Feks vil "ls -l" vise indholdet af folderen i langt format. Du får hermed redigeringsdato og rettigheder på filen vist også.
Nogle filer i Linux er skjulte. Den måde man laver skjulte filer i Linux på er ved at lade filnavnet starte med et punktum. I din home folder er der en hel række skjulte filer som indeholder indstillingerne for dine programmer. På en måde er dette analogt til registreringsdatabasen i Windows. Hvis du ikke står i din home folder ligenu så kan du komme derhen ved at skrive "cd ~" og for at se de skjulte filer i den kan du skrive "ls -a". Filerne er selvfølgelig skjulte fordi du skal være varsom med at slette og redigere i dem, så pas på hvad du gør.
Læg mærke til at du frit kan kombinere de forskellige switches til ls kommandoen. "ls -al" viser feks filer og skjulte filer i langt format.

At finde filer på sin harddisk.
Sommetider har man behov for at finde ud af hvor en bestemt fil som man kender navnet på ligger henne i filsystemet. Det er der selvfølgelig flere forskellige måder at gøre på, også i bash. En effektiv måde, især for begynderen, er at installere den pakke der hedder slocate. Denne pakke er meget populær, så den følger helt sikkert med din distribution, selv om den måske ikke er med i standard installationen. Efter at du har installeret slocate har du to nye kommandoer, nemlig "updatedb" og "locate".
Du bliver nødt til at køre updatedb med det samme. Den vil gennemsøge hele din harddisk og lave en database over alle filnavne og placeringer af filer. Det tager lidt tid. Herefter kan du bruge feks "locate inittab" og i output vil du få blandt andet placeringen af den fil på din harddisk der hedder inittab.
Du skal være opmærksom på at locate kun kan finde de filer der var på harddisk sidste gang updatedb blev kørt. Hvis en fil er kommet til sidenhen så kan locate ikke finde den. Det er godt at køre updatedb en gang om ugen, eller måske endda en gang dagligt.
Vær også opmærksom på at i nogle distributioner hedder pakken ikke længere slocate. I seneste versioner af SuSe Linux hedder den findutils-locate, så både findutils og findutils-locate pakkerne skal installeres i SuSe. Din distribution kan også variere.

At afgøre hvilken type fil
Sommetider kan man sagtens komme støde på en fil på sin harddisk som man ikke aner hvad er. Feks kan jeg ved et kig i min home folder konstatere at der ligger en fil som hedder fdipscrt dateret den 18/2-2004. Er det et billede? Et script? En compileret C fil? Jeg aner det ikke. Kan ikke huske hvor den kommer fra, men til rodehoveder med dårlig hukommelse som mig findes der heldigvis en kommando som hedder "file". Den kan genkende en masse forskellige filformater. Så jeg prøver at skrive file fdipscrt og får følgende output:

sunbox:~ # file fdipscrt
fdipscrt: Bourne-Again shell script text

Så jeg får altså oplyst at filen består af tekst, og at det drejer sig om et shell script. Nu ved jeg i det mindste hvad jeg skal åbne den med, og i en text editor kan jeg se at det drejer sig om et gammelt script jeg lavede til at konfigurere en firewall.
I mange distributioner er file med i en standard installation, men hvis det ikke virker for dig så skal du installere pakken. Den hedder også slet og ret bare "file".

Hvilke processer kører?
Når man vil se hvilke processer og services der kører på en Windows computer plejer man at trykke ctrl + alt + del og vælge at gå ind i "Windows Task Manager". Man kan få lige så meget information ud af en bash shell som man kan i Windows task manager, men først skal man lige skifte til root, så skriv allerførst "su root".
Af sikkerhedsårsager har almindelige brugere nemlig ikke ret til at se hvad der kører på maskinen af services. Det vedkommer kun den eller de som har adgang til root kontoen.
Prøv så at skrive "ps -A". Det er iøvrigt en forkortelse for process status, hvis det gør det nemmere at huske. Vær også opmærksom på at der er forskel på stort A og lille a. Det er der altid i Linux, både når det gælder fil og kommandonavne, og switches til kommandoer.
De 2 vigtigste ting i output af ps -A er nummeret ude til venstre, PID, som omtales nærmere i næste sektion, og process navnet som er ude til højre. Dette output er måske ikke specielt imponerende ved siden af task manager i Windows, men så prøv i stedet "ps -aux". Her er der afgjort mere syn for sagen, og man får oplyst bla hvilken bruger der har startet processen, hvor meget RAM den tager, hvornår den er startet, og den fulde sti til den eksekverbare fil.
Output af ps er statisk som du nok kan se. Hvis du vil have et dynamisk opdateret billede af hvordan processerne på din computer kører så brug i stedet kommandoen "top". Det ville være at gå for vidt at beskrive denne kommando til bunds her. Læs i stedet man siden. Og brug så iøvrigt "q" eller ctrl + c for at komme fra top og tilbage til bash prompt.

At stoppe en process
Vi kan selvfølgelig også stoppe processer på vores maskine, og jeg vil her give en praktisk øvelse i at gøre det.
Du skal bruge 2 bash shells nu som root. Så åbn 2 terminaler og su til root.
I den ene terminal skriver du nu "cat /dev/urandom", og du vil lægge mærke til at en masse intetsigende tegn scroller igennem din terminal meget hurtigt, og muligvis bipper din PC speaker også helt afsindigt. Skynd dig hellere at trykke ctrl + c for at stoppe det inden du bliver vanvittig, for denne kommando vil fortsætte i det uendelige.
Nu er det selvfølgelig godt nok at du kan stoppe processer du har startet interaktivt med ctrl + c, men hvad så hvis der er en anden der er logget ind på maskinen udefra og gør det samme uden du har adgang til vedkommendes terminal? For at simulere det engang starter vi processen i baggrunden ved at sætte et & bagefter den. Så kan du nemlig ikke længere afbryde den med ctrl + c.
Skriv så nu kommandoen "cat /dev/urandom &" og sådan ca det samme som før vil ske. Nu skal du stoppe processen, og det gør du ved at gå til din anden terminal og først skrive "ps -A". I output skulle du gerne kunne se at du har en process der hedder "cat" til at køre. Her hos mig står der "1561 pts/0 00:00:00 cat". Nummeret ude til venstre er processens ID, eller "PID", og det er dette vi skal stoppe. Siden PID for den process jeg vil stoppe er 1561 skriver jeg simpelthen "kill 1561", og så stopper larmen.
Nu sidder du måske og undrer dig over hvorfor vi skulle igennem denne besværlige forestilling med at finde PID og dernæst skrive kill pid. Hvorfor ikke bare skrive "kill cat"? Grunden til at vi ikke kan stoppe processen ved navn er at der er flere processer der kan have det samme navn. Det duer ikke at når vi vil stoppe en enkelt process kommer til at stoppe en masse.

At se indholdet af tekst filer
Nu har du fået lov at snuse lidt til cat, og den kan mange andre ting end at skrive tilfældige tegn ud på din skærm. Kommandoen cat betyder egentlig concatenation. Den er enormt nyttig hvis du bare hurtigt vil se indholdet af en af dine konfigurations filer feks.

Prøv at skrive "cat /etc/fstab" og du vil med det samme få indholdet af din fstab fil skrevet ud til bash, uden at skulle åbne tekst editor eller noget i den stil. Kommandoen er meget nyttig til at snuse rundt med. Meget information er i linux gemt som regulær tekst, så feks "cat /etc/passwd" vil give dig en liste over brugerkonti på maskinen.
Hvis du vil se indholdet af en stor tekst fil og gerne vil scrolle lidt op og ned i den så er cat ikke særlig ideel. Det er en kommando som "less" mere egnet til. Prøv engang at sammenligne "cat /etc/inittab" med "less /etc/inittab". Som altid tryk "q" for at komme ud af less.
Nogle gange vil man også bare se de sidste linjer i en tekst fil. Feks har man ikke lyst til at læse en 10mb log fil igennem med hverken cat eller less, hvis man bare vil se de seneste 20 linjer der er skrevet i den. Det kan vi bruge kommandoen "tail" til. Din systemlog ligger i /var/log/messages, og den kan godt med tiden gå hen og blive ret lang, så hvis vi kun vil have de sidste system beskeder kan vi skrive "tail /var/log/messages", og tail vil så give os de sidste 10 linjer i system messages loggen. Hvis vi gerne vil se mere end 10 linjer kan vi bruge -n switchen. Feks vil "tail -n20 /var/log/messages" give os de sidste 20 linjer af loggen. Der er en søster kommando til tail, og den hedder som du nok kan gætte "head", og gør det samme, men bare med toppen af filen.

At søge igennem filer for et bestemt ord
Måske du en dag leder desperat rundt i dine konfigurationsfiler fordi du har installeret en ny pakke, eller fordi et eller andet pludselig er holdt op med at virke, og du mistænker at en konfigurationsfil skal ændres. Konfigurations filer i Linux ligger typisk i /etc samt underfoldere. Også selv om kommandoen ligger i /bin/. Hvis vi feks havde brug for en dag at finde alle konfigurationsfiler hvor der står noget med "eth0" i så kunne vi gøre det på følgende måde: "grep -ri 'eth0' /etc". Grep kommandoen er til at søge igennem filer for et specielt søgeord. I eksemplet søger vi efter eth0. De to switches er henholdsvis -r som betyder recursive, og som fortæller grep at den også skal kigge i underfoldere til /etc, samt -i som står for ignore og fortæller grep at den skal ignorere forskellen mellem små og store bogstaver. Så vil grep også finde feks ETH0, og Eth0 og lignende frem.
Grep kan bruges på uendeligt mange måder. Hvis vi feks vil vide hvor mange gange et ip figurerer i vores Apache log fil så vil "grep '153.35.95.234' /var/log/apache" fortælle det.

At pakke og udpakke filer
Måske du har lagt mærke til at pakkede linux filer ofte kommer med endelsen .tar.gz, og sommetider også som .tgz. Disse filer kaldes "tarballs".
En tar fil er et "tape archive", og en gz fil er en fil der er komprimeret med GNU zip. Så ligesom du måske er vant til at filer der slutter med .zip er filer der skal udpakkes med Winzip, så skal .tar.gz filer altså pakkes ud med tar kommandoen.
Følgende vil udpakke en .tar.gz fil i den folder du står i: "tar -zxvf minfil.tar.gz"
At bruge en -z switch i tar er lidt af en snydekommando, fordi i reglen er det der sker at først bliver filen dekomprimeret med GNU unzip (gunzip), og dernæst pakkes arkivet ud.
Alternativt kunne du også først sige: "gunzip minfil.tar.gz" hvorefter du ville have en dekomprimeret tar fil, og den ville du kunne pakke ud med kommandoen "tar -xvf minfil.tar".
For at pakke en eller flere filer sammen i et tar arkiv kunne du bruge følgende: tar -cfv minfil.tar ./
Switchen x til tar står for extract, og c står for compress. I ovenstående eksempel bruger jeg et punktum før /, og dette er ikke tilfældigt og det er heller ikke en typo. Det kræver lidt forklaring:
Måske har du lagt mærke til at i output til ls -a er der altid to foldere der hedder . og ..
Et enkelt punktum betyder "denne folder", og de to punktummer betyder "folderen som er eet trin oppe i hierarkiet". Med ovenstående kommando fortæller vi tar at vi ønsker at pakke alt hvad der ligger i den nuværende folder. Vi kunne også have brugt "tar -cfv minfil.tar ./nisse/" hvilket havde fortalt tar at den skal pakke alt hvad der ligger i den underfolder der hedder nisse, og tar vil regne med at nisse folderen ligger i den folder vi står i på det tidspunkt kommandoen bliver eksekveret.
Det der med at den nuværende folder er repræsenteret med et punktum er ret vigtigt i andre sammenhænge også. Hvis vi har en eksekverbar fil til at ligge et sted hvor vi ellers ikke kan eksekvere filer, som feks i vores home directory, så kan vi fortælle bash at den skal køre den eksekverbare fil runme i denne folder ved blot at skrive "./runme". Normalt kan bash kun se eksekverbare filer som ligger i de stier der er specificeret i PATH variablen. Du kan skrive "echo $PATH" for at se hvilke foldere det drejer sig om.

Pipelining
Til sidst vil jeg nævne det at kæde forskellige kommandoer sammen. Dette er ganske vist et avanceret område, men demonstrerer kraften og nytten i en bash shell.
Pipelining går ud på at man kan omdirigere output fra et program til input i et andet program. Kort og godt.
Jeg vil demonstrere det med 2 kommandoer som vi allerede kender, nemlig ved at kæde ls kommandoen sammen med less kommandoen ved hjælp af pipelining. Det vi vil er at output af ls skal hentes ind i less så vi har mulighed for at scrolle op og ned i output af ls.
Skriv først "cd /bin" så du er i et directory af en rimelig størrelse. Dette er blot fordi det tjener eksemplet bedre. Skriv nu "ls -l" og du vil kunne se en række filer i langt format. Vi vil gerne have at output af ls -l bliver vist med less, så vi skal på en eller anden måde have dirigeret output fra ls over til input af less. Det kan vi gøre ved at skrive "ls -l | less". Nu kan du scrolle op og ned med piletasterne fordi output bliver vist med less.

Pipelining, at overføre output fra en ting til input af en anden, laves altså med tegnet |.

Der er næsten ubegrænsede muligheder i pipelining. Måske du kan gætte hvad følgende kommandoer gør?

cat /var/log/messages | grep kernel
ps aux | grep bash
ls -l /dev | grep hd | less
env | grep PATH

Jeg regner med at udvide denne artikel på sigt. Ønsker og ideer til opdateringer af artiklen modtages gerne. Håber at du fandt den brugbar.

Kommentar af nanoq d. 01. Jun 2004 | 1

En absolut god artikel. Grunden til jeg ikke giver maximal karakter er, at den ikke er sat læsevenligt op. Benyt de muligheder for formattering artikelsystemet tilbyder. :)

Kommentar af ellert d. 02. Jun 2004 | 2

Virkelig god artikel! Men jeg kunne godt tænke mig, som nanoq, lidt mere ynde..

Kommentar af the_email d. 01. Jun 2004 | 3

Super artikel. Meget forklarende og vejledende. Den består mest af basalviden så en garvet linux-bruger vil nok ikke få det store ud af den, men den er perfekt til beyndere og folk som er kommet igang kan også lærer mange gode ting fra den. Måske kan de ikke lære noget nyt, men de kan lærer en smartere/hurtigere måde at gøre det på.

Kommentar af thorvall d. 18. Feb 2005 | 4

Kommentar af mrmox2 d. 12. Aug 2004 | 5

Jeg har læst mange how-tos rundt omkring, og det her er absolut en af de bedre. rigtig god "komme-igang" ting. og dejlig nørdet (ja undskyld strych9, men det er på den fede måde).

Kommentar af skovenborg d. 01. Jun 2004 | 6

Rigtig fin artikel. Gennemgår på en forståelig og forklarende måde en masse af de mest brugte og brugbare kommandoer. Hvis man ikke kan få nok i denne artikel kan jeg lige anbefale http://www.linuxbog.dk/unix/bog/index.html.php

Kommentar af blackadder d. 25. Jun 2004 | 7

Glimrende artikel

Kommentar af 123maka d. 16. Jun 2004 | 8

God artikel :)

Kommentar af mads.peder d. 01. Jun 2004 | 9

Dejlig artikel, en god hjælp for en nybegynder i Linux som jeg :-)

Kommentar af beguze d. 30. Sep 2006 | 10

Kommentar af kennethhb d. 16. Mar 2005 | 11

den er ok, og de områder du taler omkring dækker du ret godt (og har lært nogle brugbare ting, er smånoob på området) men stadig, der mangler altså noget, lidt omkring at kopiere/slette filer og oprette mapper osv

Kommentar af squashguy d. 01. Jun 2004 | 12

Helt okay. Forslag til udvidelse: du kommer ind på piping, så synes jeg en beskrivelse af omdirigering af in/output ville være en naturlig efterfølger.

Kommentar af sandbox d. 13. Jun 2004 | 13

God og velskrevet introduktion.

Kommentar af frol d. 06. Jan 2005 | 14

Fin artikel! Hvornår kommer fortsættelsen? :o)
Windows brugere kan måske også få glæde af den hvis de downloader 'GNU BASH For Windows' fra http://www.steve.org.uk/Software/bash/ og/eller 'GNU utilities for Win32' fra http://unxutils.sourceforge.net/

Kommentar af debila d. 13. Dec 2004 | 15

Den bedste artikel om komandolinien jeg har set,og brugbar

Kommentar af gentlebug d. 27. Jul 2007 | 16

Superb.

Kommentar af kristianiversen d. 10. Dec 2006 | 17

Super artikel! Lige hvad jeg manglede for at forstå shell ordenligt!