Linux op de commandoregel

Johan Swenker

De **CLI** is de *Command Line Interface* en de **GUI** is de *Graphical User Interface*. Onder zowel Windows als Linux is GUI de standaard omgeving, waar de meeste mensen in werken. Voor veel activiteiten, denk aan browsen op het internet, werkt dit prima. Maar voor sommige andere activiteiten is de CLI handiger. Maar dan moet je er wel een gevoel voor krijgen.

Zelf combineer ik de CLI vaak met de GUI. De GUI is ideaal om vier tekstschermen naast elkaar te tonen. De GUI biedt ook prima mogelijkheden om met de muis in het ene window opdrachten te knippen en die in het andere te plakken.
Verder is de CLI de basis-interface als je via SSH of een terminal verbinding maakt met een andere computer. De Raspberry Pi die ik thuis als fileserver gebruik, heeft geen beeldscherm. Ik moet dus wel de CLI gebruiken, en uiteraard het beeldscherm van de pc waarop ik al mijn werkzaamheden uitvoer. Je kunt de CLI ook gebruiken om grafische programma's op te starten.
Als je een grafisch programma start via het menu, en er gaat iets mis, dan krijg je in het beste geval een pop-up waarin staat dat het niet lukt. Maar meestal krijg je geen feedback. Als je hetzelfde programma opstart via de CLI, gewoon de naam intypen gevolgd door enter, dan heb je iets meer kans op bruikbare foutmeldingen.
Om de een of andere reden start ik Gimp, Evince en LibreOffice altijd op vanuit de CLI.

In dit eerste artikel zullen enkele nuttige opdrachten voorbijkomen die op de CLI gegeven worden.
Je kunt een en ander uitproberen met een bootable Linux CD, zoals die van Linux Mint, te downloaden van
http://www.linuxmint.com/edition.php?id=203[1]
Sommige commando's kun je ook uitproberen in de Javascript PC Emulator van Fabrice Bellard.
Dit is een volledige Linux die in je browser draait en die te vinden is op http://bellard.org/jslinux/[2].
Veel van de voorbeelden werken ook onder Cygwin, zie https://cygwin.org/.

Xterm

Maar hoe komen we bij die command line? Het klassieke Unix had daarvoor het programma xterm.
Moderne Linux-versies hebben vele andere terminalprogramma's aan boord. Zo'n terminalprogramma is vanuit het menu op te roepen. Het icoontje waar je naar moet zoeken is dat van een beeldschermpje met letters.

Bash

De standaard CLI van Linux is `Bash`. `Bash` is een zogenaamde shell waar je opdrachten aan geeft. Het heeft dus dezelfde functie als `cmd.exe` of, beter nog, de powershell in Windows.
Alle opdrachten moet je afsluiten met de entertoets. De shell zorgt er dan voor dat Linux de opdracht gaat uitvoeren. Met een prompt geeft `Bash` aan dat je een nieuwe opdracht kunt geven.

De prompt kun je helemaal naar je eigen smaak instellen, de standaard is `gebruikersnaam@machinenaam:map$` en die dollar wordt vervangen door een hekje (#) als je root bent.

Je kunt ook een rijtje opdrachten in een bestandje zetten en dat bestandje uitvoeren, wat tot gevolg heeft dat alle opdrachten uit dat bestandje uitgevoerd worden. Er zijn veel meer shells. De meest klassieke is de Bourne shell `'sh'`.
In dit artikel zullen we `Bash (Bourne Again SHell)` gebruiken.

History

`Bash` houdt een lijst bij van opdrachten die je gegeven hebt. In die lijst kun je zoeken met de pijltjestoetsen en met `Ctrl+R` voor een echte zoekopdracht. Maar je kunt de lijst ook op je scherm krijgen. Hiervoor gebruik je de opdracht `'history'`[3].

Ik heb de opdracht `history` gebruikt om een lijst te maken van alle opdrachten. Daarna heb ik de computer gevraagd mij te vertellen welk commando vaak in die lijst voorkomt en welk commando niet zo vaak.[4] En zo geeft dit artikel een heel persoonlijke smaak van opdrachten die je op de CLI kunt geven.

ls

Veruit het meest gebruikte commando bleek `'ls'` te zijn.
Met `ls` kun je een directory-listing krijgen.
Het commando is vergelijkbaar met 'DIR' in de DOS-box.

`ls` heeft verschillende opties.
Ik gebruik `ls` meestal in de volgende vorm `'ls -ltr'`:
* de `'-'` kondigt aan dat er een rij opties komt;
* de optie `'l'` geeft de lange vorm, d.w.z. inclusief datum en omvang van het bestand;
* de optie `'t'` zorgt dat de uitvoer op tijd gesorteerd wordt;
* de optie `'r'` keert de volgorde om. Het gevolg is dat het nieuwste bestand, dus het bestand waar ik mee bezig ben, onderaan staat.
  Informatie over oudere bestanden: die is van het scherm gescrold.

Als ik opruiming houd op mijn disk, en daarbij eigenlijk alleen geïnteresseerd ben in de grote bestanden,
dan gebruik ik `'ls -lSrh'`:
* de optie `'S'` zorgt ervoor dat de uitvoer op grootte gesorteerd wordt;
* de optie `'h'` staat voor `'human'` en zorgt ervoor dat de grootte van het bestand voor mensen te lezen is.

Het gevolg is dat de ISO-images, de films en andere grote bestanden netjes op mijn scherm staan, terwijl de kleinere bestanden van het scherm af gescrold zijn.
Bovendien is 1.4G veel makkelijker te interpreteren dan 1466922644.
Als ik even niet meer weet welke opties een commando heeft, dan probeer ik een van de standaarden om hulp te krijgen.
Bij `ls` vertelt `'ls -?'` je heel behulpzaam dat je `'ls --help'` moet proberen voor meer informatie. Bij andere commando's moet je juist de optie `'-h'` gebruiken om korte hulp te krijgen.

less
`Less` is een pagineerprogramma om paginagewijs door bestanden te bladeren. Voorbeeld: `'less /etc/services'`.
De naam `less` is een woordgrapje. Van oudsher kent Unix het programma `'more'` voor precies dezelfde functionaliteit.
Maar 'less is more': met `less` kun je ook terugbladeren en zoeken.
Ik gebruik `less` vooral als de uitvoer van een commando zó groot is dat ik er in moet gaan bladeren. Dan geef ik het commando opnieuw, maar nu gevolgd door `' | less '`.
Voorbeeld: `'ls | less'`
De verticale streep, het pipe-symbool, zorgt ervoor dat de uitvoer van `ls` gebruikt wordt als invoer voor `less`.

* Voorwaarts zoeken gaat met `'/'`
* Achteruit zoeken gaat met `'?'`
* Naar het einde springen gaat met `'G'`
* Naar het begin springen gaat met `'g'`
* Afsluiten van less gaat met `'q'`

cat
Met het commando `'cat'` kun je losse bestanden op het scherm krijgen. Voorbeeld: `'cat /etc/resolv.conf'`.
Als je aan `cat` meer bestandsnamen geeft, dan worden alle bestanden achter elkaar geplakt. In het Engels heet dat concatenate. Daar komt de naam van dit commando vandaan.
Met wildcards, zoals `'*'` en `'?'` kun je heel makkelijk in één keer vele bestandsnamen opgeven.
Voorbeeld: `'cat gebruiksgegevens-*'` in een directory waar je elke dag gebruiksgegevens plaatst.

tac
`Tac` is een woordgrapje. `Tac` doet precies hetzelfde als `cat`, maar zet het bestand achterstevoren: de eerste regel komt als laatste en de laatste regel komt als eerste.
Ik gebruik `tac` alleen samen met `lsmod`.
`Lsmod` zet de laatst geladen kernel modules bovenaan, die zijn dus meestal van het scherm gescrold.
Met `'lsmod | tac'` krijg ik de laatste modules onderaan, waar ik ze prima kan lezen.

man
De opdracht `'man'` kwam niet erg vaak voor in mijn history. Toch is die zo nuttig, dat ik hem hier moet bespreken.
Met `man` kun je de manual van een commando opvragen.
Sommige handleidingen zijn vertaald in het Nederlands.

Op mijn, overigens Engelstalige, Ubuntu, krijg ik met `'man -L nl man'` de Nederlandstalige handleiding van het man-commando.

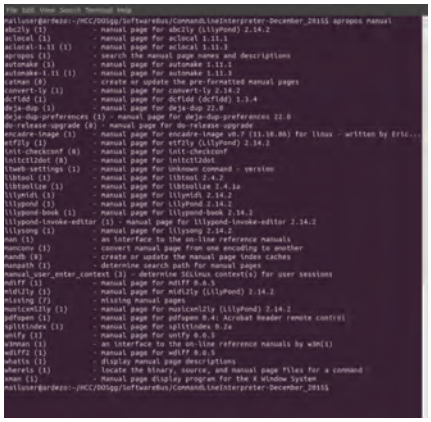
Onder water gebruikt `man` het commando `less` om de uitvoer paginagewijs op het scherm te zetten.
Bladeren, zoeken en stoppen gaat dus precies zo als bij `less`.

De handleidingen zijn verdeeld in acht secties.
Sectie 1 bevat de normale gebruikerscommando's; sectie 8 bevat de commando's voor de systeembeheerder.

Je kunt Linux-handleidingen ook als nette HTML-pagina's op internet vinden, bijvoorbeeld bij **http://linux.die.net/man/**.

info
Sommige programma's leveren hun handleiding in de vorm van info-pagina's die met het `info`-commando op te vragen zijn. Ik vind het interface van `info` zo onhandig, dat ik `info` eigenlijk nooit gebruik.

apropos
Als je niet weet welk commando je moet hebben, dan kan `apropos` soms hulp bieden.



Voorbeeld: `'apropos manual'`.
Helaas zoekt `apropos` niet al te intelligent. Bovenstaand voorbeeld vindt ook de manuals die melden `"manual page for stupid tool"`.
Maar het voorbeeld vond ook `xman`, met als korte beschrijving: `"Manual page display program for the X Window System"`

cd
Met `'cd'` kun je door de directorystructuur van het file-systeem lopen.
Met `'cd folder-naam'` ga je naar die folder toe.
Zo'n foldernaam mag een absolute naam zijn, zoals `'/etc'` of `'/var/log'`.
Het mag ook een relatieve naam zijn, zoals `..` of `mijn-subfolder`. Die subfolder moet dan wel aanwezig zijn in de huidige directory.
Met `'cd ..'` ga je één stapje omhoog, in de richting van `/`.
Een bijzondere naam is `'~'`.
Met `'cd ~'` ga je naar je de home-directory, en met `'cd ~/Bureaublad'` kom je in het bureaublad in je home-directory.
Voor een belangrijk deel is `'cd'` gelijk aan de DOS-opdracht `'CD'`.
Het grote verschil is dat bij Linux een kale `'cd'` je terugzet in je home-directory.

vi, nano, emacs, joe
Een teksteditor heb je nodig om configuratie-files aan te passen.
Klassiek Unix bevatte `vi` en is daarom op elke Unix-machine aanwezig. Linux bevat een verbeterde versie, die voor het gemak weer gewoon `vi` heet.
`Emacs` is een alternatieve editor van Richard Stallman. Er zijn vele klassieke ruzies over welke van beide beter is. Ze hebben beide één groot nadeel: ze hebben zoveel mogelijkheden dat ze voor een eerste gebruik erg lastig zijn.
De meeste beschrijvingen op internet gebruiken daarom `nano`, dat veel gebruikersvriendelijker is.

file
Met het commando `file` kun je achterhalen met wat voor soort bestand je te maken hebt. Binnen Linux is de extensie van een bestandsnaam niet relevant.
Probeer maar eens `'file /usr/bin/*'`.
Mijn systeem heeft in die map echte binaire linux-executables staan, maar ook perl-scripts, python-scripts en verschillende soorten shell-scripts.
En geen van die bestanden heeft een bijzondere extensie zoals `'.exe'`, `'.com'`, `'.bat'`, `'.py'` en `'.pl'`.

Er zijn wel standaarden. Zo verwacht ik, en met mij de file-manager van de GUI, dat **top40.mp3** een muziekbestand is.
Maar je zou een foto van een top 40-evenement best **top40.mp3** mogen noemen.
Dubbelklikken om een fotobewerkingsprogramma te openen zal dan niet meer lukken.

sudo

Uit security-overwegingen zou je alleen als een ongeprivilegeerde gebruiker moeten werken.
Stel: je geeft een verkeerd commando, of je komt op een website die je, via Javascript, malware toestuurt, dan zijn alleen je eigen bestanden kwetsbaar, en niet de systeembestanden.
Maar af en toe heb je toch wat meer rechten nodig, bijvoorbeeld bij het installeren van software. Die extra rechten krijg je met het commando `sudo`.
Voorbeeld: `'sudo make install'`. Het commando `'make install'` wordt dan met rootrechten uitgevoerd.
Maar voordat `sudo` dat doet, worden eerst een paar security-checks uitgevoerd:

- het wachtwoord van de gebruiker wordt gevraagd om er zeker van te zijn dat de gebruiker zelf achter het toetsenbord zit en
- sudo controleert of de gebruiker in het bestand `/etc/sudoers` staat.

Ik gebruik zelf regelmatig `'sudo bash'` om een shell te krijgen met rootrechten, hoewel me al eens uitgelegd is dat ik dat eigenlijk met het commando `'su'` zou moeten doen.

mc

Tijdens de CompU*fair*- en andere HCC-bijeenkomsten zie ik Linux-gebruikers die zweren bij `Midnight Commander`.
Midnight Commander lijkt op `Norton Commander` en geeft binnen de CLI iets wat heel veel lijkt op een grafische filemanager. Zelf houd ik het op ouderwetse commando's, zoals `cp`, `mv`, `rm` en `mkdir`.

* `'cp SRC DST'` kopieert het bestand SRC naar DST; dit werkt ook als DST een map is en SRC een wildcard bevat.
* `'mv SRC DST'` verplaatst het bestand SRC naar DST. Dit lijkt dus op rename, maar kan een bestand ook naar een andere map verplaatsen;
* `'rm SRC'` verwijdert het bestand SRC. En als SRC een wildcard bevat, zoals en sterretje, dan worden alle bestanden verwijderd die bij die naam passen.
  Waarschuwing: het commando `'rm *'` veegt de hele map leeg!
* `'mkdir DST'` maakt een lege directory (d.w.z. map of folder) aan met de naam DST.

Nog meer commando's

Er zijn nog veel meer commando's die ik regelmatig gebruik:

`apt-get` is bij **Debian** en de daarvan afgeleide **Ubuntu** en **Raspbian** het commando om een pakket van nieuwe commando's te installeren.
Voorbeeld: `'apt-get install locate'`
`apt-cache` vindt op basis van een paar aanwijzingen welke pakketten beschikbaar zijn, passend bij die aanwijzingen.
Voorbeeld: `'apt-cache search libreoffice pdf'`
`zypper` en `yum` zijn de vergelijkbare pakketmanagers voor **OpenSUSE** respectievelijk **Fedora**.

`locate` vindt bestanden terug op je filesysteem als je je slechts een deel van de naam van het bestand herinnert.
Voorbeeld: `'locate vakantie'` vindt alle bestanden met vakantie in de naam.

`grep` zoekt in bestanden naar stukjes tekst.
Voorbeeld: `'history |grep mount'` vindt terug hoe ik in het verleden het mount-commando gebruikt heb.

`mplayer` speelt films en muziek af.
Voorbeeld: `'mplayer *.mp3'` speelt alle mp3-files in de huidige directory af.

`ssh` gebruik je om in te loggen op een andere computer.
Voorbeeld: `'ssh -l ardezo shell.xs4all.nl'` laat me inloggen op mijn homepage bij xs4all.

`ps` geeft de processtatus weer.
Voorbeeld: `'ps axl'` geeft alle processen weer.

`df` geeft aan hoe vol de verschillende disken zijn.
Voorbeeld: `'df -h'` geeft de vullingsgraad van alle disken weer in menselijk te begrijpen Mega-, Giga- of Terabytes.

`du` geeft aan hoe vol de verschillende mappen zijn.
Voorbeeld: `'du -h|sort -h'` geeft na enige tijd de volledige lijst van mappen weer, inclusief de omvang, gesorteerd op grootte, waarbij map met de meeste data onderaan staat.

`ping` doet ongeveer hetzelfde als ping onder MS Windows: het probeert of er een netwerkverbinding te maken is met een ander systeem.
Voorbeeld: `'ping -c 10 8.8.8.8'` probeert tien keer of de publieke DNS-servers van Google te bereiken zijn.

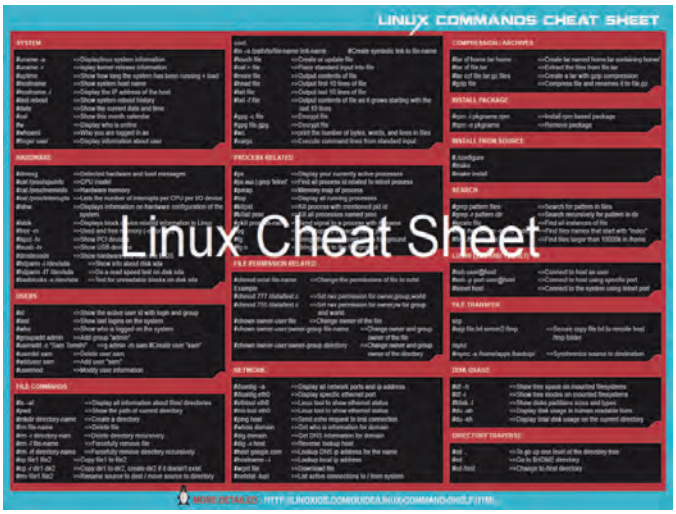
`History` gaf aan dat ik ook heel wat programmeeropdrachten geef. Meestal zijn dat oneliners: ze passen op één regel, en ik vergeet ze na gebruik.
Twee voorbeelden:

```
* 'for i in ?\ *.mp3 ; do mv "$i" 0"$i" ; done'
* 'for i in *.mp3 ; do mp3info -y 2014 -l "The
  White Album" -g "Rock" -a "The Beatles" "$i"
  ; done'
```

Over dat programmeren en wat het bovenstaande betekent, een volgende keer meer.

Ten slotte

Er zijn heel veel spiekbriefjes voor Linux te vinden op internet. Helaas zijn al die spiekbriefjes Engelstalig, en zul je dus moeten zoeken op `'cheat sheet linux'`.
Maar dan vind je ook een wereld aan beschrijvingen met betrekking tot Linux.



Specifiek voor RPi geeft **https://www.raspberrypi.org/magpi-issues/Essentials_Bash_v1.pdf**
een leuke Engelstalige pdf waarin de CLI uitgebreid beschreven wordt.

Noten

1 Kies de torrent of een mirror uit Nederland of omliggende landen.
2 Liefhebbers van dit soort gekke dingen kunnen hun hart ophalen bij **http://hackaday.com/2015/09/28/roundup-retro-computers-in-your-browser/** Daar vond ik onder andere een verwijzing naar **http://pdp11.aiju.de/**, een Version 6 Unix-emulator.
3 Tekst die geformateerd is als 'text', kun je als opdracht aan bash geven.
4 `history | while read nr opdracht rest ; do echo "$opdracht" ; done | sort | uniq -c | sort -n`