Korte uitleg: Handige gereedschappen bij shell scripts.

Bij het schrijven van shell scrips (en we nemen `bash` als voorbeeld), blijkt dat aan de ene kant veel programmeertaal constructies zoals if ... then ... else en loops, beschikbaar zijn, maar veel hulproutines, zoals andere talen die ter beschikking stellen, ontbreken. Dat komt omdat allerlei handige akties door losse programma's (gereedschappen/tools) kunnen worden gedaan. Dit alles volgens het Unix principe: maak geen honderdkoppige monsters, maar maak een tool dat één ding kan en dan heel erg goed kan.

Het nadeel hiervan is dat die tools niet in de man pagina van `bash` staan. Ze hebben natuurlijk wel ieder zelf een man pagina, maar hoe weet je dat ze bestaan en waar ze goed voor zijn. Daarom wil ik hier een lijstje presenteren van tools en waar ze handig voor zijn. Je moet daarbij bedenken dat veel van deze tools bij verstek lezen van stdin en uitvoeren naar stdout. Ze zijn dus bij uitstek geschikt om met pipes aan elkaar te koppelen. Ik zal daar ook voorbeelden van geven.

Vergeet niet om de `man` pagina van een tool te bestuderen. Ik geeft hieronder alleen hints. Je zult zelf de details moeten uitvinden.

SHELL BUILTIN COMMANDS

Deze zijn eigenlijk niet het onderwerp van deze *Korte uitleg* omdat ze in de `man` pagina van `bash` zijn te vinden. Maar bekijk die lijst dan ook eens. Er zitten o.a. de commando's in om van de terminal (stdin) te lezen (`read` en `readarray`) en er (stdout) naar te schrijven (`echo` en `printf`). En met redirect en pipes zijn dat dus de meest gebruikte commando's om gegevens naar en van je script te sturen.

`grep`

Leest een aantal regels tekst en filtert deze op inhoud.

Voorbeeld: om uit de mounttabel alleen die regels te filteren die een mount van een echt mass-storage apparaat geven:

```
henk@boven:~> mount | grep '^/dev'
/dev/sda2 on / type ext4 (rw,relatime,data=ordered)
/dev/sda3 on /home type ext4 (rw,relatime,data=ordered)
henk@boven:~>
```

Let op het "pattern" dat hier wordt gebruikt. Het wordt uitgebreid beschreven in `man grep` onder Regular Expressions. Veel mogelijkheden, en niet altijd makkelijk te overzien. Ook is het het beste om het door quoting van shell interpretatie af te schermen.

Nog een combinatie met een `bash` loop:

```
henk@boven:~> mount | grep '^/dev' | while read DEV X MP X; do echo $MP "ligt op" $DEV;
/ ligt op /dev/sda2
/home ligt op /dev/sda3
henk@boven:~>
```

`tr`

Vervangt of verwijdert tekens uit text:

```
henk@boven:~> mount | grep '^/dev' | tr -d '()' | tr ',' ' '
/dev/sda2 on / type ext4 rw relatime data=ordered
/dev/sda3 on /home type ext4 rw relatime data=ordered
henk@boven:~
```

`cut`

Knipt stukjes uit regels en laat die zien. Het volgende laat de velden 3 en 5 zien waarbij de velden gescheiden zijn door spaties:

```
henk@boven:~/test> mount |  grep '^/dev' | cut -d ' ' -f 3,5
/ ext4
/home ext4
henk@boven:~/test>
```

`head` / `tail`

Laat het begin (`head`) of het eind (`tail`) van een aantal regels zien. Het aantal kun je opgeven, dus voor de laatste vijf regels van `dmesg`:

```
henk@boven:~/test> dmesg | tail -n 5
[   50.896609] Bluetooth: BNEP filters: protocol multicast
[   50.896628] Bluetooth: BNEP socket layer initialized
[   60.757940] fuse init (API version 7.22)
[  599.860654] perf samples too long (2506 > 2500), lowering kernel.perf_event_max_samp
[ 3367.565707] perf samples too long (5003 > 5000), lowering kernel.perf_event_max_samp
henk@boven:~/test>
```

Handig is ook een `+` voor het aantal regels. Dat geeft de eerste regel van waar tot het eind wordt doorgegeven. Daarmee kun je dus de kopregel(s) uit een lijst verwijderen:

```
henk@boven:~/test> ps
  PID TTY          TIME CMD
23087 pts/2    00:00:00 bash
23674 pts/2    00:00:00 ps
henk@boven:~/test> ps | tail -n +2
23087 pts/2    00:00:00 bash
23677 pts/2    00:00:00 ps
23678 pts/2    00:00:00 tail
```

```
henk@boven:~/test>
```

wc

Telt het aantal regels, woorden en tekens dat langskomt. Dus:

```
henk@boven:~/test> wc -l /etc/passwd
35 /etc/passwd
henk@boven:~/test>
```

laat zien dat er 35 gebruikers in `/etc/passwd` zijn geconfigureerd.

sed

De stream editor. In tegenstelling tot een interactieve editor voert deze editor de opgegeven wijzigingscommando's uit op de langskomende regels.

```
henk@boven:~/test> sed -e 's/#.*$//' -e '/^[       ]*$/d' /etc/resolv.conf
search xs4all.nl
nameserver 194.109.6.66
nameserver 194.109.9.99
nameserver 194.109.104.104
henk@boven:~/test>
```

Hier zijn twee `sed` akties (aangegeven door `-e`):

1. In iedere regel worden alle strings die beginnen met een `#` tot aan het eind van de regel weggegooid.
2. Iedere regel die alleen spaties en tabs bevat (er staat een spatie en een tab tussen de `[` en de `]`) wordt verwijderd.

Let op. Er zijn wel veel configuratiebestanden waarin een `#` commentaar aangeeft, maar dat is beslist niet altijd zo!

Er zijn veel mogelijkheden in `sed` om regels te selecteren en om akties op de geselecteerde regels uit te voeren. Neem de tijd om de documentatie te bekijken en dingen uit te proberen.

awk

Nog veel uitgebreider is awk. Eigenlijk is dit een complete programmeertaal om akties op regels uit te voeren. Sommigen zweren erbij omdat het "alles kan". Als je er eenmaal handig in bent is kennelijk de neiging om `awk` dan ook voor alles te gebruiken, ook in plaats van de hierboven genoemde gereedschappen. En zelf in plaats van veel shell constructies.

Een kwestie van smaak.

sort

Soms is het nodig/handig om regels te sorteren. Het programma `sort` heeft natuurlijk een logische naam en waarschijnlijk had je al uit jezelf geprobeerd of dat bestaat als je het nodig had. Maar ik vermeld het toch maar.

Behalve simpel op ASCII sorteren kan ook op velden gesorteerd worden en zijn vele combinaties mogelijk.

Handig is vaak ook de `-u` optie, die als er meer regels hetzelfde zijn, er slechts één doorlaat.

`cmp`

Vergelijkt twee bestanden en geeft de verschillen aan. In een script is het handig omdat de returncode 0 is bij gelijkheid en 1 bij ongelijkheid. Gebruik je dan de `-s` optie om de uitvoer te onderdrukken.

Er is een vergelijkbaar tool `diff`, maar dat is beter geschikt om interactief verschillen in text bestanden te vinden (bijv. waarin verschillen twee versies van een configuratie bestand).