

Linux samenvatting

Historie

- Alle moderne computer systemen stammen af van UNIX
- UNIX werd in 1969 ontworpen in de labs van AT&T Bell
- De source code word voor iedereen ter beschikking gesteld
- De source code werd voortdurende aangepast en verbeterd door anderen.
- Jaren later werd UNIX commercieel verkocht, dit terwijl de helft van de source code door anderen geschreven was ondertussen.
- Hier kwam een rechtszaak van, wat uiteindelijk in 2 versies van UNIX resulteerde:
 - Official AT&T Unix
 - Free BSD Unix
- Veel bedrijven waaronder HP, Sun en IBM maakte hun eigen Unix versie
- Het resultaat was een soep van gelijkaardige commandos en dialecten
- Richard Stallman maakte hier een einde aan dmv het GNU project
- GNU = GNU IS NOT UNIX
- De bedoeling was een gratis besturingssysteem waar iedereen aan kon meewerken
- 90s: Linux Torvalds ontwerp een nieuwe kernel en samen met de GNU Tools vormde dit **LINUX**
- Meer dan 90% van alle supercomputer werkt op Linux
- De helft van alle smartphones draait op Linux
- Miljoenen home PCs draaien op Linux
- 70% van alle webserver draaien op Linux
- Veel tablets, dvd-players, wasmachines, modems, routers, etc draaien Linux
- = **Meest gebruikte OS ter wereld**

Distributies

- Meestal open source
- Collectie van software bovenop een Linux kernel
- Deze distros bevatten verschillende dingen:
 - Server software
 - System Management
 - Documentatie
 - Desktop apps in een centrale repository
- Distros zien er vaak gelijkaardig uit
- Zijn veilig
- Zijn makkelijk te gebruiken
- Hebben vaak een specifiek doel qua werking

De meest populaire distros zijn:

- **Red Hat**
 - **Ubuntu**
 - **Debian**
 - CentOS
 - Oracle Enterprise Linux
 - Oracle Scientific Linux
 - Linux Mint
- | | |
|--|--|
| | |
| | -> Gebaseerd op Red Hat Enterprise Linux |
| | |
| | |

- Edubuntu | -> Gebaseerd op Ubuntu
- En anderen |

Rechten

- FOSS = Free Open Source Software (tegenhanger van proprietary software = merk)
- Public Domain Software = Rechten opgegeven, mogen geen rechten op verleend worden
- Freeware = geen van beiden, maar proprietary zonder kosten.
- Voorbeelden van free software:
 - GIMP
 - MYSQL
 - GCC

GNU GPL

- Copyleft principe

Iedereen die de software gebruikt moet verbetering aan de software terug delen met de community. Deze verbetering mogen niet onder eigen naam gelicenseerd of verkocht worden.

- GPLv3 software

Deze software runnen mag op vrij basis, bij het aanpassen of distribueren van de software ga je automatisch akkoord met de voorwaarden.

Als je de software lokaal aanpast en enkel daar gebruikt moet je de software niet distribueren. Bij eender welk ander gebruik zal als distributie bekeken worden moet je de aan Copy Left voorwaarden houden.

Shell: Een programma dat commando's interpreteert (De laag tussen user en kernel)

Bash: Bourne Again Shell (meest gebruikte)

De kernel: De kern van het besturingssysteem

SUDO: Super Do User (Geeft user root privileges aan sudoers)

/: Hoofdmap root

root: user root (super user)

/root: homedirectory van user root

Manual

man whois	manual opzoeken
man syslog.conf	config file manual zoeken
man syslog.d	daemon manual opzoeken
man -k	= apropos. Zoeken in man pages via string
whatis route	korte beschrijving van een manual
whereis route	locatie van de manpage
man 5 passwd	locatie binnen een manpage meteen openen
man man	manual van de manual
mandb	update van de manuals uitvoeren

Directories

pwd	huidig pad tonen (Print Working Directory)
cd	van folder veranderen (Change Directory)
cd ~ (cd)	naar je homefolder gaan
cd ..	naar de parentmap gaan
cd .	naar de huidige map gaan
cd -	naar de vorige map gaan
cd /etc	absoluut pad
cd etc	relatief pad
tab	tab completion (systeem gokt wat je bedoeld)
ls	bestanden weergeven
ls -a	bestanden weergeven, inclusief de verborgen bestanden
ls -l	long listing (=extra info zoals rechten, owner, grootte, datum)
ls -lh	human readable format (=grootte i.p.v. bytes in GB, MB, KB, etc)
mkdir	map aanmaken
mkdir -p	parent mappen aanmaken
rmdir	map verwijderen
rmdir -p	onderliggende mappen ook verwijderen
(pushd	directory op de stack zetten)
(popd	directory van de stack halen)
lspci	hardwareinformatie opvragen

Files

Alles is een bestand in Linux, ook mappen (mappen kunnen dus ook een extensie hebben)

file file33.jpg	Geeft weer wat voor een bestand het is (jpg image file)
file -s /dev/sda	Wordt gebruikt bij speciale files in bv /dev en /proc
touch file33.jpg	Maakt een bestand aan
touch -t 200505052359 file33.jpg	Maakt een bestand aan en geeft meteen een aanmaakdatum mee
rm	Bestand wissen
rm -i	Bestand wissen, maar eerst om bevestiging vragen
rm -rf	Dient om mappen met inhoud te wissen (r=recursive, f=force)
cp	Bestand kopiëren
cp -r	Recursief kopiëren, inclusief subdirs en inhoud dus.
cp file1 file2 dir1/file3 dir2/file5	Meerdere bestanden tegelijk kopiëren naar verschillende mappen
cp -p	Behoudt timestamps en permissies van het bronbestand
mv	Bestand verplaatsen (of renamen)

rename

Complexer -> 's/txt/png/' *.txt

File content

head

eerste lijnen van een bestand opvragen

tail

laatste lijnen van een bestand opvragen

cat

inhoud van een bestand opvragen (**concatenate**)

cat "Hello" > test.txt

Steekt "Hello" in test.txt

cat > winter.txt

Maakt een lege file, dan kan je pas typen (CTRL-D om te stoppen)

cat > winter.txt << EOF

Maakt een lege file, dan kan je pas typen (EOF typen om te stoppen)

cat winter.txt > cold.txt

Bestand kopiëren

tac

Hetzelfde als cat, maar output is omgekeerd

more

Laat meer zien indien de inhoud te groot is voor het scherm

less

Laat minder zien indien de inhoud te groot is voor het scherm

strings

Zoekt ASCII tekst in een bestand en geeft deze weer

File system

man hier

Manual van de hierarchy

/

Root folder

/bin

User binaries

/home/serena/bin

Binaries voor enkel die user

/sbin

System binaries, enkel voor root users

/lib

Libraries die gebruikt worden door /bin en /sbin

/lib/modules

Kernel modules

/lib32

libraries voor 32-bit architecturen (ELF files, Executable Linkable Format)

/lib64

libraries voor 64-bit architecturen

/opt

Optionele software

/boot

Boot files, /boot/grub/grub.cfg definieert het boot menu

/etc/init.d

Scripts om daemons (background processen) te starten

/etc/X11

X Windows System (of X)

/etc/skel

Skeleton map. deze bevat de default files voor nieuwe users (bv. .bashrc)

/etc/sysconfig

Configuratie bestanden (booten, harddisks, firstboot, HW config, Keymapping)

/etc/passwd

Map van alle gebruikers

/etc/shadow

Gebruikerswachtwoorden die geëncrypteerd zijn.

/etc/login.defs:

Bevat de default settings van het password dat moet ingegeven moet worden.

/etc/group

Staan alle groepen in.

/home

Alle user mappen staan hier (\$USER)

/root

Map van de root gebruiker

/srv

Data die gedeeld kan worden (FTP, Rsync, www)

/media

Camera, USB-drive en CD-ROM-drive mount points

/mnt

Tijdelijke mount points (bv geshared mappen tussen host en guest)

/tmp

Map voor tijdelijke bestanden (~RAM), is leeg na reboot.

/dev

Device bestanden (SATA, IDE, ATAPI, USB, SCSI)

/dev/tty

/dev/tty1 stelt uw terminal (CLI) voor. /dev/pts/1 (terminal in GUI)

/dev/null

Zwart gat, hier kan je bv de **stderr** naar toe sturen

/proc

Schijnbaar lege bestanden die met de kernel communiceren

/proc/interrupts

BIOS interrupts

/proc/kcore

Enkel leesbaar met debugger, stelt uw RAM geheugen voor

/sys

Hot plug devices (USB, IEEE 1394 Firewire)

/usr

Unix System Resources (read only via NFS protocol)

/usr/bin	Unix commando's
/usr/include	Algemene bestanden die dienen voor C
/usr/lib	Libraries die niet voor users of scripts dienen
/usr/local	Map waar een admin software lokaal in kan installeren
/usr/src	Kernel Source Files
/var	Centrale map voor log files
/var/log/messages	Bestand dat info bevat over de laatste gebeurtenissen
/var/cache	Cache Data voor apps
/var/spool	Printer, mail en cron spool
/var/lib	App toestand info
/var/...	PID bestanden (Process ID), file locks

Argumenten

- Voegt extra functionaliteit aan commando's toe
- Dit gebeurt door de Command Line Scan
- Na het scannen van de lijn knipt dit het commando in stukjes op
- Dit hele process heet **Shell Expansion**
- White spaces (spaties, tabs, etc) worden automatisch verwijderen
- Om dit te vermijden moet je enkele of dubbele quotes gebruiken
 - bv **echo Hallo Hallo Hallo** wordt **Hallo Hallo Hallo**
 - bv **echo "Hallo Hallo Hallo"** wordt **Hallo Hallo Hallo**
 - Alles tussen de quotes wordt als 1 argument bekeken
- Escape karakters kunnen gebruikt worden bij echo met de optie **-e**
 - bv **echo -e "Hallo\nHallo"** wordt **Hallo**
Hallo
 - **\t -> tab**

Externe/Interne commandos

- Externe commandos staan in /sbin of /bin
- Interne commandos maken deel van de shell zelf
- Met **type** kan je bepalen of het commando intern of extern is
 - Bv. **type cd** geeft als output -> cd is a shell builtin >>> INTERN
 - Bv. **type cat** geeft als output -> cat is /bin/cat >>> EXTERN
- Sommige commandos hebben zowel een interne als externe versie
- Met **-a** kan je kan zien of een commando zowel een interne als externe versie heeft
 - Bv. **-a echo** geeft als output -> echo is a shell builtin
 echo is /bin/echo
- Om expliciet het externe commando te gebruiken moet je het volledige pad intypen
- Met **which** kan je naar binaries zoeken in de \$PATH variabele
 - Bv. **which cp ls cd mkdir pwd** geeft als output ->
 - /bin/cp
 - /bin/ls
 - /usr/bin/which: no cd in (/usr/kerberos/sbin:/usr/kerberos/bin:...)
 - /bin/mkdir
 - /bin/pwd

Aliassen

- Een alias aanmaken gebeurt met het **alias** commando
 - Bv. **alias ll='ls -lh --color=auto'**
 - Bv. **alias c='clear'**
- Met aliassen kan je bepaalde opties van een commando als standaard instellen
 - Bv. **alias rm='rm -i'**
 - Vanaf nu zal het **rm** commando altijd vragen om bevestiging.
- Aliassen bekijken doe je ook met het **alias** commando
 - Bv. **alias c ll**
 - **alias c='clear'**
 - **alias ll='ls -lh --color=auto'**
- Een alias wissen doe je met **unalias**
 - Bv. **unalias rm**
- Als je wil zien of een commando als alias werkt of niet kan je de Shell Expansions bekijken
- Dit doe je met het commando **set -x** om het aan te zetten, **set +x** is om het uit te schakelen.

Control Operators

- Deze dienen om meer dan 1 commando op 1 regel te typen
- **;** commandos scheiden
- **&** commandos naar de achtergrond sturen (de shell wacht dus niet)
- **\$?** Laatste exit code van het vorige commando
 - Bv. **rm fileblablabla**
 - **echo \$?**
 - 0 -> Gelukt
 - 1 -> Niet gelukt
- **&&** Logische EN. Het tweede commando wordt enkel uitgevoerd als het eerste lukt
- **||** Logische OF. Het tweede commando wordt enkel uitgevoerd als het eerste mislukt
- **&&** en **||** **If-Then-Else**
- **#** Commentaar, alles na dit teken wordt genegeerd door de shell
- **** Karakters **escapen** -> Het eerst volgende teken is dus een speciaal teken
- **** Kan ook een **EOL** (End-Of-Line) betekenen op het einde van een echo
 - Hierdoor kan je 1 commando over verschillende regels typen

Shell Variabelen

- **\$** Bij een \$ teken gevolgd door een woord zal de shell naar de variabele van dat woord zoeken en het vervangen door de waarde van de variabele.
 - Bv. **\$USER**, **\$HOSTNAME**, **\$UID**, **\$SHELL**, **\$HOME**
 - Shell variabelen zijn hoofdlettergevoelig!!!
- Variabelen aanmaken kan je met dit commando:
 - **MyVar=555**
 - **echo \$MyVar** geeft als output -> 555
- Quotes kunnen enkel of dubbel gebruikt worden bij variabelen, maar er is een verschil!
 - Bv. **echo "\$MyVar"** geeft als output -> 555
 - Bv. **echo '\$MyVar'** geeft als output -> \$MyVar
- **set** Geeft de variabelen weer die momenteel bestaan
- **set | more** Geeft de shell functies ook weer
- **unset** Verwijderen van een variabele
- **\$PS1** Deze variabele bepaalt je shell prompt, je kan hier speciale tekens gebruiken.
 - Bv. **PS1='\u\h:\W\$'**
 - Dit geeft als prompt:

- **user@host\$**
- **\$PATH** Deze variabele bepaalt waar de shell naar commandos gaat zoeken
- **env** Doet hetzelfde als **set**, maar exclusief de variabelen die geexporteerd zijn naar child shells
- **\$SHELL** Geeft weer welke shell gebruikt wordt
- **\$LANG** Geeft weer welke locale gebruikt wordt
- **export** Shell variabelen exporteren naar child shells, omgekeerd gaat dit niet naar de parent shell.
- Variabelen concateneren met strings kan problemen opleveren, dit los je op deze manier op:
 - prefix=Super
 - **echo Hello \${prefix}man and \${prefix}girl**
 - Geeft als output -> Hello Superman and Supergirl
- Unbound variabelen zijn variabelen die (nog) niet bestaand
 - Met het **set -o nounset** commando kan je een error laten zien bij het oproepen van een onbestaande variabele.
 - **set -o nounset of set +o nounset (aan/uit)**
 - **set -u of set+u (aan/uit)**

Shell Embedding

Shells kunnen embedded zijn op de command line. Dit betekent simpelweg dat je een nieuwe shell kan gebruiken in een bestaand commando. Dit is handig als je resultaten wil combineren 1 commando.

Bv. **echo \$var!**
echo \$(var1=5;echo \$var1)
 geeft als output: 5

De variabele var1 bestaat enkel in de subshell, de waarde wordt via een echo gedrukt, de buitenste echo drukt deze dan nog eens opnieuw.

- Backticks kunnen ook gebruikt worden om shells te embedden, op die manier kan je bv voorkomen dat je bij het **cd** commando van directory wijzigt.

Bv. **echo `cd /etc ; ls -d * | grep pass`**

Je kan ook meerdere shells embedden in mekaar, dit noemt dan **nested embedding**.

Dit kan echter alleen met \$(), niet met backticks!

Verwar backticks trouwens niet met enkel quotes:

BACKTICK: `

ENKELE QUOTE: '

Shell Options

Met **echo \$-** kan je alle opties zien die momenteel actief zijn voor de shell.

Bv. **echo \$-**
himbh
set -C ; set -u
echo \$-
himuBCH

Shell History

!!	Herhaal laatste commando
!string	Herhaal laatste commando startend met de string die je getypt hebt.
history	Bekijk de command history
history -5	Bekijk de laatste 5 commandos
!5	Voer commando nummer 5 uit
CTRL-R	Zoeken in de history
\$HISTSIZE	Hoeveel commandos er in de history bijgehouden worden
\$HISTFILE	Het bestand waar de history in opgeslaan is
\$HISTFILESIZE	Bepalen hoeveel commandos er bijgehouden moeten worden

- Regex in history
 - cat file1
 - Bv. !c:s/1/2
 - cat file2

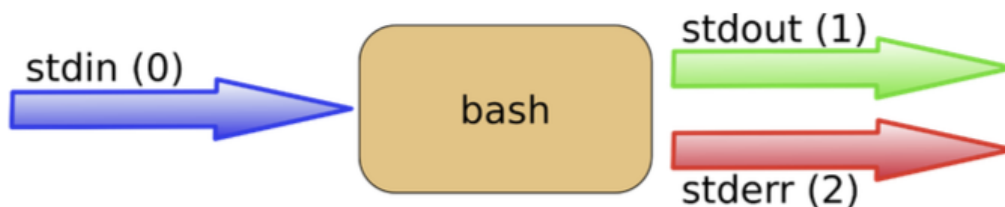
File Globbing

*	0, 1 of meer karakters	Bv.	ls File*
?	Gelijkaardig, maar enkel 1 karakter	Bv.	ls Fil?24
[]	Eender welk karakter binnen de []	Bv.	ls file[a5]
!	Logische NIET	Bv.	ls file[a5][!Z]
[a-z]	Enkel letters	Bv.	ls fil[a-z]
[0-9]	Enkel cijfers	Bv.	ls file[0-9]

File globbing kan je ook voorkomen dmv quotes of een backslash

Bv.	echo *	->	globbing
	echo *		
	echo '*'		
	echo "*"	->	geen globbing, zoekt letterlijk op *

Pipes and Commands



The keyboard often serves as **stdin**, whereas **stdout** and **stderr** both go to the display. This can be confusing to new Linux users because there is no obvious way to recognize **stdout** from **stderr**. Experienced users know that separating output from errors can be very useful.



>	output naar stdout sturen (Identiek aan 1>)
set -o	noclobber, hiermee voorkom je het per ongeluk wissen van de inhoud van een bestand door het > teken
set +o	Noclobber uitschakelen
>	Noclobber overrulen wanneer deze aanstaat
>>	Append -> output toevoegen aan file
2>	stderr bv naar /dev/null sturen
2>&1	Zowel stdout als stderr samen naar een file sturen
&>	Zelfde effect als 2>&1
<	stdin Bv. cat < test.txt Bv. tr 'onetw' 'ONEZZ' < test.txt
<<	"Here document" -> Invoer bij cat stopt bij dit woord. Bv. cat > test.txt << stop
>filename	Snel leegmaken van een bestand
> filename	Snel leegmaken van een bestand wanneer noclobber actief is

Filters

	Output doorsturen naar het volgende commando
tee	Tussenresultaten wegschrijven naar een bestand
grep	Lijnen filteren uit tekst
grep -i	case insensitive
grep -v	not-matching (NIET-functie)
grep -A1	Neemt het resultaat en de eerst volgende lijn
grep -B1	Neemt het resultaat en de lijn voordien
grep -C1	Neemt het resultaat, de lijn voordien en de lijn nadien
cut	Kolommen uit tekst nemen op basis van een delimiter Bv. cut -d: -f1,3 Bv. cut -d" " -f1 Je kan ook ook een aantal karakters uit een tekst halen: Bv. cut -c2-7 /etc/passwd
tr	translate karakters Bv. cat tennis.txt tr 'e' 'E'

	Bv. cat tennis.txt tr 'a-z' 'A-Z'	
	Bv. cat count.txt tr '\n' ' '	
	Bv. cat spaces.txt tr -s ' '	-s = squeeze spaties
	Bv. cat tennis.txt tr -d e	-d = delete karakter
wc	Word count.	
	Bv. wc -l tennis.txt	lijnen tellen
	Bv. wc -w tennis.txt	woorden tellen
	Bv. wc -c tennis.txt	karakters tellen
sort	sorteren	
	Bv. sort -k1 country.txt	sorteer op kolom 1
	Bv. sort -n -k3 country.txt	sorteer numeriek op kolom 3
uniq	duplicaten verwijderen	
	Bv. sort music.txt uniq	sorteren
	Bv. sort music.txt uniq -c	sorteren en aantal keren ervoor printen
comm	Inhoud van bestanden vergelijken	
	Bv. comm list1.txt list2.txt	
	Abba	
	Bowie	
	Cure	
	Queen	
	Sweet	
	Turner	
	Kolom1 -> Enkel list1.txt	
	Kolom2 -> Enkel list2.txt	
	Kolom3 -> Allebei	
od	text omvormen naar hexadecimale bytes	
sed	tekst aanpassen dmv regex	
	Bv. echo level5 sed 's/5/42'	-> level42
	Bv. echo level5 level7 sed 's/level/jump/g'	-> jump5 jump 7
	Bv. cat tennis.txt sed '/BE/d'	-> verwijder lijnen

Pipe Voorbeelden

who wc -l	Geeft weer hoeveel users er ingelogd zijn
who cut -d' ' -f1 sort	Een gesorteerde lijst van ingelogde users
grep bash /etc/passwd cut -d: -f1	Geeft alle bash user accounts weer

Basic Unix Tools

find /etc > etcfiles.txt	Alle bestanden in /etc en de lijst in etcfiles.txt plaatsen
find -iname	NIET hoofdlettergevoelig zoeken
find . -type f -name "*.conf"	Alle *.conf bestanden (geen mappen)
find . -name "*.conf"	Alle *.conf bestanden (incl mappen)
find /date -type d -name "*.bak"	Alle *.bak mappen
find . -newer file42.txt	Alle bestanden nieuwer dan file42.txt
find /date -name "*.odf" -exec cp {} /backup/ \;	Alle *.odf bestanden en kopiëren naar /backup
find /date -name "*.odf" -ok rm {} \;	Alle *.odf bestanden verwijderen na bevestiging
find /data -name "*.txt"	Alle txt files, ook in submappen

find /data -name *.txt

locate
updatedb
date
date +%A %d-%m-%Y
date +%s
cal
cal 2 1970
sleep
time
gzip - gunzip
zcat - zmore
bzip2 - bunzip2
bzip2 - bzip2
tar

Alle txt files in huidige map én alle files in .txt mappen

Zoekt via index = sneller
Index updaten
Datum oproepen
Saturday 17-04-2010
Seconden sinds 1969
Kalender oproepen
Kalender van Februari 1970
Aantal seconden wachten
Duur van een commando testen
Zippen/unzippen met gzip
Gzip bestanden bekijken
Zippen/unzippen met bzip2
Bzip2 bestanden bekijke n
Backup van folders bv tar -c -z -v -f
backhomefolder.tar.gz /home/
-c create new archive, -v verbose, -z compress
de archive met gzip, -f file name van het bestand

cp -r /data/*.odf /backup/

Bij miljoenen bestanden zal dit commando niet meer uitgevoerd worden. Alle filenames worden namelijk op 1 regel geplaatst. Als dit commando hierdoor te lang wordt, zal er een error volgen. De volgende oplossing werkt wel bij miljoenen files.

find /data -name "*.odf" -exec cp {} /backup/ \;

Regular Expressions

BRE	Basic Regular Expressions	
ERE	Extended Regular Expressions	-E
PRCE	Perl Regular Expression	-P

grep u names	Alle lijnen die u bevatten
grep in names	Alle lijnen die 'ia' bevatten
grep -E 'i a' list	Alle lijnen die een i of een a bevatten in BRE moet dit als: grep 'i\ a' list
grep -E 'o*' list	Alle lijnen die 0, 1 of meer o's bevatten
grep -E 'o+' list	Alle lijnen die 1 of meer o's bevatten
grep -E 'a\$' names	Alle lijnen die op a eindigen
grep -E '^F' names	Alle lijnen die met een f beginnen

Het \$ en ^ teken worden anchors genoemd in een regex

grep '\bover\b' test	Alle lijnen waar het woord 'over' in staat (geen spaties, letters, cijfers, punten, komma's, etc er voor of erna)
grep -w over text	Identiek
grep -i	Case insensitive
grep -v	Exclude
grep -w	Woord
grep -A5	5 regels erna
grep -B5	5 regels ervoor

Een regex altijd quoten, op die manier voorkom je shell expansion bij gebruik van het \$ teken

rename is een Perl script, dit commando gebruikt dus ook Perl regex

rename 's/TXT/text/' *	Alle TXT vervangen door text
rename -n 's/TXT/txt/g' aTXT.TXT	-n = laat zien wat het gaat doen ipv het effectief te doen
	s = switch
	g = global = alle occurrences
rename 's/.text/.txt/i' *	i = case insensitive

Om extensies te renamen gebruik je best het \$ teken om het einde van de string aan te geven

sed stream editor met regex

echo Sunday sed 's/Sun/Mon/'	output -> Monday
echo Sunday sed 's/Sun/##/'	output -> SunSunday
echo Sunday sed 's_\(Sun\)_\1ny_'	output -> Sunnyday
echo 2014-01-01 sed 's/....-../YYYY-MM-DD/'	output -> YYYY-MM-DD
echo 2014-04-01 sed 's/\(....\)-(..)\-(..)/\1+\2+\3/'	output -> 2014+04+01

Dit laatste (haken) noemt men **grouping**

echo -e 'today\tis\twarm'	output -> today is warm
echo -e 'today\tis\twarm' sed 's_\s_g'	output -> today is warm
cat list2 sed 's/ooo\?/A/'	? betekent optioneel
cat list2 sed 's/o\{3\}/A/'	Exact 3 o's
cat list2 sed 's/o\{2,3\}/A/'	Minimaal 2, maximaal 3 o's

VI

ESC	Command Mode
a	Typen na huidig karakter
A	Typen na huidige lijn
i	Typen voor huidig karakter
I	Typen voor huidige lijn
o	Typen op een nieuwe lijn na de huidige lijn
O	Typen op een nieuwe lijn voor de huidige lijn
x	Verwijder karakter onder de cursor
X	Verwijder karakter voor de cursor
r	Vervang karakter onder de cursor
p	Plak karakter na de cursor
xp	Verwissel twee karakters
u	Undo
.	Repeat
dd	Knip de huidige lijn
yy	Kopieer de huidige lijn
p	Plak na de huidige lijn
P	Plak voor de huidige lijn
3dd	Knip 3 lijnen
4yy	Kopieer 4 lijnen
0	Spring naar het begin van de huidige lijn
^	Spring naar het begin van de huidige lijn
\$	Spring naar het einde van de huidige lijn

d0	Verwijder tot het begin van de lijn
d\$	Verwijder tot het einde van de lijn
j	Join twee lijnen
yy	Duplicate twee lijnen
dd	Verwissel twee lijnen
w	1 woord vooruit
b	1 woord terug
3w	3 woorden vooruit
dw	Verwijder woord
yw	Kopieer woord
5yb	Kopieer 5 woorden terug
7dw	Verwijder 7 woorden
:w	Opslaan
:w naam	Opslaan met naam
:q	Quit
:wq	Opslaan en quit
ZZ	Opslaan en quit
:q!	Geforceerde quit
:w!	Geforceerd opslaan (read-only bv)
/string	Zoeken naar string
?string	Achterwaarts zoeken naar string
n	Ga naar de volgende string die hij gevonden heeft
/^string	Voorwaarts zoeken aan het begin van de lijn
/string\$	Voorwaarts zoeken aan het einde van de lijn
/br[aeio]l	Zoeken naar bral, brel, bril en brol
/\<he\>	Zoek naar he (niet here of the)
:4,8\$ s/foo/bar/g	Vervang foo met bar op lijn 4 tot en met 8
:1,\$ s/foo/bar/g	Vervang foo met bar op alle lijnen
:r fname	Lees bestand en plak de inhoud
:r !cmd	Voer commando uit en plak de output
"add	Verwijder de huidige lijn en stop deze in buffer a
"g7yy	Kopieer 7 lijnen in buffer g
"ap	Plak van buffer a
vi file1 file2 file3	Drie bestanden editen
:args	Laat bestanden zien en markeert het actieve bestand
:n	Volgend bestand editen
:e	Togglen tussen het huidige bestand en het vorige
:rew	Terug naar eerste bestand
:ab str long string	Wanneer je str intypt zal er long string komen te staan
:una str	Verwijder deze afkorting
:set number	Lijnnummering aanzetten
:set nonumber	Lijnnummering uitzetten
:syntax on	Syntax inkleuren aanzetten
:syntax off	Syntax inkleuren uitzetten
:set all	Alle opties bekijken
:set tabstop=8	Hiermee stel je het aantal kolommen van 1 tab in
:set tx	CR/LF einde
:set notx	Uitzetten

Settings opslaan voor de volgende keer doe je in ~/.vimrc voor vim en in ~/.exrc voor vi

Users

whoami: geeft de username waarmee u bent aangemeld

who: geeft een lijst van welke is aangemeld op het systeem

who am i:

w: geeft een lijst en laat zien wat ze aan het doen zijn

id: geeft uw userid, primary group id, en laat de groups zien waar jij lid van bent

su: veranderen van user bv. su tania. Je kan ook naar root user switchen de root user heeft geen wachtwoord nodig om te switchen van user.

Su - : veranderd van gebruiken en veranderd de locatie

visudo: vi voor de sudoers file

useradd: voegt een user toe. `useradd -m -d /home/yanina -c "yanina wickmayer"`
 -m geeft aan om een home folder te maken. -d is om de locatie van de homefolder in te stellen. -c (niet van groot belang) is om een descriptie toe te voegen.

usermod: veranderd de descriptie van een gebruiker -c heeft geen nut geeft gewoon wat informatie.

passwd: commando om het wachtwoord van een gebruiker te veranderen. Eerst het oud wachtwoord daarna 2x het nieuwe wachtwoord. Er zit ook een sterkte meter op het passwd. De root gebruiker heeft dit allemaal niet als die een wachtwoord wilt wijzigen van een gebruiker.

chage: wordt gebruikt om 'expire' date toe te voegen aan uw user account (-E), (-m) zet een minimum en (-M) maximum paswoord leeftijd (de leeftijd dat een paswoord mag gebruikt worden), en zet het aantal dagen voor de 'expire' date van het paswoord.

mkdir /home/laura: maak een home directorie voor de gebruiker laura aan. Als er nog geen gebruiker laura is wordt root de user van map laura.

chown: veranderd de eigenaar van een map. Bv. `chown laura:laura /home/laura`

userdel -r: verwijderd de home folder van de gebruiker samen met de gebruiker.

usermod -s /bin/bash laura: veranderd de shell van de gebruiker in dit geval wordt de shell van de gebruiker laura veranderd naar /bin/bash

usermod -L laura: Disabled het account van de gebruiker Laura. (-U om het terug te activeren)

shadow file: alle gebruikers wachtwoorden die geëncrypteerd zijn.

laura:!:17476:0:99999:7::: -> cat van de shadow file in/etc/shadow laat u het wachtwoord zien wanneer je een gebruikersaccount aanmaakt wordt dat op disabled gezet dat zie je door het !-teken.

openssl passwd: genereert verschillende afzonderlijke hashes hetzelfde wachtwoorde, hiervoor wordt een salt gebruikt. (encryptie).

openssl passwd -salt 42: salt kan gekozen worden. Nu laat hij de eerste 2 characters van de hash zien.

useradd -m -p \$(openssl passwd hunter2) Mohamed -> voorbeeld met openssl. -p laat u een gehashte paswoord ingeven.

vipw: een gepersonaliseerde vi voor de passwd file of shadow file.

Groepen

groupadd: wordt gebruikt om een gebruikersgroep aan te maken
groups laat de groepen zien waartoe een gebruiker behoort
usermod kan je ook gebruik maken om users toe te voegen aan een groep wel oppassen zonder parameters zal de user verwijderd worden uit alle groepen waar hij toe hoort. **usermod -a -G tennis inge -> -a staat voor append.**
groupmod verander de gegevens van een groep zoals groep naam. **groupmod -n darts snooker -> verander de groep naam van darts naar snooker**
groupdel verwijderd een groep bv. **groupdel tennis**
gpasswd hoofdgebruiker van een groep aanmaken. Bv **gpasswd -A serena sports -> serena kan nu mensen toevoegen en verwijderen uit de groep sports.**
gpasswd -a harry sports: nieuwe user in groep toevoegen kan nu ook door administrator serena van het vorige commando

Scripting

- Altijd beginnen met declaratie van de shell -> **#!/bin/bash --**
- **De -- dient je script tegen root-misbruik te beveiligen**
- Commentaar schrijven doe je met het # teken -> **# Voornaam Achternaam**
- Variabelen in een script bestaan enkel in het script!
- Je kan een script ook sourcen, hiermee laadt je een script in een ander script in
 - Bv. **source ./vars**
 - Ipv een script kan dit ook een .conf file zijn, hier staan dan bv al je variabelen al in gedeclareerd en geïnitieerd.
- Een script debuggen kan door je script te runnen met het **bash** commando.
- **bash -x** gebruik je shell expansion toe te passen (= zien welk commando er uitgevoerd wordt)

Scripting Loops

test

test 10 -gt 55 ; echo \$? output -> 1
[56 -gt 55] && echo true || echo false output -> true

test geeft altijd een true of false terug

if then else

```
if [ -f isit.txt ]  
then echo isit.txt exists!  
else echo isit.txt not found!  
fi
```

for loop

```
for i in 1 2 4  
do  
    echo $i  
done
```

```
for counter in `seq 1 20`  
do  
    echo counting from a to 20, now at $counter  
    sleep 1  
done
```

```
for counter in {1...20}  
do  
    echo counting from 1 to 20, now at $counter  
    sleep 1  
done
```

```
for (( counter = 1; counter <= 20; counter++ ))  
do  
    echo counting from 1 to 20, now at $counter  
    sleep 1  
done
```

while loop

```
i=100;  
while [ $i -ge 0 ]  
do  
    echo Counting down from 100 to 0, now at $i;  
    let i--;  
done
```

until loop

```
let i=100;  
until [ $i -le 0 ]  
do  
    echo Counting down from 100 to 1, now at $i;  
    let i++;  
done
```


Scripting Parameters

Een bash shell script kan parameters hebben. Deze worden in het script opgeroepen dmv het \$ teken.

\$0 is de naam van het script

\$1 is het eerste argument

\$2 is het tweede argument

...

Het **shift** commando kan in een loop door deze parameters lopen.

```
if [ "$#" == "0" ]
then
    echo You have to give at least one parameter.
    exit 1
fi
while (( $# ))
do
    echo You gave me $1
    shift
done
```

Met het **read** commando kan je een gebruiker iets laten intypen (zoals de Invoer klasse bij Java).

```
echo -n Enter a number:
read number
```

Een Script kan ook script options hebben, deze verschillende tov parameters.

```
while getopts ":afz" option;
do
    case $option in
        a)
            echo received -a
        ;; f)
            echo received -f
        ;; z)
            echo received -z
        ;; *)
            echo "invalid option -$OPTARG"
        ;; esac
done
```

Je geeft vervolgens bij het runnen van het script 0, 1 of meerdere opties mee.

Bv. `./test.sh -afz`

Een option kan ook een eigen parameter hebben.

```
while getopts ":af:z" option;
do
  case $option in
    a)
      echo received -a
  ;; f)
      echo received -f with $OPTARG
  ;; z)
      echo received -z
  ;; :)
      echo "option -$OPTARG needs an argument"
  ;; *)
      echo "invalid option -$OPTARG"
  ;; esac
done
```

Je geeft vervolgens bij het runnen van het script 0, 1 of meerdere opties mee, inclusief een argument indien een optie deze nodig heeft.

Bv. `./test.sh -zaf 42`

De shell zelf heeft ook opties, met het **shopt** commando kan je aan de shell vragen of deze opties geset zijn of niet.

`shopt -q cdspell ; echo $?`

More Scripting

Met het **eval** commando kan je de waarde van een variabele, als variabele zelf gebruiken

```
answer=42
word=$answer
eval x=\$$word ; echo $x
```

eval gebruik je ook indien je een argumenten meegeeft aan een commando dat op zijn beurt in een variabele zit. Door **eval** gebruiken zal het commando lukken, anders niet.

```
lastweek='date --date="1 week ago"'
echo $lastweek          -> lukt niet
eval $lastweek          -> nu wel
Thu Mar 8 21:36:39 CET 2012
```

Door **(())** te gebruiken kan je numerieke expressies evalueren, oftewel berekeningen testen op true of false.

```
( 42 > 33 ) && echo true || echo false
```

Met het **let** commando kan je berekeningen maken.

```
let x="3 + 4" ; echo $x
```

Met een **case** kan je zoals in Java een switch/case (of selectieblok) maken in een script.

```
echo -n "What animal did you see ? "
read animal
case $animal in
    "lion" | "tiger")
        echo "You better start running fast!"
    ;; "cat")
    ;; "dog")
        echo "Let that mouse go..."
        echo "Don't worry, give it a cookie."
    ;;
    "chicken" | "goose" | "duck" )
        echo "Eggs for breakfast!"
    ;; "liger")
    ;;
    "bafelfish")
    ;; *)
    ;; esac
```

Je kan ook **functions** gebruiken in scripts.

```
#functie
function greetings {
    echo Hello World!
    echo and hello to $USER to!
}
#main
echo We will now call a function
greetings
echo The end
```