

Linux cheatsheet

uname
who
whois
pwd
date
cal

ls -d /usr/???
[gC]*
{1..10}

alias lh="ls -lh"

man passwd / man -k SHA1
info passwd
wheris bzip2
which bzip2
type echo / type ls

ls / ls -la / ls -lR / ls -ld
ln / ln -s
du -h
head / head -n-1 / head -n5
tail
cat
less
wc -l
nl
cut -d : -f 2

tr -s "<,>" "[,]"
tr -d "@"
tr -s "[:lower:]" "[:upper:]"
sort
uniq / uniq -D
touch

groupadd
useradd
usermod

grep / grep -vE '^ij.*' / grep -i
grep -E "1[0-9]{2}-[0-9]{3}" file | cut -d : -f 2
grep -E (Posix2.0 Regex ERE)
grep -P (Perl compatible Regex PCRE)

free
ps -aux
ps -fax
pstree
top
uptime
jobs

apt-get update / apt-get upgrade
apt-get install appname
dpkg -l / dpkg --search / dpkg --s

ip addr show
env / printenv

chown

chmod / chmod u=rwx,g=rx,o= /home/user /
chmod 750 /home/user

> file.txt
>> file.txt
ls -la /usr/bin &> file.txt
ls -la /usr/bin 2> file.txt
ls -la /usr/bin > file.txt 2>&1
dd if=/dev/zero of=/home/user/f4_4 bs=1024
count=2000

kill -SIGTERM 10492
kill -SIGKILL 10492
killall -SIGKILL processname
kill \$! (\$! the last PID executed in bg)
kill \$\$ (\$\$ the scripts own PID)

lsblk
fdisk -l
mkfs / mkfs.ext4 / mkfs.ntfs
mount
umount
lsof
cat /proc/partitions
df -h
mkswap
swapon
swapoff
cat /proc/swaps
/etc/fstab
dd if=/dev/zero of=/dev/sdb1 bs=1024
count=10

fsck

/dev/zero

/dev/random

/dev/null

find . -type d -name "example"

find . -type f -iname "example.*"

find . -not -type f -iname "example.*"

find . -name "abc*" ! -name "*.php"

find . -name "*.php" -o -name "*.txt"

find . -type f -mmin +1 -mmin -5

find . -size +5M

find . -empty

find . -perm 777

find example/ -type d -exec chmod 775 {} +

find . -maxdepth 1 -type f -name "*.jpg"
-exec rm {} +

tar -cvf example.tar directory/

tar -tf example.tar

tar -xvf example.tar

gzip example.tar

gunzip example.tar

bzip2 example.tar

bunzip2 example.tar

tar -cvzf example.tar.gz directory/

tar -xvzf example.tar.gz

tar -cvjf example.tar.bz2 directory/

tar -xvjf example.tar.bz2

gzip < /directory/example > example.gz

bzip2 < /directory/example > example.bz2

sha1sum file

sed "s/pattern/newpattern/g" file

sed -i "s/pattern/newpattern/g" file

sed "s/\s*#.*/g; /\\$/ d; s/^[[:space:]]*/g" file

sed "s/\s*#.*/g; /\\$/ p; s/^[[:space:]]*/g" file

sed "s/\s*#.*/g; /\\$/ q; s/^[[:space:]]*/g" file

sed "10 q" file

seq 10

ps2pdf original.pdf commressed.pdf

yui-compressor style.css > style.min.css

wget -m --user=username

--password=password ftp://ip.of.old.host

wget -m

ftp://username:password@ip.of.old.host

Regular expressions

Format

^ and \$	Start / end of a line
.	Any character
[] and [^]	Any character (not) between the brackets
?	Zero or one time previous character / expression
* and +	Zero or more / one time previous character / expression
{x,y}	Minimum x and maximum y previous character / expression
()	Group
[a-z]	Character a to z lowercase
[0-9]	Number 0 to 9
[ab]	Character a or b
[^ab]	All except character a or b
a{4}	4 times a
(ab){4}	4 times ab
a{1,4}	From 1 to 4 times a
a+	1 to n times a

a*	0 to n times a
----	----------------

Character classes

\w and \W	“word character” (a-zA-Z_) and inverse
\b and \B	“word boundary” (boundary from a word) and inverse
\s and \S	Whitespace and inverse
[[:alpha:]]	a-zA-Z
[[:digit:]]	0-9
[[:alnum:]]	a-zA-Z0-9
[[:space:]]	Space, tab, new line, return
[[:blank:]]	Space or tab
[[:lower:]]	Lowercase letter
[[:upper:]]	Uppercase letter
[[:print:]]	Printable characters
\d and ...	Not in grep: same as [[:digit:]]

Examples

KdG student numbers: [0-9]{7}-[0-9]{2}

jan.celis@student.kdg.be:

([[:alnum:]]+\.){0,2}([[:alnum:]]+@[[:alnum:]]+\.){1,3}([[:alpha:]]{2,3}

Hexadecimal number of 4 numbers: [0-9A-Fa-f]{4}

Each number containing a minimum of 3 zeros, repeated after each other:

[0-9]*0{3}[0-9]*

Word “fix” in a text, different possibilities:

[[:space:]]fix[[:space:]]

fix\W

\<fix\>

Start with <, contains @ and ends with >:

<.+@.+>

for i in *; do mv \$i `echo \$i | tr [[:upper:]] [[:lower:]]`; done

content="KdG-Hogeschool, Nationalestraat 5, B-2000 Antwerpen"

regex="B-[0-9]{4}"

if [[\$content =~ \$regex]]; then

 echo "Found postal code"; exit 0;

else

 echo "Postal code not found" >&2; exit 1;

fi

content="KdG-Hogeschool, Nationalestraat 5, B-2000 Antwerpen"

regex="([[:alpha:]]+-[a-zA-Z]+), ([[:alpha:]]+) ([[:digit:]]+), (.) (.)"

```
[[ $content =~ $regex ]]
```

```
echo "${BASH_REMATCH[0]}"
```

```
echo "${BASH_REMATCH[1]}"
```

```
echo "${BASH_REMATCH[2]}"
```

```
echo "${BASH_REMATCH[3]}"
```

```
echo "${BASH_REMATCH[4]}"
```

```
echo "${BASH_REMATCH[5]}"
```

Bash shell scripting

```
#!/bin/bash
#!/bin/bash -x (debug)
```

```
sudo apt-get update && sudo apt-get install shellcheck
mkdir -p ~/.vim/pack/git-plugins/start
git clone https://github.com/dense-analysis/ale.git
~/.vim/pack/git-plugins/start/ale
```

```
# comments
var="Hello"
export globalvar="Hello"
clear
echo -n "Enter your name: "
read name
echo -e $var,\n$name
read -p "What is your first name? " firstname
echo "${firstname^}" / echo "${firstname^^}"
echo "Er zijn `cat /etc/passwd | wc -l` users"
echo "Er zijn $(cat /etc/passwd | wc -l) users"
```

Positional parameters

```
$0 filename
$1 (1-9) arguments
$# (get amount of positional parameters)
$* and $@ (list of all parameters)
```

Quotes

Single quotes: hard quotes, print what's between them
Double quotes: soft quotes, \$ and ` will be handled escape with \
Backquotes: command substitution or use \$(..)

Calculate with + - * / %
number=\$((2+2))
let number=2+2
\$((RANDOM % 10)) (generate a random number between 0 and 10)
bc (for floating point number)

```
chmod +x script.sh
./script.sh
source script.sh
. script.sh
```

```
/bin/true (0) - /bin/false (1)
$? (exit status -- exit 113)
```

[..] or newer version [[..]] (with regular expressions =~)
-n true if next variable has a value
-z true if the string is empty
-d true if it is a directory
-f true if it is a file
-r true if it is a readable file
-w true if there are writing permissions for the file
-x true if it is an executable file
[file1 -nt file2] true if file1 is newer then file2
[file1 -ot file2] true if file1 is older than file2
-ot reversed

[number1 .. number2]
-lt less than
-le less than or equals
-eq equals
-gt greater than
-ge greater than or equals

-ne not equals

```
[ -d "$1" ] && echo "It's a directory"
[ -f "$1" ] && echo "It's a file"
[ -x "$1" ] && { echo "Not allowed with an executable"; exit 1; }
[ $(id -u) -ne 0 ] && { echo "You are not root"; exit 1; }
[[ "$int" =~ ^-[0-9]+$ ]]
[[ $content =~ $regex ]] ($regex never quoted here)
```

```
(( int == 0 )) / (( int < 0 )) / (( ( int % 2 ) == 0 ))
```

```
[[ .. ]] && [[ .. ]] / [[ .. ]] || [[ .. ]] / ! (( int == 0 ))
```

Parameter substitution

Replace 1 time kdg by student: \${url/kdg/student}

Replace all times ht by f: \${url//ht/f}

Replace first occurrence of http by ftp: \${url/#http/ftp}

Replace last occurrence of html with aspx: \${url/%html/aspx}

default_value="10"

size=\${1:-\$default_value} # of \${1:-10} #default value

size=\${1?"Usage: \$(basename \$0) ARGUMENT"} #end script if \$1 has
not been passed as argument

Length = \${#url}

Starting from 7: \${url:7}

Starting from -4: \${url: -4}" # Of \${url:(-4)}

Starting from 0 length 4: \${url:0:4}

Starting from 7 length 10: \${url:7:10}

Remove non-greedy *. from beginning: \${url#*.*}

Remove greedy *. from beginning: \${url###*.*}

Remove non-greedy .* from end: \${url%*.*}

Remove greedy .* from end: \${url%*.*}

Change to uppercase: \${url^^}

Change to lowercase: \${url,,}

Loops and statements

IFS (space / tab / newline) / IFS=\$'\n' (only newline)

```
for filename in $(find ~/ -iname '*.txt')
do
    echo $filename
done
```

```
for filename in $(ls *.tar.gz)
do
    tar xvf $filename
done
```

```
if [ -d /etc/systemd ]; then
    echo "Directory exists"
fi
```

```
if ls ~/tmp/*.tar.gz &> /dev/null; then
    cp ~/tmp/*.tar.gz .
else
    echo "Er zijn geen tar.gz-bestanden beschikbaar"
    exit 2
fi
```

```
if [ -d ~/Music ] && [ -w ~/Music ] && [ -x ~/Music ]; then
    cd ~/Music
elif [ -d ~/Documents ] && [ -w ~/Documents ] && [ -x ~/Documents ]; then
    cd ~/Documents
```

```
else
    echo "No access to a directory"
    exit 1
fi
```

cmd1 && cmd2 (execute cmd2 if cmd1 exit code 0)
cmd1 || cmd2 (execute cmd2 only if cmd1 exit code 1)

```
teller=$1
[ ! -z "$1" ] && { echo "Expecting one parameter"; exit 1 }
while [ $teller -gt 0 ]; do
    echo $teller
    sleep 1
    $((teller--))
done
```

```
case $1 in
    move)
        echo "Move"
        ;;
    copy | kopie)
        echo "Copy"
        ;;
    delete)
        echo "Delete"
        ;;
    *)
        echo "Something else"
esac
```

```
function name {
    local foo # local variable for this function
    echo "Do something with $foo"
    return
}
```

Security

```
sudo ping -i 0,01 alfred.straffesites.be
sudo tail -f /var/log/syslog
iptables -L -n -v
iptables -F
iptables -Z (see demo configuration)
ip a
sudo tcpdump -n -i wlp4s0
```

SSL self signed

```
sudo openssl genrsa -des3 -out server.key 4096
sudo openssl req -new -key server.key -out server.csr
sudo openssl x509 -req -days 365 -in server.csr -signkey server.key -out server.crt
sudo openssl req -new -x509 -extensions v3_ca -keyout cakey.pem -out cacert.pem -days 3650
sudo echo 01 > /etc/ssl/demoCA/serial
touch /etc/ssl/demoCA/index.txt
sudo openssl ca -in server.csr -config /etc/ssl/openssl.cnf
```

SSL from Let's encrypt

```
sudo apt install snapd
sudo snap install core
sudo snap refresh core
sudo snap install --classic certbot
sudo ln -s /snap/bin/certbot /usr/bin/certbot
sudo certbot --apache
```

TCP connect

Port = open

Client	SYN ->	Server
	<- SYN-ACK	
	RST-ACK ->	

Port = closed

Client	SYN ->	Server
	<- RST	

The port is open if the server sends SYN ACK; and is closed when the server sends RST.

XMAS scan, TCP with PSH,URG,FIN flags set; port is open if server doesn't answer; closed if it sends RST. Firewall in between if filtered ICMP returns.

FIN scan, TCP with FIN set; port is open when server doesn't answer; closed with RST, firewall in between if filtered ICMP returns.

TCP ACK scan, TCP with ACK set; port is unfiltered if server sends an RST. You don't know if the port is open or closed.

Tools

```
sudo apt install wireshark tcpdump nmap ettercap-graphical p0f snort  
zeek
```

```
tcpdump -n -r portscan.pcap | cut -d ' ' -f3 | cut -d '.' -f5 >  
allesourcepoorten.txt
```

Wireshark

Filter log file, for example; “tcp.flags.syn == 1 && tcp.flags.ack == 1” or
“ip.addr eq 10.10.10.1 && tcp.port eq 12345”

TCP connect → tcp.flags.reset == 1 && tcp.flags.ack == 1
NULL scan → tcp.flags.reset == 0 && tcp.flags.ack == 0 && tcp.flags.syn
== 0 && tcp.flags.fin == 0
XMAS scan → tcp.flags.push == 1 && tcp.flags.urg == 1 && tcp.flags.fin
== 1

Portscan

```
nmap 192.168.100-254 (scan for devices)  
nmap --source-port 21 -sN alfred.traffesites.be (NULL scan)  
nmap -sX alfred.traffesites.be (XMAS scan)  
nmap -D 192.168.1.5,104.215.148.63,209.33.28.4 alfred.traffesites.be  
(Decoy scan)  
nmap -r -p 1-9999 -sT alfred.traffesites.be (TCP connect ports)  
nmap -Pn alfred.traffesites.be
```

Fingerprinting

```
p0f -r portscan.pcap
```

Snort

```
sudo vi /etc/snort/snort.conf → uncomment and modify: preprocessor  
sfportscan: proto { all } scan_type { all } memcap { 10000000 } logfile {  
portscans.log } sense_level { high }
```

```
sudo snort -r portscan.pcap -c /etc/snort/snort.conf -A full  
cat /var/log/snort/alert
```

Zeek

```
zeek -r portscan.pcap policy/misc/scan.zeek
```