LINUX

Linux Commands

Shell

El intérprete de comandos ejecuta las instrucciones introducidas con el teclado o en un script y devuelve los resultados. Este intérprete es un programa comúnmente llamado ***shell***. Es una interfaz que funciona en modo texto entre el núcleo de Linux y el usuario. Este *shell* funciona en un terminal y como todo programa puede ser compilado y ejecutado en otras plataformas.

Originalmente un terminal era una verdadera máquina con una pantalla y un teclado que se conectaba a un servidor central. En la actualidad, un terminal es un programa que emula estos terminales:

- las **consolas virtuales de texto**, el modo por defecto de Linux cuando arranca sin entorno gráfico
- las **consolas o terminales gráficos** como *xterm*, *eterm* o *konsole* que son emuladores de terminal en un entorno gráfico

Hay varios *shells* como *Bourne Shell* (sh), *C-Shell* (csh), *Korn Shell* (ksh) o *Z-Shell* (zsh). El *shell* de referencia en Linux es ***Bourne Again Shell*** (bash).

```
user@UBUNTU:~$
```

- *user* es el nombre de inicio de sesión o login del usuario
- *UBUNTU* es el nombre del anfitrión (*hostname*), el nombre lógico de la máquina conectada al terminal
- ~ es el carácter que indica que se encuentra en el directorio personal
- > o *$* es la terminación estándar del bash para un usuario sin privilegios

```
# El comando 'pwd' permite saber el directorio actual
user@UBUNTU:~$ pwd
/home/user
```

Exiten algunos atajos de teclado:

- `Ctrl + a` - ir al principio de la línea
- `Ctrl + e` - ir al final de la línea
- `Ctrl + u` - borra desde el cursor hasta el principio de la línea
- `Ctrl + k` - borra desde el cursor hasta el final de la línea
- `Ctrl + l` - borrar el contenido del terminal y mostrar la línea de comandos en la parte superior

```
# Los comandos se pueden encadenar separados por `;`:
$ date; pwd; uname
```

Existen dos tipos de comandos:

- **comandos externos** que son programas binarios presentes como archivos. Al ejecutarse se cargan en memoria y se inician como proceso.
- **comandos internos** que son propios del *shell* y se ejecutan en él.

> NOTA: También hay otros tipos como los *alias* de comandos que son atajos de comandos propios del shell

El comando `type` permite distinguir los tipos de comandos:

```
# Comando interno
$ type pwd  # pwd is a shell builtin

# Comando externo
$ type date  # date is hashed (/usr/bin/date)

# Alias de comando
$ type ll  # ll is aliased to `ls -alF'
```

Ayuda

```
# Ayuda interna en el propio comando
$ date --help

# Comando de ayuda
$ help {comando}

# Manual en línea de comandos
$ man {comando}

# Manual en línea del propio manual
$ man man

# Mostrar una sección concreta para un comando
$ man passwd # Muesta la sección '1. Programas ejecutables' por defecto del comando
$ man 5 passwd # Muestra la sección '5. Formatos de archivo' del comando

# Buscar por correspondencia dada una palabra
$ man -k passwd # Muestra todos los comandos que contienen 'passwd'

# Ayuda en formato info (enlaces, info más detallada, etcétera...)
$ info {comando}
```

Secciones

El manual de ayuda en línea se compone de secciones:

1. Instrucciones ejecutables o comandos del shell
2. Llamadas del sistema (API del núcleo…)
3. Llamadas de las librerías (funciones C…)
4. Archivos especiales (contenido de /dev como sd, hd, pts, etcétera…)
5. Formato de los archivos (/etc/passwd, /etc/hosts, etcétera…)
6. Juegos, salvapantallas, programas varios, etcétera…
7. Varios, comandos no estándares que no encuentran sitio en otra parte
8. Comandos de administración del sistema Linux
9. Subprogramas del núcleo

Package Manager

```
# Update the packages repository
$ apt update

# Upgrade packages in bulk
$ apt upgrade

# Search for a package named 'htop', for example
$ apt search htop
```

```
# Show information about a package
$ apt show htop

# Install a package named htop
$ sudo apt install htop

# Remove a package named htop
$ sudo apt remove htop

# Install multiple packages, for example htop and less
$ sudo apt install htop less

# Forzar la instalación de paquetes faltantes
$ sudo apt install -f

# Otro administrador de paquetes para Debian y derivados como Ubuntu
$ sudo apt install aptitude

# Listado de paquetes instalados ordenados por tamaño
$ dpkg-query -W --showformat='${Installed-Size;10}\t${Package}\n' | sort -k1,1n

# Listado de paquetes instalados ordenados por tamaño y mostrando prioridad
$ dpkg-query -W --showformat='${Installed-Size;10}\t${Priority}\t${Package}\n' | sort -k1,1n
```

System Information

```
# Display Linux kernel information
$ uname -a

# Display kernel release information
$ uname -r

# Display distro description
$ lsb_release -d

# Show how long the system has been running + load
$ uptime

# Show system hostname
$ hostname

# Display the IP addresses of the host
$ hostname -I

# Show system reboot history
$ last reboot

# Show the current date and time
$ date

# Display who is online
$ w

# Who you are logged in as
$ whoami

# Who you are
$ id

# Ver el histórico de comandos ejecutados en la consola
$ history
$ fc -l

# Repetir un comando del histórico
$ !{number}
$ fc -s {number}
```

```
# Repetir un comando con sudo
$ sudo !!

# Visualizar el log del sistema
$ sudo -g adm more /var/log/syslog
```

Hardware Information

```
# Listar todo el hardware
$ lshw

# Listar las tarjetas PCI
$ lspci

# Ver los dispositivos conectados a un puerto USB
$ lsusb

# Display CPU information
$ cat /proc/cpuinfo

# Display number of CPU cores
$ nproc

# Display memory information
$ cat /proc/meminfo

# Display environment variables of a process, e.g: PID 1
$ cat /proc/1/environ

# Display free and used memory ( -h for human-readable, -m for MB, -g for GB.)
$ free -h
```

System Monitoring, Statistics, Debugging

```
# Display and manage the running processes
$ top

# Display a friendly interactive process viewer (alternative to top)
$ sudo apt install htop
$ htop

# Display processor related statistics (refresh every 1 second)
$ sudo apt install sysstat
$ mpstat 1

# Display virtual memory statistics (refresh every 1 second)
$ vmstat 1

# Display disk I/O statistics (refresh every 1 second)
$ iostat 1

# List all open files on the system
$ lsof

# List files opened by the user (e.g: root)
$ lsof -u {USER}

# List files opened by a certain process with PID (e.g: 1)
$ lsof -p {PID}
```

Directory Navigation

```
# Change to '/home' directory
$ cd /home

# Change to the previous directory
$ cd -

# Go up one level of the directory tree
$ cd ..

# Display the present working directory
$ pwd
```

File and Directory

```
# Mostrar la descripción de un fichero
$ file {file.ext}

# Display disk space occupied by current directory (-h for human-readable, -s summarize)
$ du -sh {folder}

# Ver el espacio en el disco
$ df -h`

# Execute “df -h”, showing periodic updates every 1 second (-d flag shows visual updates)
$ watch -n1 df -h

# List all files (including hidden) in a listing human-readable format in the current directory
$ ls -lah . # (specifying . is optional)

# Execute “ls -lah”, showing periodic updates every 1 second (-d flag shows visual updates)
$ watch -n1 ls -lah

# Create one or more new empty file
$ touch {file1} {file2}

# Create one or more new empty file with pattern
$ touch {1..10}.txt # Create 1.txt, 2.txt, 3.txt, etc...

# Create a new directory
$ mkdir {dir1}

# Create a directory tree using -p option
$ mkdir -p dir1/dir2/dir3

# List the directory tree using tree command
$ tree {dir1}

# Copy (duplicate) file(s) from one directory to another (-v option for enabling verbose mode)
$ cp -v {file1} {dir1/file1-copy}

# Copy directory and all it’s content to a new directory
$ cp -vr {dir1} {dir1-copy}

# Rename a file
$ mv -v {file1} {file1-rename}

# Move a file into directory
$ mv -v {file1} {dir1}

# Remove a file or empty directory (-f option force deletes without asking)
$ rm {file1}

# Remove a directory and its contents recursively (-v option for enabling verbose mode)
$ rm -vr {dir1}

# Create a symbolic link (pointer) to a file or directory
$ ln -s {file1} {file1-link}
```

```
# Write a simple text to a file
$ echo "hello, world!" > hello.txt

# View the contents of a file
$ cat hello.txt

# Paginate through a large file
$ less hello.txt

# Display the first 20 lines of a file
$ head -n 20 hello.txt

# Display the last 20 lines of a file
$ tail -n 20 hello.txt

# Display the last 10 lines of a file and follow the file as it updated
$ tail -f hello.txt
```

```
# Vaciar la Papelera desde el terminal
$ sudo rm -rf ~/.local/share/Trash/*

# Copiar un fichero con progreso
$ sudo rsync -ah --progress {source} {destination}
```

```
# Buscar ficheros con una extensión en concreto
$ find . -type f -name *.jpg

# Buscar y listar ficheros con una determinada extensión
$ find . -type f -name *.jpg -exec ls'{}' \;

# Buscar y borrar ficheros con confirmación
$ find . -type f -name *.jpg -exec rm -i '{}' \;
```

Montar unidades

```
# Ver las particiones del sistema, tanto montadas como no montadas
$ sudo apt install fdisk
$ sudo fdisk -l

# Comprobar si el sistema ha reconocido una unidad USB
$ dmesg

# Montar una unidad USB asignada a /dev/sdh1 por ejemplo
$ sudo mount /dev/sdh1 /path/to/folder/

# Montar una imagen ISO
$ sudo mount -o loop /path/to/disk1.iso /path/to/folder/

# Desmontar una imagen
$ sudo umount /path/to/folder/
$ sudo umount /dev/sdh1

# Para montar unidades exFat
$ sudo apt install exfat-fuse exfat-utils
```

Utilidad compresión ficheros

```
# Comprimir un directorio usando la utilidad 'zip'
$ zip -r file.zip directorio/
```

```
# Comprimir el directorio actual
$ zip -r file.zip .

# Descomprimir un fichero .zip
$ unzip file.zip

# Descomprimir un fichero .rar con la utilidad 'rar'
$ unrar e nombre_del_fichero.rar

# Descomprimir un fichero rar en una ubicación
$ unrar e nombre_del_fichero.rar /donde/lo/quieres
```

Networking

```
# How do I determine ethernet connection speed?
$ ethtool eth0

# Comprobar la señal WIFI
$ wavemon

# Display information of all available network interfaces
$ ip addr

# Display information of eth0 interface
$ ip addr show eth0

#Display IP routing table
$ ip route

# Ping a hostname or IP address
$ ping google.com
$ ping 8.8.8.8

# Display registration information of a domain
$ whois medium.com

# DNS lookup a domain:
$ dig medium.com A     # IPv4 addresses
$ dig medium.com AAAA  # IPv6 addresses
$ dig medium.com NX    # Nameservers

$ host medium.com     # IPv4 addresses

# Display hostname and IP address of the local machine
$ hostname
$ hostname -i

# Download files from a remote HTTP server
$ wget {URL}

# Descargar todos los ficheros de un directorio con wget
$ wget -r --no-parent {URL}

# Download files from a remote HTTP server
$ curl --output 5MB.zip {URL}

# Display all process listening on TCP or UDP ports
$ netstat -plunt`
```

Process Management

A **process** is a running instance of a program.

```
# Display your currently running processes
$ ps

# Display every process on the system
$ ps auxf

# Display interactive real-time view of running processes
$ top
$ htop

# Look-up process ID based on a name
pgrep {name}

# Kill a process with a given process ID. By default TERM signal is sent
$ kill PID

# Send a custom signal to a process with given process ID
$ kill -s SIGNAL_NUMBER pid

# List all available signals
$ kill -l

# Kill a process based on a name
$ pkill {name}

# Run a command as a background job
$ (sleep 30; echo "woke up after 30 seconds") &

# List background jobs
$ jobs

# Display stopped or background jobs
$ bg

# Brings the most recent background job to the foreground
$ fg

# Brings job N to the foreground
$ fg N

# Kill job N
$ kill %N
```

File Permissions

```
# Give all permission to the owner, read execute to the group and nothing to others
$ chmod 750 file1
$ chmod u=rwx,g=rx,o= file1

# Change ownership of a file or directory to a given user and group
$ chown user:group file1

# Otorgar permiso de escritura a un usuario a una carpeta
$ chown {user} {folder} -R
```

Text Search

```
# Search for a pattern in a text file
$ grep pattern file

# For example, search for a 'root' pattern in a 'passwd' file
$ grep root /etc/passwd
```

```
# Search recursively for a pattern in a text file inside a directory
$ grep -R "/bin/bash" /etc

# Search for pattern and output N lines before (B) or after (A) pattern match
$ grep -B 5 root /etc/passwd
$ grep -A 3 root /etc/passwd

# Find files within a directory with a matching filename
$ find /etc -iname 'passwd'
$ find /etc -iname 'pass*'  # glob pattern`

# Find files based on filesize
$ find / -size +1M  #Larger than 1MB
$ find / -size -1M  #smaller than 1MB
```

Pipes and Redirection

Redirection

```
# Redirect normal output (stdout) from a command to a file
$ echo "hello" > hello.stdout.txt

# Redirect error output (stderr) from a command to a file
$ cat somefile 2> cat.stderr.txt

# Redirect both normal and error output from a command to a file. Useful for logging
$ ps auxf >& processes.txt

# Append normal output (stdout) from a command to a file unlike > which overwrites the file
$ echo "hello" >> hello2.stdout.txt

# Append error output (stderr) from a command to a file
$ cat some-unknown-file 2>> cat2.stderr.txt

# Append both normal and error output (stderr) from a command to a file
$ ps auxf &>> processes.txt
```

Pipes

The shell pipe (`|`) **is a way to** communicate between commands.

- Example 1: Let's use sort command:

```
ls -1 *.txt | sort -n    # sorts the output in ASC order
ls -1 *.txt | sort -nr   # sorts the output in DESC order
```

- Example 2: Let's use head & tail command:

```
ls -1 *.txt | sort -n | head -n 5  # show the first 5 lines
ls -1 *.txt | sort -n | tail -n 5  # show the last 5 lines
```

- Example 3: Search for a pattern in a text file:

```
cat /etc/passwd | grep root  # show lines containing string 'root'
```

Environment Variables

```
# List all environment variables
$ env

# List all environment variables (alternative)
$ printenv

# Display value of an environment variable
$ echo $HOME

# Display value of an environment variable (alternative)
$ printenv HOME

# Create an environment variable
$ export PORT=80

# Add value to existing variable
$ export PORT=$PORT:90

# Delete an environment variable
$ unset PORT
```

Persistent Environment Variables

To make environment variables persistent you need to define those variables in the bash configuration files.

- `/etc/environment` - Use this file to set up system-wide environment variables.
- `/etc/profile` - Variables set in this file are loaded whenever a bash login shell is entered.
- `~/.bashrc` - Per-user shell specific configuration files

```
# Using 'export' command to declaring environment variables in this file
$ export JAVA_HOME="/path/to/java/home"
$ export PATH=$PATH:$JAVA_HOME/bin

# Load the new environment variables into the current shell session
$ source ~/.bashrc
```

Enlaces de interés

- https://devdoc.net/linux/UnixToolbox.html
- https://tldr.sh/
- https://tldr.inbrowser.app/
- https://linuxcommandlibrary.com/
- https://www.commandlinefu.com/
- https://explainshell.com/
- https://www.gnu.org/software/software.html
- https://linuxcommand.org/lc3_man_page_index.php
- https://terminaldelinux.com/terminal/

Licencia