

Linux para Ingeniería:

La Consola en Linux y la Línea de Comandos

Luis Garreta

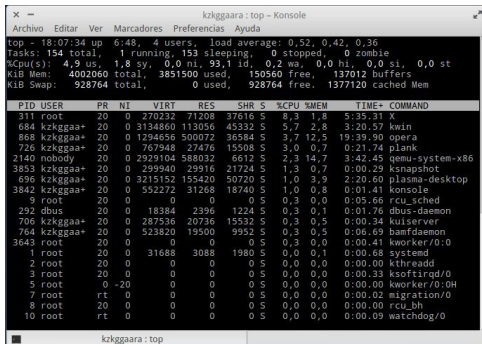
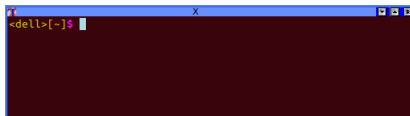
luis.garreta@javerianacali.edu.co

Ingeniería de Sistemas y Computación
Pontificia Universidad Javeriana – Cali

2 de febrero de 2018

¿Qué es la consola?

La consola o terminal (Shell) es un programa informático donde interactúa el usuario con el sistema operativo mediante una ventana que espera ordenes escritas por el usuario desde el teclado.



¿Por qué usar la consola?

- ▶ La consola permite un mayor grado de funciones y configuración de lo que queremos hacer con una aplicación o acción en general respecto del entorno gráfico.
- ▶ "A grosso modo", puedes tener un mayor control sobre tu equipo.
- ▶ En GNU/Linux la consola es algo necesario:
 - ▶ Acciones para dar o quitar permisos,
 - ▶ configurar e instalar drivers que no estén empaquetados y puedan ser ejecutados por un instalador,
 - ▶ matar procesos de una manera más efectiva,
 - ▶ ejercer como superusuario cuando estás en una cuenta cualquiera del equipo
 - ▶ y muchas acciones más que puedes vas a necesitar más adelante.

¿Puede cualquier usuario usar la consola?

- ▶ Cualquier usuario puede usar la consola siempre que sepa lo que está haciendo en ella, ya que si ejecutamos algún comando sin conocimiento y este resulta peligroso para nuestro sistema, podríamos dejar nuestro sistema inutilizable, borrar archivos necesarios, etc.
- ▶ Además, existen algunos comandos que solo pueden ser ejecutados por el *root* ya que manejan recursos propios del sistema en general como por ejemplo:
 - ▶ Creación de usuarios
 - ▶ Montaje de dispositivos
 - ▶ Manejo de procesos
 - ▶ Apagado de la máquina

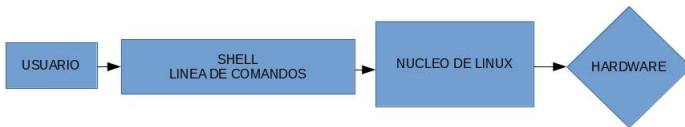
¿Qué conocimientos previos son necesarios?

- ▶ Los conocimientos previos más básicos son los comandos que hay en la consola.
- ▶ Es imposible saberlos todos de memoria,
- ▶ Pero si es recomendable que los más usados se sepan.
- ▶ A la hora de hacer configuraciones, instalaciones, modificaciones, etc. si es necesario que se tenga noción de que archivo es, su importancia en Linux, guardar una copia del archivo.
- ▶ Los comandos al escribirlos en pantalla se ejecutan en la carpeta actual donde se esté ubicado, por tanto, si se quiere realizar un acción sobre otra carpeta basta con poner la ruta después del comando.

Introducción a la línea de comandos de Linux

- ▶ Actualmente la mayoría de las distribuciones ofrecen una interfaz gráfica para realizar distintas tareas.
- ▶ Pero, muchas de estas se realizan más rápido y con más poder desde la línea de comandos

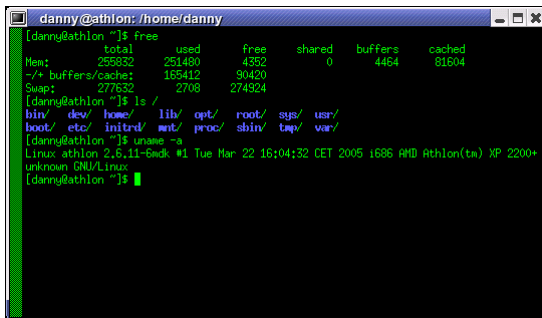
Interprete de Comandos o Shell



- ▶ El shell es un software que toma los comandos desde el teclado y se los pasa al SO
 - ▶ Así permite ejecutar instrucciones que el usuario introduce vía teclado o en un script y este le devuelve los resultados.
- ▶ Este *shell* es la concha (shell es concha en ingles) que rodea el núcleo de Linux,
- ▶ La interacción con el shell se realiza mediante una terminal o emulador de terminal

Terminal y Emuladores de Terminal

- ▶ Originalmente una terminal era una maquina que solamente podía ejecutar instrucciones y obtener resultados por la linea de comandos
- ▶ Hoy en día se utilizan programas que simulan terminales en los diferentes entornos gráficos de ventanas:
 - ▶ xterm
 - ▶ gnome-terminal
 - ▶ rxvt
 - ▶ terminator (Multiples terminales)



```
danny@athlon: /home/danny
[danny@athlon ~]$ free
              total        used        free      shared    buffers     cached
Mem:      255832      251480        4352           0        4464       81604
-/+ buffers/cache:      165412        90420
Swap:      277632         2708       274924
[danny@athlon ~]$ ls /
bin/  dev/  home/  lib/  opt/  root/  sys/  usr/
boot/  etc/  initrd/  mnt/  proc/  sbin/  tmp/  var/
[danny@athlon ~]$ uname -a
Linux athlon 2.6.11-6mdk #1 Tue Mar 22 16:04:32 CET 2005 i686 AMD Athlon(tm) XP 2200+
unknown GNU/Linux
[danny@athlon ~]$
```


Terminator: Múltiples terminales

```

mohd_sohail@LinuxAndUbuntu: ~
mohd_sohail@LinuxAndUbuntu: ~ 63x41
top - 15:45:04 up 3:44, 4 users, load average: 0.37, 1.14, 1.1
Tasks: 275 total, 1 running, 274 sleeping, 0 stopped, 0 z
Cpu(s): 13.3 us, 1.8 sy, 0.0 ni, 84.6 id, 0.3 wa, 0.0 hi,
KiB Mem : 15364488 total, 7449964 free, 5883176 used, 203134
KiB Swap: 3905532 total, 3905532 free, 0 used, 874081

  PID USER      PR  NI  VIRT  RES  SHR  S  %CPU  %MEM
12622 mohd_soh 20   0 978508 183920 78352 S 29.5 1.2
12770 mohd_soh 20   0 1072404 292720 75188 S 5.6 1.9
4037 mohd_soh 20   0 2204236 791340 107336 S 3.6 5.2
2819 root      20   0 494948 93840 57848 S 3.0 0.6
3590 mohd_soh 20   0 1703504 138108 74284 S 3.0 0.9
6892 mohd_soh 20   0 1626964 664964 218236 S 1.3 4.3
11038 mohd_soh 20   0 796292 62968 36856 S 1.3 0.4
11017 mohd_soh 20   0 1516572 510748 87152 S 1.3 3.3
10198 mohd_soh 20   0 1056388 104824 41488 S 1.0 0.7
10890 mohd_soh 20   0 1117564 351000 155900 S 1.0 2.3
3572 mohd_soh 20   0 644504 41676 25216 S 0.7 0.3
4275 mohd_soh 20   0 872880 105984 53952 S 0.7 0.7
4311 mohd_soh 20   0 1025568 205348 54116 S 0.7 1.3
6834 mohd_soh 20   0 936992 171152 75264 S 0.7 1.1
9366 mohd_soh 20   0 536892 62296 32748 S 0.7 0.4
11109 mohd_soh 20   0 48980 3912 3192 R 0.7 0.0
15030 mohd_soh 20   0 939492 204844 81084 S 0.7 1.3
  7 root      20   0 0 0 0 S 0.3 0.0
 10 root      20   0 0 0 0 S 0.3 0.0
3322 mohd_soh 20   0 44716 5176 2852 S 0.3 0.0
3401 mohd_soh 20   0 354516 9328 5472 S 0.3 0.1
4314 mohd_soh 20   0 846140 133176 55900 S 0.3 0.9
6926 mohd_soh 20   0 977292 187144 76160 S 0.3 1.2
9054 mohd_soh 20   0 982380 229976 99128 S 0.3 1.5
9075 mohd_soh 20   0 949528 203344 76056 S 0.3 1.3
10847 mohd_soh 20   0 986264 202856 74192 S 0.3 1.3
11323 root      20   0 0 0 0 S 0.3 0.0
12524 mohd_soh 20   0 929436 195084 84712 S 0.3 1.3
13148 mohd_soh 20   0 979808 208308 74296 S 0.3 1.4
13662 root      20   0 0 0 0 S 0.3 0.0
14124 mohd_soh 20   0 981820 208584 75044 S 0.3 1.4
14513 mohd_soh 20   0 1050592 284328 84504 S 0.3 1.9
15084 mohd_soh 20   0 891144 159624 71724 S 0.3 1.0
15403 mohd_soh 20   0 916784 173464 78112 S 0.3 1.1
  
```

```

mohd_sohail@LinuxAndUbuntu: ~/Desktop 63x19
mastering javascript.torrent
multiple terminals in terminator.png
privacy.png
qownnotes setup owncloud.png
qownnotes setup.png
Screenshot from 2016-07-22 17:10-11.png
script.js
style.css
themes.png
tv.png
mohd_sohail@LinuxAndUbuntu: ~/Desktop$
  
```

```

mohd_sohail@LinuxAndUbuntu: ~ 63x20
[sudo] password for mohd_sohail:

(nautilus:16129): Gtk-WARNING **: Failed to register client: GD
desktop.DBus.Error.ServiceUnknown: The name org.gnome.SessionMa
id by any .service files

** (nautilus:16129): CRITICAL **: Another desktop manager in us
won't be created

Nautilus-Shares-Messsage: Called "net usershare info" but it fail
ute child process "net" (No such file or directory)
  
```

Existen Varios Tipos de Shells

- ▶ Existen varios shells que son mas o menos lo mismo con características y sintaxis diferentes:
 - ▶ Bourne shell: El más básico disponible en todos los sistemas UNIX
 - ▶ C-shell: similar al lenguaje de programación C en sintaxis
 - ▶ Korn-shell: Basado en Bourne shell con algunas mejoras
 - ▶ Bash Shell: Bourne Again Shell, combina las ventajas del Korn Shell y del C Shell
 - ▶ ...
- ▶ El shell por defecto en la mayoría de sistemas Linux es *Bash shell*.

Línea de Comandos

El shell espera instrucciones por el teclado en una línea llamada línea de comandos o prompt. Esta línea de comandos nos ofrece cierta información fácilmente reconocible :

`[asierph@manjaro Documentos]$` █

The diagram shows the prompt `[asierph@manjaro Documentos]$` with arrows pointing to its parts:

- `asierph` points to **Usuario conectado**
- `@` points to **Posición actual en el sistema de ficheros**
- `manjaro` points to **Posición actual en el sistema de ficheros**
- `/Documentos` points to **Posición actual en el sistema de ficheros**
- `$` points to **\$ Sin privilegios** and **# Con privilegios**

\$ Sin privilegios
Con privilegios

Posición actual en el sistema de ficheros

Hostname (Nombre logico de la maquina)

Usuario conectado

Sintaxis de los comandos

- ▶ Los comandos GNU/Linux tienen una sintaxis del tipo :

Instrucción [parámetros] [argumentos]

- ▶ No es necesario introducir los parámetros y los argumentos, si no se introducen se ejecutaran los valores por defecto de estos.
- ▶ Los parámetros son opciones del comando y normalmente se escriben mediante un guion y una letra (-l por ejemplo).
- ▶ Se puede ejecutar mas de un parámetro por instrucción escribiendo guion y letra varias veces (-l -a) o uniendo las letras detrás del guion (-la).

Ejemplo1: \$ ls

Ejemplo2: \$ ls -l

Ejemplo3: \$ mkdir tmp

Ejemplo4: \$ touch a

Ejemplo5: \$ cp a tmp/

Comandos internos y externos

Los comandos pueden ser de tipo interno o externo

- ▶ **Interno** : Estos comandos son internos a la shell, se ejecutan dentro de esta y forman parte de ella.
- ▶ **Externo** : Los comandos externos son binarios dentro del sistema que son llamados cuando se ejecuta el comando asociado, este comando se carga en memoria y se inicia como proceso.

Para saber si un comando es de tipo interno o externo podemos ejecutar la instrucción:

```
type [comando].
```

En el caso de que el comando sea externo también nos indicara su ubicación.

Ayuda de las instrucciones

- ▶ Es imposible conocer cada uno de los parámetros y argumentos de todas las instrucciones que tenemos a nuestra disposición pero esto no hace falta gracias a la ayudas que nos ofrece Linux.
- ▶ Tenemos tres tipos de ayuda que nos van a ofrecer información sobre cada una de las instrucciones :

- ▶ Ayuda interna del shell:

```
$ help  
$ help cd
```

- ▶ Ayuda propia de los comandos:

```
$ mkdir --help
```

- ▶ Ayuda en línea:

```
$ man ls
```

Privilegios

- ▶ Muchos de los comandos que vamos a presentar a continuación necesitan privilegios para poder ser ejecutados.
- ▶ Para llamar a un comando con derechos de administrador tendremos que colocar antes de la instrucción el comando:

```
$ sudo comando
```

Ejemplo1: `$ sudo shutdown -h now`

- ▶ Después de ejecutar instrucciones con sudo el interprete nos pedirá las credenciales de administrador.
- ▶ También podemos ejecutar el comando:

```
$ su -
```

con el que conseguiremos derechos de root

COMANDOS DE AYUDA

Siempre que no se sabe como funciona o para que sirve un comando, hay que documentarse antes de usarlo y para ello tenemos estos comandos.

- ▶ **man** comando: muestra manual del comando que le indiquemos comando
- ▶ comando **help**: da una ayuda de los comandos
- ▶ **whatis** comando: muestra descripción del comando
- ▶ Principales:
 - ▶ **touch** nombre_archivo: crear archivo vacío
 - ▶ **ls**: listar los archivos
 - ▶ **mkdir** nombre: crear un directorio
 - ▶ **cd** nombre: cambiar de directorio
 - ▶ **rm** nombre: borrar archivos
 - ▶ **rmdir** nombre: borrar directorios
 - ▶ **cp** ruta_origen ruta_destino: copiar archivo
 - ▶ **pwd**: muestra la ruta actual o donde estoy parado
 - ▶ **mv** ruta_origen ruta_destino: mover o renombrar archivos y directorios
 - ▶ **cat** nombre: ver el contenido de un archivo // unir varios archivos en uno
 - ▶ **grep** palabra archivo: buscar un texto en archivo

- ▶ Otros:

COMANDOS DE MANEJO DE USUARIOS

► Principales:

- `passwd usuario`: cambiar la contraseña
- `adduser usuario grupo`: agregar nuevo usuario al un grupo
- `userdel usuario`: borrar un usuario
- `su usuario2`: cambiar de cualquier usuario a usuario2
- `whoami`: mostrar nombre de usuario

► Otros:

- `id usuario`: mostrar datos de identificación del usuario
- `finger usuario`: mostrar información de usuario
- `last`: información de los últimos usuarios que han usado el sistema
- `write`: manda un mensaje a la pantalla de un usuario
- `mesg`: activo o desactivo recibir mensajes
- `wall`: mensaje a todos los usuarios
- `talk`: establecer una charla con otro usuario

COMANDOS DE PROCESOS

► Principales:

- Ctrl+c: Abortar proceso:
- top: mostrar los procesos que se estan ejecutando
- ps: mostrar la lista de procesos del usuario
- kill 9 ID: matar proceso por ID
- xkill: matar proceso de forma grafica haciendo clic en la ventana a matar
- Ctrl+z: Pausar proceso actual
- bg proceso: pone un proceso en segundo plano
- fg proceso: trae a primer plano un proceso parado o en segundo plano
- proceso & : ejecuta un comando en segundo plano

COMANDOS DEL SISTEMA

- ▶ Comandos para Salir del Sistema o Limpiar
 - ▶ shutdown: apaga el sistema
 - ▶ reboot: reinicia la maquina
 - ▶ exit: cierro sesion actual
 - ▶ logout: salgo del sistema
 - ▶ clear: borro la pantalla
- ▶ Comandos para consultar estado del sistema:
 - ▶ free: muestra estado de la memoria RAM
 - ▶ history: muestra todos los comandos digitados por el usuario
 - ▶ uname a: da informacion de tu sistema operativo, kernel, usuario...
 - ▶ hostname: muestra el nombre del servidor
 - ▶ cal, date, time: muestra calendario, fecha y tiempo
 - ▶ who: muestra los usuarios que están conectados al sistema
 - ▶ df: muestra la cantidad de disco utilizado
 - ▶ echo \$nombre_variable: muestra el valor de una cadena en la pantalla

COMANDOS DE RED

- ▶ rsh servidor: se conecta a otra maquina de forma remota (remote shell)
- ▶ ftp: se conecta a otra maquina por el protocolo ftp

COMBINACION DE TECLAS

- ▶ `ctrl+L`: borra pantalla
- ▶ `ctrl+z`: suspendo proceso
- ▶ `ctrl+c`: termina proceso en ejecucion
- ▶ `tab`: completa nombre de carpetas o archivos

SIMBOLOS

- ▶ ~ path desde la raíz al home
- ▶ . directorio actual
- ▶ .. directorio superior al actual
- ▶ | pipe : redirecciona comandos
- ▶ > redireccionar salida de un comando
- ▶ < redireccionar entrada de un comando