

Universidad Tecnológica de Panamá
Sistemas Operativos I
Experiencia Práctica en Laboratorio No. 8
El Shell de Linux

Prof. Aris Castillo de Valencia

Estudiante: Gabriel Díaz

Cedula: 20-53-5198

Objetivos:

1. Utilizar la línea de comandos para desarrollar tareas básicas
2. Usar y modificar el ambiente Shell incluido y definir, referenciar y exportar variables
3. Invocar comandos dentro y fuera de la ruta definida.

Metas:

Con esta experiencia práctica se espera que el estudiante sea capaz de realizar tareas relacionadas con el Shell del sistema operativo Linux/GNU a través de comandos.

Contenidos:

- Shells de Linux
- Variables de entorno

Metodología:

Se basa en métodos intuitivos, de experimentación y demostración en que se acerca al estudiante a situaciones reales de la práctica profesional de manera que resuelva las situaciones presentadas.

Evaluación:

- Se dará 50 puntos por el desarrollo de la práctica en el aula.
- Se dará 50 puntos por la entrega del informe escrito debidamente completado y por su nivel técnico.

Recursos:

- Hardware: computadora, conexión a Internet.

- Software: Sistema operativo Linux/GNU.

Procedimiento:

- Lea cuidadosamente la guía; pruebe cada uno de los comandos listados prestando especial atención a los resultados obtenidos y a las variantes que le ofrecen las opciones de los comandos. Ponga en práctica los comandos aprendidos haciendo los ejercicios sugeridos. Desarrolle la retroinformación y súbala a la plataforma Moodle.

Introducción

Shell o intérprete de comandos de GNU/Linux: es el medio que tenemos para interactuar con la máquina con la que estamos trabajando y con su sistema operativo. Una gran parte del trabajo de UNIX consiste en emitir órdenes. Emitir una orden significa que nos estamos comunicando con el SHELL.



Bash: Existen muchos tipos de shell, el más popular es 'bash' (bourne again shell). **Bash** es un programa informático basado en Shell cuya función consiste en interpretar órdenes.

La shell **bash** se puede utilizar de modo interactivo o como un lenguaje de escritura de gran alcance. Tras el arranque, **bash** ejecuta los comandos hallados en el archivo ~/.bashrc, permitiéndoles a los usuarios personalizar su shell. La shell **bash** guarda el historial de las líneas de comando ejecutadas.

Ejemplo: Con frecuencia, los programadores de lenguajes compilados tales como C suelen hallarse en un ciclo repetitivo: editar un archivo, compilarlo y luego ejecutar el programa. A continuación, un

usuario edita un archivo que contiene un programa pequeño C y luego lo compila con el compilador C **gcc**. Después de ejecutar el programa, decide hacer algunos cambios. Hace entonces uso del historial de comandos para agilizar el proceso.

```
root@ubuntu:~# vi hello.c
root@ubuntu:~# gcc -o hello hello.c
root@ubuntu:~# ./hello
hello world
root@ubuntu:~# !n
nano hello.c
```

```
root@ubuntu:~# !g
gcc -o hello hello.c
root@ubuntu:~# !.
./hello
hello dolly
root@ubuntu:~# █
```

Nota: Observe que la shell **bash** imprime el comando seleccionado desde el historial del usuario antes de ejecutar el comando.

Ordenes más comunes:

Comando pwd:

Variable que muestra la ruta del directorio actual. Esto es muy útil, cuando estamos dentro de muchos directorios y no sabemos dónde nos encontramos.

Sintaxis: **pwd**

```
root@ubuntu:~# pwd
/home/darlene
root@ubuntu:~# █
```

Comando uname: este comando te devuelve información sobre el sistema que estas usando.

Sintaxis: **uname** [opción]

Opciones:

- **-a:** da toda la información.
- **-r:** versión del kernel

```
root@ubuntu:~# uname -a
Linux ubuntu 2.6.38-8-generic #42-Ubuntu SMP Mon Apr 11 03:31:50 UTC 2011 i686 i
686 i386 GNU/Linux
root@ubuntu:~# uname -v
#42-Ubuntu SMP Mon Apr 11 03:31:50 UTC 2011
```

Comando history:

Este comando almacena en el buffer de memoria los comandos que se han ido ejecutando en la shell y con él es posible conocer cuales se han efectuado, y citarlos por su número de orden.

Sintaxis: **history**

```
1497 clear
1498 sudo -s
1499 gedit /etc/dtn.conf
1500 sudo -s
1501 ifconfig
1502 sudo -s
1503 dtnping localhost
1504 sudo -s
1505 gedit /etc/dtn.conf
1506 hostname ubuntu-a
1507 hostname
1508 dtnping localhost
1509 hostname
1510 hostname ubuntu-a
1511 sudo -s
1512 hostname ubuntu-a
1513 hostname
1514 sudo -s
1515 gedit /etc/dtn.conf
1516 dtnping localhost
1517 dtnping dtn://ubuntu.dtn
1518 dtnrecv dtn://ubuntu.dtn
1519 sudo -s
1520 dtntperf-server
```

La primera columna indica el número de comando, en el orden en el que se ha ejecutado. La segunda evidentemente es el comando que se ha ejecutado.

Si por ejemplo de la lista anterior se desea ejecutar un comando, se haría lo siguiente:

```
root@ubuntu:~# !1509
hostname
ubuntu
root@ubuntu:~#
```

Comando echo:

Muestra en pantalla lo que se indica como argumento.

Sintaxis: **echo** *[argumento]*

```
root@ubuntu:~# echo casa
casa
```

Variables de Entorno

Las variables de entorno son porciones de memoria que el sistema toma para guardar valores específicos necesarios para el sistema.

Comando set

Nos sirve para saber las variables de entorno que hay en el sistema. También se utiliza para definir un valor a las variables de entorno del usuario.

Sintaxis: **set** | **more**

```
root@ubuntu:~# set | more
BASH=/bin/bash
BASHOPTS=checkwinsize:cmdhist:expand_aliases:extglob:extquote:force_ignore:hist
append:interactive_comments:progcomp:promptvars:sourcepath
BASH_ALIASES=()
BASH_ARGC=()
BASH_ARGV=()
BASH_CMDS=()
BASH_COMPLETION=/etc/bash_completion
BASH_COMPLETION_COMPAT_DIR=/etc/bash_completion.d
BASH_COMPLETION_DIR=/etc/bash_completion.d
BASH_LINENO=()
BASH_SOURCE=()
BASH_VERSIONINFO=([0]="4" [1]="2" [2]="8" [3]="1" [4]="release" [5]="i686-pc-linux-
gnu")
BASH_VERSION='4.2.8(1)-release'
```

Comando unset:

Remueve definiciones de variables o funciones.

Sintaxis: **unset** *[variable]*

Ejemplo:

```
root@ubuntu:~# echo $edad
23
root@ubuntu:~# unset edad
root@ubuntu:~# echo $edad

root@ubuntu:~#
```

Comando env:

Muestra las variables de entorno definidas para el usuario actual.

Sintaxis: **env**

Ejemplo:

```
root@ubuntu:~# env
SHELL=/bin/bash
TERM=xterm
USER=root
LS_COLORS=rs=0:di=01;34:ln=01;36:mh=00:pi=40;33:so=01;35:do=01;35:bd=40;33;01:cd
=40;33;01:or=40;31;01:su=37;41:sg=30;43:ca=30;41:tw=30;42:ow=34;42:st=37;44:ex=0
1;32:*.tar=01;31:*.tgz=01;31:*.arj=01;31:*.taz=01;31:*.lzh=01;31:*.lzma=01;31:*.
tlz=01;31:*.txz=01;31:*.zip=01;31:*.z=01;31:*.Z=01;31:*.dz=01;31:*.gz=01;31:*.lz
=01;31:*.xz=01;31:*.bz2=01;31:*.bz=01;31:*.tbz=01;31:*.tbz2=01;31:*.tz=01;31:*.d
eb=01;31:*.rpm=01;31:*.jar=01;31:*.rar=01;31:*.ace=01;31:*.zoo=01;31:*.cpio=01;3
1:*.7z=01;31:*.rz=01;31:*.jpg=01;35:*.jpeg=01;35:*.gif=01;35:*.bmp=01;35:*.pbm=0
1;35:*.pgm=01;35:*.ppm=01;35:*.tga=01;35:*.xbm=01;35:*.xpm=01;35:*.tif=01;35:*.t
iff=01;35:*.png=01;35:*.svg=01;35:*.svgz=01;35:*.mng=01;35:*.pcx=01;35:*.mov=01;
35:*.mpg=01;35:*.mpeg=01;35:*.m2v=01;35:*.mkv=01;35:*.ogm=01;35:*.mp4=01;35:*.m4
v=01;35:*.mp4v=01;35:*.vob=01;35:*.qt=01;35:*.nuv=01;35:*.wmv=01;35:*.asf=01;35:
*.rm=01;35:*.rmvb=01;35:*.flc=01;35:*.avi=01;35:*.fli=01;35:*.flv=01;35:*.gl=01;
35:*.dl=01;35:*.xcf=01;35:*.xwd=01;35:*.yuv=01;35:*.cgm=01;35:*.emf=01;35:*.axv=
01;35:*.anx=01;35:*.ogv=01;35:*.ogx=01;35:*.aac=00;36:*.au=00;36:*.flac=00;36:*.

```

Obtención del valor de una variable:

Para obtener el valor de una variable del SHELL debemos indicar el nombre de la variable y anteponer el símbolo \$. Cuando el SHELL encuentra una palabra que comienza por \$ supone que es una variable, y sustituye su valor por la parición de la variable.

Ejemplo:

```
root@ubuntu:~# echo $HOSTNAME
ubuntu
root@ubuntu:~#
```

Definición de variables en el **SHELL**:

Para definir una variable, empleamos la siguiente sintaxis: **nombre=valor**.

Ejemplo:

```
root@ubuntu:~# edad=23
root@ubuntu:~# nombre=darlene
root@ubuntu:~# echo $edad
23
root@ubuntu:~# echo $nombre
darlene
root@ubuntu:~#
```

Nota: Se recomienda que las variables de entorno creadas, estén en minúsculas, de esta manera, no se borra nada indispensable del sistema. Las variables de entorno, cuando las crea un usuario, solo se ejecutan en la terminal donde se crean, para poder ejecutarlas en todas las terminales tendríamos que exportarlas.

Comando export:

Exporta el valor de la variable de entorno que se especifique.

Sintaxis: **export** *nom_var*.

Ejemplo:

```
[root@ubuntu:~]$ export SECONDS=0
[root@ubuntu:~]$ echo $SECONDS
0
```

La variable \$SECONDS cuenta la cantidad de segundos desde que entramos al sistema; con este mandato la seteamos a "0".

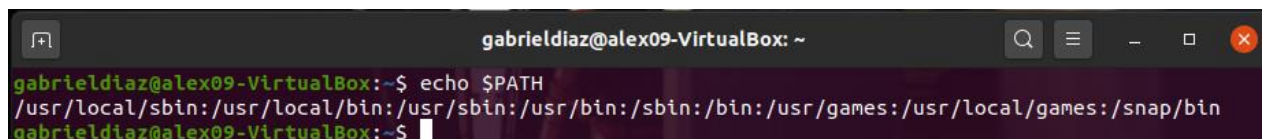
Comando exec: Reemplaza el intérprete de comandos con el programa definido.

Retroinformación.

1. Obtenga el valor de la variable PATH. Describa la salida.

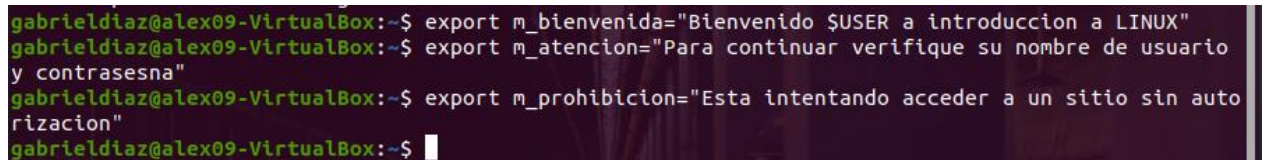
La variable de entorno \$PATH, contiene directorios que, en el caso de introducir un comando sin especificar una ruta, el comando será buscado en los directorios especificados de la variable de entorno PATH.

Al escribir el comando "echo \$PATH" se mostraría la siguiente información de directorios:



```
gabrielldiaz@alex09-VirtualBox: ~
gabrielldiaz@alex09-VirtualBox:~$ echo $PATH
/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/bin:/usr/games:/usr/local/games:/snap/bin
gabrielldiaz@alex09-VirtualBox:~$
```

2. Defina las siguientes variables m_bienvenida, m_atencion, y m_prohibicion para enviarle mensajes al usuario, de acuerdo con el título correspondiente a cada variable.



```
gabrielldiaz@alex09-VirtualBox:~$ export m_bienvenida="Bienvenido $USER a introduccion a LINUX"
gabrielldiaz@alex09-VirtualBox:~$ export m_atencion="Para continuar verifique su nombre de usuario y contrasena"
gabrielldiaz@alex09-VirtualBox:~$ export m_prohibicion="Esta intentando acceder a un sitio sin autorizacion"
gabrielldiaz@alex09-VirtualBox:~$
```

3. Utilice la orden **env** para verificar que las variables hayan sido creadas correctamente.

Variable m_bienvenida:

```
GDMSESSION=ubuntu
DBUS_SESSION_BUS_ADDRESS=unix:path=/run/user/1001/bus
m_bienvenida=Bienvenido gabrieldiaz a introduccion a LINUX
LC_NUMERIC=es_PA.UTF-8
_=/usr/bin/env
```

Variable m_atencion:

```
LC_MEASUREMENT=es_PA.UTF-8
m_atencion=Para continuar verifique su nombre de usuario y contraseña
XDG_RUNTIME_DIR=/run/user/1001
```

Variable m_prohibicion:

```
XDG_SESSION_TYPE=x11
m_prohibicion=Esta intentando acceder a un sitio sin autorizacion
GPG_AGENT_INFO=/run/user/1001/gnupg/S.gpg-agent:0:1
```

4. Ejecute las variables definidas.

```
gabrieldiaz@alex09-VirtualBox:~$ echo $m_bienvenida
Bienvenido gabrieldiaz a introduccion a LINUX
gabrieldiaz@alex09-VirtualBox:~$ echo $m_atencion
Para continuar verifique su nombre de usuario y contraseña
gabrieldiaz@alex09-VirtualBox:~$ echo $m_prohibicion
Esta intentando acceder a un sitio sin autorizacion
gabrieldiaz@alex09-VirtualBox:~$
```

5. ¿Qué aprendió de esta experiencia? ¿Cómo considera que le puede ser útil?

El sistema operativo Linux basado en Unix nos permite crear variables de entorno desde el terminal bash, no existe una limitación y se pueden crear variables con especificaciones propias que luego pueden ser usadas por scripts, por ejemplo, para crear procesos.

6. ¿Cómo considera que se puede mejorar esta experiencia? ¿Qué cambiaría?
7. ¿Qué sugerencias puede aportar?
8. Incluya material de apoyo útil para compartir con su clase.

Bibliografía:

1. Kernighan, B. y Pike, R. El Entorno de programación Unix. Prentice Hall.
2. Husain, Kamran y Parker, Timoty, et al. **Linux Unleashed**. Second Edition.