CARTILLA DEL SEMILLERO DE SO-LINUX

Autores de la investigación formativa:

1. Diego Alexander Ramírez Zuluaga.
2. Brahian Stiven Osorio Velásquez.
3. Ricardo Montoya Gallego.
4. Juan Esteban Ciro Castaño.
5. Juan Camilo Gil Córdoba.
6. Jairo Arias Sánchez.

Instructor
ALIRIO ANTONIO GUTIÉRREZ QUINTERO

Esta cartilla se publica bajo licencia de la Universidad Católica de Oriente. "Se consiente su reproducción y distribución por cualquier medio siempre que conserve el reconocimiento de sus autores, no haga uso comercial de la obra y no se realice ninguna modificación en ella.

Rionegro, noviembre de 2018

TABLA DE CONTENIDO

INTRODUCCIÓN

“La definición de la palabra Linux depende del contexto en el que se utiliza. Linux se refiere al kernel. Es el controlador central de todo lo que pasa en el equipo (veremos más detalles a continuación). Quienes dicen que su equipo "se ejecuta con Linux" generalmente se refiere al kernel y el conjunto de herramientas que vienen con él (llamados distribución). Si tienes "Experiencia con Linux", probablemente te refieres a los propios programas, aunque dependiendo del contexto, podrías hablar sobre tu capacidad de ajustar con precisión el kernel. Cada uno de estos componentes será explorado para que puedas entender exactamente qué papel juega cada uno.” 1

El software de Linux cae generalmente en una de tres categorías:

- Software de servidor – software que no tiene ninguna interacción directa con el monitor y el teclado de la máquina en la que se ejecuta. Su propósito es servir de información a otras computadoras llamados clientes. A veces el software de servidor puede no interactuar con otros equipos, sin embrago, va a estar ahí sentado y "procesando" datos.
- Software de escritorio – un navegador web, editor de texto, reproductor de música u otro software con el que tú interactúas. En muchos casos, como un navegador web, el software consultará a un servidor en el otro extremo e interpretará los datos para ti. Aquí, el software de escritorio es el cliente.
- Herramientas – una categoría adicional de software que existe para que sea más fácil gestionar el sistema. Puedes tener una herramienta que te ayude a configurar la pantalla o algo que proporcione un shell de Linux o incluso herramientas más sofisticadas que convierten el código fuente en algo que la computadora pueda ejecutar.

Linux puede usarse de dos maneras: en modo gráfico y modo no gráfico. En modo gráfico las aplicaciones corren en las ventanas que puedes cambiar el tamaño y mover. Tienes menús y herramientas que te ayudan a encontrar lo que buscas. Aquí es donde vas a usar un navegador web, tus herramientas de edición de gráficos y tu correo electrónico.

En el modo no gráfico no hay ventanas para navegar. A pesar de esto tienes editores de texto, navegadores web y clientes de correo electrónico, pero son sólo de texto. De este modo UNIX empezó antes que los entornos gráficos fueran la norma. La mayoría de los servidores también se ejecutarán en este modo, ya que la gente no entra en ellos directamente, lo que hace que una interfaz gráfica sea un desperdicio de recursos

[1] NDG Linux Essentials Spanish

OBJETIVOS

Objetivo General

Lograr que los lectores interesados por el software libre conozcan los principios sobre los cuales es utilizado el Sistema Operativo, aplicando diferentes proyectos prácticos en los que participen

Objetivos específicos

- Aprender a usar las características básicas del Shell
- Implementar y gestionar los directorios, archivos, usuarios, grupos y seguridad del sistema operativo
- Interpretar los datos estadísticos del hardware y del software a través de la línea de comandos

CAPÍTULO 1: EL EXTENSO RINCÓN DONDE CLIQUEAR NO ES LO PRIMORDIAL

El objetivo del capítulo.
Introducir al usuario en el mundo de la interfaz de línea de comandos (CLI), por lo tanto, se enseñará a utilizar las diferentes funciones básicas del Shell (definición y exploración de este término en los siguientes párrafos), las tareas a realizar serán:

- Dar una introducción a las cualidades del Bash.
- Comprender el uso de las variables del Shell
- Estudiar y emplear el globbing.
- Explorar y entender la aplicación de las comillas.

Palabras calve: (*CLI, Introducción, comandos, terminal, teclado.* prompt*)*

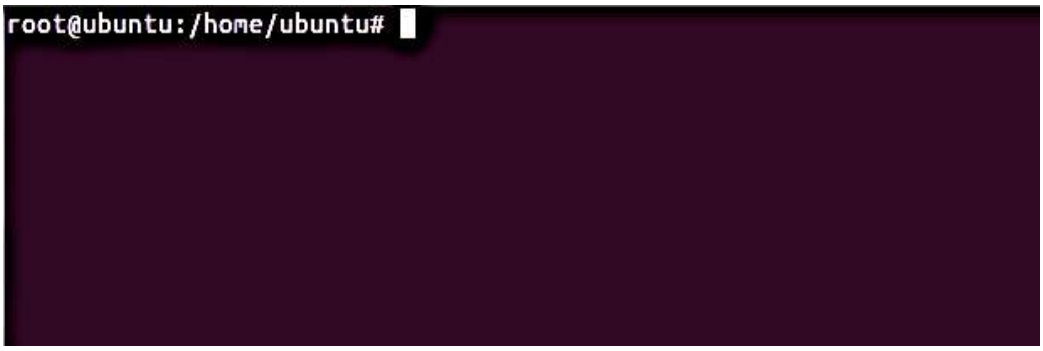
Existe un método para controlar la computadora de una forma más rápida y precisa, la pregunta es ¿cómo acceder a este? Es así como se da a conocer la interfaz de línea de comandos (CLI), esta nos da la posibilidad de manejar la computadora desde el teclado por medio de la inserción de comandos, debido a que lo habitual es manejar una interfaz gráfica de usuario (GUI), (las típicas ventanas de Windows), esto será un nuevo reto, pues se necesita práctica para aprender a utilizar y memorizar los comandos que se emplean para comunicarse con la máquina.

Introducciones al mundo de la línea de comandos.

Como se dijo anteriormente, para empezar a utilizar el entorno CLI (interfaz de línea de comandos) se debe aclarar que esto no es un entorno gráfico, toda instrucción (también llamadas comandos) se inserta por medio del teclado; el lugar donde se declaran estas órdenes se llama terminal(La interfaz de esta se muestra en la siguiente figura), para que seguidamente el Shell que funciona como el intérprete de comandos, cumpla su función de traducir estas instrucciones al sistema operativo. Si la instrucción se reconoce, la terminal dará una respuesta correspondiente, en caso contrario, se mostrará un mensaje de error.

Para empezar a introducir comandos primero, debemos buscar el programa predeterminado para esto, el cual es la “terminal” (prompt), está por lo general se puede encontrar en el inicio del sistema operativo(S.O.), en caso contrario, la podemos localizar por medio de la barra de búsqueda del S.O. de Linux, y seguidamente ejecutarla, como cualquier otro programa de pc, después se verá una línea seguida de un cursor parpadeante que tiene como función indicarle al usuario que digite algún comando.

Además de eso el Prompt, le proporciona cierta información a la persona acerca de su estado actual, como, por ejemplo: el tipo de usuario, el host de la computadora que utiliza y el directorio en el que se encuentra (detalles de estos en la siguiente figura)



Figura#1 Prompt.

Como ya se especificó anteriormente el Shell es el traductor de la máquina, pero no existe un único tipo de este, el entorno Linux proporciona diferentes tipos de Shells. El enfoque para hacer las prácticas será el Shell Bash, debido a que es el más comúnmente utilizado, esto para que se pueda ir familiarizando poco a poco con este nuevo entorno, que quizá, sea nuevo para el usuario.

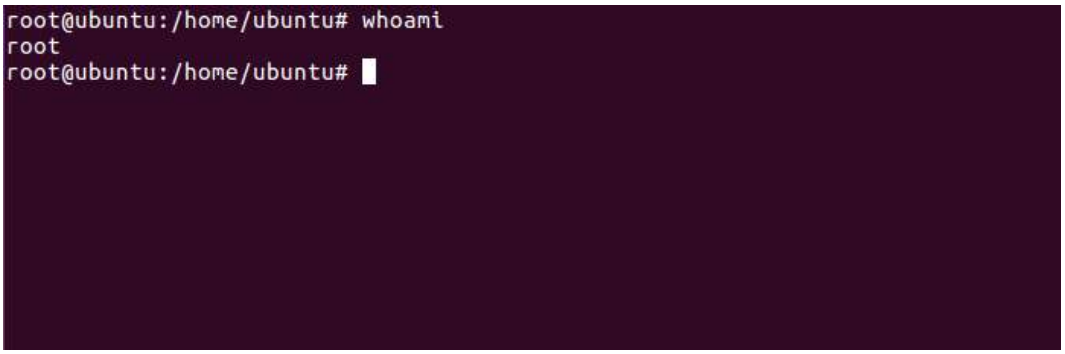
PRACTICA DE LABORATORIO DEL CAPÍTULO 1

Paso #1 Conocer nuestra información como usuarios:

Los primeros comandos en digitar tienen la función de dar a conocer el entorno, y aunque alguna información ya la contiene el Prompt, es bueno tenerlos en cuenta, debido a que habrá ocasiones en las que el Prompt no tenga la información requerida por el usuario.

Paso #1.1

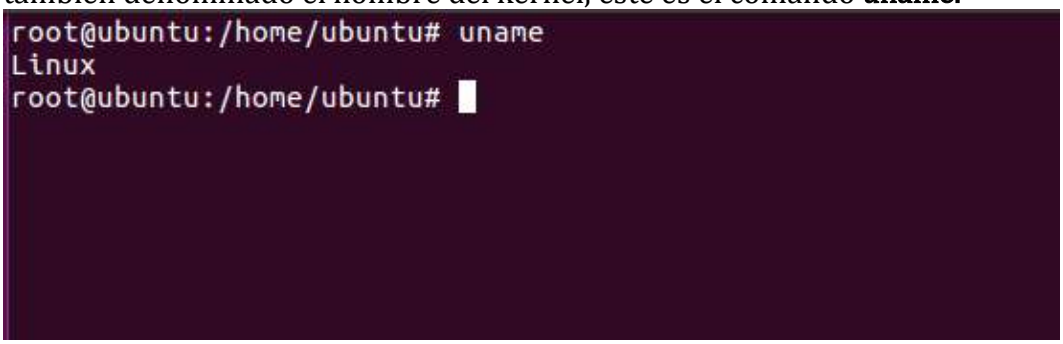
El primer comando en ejecutar será whoami (se anunciarán los comandos en negrita para diferenciarlos de los textos del laboratorio), este tiene la función de mostrar el nombre de usuario actual.



Figura#2 whoami

Paso # 1.2

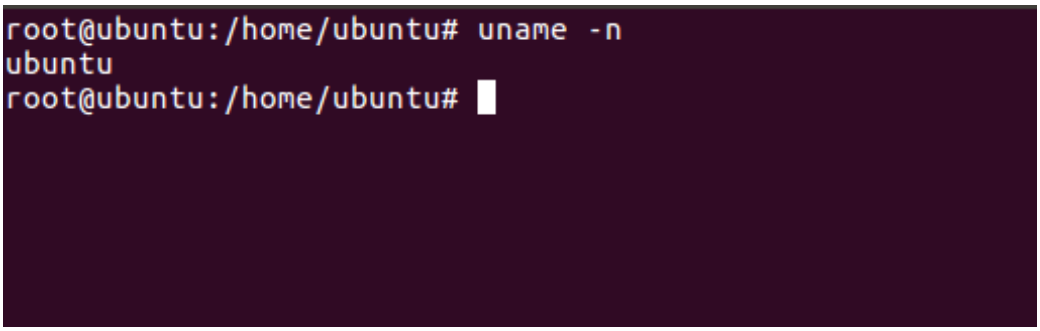
Nuestro segundo comando tiene como función mostrar la información del sistema actual, o también denominado el nombre del kernel, este es el comando **uname.**



Figura#3 uname

Paso #1.3

Hay ocasiones en las que un comando puede tener opciones y esto provoca variaciones en la salida de la instrucción, en UNIX, esto se puede lograr con algunas adiciones de caracteres después del comando, como por ejemplo un guion (-) y otro carácter. Intenta el comando **uname** -n, esto tiene como función imprimir el nombre del host del nodo de la red, también especificado en el Prompt.



Figura#4uname -n

Paso #1.4

El siguiente comando será **pwd**, se utiliza para mostrar la "ubicación" actual en los directorios de trabajo (Los directorios son un tipo de carpetas digitales donde se guardan los archivos).

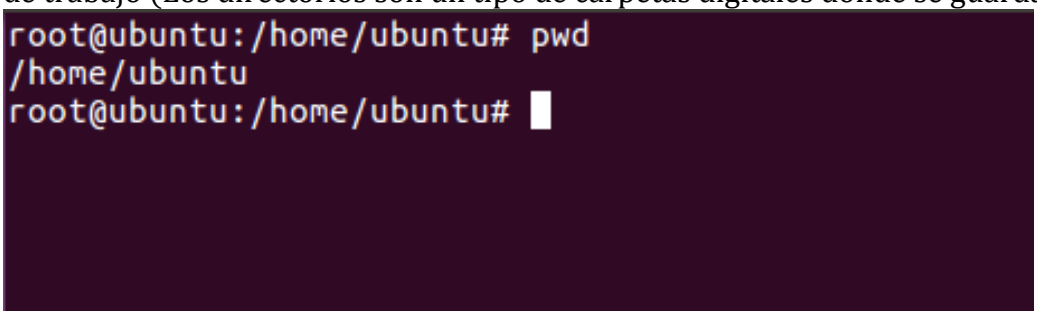


Figura5 #pwd

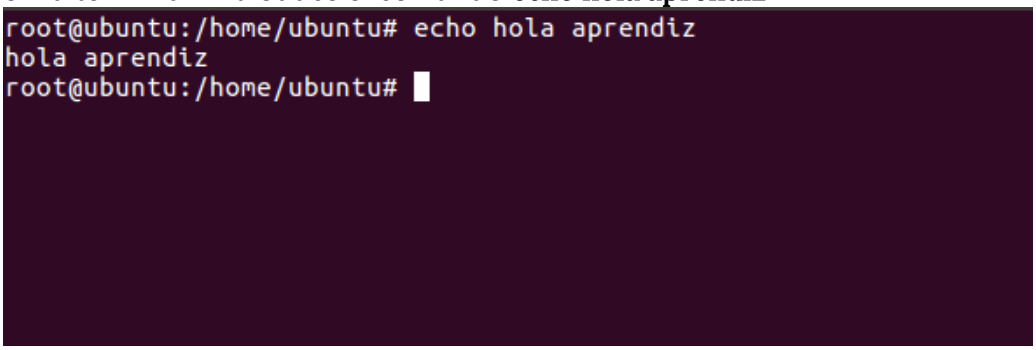
Como se puede ver en el ejemplo anterior, nuestro directorio actual de trabajo es /home/ubuntu, también conocido como directorio home. Como puedes notar en el ejemplo anterior, Linux utiliza la barra diagonal "/" para crear la ruta separando los directorios.

Paso # 2 Datos del sistema operativo y como el Shell los almacena:

Las variables del Shell se utilizan para almacenar los datos en Linux. Estos datos los utiliza el propio Shell, los programas y los usuarios.

Paso #2.1

De esta sección, el primer comando en explorar será **echo** y tiene la función de imprimir texto en la terminal. Introduce el comando **echo hola aprendiz**.

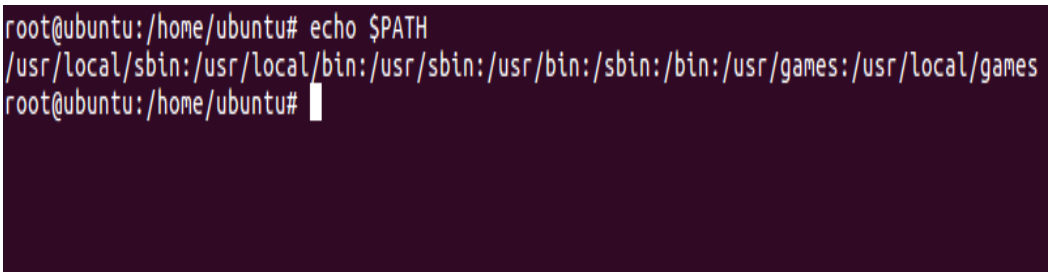


Figura#6 echo hola aprendiz

Como se puede observar, imprime el texto que tenga seguido, además de eso también imprime el valor de una variable y muestra cómo el entorno del Shell expande los metacaracteres.

Paso #2.2

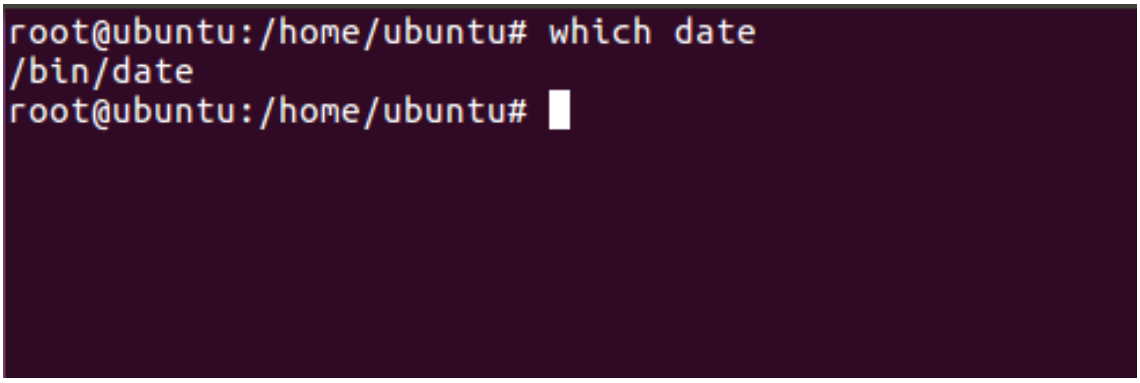
El próximo comando en analizar tiene la función de mostrar el valor de la variable **PATH**. **echo $PATH** utiliza el **$** para poder visualizar la variable y que esta no se confunda como algo que quieres imprimir en pantalla. **PATH**, tiene como objetivo encontrar la ubicación del ejecutable de cada comando.



Figura# 7echo $PATH

Paso #2.3

El último comando de esta sección en utilizar tiene una similitud con el anterior. Hablamos de **which** que se puede utilizar para determinar si existe algún archivo ejecutable con el nombre que se necesite, ubicado en un directorio que aparece en el valor de **PATH**.
Wich hace uso de PATH para determinar la ubicación de cualquier comando. Introduce en la terminal **which date**.



Figura# 8 which date

El resultado anterior explica que cuando introduces el comando “date”, el sistema ejecutará el comando /bin/date.

Paso # 3 Caracteres especiales para usar el Globbing.

En esta parte se conocerán los caracteres Glob, que tienen la función de hacer coincidir los nombres de archivos usando patrones. Estos caracteres son una característica del Shell, no es algo propio de algún comando específico. Gracias a esto, se puede utilizar los glob en cualquier comando de Linux.

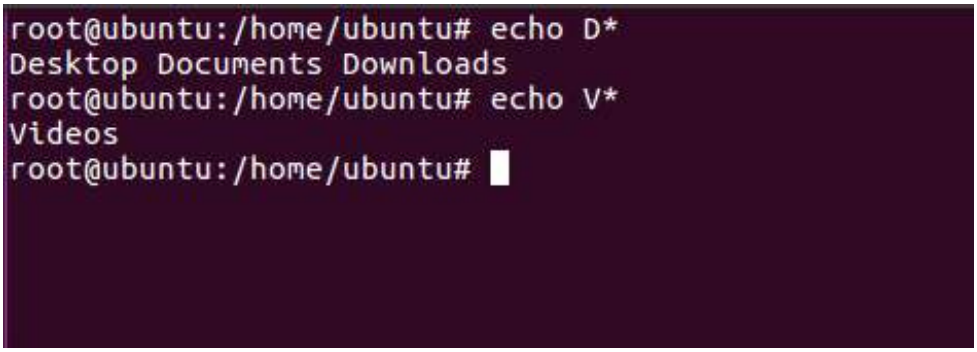
Paso # 3.1

El primer carácter glob en conocer será el asterisco (*), junto con el comando echo, para mostrar todos los nombres de archivos en el directorio actual que coinciden con el patrón dado (*), este busca cero o más coincidencias de caracteres en un nombre de archivo. Digita el comando echo *.

Figura#9 echo *

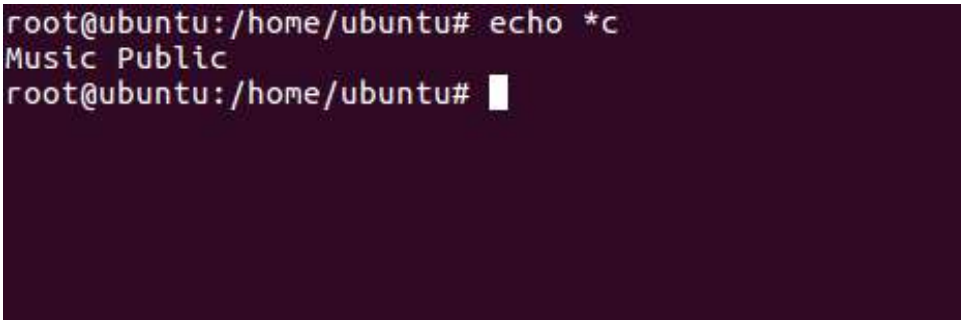
Como se puede ver en la demostración del comando anterior, todos los nombres de archivos del directorio actual coincidieron con la búsqueda debido a que no dimos un carácter en específico. La función de echo simplemente fue imprimir en pantalla los nombres que coincidieron.

Paso # 3.2: Otra forma para utilizar este carácter es colocar alguna letra, para que se muestran los nombres de directorios que empiezan por dicha letra. Digita los comandos echo D* y echo V*.



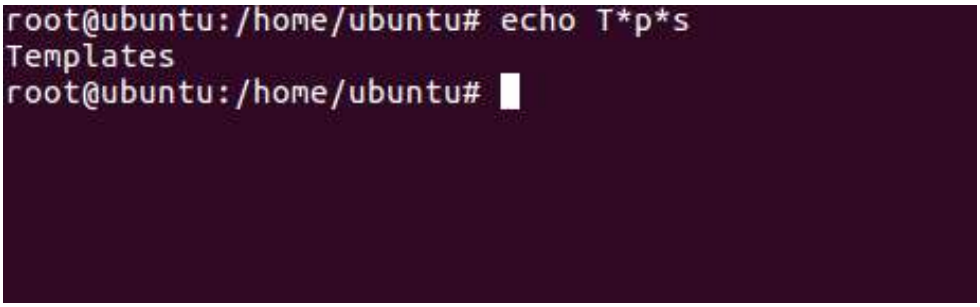
Figura# 10 echo D*

Paso # 3.3: También se puede utilizar este carácter para hacer una búsqueda distinta, por ejemplo, para mostrar los directorios que terminan en alguna letra. Digita en la terminal echo *c.



Figura# 11 echo *c

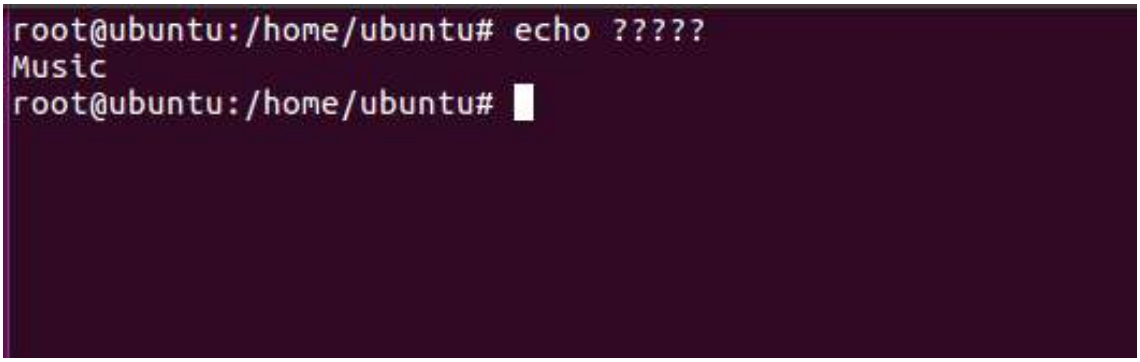
Paso # 3.4: Para hacer una búsqueda más precisa, el asterisco (*) puede aparecer más veces en medio de varios caracteres en una cadena. Introduce en la terminal echo T*p*s.



Figura# 12 echo T*p*s

Paso # 3.5

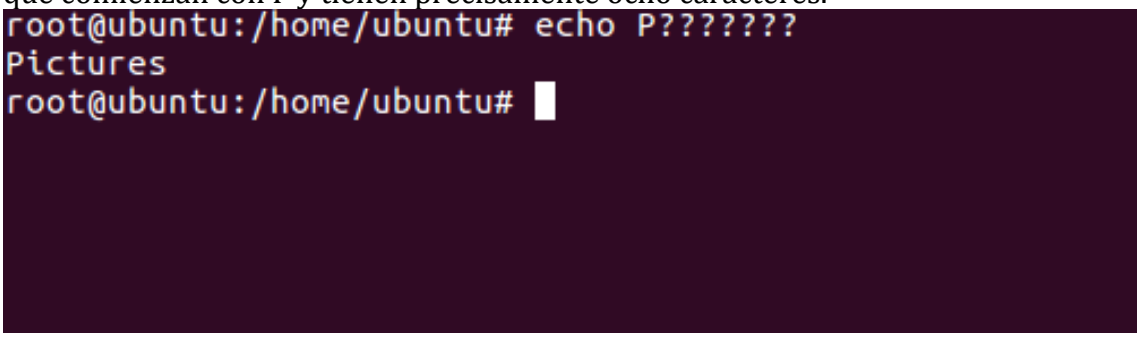
El siguiente metacaracter glob en explorar, aunque tiene una función distinta a las ya mencionadas, se utiliza de igual forma que el asterisco (*), en cualquier lugar de la instrucción del comando y puede aparecer varias veces para ser más específico. Este es el signo de interrogación (?), tiene la función de que cada signo coincida exactamente con un carácter.
Cómo cada sigo de interrogación (?), coincide con un carácter, introduciendo 5 de ellos, buscará los nombres de archivos con 5 caracteres. Introduce el comando **echo?????.**



Figura#13 echo ?????

Paso # 3.6

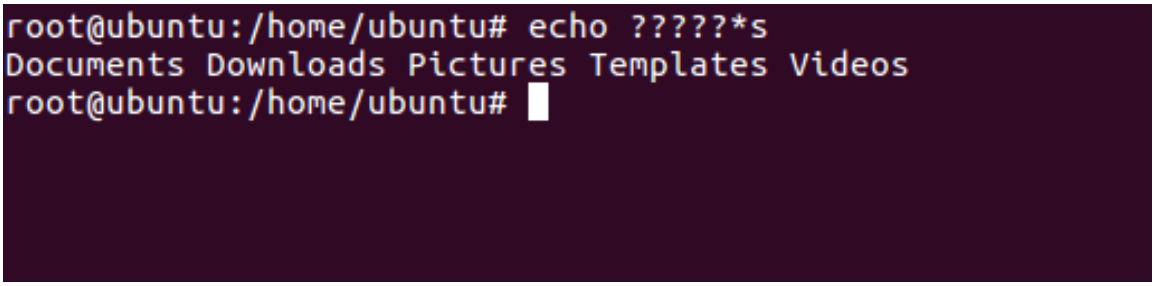
Si se quiere ser más específico en la búsqueda, puede combinarlos los signos de interrogación ? con otros caracteres, digita el comando echo P???????, para mostrar los nombres de archivos que comienzan con P y tienen precisamente ocho caracteres.



Figura#14echo P???????

Paso # 3.7

Estos caracteres glob también llamados “comodines”, se pueden combinar entre sí. Intenta el comando echo ?????*s, esto mostrará los nombres de archivo que tienen al menos seis caracteres y terminan en la letra s.



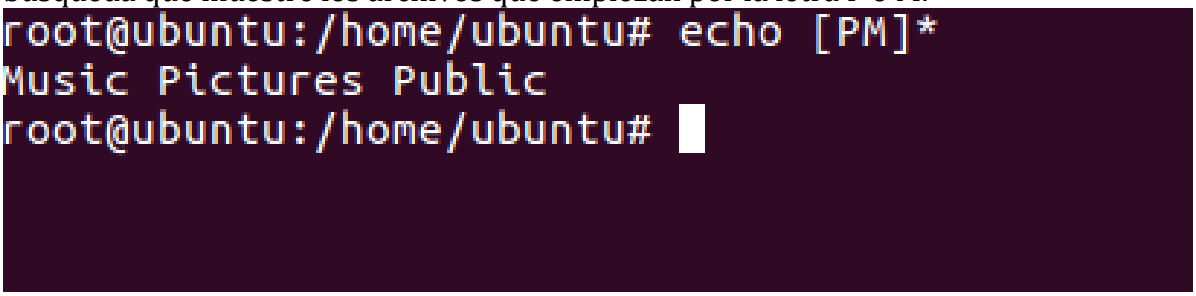
Figura# 15 echo ?????*s

Intérprete lo anterior como: Los primeros cinco “**?**” sirven para buscar archivos que comienzan con cualesquiera cinco caracteres y el “***s**” sirve para que seguidamente tenga cero o más caracteres y termine en s.

Paso # 3.8

El último carácter glob en explorar son el par de corchetes ([ ]), y tiene la función de especificar qué caracteres se le permitirán a una cadena, estos se pueden especificar con una serie o lista, o por el contrario puedes restringir caracteres incluyendo el signo de exclamación (!).

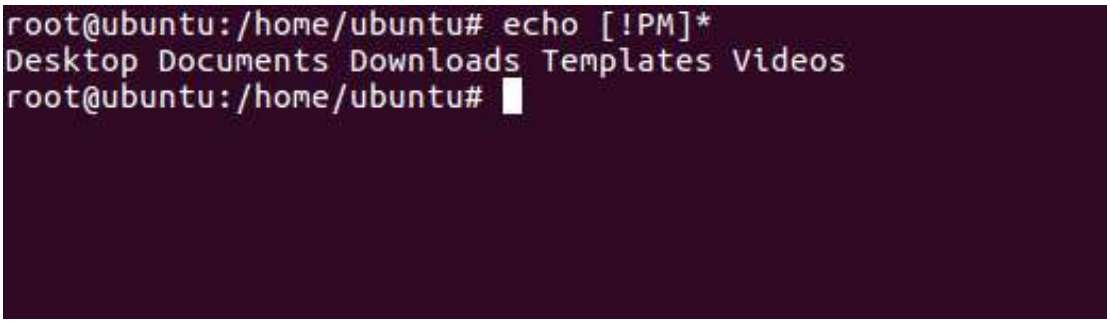
En este primer ejemplo se ejecuta el comando echo [PM]*, con este se especifica en la búsqueda que muestre los archivos que empiezan por la letra P o M.



Figura# 16 echo [PM]*

Paso # 3.9

Y en este segundo echo [! PM]*, buscamos lo contrario, es decir, el primer carácter no puede ser una P o M.



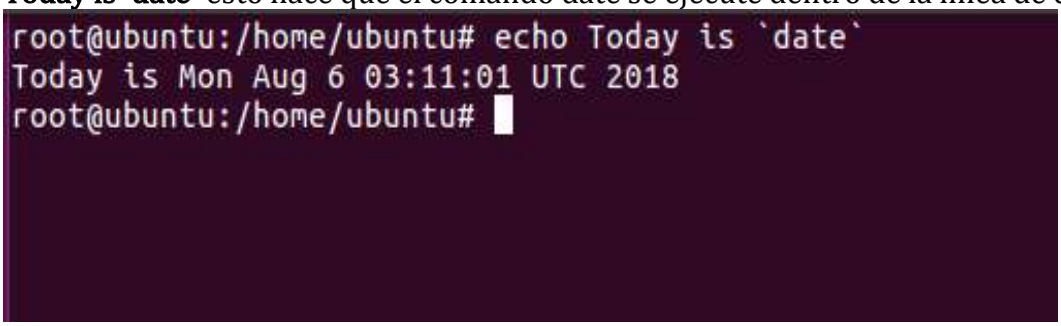
Figura# 17 echo [! PM]*

Paso # 4 Las funciones especiales de las comillas en la línea de comando de Linux.
El Shell Bash utiliza 3 tipos de comillas: simples ('), dobles (") e invertidas (`); cada una cuenta con características especiales según la acción que se quiera realizar.

Para entender la función de las comillas dobles y simples hay que pensar que no quieres que el Shell trate algunos caracteres como "especiales", como por ejemplo el asterisco (*) que se utiliza como carácter glob, pero la cuestión es ¿qué debemos hacer si se quiere utilizar simplemente como un carácter ordinario?Las comillas simples evitan que el Shell interprete estos caracteres como especiales, estas se utilizan con frecuencia para proteger una cadena de ser cambiada por el Shell.

Las dobles evitan que los caracteres glob (vistos en la sección anterior) y la "sustitución del comando" (ver comillas invertidas) se expandan y se interpreten como especiales. La invertidas provocan la "sustitución de comando", que causa la ejecución de un comando dentro de otra línea de comando. Cabe aclarar que las comillas se deben introducir en pares, de no ser así el sistema detectará el comando como incompleto.

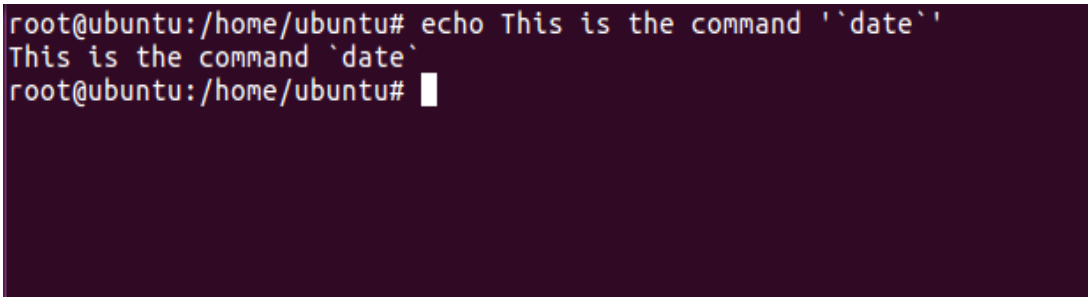
Paso # 4.1
Para empezar con la práctica, utilizaremos las comillas invertidas. Ejecuta el comando **echo Today is `date`** esto hace que el comando date se ejecute dentro de la línea de comando echo.



Figura#18. echo Today is `date`

Paso # 4.2

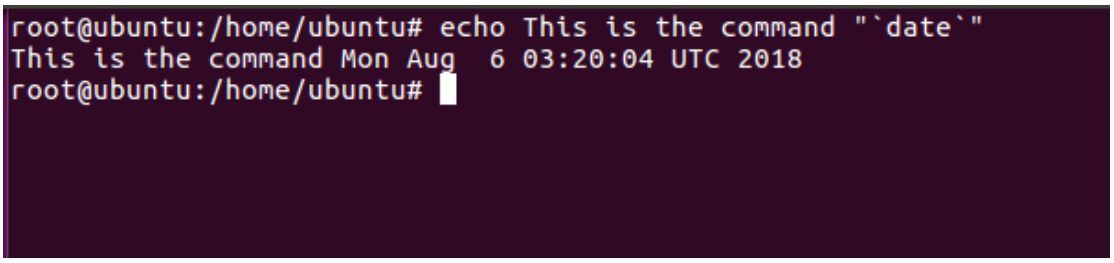
Si por el contrario no quieres que se ejecute el comando encerrado entre comillas invertidas, utiliza las simples, para encerrar la nueva instrucción y está la anulará. Introduce el commando **echo This is the command \`\`date\`\'**.



Figura# 19echo This is the command \`\`date\`\'

Paso # 4.3

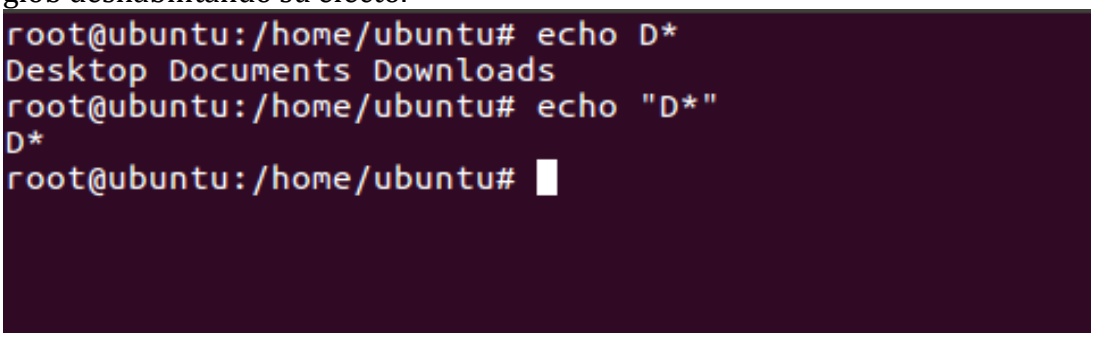
Si se encierran comillas invertidas entre comillas dobles las primeras no tendrán ningún efecto, debido a que el Shell las seguirá utilizando como una sustitución del comando. Digita en la terminal el comando **echo This is the command "\`date\`"**.



Figura#20 echo This is the command "\`date\`"

Paso # 4.4

A diferencia del comando anterior, en este se demostrará la utilidad de las comillas dobles. Ejecuta en la terminal echo D* y echo "D*", esto denota que tienen efecto sobre los caracteres glob deshabilitando su efecto.



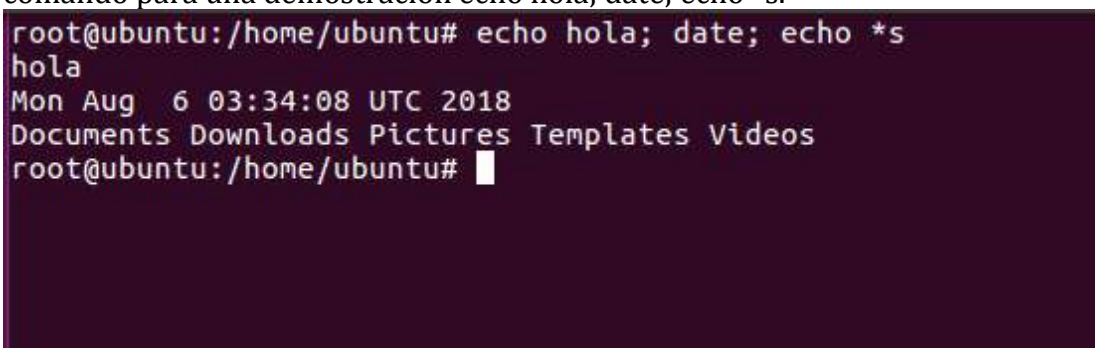
Figura#21 echo "D*"

Paso # 5 Las instrucciones de control para utilizar diversos comandos.

Es costumbre introducir y ejecutar un solo comando a la vez, pero el Shell Bash ofrece tres conectores para separar y escribir varios comandos es una misma línea.

Paso # 5.1

El punto y coma (;), será el primer separador en explorar, es el más simple, lo que permite es que al colocarlo entre múltiples comandos hará que se ejecuten uno tras otro de manera secuencial en el orden que estén en la línea de izquierda a derecha. Ejecuta el siguiente comando para una demostración echo hola; date; echo *s.



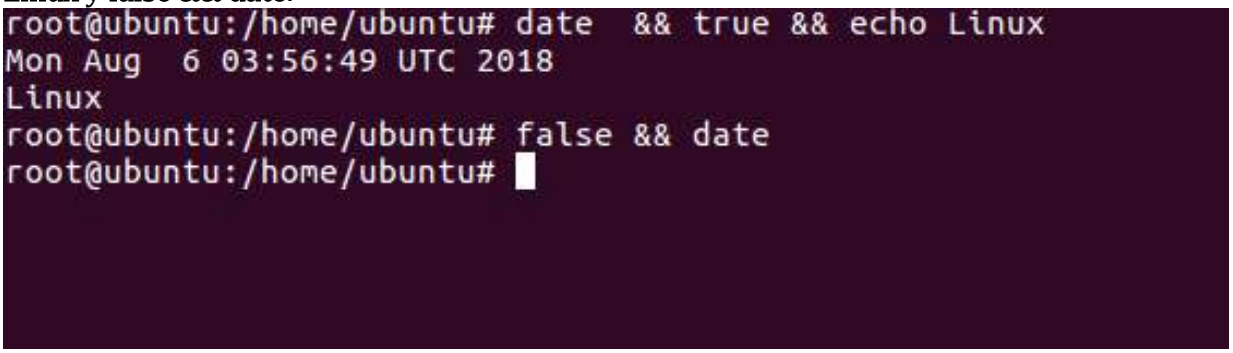
Figura#22 echo hola; date; echo *s

Los próximos separadores en conocer son conectores lógicos, y para efectos prácticos de su entendimiento se utilizarán dos ejecutables especiales, el **true** que emula el éxito de un comando y el **false** el fallo, debido a que por ahora no se pueden proporcionar ejemplos más realistas.

El siguiente separador en conocer serán los caracteres **&&**, estos crean una lógica condicional en la instrucción. Se dará el ejemplo con comando1 && comando2: si el comando 1 tiene éxito, el comando 2 se ejecuta, si por el contrario el comando 1 no tiene salida el comando 2 no se intentará ejecutar.

Paso # 5.2

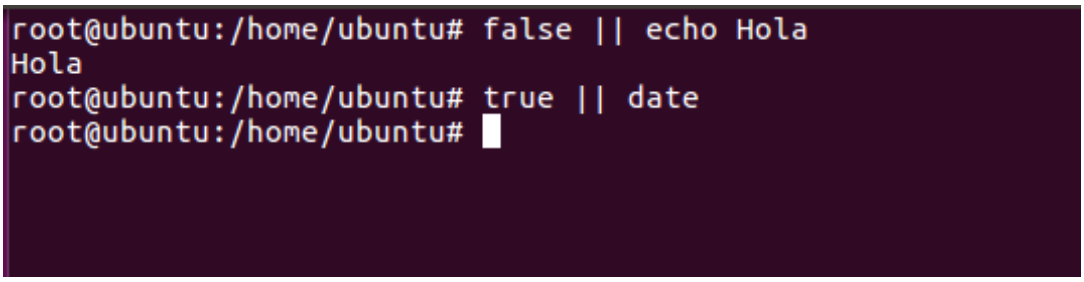
Ejecuta los siguientes comandos para ver el conector en la práctica **date && true && echo Linux** y **false && date**.



Figura#23 date && true && echo Linux

El último separador que se conocerá son los caracteres **||** y su lógica funciona así: comando1 **||** comando2, si el comando 1 falla el comando 2 se ejecuta y si por el contrario el comando 1 tiene éxito el comando 2 no se ejecuta

Paso # 5.3 introduce los siguientes comandos para ver cómo funciona **false || echo Hola** y **true || date.**



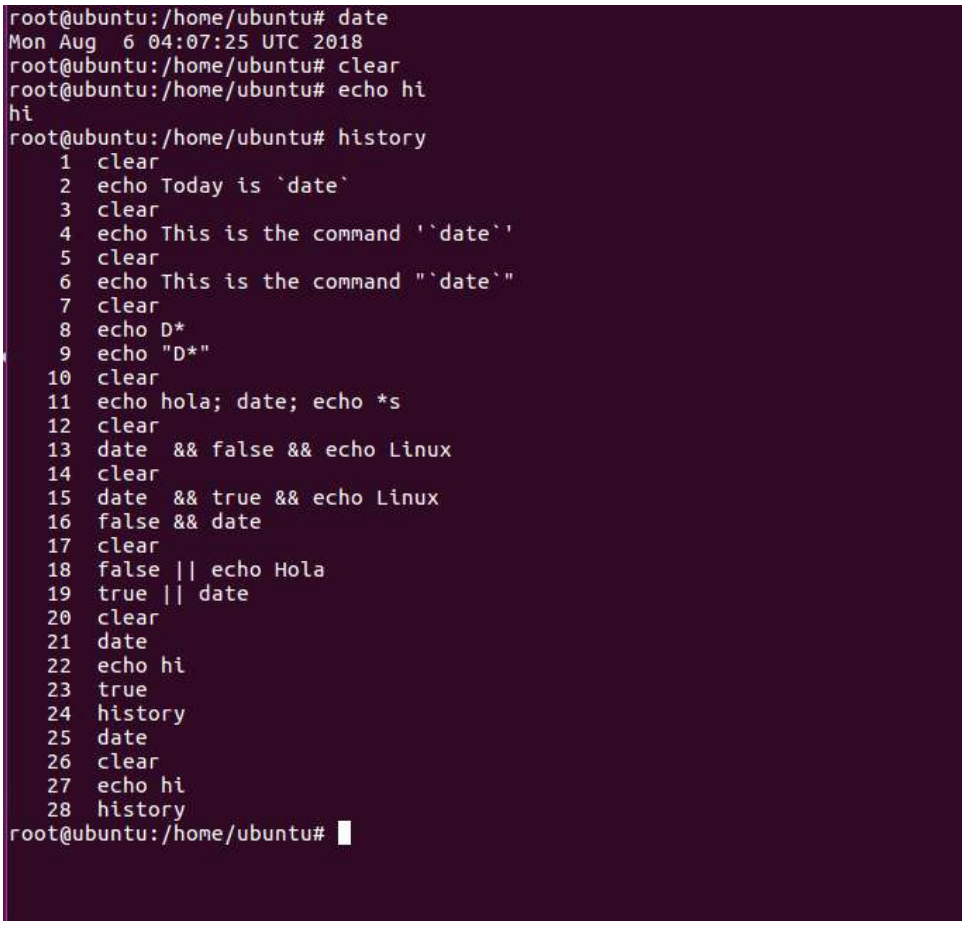
Figura# 24 false || echo Hola

Paso # 6 Cómo funciona y se utiliza el historial de Shell.

Así como un navegador, el Shell Bash tiene un historial para ver los comandos que se han ejecutado, este es accesible de varias maneras. La primera manera y más sencilla de llamar un comando, es con las flechas hacia arriba y abajo, de esta forma cada vez que presiones la flecha hacia arriba se devolverá un comando a través del historial y con la otra tecla irá hacia abajo. Una vez encontrado el comando requerido, puedes utilizar las flechas hacia la derecha o la izquierda para mover el cursor, editarlo y luego presionar enter para ejecutarlo. Otra forma de acceder al historial es ejecutar el comando **history**, y con este aparecerá un listado numerado de los últimos comandos introducidos en la terminal, el número que aparece a la izquierda de cada uno sirve para poder reutilizar el comando.

Paso # 6.1

Para iniciar la práctica de esta sección se ejecutarán algunos comandos (**date, clear y echo hi**) y por último se introducirá la instrucción **history**.

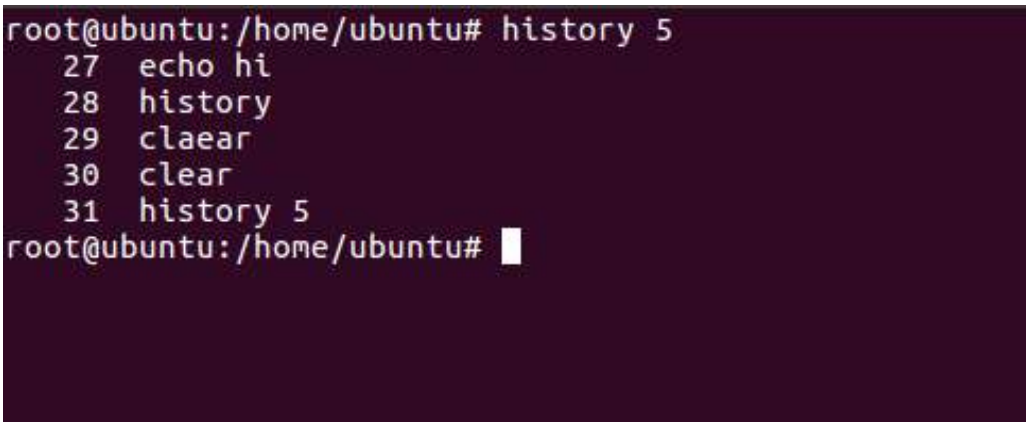


Figura#25 history

Cómo se puede ver, se observan algunos de los comandos ejecutados durante la práctica.

Paso # 6.2

También puedes darle a la terminal un parámetro, esto para limitar el número de comandos que quieres que liste el historial, el próximo comando le dará al sistema la indicación para que solo muestre los últimos 5 comando de tu historial **history 5**.



Figura# 26 history 5

Paso # 6.3

Ahora para ejecutar un comando determinado en la lista del historial debe introducir un signo de exclamación y el número que tiene el comando ejecuta **!**, por ejemplo haremos que nuevamente la terminal ejecute el comando 21, pon en la terminal **!21.**

```
root@ubuntu:/home/ubuntu# history
    1  clear
    2  echo Today is `date`
    3  clear
    4  echo This is the command '`date`'
    5  clear
    6  echo This is the command "`date`"
    7  clear
    8  echo D*
    9  echo "D*"
   10  clear
   11  echo hola; date; echo *s
   12  clear
   13  date  && false && echo Linux
   14  clear
   15  date  && true && echo Linux
   16  false && date
   17  clear
   18  false || echo Hola
   19  true || date
   20  clear
   21  date
   22  echo hi
   23  true
   24  history
   25  date
   26  clear
   27  echo hi
   28  history
   29  claear
   30  clear
   31  history 5
   32  clear
   33  history
root@ubuntu:/home/ubuntu# !21
date
Mon Aug  6 04:10:29 UTC 2018
root@ubuntu:/home/ubuntu#
```

Figura#!27 history

CAPÍTULO 2: BUSCANDO AYUDA

El objetivo:

Enseñar como la ayuda contextual que tiene el Sistema Operativo optimiza el trabajo con los comandos

Palabras clave: (**date, echo,**

En el entorno de Linux está claro que se puede manejar mediante comandos, y que generalmente eso es lo más divertido en el sistema operativo, a veces los cambios resultan entretenidos y así mismo, si eres una persona que apenas está comenzando a disfrutar el sistema, te surgirán preguntas donde tendrás que pedir ayuda. la idea de este capítulo es enseñarte a fortalecer la habilidad para buscar ayuda en este ámbito de comandos.

PRÁCTICA DE LABORATORIO DEL CAPÍTULO 2

Llevando a cabo la práctica de este laboratorio del capítulo 2. El usuario va a aprender a cómo En este capítulo verás los siguientes puntos:

1. Usaremos diferentes sistemas de ayuda para obtener ayuda en cuanto a comandos nos referimos
2. Aprenderemos a encontrar los comandos

Si apenas estás comenzando necesitas ayuda te recomiendo practicar los siguientes pasos:

Paso #1

Escribe un comando después debes presionar la tecla **Entrar**. Un ejemplo vamos a utilizar el comando **date** y el resultado va a hacer igual a este.

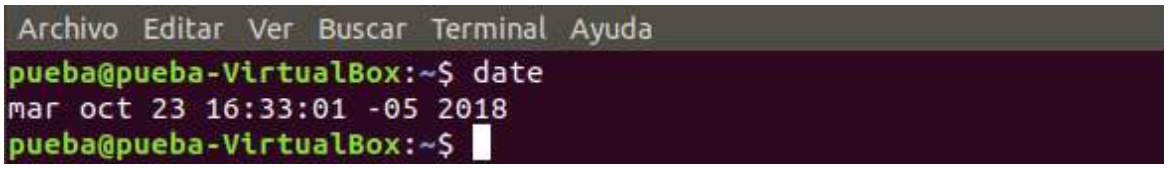


Figura # 28 fecha

Man en UNIX son una herramienta para aprender sobre los comandos, así mismo podemos aprender cómo podemos ejecutar un comando

Paso #2

En este caso utilizaremos este comando **echo** y si presionas la letra **h** recibirás más ayuda

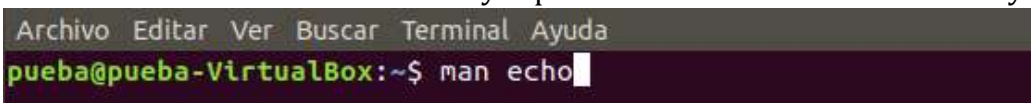


Figura #29 página del manual 2.1.2

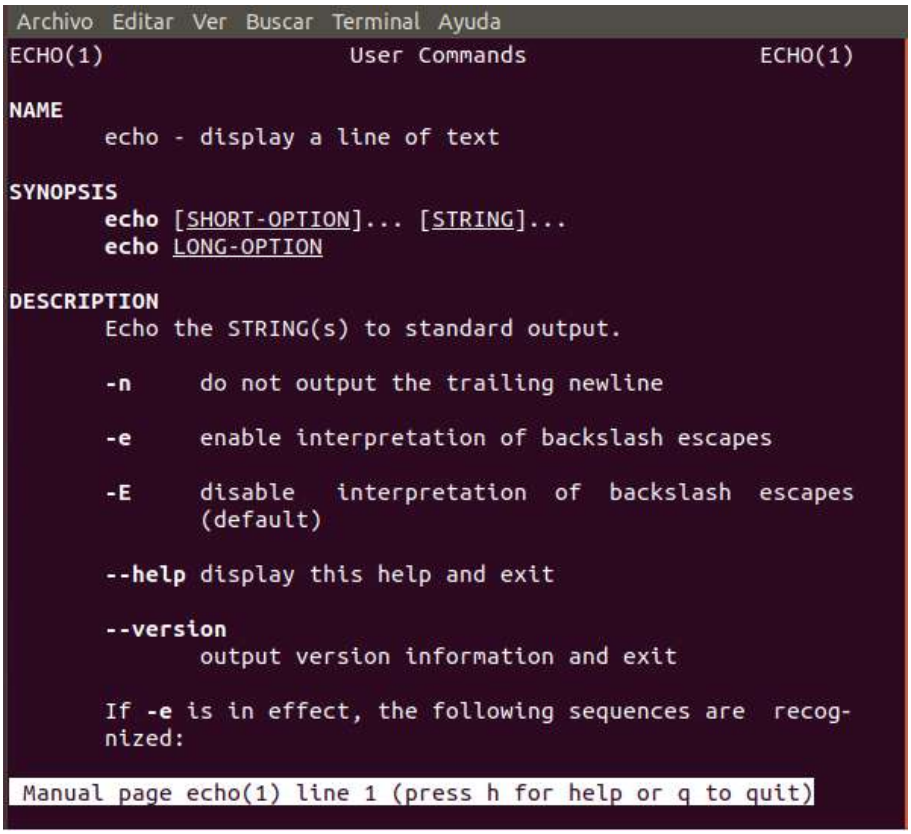


Figura # 30 página del manual para el echo 2.1.3

Tabla: Instrucciones para manejar la herramienta Man

TECLA	PROPÓSITO
Q o q	Salir de la ayuda o la página del manual
H o h	Mostrar ayuda
b o página arriba	Mover la pantalla hacia atrás
Entrar o flecha hacia abajo	Bajar una línea
Flecha hacia arriba	Mover una línea hacia arriba
F o página abajo o espacio	Mover hacia delante
/"palabra que se desea buscar"	Empieza a buscar hacia adelante
¿"palabra que se desea buscar"	Empieza a buscar hacia atrás
n	Ir al siguiente que coincida con la búsqueda
N	Mover al texto coincidente anterior

Una forma de visualizar la documentación que tiene un comando es con el comando **info -------** en el espacio va el comando.

Paso #3 Para este ejemplo usare el comando **echo**

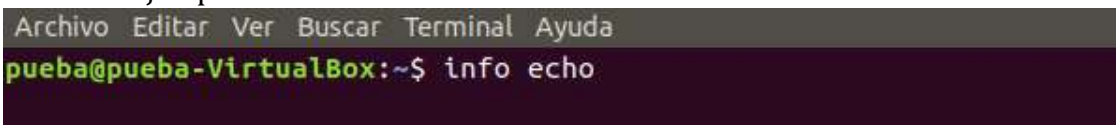


Figura #31 info echo

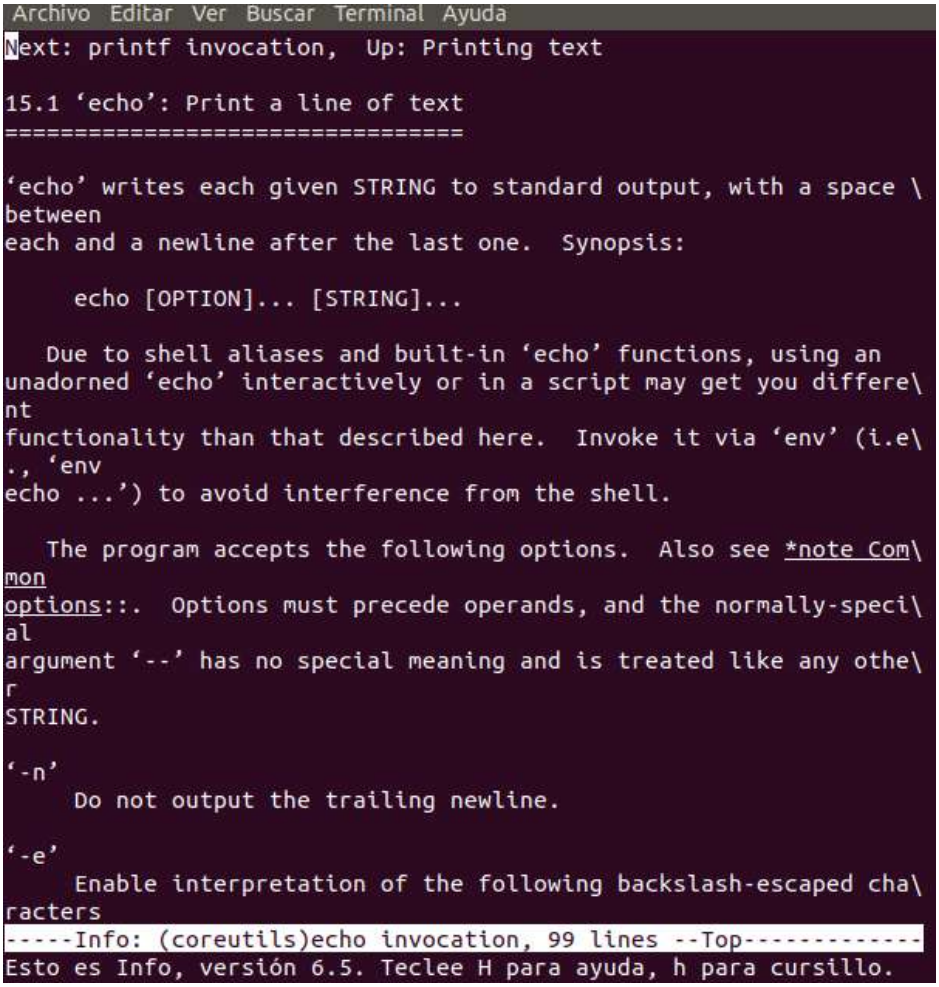


Figura #32 man 2.1.5

De ahí podemos sacar información de los comandos

Paso #4: Si alguna vez llegas necesitar más información que la que te brinda las páginas **man** podrías encontrar más en el directorio **/usr/share/doc** así:

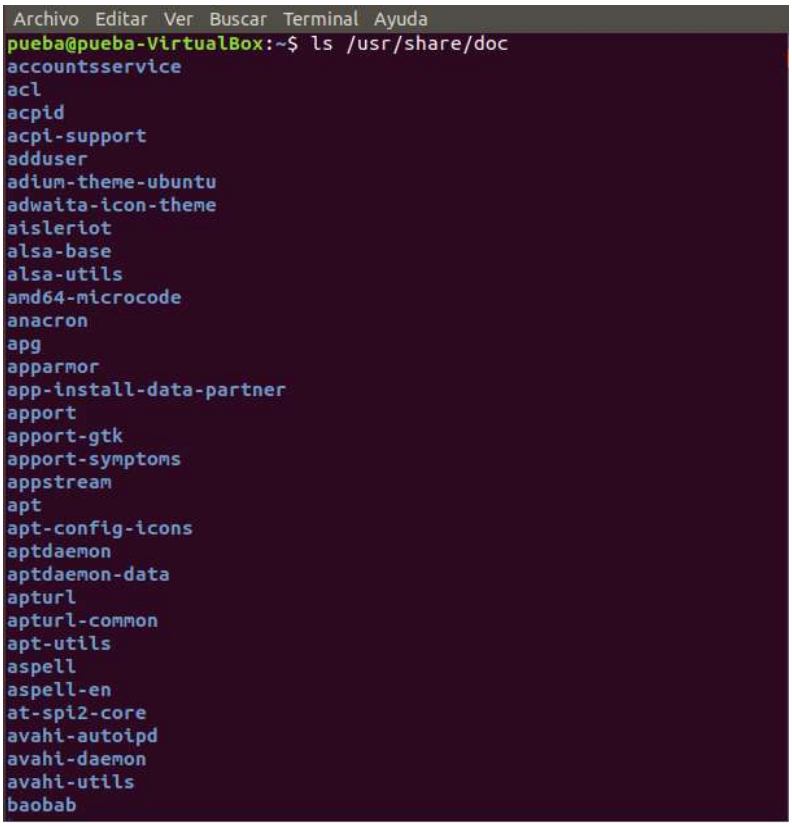


Figura # 32 lista el directorio

Búsqueda de archivos: Como el nombre lo indica en esta parte del capítulo te enseñaremos a cómo buscar un archivo ya sea del sistema o que haya creado, o que otra persona haya creado. La manera más fácil de buscar un archivo es el comando **locate**

Paso #5: Ejemplo voy a buscar el archivo **crontab**

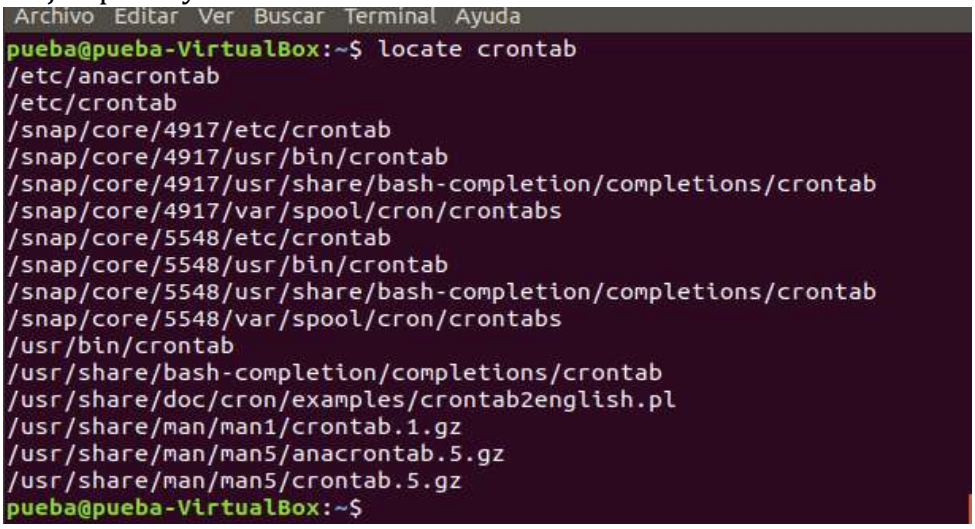


Figura #33 locate

Para encontrar los archivos que simplemente se llaman crontab

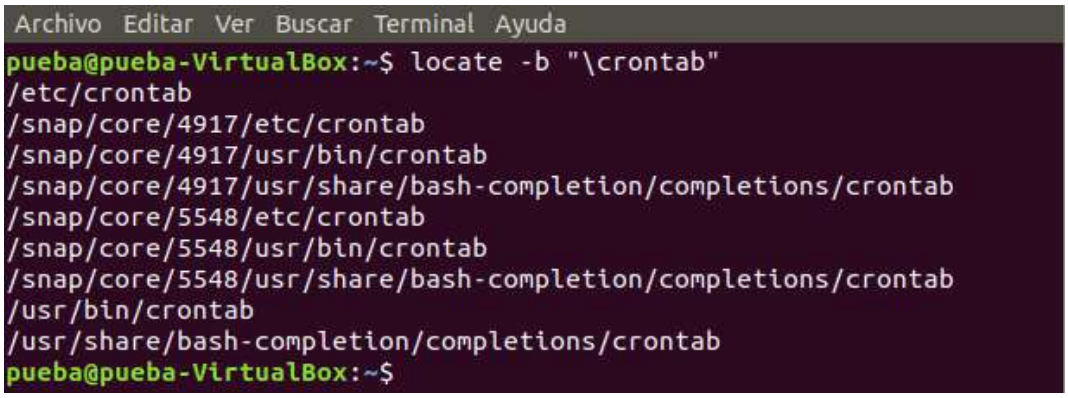


Figura #34 locate -b

Paso #6: Cuando solo quieres buscar donde se encuentra un comando para eso utilizamos el comando **whereis** y después el comando que queremos buscar, en este caso voy a buscar desde se encuentra el comando **passwd** así:

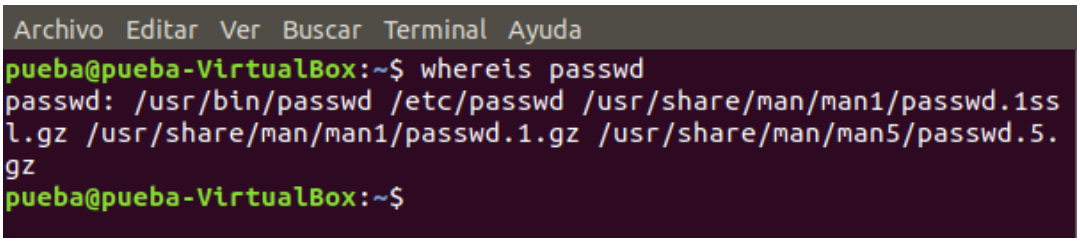


Figura #35 passwd

CAPÍTULO 3: INTERACCIÓN CON FICHEROS Y DIRECTORIOS EN LINUX

Palabras clave: (Rutas relativas, Rutas absolutas, Directorio, Directorio raíz, Directorio de trabajo actual, Shell.)

Introducción

Debido a que la mayoría de las personas están familiarizadas solamente con el sistema operativo de Windows, es importante aclarar que eliminar, copiar o mover algún archivo en este sistema operativo se hace mucho más fácil por decirlo así, debido a que es a lo que se está acostumbrado. En este capítulo se verá que en sistemas operativos basados en UNIX también se pueden realizar estas tareas, pero, la forma de llevarlas a cabo es completamente

diferente, debido a que en este sistema operativo estas acciones se llevan a cabo por medio de comandos, además de eso, se verá que estos comandos requieren saber la ubicación de los archivos (URL) .

Objetivo

Al finalizar esta práctica el aprendiz tendrá la capacidad de pasar de rutas relativas a rutas absolutas y viceversa, además de eso, sabrá también cómo visualizar sus rutas de trabajo actual y cómo trabajar en estas.

PRÁCTICA DE LABORATORIO DEL CAPÍTULO 3

Folios y ubicaciones.

Antes de iniciar, es importante decir que Linux distingue entre mayúsculas y minúsculas, es por ello por lo que a la hora de trabajar con los directorios y los archivos debemos tener bien claro cómo está nombrado cada objeto, es decir, **(RST**) no es lo mismo que (**rst**).

Paso 1- Para iniciar, vamos a ejecutar el comando pwd en la terminal para ver el directorio de trabajo.

Figura # 36 comando pwd

El comando anterior nos muestra la ruta o directorio de trabajo actual.
Paso 2- Si lo que se quiere es ver la ruta del directorio "Home" ingresamos el comando: **echo $HOME.**
Al ingresar este comando obtendremos algo similar a esto:

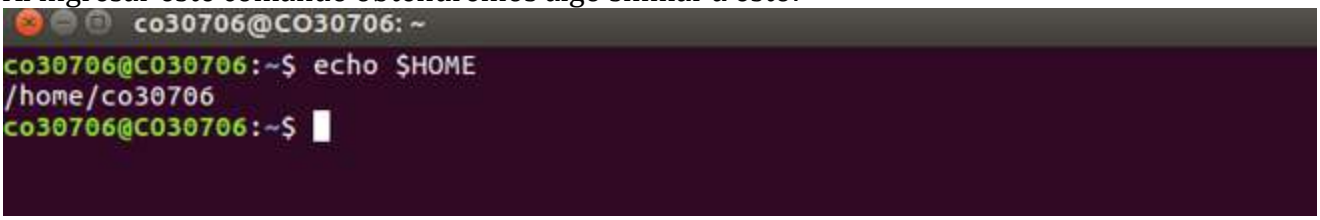


Figura 37: Comando echo $HOME

Se usa el comando **cd** con el **path** (variable de entorno o ruta de Linux) en un directorio para cambiar el directorio de trabajo actual.

Paso 3- Digita el comando: **cd /** Esto hará que el directorio raíz sea el directorio de trabajo actual, para comprobarlo digita el comando: **pwd**. Deberá salir algo similar a esto:

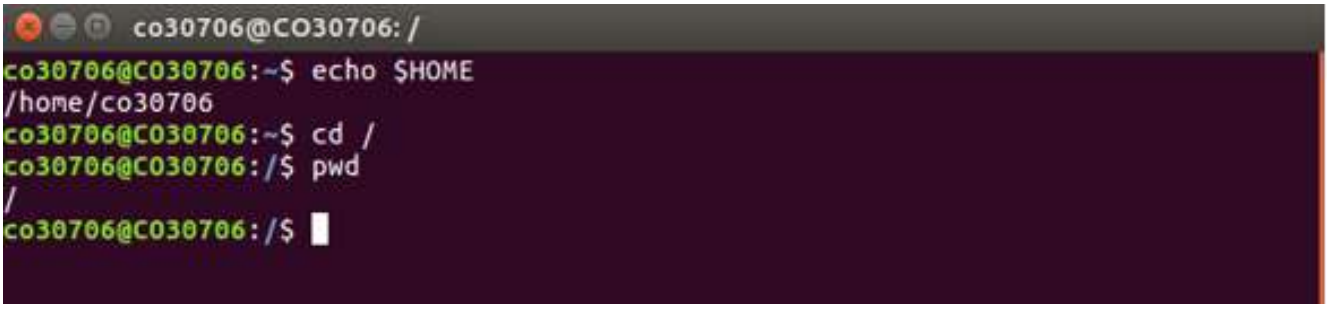


Figura 38: Comando cd /

Paso 4- Para regresar al directorio home ingresamos el comando **cd** que no necesita necesariamente una ruta para ser ejecutado y para comprobar que se ha vuelto al directorio home, ingresamos el comando **pwd** de nuevo, deberá salir algo como esto:

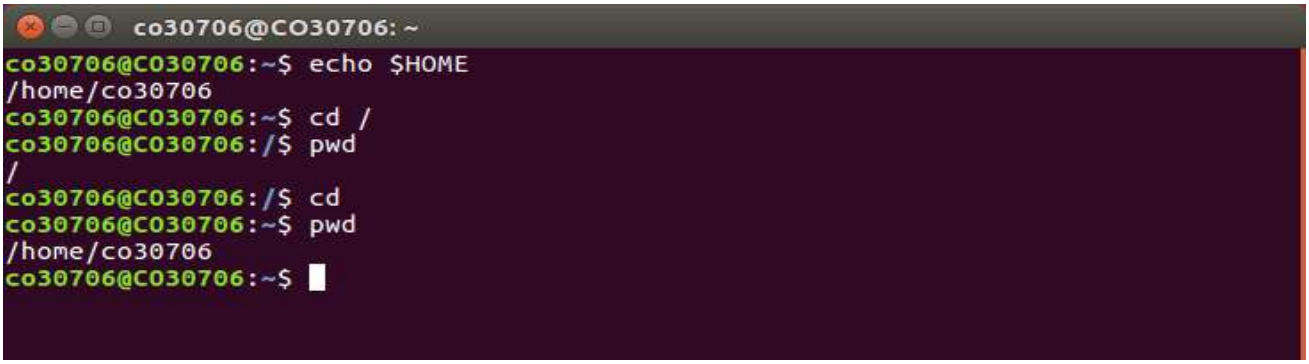


Figura 39: Comando cd

Por si no lo has notado, echa un vistazo a la estructura, se ha cambiado el **/$** por **~$,** el símbolo **(~)** representa el directorio home.

Paso 5-Ingresa el siguiente comando para tratar de cambiar el directorio home del usuario **root**: **cd ~root**

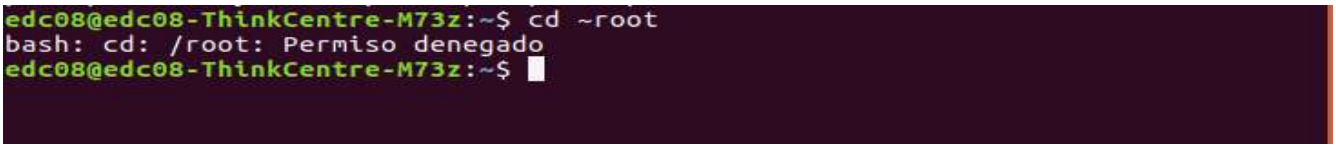
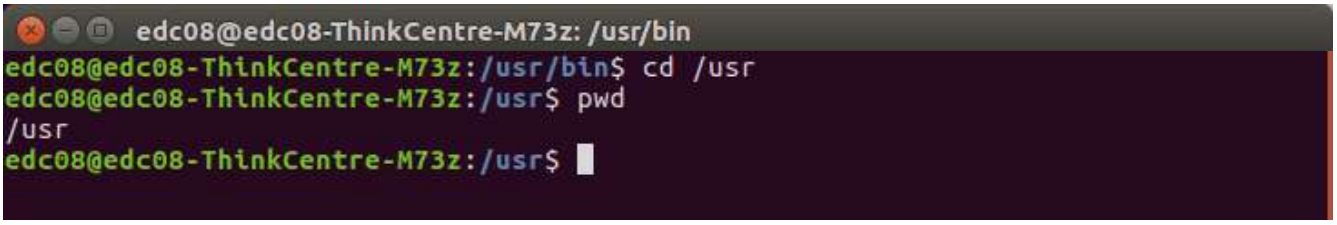


Figura 40: Comando cd ~root

Este mensaje de error que aparece en pantalla es debido a que el **shell** ha tratado de ejecutar **cd** con **/root** como argumento y no lo consiguió debido a la negación del permiso.

Paso 6-Pasa al directorio **cd /usr/bin** usando una **ruta absoluta**



Paso 7- Para cambiar al directorio **/usr**, usa el comando **cd /usr** usando una **ruta,** trabajo. Usa el **pwd** para visualizar la ruta de trabajo actual. **absoluta**.

Figura #41 Comando cd /usr/

Paso 8- Para cambiar al directorio **/usr/share/doc**, usa el comando **cd /usr/share/doc** con una ruta absoluta y observa el directorio de trabajo.

Figura #42 Comando cd /usr/share/doc

Nombre de rutas absolutas contra **rutas relativas**:
Debido a que escribir rutas extensas es algo tedioso, se sugiere utilizar nombres de rutas relativas como se verá en un siguiente ejemplo.

Paso 9- Para cambiar a directorio **/usr/share/doc/bash**, digita el comando **cd bash** usando una **ruta relativa** y observa el directorio.

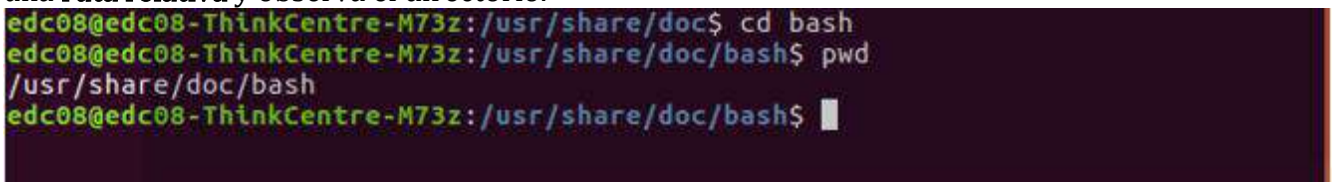
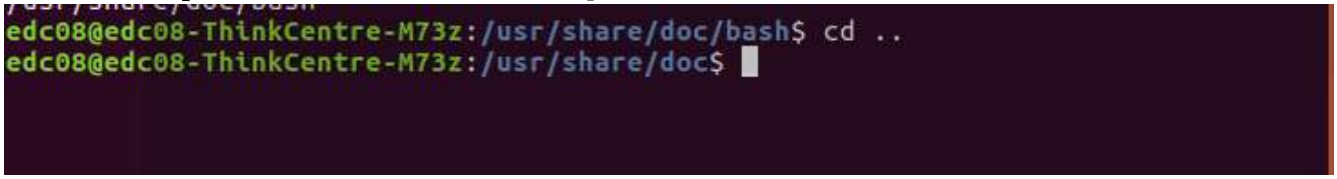


Figura #43: Comando cd Bash

Nota: Si no existiera un directorio **bash** en el directorio actual el comando usado no hubiera funcionado y nos arrojaría un error.

Paso 10- Para cambiar el directorio por encima del directorio actual usa una **ruta relativa**:
El comando **cd ..** ayudará para esto.

Los **puntos** simbolizan un nivel superior a la ubicación del directorio actual.



Figura# 44: Comando cd ..

Paso 11- Para cambiar a un nivel superior y luego a un nivel inferior **dict** desde el directorio actual usa una ruta relativa con el comando **cd ../dict**

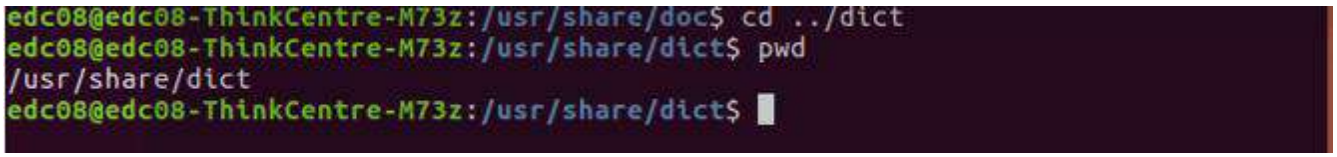


Figura # 45: Comando cd ../dict

CAPÍTULO 4: COMPRESIÓN DE ARCHIVOS

Introducción

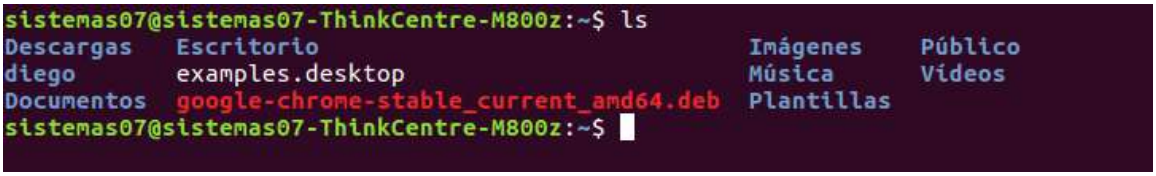
La compresión de archivos es una función que se ejecuta en todos los sistemas operativos existentes, es tan así que esta función es primordial en el momento de mover archivos entre sistemas operativos, por ejemplo, en Windows se utiliza mucho la extensión .rar y en sistemas operativos basados en UNIX su extensión típica es el .zip, la comprensión crear archivos más livianos en términos de memoria para así lograr manipularlos.

Objetivo: Conocer la función principal de la compresión de archivos.

palabras claves: (Ls, mkdir, cd, cat, wget, zip, gunzip.)

PRACTICA DE LABORATORIO DEL CAPÍTULO 4

Paso 1 para comenzar con el capítulo chequeamos las carpetas existentes en nuestro directorio. Usando el comando **ls**.



Figura#46: listar carpetas

EL comando **ls** lista todas las carpetas que tengo en mi directorio, en este caso se tiene 8 carpetas: Descargas, diego, Documentos, Imágenes, Público, Música, vídeos y Plantillas.

Paso 2 a continuación crearemos una nueva carpeta en la cual almacenaremos algunos archivos para luego proceder a comprimir. Usaremos el comando mkdir que viene del inglés **make directory**.

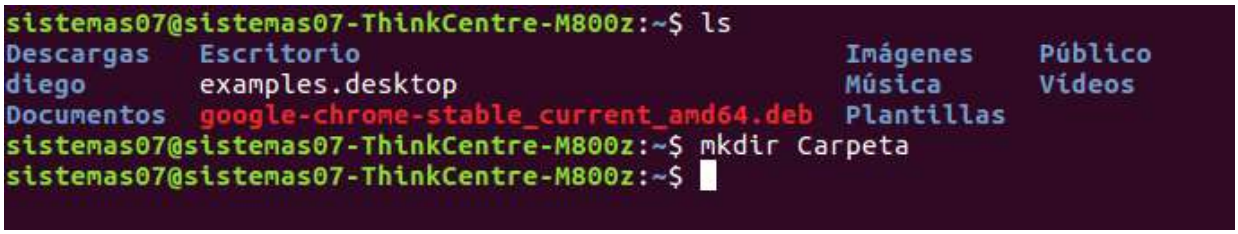
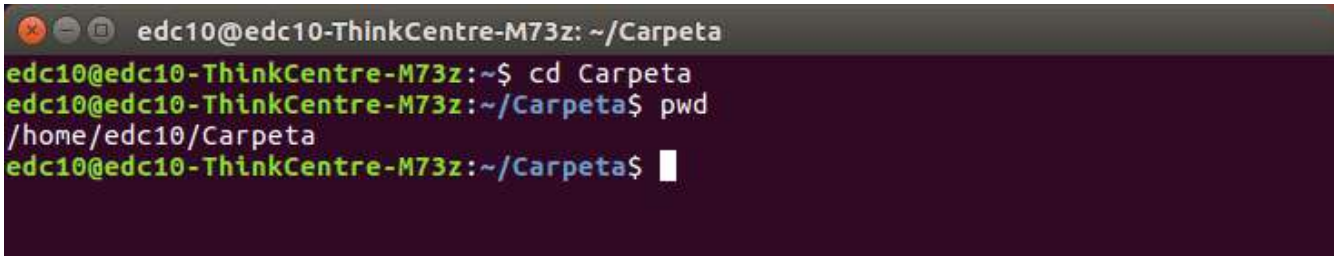


figura # 47: crear carpeta

Utilizando el comando **mkdir** creamos la carpeta con el nombre que queramos, luego volvemos a listar y podemos comprobar que la carpeta fue creada.

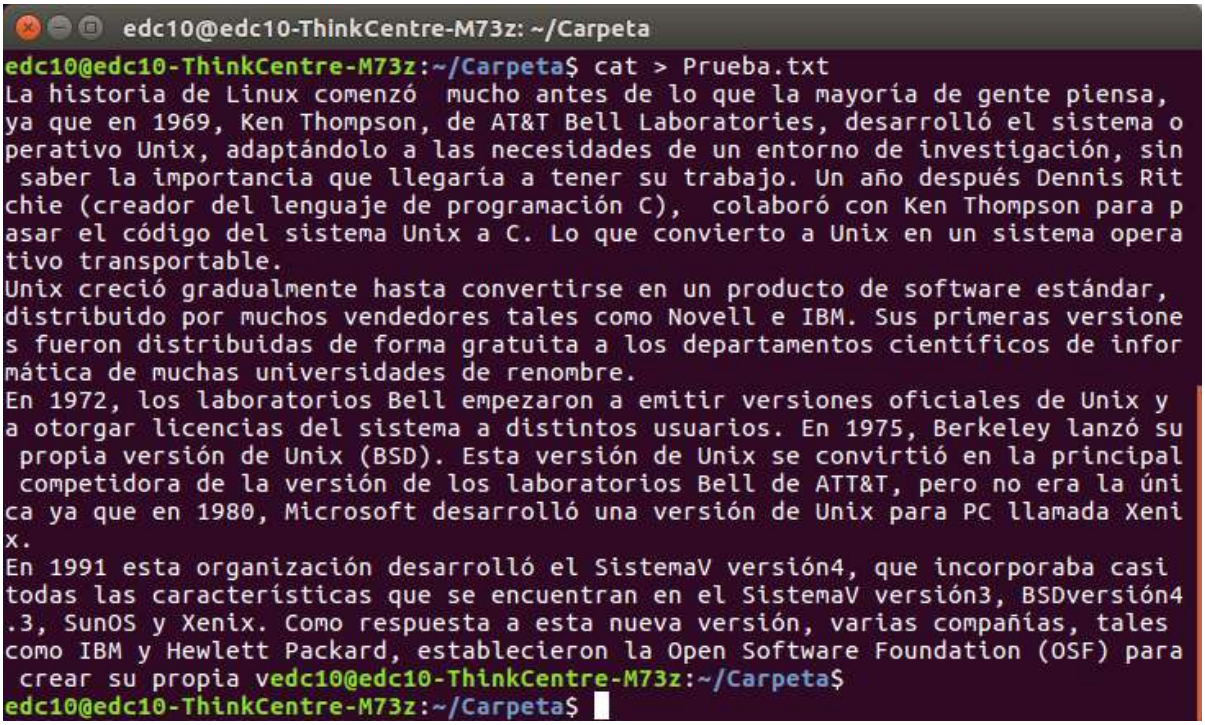
Figura # 48 carpeta creada

Paso 3: Utilizando el comando CD que es una abreviación de **change directory ,** cambiamos de directorio es decir cambiamos a la carpeta o directorio que queramos, en este caso cambiaremos al directorio Carpeta.



Figura#49: cambio de directorio

paso 4 A continuación, procederemos a crear un archivo de texto, el comando a usar será **cat > nombre del archivo** con extensión .**txt**.



Figura#50: creación de archivo de texto

paso 5 Podemos verificar que el archivo fue creado con solo listar la carpeta, luego procedemos a descargar una imagen con ayuda del comando **wget -O nombre.png http: // link completo de la imagen**, en este caso descargamos una imagen de los sistemas operativos existentes el cual nombraré sistemas.png, tenga en cuenta que el link debe ser el link de la imagen y debe mostrar la extensión de la imagen.

Figura #51: descarga de imagen

paso 6 Ahora procedemos a comprimir la carpeta para eso utilizaremos el comando **zip -r file.zip nombre del archivo a comprimir.**

```
edc10@edc10-ThinkCentre-M73z:~/Escritorio/Carpeta$ cd ..
edc10@edc10-ThinkCentre-M73z:~/Escritorio$ zip -r Carpeta.zip Carpeta
  adding: Carpeta/ (stored 0%)
  adding: Carpeta/linux.txt (deflated 48%)
  adding: Carpeta/sistemas.png (deflated 2%)
edc10@edc10-ThinkCentre-M73z:~/Escritorio$ ls -lh
total 276K
drwxrwxr-x 2 edc10 edc10 4,0K ago  3 11:36 Carpeta
-rw-rw-r-- 1 edc10 edc10 270K ago  3 11:37 Carpeta.zip
edc10@edc10-ThinkCentre-M73z:~/Escritorio$
```

Figura # 52: comprimir carpeta

Existen otros tipos de comprimir archivos y los formatos usados son bzip2, gzip etc. Estos son algunos de los que se abordarán en este tema.

paso 7 Usando el comando **gzip nombre**, comprimimos un archivo

```
edc10@edc10-ThinkCentre-M73z: ~/Escritorio
edc10@edc10-ThinkCentre-M73z:~/Escritorio$ cat > archivo.txt
En Linux hay diversas herramientas para empaquetar y comprimir archivos, tomando
 en cuenta que empaquetar es juntar dos o más archivos en un solo archivo (paque
te) y comprimir es tomar este archivo-paquete y comprimirlo a continuación te mu
estro un resumen de las más comunes, de acuerdo a la extensión que comunmente se
 acostumbra ponerles.
edc10@edc10-ThinkCentre-M73z:~/Escritorio$ gzip archivo.txt
edc10@edc10-ThinkCentre-M73z:~/Escritorio$ ls
archivo.txt.gz  carpeta  Carpeta  carpeta.tar.gz  carpeta.zip  Prueba  prueba1
edc10@edc10-ThinkCentre-M73z:~/Escritorio$
```

Figura# 53 comprimir archivo

en la imagen anterior se crea un archivo con extensión .txt y luego se le aplica el comando gzip para comprimir.

paso 8 Para descomprimirlo usamos el comando **gunzip.**

```
edc10@edc10-ThinkCentre-M73z: ~/Escritorio
edc10@edc10-ThinkCentre-M73z:~/Escritorio$ gunzip archivo.txt.gz
edc10@edc10-ThinkCentre-M73z:~/Escritorio$ ls
archivo.txt  carpeta  Carpeta  carpeta.tar.gz  carpeta.zip  Prueba  prueba1
edc10@edc10-ThinkCentre-M73z:~/Escritorio$
```

Figura#54: descomprimir archivo

CAPÍTULO 5: EMPAQUETAMIENTO Y COMPRESIÓN DE ARCHIVOS

Cuando hablamos de empaquetamiento y compresión de archivos inmediatamente pensamos en clic derecho y comprimir, esto se debe a la dependencia que poseemos por los sistemas operativos Windows, en este capítulo se hablará de cómo es el empaquetamiento y la compresión de archivos a través de sistemas operativos basados en UNIX se convierte en una metodología muy distinta ya que no tenemos pocos conocimientos en este sistema operativo, cuando nos referimos a empaquetar un archivo se alude a combinar varios archivos en uno solo mientras que compresión se refiere a hacer el archivo más pequeño eliminando información redundante, saber cómo utilizar estos métodos puede ayudar a minimizar el uso de memoria en nuestro sitio de trabajo y a la hora de usar nuestros archivos.

Objetivo: Conocer la funcionalidad del empaquetamiento de archivos.

palabras claves: (mkdir, tar, cd .., bzip2, gzip)

PRACTICA DE LABORATORIO DEL CAPÍTULO 5

En esta sección de la práctica veremos cómo empaquetar archivos, el empaquetamiento nos permite crear una copia o contener varios archivos en uno solo,se utilizará el comando tar.

paso 1 en el escritorio crearemos una carpeta la cual la llamaremos Prueba o el nombre que deseen, recordemos que para crear carpetas usaremos el comando **mkdir** el resultado debe ser el siguiente.

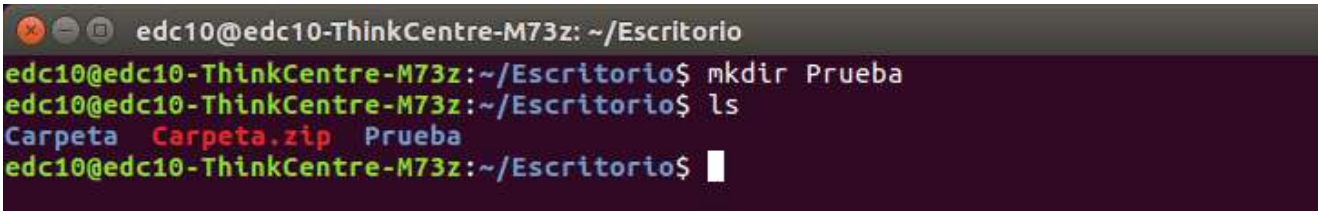
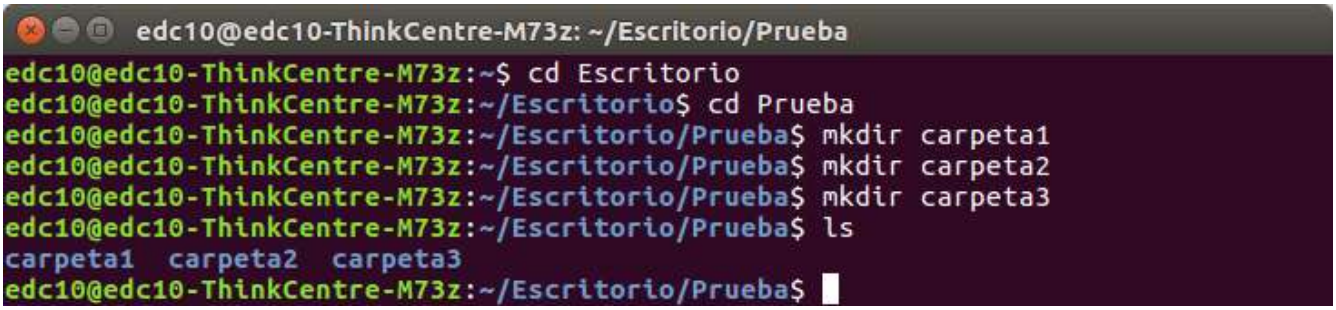


Figura #55 crear la carpeta prueba

paso 2 Como vemos se creó la carpeta llamada Prueba, ahora ingresamos en la carpeta usando el comando CD y ya dentro de ella crearemos otras 3 carpetas con los nombres **carpeta-1, carpeta-2 y carpeta-3.**



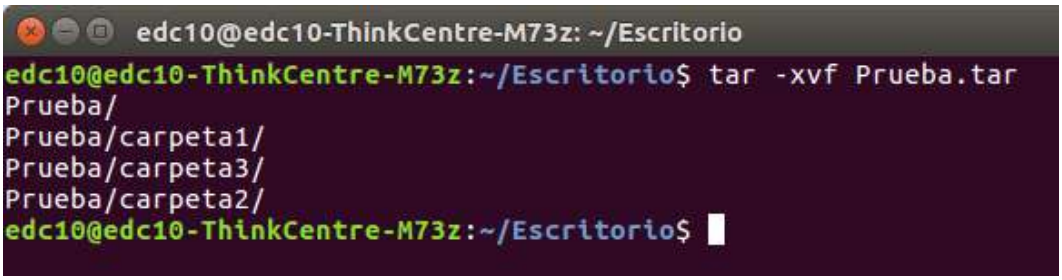
Figura# 56 crear las 3 carpetas

paso 3 Al crear las carpetas evidenciamos algo parecido a la imagen anterior, a continuación vamos a salir del directorio Prueba usando el comando **cd ..**, y comenzaremos con la empaquetado del archivo.

paso 4 Con el comando **tar -cvf file.tar nombre del directorio a empaquetar**, este será el comando a usar, **c** le indica a **tar** que va a crear un nuevo archivo, **v** le dice a tar que muestre lo que está haciendo y **F** le dice el nombre que tendrá ese archivo empaquetado con extensión **.tar**.

figura#57: empaquetamiento

paso 5 Como podemos ver el archivo ha sido empaquetado y el resultado debe ser igual a la imagen anterior, ahora procederemos a desempaquetar el archivo el cual usaremos el comando **tar -xvf Prueba.tar**, donde **x** le indica a **tar** que va a desempaquetar el fichero **Prueba.tar**, **v** le indica a tar que muestre lo que va a desempaquetar y le indica el nombre del archivo que va a desempaquetar.



figura#58: desempaquetar carpeta

paso 6 El comando tar también tiene una propiedad que nos ayuda a comprimir sin necesidad de utilizar los comandos **gzip, zip, bzip2**. En este caso invocamos la funcionalidad de **gzip** a través del comando **-z**.

Por ejemplo:

1. Crear un directorio nuevo en el escritorio llamado capeta.
2. Luego ingresaremos el comando tar -czvf carpeta/nombre.tar.gz directorio donde el nombre será el que se le desea poner y carpeta será el lugar donde alojaremos el archivo y directorio será la carpeta que queremos comprimir y empaquetar.
3. Entraremos en la carpeta creada y listamos

figura#59 empaquetar y comprimir una carpeta,

paso 7 cómo se puede evidenciar el archivo fue comprimido y a la vez empaquetado dentro del directorio carpeta, ahora procedemos a desempaquetar el cual usaremos la función **-x** el cual le dice al comando tar que desempaquete y descomprima el archivo que le asignemos, el comando a usar es **tar -xvf nombre.tar.gz**, el resultado será.

figura#60 desempaquetar y descomprimir carpeta

Al listar el contenido evidenciamos que el archivo se desempaqueto y se descomprime y su contenido está en el directorio llamado carpeta.

CAPÍTULO 6: COMPRENDIENDO EL HARDWARE.

Palabras clave (arch, lscpu, mainboard, lsusb, dmidecode)

Introducción

No es un secreto para nadie que en estos tiempos existen múltiples marcas de computadores, todas y cada una de estas empresas tiene en sus diversos productos tipos de hardware y cada serie de componentes tienen unas características especiales, dependiendo estas últimas de la necesidad para el usuario; en esta sección se hará un breve toque sobre algunos puntos importantes y aprenderemos en la terminal de Linux como acceder a estas características por medio de los comandos.

Una de las principales características en la computadora es el procesador y de él se desprenden diferentes fabricantes sin embargo lo que necesitamos de este es su arquitectura (sabiendo que x86 será un procesador a 32 bits por lo contrario un procesador de x86_64 será capaz de trabajar a 32 y 64 bits), para conocer esta información requerimos ejecutar el comando "**arch**", sin embargo este solo nos mostrará su arquitectura, para sacar más provecho de la terminal podremos ejecutar "**lscpu**" a diferencia del comando anterior este nos mostrará no solo su arquitectura sino su capacidad de proceso ya sea esta de 32 o 64, el propósito de esta información será para saber qué software podremos usar ya que algunos solo funcionan con un procesador de x86_64.

Parte importante de nuestra tarjeta madre (**mainboard**) es el tamaño de su RAM (**Random** Access Memory o memoria de acceso aleatorio, en español) la cual guarda los procesos momentáneos que estemos haciendo en la computadora, para saber su capacidad de almacenamiento ejecutaremos el comando "free" sin embargo su resultado serán cifras demasiado largas para hacer una conversión o verlo de una forma más "humana" ejecutaremos el mismo comando seguido de un "- g" de manera tal que estos valores se mostrarán en gigabytes.

Puede ser que la capacidad de puertos USB en nuestra computadora sea mucha por lo cual y para saber cuántos dispositivos están conectados a ella el comando "**lsusb**" será útil para esto.

Objetivo: identificar los componentes físicos con el cual interacciona el sistema operativo

PRACTICA DE LABORATORIO DEL CAPÍTULO 6

Paso #1 dmidecode

El siguiente comando nos permitirá evitar en lo posible desmontar por completo nuestra computadora ya que este nos mostrará información valiosa de la Basic Imput and Output System o más conocida como BIOS, la cual contiene la vida del pc. Para adquirir dicha información se escribe en la terminal el comando "**dmidecode"**, y nos mostrar algo como esto: comando: **dmidecode**

```
root@edc06-ThinkCentre-M73z:/home/edc06# dmidecode
# dmidecode 3.0
Getting SMBIOS data from sysfs.
SMBIOS 2.8 present.
83 structures occupying 1762 bytes.
Table at 0xDBEC6018.

Handle 0x0000, DMI type 0, 24 bytes
BIOS Information
        Vendor: LENOVO
        Version: FGKT29AUS
        Release Date: 03/19/2014
        Address: 0xF0000
        Runtime Size: 64 kB
        ROM Size: 4096 kB
        Characteristics:
                PCI is supported
                BIOS is upgradeable
                BIOS shadowing is allowed
                Boot from CD is supported
                Selectable boot is supported
                BIOS ROM is socketed
                EDD is supported
                5.25"/1.2 MB floppy services are supported (int 13h)
                3.5"/720 kB floppy services are supported (int 13h)
                3.5"/2.88 MB floppy services are supported (int 13h)
                Print screen service is supported (int 5h)
                8042 keyboard services are supported (int 9h)
                Serial services are supported (int 14h)
                Printer services are supported (int 17h)
                ACPI is supported
                USB legacy is supported
                BIOS boot specification is supported
                Targeted content distribution is supported
                UEFI is supported
        BIOS Revision: 1.29

Handle 0x0001, DMI type 1, 27 bytes
System Information
        Manufacturer: LENOVO
        Product Name: 10BBA0B000
        Version: ThinkCentre M73z
        Serial Number: MJ011091
        UUID: D2000800-F432-11E3-894B-0A7BE1BF1000
        Wake-up Type: Power Switch
        SKU Number: LENOVO_MT_10BB
        Family: To be filled by O.E.M.

Handle 0x0002, DMI type 2, 15 bytes
Base Board Information
        Manufacturer: LENOVO
```

Figura #61, información del pc.

De este comando podremos sacar bastante provecho y aún más si por necesidad se requiere hacer un inventario de las computadoras que hay en nuestro dominio, ya que arroja fecha de procedencia, fabricante, capacidad de almacenamiento u otros.

Paso #2 arch

Para conocer la estructura de nuestro pc o su familia, basta con ejecutar "**arch**", obteniendo algo así:

Comando: **arch**

```
root@edc-OptiPlex-9010-AIO:/home/edc# arch
x86_64
root@edc-OptiPlex-9010-AIO:/home/edc#
```

figura #62, ejecutar arch.

De esta manera se puede ver claramente qué tipo de arquitectura tiene el equipo. No obstante, para no quedarse cortos de información y para un análisis más completo con "**lscpu**" podremos ver muchas más características, tales como a cuantos bits trabaja; su fabricante e incluso el tipo de procesador que contiene entre otras más cosas, en la siguiente imagen se puede ver más claramente:

Comando: **lscpu**

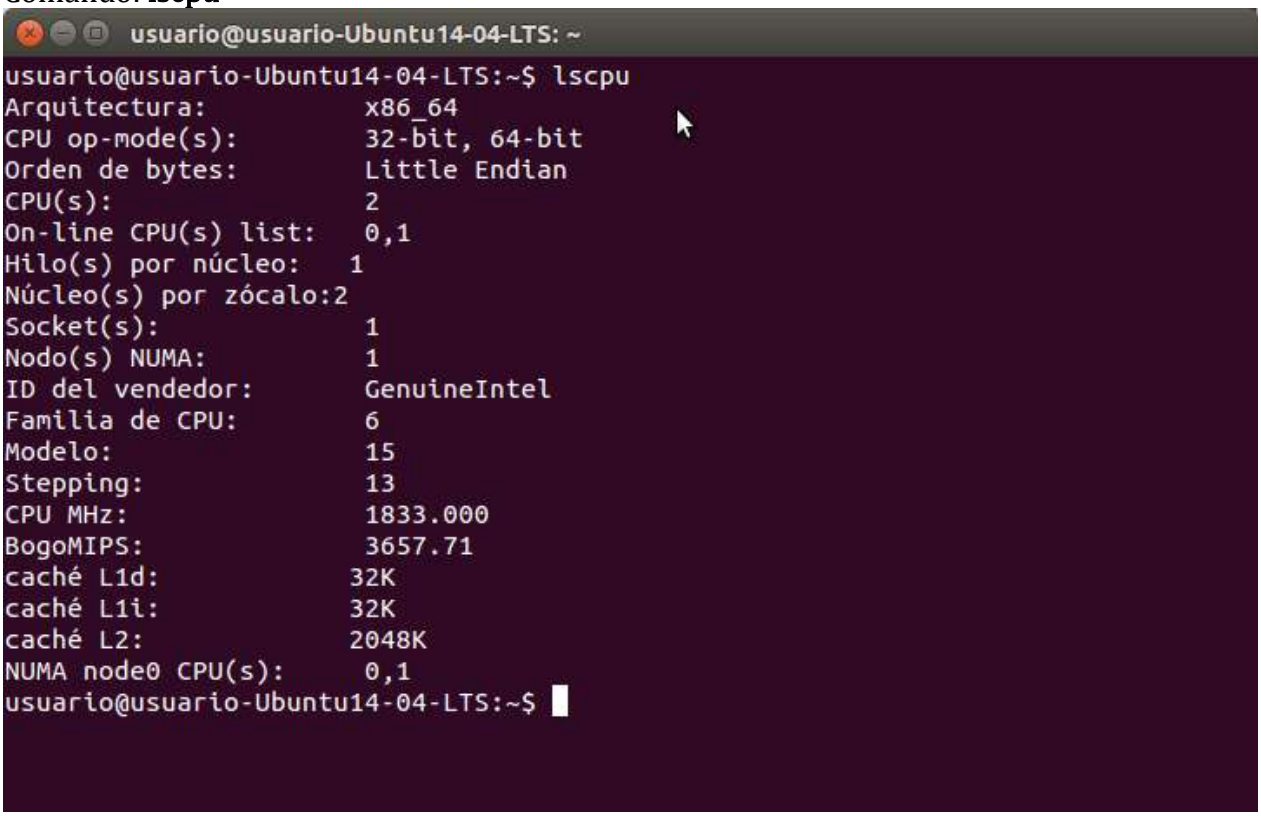


Figura #63 complementos de arch

Paso #3 Memorias

Cosa importante en nuestras computadoras es el conocimiento de su memoria ram, para ello existe un comando que nos permite visualizarla el cual es "**free**", sin embargo, este solo mostrará los valores en cantidades muy grandes, para no hacernos una idea de ello o aproximar el valor podremos visualizarlo más exacto si a este seguido de un espacio se le agrega "**-g**"; en la siguiente imagen se puede apreciar ambos casos.

Comando: **free** ó **free –g**.

```
root@edc-OptiPlex-9010-AIO:/home/edc# free
              total        used        free      shared  buff/cache   available
Memoria:     3912144      672340     1951816      305916     1287988     2652924
Swap:        4067324           0     4067324
root@edc-OptiPlex-9010-AIO:/home/edc# free -g
              total        used        free      shared  buff/cache   available
Memoria:          3           0           1           0           1           2
Swap:             3           0           3
```

Figura #64, Ram y swap.

De esta manera podemos observar su cambio significativo.

Paso #4 Puertos.

Con el siguiente comando podremos visualizar las entradas usb de nuestro computador, aun así, todas las opciones que aparecen no son en realidad las que se pueden usar, en el siguiente ejemplo se puede observar que los puertos que se pueden usar son aquello que parecen como (Linux foundation # root hub), para tener esta visualización vas ejecutar "**lsusb**".

Comando: **lsusb**

```
edc06@edc06-ThinkCentre-M73z:~$ lsub
No se ha encontrado la orden «lsub», quizás quiso decir:
 La orden «qsub» del paquete «gridengine-client» (universe)
 La orden «qsub» del paquete «torque-client-x11» (universe)
 La orden «qsub» del paquete «torque-client» (universe)
 La orden «qsub» del paquete «slurm-wlm-torque» (universe)
 La orden «lsusb» del paquete «usbutils» (main)
lsub: no se encontró la orden
edc06@edc06-ThinkCentre-M73z:~$ lsusb
Bus 002 Device 004: ID 17ef:6019 Lenovo
Bus 002 Device 003: ID 0461:4e04 Primax Electronics, Ltd
Bus 002 Device 002: ID 8087:8000 Intel Corp.
Bus 002 Device 001: ID 1d6b:0002 Linux Foundation 2.0 root hub
Bus 001 Device 004: ID 04f2:b43c Chicony Electronics Co., Ltd
Bus 001 Device 003: ID 8087:07dc Intel Corp.
Bus 001 Device 002: ID 8087:8008 Intel Corp.
Bus 001 Device 001: ID 1d6b:0002 Linux Foundation 2.0 root hub
Bus 004 Device 001: ID 1d6b:0003 Linux Foundation 3.0 root hub
Bus 003 Device 001: ID 1d6b:0002 Linux Foundation 2.0 root hub
edc06@edc06-ThinkCentre-M73z:~$
```

Figura #65, Los puertos conectores.

En La figura, se puede ver cuántos puertos hay en uso y disponibles. Teniendo en cuenta que un sistema típico de Linux tiene miles de archivos, algunos se preguntarán ¿cómo se gestiona la ubicación y el almacenamiento estos ?, pues bien, en este capítulo se ilustrara acerca de Debian y RPM que son dos sistemas gestores de paquetes muy buenos y más utilizados, ahondaremos en lo que pueden hacer y por último hablaremos de la importancia del Kernel y como por medio de comandos podemos observar o gestionar manualmente los procesos internos este.

CAPÍTULO 7: SCRIPT

Palabras claves (script, shell, bash, terminal)

Objetivo: crear desarrollos básicos que son importantes a la hora de realizar algunas acciones del sistema.

Introducción

El Script es un formato tipo archivo o texto, que nos permite escribir un código. Que al ser ejecutado se facilitan algunas tareas específicas, es decir, nos ayuda a realizar algunos trabajos de una forma más simple y eficiente. Es una herramienta muy útil ya que permite ser reutilizado (no tienes que volver a crearlo) las veces que lo desees, bien sea en diferentes tareas si este te sirve.

El sistema operativo Linux nos permite y facilita la implementación por comandos, lo cual es útil ya que podemos hacer labores con una mayor rapidez. Pero qué sucede cuando este comando se debe ejecutar a diario, o a una hora específica bien sea hoy o todos los días. Este trabajo se puede volver tedioso y en ocasiones por estar ocupados solemos olvidarlos o no hacerlo a la hora asignada, es en este momento donde entran los Scripts, ya que en estos formatos se pueden escribir comandos los cuales pueden ser utilizados más de una sola vez y también se les puede programar para que se ejecuten a una hora concreta todos los días.

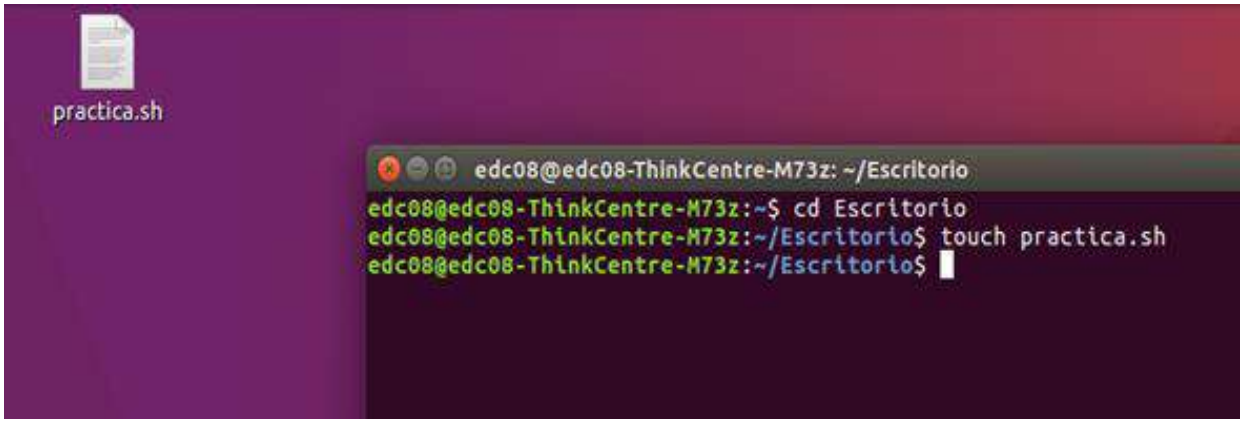
En esta pequeña práctica, vamos a llevar a cabo una introducción a los Scripts, por ejemplo, cómo se crean, que deben de llevar, como se ejecuta, etc.

Figura #66: Escritorio y terminal.

PRACTICA DE LABORATORIO DEL CAPÍTULO 7

Paso 1

Para empezar, vamos a crear un nuevo archivo, Existen muchas formas de crear un archivo bien sea presionando el click derecho y dando a la opción de nuevo archivo, además de esto por consola hay diversas formas. En esta ocasión trabajaremos por medio de la consola y lo haremos paso a paso, es muy importante que a la hora de crear el nuevo archivo que lleve la extensión "**.sh**" que es la forma abreviada de **Shell** que en programación se usa para referirse a un intérprete de comando.



Figura# 67: Creación del documento.

Como se observa en la imagen el comando "**touch**" se usa para crear un nuevo archivo, el comando "**cd**" es para en ingresar a un directorio o carpeta.

Paso 2

Como en todos los sistemas operativos el archivo se puede editar solamente abriendo, pero como se avisó anteriormente todo se va a desarrollar por consola, en el sistema de Linux existen múltiples editores como lo es el vi, nano, gedit.

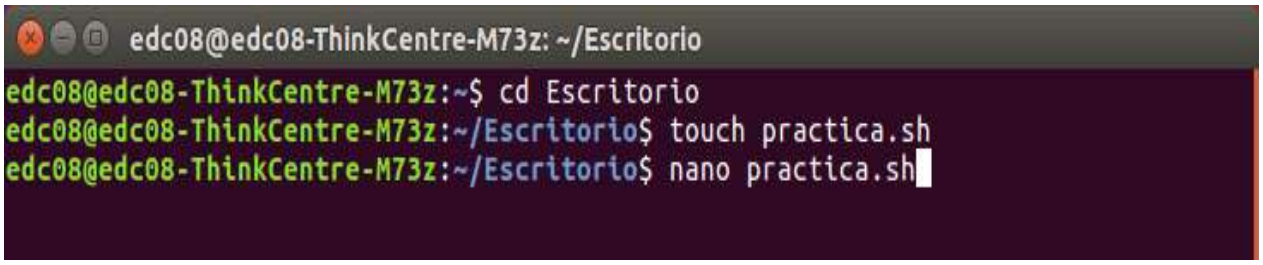


Figura #68: Comando nano.

Para desarrollar esta práctica se usará el editor "**nano**".

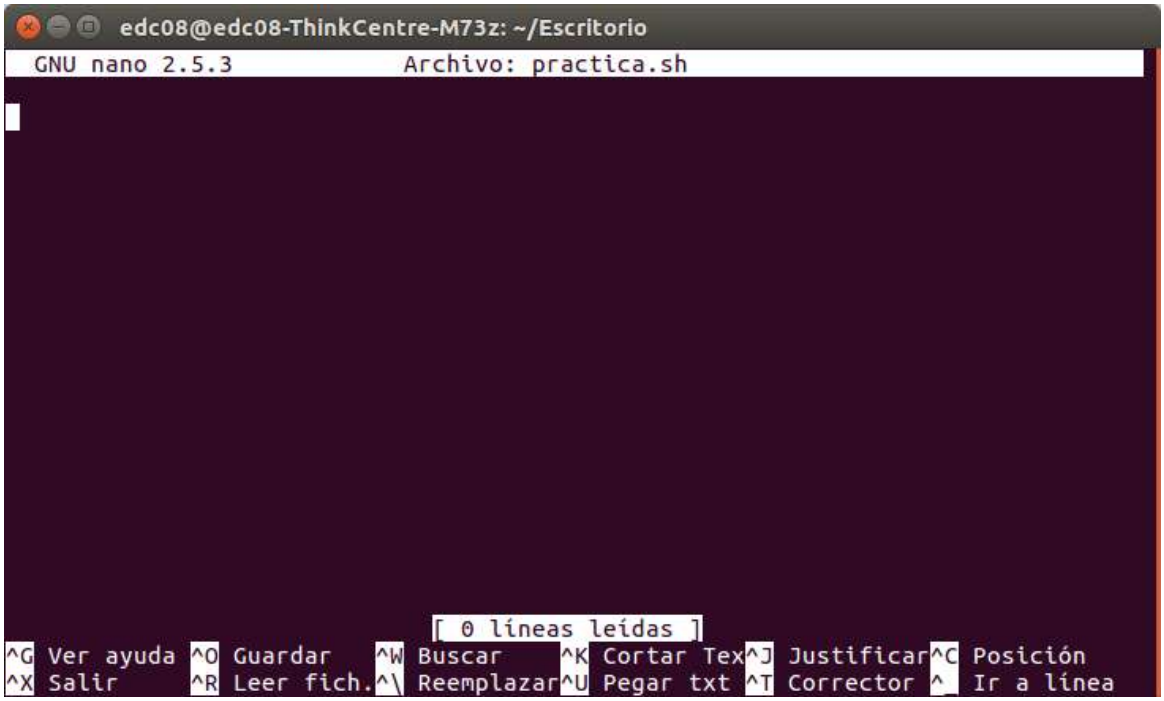


Figura # 69 Terminal nano.

Paso 3. armando el Script: Esta es la terminal del comando nano. Esta terminal es muy intuitiva y sencilla de usar a la hora de leer, editar o manipular el comando.

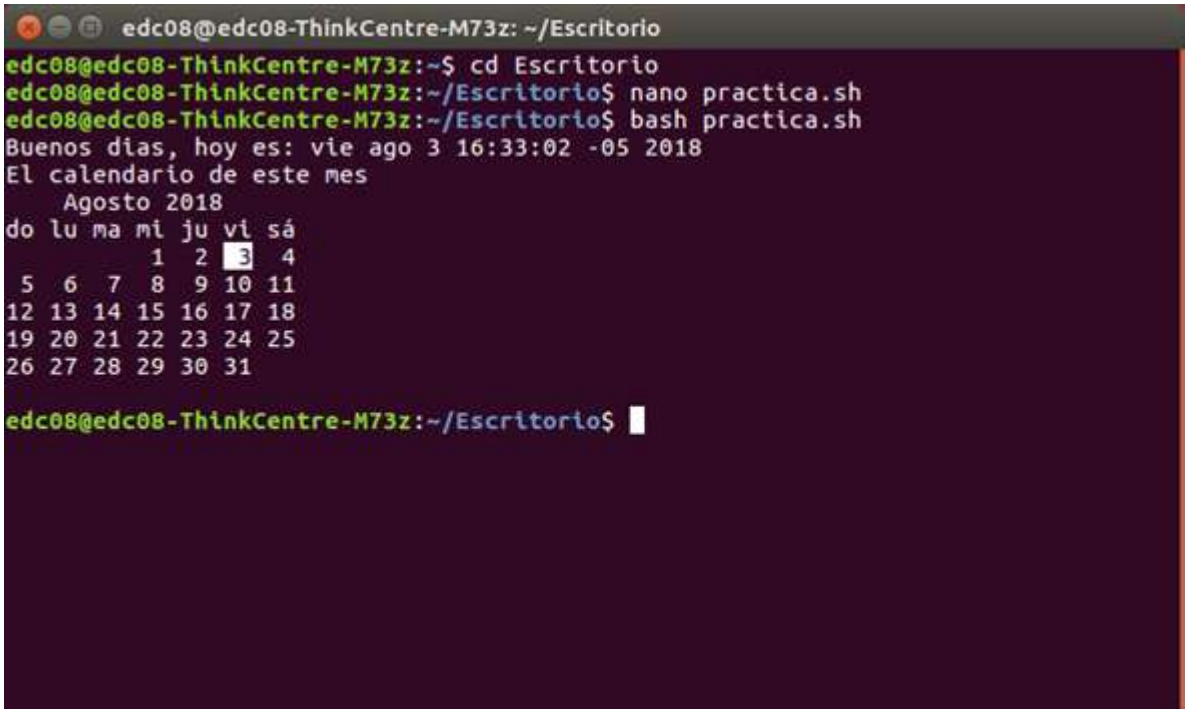
Paso 4. Contenido del script: La línea **"#!"** es muy importante ya indica que en definitiva pertenece a un ejecutable, el **"/bin/bash"** es la dirección donde se ejecutará.

Figura# 70 Línea #!/bin/bash.

A continuación, se va a realizar un script sencillo el cual permitirá hacer una visualización de cómo se hace un script.

Figura #71 Comando "practica.sh" finalizado.

Paso 5. Ejecución del script: Ahora vamos a ejecutarlo. Para hacer esto vamos a usar el comando **"bash"** desde la terminal como se muestra a continuación.



Figura# 72: Ejecución del script "practica.sh".

Un Script puede tener comando para accionar o invocar acciones del computador, como lo es por ejemplo actualizar todas las aplicaciones, y hasta el mismo sistema operativo. Basta aclarar que, para hacer estos tipos de script, debemos poner la contraseña cuando nos la pidan

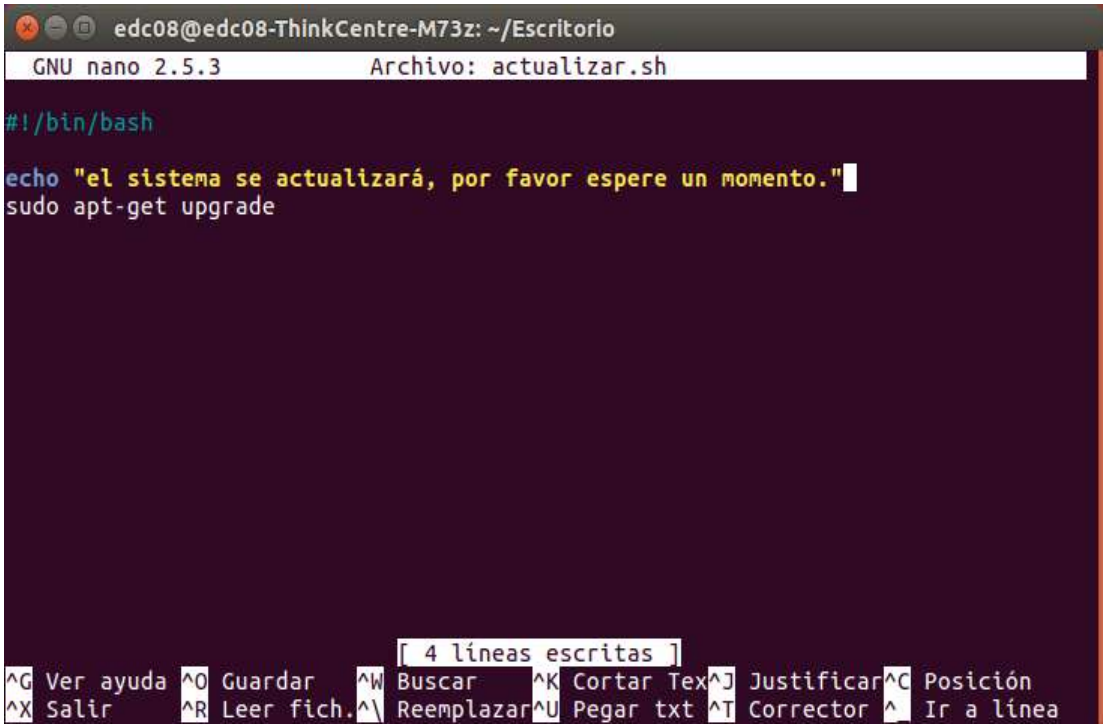
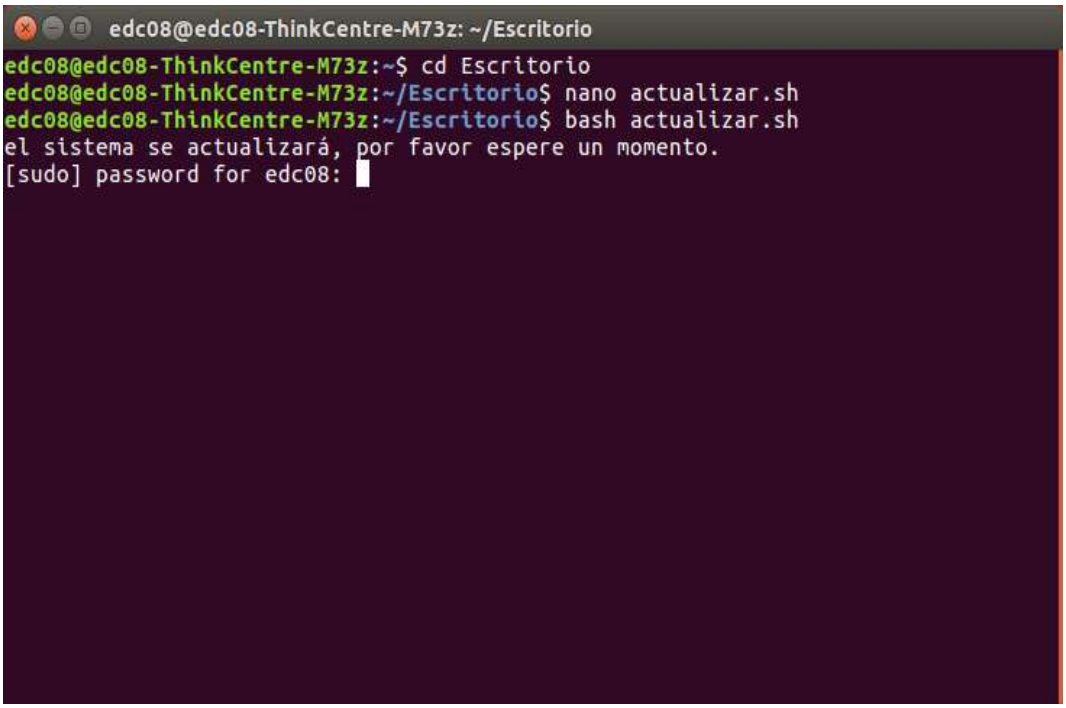


Figura #73: Script actualizar sistema.



Figura#74: Ejecución del script "actualizar.sh"

Podemos observar que al ejecutar el script **"actualizar.sh",** el sistema nos hará una petición solicitando la contraseña de usuario. Al introducir la contraseña, en pantalla nos deberá aparecer algo similar a esto, lo cual indica los programas que se van a actualizar. Se agrega la **"S"** para que el computador empiece con el proceso.

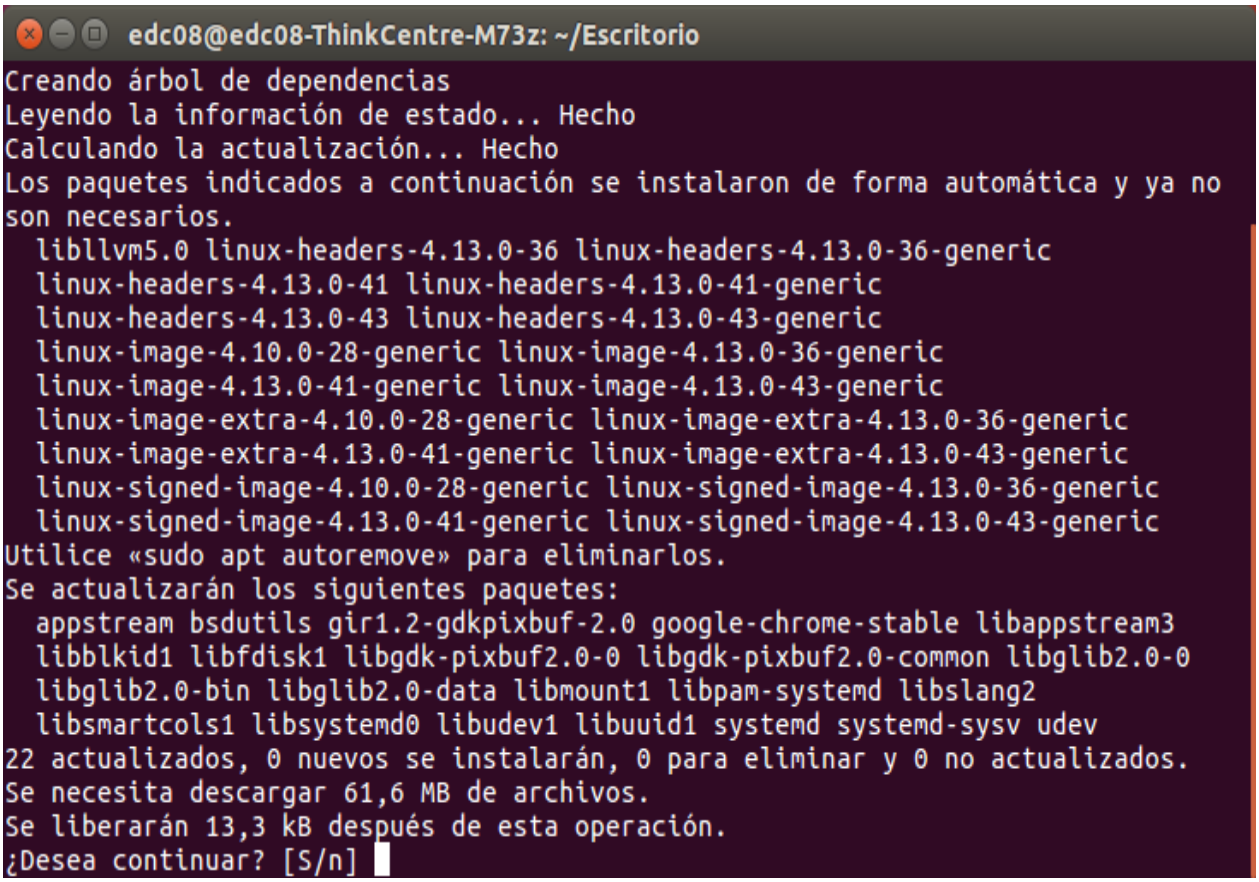


Figura #75 Script ejecutándose.

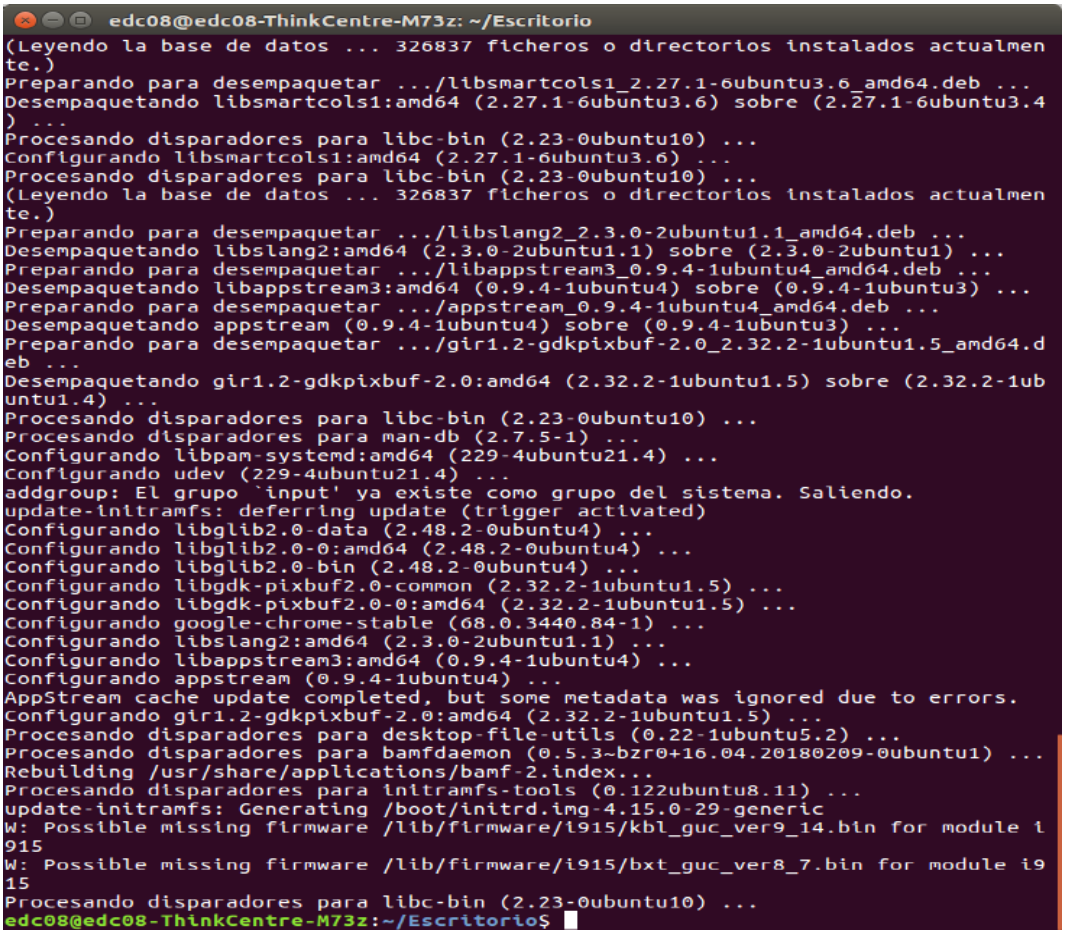


Figura #76: Finalización del comando.

Permisos de un Script

Un Script como cualquier otro archivo maneja permisos, lo cuales indican quienes pueden leer, editar o modificar y ejecutar.

Paso 1 : Primero vamos a observar los permisos que tiene el script de "**Practica.sh**", para observar estos permisos podemos hacer con el comando "**ls -lh**" (terminal).

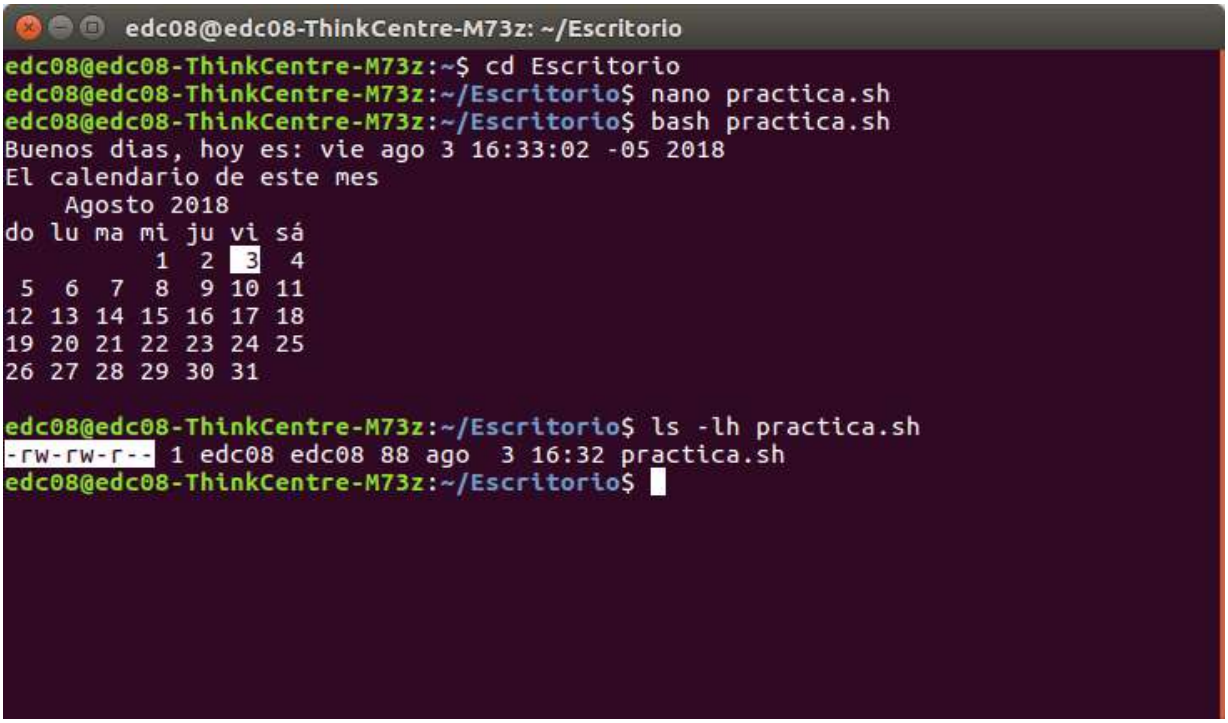


Figura #78 Permisos del archivo "practica.sh".

La línea resaltada, nos indica los permisos que tiene ese archivo, en este caso el archivo de "**practica.sh**", en seguida se enseñará una tabla con los permisos más comunes.

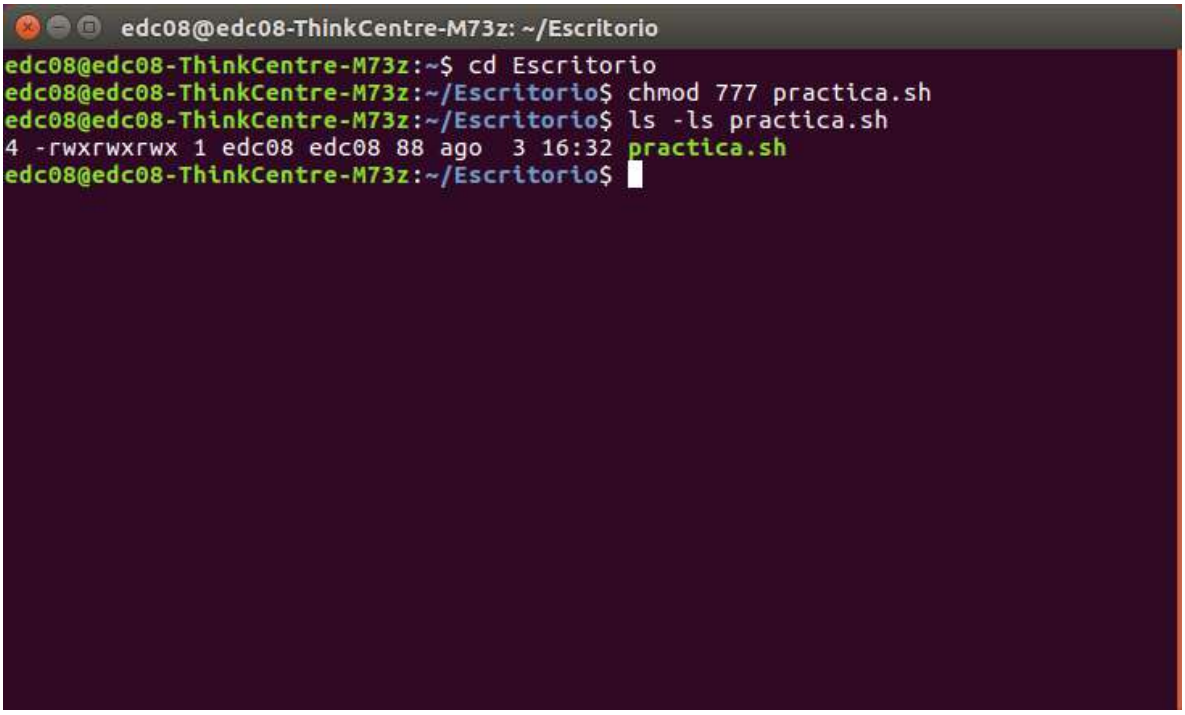
Tabla de permisos (octal)

Valores	Notación	Significado
777	(rwxrwxrwx)	No hay restricciones en los permisos. Cualquier persona puede hacer cualquier cosa.
755	(rwxr-xr-x)	El propietario del archivo puede leer, escribir y ejecutar el archivo. Todos los otros pueden leer y ejecutarlo
700	(rwx------)	El propietario puede leer, escribir y ejecutar. Nadie más tiene ningún derecho
666	(rw-rw-rw-)	Todos los usuarios pueden leer y escribir en el archivo

644	(rw-r--r--)	El propietario puede leer y escribir en el archivo, todos los demás solo pueden leer
600	(rw------)	El propietario puede leer y escribir en un archivo. Los demás no tienen ningún derecho

Paso 2

Ya conociendo para que sirven algunos permisos, podemos modificarlos en los archivos. Y eso se logra con el comando "**chmod**" como se muestra a continuación

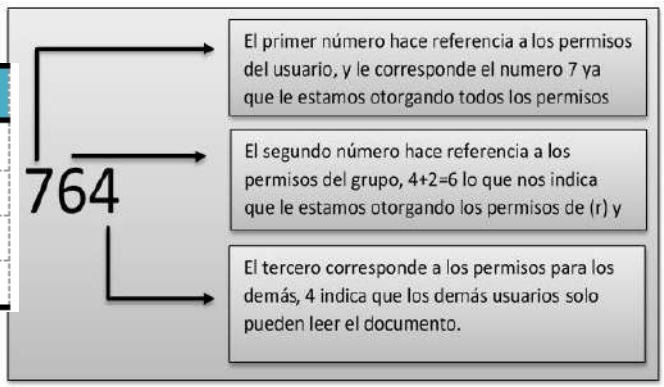


Figura# 79 Permisos modificados

Como se aprecia los permisos del archivo práctica han cambiado.

Por medio de la siguiente tabla se podrá entender mejor el funcionamiento el sistema de permisos octal (un sistema octal consiste en números del 0 al 7).

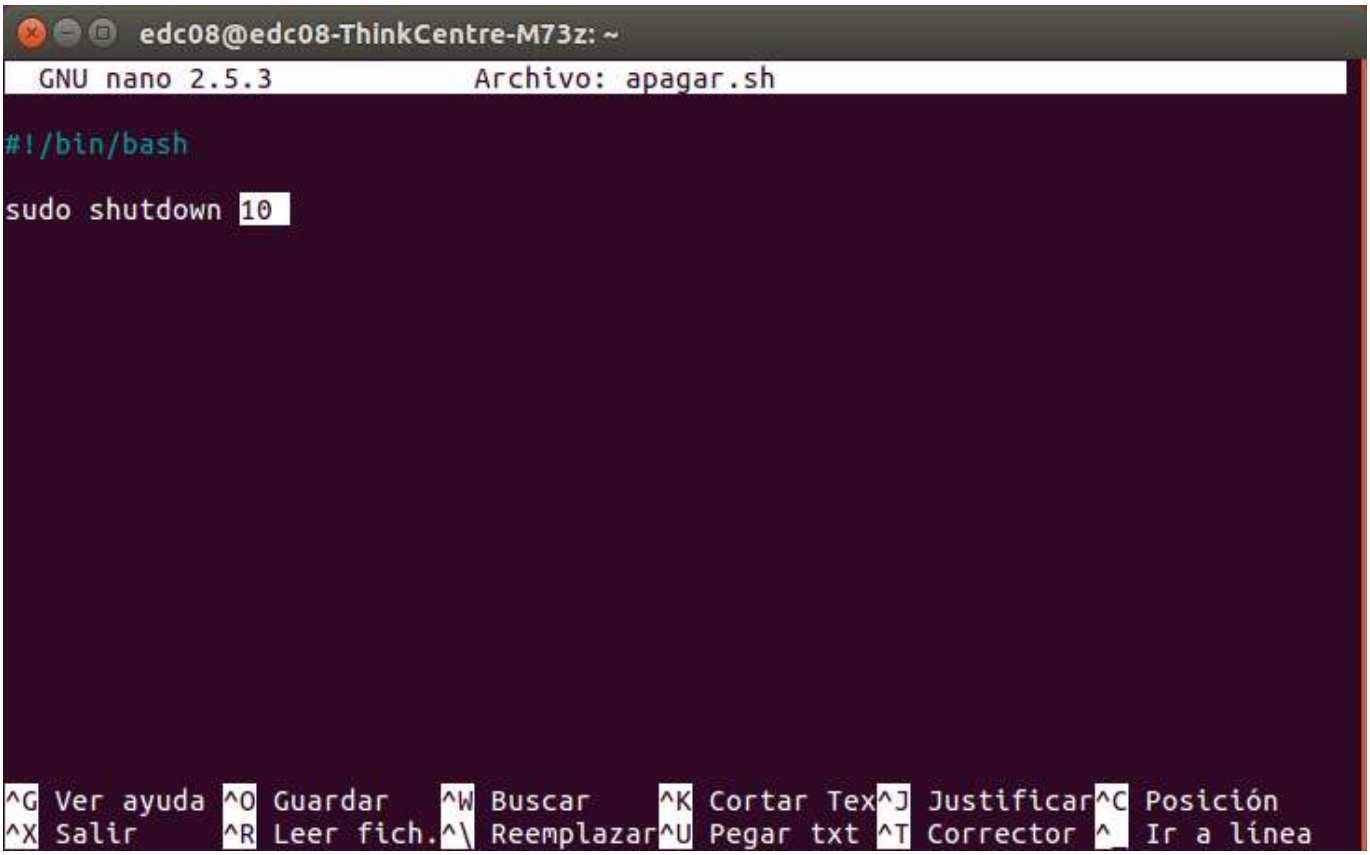
PERMISOS	NOTACION OCTAL
Lectura(r)	4
Escritura(w)	2
Ejecución(x)	1
Ninguno	0



si desea que este se apague a una hora especifica se int

ejemplo 10:00 (el computador se apaga a las 10:00AM). **Nota:** El formato debe ser 0 – 23 o formato militar, si desea que se apague a las 10:00PM, se introduce 22:00. Y si se desea apagar de inmediato se utiliza la palabra "now" que si se traduce significa ahora. Y si se desea reiniciar usaremos los siguientes.

- sudo shutdown 10 (10 minutos)
- sudo shutdown 10:00 (10:00AM)
- sudo shutdown now (inmediato)



Figura# 80 nano script

- **Metodología:** sesiones presenciales
- **Instrumento:** Cartilla formativa
- **Resultados e interpretación:** producción intelectual
- **Conclusiones:** Linux es un sistema operativo con potentes comandos que permiten optimizar los recursos del sistema, asimismo permite trabajar en ambientes de servidores en grandes compañías.
- **Bibliografía:** Recursos open source linux.org

Universidad Católica de Oriente
Programa de Ingenieria en sistemas
NIT. 890984746-7
Dirección: Sector 3, cra. 46 No. 40B 50
PBX: +(57)(4) 569 90 90
Fax: +(57)(4) 531 39 72