**Objetivos del trabajo**

**TRABAJO PRÁCTICO**

Linux

**Materia**: Arquitectura y sistemas operativos (ARSO)

**Profesor:** Javier Vastag

**Fecha de emtrega:** 29/04/2025

Matías Nicolas Bromberg, Miguel Guimar Salazar y Florencia Casagrande

**Índice**

1. **Objetivos del trabajo……………………………………………… 2**
2. **Introducción o plan de trabajo…………………………………… 3**
3. **Desarrollo…………………………………………………………… 4**
4. **Conclusiones………………………………………………………… 18**
5. **Anexo………………………………………………………………… 19**

**Objetivos:**

El objetivo del trabajo es familiarizarse con el uso de GNU/Linux, conociendo los principales comandos y aprender sobre el manejo de cuentas de usuarios y permisos.   
También el objetivo es comprender la normativa para la presentación de informes o trabajos prácticos.

**1. Introducción:**

En el presente trabajo, nos proponemos analizar y utilizar Linux, un sistema operativo diferente a los que estamos acostumbrados para así entender su funcionamiento.

Buscamos no solo familiarizarnos con nuevas herramientas digitales, sino también evaluar la capacidad de este sistema para integrar herramientas innovadoras en nuestra vida cotidiana y profesional.

**2. Desarrollo del trabajo:**

introducción

1.1 La cuenta de super usuario o también conocida como root es la cuenta con el nivel más alto de permisos en el sistema.

Este usuario tiene la capacidad de realizar cualquier tarea y acceder a cualquier archivo y configuración. :

1.6 Al ejecutar los comandos e informándome, me pude fijar que:

- Usé el comando ‘adduser’ con el apellido arias y se creó un nuevo usuario en el sistema con el nombre o apellido seleccionado y me solicita una contraseña

- Al ejecutar el comando ‘passwd’ me pide establecer o cambiar la contraseña para el usuario creado. Esta contraseña me la solicita cuando intente ingresar sesión.

- Al ejecutar el comando ‘logout’ me sacó del superusuario, cerrando la sesión actual.

Los archivos que fueron modificados fueron:

-/etc/passwd/

-/etc/shadow/

-/etc/group/

1.7 Los comandos adduser y passwd se encuentran en la dirección /user/sbin/ o /bin/ los encontré usando el comando “which adduser/passwd”.

Teclado / Terminal:

2.1 Al teclear cat /e tab se autocompletó a cat /etc/, y al apretar p tab no ocurre nada, pero al volver a apretar tab se muestra una lista de los archivos dentro de /etc/ que inician con P.

2.2 Ingresé cat /e, al tocar tab se autocompletó a cat /etc/ y luego al ingresar pas y tab nuevamente se autocompletó a cat /etc/passwd.

2.3.1 who Ingresé el comando who en la terminal y salió una lista mostrándome nombre de usuario, terminal, fecha y origen de conexión de los usuarios conectados.

2.3.2 Hice los pasos apretando el CTRL y ALT, luego apreté F3 y me mandó a otra terminal apareciendo el login del sistema.

2.3.3 Al ingresar who desde la terminal que abrí con F3, me mostró un nuevo usuario en la lista que es el que había creado anteriormente con el nombre de terminal de “tty3”.

2.3.4 Ingresé el comando who i am y me mostró una lista en donde solo mostraba el usuario que estaba usando en la terminal tty3.

La diferencia que noté es que who muestra una lista de los usuarios conectados en todas las terminales y who i am muestra al usuario conectado en la terminal que se está usando.

2.3.5 Nuevamente aprete CTRL y ALT, pero esta vez con F4, F5 y F6. En cada una de las terminales usé el comando who y who i am, y pude notar que cada vez la lista se hacía más grande mostrando todas las sesiones iniciadas en dichas terminales, y confirmé que cada vez que se pone who i am se muestra el usuario que está conectado en la terminal actual.

2.3.6 Los distintos tipos de terminales son:

Las terminales virtuales que se pueden abrir usando las teclas anteriormente usadas, CTRL + ALT F3 a F6. Sirven para tener múltiples sesiones de terminal en el mismo equipo

Terminales de consola que muestran mensajes del sistema. Sirve para iniciar sesión o acceder al sistema en modo texto.

Terminales Seriales están son las que se conectan por puertos físicos. Estos puertos sirven para la comunicación del hardware externo. ‘/dev/ttyS0/’

Terminales Pseudo-TTY estas son utilizadas sesiones de terminales virtuales o cualquier app que simule una terminal. ‘/dev/pts/0’

Ayuda:

3.1.1.1 El nodo raíz se llama (dir)top.

3.1.1.2 *Al utilizar el comando INFO, seleccionar “ cp:” y usar <ENTER> en la terminal aparece información sobre el comando “cp” (copy) y cómo se utiliza este mismo.*\*

El nodo se llama “(coreutils)dd invocation”.

El nodo siguiente es “(coreutils)dd invocation”.

Es posible moverse al próximo nodo con la tecla “n”.

Para volver al nodo anterior se debe usar la tecla “p”.

Al utilizar “u” la terminal te envía al padre del nodo en el cual te encontrabas. Si usas “u” en el nodo “(coreutils)cp” esta te envía al nodo “(coreutils)Basic operations”. Si volvemos a usar “u” nos lleva a “(coreutils)Top” y si lo volvemos a utilizar nos lleva a “(dir)Top” (el nodo raíz de info).

Para volver al nodo raíz de info, podemos usar la tecla “d”.

Para acceder al nodo cp directamente, podemos escribir “info cp” en la terminal.

Para buscar una palabra clave dentro de un nodo, podemos usar la tecla “s” y posteriormente insertar la palabra que queremos encontrar.

Para buscar la siguiente palabra clave debemos tener el buscador abierto (el cual se activa con “s”) y presionar ENTER. Esto nos llevará a otra parte del texto que tenga la misma palabra clave.

Para salir de INFO tenemos que usar la tecla 'q', al presionar esta, nos llevará a la terminal.

La diferencia entre 'man' e 'info', es que 'man' es un manual que contiene páginas dedicadas a un programa, utilidad, o función en específico, mostrando los datos principales e importantes para entender el funcionamiento de estos. 'Info' se maneja a través de nodos, donde el principal es (dir)Top.

Para ver la información de un comando en específico tenemos que colocar en la terminal "man" más el comando del cual requerimos la información. Por ejemplo "man cp" nos muestra cómo funciona el comando de copy files and directories.

Para buscar una palabra clave tenemos que utilizar "/" y escribir la palabra que estamos buscando. Esto va a remarcar todas las veces que aparezca esta misma.

Para salir del manual simplemente tenemos que utilizar la tecla "q" y esto nos devolverá a la terminal.

Para buscar una palabra clave en todas las páginas del manual, podemos utilizar el comando ‘man -k <palabraclave>’ en la terminal. Este comando busca en las descripciones cortas y en los nombres de página de manual la palabra clave. También podemos invocar el comando ‘man -K <palabraclave>’. Este comando utiliza una búsqueda de fuerza bruta en todas las páginas del manual para encontrar la palabra en cuestión.

Al utilizar el comando “man” nos indica un ejemplo muy básico de cómo se usa este comando y nos pide que pongamos otro argumento.

Al poner “man man” nos abre el manual de cómo se utiliza este mismo.

Al poner “man cp” nos muestra el manual con toda la información de este mismo comando.

Al utilizar “man printf” nos envía a la sección 1 del manual de printf.

Al utilizar “man fprintf”, “man sprintf” y “man 3 printf” nos envía a la sección número 3 del manual printf.

Al utilizar man junto a esos argumentos nos envía a la misma sección de printf (la sección 3). Lo que ocurre con “man printf” es que este “nombre” tiene más de una referencia. Esta nos envía a la sección 1 ya que ahí se encuentra la primera referencia con ese nombre. Pero podemos utilizar “man 3 printf” que nos mandará a la misma que los demás argumentos dados.

Para ver la función printf del lenguaje C, podemos usar el comando ‘man 3 printf’ para que nos muestre el manual de printf en lenguaje C.

La diferencia que podemos encontrar entre help e info, es que info se maneja a través de nodos en los que podemos seleccionar cual queremos revisar, donde el principal es (dir)Top. Por otro lado, help simplemente imprime en la terminal diferentes tópicos al cual podemos acceder utilizando “help <tópico>”.

Entre man y help se encuentran diferencias. Como que man es mucho más amplio y posee más explicaciones que help, el cual se maneja por diferentes tópicos mientras que man se maneja entre diferentes secciones de los distintos comandos que se pueden utilizar de una manera detallada separando la información entre nombre, sinopsis, descripción y más. Por otro lado está el help, el cual es mucho más breve y directo. Conteniendo muchos menos comandos que man.

Al invocar help, se nos mostrará una lista de tópicos de ayuda, los cuales podemos consultar. Estos tópicos son órdenes internas que le daremos al sistema.

Para ver la información de un comando hay que introducir en la terminal “help” seguido del comando del cual necesitamos información. En caso de que este esté en los tópicos, nos mostrará información sobre ese comando. Si el comando no aparece en la lista de tópicos en la terminal aparecerá un mensaje el cual nos recomienda que probemos buscar ese comando con man o con info. Por ejemplo, al escribir ‘help cd’, se nos mostrará información sobre este comando, que es lo que hace, y cómo utilizarlo junto a sus diferentes opciones.

Al utilizar ‘help cd’ podremos ver información sobre la función ‘cp’, cuál es su utilidad, y las diferentes opciones que nos provee el comando. En este caso, el comando help nos explica que ‘cp’ modifica el directorio actual a DIR.

Al utilizar “whereis cp” en la terminal, se nos mostrará la ubicación en donde se encuentran los archivos que concuerden con el nombre “cp”.

En este caso, nos aparece: cp: /usr/bin/cp /usr/share/man/man1/cp.1.gz

\**Al invocar “whereis ” en un directorio, se nos muestran todos los contenidos alojados dentro de ese directorio.*

La diferencia entre “whereis” y “find” es que find nos proporciona un mayor nivel de búsqueda permitiendo poner condiciones (como el peso del archivo, si se modificó en la última semana, si son archivos o directorios).

Al escribir ‘whatis cp’ en la terminal, se nos explicará brevemente para qué se utiliza el comando cp. En este caso nos explica que ‘cp’ copia archivos y directorios.

3.5.2 Ejecutando el comando ‘whatis \* bin’, se nos explicará la función de cada utilidad alojada dentro del directorio.

3.5.3 Al utilizar ‘whatis passwd’ en la terminal, se realizará una búsqueda exacta de la palabra “passwd” omitiendo cualquier otra utilidad que no tenga ese nombre. Posteriormente se mostrarán en pantalla las utilidades encontradas y se explicará la función de cada una.

Directorios:

4.1 Desde mi experiencia con el terminal de Windows, el comando cd se usa para navegar en distintos archivos desde la terminal.

4.1.1 Al utilizar el comando ‘cd /’ se cambia el ~ por el / haciéndome entender que cambio al directorio raíz del sistema archivos.

4.1.2 Al utilizar el comando ‘cd’ volvió a ponerse el ~ volviendo a estar en el directorio home.

4.1.3 Al utilizar el comando ‘cd /etc’ ahora aparece ‘/etc’ en vez del ~ dándome a entender que estoy en la ruta ‘/etc’

4.1.4 Me aparece que ‘cd..’ no es válido tanto en home como en la ruta /etc

4.1.5 Acá me varió, al utilizar ‘cd ..’ estando en la ruta de /etc se cambió el ‘/etc’ por un “/”, en cambio estando en home me aparece ‘/home’

4.2.1 La **ruta absoluta representa la ruta completa del recurso,** parte del directorio raíz hasta llegar al archivo concreto que se está buscando, en cambio la **ruta relativa representa solo una parte de la ruta**, ya que en ella se tiene en cuenta el directorio actual desde el que se está trabajando.

4.2.2.1 El comando ‘cd ~’ diría que me lleva al modo default (que es como se abre la terminal) que sería el home.

4.2.2.2 cel comando ‘cd –‘ me devolvió al anterior directorio y me muestra en donde estoy, por ejemplo, al poner ‘cd –‘ me muestra “/” y si lo uso nuevamente me muestra “/home/emaarso”

4.2.3 La diferencia entre ‘cd ~’ y ‘cd –‘ es que, ‘cd ~’ me lleva directamente a home, mientras que ‘cd –‘ me lleva al directorio anterior. Prácticamente va hacia atrás.

4.3.1 Con el comando ‘mkdir’ se puede crear un directorio

4.3.2 Con el comando ‘rmdir’ se puede borrar un directorio.

4.3.3 Si el directorio no está vació, me saltara un error “rmdir: fallo al borrar 'Probando': El directorio no está vacío”.

4.3.4 Para solucionar el problema, se puede usar el comando ‘rm -r <Nombre de la carpeta>’ para borrar la carpeta con contenido incluido.

4.4 La expresión ‘./’ hace referencia al directorio actual, sirve para cuando estás trabajando en el directorio actual acortando path.

4.5 Para poder cambiar de directorios hay varias formas, se puede usar el ‘cd /’ para ir al directorio raíz, ‘cd ~’ para ir al directorio principal, o ‘cd ..’ para moverse a un directorio padre.

Archivos:

5.1El comando cp copia archivos o directorios.

El comando mv sirve para mover archivos o directorios de un lugar a otro, o renombrarlos.

El comando rm sirve para eliminar archivos o directorios.

El comando rcp es un comando antiguo utilizado para copiar archivos entre sistemas a través de la red (a día de hoy ha sido reemplazado en gran medida por el comando scp).

El comando rsh permite ejecutar comandos en un sistema remoto a través de la red (se lo considera inseguro y ha sido reemplazado por el comando ssh).

El comando scp, al igual que el comando rsh, copia archivos entre sistemas de manera segura.

El comando ssh, al igual que el comando rsh, sirve para acceder de forma segura a una máquina remota y permite ejecutar comandos en el sistema remoto.

5.2Tras informarme, utilicé el comando cat para concatenar el contenido de dos archivos y este comando también puede usarse para ver el contenido de un archivo, combinarlo con varios archivos y redirigir la salida a un archivo.

El primer carácter “-l” indica el tipo de archivo o dispositivo y los posibles valores son:

-: Archivo regular.

d: Directorio.

l: Enlace simbólico.

c: Archivo de dispositivo de carácter.

b: Archivo de dispositivo de bloque.

p: FIFO (First In First Out), utilizado para la comunicación entre procesos.

s: Socket, utilizado para la comunicación entre procesos en red.

5.3/5.3.2Tras informarme y probarlo, pude ver que el comando touch funciona para cambiar la fecha, hora de acceso y modificación de un archivo. Si el archivo no existe, touch lo crea como un archivo vacío (un archivo vacío es un archivo que no contiene datos).

5.4 Las principales funciones que pude encontrar en touch son la de crear rápidamente archivos vacíos y actualizar la marca temporal de archivos existentes sin modificar su contenido.

Permisos:

6.1 Los siguientes 9 caracteres indican qué tipo de permiso tiene cada usuario. El usuario principal lo representa desde el 1 al 3 (sin contar el carácter que indica que tipo de archivo es), del 4 al 6 indica los permisos de grupo (que son permisos que solo cierto grupo tiene) y del 7 al 9 indica los permisos de los otros usuarios que no pertenecen a los miembros que tiene permisos de grupo.

6.2Están agrupados entre 3 grupos como ya se indicó en el punto anterior. El primero es el permiso de leer que representa el carácter ‘r’. Si se encuentra una ‘r’ en la primera posición significa que se tiene el permiso para leer el archivo, en cambio si sale ‘-’ significa que no tiene permisos para leer. En el segundo carácter se encuentra el permiso de escribir representado con el carácter ‘w’, si este aparece significa que se puede editar ese archivo, si no aparece y hay un ‘-’ en su lugar significa que no. Por último está el tercer carácter el cual representa el permiso a ejecutar con el carácter ‘x’, si este se encuentra significa que se tiene el permiso para ejecutar, en cambio si se encuentra ‘-’ en su lugar significa que no se tiene permisos para ejecutar. Por lo tanto estarían agrupados así: permiso leer (r), permiso escribir (w), permiso ejecutar (x). (rwx en orden)

6.3El comando para modificar los permisos necesarios de un archivo es el siguiente: “chmod <grupo><+/-><permiso> <archivo>” Un ejemplo puede ser el siguiente “chmod u-w <TestCHMOD.txt>”. Esto hace que el usuario no pueda escribir en el archivo TestCHMOD.txt. La opción de <grupos> pueden ser las siguientes: ‘u’: Que sería para modificar los permisos del usuario. ‘g’: Que sería para modificar los permisos del grupo. ‘o’: Que sería para modificar los permisos de los otros usuarios. (la o representa a others) ‘a’: Que sería para modificar los permisos de TODOS los grupos (osea usuarios, grupos y otros usuarios) Luego la opción de permisos puede ser las siguientes: ‘r’: Para modificar el permiso de lectura del archivo. ‘w’: Para modificar el permiso de escritura sobre el archivo. ‘x’: Para modificar el permiso de ejecución del archivo.

6.4El ‘owner’ de un archivo es el usuario que es propietario de dicho archivo. Y el ‘group’ el grupo al que pertenece. Es posible cambiar el propietario y el grupo mediante el comando ‘chown’. De la siguiente manera: ‘chown <usuarioDelNuevoPropietario>[:<GrupoNuevo>] <nombreDelArchivo>’

6.5Al intentar cambiar los permisos de un archivo perteneciente a root estando sesionados como usuarios, no se nos va a permitir la acción.

6.6Para modificar los permisos con el comando “chmod” usando octal es sencillo, en la siguiente tabla se puede observar el valor de cada número de la base octal, cada uno tiene diferentes opciones de permisos. Por ejemplo, el número 6 nos da permiso para leer y para escribir. Es decir que si pongo el siguiente comando le estaría dando el permiso de leer y escribir a todos los grupos (usuarios, grupos y otros usuarios): “chmod 666 archivocualquiera.txt” El primer carácter determina los permisos del usuario, el segundo el del grupo y el tercero indica el permiso de todos los otros usuarios. Todos estos valores se pueden ver interpretar mejor en una tabla.

6.7Lo que significan los caracteres son los siguientes: ‘r’- Nos permite ver el contenido de los directorios pero no nos deja crear ni modificar archivos o directorios. ‘w’- Nos permite editar o crear archivos y añadir más directorios. Hasta nos deja cambiar el nombre de los archivos o directorios. ‘x’- Nos permite acceder a los subdirectorios y a los archivos desde la terminal utilizando ‘cd’.

Filtros:

7.1 cat es un comando que muestra por pantalla el contenido de un archivo. No tiene paginación y lo muestra todo de una sola vez.

more, al igual que cat, sirve para mostrar el contenido de un archivo, pero a diferencia de este, permite desplazarse hacia abajo página por página.

less, al igual que more y cat, se usa para ver archivos, pero con más funciones como desplazamiento hacia arriba y abajo, búsqueda de texto, más flexibilidad y potencia

7.2. Head permite ver las primeras líneas de un archivo; por defecto, muestra las primeras 10, a diferencia de tail, que cumple con la función de mostrar las últimas líneas (en ambos casos, uno puede aumentar la cantidad de líneas utilizando -n, por ejemplo, head -n 20).

7.3El comando uniq se utiliza para eliminar líneas consecutivas de un archivo (se recomienda que esté ordenado para que funcione de forma correcta).

7.4 Grep busca texto dentro de archivos y muestra las líneas que coinciden con un patrón especificado.

VI:

8.1 Utilizar el comando ‘vi $HOME/prueba.txt’ se hicieron muchos espacios donde me permite escribir en la parte de arriba de la terminal, mientras debajo de todo aparece “"~/prueba.txt" [New File]”

8.1.2 Para poder escribir la frase tuve que apretar a o i, informándome me di cuenta que vi abre en modo comando, y para poder escribir hay que poner algún comando de inserción.

8.1.3 Para poder guardar el archivo y salir del editor, primero use ESC para salir del modo inserción y luego use el comando “:wq” para poder guardar y salir del editor.

8.1.6 Para poder completar lo del punto 8.1.5, estando dentro de archivo, aprete i para entrar en modo inserción y escribí lo pedido sin problemas ya que se escribía al inicio del archivo.

8.1.7 Para poder guardar el archivo y continuar con la edición, apreté ESC para salir del modo inserción y use el comando ‘:w’.

8.1.8 Para borrar la línea “Legajo” utilice estos dos métodos:

Teniendo el cursor en la línea de Legajo, apreté ‘dd’ y se borró.

Viendo el número de línea con el comando ‘:set number’ y borrándolo con el comando :5d.

8.1.9 Para invertir el orden de Curso y TP, usé ‘dd’ en TP y utilizando la ‘k’ subí hacia la línea encima de Curso (Sistemas Operativos) y apreté p para que se pegue la línea de TP que se había cortado con el comando ‘dd’ y queda justo encima de Curso invirtiendo los roles.

8.1.10 Para poder ubicarme en la línea 2 del archivo, utilicé el comando ‘:2’ que me redirigió instantáneamente a la línea 2.

8.1.11 Para poder copiar las líneas 2, 3 y 4 utilice el comando ‘:2,4y’

8.1.12 Para poder ubicarme al final del archivo use el comando ‘G’ en mayúscula, y pegue dos veces el contenido usando el comando ‘p’

8.1.13 Para poder deshacerme de uno de los copiados use el comando ‘u’ para “deshacer el comando anterior”

8.1.14 Para poder buscar la palabra Documento use el código ‘/Documento’ y me llevó directamente hacia Documento, para poder bajar a la siguiente ocurrencia use el comando ‘n’.

8.1.15 ¿Cómo puedo reemplazar la palabra “Documento” por “Documento:” (sin borrar, o realizar el reemplazo a mano)’

Para poder cambiar ‘Documento’ a ‘Documento:’ estando con el cursor en la línea de Documento use el siguiente comando: ‘:s/Documento/Documento:/’

8.1.16 Use el comando ‘:wq’ para poder guardar el archivo y salir.

8.1.17 Al usar el comando ‘gcc miprog.c -o miprimerprog’ me salió error, instale el paquete ya que no lo tenía instalado pero nuevamente me volvió a salir error.  
Al utilizar el comando ‘./miprimerprog’ me salió un aviso de que no existía el archivo o directorio

Al usar el comando “:%s/\/3/\/Marzo/g” hubo sustituciones en 6 líneas, en específico las líneas que tenían la fecha, línea de la 3 a la 8.

Corregí el ‘principal’ y con el comando ‘:s/principal/int main/’ cambie el “principal” por “int main”. Y le agregué la biblioteca <stdio.h>.

(Volví a poner ‘gcc miprog.c -o miprimerprog’ y no me salió el error, y al volver a poner ./miprimerprog en vez de salir que no existía directorio o archivo, me mostro el mensaje del programa.

Variables de entorno:

9.1,1El contenido de la variable HOME es: /home/<nombredeusuario>  
En la variable HOME se guarda el directorio home del usuario.

El contenido de la variable LOGNAME es: <nombredeusuario>  
Esta variable almacena el nombre del usuario utilizando el sistema.

El contenido de la variable HOSTNAME es: <NombreDelHost>  
Esta variable almacena el nombre del host del sistema.

La variable IFS no contiene nada de forma predeterminada al momento de visualizarla con ‘echo’.  
La variable IFS significa Internal Field Separator, y su función es la de indicar qué valor se utiliza como separador. Nosotros podemos cambiar ese valor con el carácter que deseemos, indicando que ese carácter a partir de ese momento va a ser un separador.

9.1.2Para ver una lista de las variables de entorno dentro del sistema, podemos utilizar el comando ‘printenv’. En esta lista se encuentra el contenido de cada una de las variables.  
También podemos ver el contenido de las variables de forma individual con el comando ‘echo’. Este último fue el comando que utilizamos para poder ver el contenido de cada variable en el punto anterior.

9.1.3Para poder crear nuestra propia variable, usamos el comando ‘export’ de esta forma: ‘export <NombreVariable>=<Valor>’.  
A continuación, creamos la variable de entorno HOLA, con el valor “Hola mundo". (Véase anexo n° 21)

9.1.4El uso que se le da a la variable de entorno PATH es el de almacenar en una lista todas las rutas donde se encuentran archivos ejecutables.  
Si el PATH no encuentra el comando simplemente no se ejecutará y nos informará que no se encontró el comando. Una manera de solucionar esto es cambiando la variable para que represente al directorio que queramos.

La variable de entorno PS1 se utiliza para mostrar el nombre de usuario y el nombre del host de forma predeterminada al principio de cada comando en la terminal.

9.1.5La variable PS2 tiene como función declarar entradas en los comandos multilínea, con el carácter “>”.  
Los comandos multilínea son aquellos comandos que se ejecutan dos en una sola línea. Por ejemplo, utilizando “cd /home/ubuntu/test ; mkdir testdir”. Ahí estamos cambiando de directorio y creando uno nuevo en una sola línea de comando. O sea que estamos ejecutando dos comandos en una sola línea.

**3. Conclusión:**

**Conforme a lo examinado en el presente trabajo, concluimos que la transición hacia diferentes sistemas operativos es una forma de salir de la zona de confort que trae nuevas herramientas a la vida cotidiana.**

**Entendemos que estos cambios traen dificultades de adaptación porque estamos acostumbrados a los sistemas dominantes como lo son Google o Mozilla Firefox. Sin embargo, creemos firmemente que, a través de la innovación y la imaginación, es posible crear nuevos sistemas operativos que sean diferentes a aquellos que ya conocemos y que nos den nuevos recursos.**

**ANEXO**

2.3.2