

**INSTITUTO POLITÉCNICO NACIONAL.**

ESCUELA SUPERIOR DE CÓMPUTO.

**SISTEMAS OPERATIVOS.**

**PRÁCTICA 2**

ENTORNO DE LINUX



**Integrantes del equipo:**

* Chavarría Vázquez Luis Enrique.
* Juárez Espinosa Ulises.
* Machorro Vences Ricardo Alberto.
* Pastrana Torres Victor Norberto.

Grupo 2CM6

Índice de contenido.

[**Glosario de términos.** 5](#_Toc53319869)

[**Máquina virtual** 5](#_Toc53319870)

[**Contenido** 6](#_Toc53319871)

[**Escritorios GNOME y KDE** 6](#_Toc53319872)

[**Entornos Command Line Interface y Graphical User Interface** 7](#_Toc53319873)

[**Terminal de Linux.** 9](#_Toc53319874)

[**Tipos de usuario en Linux** 10](#_Toc53319875)

[**Direccionamiento relativo y absoluto** 11](#_Toc53319876)

[**Redireccionamiento** 12](#_Toc53319877)

[**Clasificación de los comandos en Linux** 12](#_Toc53319878)

[Información del sistema 12](#_Toc53319879)

[Apagar (Reiniciar Sistema o Cerrar Sesión) 13](#_Toc53319880)

[Archivos y Directorios 13](#_Toc53319881)

[Encontrar archivos 15](#_Toc53319882)

[Montando un sistema de ficheros. 15](#_Toc53319883)

[Espacio de Disco 16](#_Toc53319884)

[Usuarios y Grupos 16](#_Toc53319885)

[Permisos en Ficheros (Usa ”+” para colocar permisos y ”-” para eliminar) 17](#_Toc53319886)

[Atributos especiales en ficheros (Usa ”+” para colocar permisos y ”-” para eliminar) 18](#_Toc53319887)

[Archivos y Ficheros comprimidos. 18](#_Toc53319888)

[Paquetes RPM (Red Hat, Fedora y similares). 19](#_Toc53319889)

[Actualizador de paquetes YUM (Red Hat, Fedora y similares) 20](#_Toc53319890)

[Paquetes Deb (Debian, Ubuntu y derivados). 21](#_Toc53319891)

[Actualizador de paquetes APT (Debian, Ubuntu y derivados). 21](#_Toc53319892)

[Ver el contenido de un fichero. 22](#_Toc53319893)

[Manipulación de texto. 22](#_Toc53319894)

[Establecer caracter y conversión de ficheros. 23](#_Toc53319895)

[Análisis del sistema de ficheros. 23](#_Toc53319896)

[Formatear un sistema de ficheros 24](#_Toc53319897)

[Trabajo con la SWAP. 24](#_Toc53319898)

[Salvas (Backup). 24](#_Toc53319899)

[CD-ROM. 25](#_Toc53319900)

[Trabajo con la RED ( LAN y Wi-Fi) 26](#_Toc53319901)

[Tablas IP (CORTAFUEGOS) 26](#_Toc53319902)

[Monitoreando y depurando. 27](#_Toc53319903)

[Comandos adicionales. 28](#_Toc53319904)

[**Variables de entorno** 29](#_Toc53319905)

[**Desarrollo (Códigos y ventanas de ejecución)** 30](#_Toc53319906)

[**Comando cal** 32](#_Toc53319907)

[**Comando clear antes de ejecutarse** 32](#_Toc53319908)

[**Comando clear después de ejecutarse** 33](#_Toc53319909)

[**Comando apt** 33](#_Toc53319910)

[**Comando rm** 33](#_Toc53319911)

[**Comando Date.** 33](#_Toc53319912)

[**Comando ifconfig.** 34](#_Toc53319913)

[**Comando exit antes de ejecutarse.** 34](#_Toc53319914)

[**Comando exit después de ejecutarse** 35](#_Toc53319915)

[**Comando mv.** 35](#_Toc53319916)

[**Comando echo** 36](#_Toc53319917)

[**Comando df** 36](#_Toc53319918)

[**Comando ps** 37](#_Toc53319919)

[**Comando more** 37](#_Toc53319920)

[**Comando time** 38](#_Toc53319921)

[**Comando du** 38](#_Toc53319922)

[**Comando ps –fea** 38](#_Toc53319923)

[**Comando less antes de ejecutarse** 38](#_Toc53319924)

[**Comando Less después de ejecutarse** 38](#_Toc53319925)

[**Función uname** 39](#_Toc53319926)

[**Comando pstree** 39](#_Toc53319927)

[**Comando man** 40](#_Toc53319928)

[**Comando mkdir** 40](#_Toc53319929)

[**Comando w** 40](#_Toc53319930)

[**Función cat** 40](#_Toc53319931)

[**Comando pico antes de copilarse** 40](#_Toc53319932)

[**Función pico después de copilarse** 41](#_Toc53319933)

[**Comando who** 41](#_Toc53319934)

[**Comando fg** 41](#_Toc53319935)

[**Comando nano antes de ejecutarse** 41](#_Toc53319936)

[**Comando nano después de ejecutarse** 41](#_Toc53319937)

[**Comando pwd** 41](#_Toc53319938)

[**Comando cd** 41](#_Toc53319939)

[**Comando vi antes de ejecutarse** 42](#_Toc53319940)

[**Comando vi luego de ejecutarse** 42](#_Toc53319941)

[**Comando wc** 42](#_Toc53319942)

[**Comando ls** 42](#_Toc53319943)

[**Comando apt-get** 42](#_Toc53319944)

[**Comando sudo** 43](#_Toc53319945)

[**Compilar Hola Mundo en código para nano** 43](#_Toc53319946)

[**Creación Hola Mundo** 43](#_Toc53319947)

[**Código Hola Mundo editado en nano** 43](#_Toc53319948)

[**Código Hola Mundo compilación nano** 43](#_Toc53319949)

[**Compilar Hola Mundo en código para vi** 43](#_Toc53319950)

[**Creación Hola Mundo** 43](#_Toc53319951)

[**Edición Hola Mundo en vi** 44](#_Toc53319952)

[**Ejecución hola mundo en vi** 44](#_Toc53319953)

[**Compilación y pantalla de programa2a** 44](#_Toc53319954)

[**Compilación y pantalla de programa3a** 45](#_Toc53319955)

[**Conclusiones.** 46](#_Toc53319956)

[**Chavarría Vázquez Luis Enrique.** 46](#_Toc53319957)

[**Juárez Espinosa Ulises.** 46](#_Toc53319958)

[**Machorro Vences Ricardo Alberto.** 46](#_Toc53319959)

[**Bibliografía** 47](#_Toc53319960)

Índice de figuras

Índice de tablas

# **Glosario de términos.**

## **Máquina virtual**

Es un software que simula un sistema de computación y puede ejecutar programas como si fuese una computadora real. Este software en un principio fue definido como "un duplicado eficiente y aislado de una máquina física".

# **Contenido**

## **Escritorios GNOME y KDE**

Todos nosotros hemos usado al menos una vez una computadora, pero pocas veces nos hemos preguntado porque esta es tan sencilla de usar. Esta facilidad se debe a los entornos de escritorio que en si son un conjunto de Software (Programas y/o Aplicaciones) que ayuda a que el usuario de una computadora interactúe cómodamente con esta de una forma amigable.

Entre los populares esta KDE, un proyecto internacional de software libre, orientado específicamente a desarrollar un entorno de escritorio e infraestructura de desarrollo para los sistemas operativos GNU/Linux, Mac OSX e incluso, Windows.

KDE es un entorno de escritorio moderno, y que según su propia página [1] ofrece un extraordinario soporte para hasta 75 idiomas diferentes, lo que ofrece a los usuarios un nivel de comprensión del entorno mucho más adaptado a su idioma.

Aun así, este software también tiene competencia siendo comparado normalmente con GNOME (GNU Network Object Model Environment /Entorno de Modelo de Objeto de Red GNU), otro entorno de escritorio e infraestructura de desarrollo nacido en 1997 como alternativa del propio KDE ,para sistemas operativos GNU/Linux, Unix y derivados Unix como BSD o Solaris; compuesto enteramente de software libre.

GNOME está disponible en las principales distribuciones GNU/Linux, incluyendo Fedora, Debian, Ubuntu, Manjaro Linux, Red Hat Enterprise Linux, SUSE Linux Enterprise, CentOS, Oracle Linux, Arch Linux, Gentoo​, SteamOS, entre otras. También, se encuentra disponible en Solaris, un importante sistema operativo UNIX y en Sistemas operativos Unix-like como FreeBSD.

Esta rivalidad entre ambos softwares ha variado con el tiempo al grado de que muchas veces se ha rumorado dentro de sus comunidades de una colaboración de ambos para crear un fusión de entornos de escritorio pero ninguno se ha concretado.

## **Entornos Command Line Interface y Graphical User Interface**

Una computadora no sirve de mucho si no se puede comunicar uno con ella, por eso es que desde el inicio de estas se ha buscado diferentes formas de poder lograr esto, siendo una de las más exitosos las interfaces de línea de comando o interfaces de línea de órdenes (en inglés, command-line interfaces, CLI). Estos son métodos que permite a los usuarios dar instrucciones a algún programa informático por medio de una línea de texto simple. No debe confundirse los conceptos de CLI, shell y emulador de terminal, porque no son lo mismo ya que un CLI es un método mientras que un shell y emulador de terminal son programas informáticos.

Las CLI pueden emplearse interactivamente, escribiendo instrucciones en alguna especie de entrada de texto, o pueden utilizarse de una forma mucho más automatizada (archivo batch), leyendo órdenes desde un archivo de scripts.

Esta interfaz existe casi desde los comienzos de la computación, superada en antigüedad solo por las tarjetas perforadas y mecanismos similares. Existen para diversos programas y sistemas operativos, para diverso hardware, y con distinta funcionalidad.

Por ejemplo, las CLI son parte fundamental de los shells o emuladores de terminal. Aparecen en todas las interfaces de escritorio (GNOME, KDE, Microsoft Windows) como un método para ejecutar aplicaciones rápidamente. También se utilizan en aplicaciones cliente-servidor, en gestores de bases de datos, en clientes FTP, etc. Las CLI son además un elemento fundamental de aplicaciones de ingeniería tan importantes como MATLAB y AutoCAD.

Otra forma que también ha tenido mucho éxito en la forma en la cual se comunica los usuarios con la maquina son a aquellas que apelan más a la parte gráfica. Esta orientación tiene sentido ya que al ser el humano una criatura muy visual, este puede entender mejor como proceder en diferentes situaciones si hay un apoyo grafico que le ayude, además de que una buena selección de diseños y colores puede hacer que el usuario pueda comunicarse mejor con la máquina.

Estos programas son conocidos como interfaces gráficas de usuario, conocidas también como GUI (del inglés graphical user interfaces), que cumplen con lo mencionado anteriormente, siendo un programa informático, que utiliza un conjunto de imágenes y objetos gráficos para representar la información y acciones disponibles en la interfaz. Su principal uso, consiste en proporcionar un entorno visual sencillo para permitir la comunicación con el sistema operativo de una máquina o computador.

Habitualmente las acciones se realizan mediante manipulación directa, para facilitar la interacción del usuario con la computadora. Surge como evolución de las interfaces de línea de comandos que se usaban para operar los primeros sistemas operativos y es pieza fundamental en un entorno gráfico. Como ejemplos de interfaz gráfica de usuario, cabe citar los entornos de escritorio Windows, el X-Window de GNU/Linux o el de Mac OS X, Aqua.

En el contexto del proceso de interacción persona-computadora, la interfaz gráfica de usuario es el artefacto tecnológico de un sistema interactivo que posibilita, a través del uso y la representación del lenguaje visual, una interacción amigable con un sistema informático.

## **Terminal de Linux.**

Como todos los sistemas operativos basados en Unix , Linux disponen de un intérprete de comandos u órdenes (conocido como terminal, consola,shell o bash ) que hace de interfaz entre el usuario y el propio sistema operativo. La terminal o consola es una herramienta poderosa capaz de realizar tareas de forma más rápida que mediante la interfaz gráfica del sistema operativo.

Si bien es cierto, que para un usuario normal no tiene mucha relevancia el aprenderlo, para un usuario que quiere convertirse en un desarrollador o administrador de sistemas operativos cobra gran importancia debido a que va a tener que utilizar esta herramienta constantemente, ya sea para navegar por el sistema/directorios de forma rápida y sencilla, iniciar o detener procesos, instalar librerías, etc.

Un terminal es una forma de acceder al sistema sin utilizar la interfaz gráfica, es decir, realizar todo tipo de tareas en formato texto. La forma de utilizar el sistema de este modo es mediante órdenes.

El terminal muestra en pantalla un indicador de línea de órdenes (en inglés se utiliza la palabra prompt que literalmente traduciría "prontuario" pero se puede definir como ayuda visual o palabra que ayuda) esperando que el usuario introduzca una orden. Este indicador finaliza generalmente por un caracter $, cuando eres un usuario normal, o # cuando se es un súper usuario (administrador).

Para acceder a una terminal se puede hacer de dos formas, una es con una aplicación como el terminal de GNOME, xterm o konsole de KDE, que son emuladores de la terminal dentro de una interfaz visual. Otra forma es salirse del entorno gráfico y acceder a un entorno completamente en modo texto, algo así como entrar en sólo símbolo de sistema en Windows 98.

Tomando como base a Ellen Seiver [2] cada vez que se ejecuta una orden, el sistema operativo le abre automáticamente tres interfaces (en los sistemas operativos tipo UNIX se utiliza el término archivo): la entrada estándar, la salida estándar y el error estándar.

* La entrada estándar (stdin) se refiere al archivo por el que una orden recibe su entrada (por defecto, es el teclado).
* La salida estándar (stdout) se refiere al archivo por el que una orden presenta sus resultados (por defecto, es la pantalla o más concretamente la ventana en la que se está ejecutando el intérprete de órdenes).
* El error estándar (stderr) se refiere al archivo por el que una orden presenta los mensajes que va generando cuando ocurre un error (por defecto, también es la pantalla).

## **Tipos de usuario en Linux**

Todo sistema que este pensado para tener más de un usuario (multiusuario) tiene que darles diferentes características y capacidades a estos ya que si no se hiciera estos podrían traer unenorme problema de seguridad. Dentro de Linux se puede identificar tres tipos de usuarios en Linux:

* Usuario ROOT: También llamado superusuario o administrador. Es el único usuario con privilegios sobre el control total del sistema, y por tanto, el único que tiene derecho a administrar las cuentas de usuario del sistema.
* Usuarios normales o finales: Son los usuarios habituales del sistema, que utilizarán los recursos de éste. Cada usuario sólo podrá personalizar su entorno de trabajo.
* Usuarios especiales o de sistema: Son incorporados por el propio sistema, y se encargan de administrar los demonios del mismo. Estos usuarios no pueden iniciar sesión en el sistema ni tener un shell donde trabajar, por tanto no tienen contraseña asignada. Ejemplos son bin, daemon, adm, lp, sync, shutdown, mail, operator, squid, apache, etc.

Linux permite además la creación de grupos de usuarios, para poder establecer permisos y restricciones por grupos en lugar de por usuarios individuales.

Por defecto, cada usuario pertenecerá a un grupo que se crea de forma automática con su mismo nombre de usuario. Será el usuario root quienque pueda asignar los usuarios a los distintos grupos del sistema.

En el sistema existen una serie de grupos predefinidos por defecto. Cada usuario posee un directorio personal o directorio HOME en el que se almacena la configuración de su entorno o perfil así como los datos personales de dicho usuario. Por defecto, a ese directorio sólo tendrá acceso el mismo usuario, a no ser que él específicamente conceda permisos a otros usuarios para poder acceder.

Linux identifica a los usuarios y grupos creados en el sistema por un número, único para cada uno de ellos, y no por el nombre que le pongamos al usuario o al grupo. Lo asigna el sistema automáticamente, aunque el usuario root también lo puede establecer manualmente.

## **Direccionamiento relativo y absoluto**

Cuando se trabaja con comandos es habitual tener que pasar como parámetros archivos o directorios. Para indicar un archivo o directorio se utiliza la ruta o path, que puede ser absoluta o relativa.

Las rutas relativas indican el camino para encontrar un elemento, pero basándose en el directorio desde el que se ejecuta la orden, osea, desde el directorio en donde se encuentran posicionados. En la imagen 1 se puede observar mejor esto

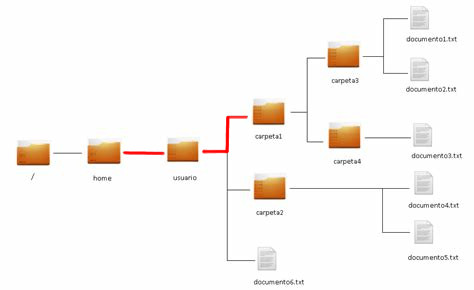


Imagen 1

Si se encuentra en el directorio /home y se quisiera mover al directorio /home/usuario/carpeta1 se debería indicar el camino hasta donde se quiere ir partiendo desde donde se encuentra.

Por otro lado las rutas absolutas son rutas que no dependen del lugar desde donde se este llamando una acción. El sistema de ficheros es una estructura jerárquica que en el caso de Linux tiene una raíz que se indica cuando se pone solamente el carácter barra / . La raíz contiene los directorios principales del sistema que a su vez tendrán subdirectorios en su interior, esta estructura de directorios se corresponde con el estándar de jerarquías de directorios FHS.

Un ejemplo de esta forma de direccionamiento se puede ejemplificar en la imagen 2. Suponiendo que se está en el directorio /home y que se quiere mover hacia el directorio /home/cristian/carpeta1 la forma de hacer esto usando una ruta absoluta seria indicando el camino desde el directorio raiz (/) hasta donde se encuentra el directorio destino

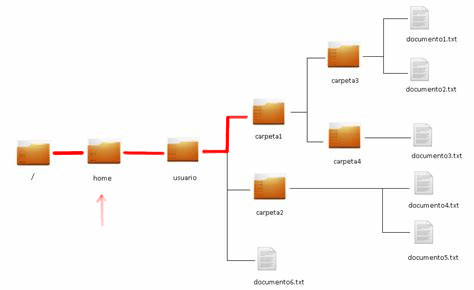
.

Imagen 2

Es muy recomendable usar estas rutas sobre las relativas ya que a pesar de que suelen ser bastante largas, tienen como ventaja que funcionan siempre, independientemente del lugar desde el que se ejecute la orden.

## **Redireccionamiento**

[4] En Linux cualquier comando que se teclea, utiliza la entrada estándar [STDIN].

La salida o resultado del comando se conoce como la salida estándar [STDOUT] y es generalmente la terminal.

Sí hay errores en la ejecución estos se conocen como la salida de error estándar [STDERR] y es también por default la terminal.

Es posible y necesario además redireccionar la entrada y la salida sobre archivos en vez de utilizar la entrada y salida estándar.

Los podemos clasificar de la siguiente manera:

FILE DESCRIPTORS – FD

* STDIN Dispositivo de entrada estándar, por defecto el teclado.
* STDOUT Dispositivo de salida estándar, por defecto la terminal o pantalla.
* STDERR Dispositivo de salida de mensajes de error estándar, por defecto la terminal o pantalla.

REDIRECTION OPERATORS

A continuación se muestra una tabla donde se detallan cada uno de estos operadores.

|  |  |
| --- | --- |
| Símbolo | Descripción |
| *>* | Redirecciona **stdout** hacía un archivo. Lo crea si no existe, si existe lo sobreescribe. |
| ls -l > lista.txt (La salida del comando se envía a un archivo en vez de la terminal.) |
| *>>* | Redirecciona **stdout** hacía un archivo. Lo crea si no existe, si existe concatena la salida al final de este. |
| ps -ef >> procesos.txt (Concatena al archivo procesos.txt la salida del comando.) |
| *<* | Redirecciona **stdin** desde un archivo. El contenido de un archivo es la entrada o input del comando. |
| mail user < texto.txt (El cuerpo del correo a enviar proviene desde un archivo, en vez del teclado).) |
| *2> 2>>* | Redirecciona **stderr** hacía un archivo. Crea (>) o concatena (>>) la salida de errores a un archivo. (ver ejemplos) |
| *1>&2* | Redirecciona **stdout** hacía donde **stderr** apunte. (ver ejemplos) |
| *2>&1* | Redirecciona **stderr** hacía donde **stdout** apunte. (ver ejemplos) |
| OTROS REDIRECCIONAMIENTOS QUE NO UTILIZAN FDs | |
| *<<* | Conocido como **HERE-DOCUMENT** o **HereDoc** (ver ejemplos) |
| *<<<* | Conocido como **HERE-STRING** (ver ejemplos) |
| *|* | El símbolo *|* (pipe) es un tipo de redireccionamiento ya que la salida (**stdout**) de un comando es la entrada (**stdin**) de otro. |
| ls /etc | grep services (La salida del comando a la izqueirda de *|* se convierte en la entrada del comando a la derecha.) |
| *tee* | El comando tee redirecciona la salida (**stdout**) a ambos, un archivo y a la terminal. |
| ps -ef | tee procesos.txt (La salida de ps se muestra en la terminal y al mismo tiempo se redirecciona al archivo *procesos.txt*. Con la opción *-a* (tee -a) concatena al archivo.) |

## **Clasificación de los comandos en Linux**

### Información del sistema

* arch: mostrar la arquitectura de la máquina (1).
* uname -m: mostrar la arquitectura de la máquina (2).
* uname -r: mostrar la versión del kernel usado.
* dmidecode -q: mostrar los componentes (hardware) del sistema.
* hdparm -i /dev/hda: mostrar las características de un disco duro.
* hdparm -tT /dev/sda: realizar prueba de lectura en un disco duro.
* cat /proc/cpuinfo: mostrar información de la CPU.
* cat /proc/interrupts: mostrar las interrupciones.
* cat /proc/meminfo: verificar el uso de memoria.
* cat /proc/swaps: mostrar ficheros swap.
* cat /proc/version: mostrar la versión del kernel.
* cat /proc/net/dev: mostrar adaptadores de red y estadísticas.
* cat /proc/mounts: mostrar el sistema de ficheros montado.
* lspci -tv: mostrar los dispositivos PCI.
* lsusb -tv: mostrar los dispositivos USB.
* date: mostrar la fecha del sistema.
* date 041217002011.00: colocar (declarar, ajustar) fecha y hora.
* clock -w: guardar los cambios de fecha en la BIOS.

### Apagar (Reiniciar Sistema o Cerrar Sesión)

* shutdown -h now: apagar el sistema (1).
* init 0: apagar el sistema (2).
* telinit 0: apagar el sistema (3).
* halt: apagar el sistema (4).
* shutdown -h hours:minutes &: apagado planificado del sistema.
* shutdown -c: cancelar un apagado planificado del sistema.
* shutdown -r now: reiniciar (1).
* reboot: reiniciar (2).
* logout: cerrar sesión.

### Archivos y Directorios

* cd /home: entrar en el directorio “home”.
* cd ..: retroceder un nivel.
* cd ../..: retroceder 2 niveles.
* cd: ir al directorio raíz.
* cd ~user1: ir al directorio user1.
* cd –: ir (regresar) al directorio anterior.
* pwd: mostrar el camino del directorio de trabajo.
* ls: ver los ficheros de un directorio.
* ls -F: ver los ficheros de un directorio.
* ls -l: mostrar los detalles de ficheros y carpetas de un directorio.
* ls -a: mostrar los ficheros ocultos.
* ls \*[0-9]\*: mostrar los ficheros y carpetas que contienen números.
* tree: mostrar los ficheros y carpetas en forma de árbol comenzando por la raíz.(1)
* lstree: mostrar los ficheros y carpetas en forma de árbol comenzando por la raíz.(2)
* mkdir dir1: crear una carpeta o directorio con nombre ‘dir1’.
* mkdir dir1 dir2: crear dos carpetas o directorios simultáneamente (Crear dos directorios a la vez).
* mkdir -p /tmp/dir1/dir2: crear un árbol de directorios.
* rm -f file1: borrar el fichero llamado ‘file1’.
* rmdir dir1: borrar la carpeta llamada ‘dir1’.
* rm -rf dir1: eliminar una carpeta llamada ‘dir1’ con su contenido de forma recursiva. (Si lo borro recursivo estoy diciendo que es con su contenido).
* rm -rf dir1 dir2: borrar dos carpetas (directorios) con su contenido de forma recursiva.
* mv dir1 new\_dir: renombrar o mover un fichero o carpeta (directorio).
* cp file1: copiar un fichero.
* cp file1 file2: copiar dos ficheros al unísono.
* cp dir /\* .: copiar todos los ficheros de un directorio dentro del directorio de trabajo actual.
* cp -a /tmp/dir1 .: copiar un directorio dentro del directorio actual de trabajo.
* cp -a dir1: copiar un directorio.
* cp -a dir1 dir2: copiar dos directorio al unísono.
* ln -s file1 lnk1: crear un enlace simbólico al fichero o directorio.
* ln file1 lnk1: crear un enlace físico al fichero o directorio.
* touch -t 0712250000 file1: modificar el tiempo real (tiempo de creación) de un fichero o directorio.
* file file1: salida (volcado en pantalla) del tipo mime de un fichero texto.
* iconv -l: listas de cifrados conocidos.
* iconv -f fromEncoding -t toEncoding inputFile > outputFile: crea una nueva forma del fichero de entrada asumiendo que está codificado en fromEncoding y convirtiéndolo a ToEncoding.
* find . -maxdepth 1 -name \*.jpg -print -exec convert ”{}” -resize 80×60 “thumbs/{}” \;: agrupar ficheros redimensionados en el directorio actual y enviarlos a directorios en vistas de miniaturas (requiere convertir desde ImagemagicK).

### Encontrar archivos

* find / -name file1: buscar fichero y directorio a partir de la raíz del sistema.
* find / -user user1: buscar ficheros y directorios pertenecientes al usuario ‘user1’.
* find /home/user1 -name \\*.bin: buscar ficheros con extensión ‘. bin’ dentro del directorio ‘/ home/user1’.
* find /usr/bin -type f -atime +100: buscar ficheros binarios no usados en los últimos 100 días.
* find /usr/bin -type f -mtime -10: buscar ficheros creados o cambiados dentro de los últimos 10 días.
* find / -name \\*.rpm -exec chmod 755 ‘{}’ \;: buscar ficheros con extensión ‘.rpm’ y modificar permisos.
* find / -xdev -name \\*.rpm: Buscar ficheros con extensión ‘.rpm’ ignorando los dispositivos removibles como cdrom, pen-drive, etc.…
* locate \\*.ps: encuentra ficheros con extensión ‘.ps’ ejecutados primeramente con el command ‘updatedb’.
* whereis halt: mostrar la ubicación de un fichero binario, de ayuda o fuente. En este caso pregunta dónde está el comando ‘halt’.
* which halt: mostrar la senda completa (el camino completo) a un binario / ejecutable.

### Montando un sistema de ficheros.

* mount /dev/hda2 /mnt/hda2: montar un disco llamado hda2. Verifique primero la existencia del directorio ‘/ mnt/hda2’; si no está, debe crearlo.
* umount /dev/hda2: desmontar un disco llamado hda2. Salir primero desde el punto ‘/ mnt/hda2.
* fuser -km /mnt/hda2: forzar el desmontaje cuando el dispositivo está ocupado.
* umount -n /mnt/hda2: correr el desmontaje sin leer el fichero /etc/mtab. Útil cuando el fichero es de solo lectura o el disco duro está lleno.
* mount /dev/fd0 /mnt/floppy: montar un disco flexible (floppy).
* mount /dev/cdrom /mnt/cdrom: montar un cdrom / dvdrom.
* mount /dev/hdc /mnt/cdrecorder: montar un cd regrabable o un dvdrom.
* mount /dev/hdb /mnt/cdrecorder: montar un cd regrabable / dvdrom (un dvd).
* mount -o loop file.iso /mnt/cdrom: montar un fichero o una imagen iso.
* mount -t vfat /dev/hda5 /mnt/hda5: montar un sistema de ficheros FAT32.
* mount /dev/sda1 /mnt/usbdisk: montar un usb pen-drive o una memoria (sin especificar el tipo de sistema de ficheros).

### Espacio de Disco

* df -h: mostrar una lista de las particiones montadas.
* ls -lSr |more: mostrar el tamaño de los ficheros y directorios ordenados por tamaño.
* du -sh dir1: Estimar el espacio usado por el directorio ‘dir1’.
* du -sk \* | sort -rn: mostrar el tamaño de los ficheros y directorios ordenados por tamaño.
* rpm -q -a –qf ‘%10{SIZE}t%{NAME}n’ | sort -k1,1n: mostrar el espacio usado por los paquetes rpm instalados organizados por tamaño (Fedora, Redhat y otros).
* dpkg-query -W -f=’${Installed-Size;10}t${Package}n’ | sort -k1,1n: mostrar el espacio usado por los paquetes instalados, organizados por tamaño (Ubuntu, Debian y otros).

### Usuarios y Grupos

* groupadd nombre\_del\_grupo: crear un nuevo grupo.
* groupdel nombre\_del\_grupo: borrar un grupo.
* groupmod -n nuevo\_nombre\_del\_grupo viejo\_nombre\_del\_grupo: renombrar un grupo.
* useradd -c “Name Surname ” -g admin -d /home/user1 -s /bin/bash user1: Crear un nuevo usuario perteneciente al grupo “admin”.
* useradd user1: crear un nuevo usuario.
* userdel -r user1: borrar un usuario (‘-r’ elimina el directorio Home).
* usermod -c “User FTP” -g system -d /ftp/user1 -s /bin/nologin user1: cambiar los atributos del usuario.
* passwd: cambiar contraseña.
* passwd user1: cambiar la contraseña de un usuario (solamente por root).
* chage -E 2011-12-31 user1: colocar un plazo para la contraseña del usuario. En este caso dice que la clave expira el 31 de diciembre de 2011.
* pwck: chequear la sintaxis correcta el formato de fichero de ‘/etc/passwd’ y la existencia de usuarios.
* grpck: chequear la sintaxis correcta y el formato del fichero ‘/etc/group’ y la existencia de grupos.
* newgrp group\_name: registra a un nuevo grupo para cambiar el grupo predeterminado de los ficheros creados recientemente.

### Permisos en Ficheros (Usa ”+” para colocar permisos y ”-” para eliminar)

* ls -lh: Mostrar permisos.
* ls /tmp | pr -T5 -W$COLUMNS: dividir la terminal en 5 columnas.
* chmod ugo+rwx directory1: colocar permisos de lectura ®, escritura (w) y ejecución(x) al propietario (u), al grupo (g) y a otros (o) sobre el directorio ‘directory1’.
* chmod go-rwx directory1: quitar permiso de lectura ®, escritura (w) y (x) ejecución al grupo (g) y otros (o) sobre el directorio ‘directory1’.
* chown user1 file1: cambiar el dueño de un fichero.
* chown -R user1 directory1: cambiar el propietario de un directorio y de todos los ficheros y directorios contenidos dentro.
* chgrp group1 file1: cambiar grupo de ficheros.
* chown user1:group1 file1: cambiar usuario y el grupo propietario de un fichero.
* find / -perm -u+s: visualizar todos los ficheros del sistema con SUID configurado.
* chmod u+s /bin/file1: colocar el bit SUID en un fichero binario. El usuario que corriendo ese fichero adquiere los mismos privilegios como dueño.
* chmod u-s /bin/file1: deshabilitar el bit SUID en un fichero binario.
* chmod g+s /home/public: colocar un bit SGID en un directorio –similar al SUID pero por directorio.
* chmod g-s /home/public: desabilitar un bit SGID en un directorio.
* chmod o+t /home/public: colocar un bit STIKY en un directorio. Permite el borrado de ficheros solamente a los dueños legítimos.
* chmod o-t /home/public: desabilitar un bit STIKY en un directorio.

### Atributos especiales en ficheros (Usa ”+” para colocar permisos y ”-” para eliminar)

* chattr +a file1: permite escribir abriendo un fichero solamente modo append.
* chattr +c file1: permite que un fichero sea comprimido / descomprimido automaticamente.
* chattr +d file1: asegura que el programa ignore borrar los ficheros durante la copia de seguridad.
* chattr +i file1: convierte el fichero en invariable, por lo que no puede ser eliminado, alterado, renombrado, ni enlazado.
* chattr +s file1: permite que un fichero sea borrado de forma segura.
* chattr +S file1: asegura que un fichero sea modificado, los cambios son escritos en modo synchronous como con sync.
* chattr +u file1: te permite recuperar el contenido de un fichero aún si este está cancelado.
* lsattr: mostrar atributos especiales.

### Archivos y Ficheros comprimidos.

* bunzip2 file1.bz2: descomprime in fichero llamado ‘file1.bz2’.
* bzip2 file1: comprime un fichero llamado ‘file1’.
* gunzip file1.gz: descomprime un fichero llamado ‘file1.gz’.
* gzip file1: comprime un fichero llamado ‘file1’.
* gzip -9 file1: comprime con compresión máxima.
* rar a file1.rar test\_file: crear un fichero rar llamado ‘file1.rar’.
* rar a file1.rar file1 file2 dir1: comprimir ‘file1’, ‘file2’ y ‘dir1’ simultáneamente.
* rar x file1.rar: descomprimir archivo rar.
* unrar x file1.rar: descomprimir archivo rar.
* tar -cvf archive.tar file1: crear un tarball descomprimido.
* tar -cvf archive.tar file1 file2 dir1: crear un archivo conteniendo ‘file1’, ‘file2′ y’dir1’.
* tar -tf archive.tar: mostrar los contenidos de un archivo.
* tar -xvf archive.tar: extraer un tarball.
* tar -xvf archive.tar -C /tmp: extraer un tarball en / tmp.
* tar -cvfj archive.tar.bz2 dir1: crear un tarball comprimido dentro de bzip2.
* tar -xvfj archive.tar.bz2: descomprimir un archivo tar comprimido en bzip2
* tar -cvfz archive.tar.gz dir1: crear un tarball comprimido en gzip.
* tar -xvfz archive.tar.gz: descomprimir un archive tar comprimido en gzip.
* zip file1.zip file1: crear un archivo comprimido en zip.
* zip -r file1.zip file1 file2 dir1: comprimir, en zip, varios archivos y directorios de forma simultánea.
* unzip file1.zip: descomprimir un archivo zip.

### Paquetes RPM (Red Hat, Fedora y similares).

* rpm -ivh package.rpm: instalar un paquete rpm.
* rpm -ivh –nodeeps package.rpm: instalar un paquete rpm ignorando las peticiones de dependencias.
* rpm -U package.rpm: actualizar un paquete rpm sin cambiar la configuración de los ficheros.
* rpm -F package.rpm: actualizar un paquete rpm solamente si este está instalado.
* rpm -e package\_name.rpm: eliminar un paquete rpm.
* rpm -qa: mostrar todos los paquetes rpm instalados en el sistema.
* rpm -qa | grep httpd: mostrar todos los paquetes rpm con el nombre “httpd”.
* rpm -qi package\_name: obtener información en un paquete específico instalado.
* rpm -qg “System Environment/Daemons”: mostar los paquetes rpm de un grupo software.
* rpm -ql package\_name: mostrar lista de ficheros dados por un paquete rpm instalado.
* rpm -qc package\_name: mostrar lista de configuración de ficheros dados por un paquete rpm instalado.
* rpm -q package\_name –whatrequires: mostrar lista de dependencias solicitada para un paquete rpm.
* rpm -q package\_name –whatprovides: mostar la capacidad dada por un paquete rpm.
* rpm -q package\_name –scripts: mostrar los scripts comenzados durante la instalación /eliminación.
* rpm -q package\_name –changelog: mostar el historial de revisions de un paquete rpm.
* rpm -qf /etc/httpd/conf/httpd.conf: verificar cuál paquete rpm pertenece a un fichero dado.
* rpm -qp package.rpm -l: mostrar lista de ficheros dados por un paquete rpm que aún no ha sido instalado.
* rpm –import /media/cdrom/RPM-GPG-KEY: importar la firma digital de la llave pública.
* rpm –checksig package.rpm: verificar la integridad de un paquete rpm.
* rpm -qa gpg-pubkey: verificar la integridad de todos los paquetes rpm instalados.
* rpm -V package\_name: chequear el tamaño del fichero, licencias, tipos, dueño, grupo, chequeo de resumen de MD5 y última modificación.
* rpm -Va: chequear todos los paquetes rpm instalados en el sistema. Usar con cuidado.
* rpm -Vp package.rpm: verificar un paquete rpm no instalado todavía.
* rpm2cpio package.rpm | cpio –extract –make-directories \*bin\*: extraer fichero ejecutable desde un paquete rpm.
* rpm -ivh /usr/src/redhat/RPMS/`arch`/package.rpm: instalar un paquete construido desde una fuente rpm.
* rpmbuild –rebuild package\_name.src.rpm: construir un paquete rpm desde una fuente rpm.

### Actualizador de paquetes YUM (Red Hat, Fedora y similares)

* yum install package\_name: descargar e instalar un paquete rpm.
* yum localinstall package\_name.rpm: este instalará un RPM y tratará de resolver todas las dependencies para ti, usando tus repositorios.
* yum update package\_name.rpm: actualizar todos los paquetes rpm instalados en el sistema.
* yum update package\_name: modernizar / actualizar un paquete rpm.
* yum remove package\_name: eliminar un paquete rpm.
* yum list: listar todos los paquetes instalados en el sistema.
* yum search package\_name: Encontrar un paquete en repositorio rpm.
* yum clean packages: limpiar un caché rpm borrando los paquetes descargados.
* yum clean headers: eliminar todos los ficheros de encabezamiento que el sistema usa para resolver la dependencia.
* yum clean all: eliminar desde los paquetes caché y ficheros de encabezado.

### Paquetes Deb (Debian, Ubuntu y derivados).

* dpkg -i package.deb: instalar / actualizar un paquete deb.
* dpkg -r package\_name: eliminar un paquete deb del sistema.
* dpkg -l: mostrar todos los paquetes deb instalados en el sistema.
* dpkg -l | grep httpd: mostrar todos los paquetes deb con el nombre “httpd”
* dpkg -s package\_name: obtener información en un paquete específico instalado en el sistema.
* dpkg -L package\_name: mostar lista de ficheros dados por un paquete instalado en el sistema.
* dpkg –contents package.deb: mostrar lista de ficheros dados por un paquete no instalado todavía.
* dpkg -S /bin/ping: verificar cuál paquete pertenece a un fichero dado.

### Actualizador de paquetes APT (Debian, Ubuntu y derivados).

* apt-get install package\_name: instalar / actualizar un paquete deb.
* apt-cdrom install package\_name: instalar / actualizar un paquete deb desde un cdrom.
* apt-get update: actualizar la lista de paquetes.
* apt-get upgrade: actualizar todos los paquetes instalados.
* apt-get remove package\_name: eliminar un paquete deb del sistema.
* apt-get check: verificar la correcta resolución de las dependencias.
* apt-get clean: limpiar cache desde los paquetes descargados.
* apt-cache search searched-package: retorna lista de paquetes que corresponde a la serie de paquetes buscados.

### Ver el contenido de un fichero.

* cat file1: ver los contenidos de un fichero comenzando desde la primera hilera.
* tac file1: ver los contenidos de un fichero comenzando desde la última línea.
* more file1: ver el contenido a lo largo de un fichero.
* less file1: parecido al commando ‘more’ pero permite salvar el movimiento en el fichero así como el movimiento hacia atrás.
* head -2 file1: ver las dos primeras líneas de un fichero.
* tail -2 file1: ver las dos últimas líneas de un fichero.
* tail -f /var/log/messages: ver en tiempo real qué ha sido añadido al fichero.

### Manipulación de texto.

* cat file1 file2 .. | command <> file1\_in.txt\_or\_file1\_out.txt: sintaxis general para la manipulación de texto utilizando PIPE, STDIN y STDOUT.
* cat file1 | command( sed, grep, awk, grep, etc…) > result.txt: sintaxis general para manipular un texto de un fichero y escribir el resultado en un fichero nuevo.
* cat file1 | command( sed, grep, awk, grep, etc…) » result.txt: sintaxis general para manipular un texto de un fichero y añadir resultado en un fichero existente.
* grep Aug /var/log/messages: buscar palabras “Aug” en el fichero ‘/var/log/messages’.
* grep ^Aug /var/log/messages: buscar palabras que comienzan con “Aug” en fichero ‘/var/log/messages’
* grep [0-9] /var/log/messages: seleccionar todas las líneas del fichero ‘/var/log/messages’ que contienen números.
* grep Aug -R /var/log/\*: buscar la cadena “Aug” en el directorio ‘/var/log’ y debajo.
* sed ‘s/stringa1/stringa2/g’ example.txt: reubicar “string1” con “string2” en ejemplo.txt
* sed ‘/^$/d’ example.txt: eliminar todas las líneas en blanco desde el ejemplo.txt
* sed ‘/ \*#/d; /^$/d’ example.txt: eliminar comentarios y líneas en blanco de ejemplo.txt
* echo ‘esempio’ | tr ‘[:lower:]’ ‘[:upper:]’: convertir minúsculas en mayúsculas.
* sed -e ‘1d’ result.txt: elimina la primera línea del fichero ejemplo.txt
* sed -n ‘/stringa1/p’: visualizar solamente las líneas que contienen la palabra “string1”.

### Establecer caracter y conversión de ficheros.

* dos2unix filedos.txt fileunix.txt: convertir un formato de fichero texto desde MSDOS a UNIX.
* unix2dos fileunix.txt filedos.txt: convertir un formato de fichero de texto desde UNIX a MSDOS.
* recode ..HTML < page.txt > page.html: convertir un fichero de texto en html.
* recode -l | more: mostrar todas las conversiones de formato disponibles.

### Análisis del sistema de ficheros.

* badblocks -v /dev/hda1: Chequear los bloques defectuosos en el disco hda1.
* fsck /dev/hda1: reparar / chequear la integridad del fichero del sistema Linux en el disco hda1.
* fsck.ext2 /dev/hda1: reparar / chequear la integridad del fichero del sistema ext 2 en el disco hda1.
* e2fsck /dev/hda1: reparar / chequear la integridad del fichero del sistema ext 2 en el disco hda1.
* e2fsck -j /dev/hda1: reparar / chequear la integridad del fichero del sistema ext 3 en el disco hda1.
* fsck.ext3 /dev/hda1: reparar / chequear la integridad del fichero del sistema ext 3 en el disco hda1.
* fsck.vfat /dev/hda1: reparar / chequear la integridad del fichero sistema fat en el disco hda1.
* fsck.msdos /dev/hda1: reparar / chequear la integridad de un fichero del sistema dos en el disco hda1.
* dosfsck /dev/hda1: reparar / chequear la integridad de un fichero del sistema dos en el disco hda1.

### Formatear un sistema de ficheros

* mkfs /dev/hda1: crear un fichero de sistema tipo Linux en la partición hda1.
* mke2fs /dev/hda1: crear un fichero de sistema tipo Linux ext 2 en hda1.
* mke2fs -j /dev/hda1: crear un fichero de sistema tipo Linux ext3 (periódico) en la partición hda1.
* mkfs -t vfat 32 -F /dev/hda1: crear un fichero de sistema FAT32 en hda1.
* fdformat -n /dev/fd0: formatear un disco flooply.
* mkswap /dev/hda3: crear un fichero de sistema swap.

### Trabajo con la SWAP.

* mkswap /dev/hda3: crear fichero de sistema swap.
* swapon /dev/hda3: activando una nueva partición swap.
* swapon /dev/hda2 /dev/hdb3: activar dos particiones swap.

### Salvas (Backup).

* dump -0aj -f /tmp/home0.bak /home: hacer una salva completa del directorio ‘/home’.
* dump -1aj -f /tmp/home0.bak /home: hacer una salva incremental del directorio ‘/home’.
* restore -if /tmp/home0.bak: restaurando una salva interactivamente.
* rsync -rogpav –delete /home /tmp: sincronización entre directorios.
* rsync -rogpav -e ssh –delete /home ip\_address:/tmp: rsync a través del túnel SSH.
* rsync -az -e ssh –delete ip\_addr:/home/public /home/local: sincronizar un directorio local con un directorio remoto a través de ssh y de compresión.
* rsync -az -e ssh –delete /home/local ip\_addr:/home/public: sincronizar un directorio remoto con un directorio local a través de ssh y de compresión.
* dd bs=1M if=/dev/hda | gzip | ssh user@ip\_addr ‘dd of=hda.gz’: hacer una salva de un disco duro en un host remoto a través de ssh.
* dd if=/dev/sda of=/tmp/file1: salvar el contenido de un disco duro a un fichero. (En este caso el disco duro es “sda” y el fichero “file1”).
* tar -Puf backup.tar /home/user: hacer una salva incremental del directorio ‘/home/user’.
* ( cd /tmp/local/ && tar c . ) | ssh -C user@ip\_addr ‘cd /home/share/ && tar x -p’: copiar el contenido de un directorio en un directorio remoto a través de ssh.
* ( tar c /home ) | ssh -C user@ip\_addr ‘cd /home/backup-home && tar x -p’: copiar un directorio local en un directorio remoto a través de ssh.
* tar cf – . | (cd /tmp/backup ; tar xf – ): copia local conservando las licencias y enlaces desde un directorio a otro.
* find /home/user1 -name ‘\*.txt’ | xargs cp -av –target-directory=/home/backup/ –parents: encontrar y copiar todos los ficheros con extensión ‘.txt’ de un directorio a otro.
* find /var/log -name ‘\*.log’ | tar cv –files-from=- | bzip2 > log.tar.bz2: encontrar todos los ficheros con extensión ‘.log’ y hacer un archivo bzip.
* dd if=/dev/hda of=/dev/fd0 bs=512 count=1: hacer una copia del MRB (Master Boot Record) a un disco floppy.
* dd if=/dev/fd0 of=/dev/hda bs=512 count=1: restaurar la copia del MBR (Master Boot Record) salvada en un floppy.

### CD-ROM.

* cdrecord -v gracetime=2 dev=/dev/cdrom -eject blank=fast -force: limpiar o borrar un cd regrabable.
* mkisofs /dev/cdrom > cd.iso: crear una imagen iso de cdrom en disco.
* mkisofs /dev/cdrom | gzip > cd\_iso.gz: crear una imagen comprimida iso de cdrom en disco.
* mkisofs -J -allow-leading-dots -R -V “Label CD” -iso-level 4 -o ./cd.iso data\_cd: crear una imagen iso de un directorio.
* cdrecord -v dev=/dev/cdrom cd.iso: quemar una imagen iso.
* gzip -dc cd\_iso.gz | cdrecord dev=/dev/cdrom –: quemar una imagen iso comprimida.
* mount -o loop cd.iso /mnt/iso: montar una imagen iso.
* cd-paranoia -B: llevar canciones de un cd a ficheros wav.
* cd-paranoia – ”-3”: llevar las 3 primeras canciones de un cd a ficheros wav.
* cdrecord –scanbus: escanear bus para identificar el canal scsi.
* dd if=/dev/hdc | md5sum: hacer funcionar un md5sum en un dispositivo, como un CD.

### Trabajo con la RED ( LAN y Wi-Fi)

* ifconfig eth0: mostrar la configuración de una tarjeta de red Ethernet.
* ifup eth0: activar una interface ‘eth0’.
* ifdown eth0: deshabilitar una interface ‘eth0’.
* ifconfig eth0 192.168.1.1 netmask 255.255.255.0: configurar una dirección IP.
* ifconfig eth0 promisc: configurar ‘eth0’en modo común para obtener los paquetes (sniffing).
* dhclient eth0: activar la interface ‘eth0’ en modo dhcp.
* route -n: mostrar mesa de recorrido.
* route add -net 0/0 gw IP\_Gateway: configurar entrada predeterminada.
* route add -net 192.168.0.0 netmask 255.255.0.0 gw 192.168.1.1: configurar ruta estática para buscar la red ‘192.168.0.0/16’.
* route del 0/0 gw IP\_gateway: eliminar la ruta estática.
* echo “1” > /proc/sys/net/ipv4/ip\_forward: activar el recorrido ip.
* hostname: mostrar el nombre del host del sistema.
* host www.example.com: buscar el nombre del host para resolver el nombre a una dirección ip(1).
* nslookup www.example.com: buscar el nombre del host para resolver el nombre a una direccióm ip y viceversa(2).
* ip link show: mostar el estado de enlace de todas las interfaces.
* mii-tool eth0: mostar el estado de enlace de ‘eth0’.
* ethtool eth0: mostrar las estadísticas de tarjeta de red ‘eth0’.
* netstat -tup: mostrar todas las conexiones de red activas y sus PID.
* netstat -tupl: mostrar todos los servicios de escucha de red en el sistema y sus PID.
* tcpdump tcp port 80: mostrar todo el tráfico HTTP.
* iwlist scan: mostrar las redes inalámbricas.
* iwconfig eth1: mostrar la configuración de una tarjeta de red inalámbrica.
* whois www.example.com: buscar en base de datos Whois.

### Tablas IP (CORTAFUEGOS)

* iptables -t filter -L: mostrar todas las cadenas de la tabla de filtro.
* iptables -t nat -L: mostrar todas las cadenas de la tabla nat.
* iptables -t filter -F: limpiar todas las reglas de la tabla de filtro.
* iptables -t nat -F: limpiar todas las reglas de la tabla nat.
* iptables -t filter -X: borrar cualquier cadena creada por el usuario.
* iptables -t filter -A INPUT -p tcp –dport telnet -j ACCEPT: permitir las conexiones telnet para entar.
* iptables -t filter -A OUTPUT -p tcp –dport http -j DROP: bloquear las conexiones HTTP para salir.
* iptables -t filter -A FORWARD -p tcp –dport pop3 -j ACCEPT: permitir las conexiones POP a una cadena delantera.
* iptables -t filter -A INPUT -j LOG –log-prefix “DROP INPUT”: registrando una cadena de entrada.
* iptables -t nat -A POSTROUTING -o eth0 -j MASQUERADE: configurar un PAT (Puerto de traducción de dirección) en eth0, ocultando los paquetes de salida forzada.
* iptables -t nat -A PREROUTING -d 192.168.0.1 -p tcp -m tcp –dport 22 -j DNAT –to-destination 10.0.0.2:22: redireccionar los paquetes diriguidos de un host a otro.

### Monitoreando y depurando.

* top: mostrar las tareas de linux usando la mayoría cpu.
* ps -eafw: muestra las tareas Linux.
* ps -e -o pid,args –forest: muestra las tareas Linux en un modo jerárquico.
* pstree: mostrar un árbol sistema de procesos.
* kill -9 ID\_Processo: forzar el cierre de un proceso y terminarlo.
* kill -1 ID\_Processo: forzar un proceso para recargar la configuración.
* lsof -p $$: mostrar una lista de ficheros abiertos por procesos.
* lsof /home/user1: muestra una lista de ficheros abiertos en un camino dado del sistema.
* strace -c ls >/dev/null: mostrar las llamadas del sistema hechas y recibidas por un proceso.
* strace -f -e open ls >/dev/null: mostrar las llamadas a la biblioteca.
* watch -n1 ‘cat /proc/interrupts’: mostrar interrupciones en tiempo real.
* last reboot: mostrar historial de reinicio.
* lsmod: mostrar el kernel cargado.
* free -m: muestra el estado de la RAM en megabytes.
* smartctl -A /dev/hda: monitorear la fiabilidad de un disco duro a través de SMART.
* smartctl -i /dev/hda: chequear si SMART está activado en un disco duro.
* tail /var/log/dmesg: mostrar eventos inherentes al proceso de carga del kernel.
* tail /var/log/messages: mostrar los eventos del sistema.

### Comandos adicionales.

* apropos …keyword: mostrar una lista de comandos que pertenecen a las palabras claves de un programa; son útiles cuando tú sabes qué hace tu programa, pero de sconoces el nombre del comando.
* man ping: mostrar las páginas del manual on-line; por ejemplo, en un comando ping, usar la opción ‘-k’ para encontrar cualquier comando relacionado.
* whatis …keyword: muestra la descripción de lo que hace el programa.
* mkbootdisk –device /dev/fd0 `uname -r`: crear un floppy boteable.
* gpg -c file1: codificar un fichero con guardia de seguridad GNU.
* gpg file1.gpg: decodificar un fichero con Guardia de seguridad GNU.
* wget -r www.example.com: descargar un sitio web completo.
* wget -c www.example.com/file.iso: descargar un fichero con la posibilidad de parar la descargar y reanudar más tarde.
* echo ‘wget -c www.example.com/files.iso‘ | at 09:00: Comenzar una descarga a cualquier hora. En este caso empezaría a las 9 horas.
* ldd /usr/bin/ssh: mostrar las bibliotecas compartidas requeridas por el programa ssh.
* alias hh=’history’: colocar un alias para un commando –hh= Historial.
* chsh: cambiar el comando Shell.
* chsh –list-shells: es un comando adecuado para saber si tienes que hacer remoto en otra terminal.
* who -a: mostrar quien está registrado, e imprimir hora del último sistema de importación, procesos muertos, procesos de registro de sistema, procesos activos producidos por init, funcionamiento actual y últimos cambios del reloj del sistema.

## **Variables de entorno**

[5]Las variables de entorno son valores dinámicos que afectan a los programas o procesos que se ejecutan en un servidor. Existen en todos los sistemas operativos y su tipo puede variar. Las variables de entorno se pueden crear, editar, guardar y eliminar.

En Linux, las variables de entorno son marcadores de posición para la información almacenada dentro del sistema que pasa datos a los programas iniciados en shells (intérpretes de comando) o sub-shells.

[6] Las variables de entorno se utilizan para almacenar algunos valores que pueden ser utilizados por los scripts desde el shell. Este sistema operativo nos permite crear, crear nuestras propias variables.

El shell de bash tiene dos tipos de variables de entorno:

Variables globales

El sistema Linux establece algunas variables de entorno globales cuando inicias sesión en tu sistema y siempre son en letras mayúsculas para diferenciarlas de las variables de entorno definidas por el usuario podemos verlas, escribiendo el comando printenv. Podemos imprimir solo una de ellas, escribiendo el comando echo seguido de $NombreDeVariable.

Variables locales

Linux también define algunas variables de entorno locales estándar de forma predeterminada.

Para ver las variables globales y locales del shell que está ejecutando y las que están disponible para ese shell, escribe el comando set.

Declarar variables locales

Para declarar tus propias variables de entorno directamente desde el shell, escribe el nombreDeVariable que desees, seguido de un signo igual y el valor SIN espacios:

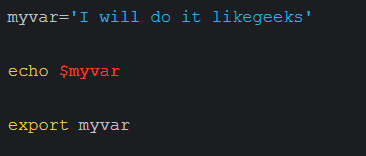
miVarible=Hola

Si quieres que tu variable sea una cadena solo debes poner su valor entre comillas

miVarible=’Hola a todos’

Declarar variables globales (variables de exportación)

Para declarar una variable de entorno global, debes declarar una variable de entorno local y luego utilizar el comando export de la siguiente manera:



Sin embargo cuando se declara de esta manera al salir del Shell la variable desaparecerá.

Si se desea que sea permanente debe hacerse de la siguiente manera:

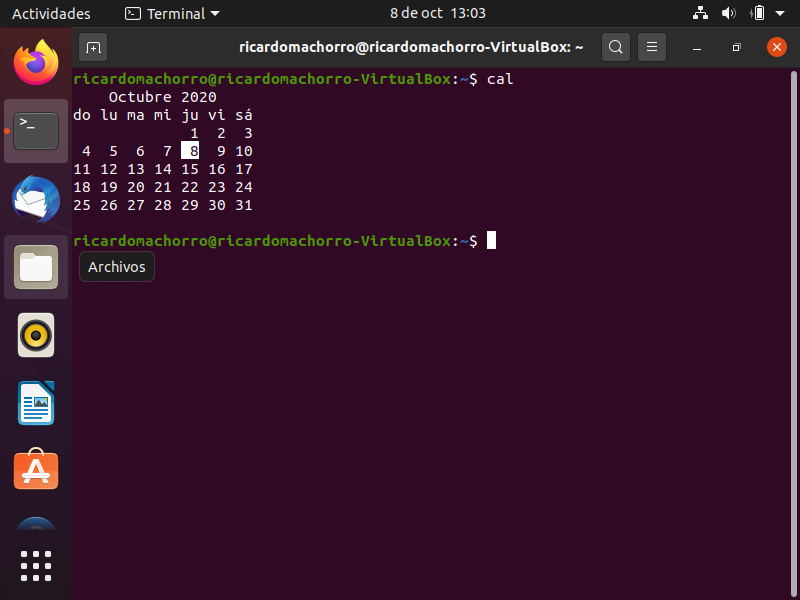




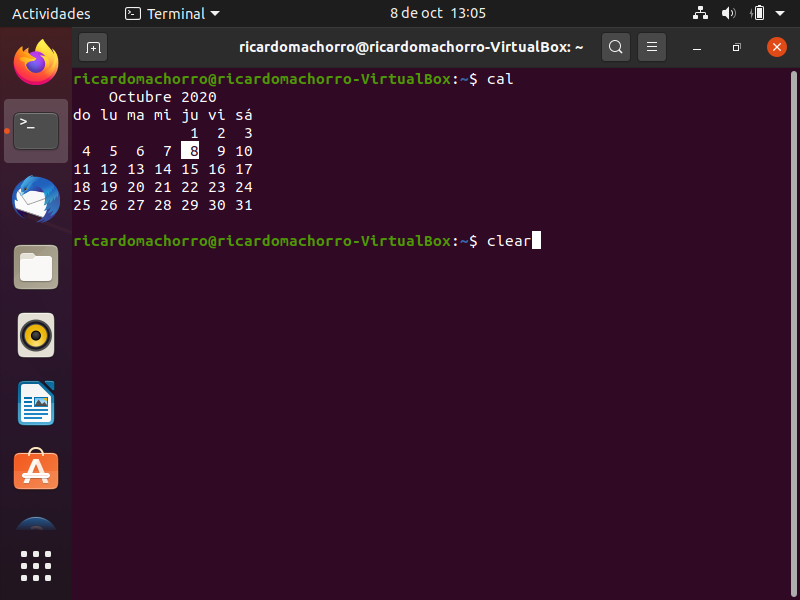
# **Desarrollo (Códigos y ventanas de ejecución)**

|  |  |
| --- | --- |
| **COMANDO** | **FUNCIONAMIENTO** |
| **cal** | Muestra un calendario en la salida standard. |
| **clear** | Es un comando de linux que borra toda la información del terminal y lo deja como si fuera nuevo. |
| **apt** | es un programa de gestión de paquetes creado por el proyecto Debian. |
| **rm** | es un comando de la familia de sistemas operativos Unix usada para eliminar archivos y directorios del sistema de archivos. |
| **date** | es un comando existente en sistemas Unix y tipo unix que muestra la hora y la fecha del sistema y el administrador también puede cambiarla. |
| **ifconfig** | es un programa disponible en varias versiones del sistema operativo UNIX, que permite configurar o desplegar numerosos parámetros de las interfaces de red residentes en el núcleo, como la dirección IP (dinámica o estática), o la máscara de red. |
| **exit** | es un comando que sale del shell actual. salida ejecutivo puede pagar el valor de estado de salida especificado |
| **mv** | es un comando de Unix usado para mover o renombrar archivos o directorios del sistema de archivos. |
| **echo** | es un comando para la impresión de un texto en pantalla. |
| **df** | es un comando que nos informa acerca del espacio total, ocupado y libre en nuestro sistema. |
| **ps** | es un comando asociado en el sistema operativo UNIX (estandarizado en POSIX y otros) que permite visualizar el estado de un Proceso (informática). |
| **more** | es un comando para ver (pero no modificar) el contenido de un archivo o comando y visualizarlo por páginas. |
| **time** | es un comando utilizado para determinar el tiempo de ejecución de una operación específica. |
| **du** | es un comando estándar de los sistemas operativos de la familia Unix. Se usa para estimar el uso de espacio en disco duro de un archivo, un directorio en particular o de archivos en un sistema de archivos. |
| **ps -fea** | Muestra una instantánea de "todos" los procesos del sistema, los parámetros con los que se levanto el proceso y también los procesos de otros usuarios |
| **less** | El comando Less de linux se usa para mostrar el texto en la pantalla del terminal. |
| **uname** | uname sirve para imprimir información del sistema linux. |
| **pstree** | es un comando poderoso y útil para mostrar procesos en ejecución en Linux. |
| **man** | es una herramienta de sistemas Unix que se utiliza para documentar y aprender sobre comandos, archivos, llamadas de sistema, etc., en un sistema operativo tal como GNU/Linux. |
| **mkdir** | es una orden de los sistemas operativos UNIX, DOS, OS/2 y Microsoft Windows usada para crear un nuevo subdirectorio o carpeta del sistema de archivos |
| **w** | es un programa de Unix que muestra información sobre los usuarios que han iniciado sesión en el sistema y qué están haciendo. |
| **kill -1-9** | Fuerza a que un proceso termine |
| **cat** | Concatena archivos y los muestra en el salida estándar, es utilizado en el sistema operativo Unix con licencia GNU. |
| **pico** | es un sencillo editor de texto basado en el editor del sistema de mensajes Pine. Al igual que en Pine, los comandos son desplegados en la parte inferior de la pantalla, y se proporciona ayuda sensible al contexto. Conforme |
| **who** | Este comando muestra rápidamente los usuarios que actualmente están autentificados en el sistema. |
| **trap -l** |  |
| **fg** | Reanuda trabajos suspendidos poniéndolos en foreground (primer plano) o trabajos en background los pasa a primer plano. |
| **nano** | es un editor de texto para sistemas Unix basado en curses. |
| **bash** | Es un programa informático cuya función consiste en interpretar órdenes. |
| **pwd** | se utiliza para imprimir el nombre del directorio actual en una sesión de comandos bajo un sistema operativo Unix o derivado. |
| **cd** | sirve para cambiar la ruta actual de la terminal y ubicarnos en una carpeta o directorio específico. |
| **vi** | vi es un editor de texto que maneja en memoria el texto entero de un archivo. Es el editor clásico de UNIX (se encuentra en todas las versiones). |
| **wc** | es un comando utilizado en el sistema operativo Unix que permite realizar diferentes conteos desde la entrada estándar, ya sea de palabras, caracteres o saltos de líneas. |
| **su** |  |
| **ls** | es muy útil para ver los archivos y directorios que tenemos dentro del directorio en el que estamos. |
| **apt-get** | Instala/informa sobre los paquetes resolviendo las dependencias, los paquetes que instala los consigne de Internet (de /etc/apt/sources.list). |
| **sudo** | que permite a los usuarios ejecutar programas con los privilegios de seguridad de otro usuario (normalmente el usuario root) de manera segura, convirtiéndose así temporalmente en súper usuario. |
| **ls -la** |  |

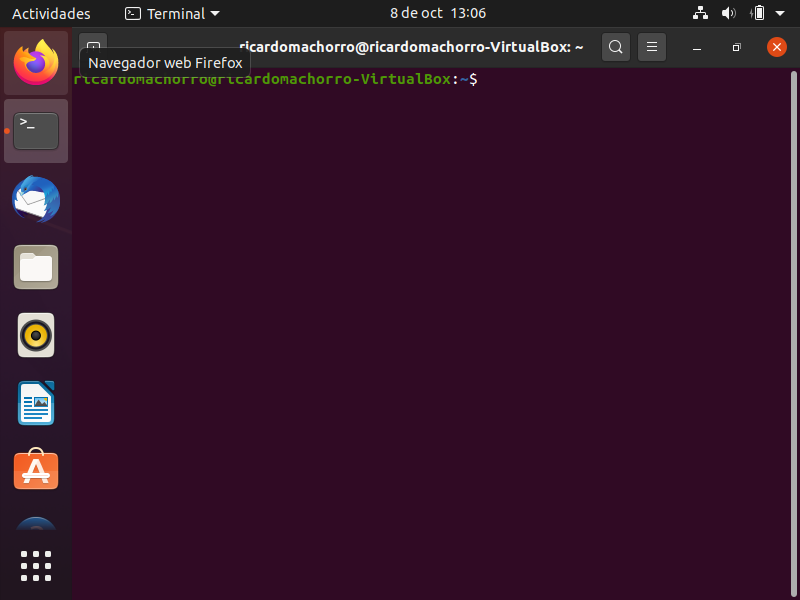
## **Comando cal**



## **Comando clear antes de ejecutarse**

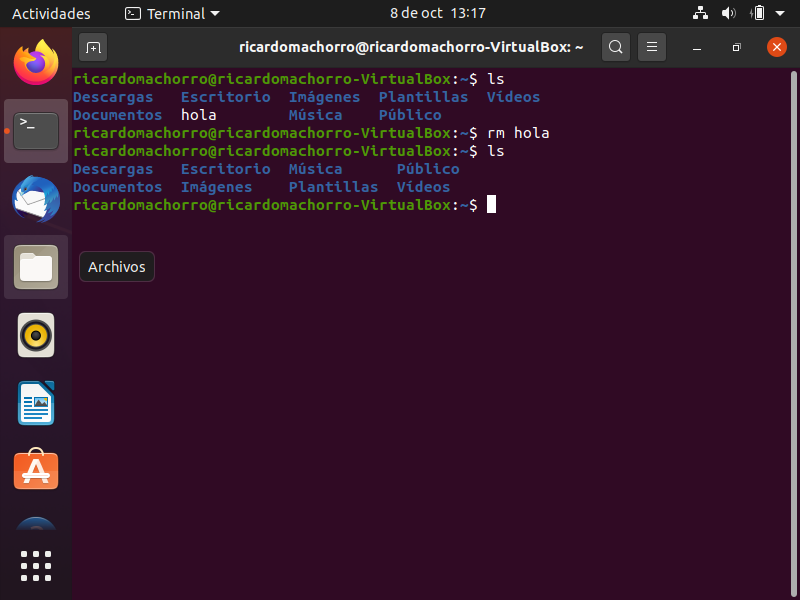


## **Comando clear después de ejecutarse**

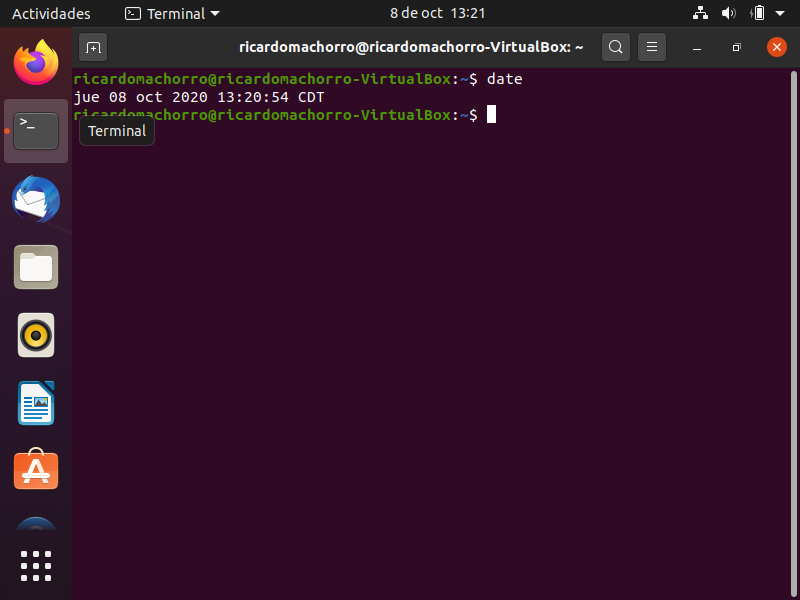


## C:\Users\Ricardo Alberto\Documents\Quinto Smeestre\FuncionApt.png**Comando apt**

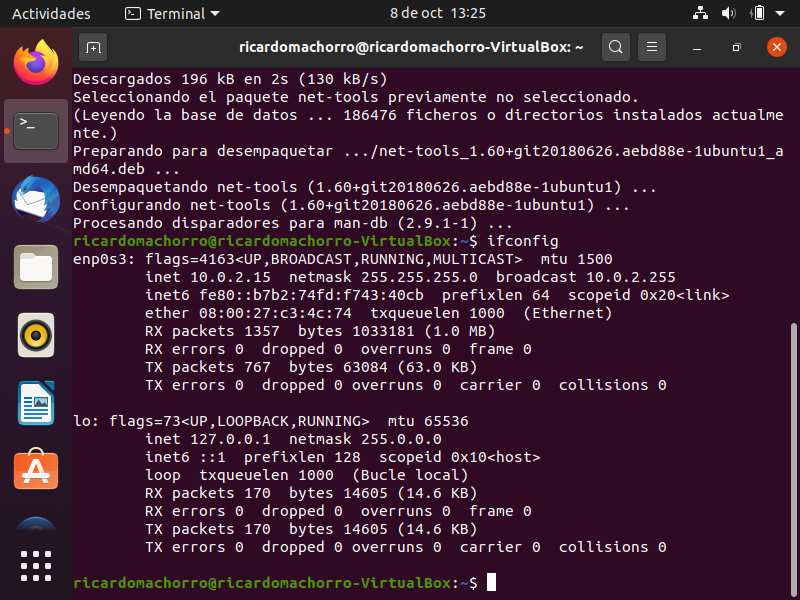
## **Comando rm**



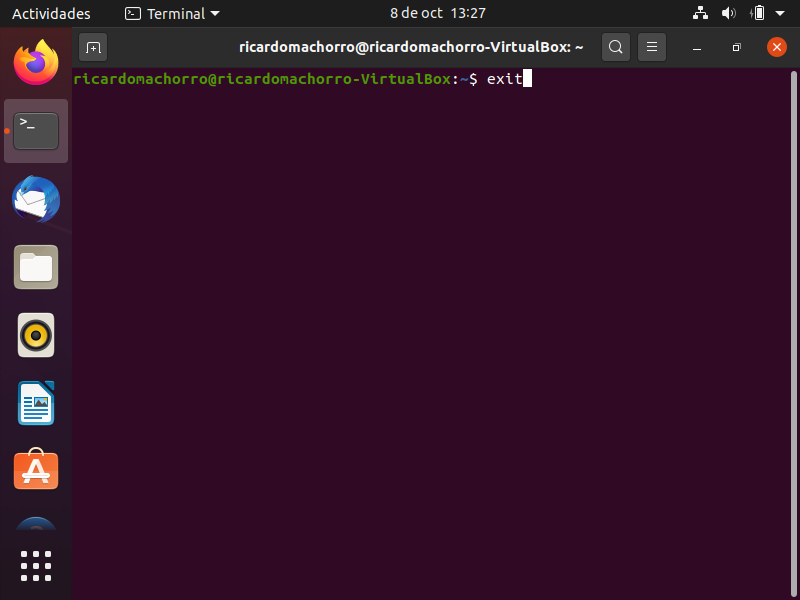
## **Comando Date.**



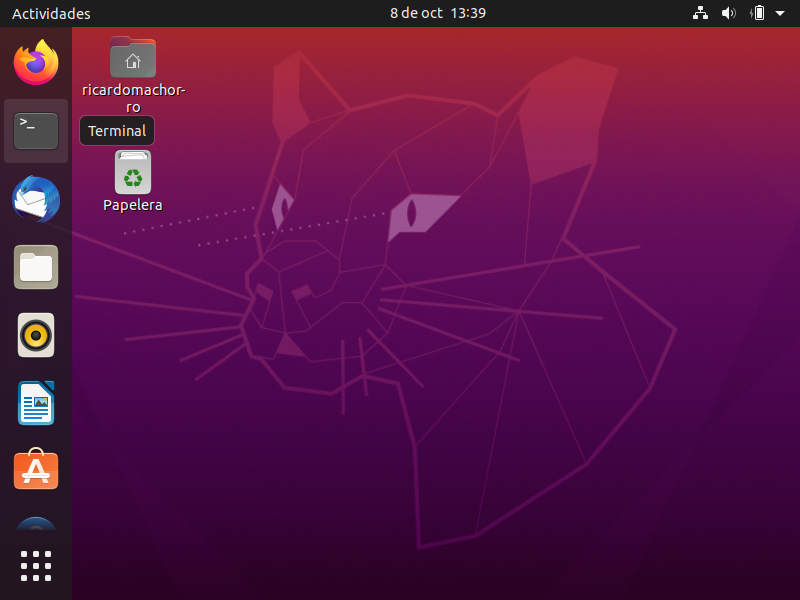
## **Comando ifconfig.**



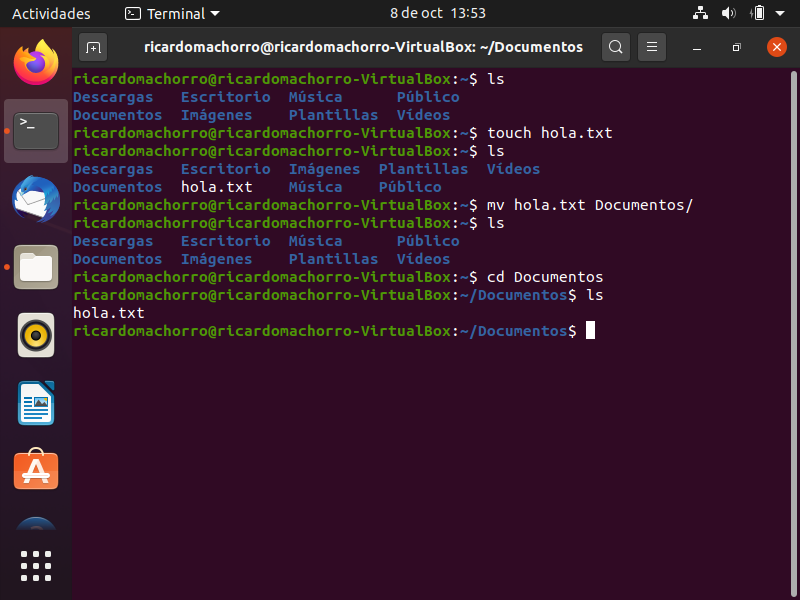
## **Comando exit antes de ejecutarse.**



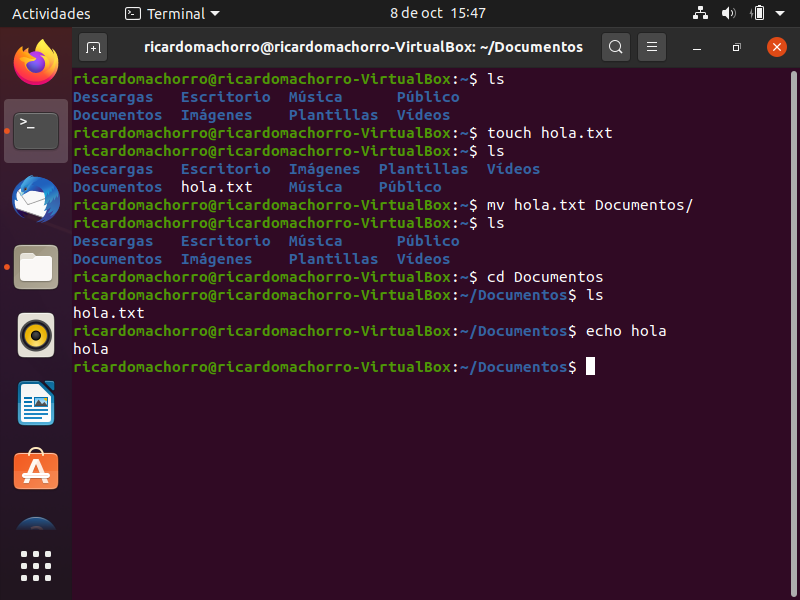
## **Comando exit después de ejecutarse**



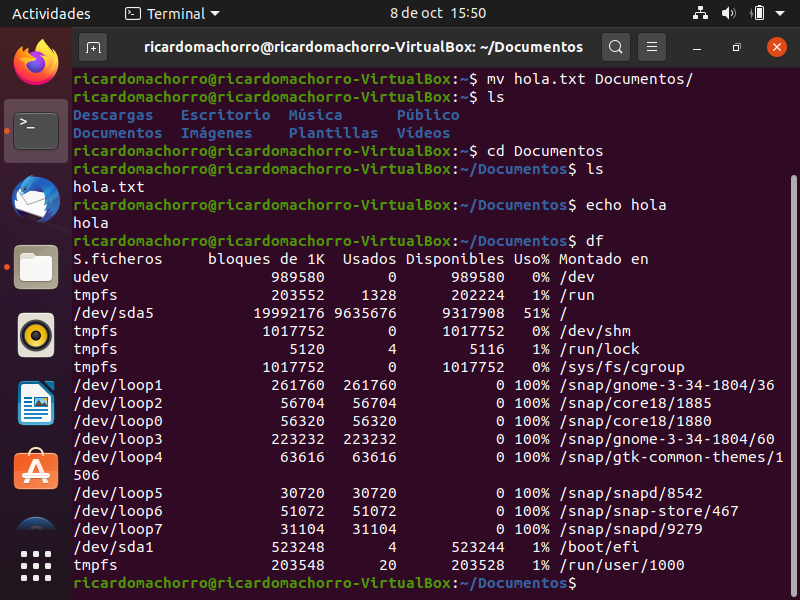
## **Comando mv.**



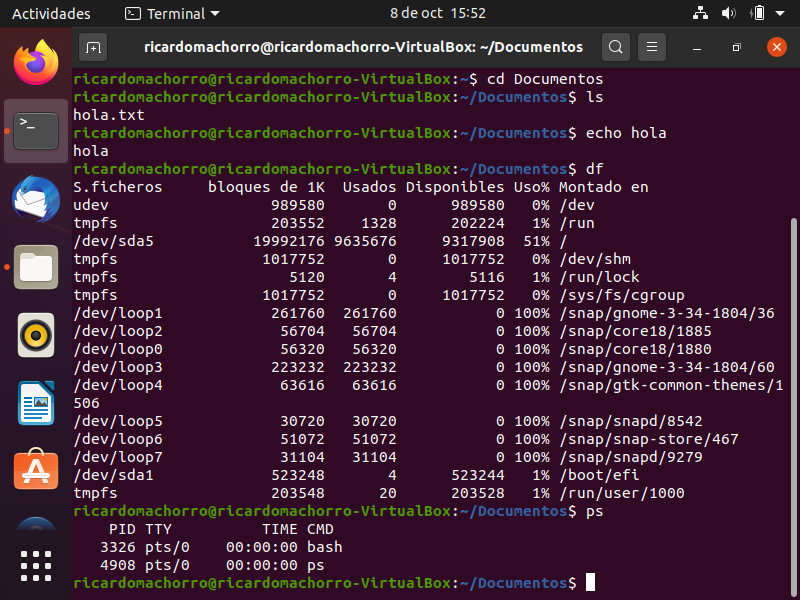
## **Comando echo**



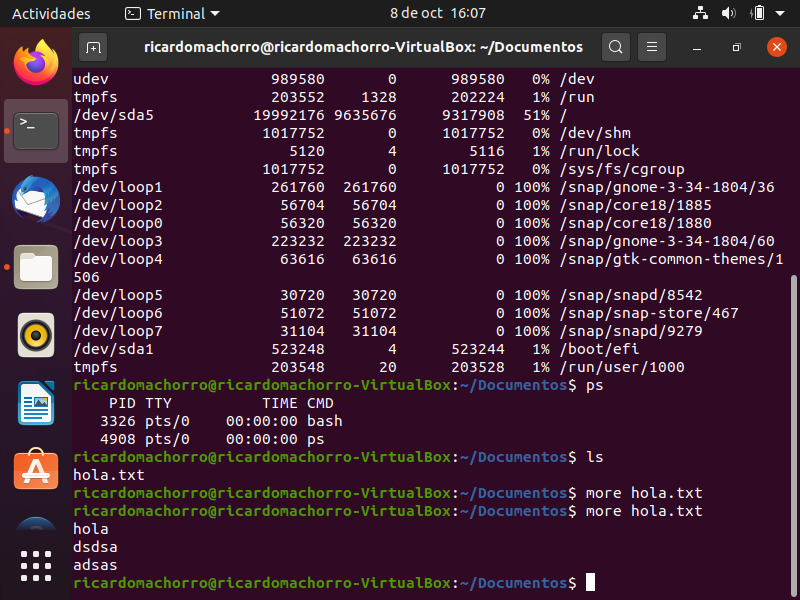
## **Comando df**



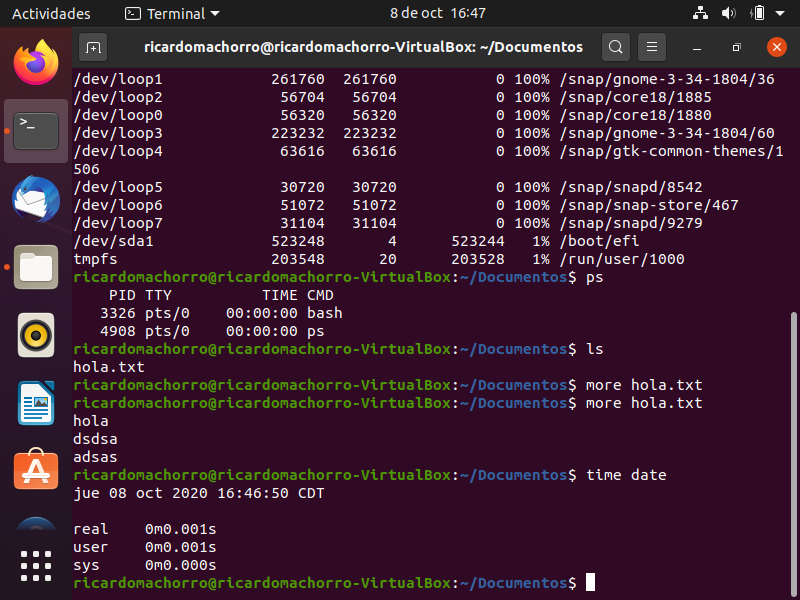
## **Comando ps**



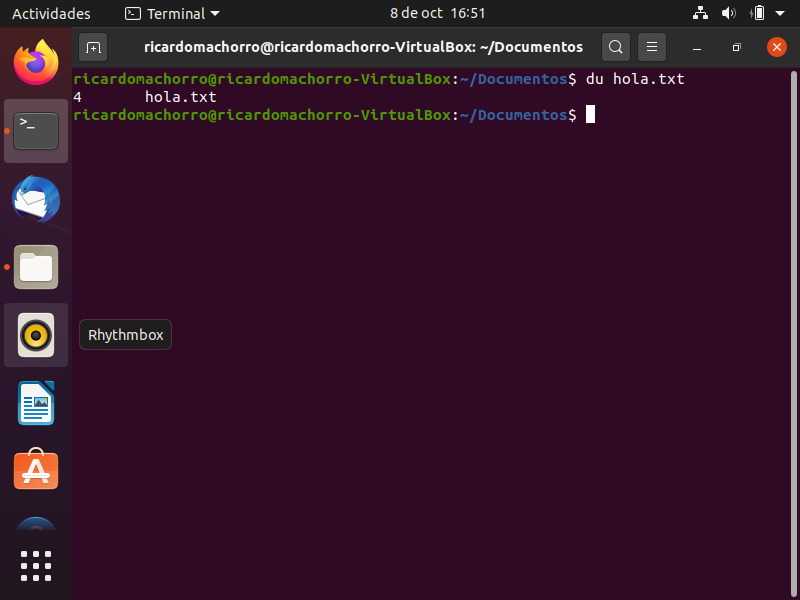
## **Comando more**



## **Comando time**



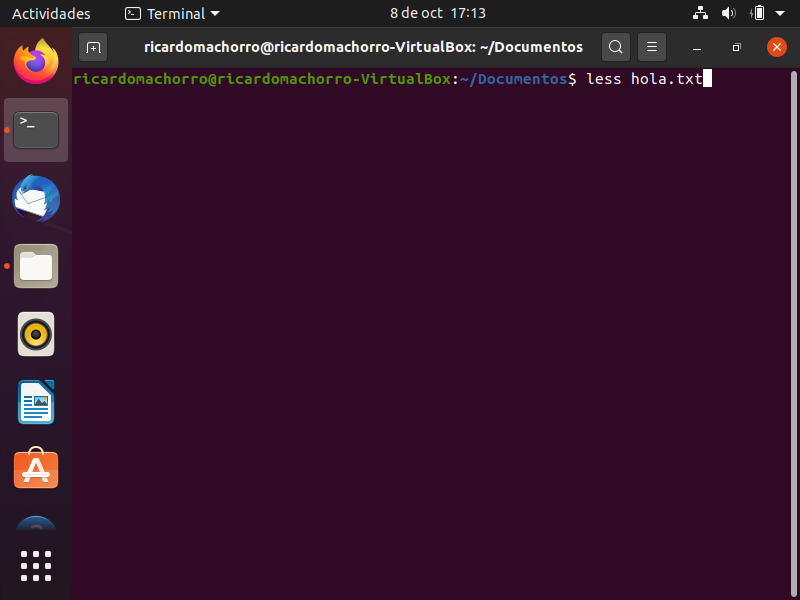
## **Comando du**



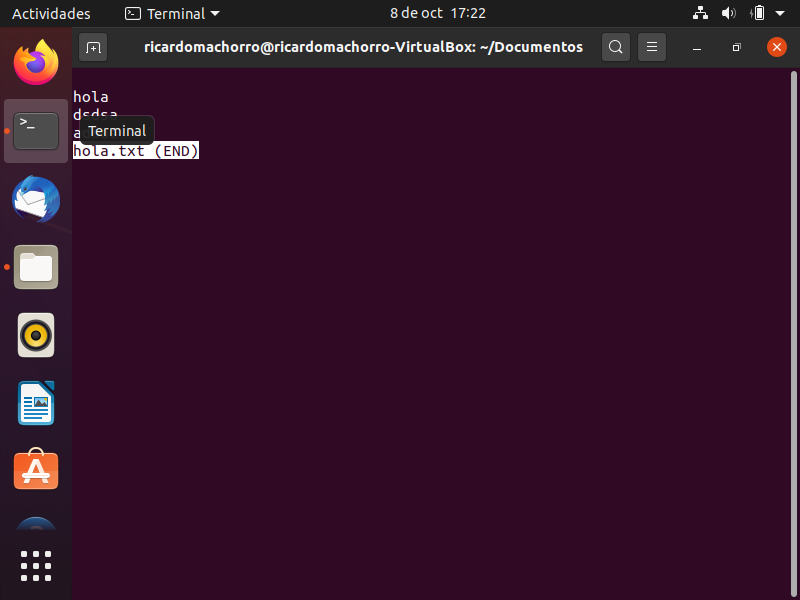
## **Comando ps –fea**



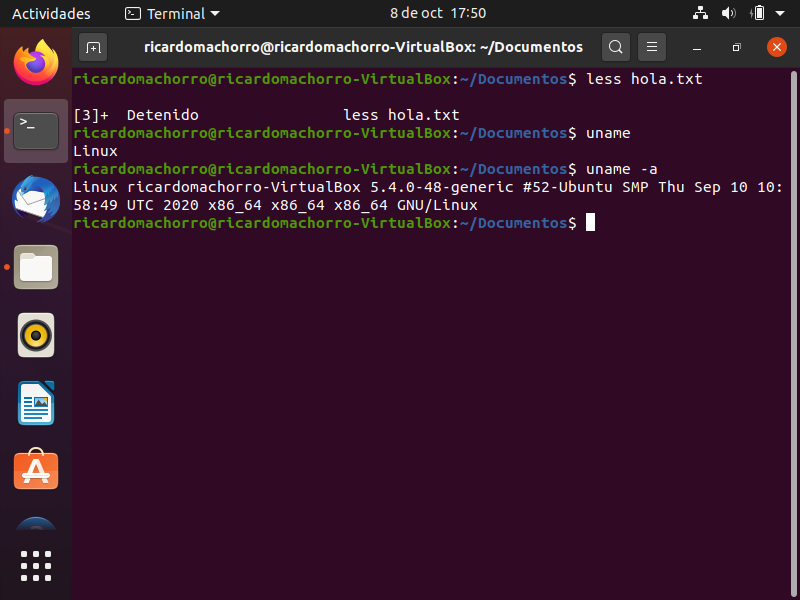
## **Comando less antes de ejecutarse**



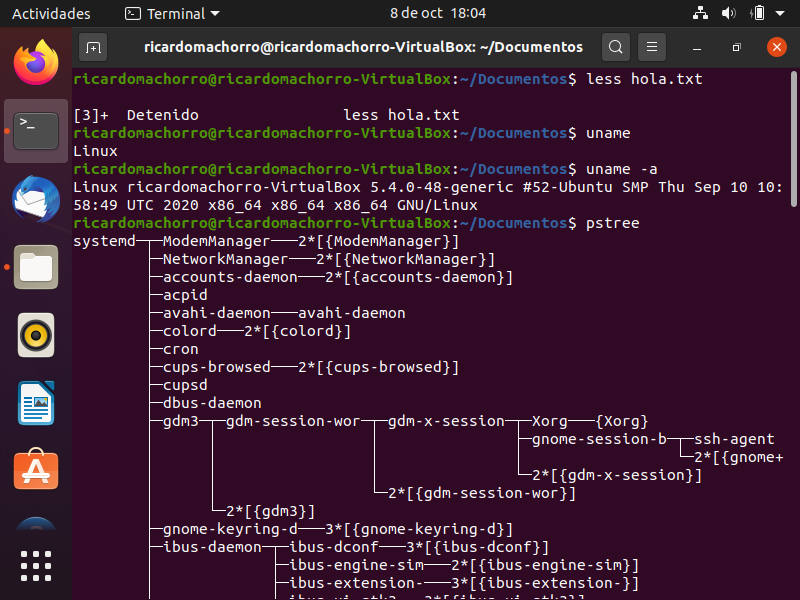
## **Comando Less después de ejecutarse**



## **Función uname**



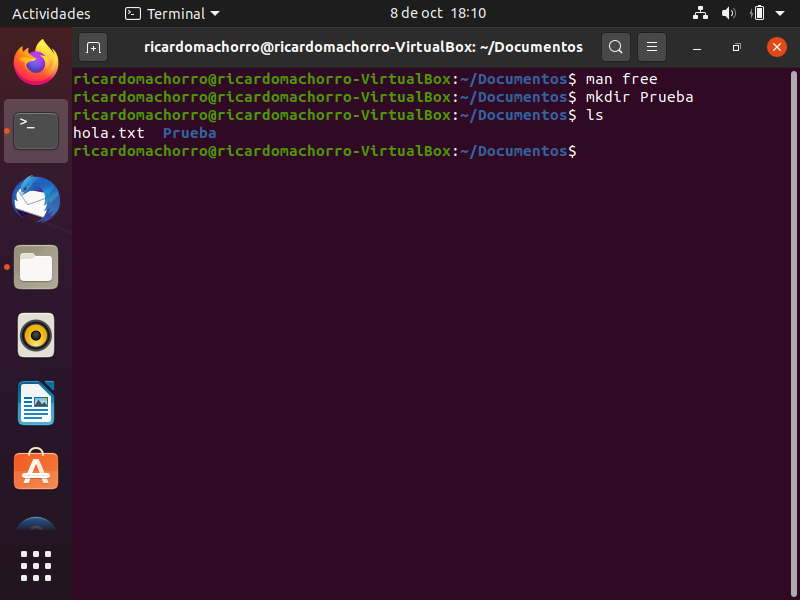
## **Comando pstree**



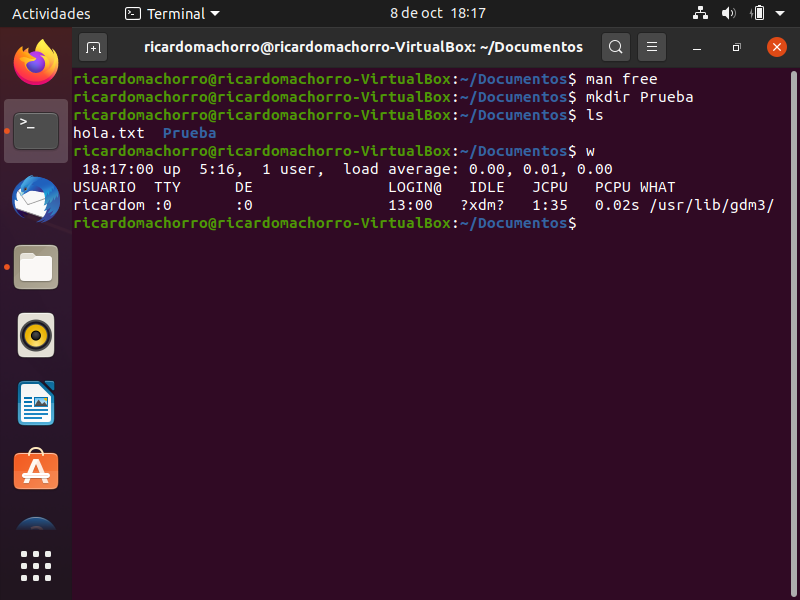
## **Comando man**



## **Comando mkdir**

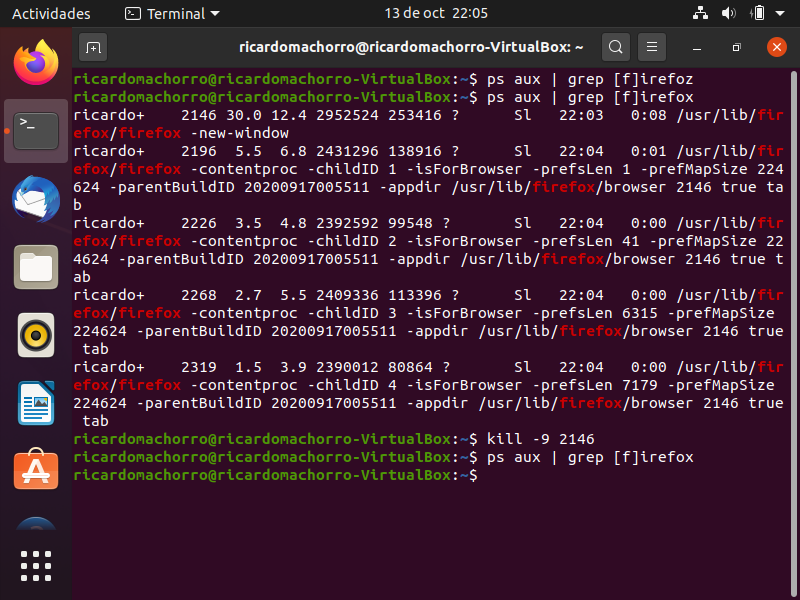


## **Comando w**

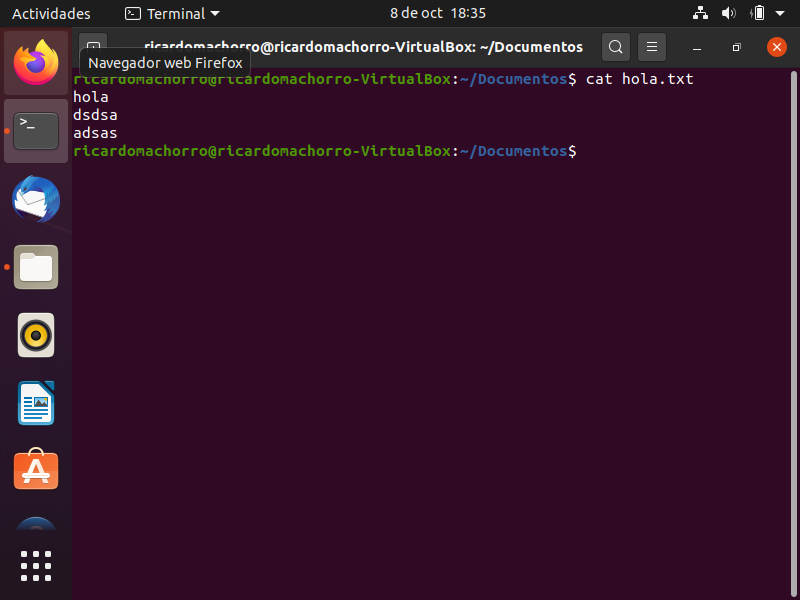


## 

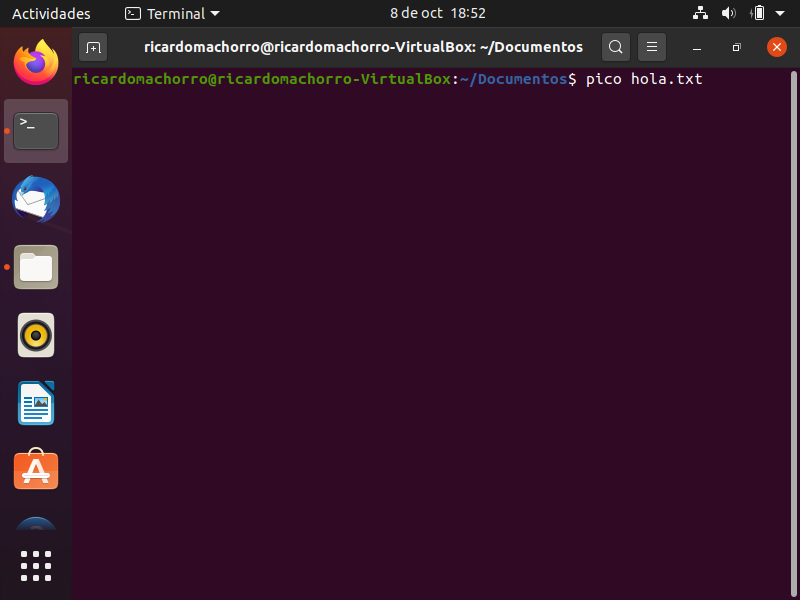
## **Comando Kill -9**



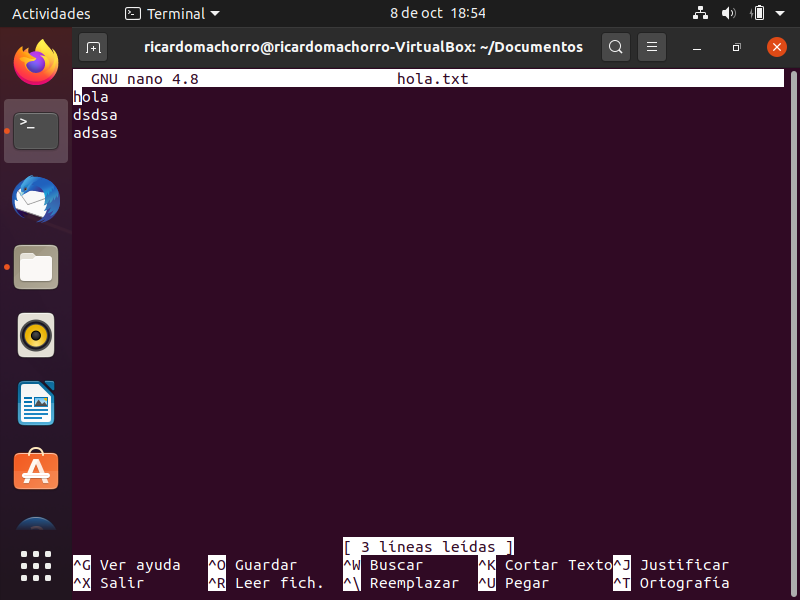
## **Función cat**



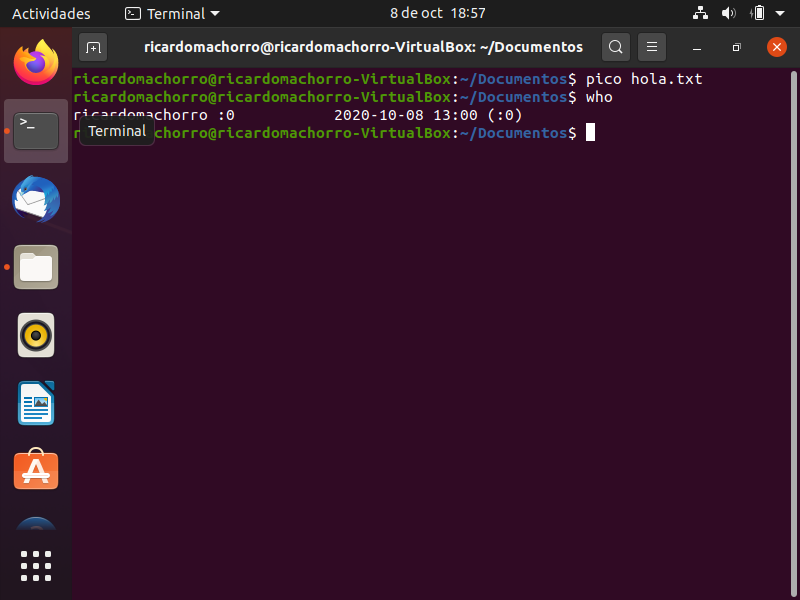
## **Comando pico antes de copilarse**



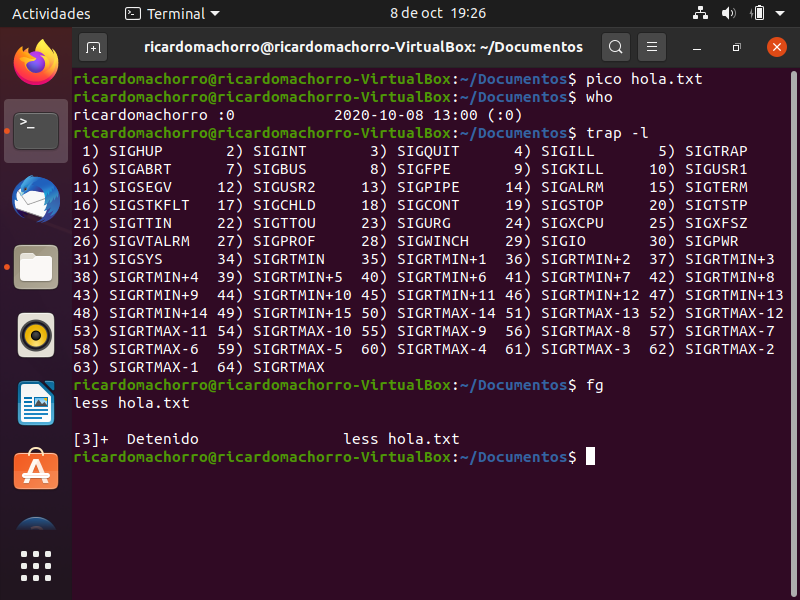
## **Función pico después de copilarse**



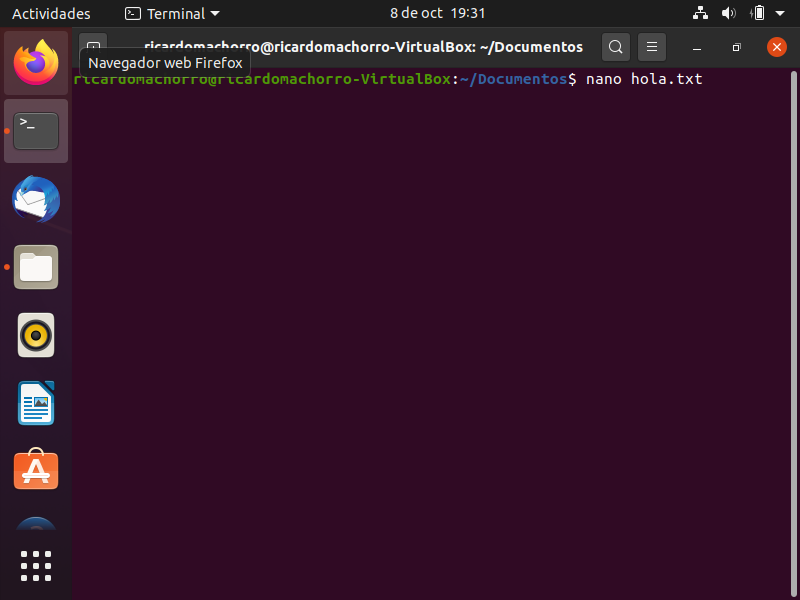
## **Comando who**



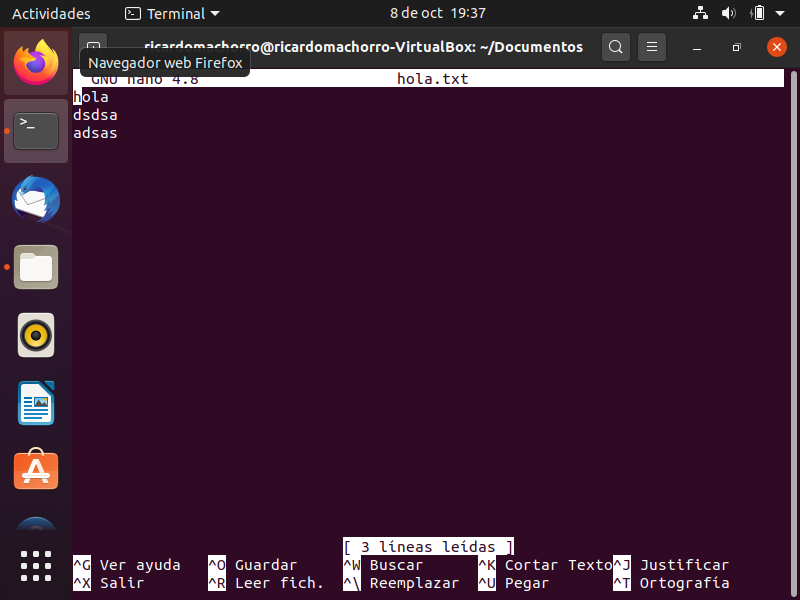
## **Comando fg**



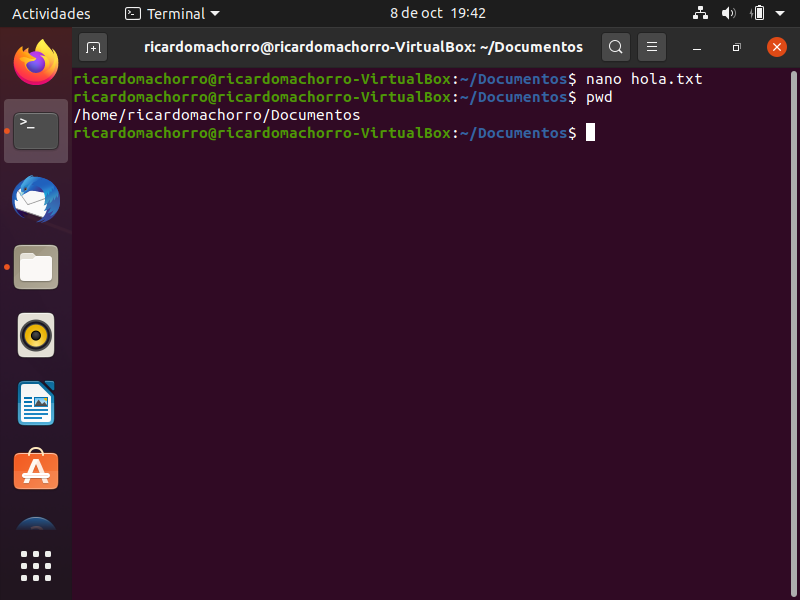
## **Comando nano antes de ejecutarse**



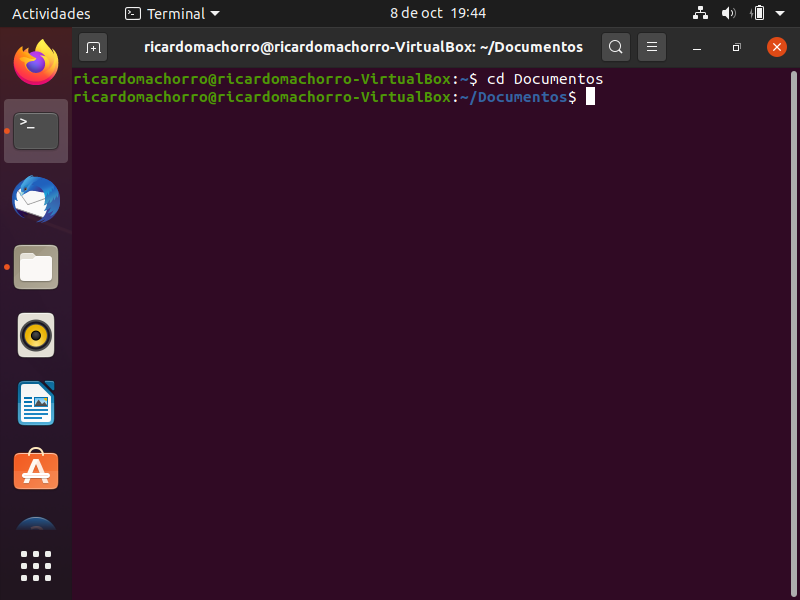
## **Comando nano después de ejecutarse**



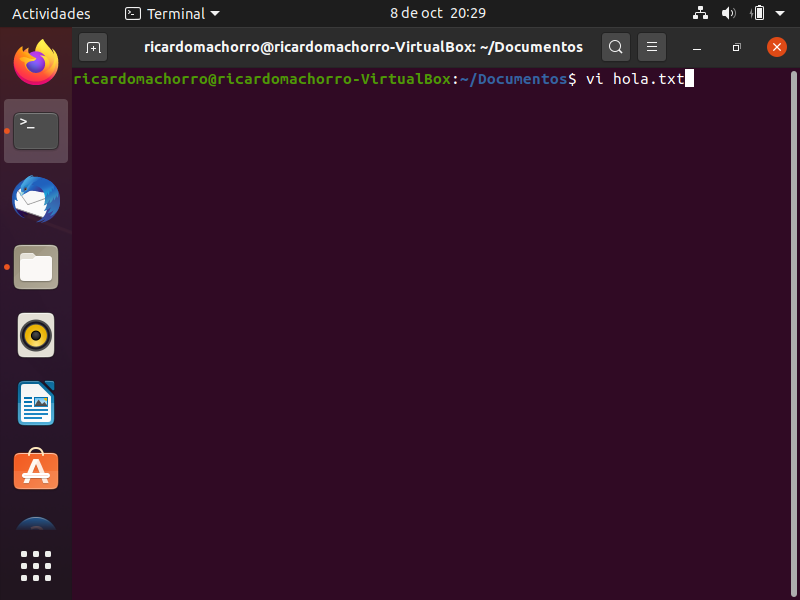
## **Comando pwd**



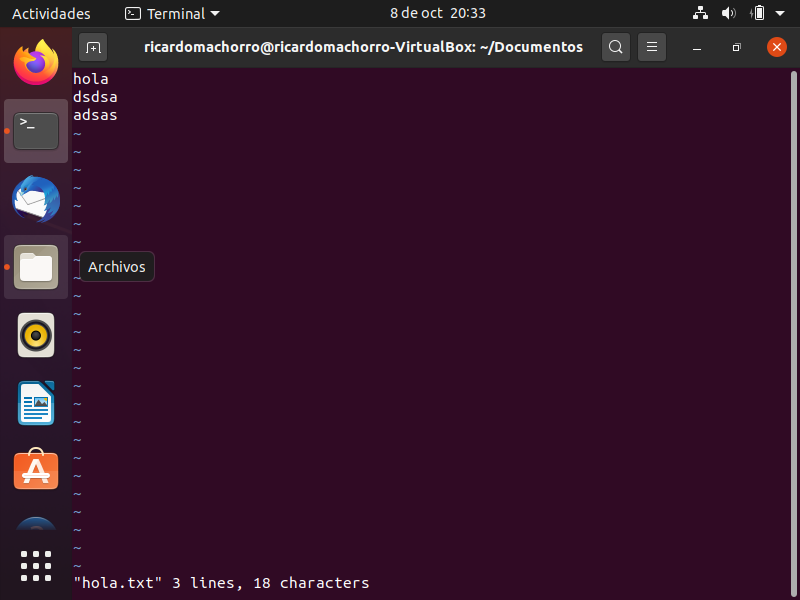
## **Comando cd**



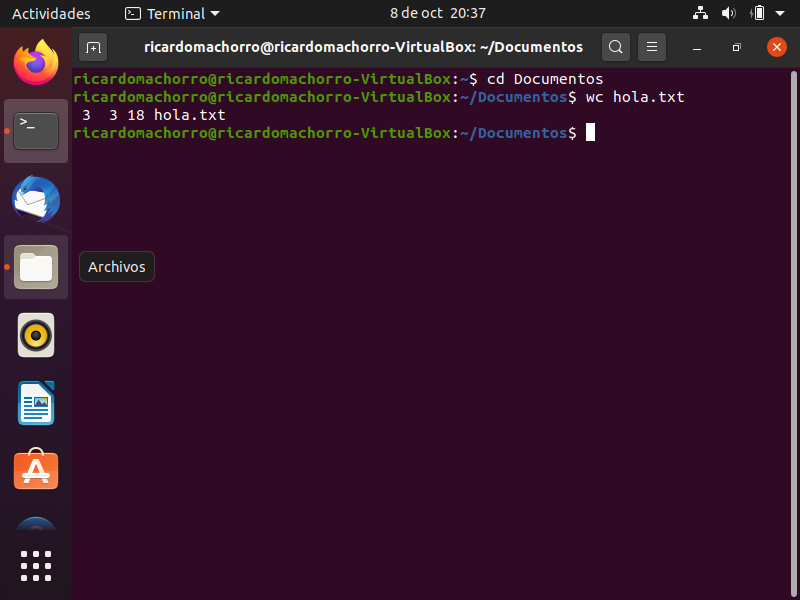
## **Comando vi antes de ejecutarse**



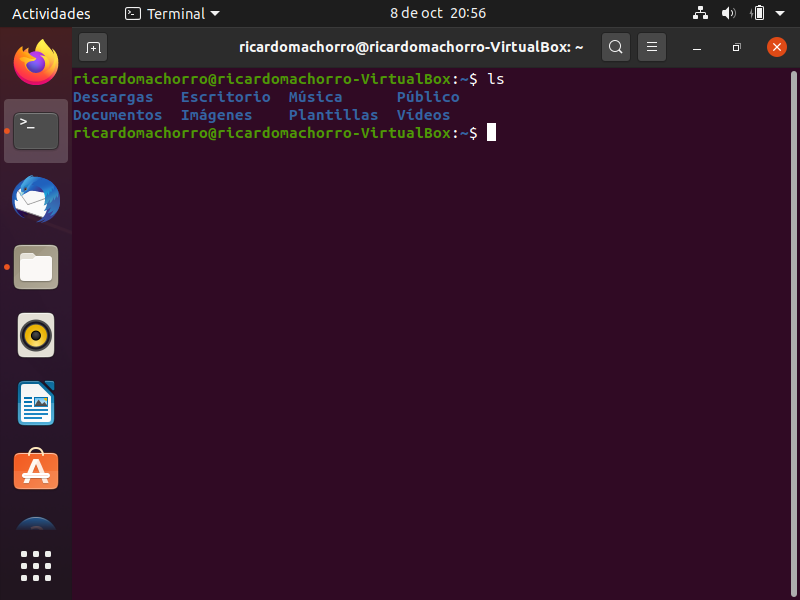
## **Comando vi luego de ejecutarse**



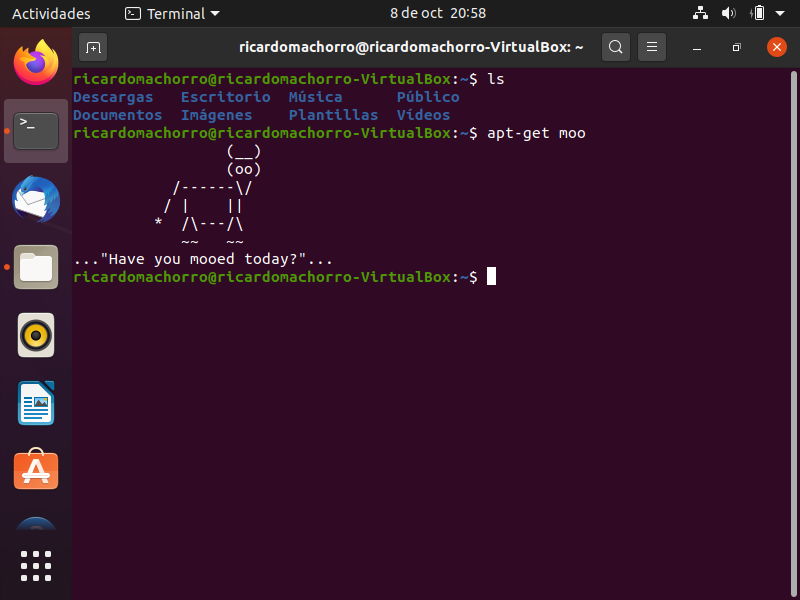
## **Comando wc**



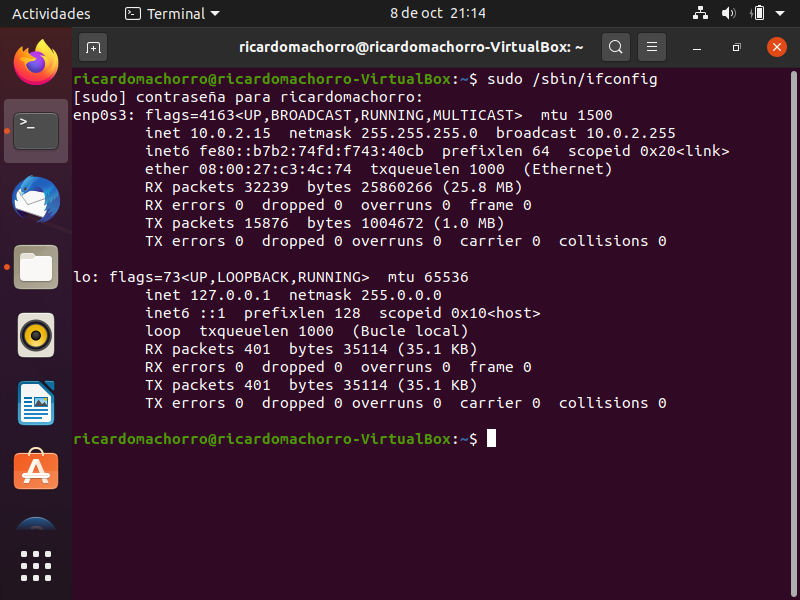
## **Comando ls**



## **Comando apt-get**

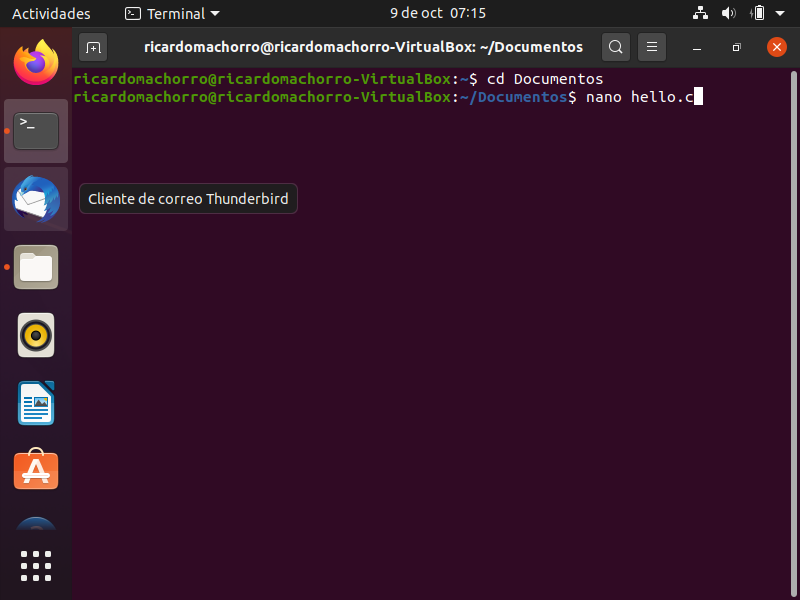


## **Comando sudo**

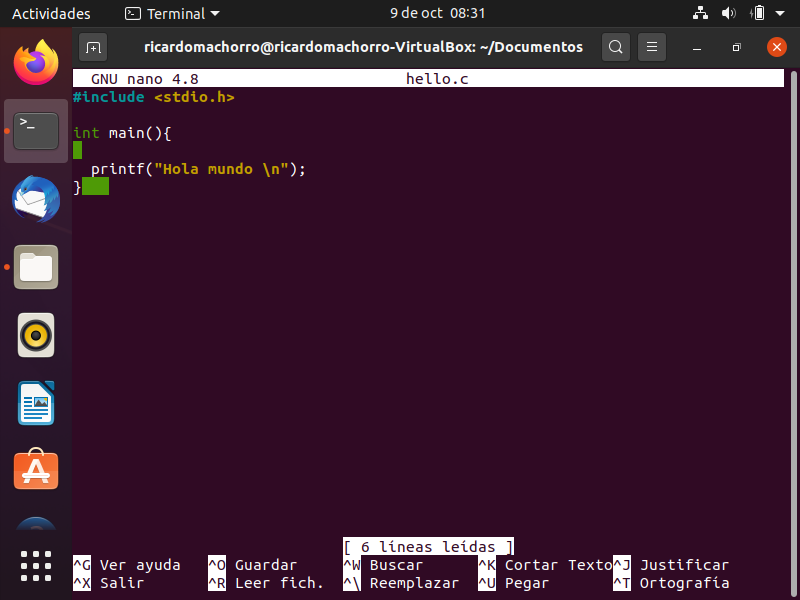


## **Compilar Hola Mundo en código para nano**

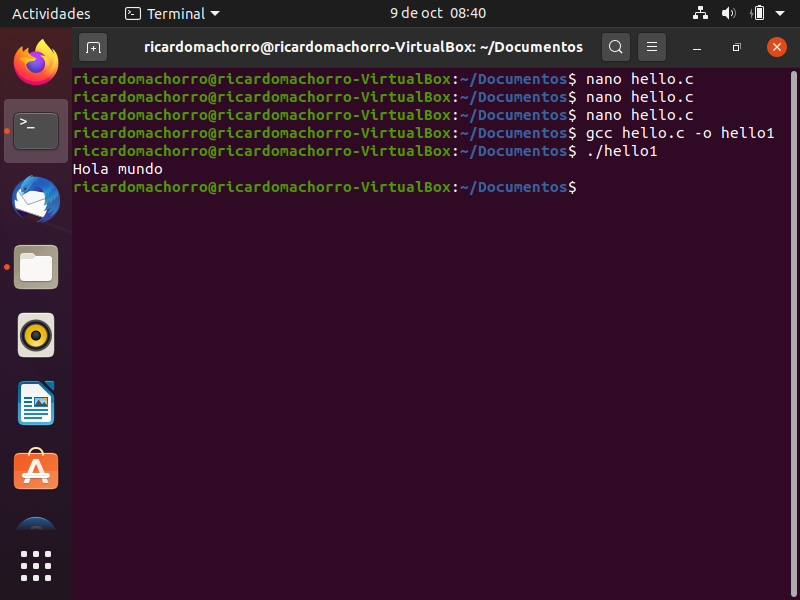
## **Creación Hola Mundo**



## **Código Hola Mundo editado en nano**

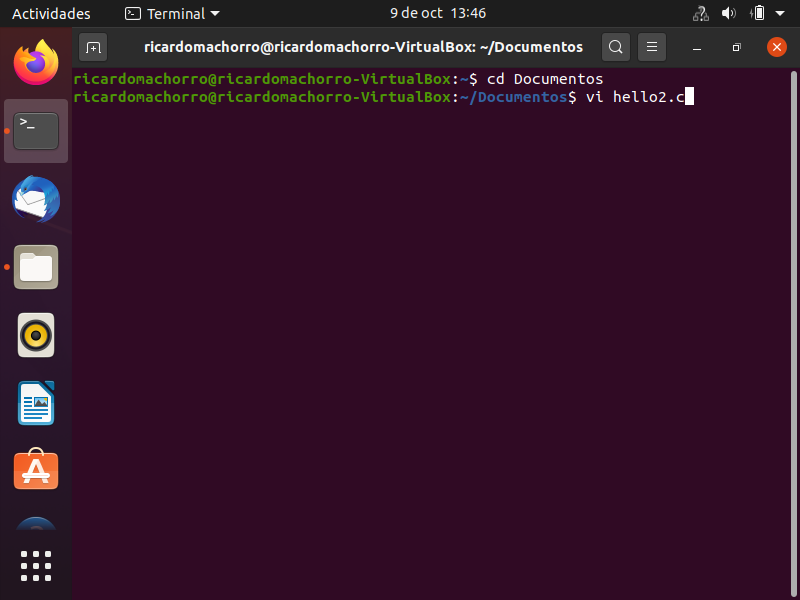


## **Código Hola Mundo compilación nano**

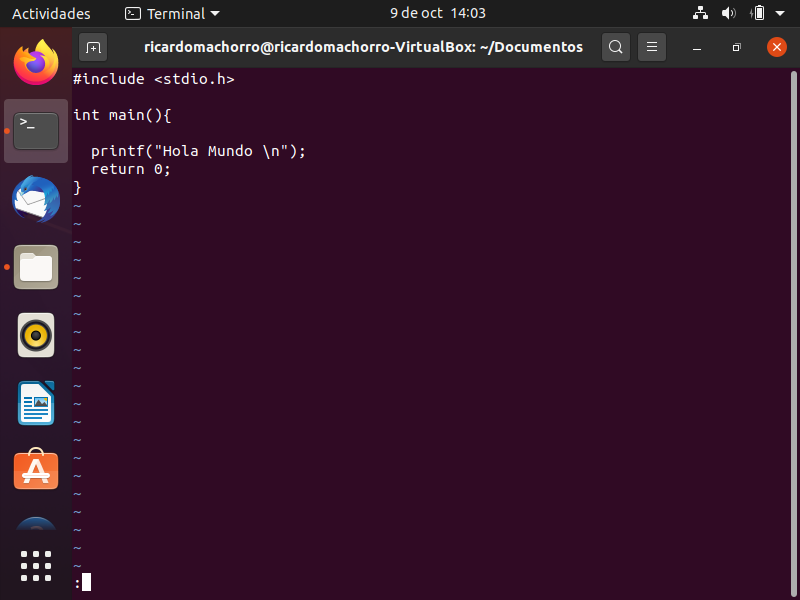


## **Compilar Hola Mundo en código para vi**

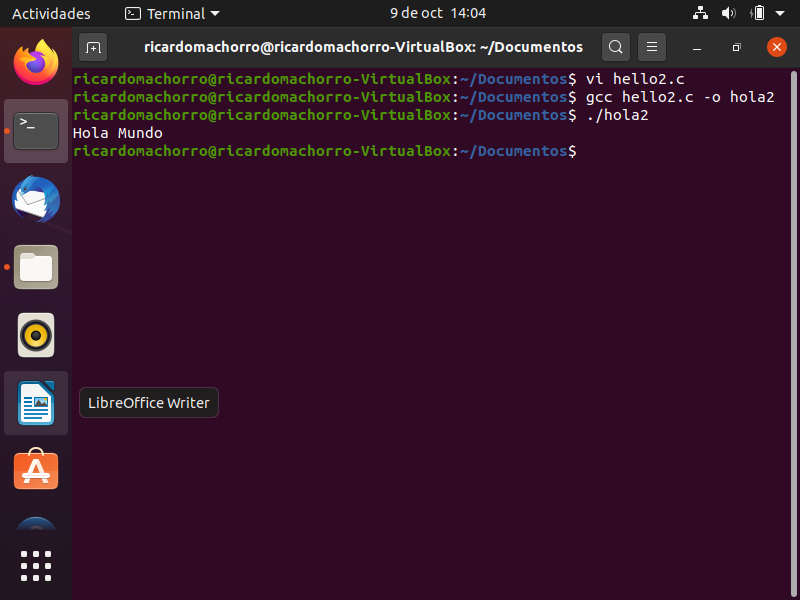
## **Creación Hola Mundo**



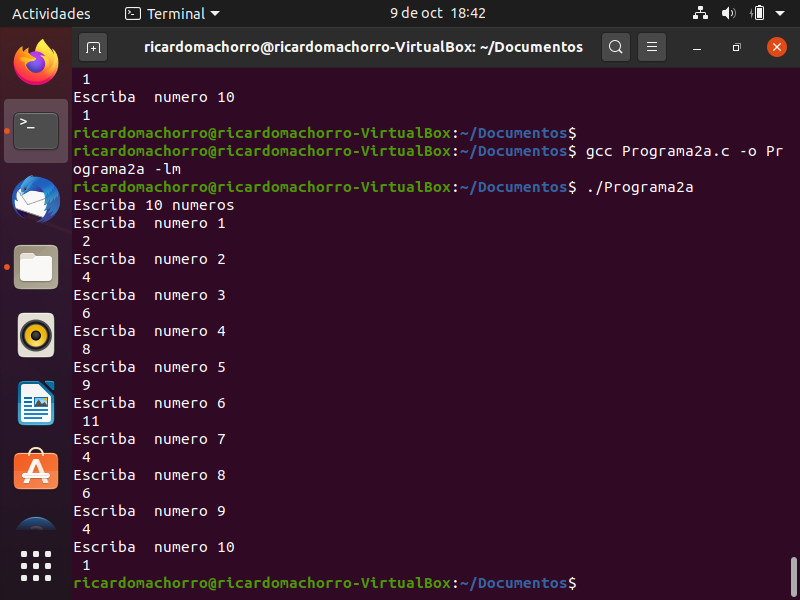
## **Edición Hola Mundo en vi**



## **Ejecución hola mundo en vi**



## **Compilación y pantalla de programa2a**





## **Compilación y pantalla de programa3a**



## Redireccionamiento utilizando ‘>’ (crea archivo, sobreescribe si existe)

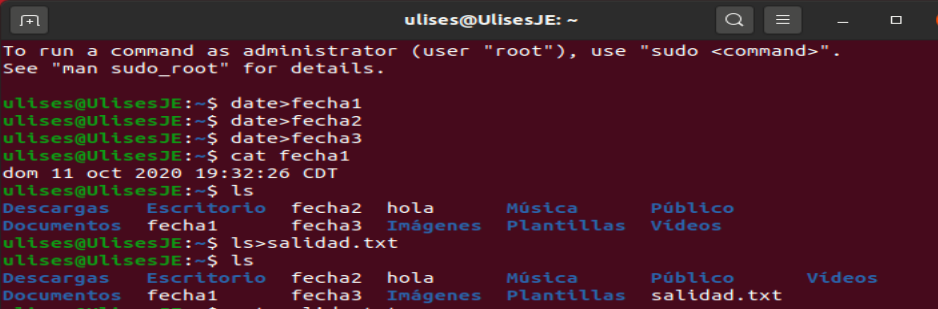
Ejemplo 1

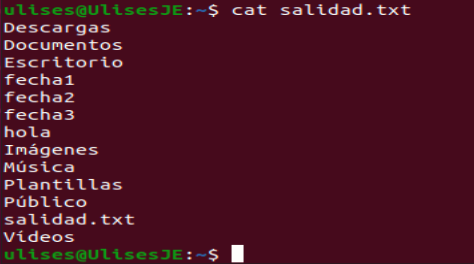
Primero creamos tres archivos donde guardamos la fecha, después con el comando cat mostramos el contenido de uno de estos archivos.

Con el comando ls mostramos los archivos que tenemos en el equipo, usamos ‘>’ para guardar el resultado del comando ls en un archivo de texto llamado salidad.txt

Después mostramos los archivos que tenemos en el equipo, y vemos que ya ha sido creado salidad.txt

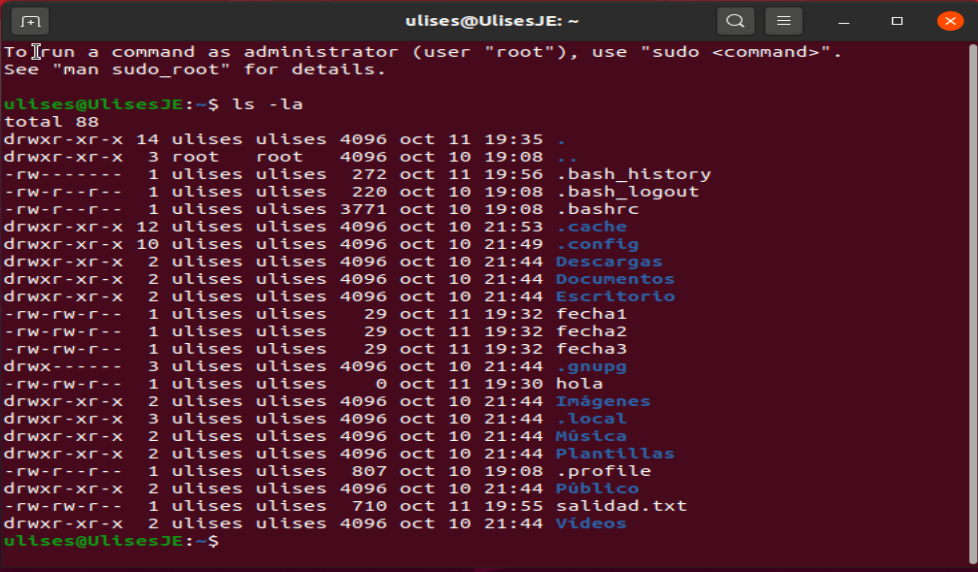
Por último con el comando cat mostramos lo que contiene este archivo, y notamos que se contiene a sí mismo, esto sucede porque ‘>’ crea los archivos antes de realizar el redireccionamiento, así que cuando realizo el redireccionamiento el archivo salidad.txt ya había sido creado y también se guardó.





Ejemplo 2

En este ejemplo, se ejecuta un comando, posteriormente se redirecciona el resultado de ese comando a un archivo llamdo valga la redundancia archivo.txt así que ya no se ve el resultado del comando en la terminal, sino que se guarda en ese archivo.



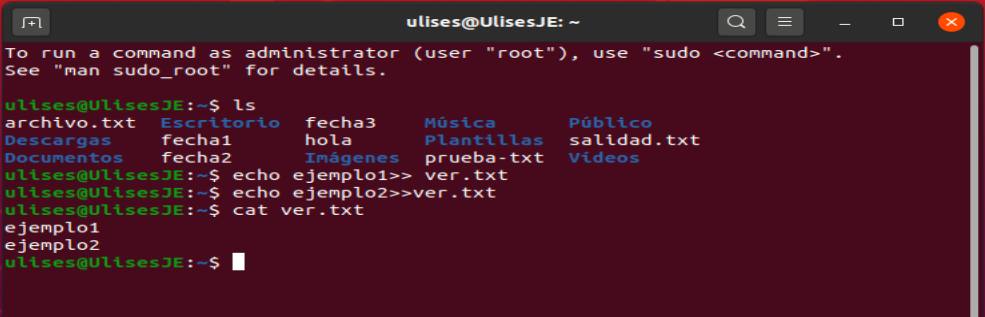
## 

## Redireccionamiento usando ‘>>’ (crea si no existe, agrega al archivo existente)

Ejemplo 1

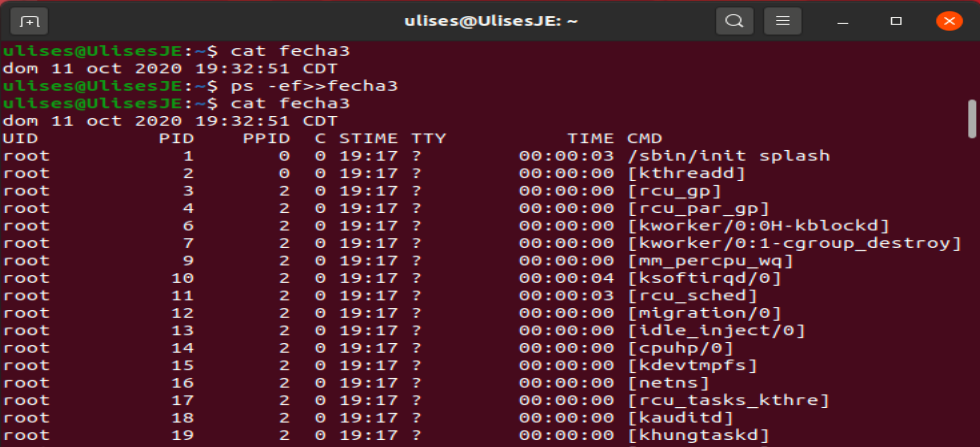
Se realiza una impresión de las palabras ejemplo1 y ejemplo2. Cuando se imprimió ejemplo 1 se creó el archivo ver.txt, del mismo modo con ejemplo2, sin embargo para redireccionar se usó ‘>>’ lo que permitió que ejemplo2 se agregara en el archivo y no se sobrescribiera.

Al momento de ver su contenido se puede observar que efectivamente ambas palabras están en el archivo.



Ejemplo 2

Tenemos un archivo llamado ‘fecha 3’ el cual contenía una instantánea de una fecha determinada, ejecutamos un comando y lo redireccionamos a ‘fecha 3’, al momento de ver que contiene el archivo ‘fecha 3’, se observa que efectivamente se añadió el resultado de ese comando y no se sobrescribió.

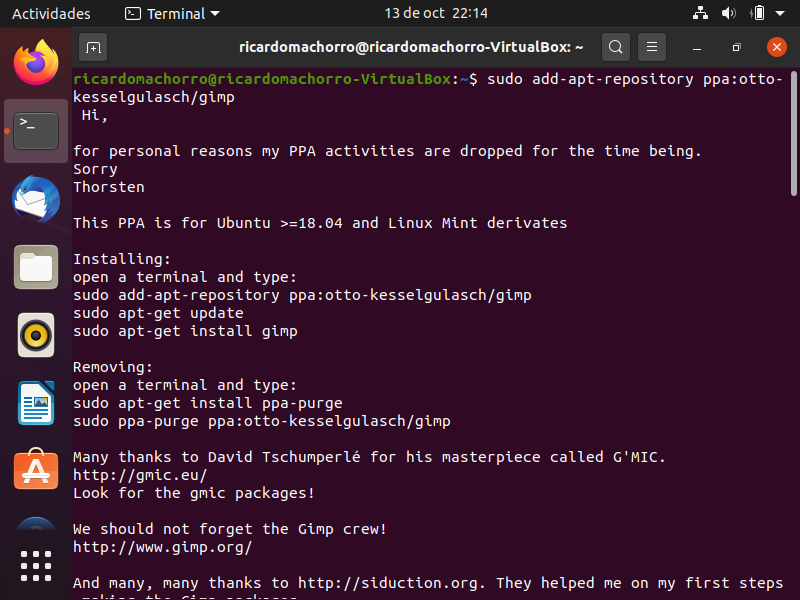


## Instalar en línea de comandos del paquete gimp

Se escribe los comandos:

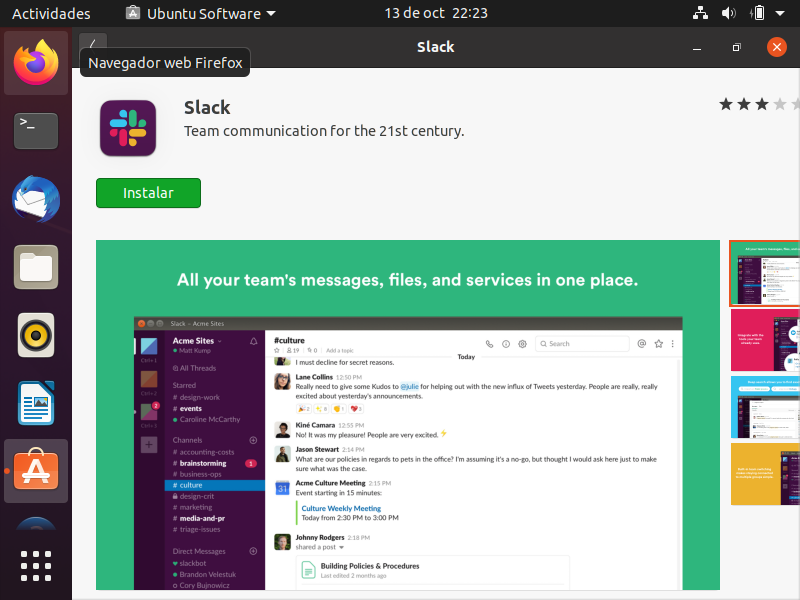
* sudo add-apt-repository ppa:otto-kesselgulasch/gimp
* sudo apt update
* sudo apt install gimp

para que así se descargue la versión oficial de gimp de su repositorio

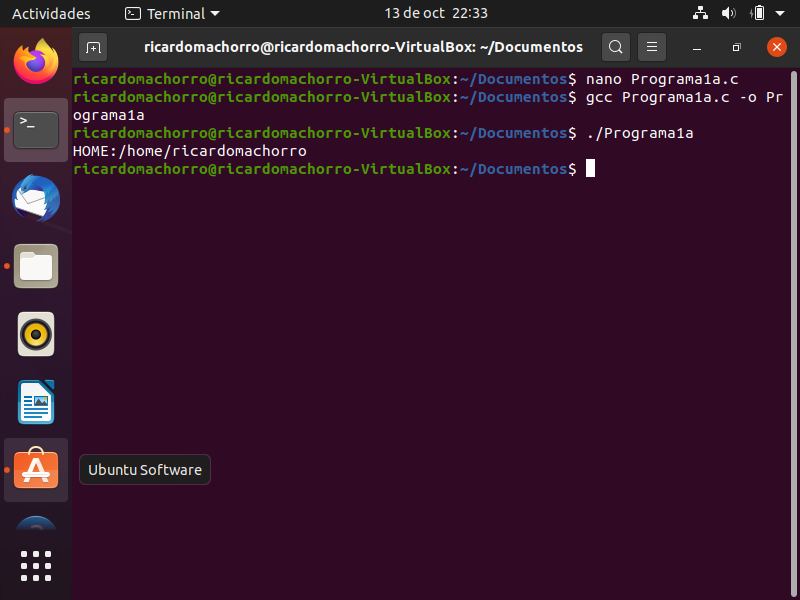




Instalar slack por el entrono gráfico



Resultado compilación Progrma1a.c



# **Conclusiones.**

## **Chavarría Vázquez Luis Enrique.**

## **Juárez Espinoza Ulises.**

## **Machorro Vences Ricardo Alberto.**

Esta práctica me enseño que primera instancia que Linux tienen muchas funciones que le hacen merecedor de su reputación de la caja de herramientas de los sistemas operativos, permitiendo así poder expandir no solo el conocimiento técnico del funcionamiento y control real de los sistemas operativos sino que también permite ayudar a familiarizarse con el entorno de la consola, aunque este grado de libertad también puede ser algo intimidador para el usuario, como de hecho fue mi caso al principio ya que yo había estado acostumbrado al seguro y hermético Windows.

Además, la práctica me ayudo en cierto grado a acostumbrarme y aprender a usar más la consola y no usar atajos como podría ser copiar el texto o usar comandos anteriores para ahorrar tiempo, que creo que es muy importante porque si no se hace de manera manual o propia no se aprende verdaderamente lo que hace inútil la práctica hasta cierto grado cosa que afecta más al alumno que al maestro.

Algo que también aprendí de esta práctica es que Linux pide muchos permisos incluso cuando se ejecutan comandos que relativamente son sencillos, o con códigos en lenguaje C que no afectan en realidad al sistema o interactúan con un elemento importante de este, porque en uno de los códigos necesitaba usar la librería de <math.h> para usar las operaciones matemáticas de potencias o raíces cuadradas y la línea de comandos de Linux me preguntaba ser administrador, cosa que me extraño.

La práctica además me ayudo a practicar mis habilidades de programación en C porque yo no había codificado en este lenguaje en casi un año, ya que me había concentrado en la parte web de la programación con lenguajes como Dart o Javascript y dentro de la parte más lógica me puse a aprender principalmente Python. Esto lo considero bueno porque muchos de los lenguajes mencionados anteriormente a diferencia de C tienen muchos métodos y estructuras de datos ya incluidas que facilitan acciones, pero que pueden hacer que se olvide como hacerlos.

En resumen, yo vi esta práctica como una forma no solo de poder expandir mi conocimiento en los temas que se solicitan en esta, sino que también ayudo a practicar habilidades y conocimientos de materias pasadas que por cuestiones de tiempo no he podido reforzar, y que además me ayudo a quitarme un poco el miedo que tengo a la consola y al sistema operativo y ver que en realidad este es una herramienta muy útil.

# **Bibliografía**

[1] KDE (2019, Julio,15). Que es KDE [Online]. Available: https://userbase.kde.org/What\_is\_KDE/es

[2] Ellen Seiver,Linux in a nutshell.sexta edicipon.Ciudad deMéixco: O’Reilly, 2009

[3] https://blog.desdelinux.net/mas-de-400-comandos-para-gnulinux-que-deberias-conocer/

[4] S. González D. (2020). Redireccionamiento en Linux [Online]. Available: https://www.linuxtotal.com.mx/index.php?cont=redireccionamiento-en-linux

[5] Deyimar A. (2019, Noviembre, 25). Variables de entorno de Linux: cómo leerlas y configurarlas en un VPS de Linux [Online]. Available: https://www.hostinger.mx/tutoriales/variables-de-entorno-linux-como-leerlas-y-configurarlas-vps/

[6] Mokhtar Ebrahim (2017, Febrero, 4). Aprenda Variables De Entorno De Linux Guía Paso A Paso [Online]. Available: https://likegeeks.com/es/variables-de-entorno-de-linux/