Comandos básicos de termux | RyuseiSanz

Comandos apt

apt funciona con una base de datos de paquetes disponibles. Si la base de datos no se actualiza, el sistema no sabrá si hay paquetes nuevos disponibles, es por ello que debemos ejecutar la actualización del repositorio en cualquier sistema Linux antes de realizar una nueva instalación, con estos comandos también podremos hacer innumerables cosas. Aquí te dejo una lista de ellas

- **Actualizar lista de paquetes instalados**

Este comando sirve para ver si hay algo nuevo en los repositorios

$Apt Update

- **Actualizar paquetes instalados**

Una vez haya sido actualizada la base de datos del paquete, el siguiente paso será actualizar los paquetes instalados. La forma más conveniente es actualizar todos los paquetes que tienen actualizaciones disponibles.

$apt upgrade

Aunque, si deseamos realizar una actualización completa podemos ejecutar el siguiente comando:

$*apt full-upgrade*

El método de actualización completa funciona igual que la actualización, excepto que si la actualización del sistema necesita la eliminación de un paquete ya instalado en el sistema, lo hará, mientras que el comando de actualización normal no lo ejecutará.

- **Instalando paquetes**

El comando apt nos ofrece una forma sencilla para la instalación de nuevos paquetes en el sistema.

$**apt install**

- El comando apt nos ofrece la opción de auto-completar, de esta manera, si no estamos seguros del nombre exacto del paquete, podremos escribir

algunas letras y presionar tab para desplegar sugerencias de todos los paquetes disponibles con esas letras.

Por ejemplo, apt red y veremos todos los paquetes disponibles con esa palabra

- En el caso de ser necesario instalar diversos paquetes en el sistema, apt nos ofrece la posibilidad de instalarlos en una sola línea y no uno por uno.
  $apt install < paquete1> < paquete2> <paquete3>
- **Eliminar paquetes usando apt**

Otra de las tareas frecuentes que realizamos en ambientes Linux es la de remover paquetes y aplicaciones que ya no usamos como tal.

$apt remove <nombre_paquete>

Otra forma de desinstalar paquetes en Linux es usando el parametro purge, el cual usaremos de la siguiente manera

$apt purge <nombre_paquete>

- **Búsqueda de paquetes**

En muchas ocasiones es posible que deseemos instalar un determinado paquete, pero no tengamos claro cuál es el nombre especifico de este.

El comando apt nos permite obtener un listado de los paquetes disponibles con un determinado termino

$apt search <termino>

- **Ver el contenido de un paquete usando apt**

Esta opción es practica cuando deseamos obtener información sobre el paquete tal como sus dependencias, el tamaño de instalación y descarga, las diferentes fuentes de las que está disponible el paquete, la descripción del contenido del paquete entre otras cosas.

$apt show <nombre_paquete>

- **Ver los paquetes instalados en el sistema**

Podremos ver todos los paquetes instalados en el sistema

$apt list –installed

- **Ver archivos de un paquete instalado**

Lista de archivos instalados de un paquete

$dpkg -L <package>

- **Ver los paquetes disponibles**

Enumera todos los paquetes disponibles.

$apt list

Comandos

- **Buscar los paquetes disponibles**

$apt search <query>

Comandos Para Crear, Leer o Editar Archivos de Texto desde la Terminal

En esta parte verás de forma muy rápida y directa como crear y leer archivos de texto desde la linea de comandos, incluir lineas de texto en un archivo, o imprimir su contenido por pantalla para poder visualizarlo de antemano.

Para ello, aprenderás a utilizar de forma comandos como touch, cat o more, y también a defenderte medianamente bien con Nano, un editor de archivos en modo texto, al igual que Gedit pero para ser usado desde la termina. Dicho esto, vamos allá!

1. Como Crear Archivos de Texto

Como en los casos anteriores, voy a mostrar un listado con algunos ejemplos de los comandos más utilizados a la hora de crear y leer archivos de texto. Para este caso, me centraremos principalmente en cuatro comandos, y a partir de sus múltiples opciones, vas a ir viendo todas las posibilidades que ofrecen.

- **Touch**

El comando touch te va a permitir, entre otras cosas, crear un archivo de texto vacío (en formato .txt) al que, posteriormente, podrás agregarle el contenido que desees de forma manual, o mediante el uso de otros comandos que veremos.

$ touch nombrearchivo

Esta opción te generará un archivo de texto vacío de nombre nombrearchivo, en el directorio en el que te encuentres (por defecto el directorio del usuario con el que estás logueado en la terminal)

$ touch nombrearchivo1 nombrearchivo2 nombrearchivo3

En este caso, el comando te generará tres archivos de texto vacíos, con los nombres nombrearchivo1, nombrearchivo2, y nombrearchivo3, y en el directorio de trabajo en el que te encuentres.

2. Imprimir el Contenido de un Archivo de Texto

- **Cat**

El comando cat es uno de los comandos más utilizados cuando se trata de manejar archivos de texto (en formato .txt) desde la terminal. Entre sus múltiples opciones, está la posibilidad de crear un archivo, imprimir por pantalla su contenido, etc. Veamos algunos ejemplos:

Este comando te creará un archivo de texto vacío, de nombre nombrearchivo, y te permitirá teclear el contenido que desees introducirle. Una vez tecleado el contenido, puedes finalizar mediante la combinación CTRL+D.

$ cat > nombrearchivo

contenido línea 1

contenido línea 2

CTRL+D

Esta orden te permitirá imprimir por pantalla todas las lineas de texto del archivo de texto de nombre nombrearchivo. El archivo indicado debe encontrarse en el directorio de trabajo actual.

$ cat nombrearchivo

Esto te imprimirá por pantalla, de forma conjunta, el contenido de los archivos indicados, en este caso nombrearchivo1 y nombrearchivo2.

$ cat nombrearchivo1 nombrearchivo2

Este comando te imprimirá por pantalla el contenido del archivo de texto de nombre nombrearchivo, mostrándote, además, el número de línea al principio de cada línea de texto.

$ cat -n nombrearchivo

Muy similar al anterior, con la diferencia de que a la hora de numerar las líneas de nombrearchivo, solo se numeran aquellas que contienen texto, y se descartan las lineas en blanco.

$ cat -b nombrearchivo

Hay muchas más opciones, pero creo que con estas pocas ya ves más o menos por dónde va la cosa. Como suelo decir, si quieres profundizar más en el uso del comando lo más recomendable es ir a la misma página de manual tecleando man cat en la misma terminal.

- **More**

El comando more es otro comando útil para imprimir por pantalla el contenido de un archivo de texto. Esencialmente es igual que el comando cat, con la diferencia de que el comando more pagina el contenido, y es más adecuado cuando para leer archivos largos. Veamos algunu ejemplo:

Esto te permitirá imprimir por pantalla el contenido del archivo ‘nombrearchivo’, pero con el resultado paginado. Así, primero se mostraran las líneas que quepan en una pantalla sin hacer scroll. El resto serán accesibles mediante la tecla espacio.

$ more nombrearchivo

- **Less**

El comando less, al igual que los comandos cat y more,te permitirá leer el contenido de un archivo de texto. A diferencia de los otros dos, éste te mostrará el contenido en modo editor de texto, y para moverte por el contenido deberás utilizar combinaciones de teclado.

Con esta opción podrás leer al contenido del archivo indicado, moviéndote por las distintas líneas a través del teclado.

$ less nombrearchivo

Una vez en modo texto, tienes varias opciones para desplazarte verticalmente a lo largo de las diferentes líneas.

Una opción es haciendo scroll con el ratón, que si utilizas la aplicación de la terminal que vienen con el entorno de escritorio podrás hacer sin problema.

Si no funciona esto, puedes utilizar los controles de desplazamiento del teclado. Al estar en modo lectura, deberían funcionar sin problema tanto si estás en el entorno de escritorio o en una sesión de terminal TTY.

Otra opción es utilizar la tecla g y luego Enter. Esto avanzará, por defecto, una línea adelante, pero puedes avanzar cualquier número de líneas que deseas, introduciendo el número justo después de marcar la g.

:1

Otra opción es utilizar la tecla espacio. Esto hará algo similar a un AvPag.

Finalmente, para salir del modo lectura y volver a la terminal, tan solo debes marcar la tecla q.

3. Leer y Editar Archivos de Texto con Nano

- **Nano**

Nano es un editor de textos para la terminal, que más que para leer archivos sirve para modificarlos y editarlos, aunque para esta guía también nos vale perfectamente para abrir el archivo y visualizar su contenido desde la línea de

comandos. A continuación tienes las principales opciones que ofrece, así que te animo a que vayas jugando con ellas para ir familiarizándote.

Primero de todo, veamos como abrir el editor de textos Nano, algo que puedes hacer fácilmente con el siguiente comando.

$ nano

Una vez abierto, si te fijas en la parte inferior, verás que se muestra las diferentes combinaciones de teclas que necesitaras a la hora de trabajar con archivos.

CTRL+R

Combinación de teclas para indicarle un archivo de texto a Nano para que lo abra y muestre su contenido por la consola.

CTRL+V

Estando dentro de Nano y con el archivo abierto en la consola, esta combinación sirve para avanzar a la página siguiente.

CTRL+Y

De modo similar a la combinación anterior, esta sirve para retroceder a la página anterior.

CTRL+W

Esta combinación te servirá para introducir un carácter o grupo de caracteres y buscar en el texto cualquier letra o palabra que coincida con el parámetro de búsqueda.

CTRL+X

Para cerrar el archivo una vez lo hayas terminado de visualizar en la consola.

Comando Chmod

El comando chmod se usa para cambiar los permisos de archivos o directorios. En Linux y otros sistemas operativos como Unix, hay un conjunto de reglas para cada archivo que define quién puede acceder a ese archivo, y cómo se puede acceder a él. Estas reglas se llaman permisos de archivo o modos de archivo. El nombre de comando chmod significa “modo de cambio” y se usa para definir la forma en que se puede acceder a un archivo.

Sintaxis del comando chmod

En general, los comandos chmod toman la forma:

chmod opciones permisos "nombre del archivo"

Si no se especifican opciones, chmod modifica los permisos del archivo especificado en “nombre de archivo” por los permisos especificados en “permisos”.

Permisos define los privilegios para el propietario del archivo (el “usuario”), los miembros del grupo que posee el archivo (el “grupo”) y cualquier otra persona (“otros”). Hay dos formas de representar estos permisos: con símbolos (caracteres alfanuméricos) o con números octales (los dígitos del 0 al 7).

Digamos que yo soy es el propietario de un archivo llamado “mi-archivo.txt” y quiero establecer los permisos de la siguiente forma:

1. El usuario puede leer, escribir y ejecutar el archivo.
2. Los miembros del grupo pueden leer y ejecutar el archivo.
3. Otros usuarios solo pueden leer el archivo.

El comando que tendríamos que usar seria:

chmod u = rwx, g = rx, o = r mi-archivo.txt

Este ejemplo usa notación de permisos simbólicos. Las letras u, g y o representan “usuario”, “grupo” y “otro”. El signo igual (” = “) significa “establecer los permisos exactamente así” y las letras “r”, “w” y “x” significan “leer”, “escribir” y “ejecutar”, respectivamente. Las comas separan las diferentes clases de permisos, y no hay espacios entre ellos.

La representación de este mismo comando chmod, pero en notación de octales seria:

chmod 754 mi-archivo.txt

Aquí los dígitos 7, 5 y 4 representan individualmente los permisos para el usuario, grupo y otros, en ese orden. Cada dígito es una combinación de los números 4, 2, 1 y 0:

- 4 significa “leer”,
- 2 significa “escribir”,
- 1 significa “ejecutar”, y
- 0 significa “sin permiso”.

Entonces 7 es la combinación de permisos 4 + 2 + 1 (leer, escribir y ejecutar), 5 es 4 + 0 + 1 (leer, no escribir y ejecutar), y 4 es 4 + 0 + 0 (leer, no escribir y no ejecutar).

Opciones del comando chmod

- **-c, –changes:** Como –verbose, pero da salida detallada solo cuando se realiza un cambio.
- **-f, –silent, –quiet**: Modo silencioso; se usa para suprimir la mayoría de los mensajes de error.
- **-v, –verbose**: Modo detallado; muestra un mensaje de diagnóstico por cada archivo procesado.
- **–no-preserve-root**: Se usa para definir que no trate ‘/ ‘(el directorio raíz) de ninguna manera especial, que es la configuración predeterminada.

- **–preserve-root**: Se usa para indicar que no opere recursivamente en ‘/ ‘.
- **–reference = RFILE**: Indica que debe establecer los permisos para que coincidan con los del archivo RFILE, ignorando cualquier MODO especificado.
- **-R, –recursivo**: Cambia archivos y directorios recursivamente.
- **–help**: Muestra un mensaje de ayuda y también funciona para salir.
- **–versión**: Da información referente a la versión que se está usando.

Visualización de permisos de archivos

Una manera rápida y fácil de enumerar los permisos de un archivo es con la opción de listado largo (-l) del comando ls. Por ejemplo, para ver los permisos de mi-archivo.txt, puedo usar el comando:

- ls -l mi-archivo.txt

Esto mostrará un resultado similar al siguiente:

- rwxrw-r-- 1 yerita02 ryuseisanz 123 Feb 25 15:36 mi-archivo.txt

Esto es lo que significa cada parte de esta información:

" – ": El primer carácter representa el tipo de archivo: "-" para un archivo normal, "d" para un directorio," l" para un enlace simbólico.

- **rwx**: Los siguientes tres caracteres representan los permisos para el propietario del archivo: en este caso, el propietario puede leer escribir y ejecutar el archivo.
- **rw-**: Los siguientes tres caracteres representan los permisos para los miembros del grupo de archivos. En este caso, cualquier miembro del grupo propietario del archivo puede leer y escribir en el fichero. El guion al final es un marcador de posición; los miembros del grupo no tienen permiso para ejecutar este archivo.
- **r–**: Los permisos para "otros" (todos los demás). Otros sólo pueden leer este archivo.
- **"1":** La cantidad de enlaces fijos a este archivo.
- **ryuseisanz**: El propietario del archivo
- **ryuseistaff:** El grupo al que pertenece el archivo.
- **123**: El tamaño del archivo en bloques.
- **Feb 25 15:36:** El mtime del archivo (fecha y hora de la última modificación del archivo).
- **mi-archivo.txt**: El nombre del archivo.

Ejemplos

- ***chmod 644 file.js***

Establece los permisos de file.js en "el propietario puede leer y escribir, el grupo solo puede leer y otros solo pueden leer".

- ***chmod -R 755 mis-archivos***

Recursivamente (-R) Cambia los permisos del directorio mis-archivos, y todas las carpetas y archivos que contiene, al modo 755 : el usuario puede leer, escribir y ejecutar; los miembros del grupo y otros usuarios pueden leer y ejecutar, pero no pueden escribir.

- ***chmod u = rw mi-foto.jpg***

Cambia los permisos para el propietario de mi-foto.jpg para que el propietario pueda leer y escribir el archivo. No cambia los permisos para el grupo o para otros.

- ***chmod 755 archivo.cgi***

Establece los permisos de archivo.cgi en “leer, escribir y ejecutar por el propietario” y “leer y ejecutar por el grupo y todos los demás”.

- ***chmod 666 archivo.txt***

Establece los permisos de archivo.txt para “leer y escribir por todos”.

- ***chmod a = rw archivo.txt***

Cumple lo mismo que el comando anterior, usando notación simbólica.

Comandos Basicos Simples

1. **Termux-setup-storage**

Este comando se utiliza para otorgarle permisos de almacenamiento a termux, tambien con esto crearemos algunos accesos directos.
$termux-setup-storage

2. **Cd (Change directory)**

Funciona para abrir carpetas
$cd example
- funciona para regresar a la carpeta anterior
$cd

3. **Pwd (imprimir directorio de trabajo)**

Funciona para ver en que parte de la estructura del escritorio te encuentras
$pwd

4. **Ls (listar)**

funciona para listar o visualizar el contenido de un directorio
$ls

- **ls -l**

Este comando funciona para listar el contenido de una carpeta y ver su fecha y permiso
$ls -l
El primer carácter de la línea indica el tipo de archivo. Un guión (-) es un archivo corriente; una d indica un directorio, y otros caracteres pueden indicar tipos de archivos especiales.
Los nueve caracteres siguientes indican los permisos del archivo o el directorio. Dichos caracteres están formados por tres grupos de tres elementos, que indican los permisos del propietario del archivo, del grupo de dicho propietario y del mundo respectivamente. Los permisos para emptyfile son rw-r--r--, que indican que el propietario del archivo puede leerlo y escribir en él, que todos pueden leerlo y nadie puede ejecutarlo. Los permisos del directorio veggies2 son rwxr-xr-x, que indican que todos tienen permiso para leerlo y ejecutarlo, pero sólo su propietario puede escribir en él.

Además de los permisos de archivo, la pantalla muestra la siguiente información:
-El número de enlaces con dicho archivo o directorio.
-El nombre del propietario (user2 en este caso).
-El número de bytes (caracteres) del archivo.
-Fecha y hora en la que el archivo o el directorio fue actualizado la última vez.
-Nombre del archivo o directorio.

5. **";" (Punto y coma)**

Funciona para darle una orden adicional a termux.
$cd github/tools/**;**ls (aqui le estamos diciendo que nos lleve hacia la ruta que anteriormente especificamos y liste los directorios que hay ahí, en pocas palabras le di las ordenes "ve a tools y muestrame su contenido")

6. **Cd .. (Dos puntos)**

Funciona para retroceder carpetas
$cd ../../../usr/etx/;ls (aqui le estoy diciendo que me retroceda 3 carpetas (retrocede 1 por cada ../ , luego le dijimos que nos abriera la carpeta usr, luego

la carpeta etc y por ultimo que nos listara el contenido de etc

7. **Rm (remove)**

funciona para borrar archivos especificos, si ponemos la ruta especifica del archivo podemos borrarlo desde donde quiera que estemos
$rm example.txt

$rm /data/data/com.termux/files/usr/etc/motd2;ls /data/data/com.termux/files/usr/etc (aqui ordenamos borrar el archivo motd2 y le adicionamos que nos listara los archivos de la ruta especifica)

- rm -r: este comando funciona para borrar directorios y todo su contenido, no podemos utilizar el comando "rm" para borrar directorios, solo para eliminar archivos, asi que este comando lo utilizaremos especificamente para eliminar directorios.
  $rm -r example

8. **Cp (copy)**

Comando para copiar archivos
$cp example
$cp /data/data/com.termux/files /data/data/com.termux/files2;ls /data/data/com.termux/ (Aqui ordenamos que nos copiara la carpeta "files" a una ruta especifica pero con otro nombre (lo copiamos a la misma ruta donde esta ubicado el archivo) ya que no pueden existir 2 archivos con el mismo nombre, ademas adicionamos que nos listara los archivos de dicha carpeta

9. **Git clone**

este comando funciona para clonar repositorios de la pagina github, estos archivos que clonaremos con el pasar del tiempo son muy importantes segun la herramienta o paquete que queramos instalar, por eso en muchos paquetes es necesario instalar complementos o aveces hasta el mismo paquete
git clone https://github.com/example/example-example.git
$chmod: El comando chmod se usa para cambiar los permisos de archivos o directorios. existen un conjunto de reglas para cada archivo que define quién puede acceder a ese archivo, y cómo se puede acceder a él. Estas reglas se llaman permisos de archivo o modos de archivo .este comando significa “modo de cambio” y se usa para definir la forma en que se puede acceder a un archivo.

$chmod 777 archive.py
(Para saber mas sobre este archivo (es complicado) te dejo el documento que he creado: https://icutit.ca/87Hx

10. Whoami (¿Quién soy yo)

Este comando muestra el nombre del usario que ejecuta el comando
$whoami

11. Mv (mover)

este comando se utiliza para mover archivos o directorios.
$mv archivo /data/data/com.termux/files/home
(Donde "mv archivo" especifica cual archivo moveremos, y /data/data/com.termux/files/home vendria a ser la ruta donde lo moveremos).
-si queremos mover el archivo a cualquier ubicación (si, hasta en la misma del archivo) simplemente debemos hacer esto:
$mv archivo archivo_ryu
Aquí lo que hicimos fue mover el archivo a la misma ubicación (ya que al no especificar una ruta, la terminal lo mueve hacia la ruta en la que actualmente se encuentra) pero con otro nombre.

12. Su (Superususario)

Este comando nos concede permisos de supe usuario
&su

13. Clear

Clear (de limpiar), es un sencillo comando que limpiara nuestra terminal por completo dejándola como recién abierta.

$clear

Si te gusto el tutorial apóyame dando gracias xd

By RyuseiSanz