Visita nuestro sitio SoloLinux.es

MAGAZINE
SOLO LINUX

Nº 13

Tu revista, la revista de tod@s

FEBRERO 2020

Uso del **comando dmesg** en Linux

Instalar **sudo** en Android

Solución a la vulnerabilidad de **sudo 2020**

Como usar el **comando w** con ejemplos

Uso y ejemplos del **comando who**

Qué son los **procesos en Linux** y cómo identificarlos

MANUALES, SCRIPTS, SOFTWARE, HARDWARE, DISTROS LINUX, SEGURIDAD, REDES Y MUCHO MAS EN LA WEB...

- Numero 2. Proximamente
- Últimas novedades
- Trucos
- Consejos útiles
- Manuales paso a paso
- Debates abiertos
- Opiniones de expertos
- Artículos

¡Si crees que puedes ayudar contacta con nosotros!

SoloWordpress

Tu revista sobre Wordpress

Si el formato digital no te convence, también tenemos todo el contenido en una Página Web

¡Visítanos!

www.solowordpress.es

Bienvenido a la Revista SOLOLINUX

Os presento el numero 13 de la revista SoLoLinux. Comenzamos con nuestro segundo año editorial. Desde SOLOLINUX esperamos que os guste este numero.

Igual que en números anteriores nos gustaría animar a todos nuestros lectores para que nos envíen sus **opiniones sobre el Software Libre o sobre GNU/Linux**, pueden enviarlo a adrian@sololinux.es, con ello queremos proponer que cada mes se publicada una o varias de esas opiniones sobre lo mencionado en la nueva sección de la revista *OPINIÓN DEL LECTOR*. **Queremos saber la opinión de todos**. Se intentara incluir el máximo de opiniones en cada numero, pero si no sale la tuya este mes no desesperes, al siguiente podría tener un hueco en la revista. **ANÍMENSE Y ENVÍEN SUS OPINIONES.** Gracias.

Hablar es barato enseñame el código

Al igual que lo anteriormente mencionado, nos gustaría promover un espacio en la revista sobre los eventos de Software Libre y GNU/Linux en todo el mundo. Los organizadores de estos eventos pueden ponerse en contacto con migo a través de correo electrónico, adrian@sololinux.es

Sin mas **quiero agradecer a todos** los que hacéis posible que esta revista siga adelante.
Personalmente agradezco a Sergio todo su trabajo en la multitud de artículos que realiza a lo largo del mes para que esta revista pueda tener suficiente información mes a mes.

Gracias a TOD@S

Compartan esta revista en sus redes sociales o web.
Revista digital **SOLOLINUX MAGAZINE**. Tu revista, la revista de todos.

Adrián A. A.

Esta revista es de **distribución gratuita**, si lo consideras oportuno puedes ponerle precio.
Tu también puedes ayudar, contamos con la posibilidad de hacer donaciones para la REVISTA, de manera muy simple a través de **PAYPAL**

AYUDANOS A SEGUIR CRECIENDO

PayPal™ Apoyo con una donación

Síguenos en las Redes:

La revista SOLOLINUX esta realizada con Libre Office Impress 6.2.8.

Nuestras Webs:

www.sololinux.es
www.solowordpress.es

SoloLinux

EDITORIAL

Edición:
- Adrián Almenar
adrian@sololinux.es

Redacción:
- Sergio G. B. (Administrador y redactor artículos SoloLinux)
info@sololinux.es

- Henry G. R (Redactor artículos SoloWordPress)
info@solowordpress.es

Agradecimientos:

Contacto:
Para cualquier consulta sobre las revistas, publicidad o colaboraciones escribir un email a:
adrian@sololinux.es

MANUALES

DESARROLLOS WEB

SEGURIDAD

HARDWARE

DISTROS LINUX

SCRIPTS

THANKS!

TU PUBLICIDAD AQUÍ

QUIERES APARECER EN LA REVISTA, GANAR CON ELLO MAS VENTAS EN TU WEB, MAS SEGUIDORES EN TUS REDES SOCIALES...

SOLO TIENES QUE MANDAR UN CORREO A adrian@sololinux.es

Y TE EXPLICAMOS COMO

INSTITUTO LINUX

[UNIVERSIDAD TECNOLOGICA NACIONAL] FRD [FACULTAD REGIONAL DELTA]

Linux Professional Institute APPROVED TRAINING PARTNER

THE LINUX FOUNDATION AUTHORIZED TRAINING PARTNER

100% ON LINE

CURSO LINUX SYSTEM ADMINISTRATOR

IMPERDIBLE

U$S 97

VALOR U$S 297

TUTOR
FABIÁN AMPALIO

+54 9 11 6969 9993

SEGUINOS EN Instagram @fabianampalio

@exameneslinux @fabianampalio www.linkedin.com/in/Fabian-Ampalio

Reparar el sistema de archivos con fsck

Seguro que alguna vez te has encontrado con errores en un sistema de archivos. Los sistemas de archivos son una jerarquía de directorios, que determinan como se guardan y organizan los archivos en el sistema.

Cada sistema de archivos tiene asignado un conjunto de reglas, que controlan como y cuando se concede un espacio de disco a los archivos. A veces, puede ocurrir que los sistemas de archivos se corrompan y provoquen fallos aleatorios en linux, aplicaciones con errores extraños, reinicios frecuentes, etc.

Reparar el sistema de archivos con fsck

La herramienta fsck (**verificación de consistencia del sistema de archivos**) es muy buena, pero a no ser un caso de extrema necesidad no la uses en una partición montada. Existe una alta posibilidad de que tu sistema de archivos se dañe gravemente.

```
sergio@sololinux ~ $ fsck -A
fsck de util-linux 2.27.1
e2fsck 1.42.13 (17-May-2015)
/dev/sda1 está montado.


¡¡ATENCIÓN!!  El sistema de ficheros está montado. Si se continúa se PROVOCARÁN
GRAVES daños al sistema de ficheros.

¿De verdad quiere continuar?<n>?                    www.sololinux.es
```

Identificar particiones

Lo más practico y recomendable es iniciar tu maquina desde un USB live. Una vez inicia linux live identificas las particiones con el siguiente comando…

```
sudo fdisk -l
```

Reparar una partición

En nuestro ejemplo queremos reparar la **partición sdb1**.

Primero desmontamos la partición, y después la verificamos y reparamos con **fsck** (las opciones **-a** / **-y** reparan los errores automáticamente)

```
umount /dev/sdb1

sudo fsck -a /dev/sdb1

o

sudo fsck -y /dev/sdb1
```

Forzar la verificación en particiones montadas

Aunque no es recomendable, a veces es necesario escanear un dispositivo montado. Para evitar que la aplicación verifique si la partición esta montada o no, usamos la opción **-M**.

```
sudo fsck -M /dev/sdb1
```

Especificar el sistema de archivos

También podemos especificar el tipo de sistema de archivos que queremos verificar. En el ejemplo ext4 con la opción **-t**.

```
fsck -t ext4 /dev/sdb1
```

Verificar y reparar todos los sistemas de archivos a la vez

Si tienes varios sistemas de archivos los puedes verificar todos a la vez con **-A**.

```
fsck -A
```

Evitamos que escanee el sistema de archivos raíz y reparamos.

```
fsck -AR -y
```

Que fsck se ejecute al iniciar el sistema

Algunas distribuciones linux ejecutan la herramienta cada 30 arranques, sobre todo las derivadas de **Ubuntu**. Si no es tu caso, o simplemente quieres modificar el valor ejecuta el comando «**tune2fs**».

En el ejemplo cada 20 arranques:

```
tune2fs -c 20 /dev/sdb1
```

En otras ocasiones, tal vez resulte más funcional ejecutar «**tune2fs**» cada X días.

En el ejemplo cada 6 días:

```
sudo tune2fs -i 6d /dev/sdb1
```

Otra alternativa interesante es, **ejecutar fsck** cada vez que inicie el sistema. Para lograr esta operación debemos crear un archivo (vació) llamando «**forcefsck**», e insertarlo en la raíz.

```
sudo touch /forcefsck
```

Nota final

Como norma general no tienes que preocuparte por el sistema de archivos; Linux es lo suficientemente inteligente como para asegurarse que todo funciona bien. Aun así, si por algún caso tu sistema se corrompe, en este articulo hemos visto las formas más sencillas de solucionar el problema.

Generador de passwords con selector de seguridad

Recién salido del horno os presento este interesante **script bash**, que nos genera automáticamente una contraseña con el tipo de seguridad (fuerza) que necesites.

El script te ofrece la opción de seis **niveles de seguridad**:

- **Super extrema**
- **Extrema**
- **Muy fuerte**
- **Fuerte**
- **Normal**
- **Sencilla**

El script es bastante simple, no tendrás ninguna dificultad para crear el tuyo propio o en ampliar el que yo te propongo.

Generador de passwords con selector de seguridad

Detalles y ejemplos de las opciones

Vemos los comandos usados en el script para generar las passwords, también algunos ejemplos de salida.

Super extrema

```
cat /dev/urandom | tr -dc [:print:] | tr -d
'[:space:]\042\047\134' | fold -w 48 | head -n 1
```

Ejemplo de salida…
PASS-SUPER EXTREMA:
]MA^jC<u7Mc[B%TI(}kkys^y0[Kx}+qN4>g[L3$`&%8GMOqD

Extrema

```
openssl rand -base64 32
```

Ejemplo de salida…
PASS-EXTREMA:
nNURVmqkLw4pzhHta3PYHWkMXJcbQT74d0+N7Ubh3xE=

Muy fuerte

```
dd if=/dev/urandom bs=1 count=32 2>/dev/null |
base64 -w 0 | rev | cut -b 2- | rev
```

Ejemplo de salida…
PASS-MUY FUERTE:
I3HoF6Ukc1sziSWPaup5oGM1rpD06om0PVlUfgrxZTY

Fuerte

```
date | sha256sum | base64 | head -c 32 ; echo
```

Ejemplo de salida…
PASS-FUERTE:
NzgzOWRlNjgzNTU4OGFjZTUxNjEzZjEy

Normal

```
date | md5sum
```

Ejemplo de salida…
PASS-NORMAL:
6f9c3f971b087b22141c3be6d60a1bfd

Sencilla

```
openssl rand -hex 5
```

Ejemplo de salida…
PASS-SENCILLA:
6959a9cc94

Script generador de passwords

Creamos el script con nuestro editor favorito.

```
nano passGenerator.sh
```

Copia y pega lo siguiente:

```
#!/bin/bash
# scripts/passGenerate.sh
# author: SergioG.B.-SoloLinux.es

echo -e "Selecciona el tipo de seguridad: "
echo -e "1) Super extrema "
echo -e "2) Extrema "
echo -e "3) Muy fuerte "
echo -e "4) Fuerte "
echo -e "5) Normal "
echo -e "6) Sencilla "
echo -e "q) Salir "
read -p "> " choice

if [ "$choice" = "1" ]; then
        echo "PASS-SUPER EXTREMA: "; cat /dev/urandom
| tr -dc [:print:] | tr -d '[:space:]\042\047\134' |
fold -w 48 | head -n 1
elif [ "$choice" = "2" ]; then
        echo "PASS-EXTREMA: "; openssl rand -base64
32
elif [ "$choice" = "3" ]; then
        echo "PASS-MUY FUERTE: "; dd if=/dev/urandom
bs=1 count=32 2>/dev/null | base64 -w 0 | rev | cut
-b 2- | rev
elif [ "$choice" = "4" ]; then
        echo "PASS-FUERTE: "; date | sha256sum |
base64 | head -c 32 ; echo
elif [ "$choice" = "5" ]; then
        echo "PASS-NORMAL: "; date | md5sum
elif [ "$choice" = "6" ]; then
        echo "PASS-SENCILLA: "; openssl rand -hex 5
elif [ "$choice" = "q" ]; then
        echo -e "Script cerrado"; exit 0;
else
        echo "Error - se cierra el script.."
fi
```

Guarda el archivo y cierra el editor.

Ahora le concedemos los permisos de ejecución necesarios.

```
sudo chmod +x passGenerate.sh
```

Lo ejecutamos con alguno de los comandos propuestos.

```
./passGenerate.sh
```

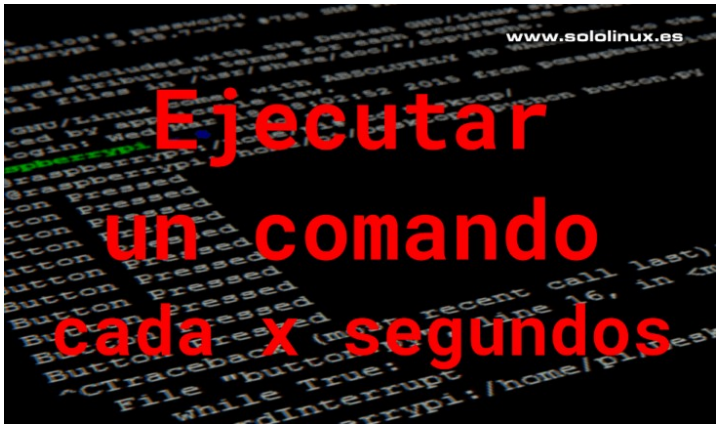
```
bash passGenerate.sh
```

Listo!!!.

Síguenos en las Redes:




Como ejecutar un comando cada x segundos con watch



Con las **tareas cron** podemos programar la ejecución de cualquier comando o script en un tiempo definido, pero no siempre es necesario usar **cron**, existe una alternativa más fácil de utilizar si es para simples comandos o pequeños **scripts**, el **comando watch**.

Debemos recordar que cron no maneja tiempos inferiores a un minuto, algo que el **comando watch** ejecuta si ningún problema. Por defecto **watch** está predefinido a una repetición cada dos segundos, pero tranquilo podemos modificar ese valor fácilmente.

Como ejecutar un comando cada x segundos con watch
El **comando watch** puede ejecutar cualquier orden, comando, o script que tu le indiques. Su manejo es sencillo, observa su sintaxis.

```
watch [-dhvt] [-n <seconds>] [--differences[=cumulative]] [--help] [--interval=<seconds>] [--no-title] [--version] <command>
```

Aunque puede funcionar como comando más orden, también tiene unas opciones interesantes.

Opción corta	Opción extendida	Uso
-n	--interval	Define el tiempo entre ejecución y ejecución
-d	--diferences	Marca la diferencia entre una ejecución y otra
-b	--beep	Emite un beep del sistema si se producen errores
-p	--precise	Refresca la pantalla con cada ejecución
-e	--errexit	Cierra la orden si se produce algún error
-c	--color	Interpreta los códigos de escape con colores ANSI
-x	--exec	Ejecutar el comando con exec
-t	--no-title	Que no se muestre la frecuencia de refresco ni la fecha actual

A modo de ejemplo usamos el comando free de forma simple.

```
watch free
```

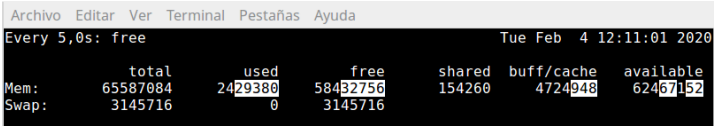
```
Archivo  Editar  Ver  Terminal  Pestañas  Ayuda
Every 2,0s: free                                    Tue Feb  4 10:21:15 2020

               total        used        free      shared  buff/cache   available
Mem:        65587084    17056512    43627136      155944     4903436    47838336
Swap:        3145716           0     3145716
```

Si quieres que se ejecute cada 15 segundos:

```
watch -n 15 free
```

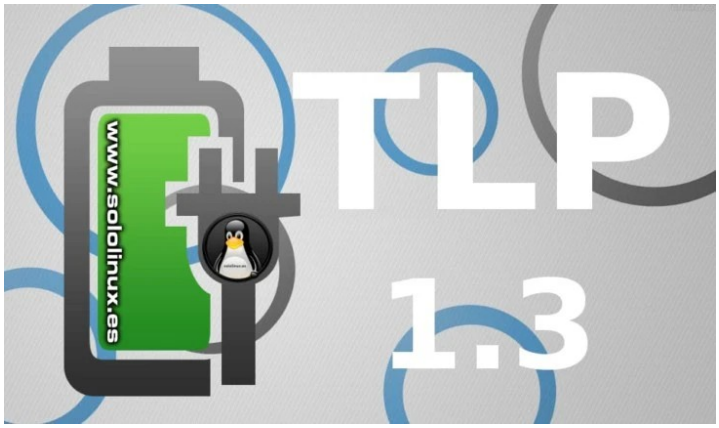
En el siguiente ejemplo definimos que se ejecute cada 5 segundes y que nos indique los cambios.

```
watch -n 5 -d free
```



Como hemos visto en el articulo, watch puede ser muy útil en algunos momentos. La mayoría de aplicaciones de monitoreo de servidores hacen uso de watch.

Ahorrar batería con TLP en linux



Después de ocho meses de desarrollo, por fin se lanza el nuevo **TLP 1.3**. La herramienta avanzada de **administración de energía para Linux**, viene con un nuevo esquema de configuración, mejoras de tlp-stat, y una solución para los portátiles que informan erróneamente del estado de la batería o la CA.

TLP viene con una configuración predeterminada ya optimizada para ahorrar batería, tan solo debes instalar la aplicación y ella hará su trabajo sin intervención del usuario. A pesar de lo dicho es altamente personalizable, puedes revisar su archivo de configuración (/etc/tlp.conf), y modificar sus valores. En el articulo de hoy vemos como instalar **TLP** en cualquier distribución linux.

Ahorrar batería con TLP en linux

Ahora vemos como podemos instalar TLP en cualquier linux, si eres usuario de un pc portátil notaras la diferencia.

Instalar TLP en Ubuntu, Linux Mint, y derivados

```
sudo add-apt-repository ppa:linrunner/tlp

sudo apt update

sudo apt install tlp tlp-rdw
```

Instalar TLP en Debian

Para versiones anteriores de TLP ejecuta…

```
apt install tlp tlp-rdw
```

Si quieres (recomendado) tener TLP 1.3, debemos agregar el deb que corresponda a tu versión de Debian.

```
nano /etc/apt/sources.list
```

Copia y pega el que corresponda.

```
# Debian 10 Buster
deb http://ftp.debian.org/debian buster-backports
main

# Debian 9 Strech
deb http://ftp.debian.org/debian stretch-backports-
sloppy main
```

Guarda el archivo y cierra el editor.

Actualizamos.

```
sudo apt update
```

La instalación también cambia dependiendo de tu Debian.

```
# Debian 10 Buster
apt -t buster-backports install tlp tlp-rdw

# Debian 9 Strech
apt -t stretch-backports-sloppy install tlp tlp-rdw
```

Instalar TLP en Arch, Manjaro, y derivados

```
pacman -S tlp tlp-rdw
```

Habilitamos el servicio.

```
systemctl enable tlp.service

systemctl enable NetworkManager-dispatcher.service
```

Para evitar conflictos se recomienda enmascarar los servicios.

```
systemctl mask systemd-rfkill.service

systemctl mask systemd-rfkill.socket
```

Instalar TLP en Fedora y derivados

```
dnf install tlp tlp-rdw
```

Instalar TLP en CentOS, RHEL, y derivados

```
yum install tlp tlp-rdw
```

Iniciamos la herramienta.

```
tlp start
```

Instalar TLP en Open Suse, Suse, y derivados

```
zypper install tlp tlp-rdw
```

Iniciamos la aplicación.

```
tlp start
```

Una vez concluida la instalación, TLP ya está en marcha, puedes verificar la configuración actual con el siguiente comando:

```
sudo tlp-stat -c
```

Ejemplo…

```
sololinux # sudo tlp-stat -c
— TLP 1.3.0 ——————————
+++ Configured Settings:
defaults.conf L0004: TLP_ENABLE=»1″
defaults.conf L0005: TLP_PERSISTENT_DEFAULT=»0″
defaults.conf L0006: DISK_IDLE_SECS_ON_AC=»0″
defaults.conf L0007: DISK_IDLE_SECS_ON_BAT=»2″
defaults.conf L0008: MAX_LOST_WORK_SECS_ON_AC=»15″
defaults.conf L0009: MAX_LOST_WORK_SECS_ON_BAT=»60″
defaults.conf                                 L0010:
CPU_ENERGY_PERF_POLICY_ON_AC=»balance_performance»
defaults.conf                                 L0011:
CPU_ENERGY_PERF_POLICY_ON_BAT=»power»
defaults.conf L0012: SCHED_POWERSAVE_ON_AC=»0″
defaults.conf L0013: SCHED_POWERSAVE_ON_BAT=»1″
defaults.conf L0014: NMI_WATCHDOG=»0″
```

Alerta de batería baja: script bash

Después del ultimo articulo sobre **TLP**, hoy vemos un **script bash** que con la ayuda del **comando watch**, o con una **tarea cron**, nos lanzara una alerta indicándonos que tenemos la batería baja.

Para que el script funcione correctamente necesitamos tres herramientas; Normalmente vienen preinstaladas, si no es así lo tendrás que hacer manualmente.

- **acpi**: Este comando nos dice el porcentaje de batería cargada, si se está cargando o descargando, y el tiempo restante que falta para que se descargue la batería por completo. Ejemplo: **[acpi –b]**–> Battery 0: Discharging, 15%, 00:43:36 remaining
- **espeak**: El comando espeak es muy curioso, también desconocido. Podemos usarlo para introducir un texto en la terminal que se reproducirá en los altavoces de nuestro pc, por ejemplo: [espeak «visita sololinux»](entre comillas dobles). Si no lo tienes, lo instalas. Ejemplo: sudo apt install speak.
- **notify-send**: Con notify-send lanzamos ventanas emergentes que notifican lo ingresado. Ejemplo: [notify-send «Visila SoloLinux.es»](entre comillas dobles).

También haremos uso de las siguientes herramientas:

- **cut**: Permite extraer la sección indicada de un archivo de texto antes de su salida.
- **sed**: Usamos sed para reemplazar caracteres.
- **grep**: Nos busca cadenas en un archivo.

Alerta de batería baja: script bash
Creamos el script.

```
nano alerta-bateria.sh
```

Copia y pega el script (puedes modificar lo que quieras.

```
#!/bin/bash
#
 battery_level=$(acpi -b |cut -d ", " -f2| sed
's/%//g')

 echo $battery_level #$battery_level=15

#Si el cargador esta conectado se muestra "charging"
#Si el cargador no esta conectado se muestra
"discharging".
#Si acpi -b detecta carga, "grep -c" devuelve 1.
#Si acpi -b no detecta carga, "grep -c" devuelve 0.
 ac_power=$(acpi -b|grep -c "Charging")
 echo $ac_power #1 if charging(plugged in) and 0 if
discharging (not plugged in)

#La bateria que carga llega al 100%.
 if [[ $ac_power -eq 1 && $battery_level -eq 100 ]]
#if charging and battery_level==100
then
 export DISPLAY=:0.0
 notify-send "La batería esta cargada" "Carga:
$battery_level% ";

#Un audio nos indica que la bateria está cargada.
 espeak "bateria cargada" -s 140
 fi

#Mensaje emergente y por audio, nos queda el 20%
 if [[ $ac_power -eq 0 && $battery_level -lt 20 ]]
then
 export DISPLAY=:0.0
 notify-send "Queda poca batería" "Carga:
$battery_level% ";
 espeak "Batería baja" -s 140

fi
```

Guarda el archivo y cierra el editor.

Le damos permisos.

```
chmod +x alerta-bateria.sh
```

Lo podemos ejecutar manualmente, pero lo recomendable es crear una tarea cron que se ejecute cada cierto tiempo, en nuestro ejemplo cada 30 minutos.

```
nano /etc/crontab
```

Agregamos la tarea.

```
THIS_IS_CRON=1
 */30 * * * * /ruta/alerta-bateria.sh
```

Guarda el archivo y cierra el editor.

Ya tenemos todo listo, solo nos falta reiniciar cron.

```
service cron restart

o

service crond restart
```

Cómo hacer un gif animado con Gimp

Gif (**Graphics Interchange Format**), es un formato de archivo diseñado para aligerar el peso de las imágenes. Dada su buena compresión y alta versatibilidad, actualmente su uso más común es para crear **imágenes animadas** sencillas para sitios web.

La imágenes animadas con formato gif, son como si fueran pequeñas películas especialmente adaptadas para la **World Wide Web**. En este articulo, a petición de una lectora de **sololinux.es** vemos como crear un gif animado en gimp.

En este articulo usamos **Gimp 2.8.22**, pues considero que la versión 3.x es excesivamente pesada, y para la labor que yo realizo es suficiente. El proceso es exactamente el mismo en otras versiones como **Gimp 2.10**, o superiores.

Cómo hacer un gif animado con Gimp
En nuestro ejemplo vamos a usar un fondo base, y el **logo de sololinux** que ira alternando entre la versión negra y la blanca.

Nosotros nos aseguramos que los dos logotipos tienen exactamente el mismo tamaño y posición, si son diferentes imágenes dependerá del resultado que quieras.

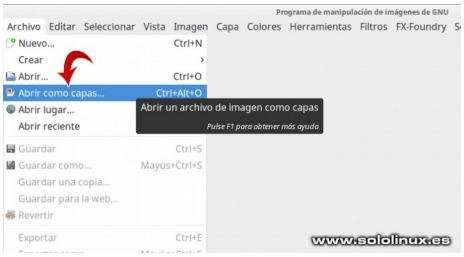
Ahora creamos y guardamos una imagen con la plantilla base.

Continuamos insertando el **logo sololinux** de color blanco en la plantilla base. Guarda la imagen con otro nombre.

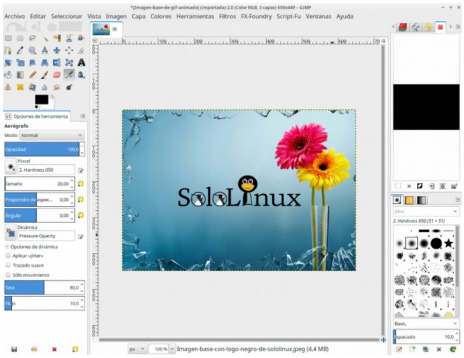
Para concluir hacemos la misma operación con el logotipo negro.

Bien… en este momento ya tenemos las tres imágenes creadas así que vamos a generar el **gif animado**. Es importante que no tengas ninguna imagen, ni capa abierta en **Gimp**, si no estas seguro cierra la aplicación y la abres de nuevo.

Para crear el gif animado pulsamos en archivo y en abrir como capas. Selecciona las imágenes que guardamos anteriormente.



Nos aparece algo similar a la siguiente imagen.



Hacemos click en **Archivo y Exportar como…/**. En la parte inferior derecha de la ventana que se acaba de abrir tienes un desplegable, en el podemos marcar la extensión de archivo que más nos interese. Para gif animado seleccionamos «**Imagen GIF (*.gif)**».

Nos aparecen las opciones. Asegúrate de que estén marcadas las opciones marcadas con una flecha, en retraso entre cuadros indica el tiempo entre imagen e imagen.

Pulsamos en exportar. El resultado final es…

https://www.sololinux.es/wp-content/uploads/2020/02/SoloLinux.gif

Cómo copiar un archivo a varias carpetas o directorios

Una de las tareas más comunes es copiar archivos a una carpeta o directorio. A estas alturas, más o menos todos sabemos como hacerlo; usamos el **comando cp**.

```
cp imagen.png /destino/

cp imagen1.png imagen2.png /destino/
```

Pero que ocurre si queremos copiar un archivo a varios directorios?, pues absolutamente nada porque el **comando cp** no permite esa operación. En el articulo de hoy vemos otras alternativas al comando cp que nos permiten copiar un archivo a varias carpetas o directorios.

Cómo copiar un archivo a varias carpetas
Vemos varias formas de lograr nuestro propósito, comenzamos con xargs que tal vez sea el más engorroso, y continuaremos con otros comandos más simples.

Copiar un archivo con xargs
Si utilizas ***xargs*** debes tener en cuenta que hay que incluir opciones y el comando cp.

- **-n 1** : indica a xargs que debe usar un argumento por línea de comando, y volver al comando cp.
- **cp** : comando cp.
- **-v** : si quieres el modo detallado (opcional)

Vemos un ejemplo de uso:

```
echo directorio1 directorio2 directorio3 |
xargs -n 1 cp archivo.txt
```

Copiar un archivo con find
Otra excelente alternativa que nos permite copiar un archivo a varios directorios, es el **comando find**.

```
find directorio1 directorio2 -exec cp
archivo.txt {} \;
```

Si el directorio de destino contiene subdirectorios, debes asegurarte que el archivo no se copie en ellos, para ello usamos «-maxdepth 0».

```
find directorio1 directorio2 -maxdepth 0 -exec
cp archivo.txt {} \;
```

Copiar un archivo con loop en shell
Otra solución interesante para usuarios avanzados es con loop (bucle) en shell.

```
for dir in *; do [ -d “$dir” ] && cp
/full_path/imagen.png “$dir” ; done
```

La ejecución anterior copiará el archivo **/full_path/imagen.png** en todos los directorios de la ruta indicada, o en la ubicación actual.

Copiar un archivo con GNU parallel
GNU parallel es una herramienta shell (poco conocida), que ejecuta trabajos en paralelo en uno o varios **sistemas unix**.

Entre los múltiples usos del comando, uno de ellos es dividir la entrada y canalizarla como comandos en paralelo. Dicho esto podemos utilizar **parallel** para multiplicar el **comando cp**.

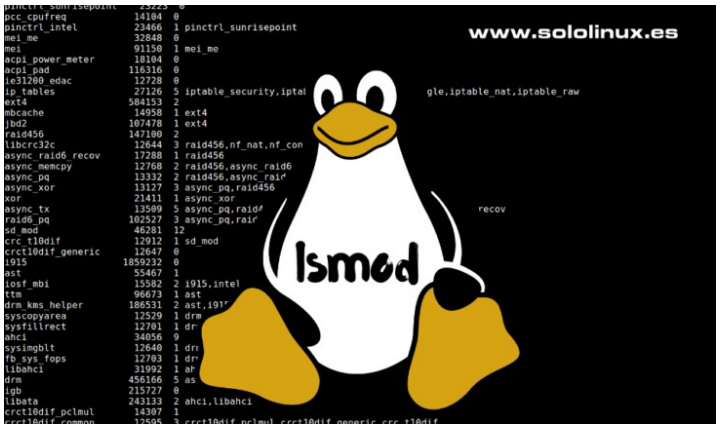
```
parallel cp -v /ruta/archivo.txt :::
/directorio1/, /directorio2/, /directorio3/
```

Copiar un archivo con tee
El **comando tee** también nos permite copiar un archivo a múltiples destinos. Vemos un ejemplo.

```
tee ~/directorio1/archivo.txt
~/directorio2/archivo.txt < ~/archivo.txt
```

Síguenos en las Redes:

Comando lsmod en linux



El **comando lsmod** es una herramienta en línea de comandos que muestra información sobre los módulos del **kernel Linux** que tenemos cargados.

El kernel de Linux tiene un diseño modular. Cada módulo del núcleo es un fragmento de código que amplia las funciones del núcleo, que como norma general se compilan como módulos cargables (o se integran).

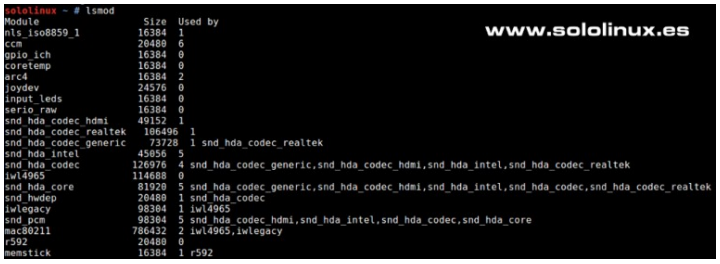
Los módulos que son cargables se pueden descargar e integrar en el kernel de manera simple, ademas sin el requisito posterior de tener que reiniciar el sistema. En este articulo conocemos la herramienta capaz de imprimir los módulos de nuestro kernel.

Comando lsmod en linux
La **herramienta lsmod** no admite opciones, su única misión es leer el contenido de «**/proc/modules**» e imprimir un listado formateado en pantalla.

Las tres columnas que genera el comando, son:
- **Module**: nombre del modulo.
- **Size**: tamaño del módulo en bytes.
- **Used by**: arroja un valor numérico que indica cuántas instancias del módulo se están usando. utilizan actualmente. Si el valor es cero el módulo no se está usando. Lo siguiente al numero indica lo que está utilizando el módulo.

Vemos un ejemplo del comando lsmod.

```
lsmod
```



También podemos especificar el modulo con **grep**, en el ejemplo el modulo sdhci.

```
lsmod | grep sdhci
```

Salida…

```
sololinux ~ $ lsmod | grep sdhci
sdhci_pci      32768  0
sdhci          49152  1  sdhci_pci
```

Canales de Telegram:
- Canal SoloLinux
- Canal SoloWordpress

SoloLinux

Esta revista es de **distribución gratuita**, si lo consideras oportuno puedes ponerle precio. Tu también puedes ayudar, contamos con la posibilidad de hacer donaciones para la REVISTA, de manera muy simple a través de **PAYPAL**

AYUDANOS A SEGUIR CRECIENDO

PayPal™ Apoyo con una donación

Solución a la vulnerabilidad de sudo 2020

En estos últimos, días he observado mucho revuelo respecto a la primera gran vulnerabilidad descubierta en **sudo** en este año **2020**.

En contra de lo informado en otros sitios web (no actualizan las noticias), en los cuales se decía que solo afectaba a versiones anteriores de la 1.8.26; debes saber que recientemente se ha descubierto que el problema no se queda en esa versión, sino que se amplia hasta sudo 1.8.30.

Por suerte el problema no afecta a linux de forma predeterminada, solo si habilitas el **pwfeedback**. La indicación **pwfeedback** obliga a imprimir asteriscos al insertar nuestra contraseña sudo, tal como indicamos en este articulo.

Dejando aparte el porqué del problema, el contratiempo deriva en que **pwfeedback** no se cierra de manera automática cuando no se esta utilizando, con la consecuencia fatal de estar expuestos continuamente. En este articulo vemos como solucionar el problema si es tu caso.

Solución a la vulnerabilidad de sudo 2020
Seamos claros, realmente necesitas ver asteriscos en la **password sudo**?, yo creo que no, tan solo es un efecto visual totalmente innecesario.

Lo primero que hacemos es comprobar si tenemos habilitado «**pwfeedback**».

```
sudo -l
```

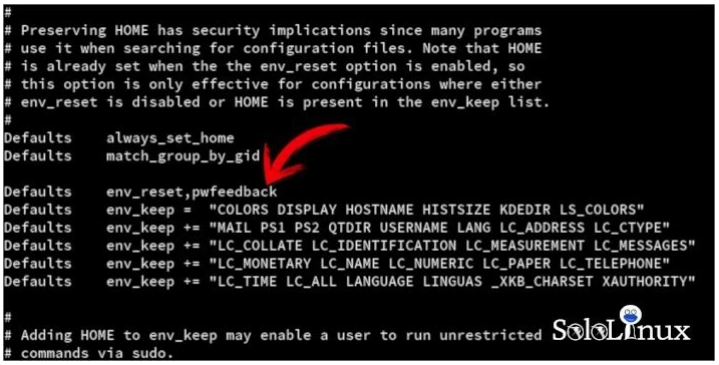
Ejemplo…

```
$ sudo -l
    Coincidiendo entradas por defecto para
sergio en sololinux:
       insults, pwfeedback, mail_badpass,
mailerpath=/usr/sbin/sendmail

    El usuario sergio puede ejecutar los
siguientes comandos en sololinux:
       (ALL : ALL) ALL
```

Como verificamos en el ejemplo, nosotros si lo tenemos habilitado. Para solucionarlo ejecuta el siguiente comando.

```
sudo visudo
```

Vemos algo similar a lo indicado en la siguiente imagen.



La solución es tan simple como borrar la linea indicada en la anterior imagen, o insertar un signo de exclamación delante.

```
Defaults !pwfeedback
```

Una vez borrada o editada la indicación, guarda el archivo y cierra el editor. Sudo no necesita ser reiniciado para tomar la nueva configuración, cada vez que lo llamas re-lee el archivo. A partir de la versión sudo 1.8.31 el problema está corregido.

La vulnerabilidad de sudo a sido solucionada.

Uso del comando dmesg en linux

El **comando dmesg** (diagnostic message / mensajes de diagnóstico), es una herramienta exclusiva de **sistemas basados en Unix** con la capacidad de listar los avisos temporales generados por el **kernel**.

Estos avisos se guardan en lo que se conoce como **buffer**, y contiene los mensajes más importantes que se generaron al iniciar el sistema (también los cambios durante su funcionamiento), por ejemplo de los **drivers y el hardware**. Estos (normalmente) se guardan mediante «**syslog**», siendo **dmesg** la herramienta con capacidad de leer e imprimir (en formato humano) los avisos del kernel.

Poder examinar los mensajes de arranque del núcleo, te ayudaran a solucionar **problemas relacionados con el hardware**. En este articulo vemos los conceptos principales que debes conocer de «**dmesg**».

Uso del comando dmesg en linux

Uso básico de dmesg

La sintaxis de dmesg es muy simple.

```
dmesg [OPCIONES]
```

No es obligatorio que apliques opciones, lo puedes ejecutar tal cual.

```
dmesg
```

Normalmente esta herramienta se ejecuta sobre cualquier usuario, si por un extraño caso no te permite su uso en usuarios sin permisos, recibirás un mensaje similar a:

```
dmesg: read kernel buffer failed: Operation
not permitted
```

El error anterior se produce por el ajuste del parámetro «kernel.dmesg_restrict». Solucionamos el problemas ajustando los privilegios.

```
sudo sysctl -w kernel.dmesg_restrict=0
```

La salida impresa del comando puedes ser muy larga, y esto dificulta su lectura. Para ver el archivo linea por linea…

```
dmesg --color=always | more
```

La herramienta dmesg nos permite filtrar los mensajes por dispositivo. En nuestro caso y a modo de ejemplo, extraemos un pendrive usb y un raton usb, pasados unos segundos los insertamos de nuevo.

```
dmesg | grep -i usb
```

Ejemplo…

```
sololinux ~ # dmesg | grep -i usb
[ 5265.936151]  usb 2-2: USB disconnect, device number 3
[ 5292.948110]  usb 3-1: new low-speed USB device number 2 using
uhci_hcd
[ 5293.144136]  usb 3-1: New USB device found, idVendor=046d,
idProduct=c077
[ 5293.144142]  usb 3-1: New USB device strings: Mfr=1, Product=2,
SerialNumber=0
[ 5293.144146]  usb 3-1: Product: USB Optical Mouse
[ 5293.144149]  usb 3-1: Manufacturer: Logitech
[ 5293.159907]  input: Logitech USB Optical Mouse as
/devices/pci0000:00/0000:00:1d.1/usb3/3-1/3-
1:1.0/0003:046D:C077.0004/input/input14
[ 5293.160911]  hid-generic 0003:046D:C077.0004: input,hidraw2:
USB HID v1.11 Mouse [Logitech USB Optical Mouse] on usb-
0000:00:1d.1-1/input0
[ 5300.036089]  usb 1-2: new high-speed USB device number 5 using
ehci-pci
[ 5300.198040]  usb 1-2: New USB device found, idVendor=13fe,
idProduct=4200
[ 5300.198046]  usb 1-2: New USB device strings: Mfr=1, Product=2,
SerialNumber=3
[ 5300.198050]  usb 1-2: Product: USB DISK 2.0
[ 5300.198054]  usb 1-2: SerialNumber: C900589889F91731
[ 5300.322756]  usb-storage 1-2:1.0: USB Mass Storage device
detected
[ 5300.333790]  scsi host2: usb-storage 1-2:1.0
[ 5300.334384] usbcore: registered new interface driver usb-storage
[ 5300.353236]  usbcore: registered new interface driver uas
[ 5301.342373]  scsi 2:0:0:0: Direct-Access      USB DISK 2.0
PMAP PQ: 0 ANSI: 6
```

Formatear la salida de dmesg

Dmesg nos permite aplicar opciones de formateo para que visualmente sea más comprensible. La más utilizada es la salida para humanos, opción -H.

```
dmesg -H
```

Para imprimir los tiempos…

```
dmesg -T
```

Ejemplo…

```
[lun feb 10 08:06:14 2020] usb 1-2: new high-speed USB device
number 5 using ehci-pci
[lun feb 10 08:06:14 2020] usb 1-2: New USB device found,
idVendor=13fe, idProduct=4200
[lun feb 10 08:06:14 2020] usb 1-2: New USB device strings: Mfr=1,
Product=2, SerialNumber=3
```

Las marcas de tiempo admiten varios formatos al hacer uso de «–time-format». Se admiten los siguientes:

- ctime
- reltime
- notime
- delta
- iso

En el ejemplo aplicamos «iso».

```
dmesg --time-format=iso
```

Salida en iso…

```
2020-02-10T08:05:39,936151+0200 usb 2-2: USB disconnect, device number 3
2020-02-10T08:06:06,948110+0200 usb 3-1: new low-speed USB device number 2 using uhci_hcd
2020-02-10T08:06:07,144136+0200 usb 3-1: New USB device found, idVendor=046d, idProduct=c077
```

Comando dmesg en tiempo real.

```
dmesg -w
```

Aplicando varias opciones a la vez.

```
dmesg -H -T
```

Filtrar la salida de dmesg

Se permite filtrar los mensajes por quien los genero, en este caso las opciones son las siguientes:

- **user** – mensajes a nivel de usuario
- **kern** – mensajes del kernel
- **daemon** – demonios del sistema
- **auth** – mensajes de seguridad y permisos
- **mail** – sistema de correos
- **lpr** – impresoras en línea
- **news** – novedades de la red
- **syslog** – avisos internos de syslogd

Para lograr el objetivo utilizamos «-f», que ademas nos permite especificar varios filtros (si es necesario).

```
# mensajes de usuario
dmesg -f user

# mensajes de usuario, kernel y demonios
dmesg -f user,kern,daemon
```

Otra opción de filtrado es por importancia del aviso, vemos las opciones por orden de importancia.

- **emerg** – error critico del sistema
- **alert** – debes solucionar el error rápidamente o el sistema entrara en modo «emerg»
- **crit** – en entorno esta en condiciones críticas
- **err** – errores
- **warn** – advertencias
- **notice** – el sistema es normal pero pon atención a este punto
- **info** – nota informativa
- **debug** – avisos de depuración

Para lograr el objetivo utilizamos «-l», que ademas nos permite especificar varios filtros (si es necesario).

```
# mensajes de emergencia
 dmesg -l emerg

# mensajes de emergencia, de alerta y críticos
 dmesg -l emerg,alert,crit
```

Borrar los mensajes del buffer

A diferencia de la lectura de mensajes, para borrar todos los mensajes debes ser root o usuario con privilegios. Antes de vaciar todos los avisos, existe la opción de imprimir en pantalla y borrar, así los borras y visualizas a la vez.

```
sudo dmesg -c
```

Si quieres borrar todo sin imprimir nada…

```
sudo dmesg -C
```

Como ultimo apunte del articulo, dmesg también permite guardar los avisos en un archivo antes de borrarlo por completo.

```
dmesg > dmesg_messages
```

Existen más opciones que puedes ver en su manual al ejecutar el siguiente comando.

```
man dmesg
```

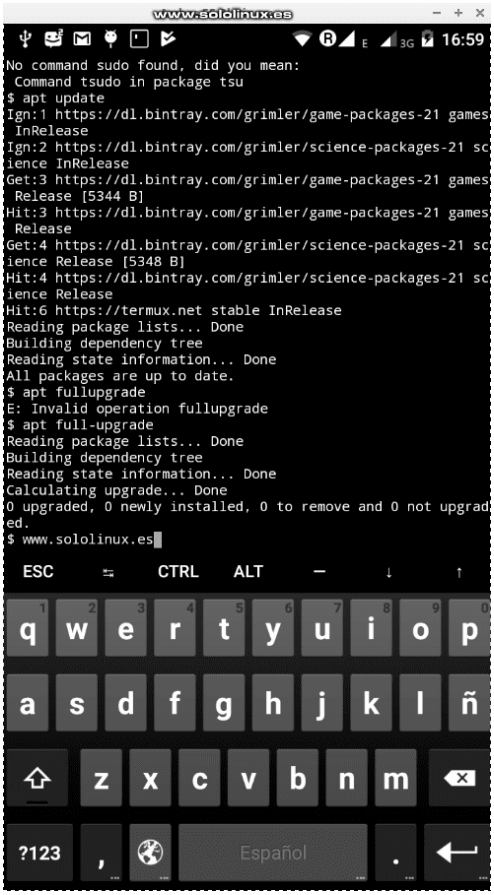
Instalar sudo en Android

Aunque a veces ponen algunas trabas, en **Android** puedes personalizar y editar cualquier configuración. Y es evidente que si eres linuxero quieres tener sudo instalado en tu dispositivo, ya que se amplían las funciones como si fuera cualquier distribución linux.

En este artículo, vemos como **instalar sudo en Android** en el **emulador de terminal Termux**. La elección del **terminal Termux** no es casualidad, simplemente es el mejor, apenas notaras la diferencia con la terminal de tu **distro linux**.

Lo primero que hacemos es instalar Termux (en tu dispositivo), desde alguna de las dos opciones que te propongo:

- Instalar Termux desde PlayStore
- Instalar Termux desde F-Droid



Una vez instalado Termux comenzamos la instalación de sudo.

Sudo solo funciona en dispositivos rooteados, asegúrese antes de continuar.

Instalar sudo en Android

Abrimos la herramienta Termux, es recomendable actualizar el dispositivo como lo hacemos normalmente en linux.

```
apt update && apt upgrade
```

Necesitamos **GIT**.

```
pkg install git
```

Clonamos el repositorio de **Gitlab**.

```
git clone https://gitlab.com/st42/termux-sudo.git
```

Abrimos el directorio **termux-sudo.**

```
cd termux-sudo
```

Necesitamos tener instalada la biblioteca «**ncurses-utils**».

```
pkg install ncurses-utils
```

Copiamos sudo en /data/ (la ruta puede variar).

```
cat sudo > /data/data/com.termux/files/usr/bin/sudo
```

Concedemos los permisos requeridos.

```
chmod 700 /data/data/com.termux/files/usr/bin/sudo
```

A partir de ahora puedes acceder con sudo a tu dispositivo Android, ten cuidado, no borres nada a no ser que sepas lo que haces. Todo lo que hagas es bajo tu responsabilidad. Existen aplicaciones en PlayStore que instalan sudo, pero no sabemos a ciencia cierta que más instalan.

Como curiosidad puedes comenzar instalando htop.

```
sudo apt install htop
```

Ejecutar htop.

```
htop
```

Cómo eliminar Snap completamente del sistema

Hace otros días alguien me comento que tenía problemas para **borrar snap**, en contra de lo que puede parecer, el no se refería a las aplicaciones, sino a la propia herramienta en si.

Aprovechando la consulta, hacemos este mini articulo donde vemos como borrar totalmente la herramienta, y sus aplicaciones. Creamos el articulo tomando como base un Ubuntu, pero el proceso es similar en otras distribuciones linux.

Cómo eliminar Snap completamente del sistema
Lo primero que debemos hacer, es listar las aplicaciones instaladas en forma de paquetes snap.

```
sudo snap list
```

En nuestro ejemplo tenemos instalado el editor Atom…

```
sololinux # snap list
Name           Version         Rev       Tracking          Publisher       Notes
atom           1.44.0          247       stable            snapcrafters    classic
core           16-2.43.2       8592      stable            canonical✓      core
```

Eliminamos las herramientas instaladas (Atom).

```
sudo snap remove atom
```

Una vez las tenemos todas fuera de nuestro sistema, vamos a eliminar la aplicación base que también consume lo suyo.

```
sudo apt autoremove --purge snapd
```

Ahora borramos el directorio de la aplicación.

```
sudo rm -rf ~/snap
```

También eliminamos el apunte de /var/cache.

```
sudo rm -rf /var/cache/snapd
```

En este momento ya no debería existir ningún rastro de snap, por si acaso ejecutamos purge (en Ubuntu, Linux Mint, y derivados).

```
sudo apt purge snapd
```

Tal vez te interesen los siguientes artículos anteriores.

- Qué es y como instalar Snap
- Liberar espacio en el disco ocupado por Snap
- Snap vs Flatpak

Qué es Load Average en linux, explicado por un sysadmin

Herramientas y comandos propietarios de **sistemas basados en Unix**, nos muestran el «**Load Average**», conocido en nuestro idioma materno como «**promedio de carga**».

El load average en Linux, son los promedios de carga del sistema que nos muestra la demanda de subprocesos (tareas) que se están ejecutando, y los que esperan. La gran mayoría de herramientas y comandos imprimen tres valores de promedios.

- (1) – Ultimo minuto.
- (5) – Últimos cinco minutos.
- (15) – Últimos quince minutos.

Pero ojo!!!, estos tres números (promedios de carga) vinculados a 1, 5 y 15 minutos, realmente no son promedios, y tampoco son 1, 5 y 15 minutos exactos. Son las constantes usadas en una ecuación, que calcula las sumas de valores que se van alterando cada cinco segundos. La verdad es, que el load average de 1, 5, y 15 minutos no es real, son muchos más.

Explicar la ecuación es largo y complejo, si tienes esa inquietud puedes revisar este articulo.

Por ejemplo, si tenemos un sistema con una carga 0 (cero) e iniciamos una tarea en bucle que requiera de procesador, se supone que pasado un minuto el promedio de carga sería 1 (uno), pues no, no es uno, depende en gran manera del tipo de tarea pero oscilará entre 0.3 y 0.6.

Esto tiene su explicación por dos motivos, el primero y más importante es por el tipo de calculo ecuacional, el segundo es, porque antes los sistemas basados en Unix solo calculaban los procesos a la espera de CPU, Linux también cuenta los procesos que esperan a otros procesos, por ejemplo, los que permanecen a la espera de leer o escribir en el disco.

Es algo normal que a veces nos encontremos valores que no tienen ningún sentido, podemos tener la carga con un valor 0.1, y de repente nos encontramos con 3. Como te dije antes, son promedios que a veces se alargan en el tiempo.

Si tu sistema tiene múltiples CPU o una CPU de varios núcleos. El promedio de carga funciona de manera diferente. Por ejemplo, si tienes un promedio de 2 en un sistema con una sola CPU, esto significa que tu sistema se sobrecargó en un 100%, osea, un proceso estaba usando la CPU en exclusica mientras otro proceso esperaba.

En un sistema con dos CPU, esto sería un uso completo: dos procesos diferentes usaban las dos CPU durante todo el tiempo. En un sistema con cuatro CPU, dos procesos utilizaban dos CPU, mientras que las dos restantes permanecen a la espera.

Como anécdota te puedo decir, que el otro dia haciendo un backup de un sitio web pesado (75Gb), al comprimir… el **load average** se disparó 12 (con un Xeon E5-2670). Lo curioso es que pasados 3 minutos de terminar la compresión el primer valor aun lo teníamos en 11.

Como ver el Load Average

Uff, buena pregunta. La gran mayoría de herramientas ofrecen este dato, pero tranquilo no es necesario que instales ninguna herramienta ajena para obtener un dato que a título orientativo es perfecto.

Si hablamos de comandos básicos tenemos «**uptime**» y «**w**». Vemos unos ejemplos:

```
uptime
```

Salida…

```
sololinux ~ # uptime
11:08:00    up    3:39,    1 user,    load average: 0,65, 0,63, 0,64
```

```
w
```

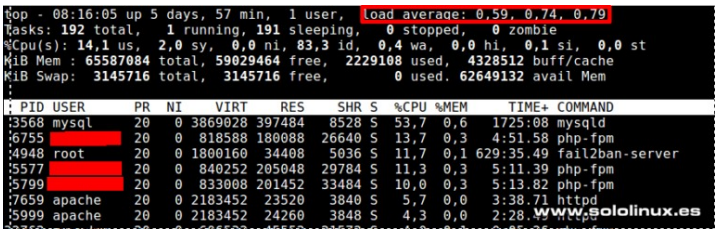
Salida…

```
sololinux ~ # w
   11:10:03 up 3:41, 1 user, load average: 0,56, 0,67, 0,65
USUARIO     TTY      DE       LOGIN@   IDLE   JCPU   PCPU  WHAT
sololinux   tty7     :0       07:35    3:41m  5:01   0.24s  /sbin/upstart –
```

Como aplicaciones de monitoreo básicas nos encontramos con las archiconocidas **Top** y **Htop**.

```
top
```

Imagen de ejemplo…



Htop es una aplicación externa, pero viene incluida en todos los repositorios oficiales de las distros linux (recomendada). Su instalación es simple…

Ubuntu, Debian, Linux Mint, y derivados:

```
sudo apt install htop
```

CentOS, RHEL, Fedora, y derivados:

```
sudo yum install htop

sudo dnf install htop
```

Arch Linux, Manjaro, y derivados:

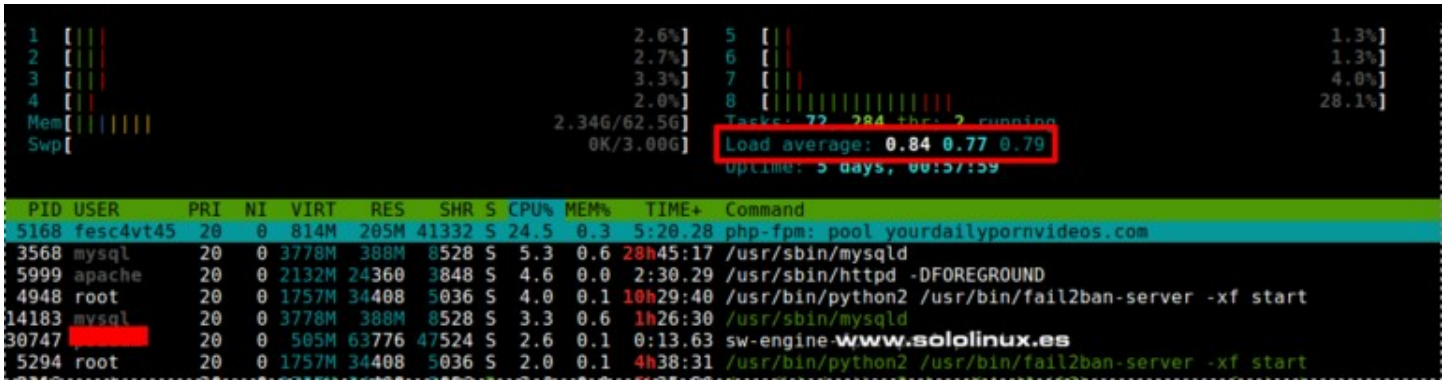
```
sudo pacman -S htop
```

Open Suse, Suse, y derivados:

```
sudo zypper install htop
```

Ejecutamos htop.

```
htop
```



Conclusión final
Es importante saber cuántas CPU tiene tu maquina. Un promedio de carga de 5 indica que un sistema está sobrecargado masivamente, pero si cuentas con 6 cpu, tu sistema es correcto.

Es útil en servidores y otros sistemas empresariales, para detectar posibles errores. Aun así, no son valores definitivos, pero por lo menos orientativos.

Síguenos en las Redes:

Como usar el comando w con ejemplos

El **comando w**, es un gran desconocido por los usuarios noveles que cuando se topan con el, lo utilizan a diario. Si buscas un chivato que te diga quien se conecta al sistema, a que hora, que hace, y más. El **comando w** es para ti.

El encabezado de la salida del comando, ya nos muestra información valiosa, como la hora actual, cuánto tiempo ha estado funcionando el sistema, cuántos usuarios están actualmente conectados y los **promedios de carga** del sistema.

La salida del comando nos aporta mucha más información, en este articulo vemos con detalle y ejemplos en formato de imagen, cómo debes operar con esta herramienta que viene por defecto en todas las **distribuciones linux**.

Como usar el comando w

Para una mejor comprensión lectora, desglosamos el articulo en varias secciones: sintaxis, salida, y comando con opciones.

Sintaxis del comando w

Como es habitual en este tipo de herramientas, su uso es extremadamente simple. Observa la sintaxis.

```
w [opciones] usuario [...]
```

Campos de la salida

Al ejecutar **w** nos encontramos con estos campos:
USUARIO TTY DE LOGIN@ IDLE JCPU PCPU WHAT

- **USUARIO/USER** – Nombre del usuario.
- **TTY** – Tipo de terminal (depende del entorno).
- **DE/FROM** – Si es posible imprime el hostname o la ip.
- **LOGIN@** – Tiempo de actividad de la sesión.
- **IDLE** – Tiempo desde la última actividad.
- **JCPU** – Tiempo de uso de los procesos sujetos al tty.
- **PCPU** – Tiempo de uso del proceso actual sujeto al tty.
- **WHAT** – Proceso que ejecuta el usuario en este momento (depende de PCPU).

Comando w con ejemplos

Comando w

Puedes ejecutar el comando w sin ninguna opción.

```
w
```

```
sololinux ~ # w
 18:10:44 up  2:44,  3 users,  load average: 0,86, 0,89, 0,91
USUARIO  TTY      DE              LOGIN@   IDLE   JCPU   PCPU WHAT
sergio   tty7     :0              15:27    2:43m  2:33   0.22s /sbin/upstart -
demo1    tty8     :1              17:39    2:43m  2.67s  0.15s /sbin/upstart -
demo2    tty9     :2              17:40    2:43m  3.28s  0.16s /sbin/upstart -
sololinux ~ # www.sololinux.es
```

Comando w -h

Al aplicar la opción h, no se imprime el encabezado.

```
w -h
```

```
sololinux ~ # w -h
sergio   tty7     :0              15:27    2:45m  2:35   0.22s /sbin/upstart --user
demo1    tty8     :1              17:39    2:45m  2.67s  0.15s /sbin/upstart --user
demo2    tty9     :2              17:40    2:45m  3.29s  0.16s /sbin/upstart --user
sololinux ~ # www.sololinux.es
```

Comando w usuario

Podemos especificar los datos de un usuario en particular.

```
w usuario
```

```
sololinux ~ # w demo1
 18:20:34 up  2:54,  3 users,  load average: 0,76, 0,69, 0,78        www.sololinux.es
USUARIO  TTY      DE              LOGIN@   IDLE   JCPU   PCPU WHAT
demo1    tty8     :1              17:39    2:53m  2.68s  0.15s /sbin/upstart --user
sololinux ~ #
```

Comando w -u

Ignorar los usuarios del proceso que se ejecuta actualmente.

```
w -u
```

```
sololinux ~ # w -u
 18:12:57 up  2:46,  3 users,  load average: 0,70, 0,76, 0,86
USUARIO  TTY      DE              LOGIN@   IDLE   JCPU   PCPU WHAT
sergio   tty7     :0              15:27    2:46m  2:37   0.22s /sbin/upstart --user
demo1    tty8     :1              17:39    2:46m  2.67s  0.15s /sbin/upstart --user
demo2    tty9     :2              17:40    2:46m  3.29s  0.16s /sbin/upstart --user
sololinux ~ # www.sololinux.es
```

Comando w -s

Si aplicamos la opción s, la salida es en formato corto. No se imprime el tiempo de inicio de sesión, JCPU ni tampoco el campo PCPU.

```
w -s
```

```
sololinux ~ # w -s
 18:14:35 up  2:48,  3 users,  load average: 0,60, 0,72, 0,83
USUARIO  TTY      DE                IDLE WHAT
sergio   tty7     :0                2:47m /sbin/upstart --user
demo1    tty8     :1                2:47m /sbin/upstart --user
demo2    tty9     :2                2:47m /sbin/upstart --user
sololinux ~ # www.sololinux.es
```

Comando w -f
No imprimir el hostname o ip del usuario.

```
w -f
```

```
sololinux ~ # w -f
 18:15:05 up  2:48,  3 users,  load average: 0,62, 0,71, 0,82
USUARIO  TTY        LOGIN@   IDLE   JCPU   PCPU WHAT
sergio   tty7      15:27    2:48m  2:40   0.22s /sbin/upstart --user
demo1    tty8      17:39    2:48m  2.67s  0.15s /sbin/upstart --user
demo2    tty9      17:40    2:48m  3.29s  0.16s /sbin/upstart --user
sololinux ~ # www.sololinux.es
```

Comando w -i
Si es posible imprimirá la ip del usuario y no el host.

```
w -i
```

```
sololinux ~ # w -i
 18:16:59 up  2:50,  3 users,  load average: 0,61, 0,70, 0,81      www.sololinux.es
USUARIO  TTY      DE              LOGIN@   IDLE   JCPU   PCPU WHAT
sergio   tty7     :0              15:27    2:50m  2:43   0.22s /sbin/upstart --user
demo1    tty8     :1              17:39    2:50m  2.68s  0.15s /sbin/upstart --user
demo2    tty9     :2              17:40    2:50m  3.30s  0.16s /sbin/upstart --user
sololinux ~ #
```

Comando w -o
Salida estilo antiguo (espacios en blanco para tiempos de inactividad de menos de un minuto).

```
w -o
```

```
sololinux ~ # w -o
 18:19:26 up  2:52,  3 users,  load average: 0,58, 0,64, 0,77      www.sololinux.es
USUARIO  TTY      DE              LOGIN@   IDLE   JCPU   PCPU WHAT
sergio   tty7     :0              15:27    2:52   2:46m         /sbin/upstart --user
demo1    tty8     :1              17:39    2:52                 /sbin/upstart --user
demo2    tty9     :2              17:40    2:52                 /sbin/upstart --user
sololinux ~ #
```

Comando w -V
Identificar la versión de la herramienta.

```
w -V
```

```
sololinux ~ # w -V
w from procps-ng 3.3.10
sololinwww.sololinux.es
```

Síguenos en las Redes:




Uso y ejemplos del comando who

El **comando who**, es una herramienta básica de **sistemas Unix** con una función bien definida. Aportar datos sobre los usuarios del sistema. En cierta manera es muy similar al **comando w**, pero quizás un poco más limitado.

- Hora del último arranque del sistema
- Nivel de ejecución actual del sistema
- Lista de usuarios conectados

Es destacable que **who** está incluido en la **Single Unix Specification**, que es la norma de estándares para que un sistema operativo se pueda denominar **Unix**. En este articulo conocemos **who** y sus opciones.

Uso y ejemplos del comando who

La sintaxis de la herramienta es similar a otras propietarias de Unix.

```
who [options] [filename]
```

Tiene varias opciones, pero también lo puedes ejecutar en su forma básica.

```
who
```

Ejemplo de salida…

```
sololinux ~ # who
sergio   tty7         2020-02-13 07:21 (:0)
sololinux ~ # www.sololinux.es
```

En la siguiente tabla vemos las opciones disponibles:

Opción corta	Opción extendida	Función
-a	--all	Todas las opciones más utilizadas a la vez: -b, -d, --login, -p, -r, -t, -T, -u.
-b	--boot	Fecha y hora del último inicio del sistema.
-d	--dead	Imprime los procesos muertos.
-H	--heading	Encabezados explicativos en cada columna.
	--ips	Si es posible se imprimirá la ip y no el host.
-l	--login	Muestra los procesos del login del sistema.
	--lookup	Si es posible canonicalizara los nombres de host a través de DNS.
-m		Imprime sólo el nombre del host y del usuario asociado con la entrada (terminal donde se ejecuto el comando).
-p	--process	Imprime los procesos activos generados por init.
-q	--count	Nos muestra el número y nombre de los usuarios conectados.
-r	--runlevel	Imprime el nivel de ejecución actual (runlevel). ***Ver NOTA al final del articulo***
-s	--short	Imprime el nombre, tipo, y fecha con horario.
-t	--time	Si es posible imprimirá la ultima vez que se modificó el reloj del sistema.
-T -w	--mesg	Agrega un caracter que indica el estado de la terminal. Si se puede escribir (+), si no se permite (?).
-u	--users	Imprime el tiempo de cada usuario y la ID de proceso.
	--writable	Mismo uso que la opción -T.
	--message	Mismo uso que la opción -T.
	--version	Imprime la versión de la herramienta.
	--help	Imprime la ayuda del comando.

Ejemplos de who

El más utilizado (aparte del básico who), es con la opción -a.

```
who -a
```

```
sololinux ~ # who -a
          arranque del sistema 2020-02-13 07:20
          `run-level' 5 2020-02-13 07:20
LOGIN     tty1         2020-02-13 07:20              1187 id=tty1
sergio   + tty7        2020-02-13 07:21 12:08        1221 (:0)
sololinux ~ # www.sololinux.es
```

Otro comando a destacar es, agregar el encabezado explicativo, y el carácter que nos indica el estado de la terminal. Si revisas la tabla de opciones verás que necesitamos las opciones **-H** y **-T**, se permite insertarlos juntos o por separado.

```
who -T -H
```

```
sololinux ~ # who -T -H
NOMBRE     LÍNEA       TIEMPO           COMENTARIO
sergio   + tty7        2020-02-13 07:21 (:0)
sololinux ~ # www.sololinux.es
```

Como último ejemplo fusionamos los dos anteriores, **-a, -H y -T**. Recuerda que con la opción -a se incluye -T, no es necesario que la insertes.

```
who -aH
```

```
sololinux ~ # who -aH
NOMBRE     LÍNEA       TIEMPO                 PID COMENTARIO SALIDA
          arranque del sistema 2020-02-13 07:20
          `run-level' 5 2020-02-13 07:20
LOGIN     tty1         2020-02-13 07:20              1187 id=tty1
sergio   + tty7        2020-02-13 07:21 12:24        1221 (:0)
sololinux ~ # www.sololinux.es
```

Nota: Qué es runlevel

En la tabla de opciones nos encontramos con la opción -**r / –runlevel**. El **runlevel** o nivel de ejecución, es un número (solo un dígito) preestablecido que define el estado operacional de un sistema Unix (como Linux) manejado por init.

Dependiendo del nivel de ejecución, se permiten unas combinaciones de procesos en ejecución, u otras. El kernel estándar de Linux admite siete niveles de ejecución diferentes, las conocemos.

- **0** – System healt.
- **1** – Usuario único.
- **2** – Múltiples usuarios sin NFS (sistema de archivos en red).
- **3** – Multiples usuarios en linea de comandos.
- **4** – Definido por el usuario.
- **5** – Múltiples usuarios en la GUI (interfaz gráfica de usuario).
- **6** – Reboot.

Inxi: Identificar el hardware del sistema

Inxi es un potente **script bash**, con la capacidad de identificar el hardware del sistema e imprimir los datos de una forma limpia y clara.

Muestra información sobre el hardware del sistema (disco duro, tarjeta gráfica, de audio, de red, CPU, RAM, etc…), así como otros detalles del sistema como los drivers, Xorg, distribución linux, entorno de escritorio, kernel, procesos, tiempo de actividad, y muchos más.

Su uso no está muy extendido por desconocimiento, pero seguro que si lo pruebas una vez se convertirá en tu herramienta perfecta. En este articulo tratamos de acercar al usuario la herramienta inxi, así como sacar el máximo provecho de la misma.

Inxi: Identificar el hardware del sistema
Inxi suele venir preinstalado en la mayoría de sistemas linux, si no es tu caso vemos como hacerlo.

Debian, Ubuntu, Linux Mint, y derivados:

```
sudo apt install inxi
```

Centos, RHEL, Fedora, y derivados:

```
#CentOS-RHEL
 sudo yum install inxi

#Fedora
 sudo dnf install inxi
```

Arch Linux, Manjaro, y derivados:

```
sudo pacman -S inxi
```

Open Suse, Suse, y derivados:

```
sudo zypper install inxi
```

Una vez instalada, puedes ejecutar la herramienta inxi en su forma básica.

```
inxi
```

Podrás ver una salida similar a:
[root@ejemplo ~]# inxi
CPU: Quad Core Intel Xeon E3-1245 v5 (-MT MCP-) speed/min/max: 3681/800/3900 MHz Kernel: 3.10.0-1062.12.1.el7.x86_64 x86_64
Up: 7d 14h 40m Mem: 14477.7/64049.9 MiB (22.6%)
Storage: 1.40 TiB (9.8% used) Procs: 195 Shell: bash 4.2.46 inxi: 3.0.37

Inxi también nos permite ampliar la información de forma detallada con su opción -F.

```
inxi -F
```

Podemos ejecutar otras opciones para especificar lo que queremos identificar realmente. Vemos algunos ejemplos.

Para identificar el sistema ejecutamos el siguiente comando.

```
inxi -S
```

System: Host: xxxx.xxxx.xxxx Kernel: 3.10.0-1062.12.1.el7.x86_64 x86_64 bits: 64 Console: tty 0
Distro: CentOS Linux release 7.7.1908 (Core)

Saber que placa base, bios, y más.

```
inxi -M
```

Machine: Type: Kvm System: Supermicro product: SYS-5039MS-H12TRF-ON002 v: 0123456789 serial: S225408X6330575
Mobo: Supermicro model: X11SSE-F v: 1.01 serial: ZM163S009973 UEFI [Legacy]: American Megatrends v: 2.2
Date: 05/16/2018

Quieres saber que cpu tienes?

```
inxi -C
```

CPU: Topology: Quad Core model: Intel Xeon E3-1245 v5 bits: 64 type: MT MCP L2 cache: 8192 KiB
Speed: 2013 MHz min/max: 800/3900 MHz Core speeds (MHz): 1: 893 2: 879 3: 826 4: 821 5: 1765 6: 848 7: 851 8: 1508

Ahora identificamos la tarjeta gráfica…

```
inxi -G
```

Graphics: Card: Intel Mobile 945GM/GMS 943/940GML Express Integrated Graphics Controller
Display Server: X.Org 1.18.4 drivers: intel (unloaded: fbdev,vesa)
Resolution: 1280×1024@60.02hz
GLX Renderer: Mesa DRI Intel 945GM GLX Version: 1.4 Mesa 18.0.5

Detalles del hardware de audio.

```
inxi -A
```

Audio: Card Intel NM10/ICH7 Family High Definition Audio Controller
driver: snd_hda_intel
Sound: Advanced Linux Sound Architecture v: k4.15.0-74-generic

Los dispositivos de red.

```
inxi -N
```

Network: Card-1: Marvell 88E8038 PCI-E Fast Ethernet Controller driver: sky2
Card-2: Intel PRO/Wireless 4965 AG or AGN [Kedron] Network Connection
driver: iwl4965

Dispositivos de almacenamiento.

```
inxi -D
```

Drives: Local Storage: total: 1.40 TiB used: 139.99 GiB (9.8%)
ID-1: /dev/sda vendor: Samsung model: MZ7LN512HMJP-00000 size: 476.94 GiB
ID-2: /dev/sdb vendor: Samsung model: MZ7LN512HMJP-00000 size: 476.94 GiB
ID-3: /dev/sdc vendor: Samsung model: MZ7LN512HMJP-00000 size: 476.94 GiB

Identificamos el raid (si existe).

```
inxi -R
```

RAID: Device-1: md1 type: mdraid status: active raid: raid-5 report: 3/3 UUU Components:
online: sda2~c0 sdb2~c1 sdc2~c3
Device-2: md0 type: mdraid status: active raid: raid-5 report: 3/3 UUU Components:
online: sda3~c0 sdc3~c3 sdb3~c1

Las particiones del sistema.

```
inxi -P
```

Partition: ID-1: / size: 924.97 GiB used: 139.83 GiB (15.1%) fs: ext4 dev: /dev/md1
ID-2: /boot size: 1.93 GiB used: 168.6 MiB (8.5%) fs: ext4 dev: /dev/md0
ID-3: swap-1 size: 1024.0 MiB used: 0 KiB (0.0%) fs: swap dev: /dev/sda1
ID-4: swap-2 size: 1024.0 MiB used: 0 KiB (0.0%) fs: swap dev: /dev/sdb1
ID-5: swap-3 size: 1024.0 MiB used: 0 KiB (0.0%) fs: swap dev: /dev/sdc1

Los sensores.

```
inxi -S
```

Sensors: System Temperatures: ipmi cpu: 41 C mobo: 38 C
Fan Speeds (RPM): ipmi cpu: 7700 fan-1:
System Temperatures: lm-sensors cpu: 29.8 C mobo: 27.8 C
Fan Speeds (RPM): lm-sensors N/A

Información de uso.

```
inxi -I
```

Info: Processes: 193 Uptime: 7d 14h 50m Memory: 62.55 GiB used: 14.23 GiB (22.8%) Init: systemd runlevel: 3
Shell: bash inxi: 3.0.37



Qué son los procesos en linux y cómo identificarlos

Qué son los procesos en linux y cómo identificarlos

Qué son los procesos

Tal vez no eres consciente, pero cuando ejecutamos un comando, herramienta, aplicación, software, etc, en Linux, se genera (como mínimo) un nuevo proceso.

Para los más profanos, podríamos decir que es como un software que trabaja en el interior de un sistema operativo. Esta tarea tiene su forma independiente de trabajar, una misión exclusiva, y sus propios permisos. Datos e instrucciones de cómo proceder, contador, registros, y otros parámetros es lo que incluye un proceso.

Durante el tiempo que dure el proceso hará uso de los recursos del sistema, la CPU para ejecutar sus instrucciones propias, y de la **memoria ram** para almacenar datos. Es evidente que también utiliza los archivos propietarios del sistema, además de interactuar con los dispositivos de almacenamiento.

Los procesos permiten la interacción con el sistema para que la gestión sea a nivel superior, mientras tanto el kernel de Linux se encarga del nivel inferior.

PID (número de identificación del proceso)

Linux asigna un número único de cinco dígitos a cada proceso. Al número generado se le conoce como PID (número de identificación de proceso), y es una manera excelente para identificar los procesos, ya que no es posible tener dos PID iguales, cada proceso tiene su número exclusivo hasta que muera.

Si quieres saber el PID (pueden ser varios) de un proceso específico, es tan sencillo como ejecutar el siguiente comando (debes saber el nombre):

```
pgrep [proceso]
```

Procesos ejemplo del **navegador chomium**…

```
 sololinux ~ # pgrep chromium
2425
2446
2449
2472
2477
```

Existen otras alternativas, pero este comando es el más sencillo y fácil de usar.

Tipos de procesos

En linux nos encontramos con dos tipos de procesos.

- Background processes
- Frontend processes

Background processes

Se conocen como «**Background processes**» (procesos en segundo plano), a los procesos automáticos o no interactivos. Este tipo de procesos inician automáticamente sin intervención del usuario, tampoco requieren ninguna respuesta del mismo. Estos procesos son independientes, y una vez son creados harán su trabajo sin ninguna interacción con la entidad que los originó. Un ejemplo claro lo tenemos con los demonios y servicios del sistema.

Frontend processes

Los «**Frontend processes**» son procesos interactivos, para que me entiendas… fueron iniciados o lanzados por algún usuario. Un ejemplo claro son los comandos que ejecutamos en la **terminal**, siempre comienzan en primer plano. Otros procesos frontend son los que se inician desde interfaces gráficas, y permiten interactuar con ellos.

Procesos primarios y procesos secundarios

Es algo normal que varios usuarios trabajen simultáneamente en Linux, cada uno usa diferentes aplicaciones y comandos. Para poder diferenciar las instancias de ejecución, el núcleo identifica cada una de ellas.

Entonces los procesos se dividen en **procesos primarios** (también llamados padre), que son procesos que generan otros procesos en tiempo de ejecución, y los **procesos secundarios** (también llamados hijo) que son creados por otros procesos en ejecución.

Un detalle a tener en cuenta es, que cuando un proceso hijo se elimina antes que su padre, se convierte en un **proceso zombi** hasta que el primario informe al núcleo de su estado actual.
El problema puede surgir si el proceso primario finaliza antes que el secundario, en este caso el proceso hijo puede ser adoptado por init o por otro proceso diferente. Esto repercute negativamente en el rendimiento del sistema.

Estados de un proceso

Es importante entender los estados de un proceso, su ciclo de vida, y cómo los núcleos de la CPU lo tratan. En Linux tenemos los siguientes estados:

- **Running/Runnable – (R)**: Son los procesos en ejecución que están haciendo uso de la CPU.
- **Waiting (o sleeping)**: Son los procesos que están a la espera de que un recurso específico esté disponible (por ejemplo, E / S), o que suceda algo esperado. Estos se pueden clasificarse en:
  - **a –** procesos de espera cuya tarea pueden ser interrumpida por señales, o asesinados antes de que se termine su trabajo.
  - **b –** procesos de espera cuyo trabajo no puede ser interrumpido por ninguna señal o evento.
- **Stopped**: Un proceso se detiene al recibir la señal SIGSTOP. La ejecución del proceso se suspende y solo administrará las señales SIGKILL y SIGCONT. Por ejemplo, un proceso que se está depurando se encuentra Stopped (detenido).
- **Zombie**: En este caso el proceso no está vivo ni muerto. Simplemente termino su tarea con un exit (), pero está esperando su entrada a la tabla de procesos.

El proceso init

En linux el proceso init es el padre de todos los procesos. Es el primer proceso en iniciar cuando arranca Linux. Su función principal es importantísima, ya que es el encargado de crear el resto de procesos a partir de /etc inittab. No tiene proceso padre lo inicia el kernel, por tanto no se puede matar.

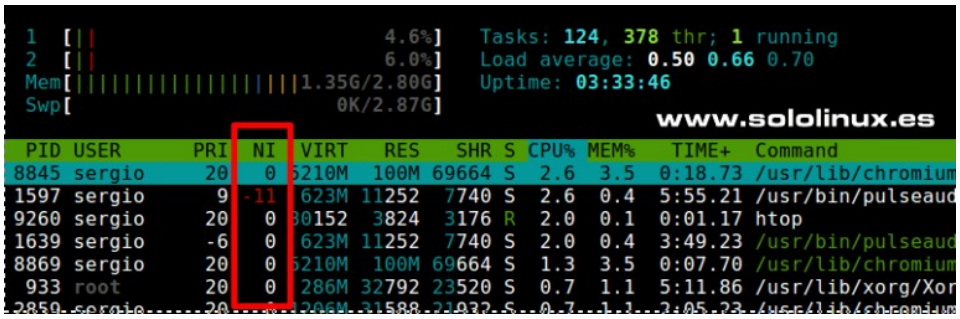
Ver los procesos en ejecución
Existen muchas herramientas y comandos para ver los procesos que se están ejecutando actualmente, como **htop, top**, etc. Pero sin duda alguna el mejor para esta tarea es, el **comando ps**.

```
ps ax
```

```
 sololinux ~ # ps ax
   PID TTY      STAT   TIME COMMAND
     1 ?        Ss     0:04 /sbin/init
     2 ?        S      0:00 [kthreadd]
     4 ?        I<     0:00 [kworker/0:0H]
     6 ?        I<     0:00 [mm_percpu_wq]
     7 ?        S      0:00 [ksoftirqd/0]
     8 ?        I      0:09 [rcu_sched]
     9 ?        I      0:00 [rcu_bh]
    10 ?        S      0:00 [migration/0]
    11 ?        S      0:00 [watchdog/0]
    12 ?        S      0:00 [cpuhp/0]
    13 ?        S      0:00 [cpuhp/1]
    14 ?        S      0:00 [watchdog/1]
    15 ?        S      0:00 [migration/1]
    16 ?        S      0:04 [ksoftirqd/1]
    18 ?        I<     0:00 [kworker/1:0H]
    19 ?        S      0:00 [kdevtmpfs]
    20 ?        I<     0:00 [netns]
    21 ?        S      0:00 [rcu_tasks_kthre]
    22 ?        S      0:00 [kauditd]
    25 ?        S      0:00 [khungtaskd]
```

Prioridad de los procesos
En Linux, todos los procesos tienen una prioridad de ejecución. Los procesos con una mayor prioridad obtienen más CPU que otros con una menor prioridad, todo depende de su importancia para el sistema.

En este ejemplo utilizamos la herramienta htop para conocer la prioridad de los procesos, la columna NI indica su valor:



Si eres root puedes modificar la prioridad de un proceso con el siguiente comando.

```
renice [valor] [proceso]
```

También es posible ejecutar un comando o aplicación con una prioridad definida.

```
nice -n 10 htop
```

Conclusión
Independientemente de que uses la terminal o la GUI, siempre tendrás procesos activos. Pueden estar corriendo, detenidos, durmiendo o como zombis. Existen muchas herramientas que permiten ver y administrar los procesos.

El comando kill se encarga de terminar procesos.

Aprender a administrar los procesos es una tarea importante que debes manejar con soltura, independientemente del nivel de conocimientos que tengas.

Instalar Android en Ubuntu y derivados

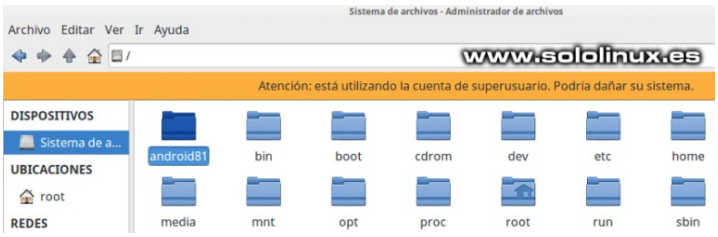
Es indiscutible que Android, es el sistema operativo más utilizado en smartphones y otros dispositivos inteligentes. Incluso más extendido que el **iOS** de los **iPhone**.
La evolución de Android es constante, e incluso muchos desarrolladores tratan de incrustarlo en sistemas de tipo **PC** (personal computer). La versión open source de Android, se puede instalar en máquinas con linux (como sistema operativo base), y aunque es posible que suceda algún error imprevisto vamos por buen camino.
En este artículo, vemos cómo instalar el **sistema operativo Android-x86** en Ubuntu, Linux Mint o cualquier derivado. Antes de comenzar necesitamos un par de cosas que detallamos a continuación.

- **Android-x86** – Por su estabilidad y ligereza nosotros elegimos Android 8.1 en su versión de 32bits. La descargamos: android-x86-8.1-r3.iso. Para otras versiones o arquitecturas visita su zona de descargas oficial.
- **Grub Customizer** – Esta aplicación nos permite configurar el sistema operativo predeterminado, agregar o eliminar entradas de inicio, de forma sencilla. Si no recuerdas como se instala, revisa este articulo.

Instalar Android en Ubuntu y derivados
Ahora debemos crear una carpeta en la raiz del sistema para alojar Android, en nuestro ejemplo la llamaremos «android81». Abrimos Thunar con permisos.

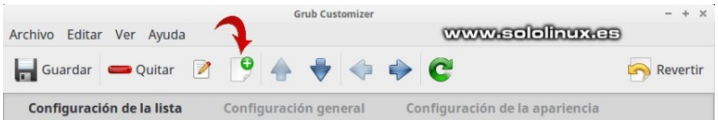
```
sudo thunar
```

Crea el nuevo directorio.



Abrimos la carpeta **android81**, pegamos la iso de Android que descargamos anteriormente (/android81/). Sobre la iso, pulsa el botón derecho del ratón y descomprime la imagen, al concluir puedes borrar la iso. En el mismo directorio (android81), creamos otra carpeta llamada «data». Cierra Thunar.

Ya casi tenemos preparado nuestro **sistema Android,** solo nos falta crear una **nueva entrada en el Grub.** Abrimos la herramienta **Grub Customizer**, y pulsamos en crear nuevo archivo de entrada.



Rellenamos los datos (si has descargado otra versión, debes modificar los valores o rutas que correspondan):

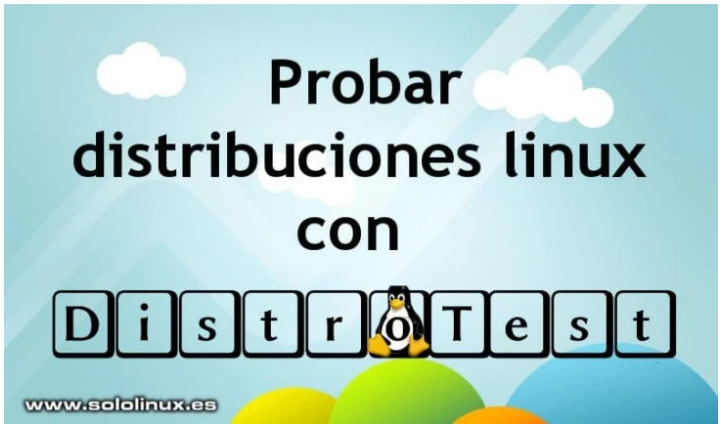
- **Nombre** – En nuestro ejemplo **Android OS 8.1**
- **Tipo** – Nos aparecen varias alternativas, debes seleccionar **Otro**
- **Secuencia de arranque** – Insertamos el siguiente script…

```
insmod part_gpt
 search --file --no-floppy --set=root
/androidos81/system.sfs
 linux /androidos81/kernel root=/dev/ram0
androidboot.selinux=permissive
buildvariant=userdebug SRC=/androidos81
 initrd /androidos81/initrd.img
```

Para terminar pulsa en aceptar y guardar menú. Reinicias tu sistema y podrás ver la opción de iniciar desde **Android OS 8.1**.

Nota final: Como ya dije anteriormente, aún no es un sistema considerado como estable, pero vamos por buen camino. Para desinstalar Android, solo tienes que borrar la entrada de **Grub Customizer** y el directorio completo **android81**. Estas instrucciones también son validas en otras distribuciones linux. Suerte.

Probar distribuciones linux sin instalarlas con DistroTest



Probar distribuciones linux sin instalarlas con DistroTest
La manera maś habitual de probar una **distribución linux** es, descargar su versión live y verificar su funcionamiento antes de decidirnos si la instalamos o no.

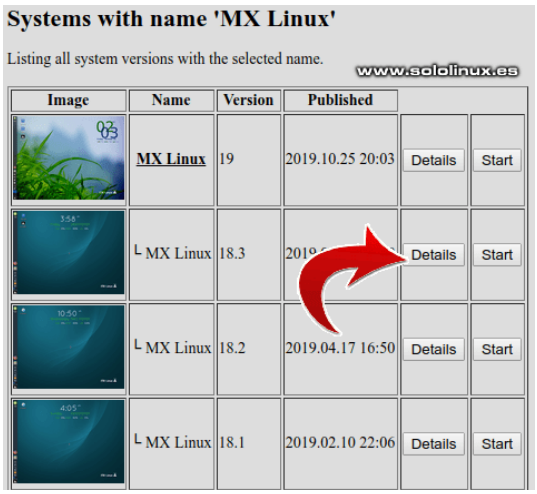
Existe otra manera, mucho más fácil y menos engorrosa, la ofrece DistroTest.net

Distrotest es un sitio web que ofrece 270 **distribuciones linux**, en 962 versiones (actualmente). Nos permite probar casi cualquier linux, sin tener que instalar o generar un **USB Live**. Es interesante que podemos interactuar con las distros desde una ventana del propio navegador, o mediante alguna herramienta de escritorio remoto, como **Remmina**.

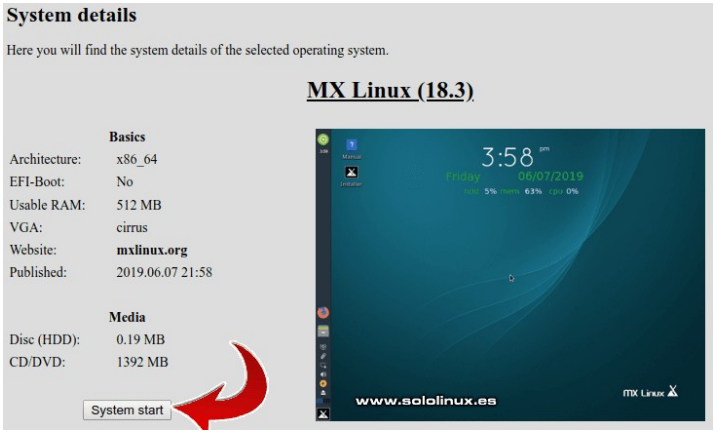
Testear distribuciones linux con DistroTest

Accedemos a DistroTest.net

En nuestro articulo de hoy, como ejemplo nos decantamos por **MX Linux**. Al pulsar en la distro MX Linux del listado que aparece en la web, nos aparece una ventana con todas las versiones disponibles. Tenemos dos opciones, ver detalles, o iniciar directamente. Nosotros elegimos detalles.

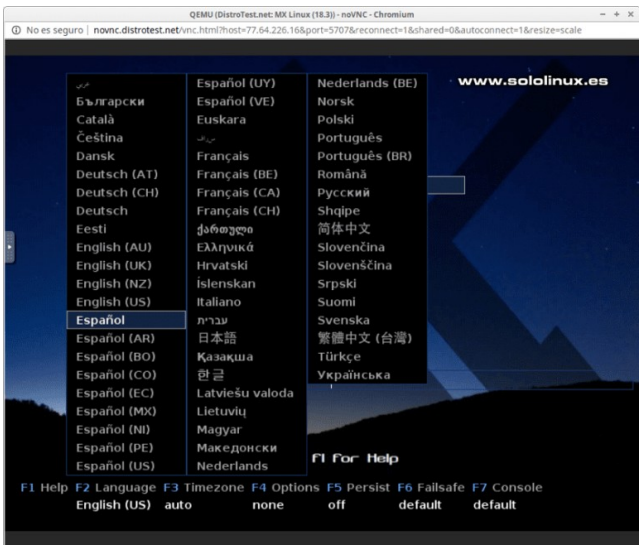


Esta ventana nos aporta datos sobre la **distribución linux** que vamos a testar. Pulsamos en iniciar sistema.

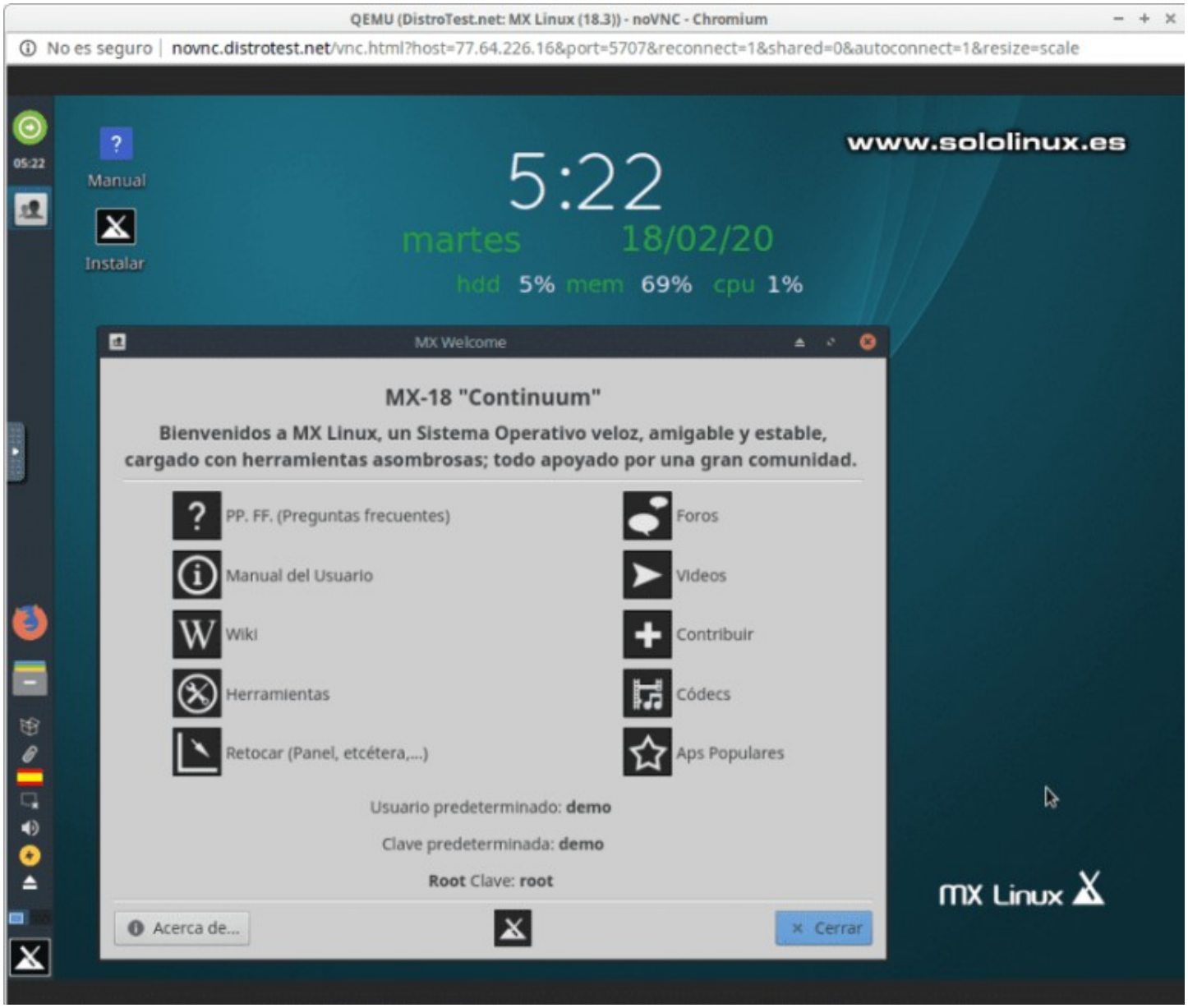


Se ofrecen varias opciones, lo más sencillo y rápido es utilizar el VNC-Viewer.

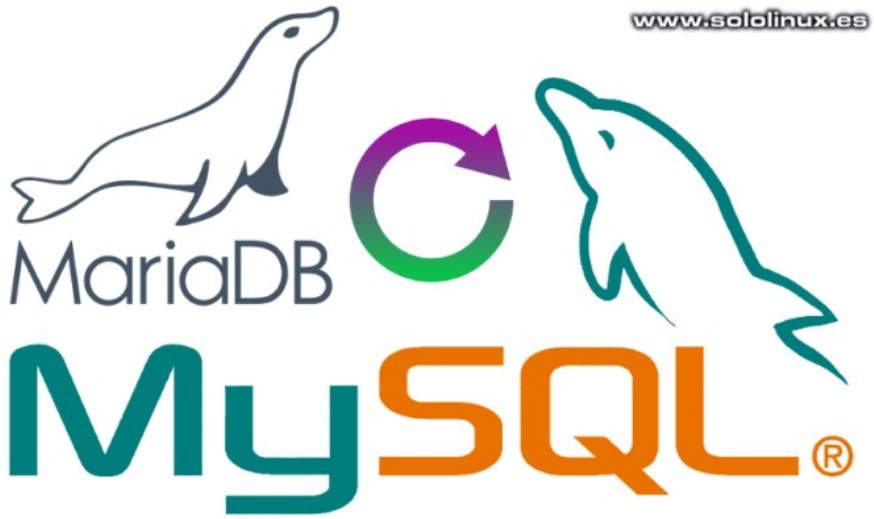
El sistema empieza a cargar, en pocos segundos vemos la típica pantalla de inicio MX Linux. Antes de continuar, habilitamos el lenguaje Español con la **tecla F2**. Pulsa **Enter**, para continuar con el arranque del sistema.



Listo, ya podemos verificar como funciona MX Linux 18.3.



Como iniciar, detener, reiniciar, habilitar MariaDB y MySQL



MySQL y su bifurcación MariaDB, son los dos sistemas de **gestión de base de datos** más utilizados. Incluso podríamos afirmar que, en todo lo que gira alrededor de linux… MySQL ya está en un segundo plano.

También es cierto que existe una cierta confusión sobre los comandos para iniciar, detener, reiniciar, habilitar, y ver el status de estos dos gestores de base de datos. En muchos sitios aparecen como si fueran los mismos, y no siempre es así.

Solo en las distribuciones que siguen utilizando el veterano administrador **SysVinit**, los comandos son los mismos. Actualmente la mayoría de **distribuciones linux** hacen uso de **Systemd**, y entonces sí que son diferentes (normalmente).

En este articulo vemos los comandos para iniciar, detener, reiniciar, habilitar, y conocer el estado de **MariaDB y MySQL**, dependiendo del administrador que usen, **Systemd** o **SysVinit**.

Comandos que modifican el estado de MariaDB y MySQL

Iniciar MariaDB / MySQL

Systemd

```
# MariaDB
 systemctl start mariadb.service
o
 systemctl start mariadb

# MySQL
 systemctl start mysql.service
o
 systemctl start mysql
```

SysVinit

```
# MariaDB / MySQL
 service mysql start
o
 /etc/init.d/mysql start
```

Detener MariaDB / MySQL

Systemd

```
# MariaDB
 systemctl stop mariadb.service
o
 systemctl stop mariadb

# MySQL
 systemctl stop mysql.service
o
 systemctl stop mysql
```

SysVinit

```
# MariaDB / MySQL
 service mysql stop
o
 /etc/init.d/mysql stop
```

Reiniciar MariaDB / MySQL

Systemd

```
# MariaDB
systemctl restart mariadb.service
o
systemctl restart mariadb

# MySQL
systemctl restart mysql.service
o
systemctl restart mysql
```

SysVinit

```
# MariaDB / MySQL
 service mysql restart
o
 /etc/init.d/mysql restart
```

Recargar MariaDB / MySQL

Systemd

```
# MariaDB
 systemctl reload mariadb.service
o
 systemctl reload mariadb

# MySQL
 systemctl reload mysql.service
o
 systemctl reload mysql
```

SysVinit

```
# MariaDB / MySQL
 service mysql reload
o
 /etc/init.d/mysql reload
```

Habilitar MariaDB / MySQL con el inicio del sistema

Systemd

```
# MariaDB
 systemctl enable mariadb.service
o
 systemctl enable mariadb

# MySQL
 systemctl enable mysql.service
o
 systemctl enable mysql
```

SysVinit

```
# MariaDB / MySQL
 chkconfig mysqld on
```

Conocer el estado de MariaDB / MySQL

Systemd

```
# MariaDB
systemctl status mariadb.service
o
systemctl status mariadb

# MySQL
systemctl status mysql.service
o
systemctl status mysql
```

SysVinit

```
# MariaDB / MySQL
 service mysql status
```

Síguenos en las Redes:

SoloLinux

Esta revista es de **distribución gratuita**, si lo consideras oportuno puedes ponerle precio. Tu también puedes ayudar, contamos con la posibilidad de hacer donaciones para la REVISTA, de manera muy simple a través de **PAYPAL**

AYUDANOS A SEGUIR CRECIENDO

PayPal™ Apoyo con una donación

Extraer las claves wifi WPA / WPA2 con Fluxion

A veces, nos podemos encontrar con la situación que necesitemos averiguar las keys wifi de una red a la que (actualmente) no tenemos acceso.

Seguro que muchos conocéis la herramienta linset de vk496, muy buena para ataques **MITM** (main in the middle) a WPA y WPA2. Sin embargo ya tiene sus años, y no parece que el autor le apetezca actualizarla.

Estamos de suerte, una aplicación llamada Fluxion es su sucesora. Esta herramienta de auditoría de seguridad e ingeniería social, tiene menos errores y más funciones. Con ella puedes recuperar la clave **WPA / WPA2 (hakear wifi)**, de un objetivo lanzando un ataque de ingeniería social (phishing).

Extraer las claves wifi WPA / WPA2 con Fluxion
Lo primero que hacemos es instalar fluxion (necesitas git).

Clonamos el repositorio de github.

```
git clone https://www.github.com/FluxionNetwork/fluxion.git
```

Abrimos la carpeta de la herramienta Fluxion, y la ejecutamos.

```
cd fluxion
```

```
./fluxion.sh
```

El script verifica que tu sistema tiene todas las librerías, y aplicaciones (ajenas) necesarias para que **Fluxion** funcione correctamente. Si aparece una pantalla similar a la imagen inferior… necesitas instalar los complementos faltantes.

Note preocupes, no pasa nada, Fluxion instalará por ti todo lo necesario con el siguiente comando (si está en los repositorios de tu sistema).

```
./fluxion.sh -i
```

En sololinux.es no fomentamos el pirateo, ni ayudaremos a sacar claves wifi wpa, ni claves wifi wpa2. Por tanto si has llegado a este punto del articulo es porque sabes lo que haces, y siempre bajo tu responsabilidad.

Lo que sí que podemos hacer es explicarte los pasos a seguir, son más simples de lo que puedan parecer, ya lo veras.

- Buscar una red inalámbrica objetivo.
- Lanzar el ataque Handshake Snooper.
- Capturar el Handshake (apretón de manos).
- Lanzamos el portal cautivo.
- Necesitas generar un AP falso que imite el punto de acceso original.
- Con un servidor DNS, se redirigen todas las solicitudes a nuestro sistema (al portal cautivo).
- Se crea un servidor web, que solicita a los usuarios que inserten de nuevo su clave WPA / WPA2.
- Debes desautentificar todos los clientes del objetivo para que se conecten a la AP del portal cautivo.
- Los intentos de autenticación en el portal cautivo se verifican con el archivo Handshake.
- Una vez tengas una clave correcta, el ataque termina de forma automática.
- La clave se guarda y los clientes se conectaran a su AP original.

Como puedes comprobar su uso es simple. Recuerda que hackear keys wifi que no sean tuyas, puede estar penado por ley. Tu sabrás lo que haces.

Nota de los autores de Fluxion:

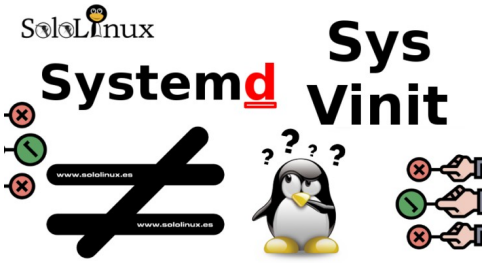
El uso de Fluxion para atacar infraestructuras sin previo consentimiento mutuo podría considerarse una actividad ilegal y sus autores / desarrolladores lo desaconsejan. Es responsabilidad del usuario final obedecer todas las leyes locales, estatales y federales aplicables. Los autores no asumen ninguna responsabilidad y no son responsables del mal uso o daño causado por este programa.

Diferencias entre Systemd y SysVinit (SysV)

Systemd es el sistema de inicio y administración del sistema, que han adoptado la mayoría de distribuciones Linux. Más moderno, más complejo, y quizás con un excesivo control sobre el sistema generó una gran polémica en su implantación.

Excelentes distribuciones se negaron en rotundo a usarlo, como por ejemplo ALT Linux, Gentoo, Knoopix, PCLinux OS. En otras como Debian, los usuarios disconformes crearon un derivado, Devuan.

Lo que está claro, es que le pese a quien le pese Systemd ganó la batalla sobre el administrador tradicional de init, SysVinit. Systemd es compatible con los scripts de inicio SysV y LSB, además funciona como reemplazo directo de sysvinit.



Diferencias entre Systemd y SysVinit

Systemd es el primer proceso iniciado por el kernel y siempre tendrá el **pid 1**. **Fedora 15** fue la primera distribución linux que adoptó systemd, poco a poco todas fueron en cascada (creo que la última grande fue **CentOS 7**).

Esta herramienta en línea de comandos administra todos los demonios, también los servicios systemd que nos permiten iniciar, reiniciar, detener, habilitar, deshabilitar, recargar y muchos más. Además, es importante saber que **Systemd** no usa **scripts bash** para manejar el sistema como SysVinit, utiliza unos archivos llamados «.**service**» que se suponen más rápidos. También destacamos que **systemd** clasifica todos los demonios en **cgroups**.

Qué es SysVinit

SysVinit, más conocido como **SysV**, es uno de los primeros sistemas de inicio para sistemas **UNIX/Linux**. El primer proceso que inicia el kernel es **init**, y se mantiene activo mientras la máquina siga en funcionamiento, es el proceso principal.

La gran mayoría de distros Linux utilizaban SysV, y aunque se crearon otras alternativas como **Service Managament, Upstart**, etc, ninguna llegó a implantarse en masa. Hasta que llegó Systemd y fue adoptada por la gran mayoría de distros, debemos reconocer que el SysVinit actual es una herramienta en desuso.

Qué es Systemd

Systemd es la herramienta init y administradora del sistema, que usan como estándar las distribuciones linux. Permite manejar todo el sistema systemd.

Principales características de Systemd

- Systemd ofrece una paralelización muy potente.
- Hace uso de la activación de socket y D-Bus para iniciar los servicios.
- Soporta el inicio de los demonios bajo demanda.
- Controla los procesos que utilizan el cgroups de Linux.
- Permite crear instantáneas y posterior restauración del estado del sistema.
- Maneja los puntos de montaje y desmontaje automáticamente.
- Implanta una lógica de control del servicio basada en dependencias transaccionales.

Comandos de Systemd y SysVinit

Vemos las diferencias de los comandos más utilizados por los dos sistemas, al servicio lo denominamos ejemplo.

Descripción	Comando systemd	Comando SysVinit
Iniciar un servicio	systemctl start ejemplo	service ejemplo start
Detener un servicio	systemctl stop ejemplo	service ejemplo stop
Reiniciar un servicio	systemctl restart ejemplo	service ejemplo restart
Recargar un servicio	systemctl reload ejemplo	service ejemplo reload
Reiniciar si el servicio se está ejecutando	systemctl condrestart ejemplo	service ejemplo condrestart
Verificar si el servicio se está ejecutando	systemctl status ejemplo	service ejemplo status
Habilitar servicio al iniciar el sistema	systemctl enable ejemplo	chkconfig ejemplo on
Deshabilitar servicio al iniciar el sistema	systemctl disable ejemplo	chkconfig ejemplo off
Verificar si el servicio iniciará con el sistema	systemctl is-enabled ejemplo	chkconfig ejemplo -list
Lista de servicios habilitados y deshabilitados al iniciar el sistema (con niveles de ejecución)	systemctl list-unit-files –type=service	chkconfig
Recargar la herramienta después de modificaciones	systemctl daemon-reload	chkconfig ejemplo -add

Diferencia entre Targets y Runlevels
Systemd usa el concepto Targets (objetivos), que tiene un propósito muy parecido al de los Runlevels (niveles de ejecución) de SysV, pero con una forma de operar diferente. En SysVinit se interpretan con números enteros, y en Systemd se especifica el propósito.

Descripción	Comando systemd	Comando SysVinit
Detener el sistema	runlevel0.target, poweroff.target, systemctl halt	0, halt
Modo usuario único	runlevel1.target, rescue.target	1, S, single
Multiusuario	runlevel2.target, multi-user.target	2
Multiusuario con red	runlevel3.target, multi-user.target	3
Experimental (No User)	runlevel4.target, multi-user.target	4
Multiusuario con gráficos y red	runlevel5.target, graphical.target	5
Reiniciar el sistema	runlevel6.target, reboot.target, systemctl reboot	6, reboot
Emergency shell	emergency.target	emergency

Otros comandos de Systemd
Como ultimo apunte del articulo es evidente que systemd es el sistema más extendido, por lo tanto siempre es conveniente conocer algunos comandos extra, seguro que te serán útiles.

Uso del comando	Comando systemd
Manejar el nombre del host	hostnamectl
Herramienta de control de los demonios	teamdctl
Fecha y hora del sistema	timedatectl
Registro de uso de systemd	journalctl
Tiempo de inicio del sistema	systemd-analyze / systemd-analyze time
Tiempo en iniciar cada servicio con el sistema	systemd-analyze blame
Eliminar procesos de un servicio	systemctl kill ejemplo
Listar servicios que se están ejecutando	systemctl
Estado de systemd	systemctl status
Ejecutar systemd en una máquina remota	systemctl status ejemplo -H usuario@hostremoto

Diferencias entre MySQL y MariaDB

Al mencionar **MySQL y MariaDB,** hablamos de los sistemas de gestión de bases de datos más usados a nivel mundial. La competencia es sana siempre que sea constructiva, pero no siempre fue así, o tal vez si, lo explicamos un poco.

Ulf Michael Widenius (conocido como Monty), lanzó en 1996 junto a otros colaboradores la primera versión de MySQL. Hablamos del primer sistema de gestión de bases de datos de código abierto, puedes imaginar que supuso un bombazo con rotundo éxito a nivel mundial.

Poco tardaron en llover las ofertas de compra, pero en MySQL imponen sus condiciones. En 2008 MySQL se formaliza su venta a **Sun Microsystems** (1.000 millones de dólares), pero se obliga a Sun a comprometerse sobre la comercialización de MySQL, siempre será open source. Además, Monty y su equipo pasarán a formar parte de Sun Microsystem como responsables del desarrollo del sistema de gestión de bases de datos. Estamos en 2008.

Poco duró la feliz alianza, muy poco. Apenas unos meses después, en enero de 2009, el gigante **Oracle Corporation** se hace con **Sun Microsystem** a cambio de 5.750 millones de dólares, al día siguiente **Monty** y su equipo de desarrolladores se marchan para montar su propia empresa.

Ríos de tinta se han escrito sobre el porqué se marcharon de la nueva adquisición de Oracle, yo creo que es evidente. **Oracle Corporation** pagó un sobrecoste del 42% por acción del valor real, algo realmente sospechoso e inusual. El nuevo propietario tampoco quiso comprometerse sobre MySQL como 100% open, a la vista lo tenemos ya que actualmente las versiones mejoradas de MySQL tienen un alto coste.

Mi opinión personal: Oracle no solo se quitó de un plumazo una competencia real, sino que se quedó con los desarrollos del sistema de gestión de bases de datos orientados a internet (sitios web). Poco después de la adquisición, Oracle lanzó la nueva versión de su base de datos Oracle con propiedades integradas específicas para sitios web. En la siguiente agregaron más de 400 funciones nuevas, que como denunció públicamente **Ulf Michael** provenían de MySQL y sus proyectos futuros.

Monty y su equipo de desarrolladores crean una nueva empresa, y nace **MariaDB**, una bifurcación de MySQL de alto rendimiento en la que confían la gran mayoría de **distribuciones linux** actuales. Los creadores originales de MySQL expusieron sus motivos del porqué nace MariaDB.

- Los principales creadores de MySQL deben permanecer juntos, es la única forma posible de mantener un sistema de gestión de bases de datos, excelente.
- Fomentar la continuidad del desarrollo por parte de la comunidad.
- Garantizar que siempre esté disponible MySQL de forma gratuita.

Una vez conocemos un poco de la historia de MySQL, vemos sus principales diferencias.

Diferencias entre MySQL y MariaDB
Quien usa MariaDB y MySQL
Vemos algunos de los principales clientes de los dos sistemas de bases de datos.

MariaDB
Grandes corporaciones y los gigantes de linux utilizan MariaDB, podemos señalar las más conocidas: Google, Craigslist, Wikipedia, archlinux, RedHat, CentOS, Fedora, Suse, Ubuntu, AWS, 1and1, BlaBlaCart, Nokia, Samsung, y muchos más.

MySQL
Muchas de las grandes empresas que usan MySQL, no tienen fácil su migración a otro sistema. La lista es larga: GitHub, US Navy, NASA, Tesla, Netflix, WeChat, Facebook, Zendesk, Twitter, Zappos, YouTube, Spotify, y muchos más.

Compatibilidad entre MySQL y MariaDB
MySQL: MySQL es un sistema de gestión de bases de datos relacionales (RDBMS). Al igual que otros sistema similares usa tablas, restricciones, disparadores, roles, procedimientos almacenados y vistas como componentes centrales de trabajo. Cada tabla se compone de filas, y cada fila contiene un mismo conjunto de columnas. MySQL utiliza claves primarias para identificar cada fila (llamado registro) de la tabla, y claves externas para verificar la integridad referencial de dos tablas relacionadas.

MariaDB: Como MariaDB es una bifurcación de MySQL, la estructura de la base de datos y los índices son los mismos. Esto permite migrar de MySQL a MariaDB sin tener que modificar nuestras aplicaciones, los datos y estructuras serán los mismos.

La conclusión es:
- Los archivos de configuración de datos y las tablas son compatibles.
- Las estructuras, protocolos y las API de cliente son las mismas.
- Los conectores MySQL también trabajan con MariaDB.

El equipo de desarrollo de MariaDB intenta mantener una compatibilidad directa, mensualmente se compara y combina el código MariaDB con el código MySQL. Incluso la mayoría de comandos son los mismos, como `mysqldump` o `mysqladmin`. Aun así, existen diferencias entre MariaDB y MySQL que podrían llegar a causar problemas de compatibilidad mínimos.

La migración de MySQL a MariaDB por parte de potencias empresariales como, Samsung o Google, no sentó nada bien en el seno de Oracle. Ahora mismo intentan diferenciarse de forma gradual, y un buen ejemplo lo encontramos en el diccionario de datos internos que se desarrolló para MySQL 8, que altera la forma en que los metadatos se guardan y utilizan. MariaDB no tiene esa función, y esto puede ser el principio del fin de la compatibilidad a nivel de archivo de datos entre MySQL y MariaDB.

Rendimiento y los índices
Es evidente que el sistema de índices mejoran el rendimiento de la base de datos, ya que permite al servidor localizar y recuperar filas de manera mucho más rápida. Pero ojo, los índices pueden llegar a sobrecargar el servidor de bases de datos, se deben usar con prudencia.

A pesar de lo dicho, el índice es indispensable. Si no tuviéramos índice, una búsqueda comenzaría por la primera fila de una tabla y correlativamente seguiría buscando hasta encontrar el objeto. El tiempo de espera se podría demorar en exceso.

Como norma general los índices MySQL y MariaDB (PRIMARY KEY, UNIQUE, INDEX y FULLTEXT) se guardan como **B-trees** (**Árbol-B**). Existen excepciones como índices para tipos de datos espaciales, estos usan **Árboles-R**. También se admiten índices hash, y el motor InnoDB que utiliza listas invertidas para los índices FULLTEXT.

Diferencias en la sintaxis
Las consultas son exactamente las mismas, en este apartado no tenemos nada que reprochar. Como ejemplo vemos cómo seleccionar registros de la tabla clientes.

```
# MariaDB
SELECT *
FROM clientes;

# MySQL
SELECT *
FROM clientes;
```

Sistemas compatibles con MariaDB y MySQL
Dónde y cómo instalar MySQL
El sistema MySQL está escrito en C y C ++, y tiene binarios para los sistemas: Microsoft Windows, OS X, Linux, AIX, BSDi, FreeBSD, HP-UX, IRIX, NetBSD, Novell Netware y otros.

Puedes **descargar MySQL** desde su página oficial de descargas. Existe mucha documentación e instrucciones para los sistemas operativos más utilizados, puedes descargarlos aquí.

Dónde y cómo instalar MariaDB
MariaDB está escrito en C, C ++, Bash y Perl, y tiene soporte para los siguientes sistemas: Microsoft Windows, Linux, OS X, FreeBSD, OpenBSD, Solaris, y muchos más.

Puedes descargar MariaDB desde su página oficial, pero al ser un sistema tan extendido en la comunidad linux, seguro que la encontraras en los repositorios de tu distribución linux. Su documentación oficial es amplia, pero mi recomendación es que acudas a la comunidad de tu distro, seguro que podrás aclarar tus dudas, ademas y no menos importante en tu idioma materno.

Recuerda que MariaDB se diseño para ser el reemplazo comunitario de MySQL, el proceso de desinstalar MySQL e instalar MariaDB es bastante simple. Nunca olvides que después de ejecutar la operación debes usar `mysql_upgrade`.

Agrupación y replicación
Qué es la agrupación
La agrupación en clúster de bases de datos, es el uso de almacenamiento compartido y más servidores front-end. Los servers front-end comparten una dirección IP y el nombre de la red del clúster, que los usuarios utilizan. Se calcula cual de ellos atiende la solicitud de clientes actual.

Qué es la replicación
La replicación es el proceso que nos permite tener varias copias generadas automáticamente, de las bases de datos. Se conocen como «maestras» y «esclavas» y sus beneficios son importantes.

- Mejor soporte.
- Mejora considerable del rendimiento al tener la carga distribuida.
- Permite operar en una bases de datos esclava, sin reducir el rendimiento de la base de datos principal (cargas intensivas o de larga duración).

MySQL
La replicación en MySQL es asincrónica y unidireccional, un servidor actúa como maestro y otros como esclavos. Permite replicar todas las bases de datos, bases de datos específicas, e incluso una sola tabla.

MySQL Cluster proporciona el soporte para los clústeres compartidos, incluye la auto-fragmentación para un correcto mantenimiento del sistema de gestión MySQL.

MariaDB
MariaDB también ofrece replicación maestro-maestro y maestro-esclavo. A partir de MariaDB 10.1, Galera se incluye como clúster en MariaDB. Habilitar la agrupación es muy fácil, tan solo debemos configurar unos pocos parámetros.

Conectores de bases de datos
Los conectores son estándares de acceso, que tienen como objetivo hacer posible el acceso a los datos de una base de datos.

Conectores de MySQL
MySQL ofrece una variedad de conectores de bases de datos, se incluyen: C, C ++, Delphi, Perl, Java, Lua, .NET, Node.js, Python, PHP, Lisp, Go, R, D y Erlang.

Conectores de MariaDB
MariaDB también tiene una gran variedad de conectores, como: ADO.NET, C, C ++, D, Java, JavaScript, ODBC, Perl, PHP, Python, Ruby y Visual Studio.

De donde viene el nombre de MySQL, MariaDB y MaxDB
Curiosa anécdota, jajaj. Todo es idea del creador original, está claro que hablamos de **Michael Widenius** (Monty) y de sus hijos.

- **MySQL**: la primera hija de Monty se llama «**My**», un curioso nombre tailandés.
- **MaxDB**: **MaxDB** es otro sistema de gestión creado por el mismo autor, el nombre se debe a su hijo «**Max**».
- **MariaDB**: así se llama la hija de Monty (de su segundo matrimonio), «**Maria**».

Microsoft Defender Advanced Threat Protection llega a linux

Microsoft Defender Advanced Threat Protection, más conocido como **Microsoft Defender ATP** aterriza en Linux, y ya empiezan a ser bastante molestos los intentos de incursión en nuestros **sistemas Unix**.

Disponible en modo vista previa pública, permite a los administradores y profesionales de seguridad probar el producto en seis distribuciones de Linux diferentes: RHEL 7+, CentOS 7+, Ubuntu 16 LTS o superior, SUSE 12+, Debian 9+ y Oracle EL 7.

Pero ojo!!!, un detalle importante. **Si quieres probar Microsoft Defender ATP**, **debes ser usuario registrado de Microsoft**. Realmente me parece un absurdo.

Al usar el cliente de punto final ATP de Microsoft Defender, los administradores de Linux tendrán acceso al antivirus en línea de comandos que envía automáticamente cualquier amenaza detectada al Centro de seguridad de Microsoft Defender.

```
cloudAutomaticSampleSubmission          : true
cloudEnabled                            : true
definitionsUpdatedMinutesAgo            : 2
definitionsVersion                      : 78669
edrDeviceTags                           : []
edrEarlyPreviewEnabled                  : "disabled"
edrMachineId                            : "fd04696d0befc83549d1e1cdb3e3f83a3b473e35"
healthy                                 : true
licensed                                : true
logLevel                                : "info"
machineGuid                             : null
realTimeProtectionAvailable             : true
realTimeProtectionEnabled               : true
versionEngine                           : "100.74.77"
```

Microsoft Defender ATP no se conforma sólo con Linux, tienen previsto versiones para iOS y Android, que se lanzarán a lo largo del año en curso, 2020.

Como siempre digo, en mis maquinas no, ni verlo.

Los mejores linux del 2020 para ejecutar desde un USB

Si de algo podemos presumir en **Gnu/linux**, es de la gran cantidad de distribuciones que tenemos a nuestra disposición. Muchas son similares, pero cada una tiene su particularidad.

Por necesidades laborales debía hacer uso de una distro linux directamente desde un pendrive usb, o sea… lo que se conoce como una **Live USB portable y persistente**.

Durante varios días estuve probando montones de distros, buscaba la que mejor se adaptara a mis necesidades. Así que, como ahora mismo tengo este tema aún en mente, creo que es bueno compartirlo e intentar ayudar a otros usuarios que talvez tengan la misma necesidad.

Las explicaciones que aporto son básicas, pero tu, debes tener en cuenta para que la necesitas. Es lo mismo una distro creada para reparar otros sistemas, que un sistema portable completo. También debes tener en cuenta su peso, puede ser fundamental.

Aun teniendo en cuenta el tamaño, todas las distribuciones linux propuestas, están pensadas para ejecutarse sin excesivos recursos de hardware. Algunas son complejas y otras minimalistas, tú decides.

Los mejores linux del 2020 para un USB

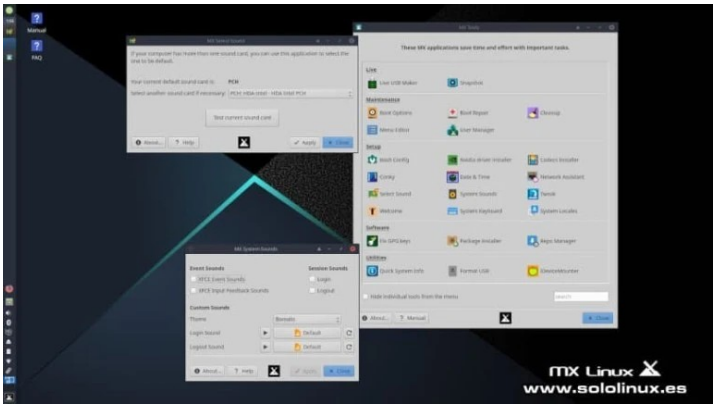
Knoppix

Cuando hablamos de **Knoppix** hay que quitarse el sombrero, estamos ante un sistema operativo basado en Debian diseñado para ejecutarse directamente desde una unidad USB o DVD, casi dos décadas avalan a esta excelente y completa **distribución linux**.

El tamaño de la iso (4,3 Gb) puede asustar a más de uno, pero todo lo contrario. Gracias a su sistema de descompresión al vuelo, opera de una forma rápida y eficiente. Si lo que buscas es un linux super completo, Knoppix es tu distro, tiene todo tipo de aplicaciones.

Puedes **descargar Knoppix vía torrent desde este** enlace.

MX Linux



MX Linux es una distribución Linux basada en **antiX**, que a su vez provenía de **MEPIS** (Mepis bebía de la fuente de Debian). Diseñada para funcionar de manera correcta tanto en sistemas antiguos como otros más modernos, debemos reconocer que entre todos hicieron un trabajo excelente.

Muy fácil de configurar, es lo suficientemente simple como para que los usuarios menos experimentados, la puedan utilizar sin problemas. **MX Linux** es también un linux potente y seguro, que permite trabajar desde una memoria usb seleccionando el modo live persistente en el menú de arranque.

Puedes descargar **MX Linux (1,6 Gb)** desde su página oficial.

Peppermint OS

Peppermint OS es ligero, rápido, y totalmente personalizable. Basado en Lubuntu está diseñado para integrarse con aplicaciones basadas en web y servicios en la nube.

Tiene una particularidad un tanto especial, y es que combina la funciones y el menú de aplicaciones de XFCE, con el entorno de escritorio LXDE. De esta forma nos encontramos con un sistema elegante, bonito y potente. Viene con todas las aplicaciones necesarias para que sea tu estación de trabajo portátil.

Puedes **descargar Peppermint OS (1,5 Gb)** desde su página oficial (deberás aceptar la session, incomprensiblemente no ofrece https).

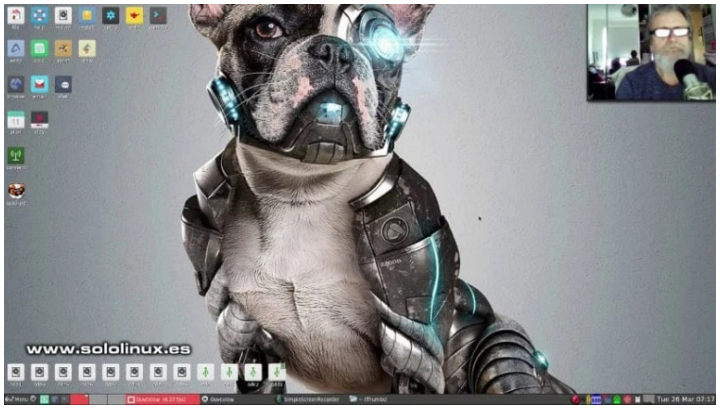
Kali Linux

Que vamos a decir a estas alturas de **Kali Linux**, todos la conocemos. Esta distribución linux basada en Debian testing, tiene un peso de 2.7 Gb que no le impide ser muy rápida. Principalmente desarrollada para pruebas de penetración y análisis forense digital . Viene con más de 300 herramientas creadas por un grupo de expertos en seguridad.

Diseñada para ejecutarse en una unidad flash con total soltura, tambien resulta valida para otros menesteres más comunes. Si eres fan de la seguridad cibernética / forense, **Kali Linux** es de las mejores.

Si quieres descargar **Kali Linux**, hazlo desde su página oficial,

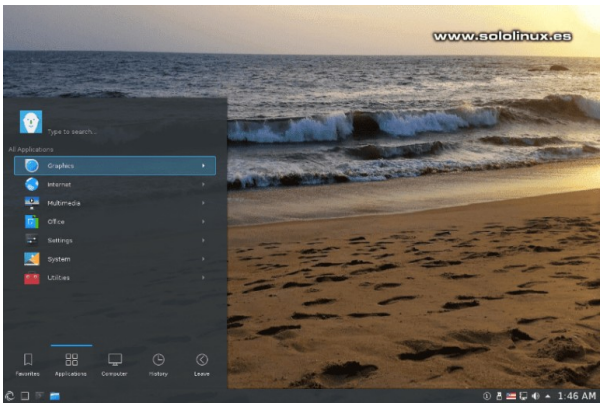
Puppy Linux



Si hablamos de distros ligeras, completas y que se ejecuten desde un pendrive. **Puppy Linux** debe aparecer en la lista.

Con una interfaz amigable, esta distro puede ejecutarse exclusivamente desde la ram, tan solo pesa 350Mb. Puedes **descargar Puppy Linux** en sus diferentes versiones desde su página oficial.

Porteus



Porteus, es una distribución basada en **Slackware** que en poco menos de 300 MB, nos ofrece de todo. Su escritorio LXDE es capaz de iniciar en menos de 15 segundos, además viene con un sistema de módulos que permite instalar o desinstalar cualquier cosa con tan solo un clic.

Tiene soporte para muchos idiomas, y una comunidad bastante activa. Puedes descargar porteus desde este enlace.

Slax

Conocida como el **sistema operativo de bolsillo**, **Slax** es una distribución LiveCD/USB basada en Debian. Tiene un escritorio minimalista (**Fluxbox**) que la convierte en super ligera. Su funcionamiento es excelente y con su tamaño reducido (260 Mb), solo necesita para un correcto funcionamiento de 256mg de ram.

Existen versiones de 32 y 64 bits, y sobre ella se inspiraron otras grandes distribuciones como la española **WifiSlax**, referente en pruebas de penetración y análisis de redes. Puedes descargar Slax desde su página oficial

Otras distribuciones…
Existen otras opciones excelentes, como…

- **Tiny Core Linux**
- **SliTaz**
- **Ubuntu GamePack**
- **FatDog64**

Saber a qué proceso corresponde un PID

Es algo común que algunas herramientas nos indiquen el número de PID (identificador de proceso), pero no el proceso al que corresponde la herramienta o aplicación.

Otras como » **el comando ps**», nos informan del PID y del proceso de la aplicación, por ejemplo:

```
sololinux ~ # ps -e
  PID TTY          TIME CMD
    1 ?        00:00:05 systemd
    2 ?        00:00:00 kthreadd
    4 ?        00:00:00 kworker/0:0H
    6 ?        00:00:00 mm_percpu_wq
    7 ?        00:00:00 ksoftirqd/0
    8 ?        00:00:02 rcu_sched
    9 ?        00:00:00 rcu_bh
   10 ?        00:00:00 migration/0
   11 ?        00:00:00 watchdog/0
   12 ?        00:00:00 cpuhp/0
   13 ?        00:00:00 cpuhp/1
    etc, etc...
```

La lista puede ser interminable y revisar línea por línea no es cómodo. Por suerte el «comando ps» junto con «grep» nos ayudará en esta operación. Su sintaxis es la siguiente:

```
ps -p PID -o comm=
```

En nuestro ejemplo de uso detectamos que el proceso 1900 tiene un elevado consumo, vamos a ver a qué corresponde.

```
ps -p 1900 -o comm=
```

```
sololinux ~ # ps -p 1900 -o comm=
chromium-browse
sololinux ~ #
```

Bien, ya sabemos a quién corresponde; al navegador Chromium.

En el caso contrario, o sea si queremos saber el pid del proceso (herramienta o aplicación). Ejecutamos lo siguiente:

```
ps aux | grep chromium
```

En Cromium el listado de procesos puede ser interminable, todo depende de las ventanas abiertas, extensiones instaladas, etc…

Cómo listar las tareas cron programadas en linux

Cron es un demonio que nos permite programar la ejecución de tareas a intervalos definidos. Más conocidas como **tareas cron**, las podemos planificar por minuto, hora, día del mes, mes, día de la semana y cualquiera de sus combinaciones posibles.

Las **tareas Cron** son especialmente útiles en operaciones relativas al mantenimiento del sistema. Por ejemplo… automatizar las **copias de seguridad**, enviar correos, verificar actualizaciones, limpiar el sistema, etc.

Más o menos todos sabemos como operar con estas tareas, lo que tal vez no conoces es cómo listar las **tareas cron** que tienes **programadas en linux**. Hoy vemos como imprimir en pantalla esos listados.

Listar las tareas cron programadas en linux
La ubicación y forma de visualizar los archivos, puede variar dependiendo del sistema y del tipo de usuario. Así que vemos la formula para los usuarios, y como ver las tareas propietarias del sistema.

Listar las tareas cron de usuario
La ubicación de los archivos de tareas es diferente si es un derivado de RHEL o de Debian, las vemos.

Ubicación en Rhel, CentOS, Fedora y derivados

```
/var/spool/cron
```

Ubicación en Debian, Ubuntu, Linux Mint y derivados

```
/var/spool/cron/crontabs
```

Listar tareas cron de usuario
Para poder visualizar todas las tareas cron del usuario actual, ejecutamos lo siguiente.

```
crontab -l
```

Si aplicamos la opción «-u», podemos listar las tareas de otro usuario. Pero debes tener en cuenta que los archivos son propiedad del usuario que los creo, y solo si eres root o tienes privilegios sudo podrás ver las tareas del resto de usuarios.

```
sudo crontab -u usuario -l
```

También puedes saber que usuarios del sistema han creado sus propias tareas cron, para ello nos aprovechamos de spool.

```
 sudo ls -1 /var/spool/cron

# o

 sudo ls -1
/var/spool/cron/crontabs
```

```
SHELL=/bin/bash
PATH=/sbin:/bin:/usr/sbin:/usr/bin
MAILTO=root

# For details see man 4 crontabs

# Example of job definition:
# .---------------- minute (0 - 59)
# |  .------------- hour (0 - 23)
# |  |  .---------- day of month (1 - 31)
# |  |  |  .------- month (1 - 12) OR jan,feb,mar,apr ...
# |  |  |  |  .---- day of week (0 - 6) (Sunday=0 or 7) OR sun,mon,tue,wed,thu,fri,sat
# |  |  |  |  |
# *  *  *  *  * user-name  command to be executed
```


Listar las tareas cron del sistema
Las tareas cron del sistema las podemos encontrar en «/etc/crontab», y en el directorio «/etc/cron.d». Para imprimir todas en pantalla usamos el **comando cat**.

```
cat /etc/crontab /etc/cron.d/*
```

Cron permite otros tipos de programación «predefinidos» que puedes encontrar en «/etc»:

- cron.daily
- cron.hourly
- cron.monthly
- cron.weekly

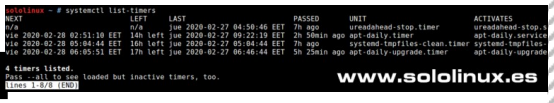
Por ejemplo, para ver las tareas programadas semanalmente puedes ejecutar el siguiente comando.

```
ls -l /etc/cron.weekly/
```

Si no hay ninguna salida es, porque no existe ninguna tarea cron semanal.

Temporizador de Systemd
Si tu distribución linux usa Systemd como sistema init, también tienes unos archivos temporizadores que se utilizan como alternativa al demonio cron estándar. Normalmente son propiedades del sistema, pero puedes verlos al ejecutar el siguiente comando.

```
systemctl list-timers
```



Comparar tamaños de cadenas y enteros con un script bash

A modo educativo, hoy vamos a ver un script bash que nos compara el tamaño de dos cadenas de caracteres alfanuméricos, o valores de **números enteros**.

Como es el primer script de ese tipo, tampoco nos vamos a complicar mucho. Nosotros insertamos dos cadenas o valores enteros, y el script bash nos dirá si es mayor, menor, o son iguales.

Para lograr nuestro objetivo usamos tres operadores básicos:

- **== / =** : equivale a igual.
- **!=** : equivale a diferente.
- **<** : equivale a menor.
- **>** : equivale a mayor.

Insisto que este script es solo a modo educacional, no es válido para producción. Pero aún siendo así, al introducir dos cadenas alfanuméricas con distinto número de caracteres te dirá cual es mayor o menor. Si solo insertas números enteros, el resultado será el valor real de mayor o menor cantidad.

Comparar tamaños de cadenas y números enteros
Creamos el script.

```
nano compara.sh
```

Copia y pega lo siguiente.

```
#!/bin/bash
# Comparar valores y cadenas

 while true; do

    # Recopilar los valores VAR1 y VAR2
    read -r -p "Introduce el primer valor: " VAR1
    read -r -p "Ahora el valor a comparar: " VAR2

    # Comprobar si $VAR1 es menor que $VAR2
    if [[ "$VAR1" < "$VAR2" ]];then
       echo "$VAR1 es menor que $VAR2"
    # Comprobar si $VAR1 es mayor que $VAR2
    elif [[ "$VAR1" > "$VAR2" ]];then
       echo "$VAR1 es mayor que $VAR2"
    # Comprobar si $VAR1 es igual que $VAR2
    elif [[ "$VAR1" == "$VAR2" ]];then
       echo "$VAR1 es igual a $VAR2"
    # Comprobar si $VAR1 es diferente a $VAR2
    elif [[ "$VAR1" != "$VAR2" ]];then
       echo "$VAR1 es diferente a $VAR2"
    else
    # Si se produce algun error
       echo "Los valores no pueden ser comparados"
   fi
done
```

Guarda el archivo y cierra el editor.

Ahora le concedemos los permisos necesarios.

```
chmod u+x compara.sh
```

Lo ejecutas…

```
 ./compara.sh

o

 bash compara.sh
```

Nosotros insertamos unos valores de ejemplo, los vemos:

```
sololinux sergio # bash compara.sh
```

Igual valor

```
Introduce el primer valor: 123456
Ahora el valor a comparar: 123456
123456 es igual a 123456
```

Menor valor

```
Introduce el primer valor: 123456
Ahora el valor a comparar: 123457
123456 es menor que 123457
```

Mayor valor

```
Introduce el primer valor: 123457
Ahora el valor a comparar: 123456
123457 es mayor que 123456
```

Menor tamaño

```
Introduce el primer valor: zxcvb
Ahora el valor a comparar: zxcvbnm
zxcvb es menor que zxcvbnm
```

Mayor tamaño

```
Introduce el primer valor:
asdfghjkl
Ahora el valor a comparar: asdfgh
asdfghjkl es mayor que asdfgh
```

Recuerda que este script es tan solo a modo educativo, tan solo es una introducción a los operadores que se pueden utilizar en los scripts bash.

Síguenos en las Redes:

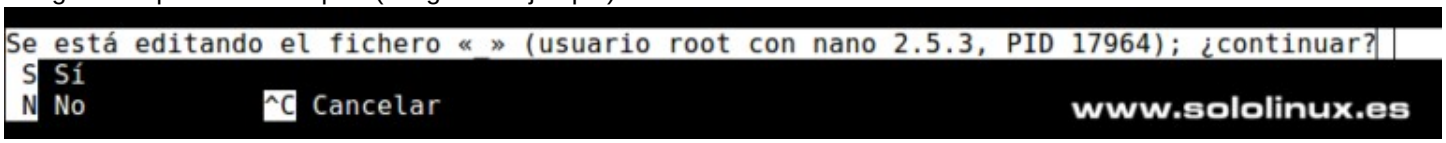
Verificar el sistema de archivos en el proceso de arranque

Todos sabemos que la herramienta por excelencia para **reparar nuestro sistema de archivos, es Fsck.** Lamentablemente cuando nos encontramos con un sistema de archivos corrupto, ya es tarde y no podemos ejecutar **fsck**, debemos recurrir a un **live usb** para iniciar el sistema y poder hacer uso de ella.

Puedes evitar esta situación de manera sencilla; ¿como?, pues muy fácil. En este articulo vamos a configurar el **grub** de inicio para que fuerce la verificación del sistema y lo repare si es necesario, todo durante el proceso de arranque.

Verificar el sistema de archivos en el proceso de arranque
Lo que tenemos que hacer es agregar al Grub dos comandos, el primero verifica el sistema de archivos y el segundo lo repara (si es necesario). Este método ha sido comprobado hoy mismo en tres sistemas, **CentOS 7, Debian 9 y Linux Mint Sylvia**.

Abrimos el archivo de configuración general del **Grub**. Nosotros usamos **nano**, por tanto nos aparece un aviso de seguridad que debes aceptar (imagen de ejemplo).



Para editar el archivo en Rhel, CentOS, Fedora y derivados, debes ejecutar el siguiente comando:

```
nano /etc/sysconfig/grub
```

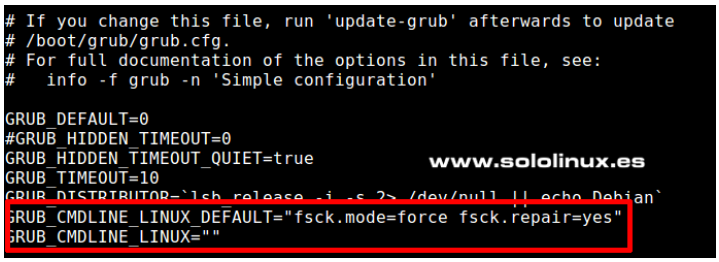
Para editar el archivo en Debian, Ubuntu, Linux Mint y derivados, ejecuta…

```
nano /etc/default/grub
```

Ahora, donde **GRUB_CMDLINE_LINUX** debes agregar los siguientes comandos.

```
fsck.mode=force fsck.repair=yes
```

Guarda el archivo y cierra el editor. Vemos ejemplos en Linux Mint y en CentOS.



Es necesario actualizar el Grub.

```
sudo update-grub
```

Bien, ya lo tienes. Puedes reiniciar el sistema y verificar que se verifica el sistema de archivos. Ojo!!!, si tienes una imagen de inicio (boot splash) no podrás visualizar el proceso, para quitar el **boot splash** sigue las instrucciones de este articulo.

Cómo borrar y formatear dispositivos usb correctamente

Antes de usar una **tarjeta SD** o una **unidad USB** (**pendrive**), conviene **formatear y particionar** correctamente el dispositivo. Normalmente, la mayoría de dispositivos USB vienen preformateadas con el sistema de archivos FAT y no lo necesitan. Sin embargo, a veces nos vemos obligados a ello.

En este articulo veremos como formatar tu dispositivo con la herramienta **parted**, pero antes debes recordar que el formateo es un proceso destructivo, por tanto se borrarán todos los datos que contenga el dispositivo. Haz una copia de seguridad.

Cómo borrar y formatear dispositivos usb
Parted es la herramienta perfecta para crear y administrar tablas de particiones. Viene preinstalado en la mayoría de las **distribuciones de Linux** actuales. Puedes verificar si la tienes instalada con el siguiente comando.

```
parted --version
```

```
sololinux ~ # parted –version
parted (GNU parted) 3.2
Copyright (C) 2014 Free Software Foundation, Inc.
Licencia GPLv3+: GNU GPL versión 3 o superior <http://gnu.org/licenses/gpl.html>
```

Identificar el dispositivo
Lo primero que debemos hacer es identificar el dispositivo.

```
lsblk
```

En nuestro ejemplo podemos comprobar que nuestro pendrive es el disco «sdb».

```
 sololinux ~ # lsblk
 NAME   MAJ:MIN RM   SIZE RO TYPE MOUNTPOINT
 sdb      8:16   1  14,4G  0 disk
├─sdb2   8:18   1    64M  0 part
└─sdb1   8:17   1   641M  0 part /media/sergio/ANARCHYV1010
 sr0     11:0    1  1024M  0 rom
 sda      8:0    0 465,8G  0 disk
├─sda2   8:2    0     1K  0 part
├─sda5   8:5    0   2,9G  0 part [SWAP]
└─sda1   8:1    0 462,9G  0 part /
```

Borrar los datos completamente
Para que los datos sean irrecuperables (puede caer en manos ajenas), es conveniente llenar el disco de ceros como si acabara de salir de fábrica. Este proceso tarda un poco, pero es necesario sobre todo si vas a regalar o prestar el pendrive.

```
sudo dd if=/dev/zero of=/dev/sdb bs=4096 status=progress
```

```
sololinux ~ # sudo dd if=/dev/zero of=/dev/sdb bs=4096 status=progress
529170432 bytes (529 MB, 505 MiB) copied, 27,0057 s, 19,6 MB/s
```

Al concluir, saltara un aviso como que ya no queda más espacio en el dispositivo usb.

dd: error writing ‘/dev/sdb’: No space left on device

Crear una partición y formatear
Los sistemas de archivos más comunes en este tipo de dispositivos, son EXT4 y FAT32. Vemos cómo particionar y formatear nuestro disco de almacenamiento USB en FAT32 y EXT4. El proceso es similar pero no igual.

En FAT32
Creamos la tabla de particiones.

```
sudo parted /dev/sdb --script -- mklabel msdos
```

Ahora una partición Fat32 que ocupe todo el espacio del dispositivo.

```
sudo parted /dev/sdb --script -- mkpart primary fat32 1MiB 100%
```

Una vez tenemos la partición creada, formateamos el usb.

```
sudo mkfs.vfat -F32 /dev/sdb1
```

Nuestro pendrive ya está listo para operar.

En EXT4
Creamos una tabla partición GPT.

```
sudo parted /dev/sdb --script --mklabel gpt
```

Continuamos con un partición EXT4 que ocupe el cien por cien del dispositivo usb.

```
sudo parted /dev/sdb --script -- mkpart primary ext4 0% 100%
```

Solo nos falta formatear.

```
sudo mkfs.ext4 -F /dev/sdb1
```

Una vez concluya el proceso, el dispositivo ya estará listo para su uso.

Nuevo controlador NVIDIA 440.64 lanzado para linux

Como ya comentamos en un articulo anterior, ahora mismo **NVIDIA** colabora y mucho con la comunidad linux. Hoy se anuncia su nuevo controlador 440.64 con soporte predeterminado para el futuro **kernel Linux 5.6**.

NVIDIA 440.64 es el ultimo controlador estable lanzado por Nvidia para la serie **Geforce 440**. Agrega soporte para dos unidades de procesamiento gráfico NVIDIA, GeForce MX330 y GeForce MX350, casi nada.

Entre las correcciones de errores más importantes, podemos destacar que en este controlador se resuelve el problema del modo DPMS (visualización de señalización de administración de energía), que tanto traía de cabeza a muchos usuarios.

Normalmente no solemos anunciar este tipo de novedades en sololinux.es, pero este me parece significativo ya que es la primera vez que NVIDIA se adelanta a la versión final estable del kernel linux. Gracias NVIDIA por colaborar con la comunidad linux (no como otros).

¿Cómo instalar el Driver NVIDIA 440.64?
Puede descargar cualquier controlador de gráficos desde el administrador de paquetes de tu distribución.

Como Nvidia 440.64 es una versión muy nueva, es posible que no la encuentres, entonces la descargas **desde aquí**.

Si utilizas SuSE (Open Suse), lee su instalador específico, aquí.

Una vez descargado el driver, abres la terminal y ejecutas el siguiente comando (desde el directorio donde esta el controlador).

```
sh ./NVIDIA-Linux-x86_64-440.64.run
```

Una vez termine el proceso reinicias el sistema…… y listo.

```
reboot
```

THANKS!

SoloWordpress

SoloLinux

Canales de Telegram: Canal SoloLinux – Canal SoloWordpress

Esta revista es de **distribución gratuita**, si lo consideras oportuno puedes ponerle precio. Tu también puedes ayudar, contamos con la posibilidad de hacer donaciones para la REVISTA, de manera muy simple a través de **PAYPAL**

AYUDANOS A SEGUIR CRECIENDO

PayPal™
Apoyo con una donación