# Encriptación LUKS con CRYPTSETUP

Publicado el El Informatico - 13 de agosto de 2021 -


En éste artículo aprenderás a usar el programa CRYPTSETUP para cifrar tus unidades de disco, memorias USB, e incluso crear contenedores de archivos cifrados. CRYPTSETUP es una aplicación que sólo está disponible en entornos Linux, pero que puede usarse en Windows a traves de **WSL**.

**En éste artículo no se explica cómo instalar Linux en una unidad cifrada**, eso es una operación que se realiza desde el instalador del sistema operativo. Simplemente se explica el funcionamiento de la aplicación CRYPTSETUP para cifrar volúmenes y contenedores de archivos.

## ¿Qué es LUKS?

LUKS (Linux Unified Key Setup) es una especificación para la encriptación de discos duros en Linux, basada una versión mejorada de CRYPTSETUP y usando el subsistema dm-crypt implementado en el kernel.

Gracias a LUKS y a CRYPTSETUP podemos encriptar dispositivos de disco y memorias USB, de modo que en caso de pérdida o robo, o si vamos a viajar al extranjero con un portatil, será mucho más difícil que un posible atacante pueda acceder a nuestros datos más sensibles.

## Instalación

CRYPTSETUP se encuentra disponible en todas las distribuciones estandard de Linux

como Ubuntu, Arch, SuSE, etc. Para instalarlo, basta con instalar el paquete correspondiente a través del gestor de paquetes de tu distribución. Esto también se aplica a WSL.

> **RECUERDA**: En Linux, el prompt de la linea de comandos cambia dependiendo de si eres usuario o superusuario. Así pues, un comando precedido del símbolo $ indica que se puede ejecutar como usuario, mientras que el símbolo # indica que se debe ejecutar como superusuario. Para ejecutar comandos como superusuario, añade el comando 'sudo' al principio del comando. Por ejemplo: sudo apt install cryptsetup.

En Ubuntu:

```
# apt install cryptestup
```

Y en Arch:

```
# pacman -S cryptsetup
```

## Cifrar un volumen entero (unidad de disco, memoria USB...)

Conecta la unidad de disco o memoria USB al ordenador. A continuación, deberás identificar a qué dispositivo se corresponde mediante el comando *lsblk*. Si el dispositivo tiene formato y está montado, lsblk te dirá en qué directorio se encuentra montado el dispositivo.

```
$ lsblk
```

```
NAME        MAJ:MIN RM   SIZE RO TYPE MOUNTPOINTS
sda           8:0    0   1,8T  0 disk
└─sda1        8:1    0   1,8T  0 part
sdb           8:16   0   2,7T  0 disk
├─sdb1        8:17   0   1,2T  0 part
├─sdb2        8:18   0   250G  0 part
└─sdb3        8:19   0   1,2T  0 part /home
sdc           8:32   0   3,6T  0 disk
├─sdc1        8:33   0   128M  0 part
└─sdc2        8:34   0   3,6T  0 part /run/media/enigmatico/DATOS
sdd           8:48   1  28,9G  0 disk
nvme0n1     259:0    0 465,8G  0 disk
├─nvme0n1p1 259:1    0   450M  0 part
├─nvme0n1p2 259:2    0   100M  0 part
├─nvme0n1p3 259:3    0    16M  0 part
├─nvme0n1p4 259:4    0 198,9G  0 part
├─nvme0n1p5 259:5    0   514M  0 part
├─nvme0n1p6 259:6    0  15,3G  0 part [SWAP]
├─nvme0n1p7 259:7    0 249,5G  0 part /var/log
│                                    /var/cache
│                                    /
└─nvme0n1p8 259:8    0     1G  0 part /boot/efi
```

En mi caso se trata de una memoria USB de 32GB que se encuentra en **sdd**, con lo que el directorio del dispositivo se encuentra en **/dev/sdd**. **En el tuyo será distinto**. Es muy importante asegurarte de que has identificado correctamente el dispositivo, **ya que el siguiente paso requiere dar formato de bajo nivel al dispositivo, destruyendo todos los datos en el proceso**. Así que si te equivocas de dispositivo, estarás en apuros.

Para dar formato utilizaremos el comando **dd** de Linux para copiar datos desde /dev/zero, poniendo así la unidad a cero:

```
# dd if=/dev/zero of=<Ruta del dispositivo> bs=<tamaño de bloque> count=
<número de bloques> status=progress
```

En mi caso (no introduzcas éste comando, es un ejemplo), yo usaría el siguiente comando:

```
# dd if=/dev/zero of=/dev/sdd bs=1M count=32768 status=progress
```

El parámetro 'if' se corresponde con el dispositivo de origen desde el cual copiaremos los datos. En éste caso especificamos el dispositivo /dev/zero (en lugar de random) para poner a 0 todos los bytes. 'of' es la salida, el dispositivo al que copiaremos los datos, en éste caso nuestro dispositivo. 'bs' especifica el tamaño del bloque a copiar, en éste caso seran 32768 bloques de 1MB (32768MB = 32GB).

Por último, 'status' indica el tipo de estado que queremos mostrar en la consola. En este caso le indicamos que nos muestre el progreso de la copia, ya que en caso contrario no tendremos ninguna indicación del progreso.

Al ejecutar el comando, dd pondrá a cero nuestra unidad hasta llegar a los 32GB.

```
30990614528 bytes (31 GB, 29 GiB) copied, 1962 s, 15,8 MB/s
```

El siguiente paso es formatear la unidad como un volumen LUKS. Para hacer ésto, usaremos el programa CRYPTSETUP con la opción luksFormat en la unidad que acabamos de poner a cero.

```
# cryptsetup --cipher aes-xts-plain64 --key-size 512 --hash sha512 --
iter-time 1500 -v luksFormat <Ruta del dispositivo>
```

El parámetro 'cipher' te permite especificar la cadena de cifrado para los bloques de datos. Puedes ver una lista de algoritmos disponibles con el siguiente comando:

```
$ cat /proc/crypto
```

Por lo general, para cifrar unidades de disco, lo recomendable es usar aes con xts. El parámetro 'key-size' establece el tamaño de la clave. 'hash' establece el algoritmo usado para generar la clave de salteado usada en la generación de contraseñas. 'iter-time' establece el tiempo mínimo, en milisegundos, deseado para descifrar la unidad. A mayor tiempo, mas complicado será atacar por fuerza bruta el cifrado de la unidad, pero también incrementará el consumo de memoria.

Por último, el parámetro 'v' especifica que queremos mostrar todos los mensajes en la consola (*verbose*).

CRYPTSETUP nos informará de que al realizar ésta operación se eliminarán todos los datos de la unidad. Para confirmar la operación, escribe YES en mayúsculas. A continuación, te pedirá una contraseña para el volúmen. Establece una contraseña segura.

```
WARNING!
========
This will overwrite data on /dev/sdd irrevocably.

Are you sure? (Type 'yes' in capital letters): YES
Enter passphrase for /dev/sdd:
Verify passphrase:
Key slot 0 created.
Command successful.
```

> **CONTRASEÑAS SEGURAS**: Una contraseña segura debe contener como mínimo 12 caracteres, incluir letras, números, y símbolos mezclados, y no incluir palabras o números que sean fáciles de adivinar. La contraseña esta mejor guardada en tu cabeza, sin anotar en ningún lado.

Una vez creado el contenedor LUKS en el dispositivo, montalo con CRYPTSETUP usando el siguiente comando:

```
# cryptsetup luksOpen <Ruta del dispositivo> <Nombre del dispositivo>
```

Ruta del dispositivo es la ruta, y Nombre del dispositivo es el nombre que le quieras dar al dispositivo (Puedes inventartelo). Por ejemplo, en mi caso, usaré el siguiente comando:

```
# cryptsetup luksOpen /dev/sdd device
```

CRYPTSETUP nos pedirá la contraseña del contenedor que hemos especificado antes. Una vez suministrada, en caso de desbloquear correctamente el dispositivo, se montará en /dev/mapper/<nombre del dispositivo>

```
$ ls /dev/mapper
control  device
```

A partir de ahora, todas las operaciones que queramos realizar sobre el contenedor cifrado se harán a traves de /dev/mapper/<nombre del dispositivo>. Ahora que ya tenemos el contenedor cifrado, es hora de darle un formato. Para ello, usaremos el comando **mkfs**. Por ejemplo, si queremos darle formato EXT4:

```
# mkfs.ext4 /dev/mapper/device
```

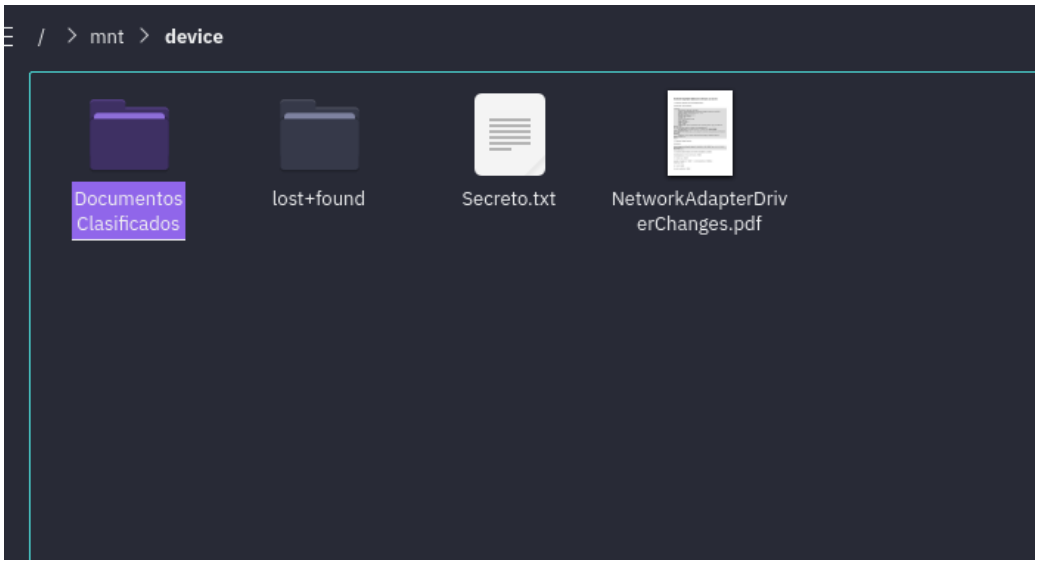
```
mke2fs 1.46.3 (27-Jul-2021)
Creating filesystem with 7564544 4k blocks and 1892352 inodes
Filesystem UUID: 21104085-...
Superblock backups stored on blocks:
        31768, 95304, 163840, 229375, 294812, 819200, 884726, 1605632,
2654208,
        4096000

Allocating group tables: done
Writing inode tables: done
Creating journal (32768 blocks): done
Writing superblocks and filesystem accounting information: done
```

Por último, para poder acceder a nuestro contenedor, lo montamos con el comando **mount**. Recomiendo crear una carpeta destinada a ello en /mnt o /media/<nombre de usuario>. Por ejemplo:

```
# mkdir /mnt/device
# mount /dev/mapper/device /mnt/device
```

A partir de éste momento, ya podemos acceder a nuestro contenedor cifrado desde la carpeta **/mnt/device**.




*Acceso al contenedor cifrado que hemos creado desde Dolphin*

## En caso de no disponer de permisos...

Es posible que no tengas permisos de escritura o incluso lectura al montar el dispositivo. Hay varias posibles causas.

1. **Estas intentando acceder como usuario** a una unidad LUKS con formato EXT o BTRFS. Para solucionar éste problema, basta con asignar permisos a tu usuario para acceder al contenedor.

```
# chmod -R 0700 <punto de montaje>
# chown -R <usuario> <punto de montaje>
```

<Punto de montaje> se refiere a la carpeta donde se encuentra montada la unidad. En mi caso sería /mnt/device.

Recomiendo darle permisos sólo al usuario que vaya a acceder al contenedor, por eso utilizo la máscara 0700 para los permisos, de modo que tanto los grupos como el resto de usuarios no pueden acceder al contenido del contenedor.

2. **Has montado una unidad LUKS con formato FAT o NTFS sin permisos de escritura o lectura**. Desmonta la unidad con umount y despues montala de nuevo otorgando los permisos adecuados:

```
# umount <punto de montaje>
# mount -t vfat /dev/mapper/<Nombre del dispositivo> <punto de montaje> -
o rw,umask=0000
```

## Montar y desmontar la unidad

Si estas usando KDE/Plasma, el gestor de archivos Dolphin puede reconocer la unidad que estas conectando automáticamente. Para montarla manualmente, sigue los siguientes pasos:

1. Usa cryptsetup luksOpen para abrir el contenedor y escribe la contraseña. Ejemplo:

```
# cryptsetup luksOpen <ruta del dispositivo> <nombre del dispositivo>
```

2. Monta el dispositivo:

```
# mount /dev/mapper/<nombre del dispositivo> <punto de montaje>
```

**Para desmontarla**, sigue los siguientes pasos:

1. Cierra todas las aplicaciones que esten usando el putno de montaje y desmonta la unidad:

```
# umount <punto de montaje>
```

2. Cierra el contenedor:

```
# cryptsetup luksClose <nombre del dispositivo>
```

## Creación de archivos cifrados con CRYPTSETUP

Si lo que quieres es crear un archivo de contenedor cifrado, estos son los pasos:

1. Crea un archivo del tamaño deseado.

   Usando el comando DD podemos crear un archivo del tamaño que queramos. Eso si, si estamos usando LUKS2 (que es el caso por defecto), debemos de asignar un tamaño mínimo de aproximadamente unos 100MB, o cambiar el tamaño por defecto de las cabeceras por uno más pequeño. Recuerda que puedes cambiar el nombre del archivo (imagen.img) y del contenedor (imagen) por lo que quieras (es sólo un ejemplo):

```
$ dd if=/dev/zero of=imagen.img bs=100M count=1 status=progress
```

2. Dale formato LUKS al archivo con CRYPTSETUP y establecer una contraseña:

```
$ cryptsetup --cipher aes-xts-plain64 --key-size 512 --hash sha512 --
iter-time 1500 -v luksFormat imagen.img
```

3. Abrir el archivo con CRYPTSETUP:

```
# cryptsetup luksOpen imagen.img imagen
```

4. Darle formato al contenedor

```
# mkfs.ext4 /dev/mapper/imagen
```

5. Montar el contenedor

```
# mount /dev/mapper/imagen /mnt/imagen
```

Para desmontar el dispositivo, desmonta el punto de montaje con umount y cierra el contenedor con cryptsetup luksClose. Para montar y desmontar el contenedor una vez creado, sigue los mismos pasos que con los dispositivos de almacenamiento, pero especificando la ruta del archivo en lugar de un dispositivo.

> **IMPORTANTE**: NO PIERDAS LA CONTRASEÑA del contenedor cifrado. **Si pierdes u olvidas la contraseña, puedes decir adiós a los datos que haya dentro del mismo**.



ANTERIOR

Se acabaron las bromas. A partir de ahora vas a estar constantemente vigilado en todas partes.

SIGUIENTE

Obtén información meteorológica con Python

Actualidad | Android | Básicos | Ciberseguridad | Criptografía | Emulación / Virtualización | FOSS | Hacking | Informática | Internet | Juegos | Opinion | Otros | Personal | Privacidad | Programación | Tecnología | Time Machine | Tutoriales

## RSS

Subscribirse al feed RSS