# ¡Socorro!. ¡Lo he perdido todo!

## Recuperación de archivos borrados o corruptos con Linux

María Dolores Nogueras Atance, Antonio Gómez García

**En un instituto de educación secundaria, al igual que en otros entornos de trabajo que presuponen un continuo intercambio de archivos entre compañeros y equipos, es escalofriantemente posible que se produzca la pérdida de algún documento. Exámenes, documentos de uso administrativo, programaciones didácticas, o lo peor de todo, ¡listas de notas de exámenes corregidos!. Este artículo pretende aportar un punto de partida para que el lector pueda arreglar estropicios de este tipo desde la utilización de nuestro queridísimo GNU/Linux.**

es@lpmagazine.org

Lunes, 8'30 de la mañana. Un centro educativo cualquiera, en un pueblecito mesetario cualquiera. Atravieso la entrada principal con la parsimonia y aburrida seguridad que me otorgan tanto el haber hecho lo mismo todos los días laborables de los últimos seis años como el malhumor que suelen provocar otra noche de escaso sueño y la lluvia que empieza a empapar mi maletín y la cazadora que trata (sin éxito) de proteger mi cuerpo de los ataques de un octubre que quiere hacer honor a su fama de inclemente y antipático. Mis ojos no necesitan esforzarse para adaptarse a la pálida luz de los fluorescentes que el bedel se ha preocupado de conectar casi una hora antes de que el instituto inicie otra jornada de trabajo y estudio. Al fin y al cabo, afuera no había luz suficiente para iluminar la cabeza de un alfiler. Definitivamente, octubre no es mi mes favorito.

Antes incluso de que arranque la cazadora de mi cuerpo enteco y aterido, y liberarme así de la acusadora mirada del conserje que utiliza un ensordecedor silencio para señalarme que (yo también) estoy empapando más aún la entrada al centro con el agua que chorreo, una frase que encierra toda la urgencia de un grito termina de devolverme a la realidad de otra seductora y animosa jornada laboral que empieza para mí...

¡Antonio!. ¡Tienes que ayudarme! ¡Me he metido en un lío gordísimo!.

Sí, señor... ¡Ése soy yo!. Mi nombre es Gómez... Antonio Gómez. No estoy a tu servicio, ni al de su graciosa majestad, pero soy algo así como el encargado de que todo vaya, si no bien, al menos no tan mal como podría ir sin mi intervención, en lo tocante a la parte informática que atañe al funcionamiento diario del instituto.

Soy el encargado TIC (Tecnologías de la Información y la Comunicación) de mi centro educativo.

¡Está bien!. Puede que no haya sido la más airosa de las introducciones en los artículos que hemos tenido el honor de aportar a esta publicación hasta el momento. Pero a Antonio siempre le ha gustado la novela negra, y llevaba meses planeando una introducción como ésta. Sólo necesitaba una oportunidad adecuada para utilizarla de un modo que no pareciera muy forzado, y éste nos ha parecido el tema ideal para utilizarlo.

Al fin y al cabo, ya nos ha ocurrido varias veces, en nuestros centros educativos (por extensión, suponemos que lo mismo ocurrirá en cualquier entorno de trabajo que precise del uso de ordenadores), que alguien nos requiera de modo parecido al que hemos querido dramatizar en las anteriores líneas. El efecto de un troyano (omnipresente entre trabajadores tan asiduos al uso de pendrives como solemos ser los profesores, que además conectamos y desconectamos dichos aparatitos en casi todos los PC's del centro con la alegría suicida que otorga el completo desconocimiento), un borrado accidental, desconectar el pendrive sin desmontarlo primero, e incluso (y no es ninguna leyenda urbana), el hecho de guardar estos dispositivos de almacenamiento cerca de fuentes de energía electromagnética de una cierta intensidad como han demostrado ser algunos modelos específicos de teléfonos móviles (en un bolso, en la cartera, etc...), puede dejar a nuestro anonadado profesor compuesto y sin datos. Y es aquí donde Murphy se muestra más cruel: siempre,

siempre, siempre perdemos el archivo más importante, aquél del que íbamos a hacer una copia de seguridad ("justo ahora lo iba a hacer, Antonio, te lo juro"), y que nos van a pedir mañana a primera hora.

A lo largo del presente artículo, pretendemos demostrar cómo podemos valernos de nuestro amadísimo GNU/Linux, no sólo para enmendar, al menos en parte, el estropicio hecho, sino también para comprender un poco mejor cómo funcionan los dispositivos de almacenamiento de tipo magnético, de modo que la próxima vez estemos mejor preparados para, no sólo arreglar desperfectos del tipo que nos ocupa, sino aún mejor: prevenirlos y evitarlos.

Para ello, empezaremos con una pequeña introducción al funcionamiento lógico de los discos duros y los pendrives, de modo que nos podamos hacer una imagen de conjunto sobre cómo se producen estos problemas de pérdida o de corrupción de datos. A continuación, y entrando ya de lleno en la utilización de herramientas *Open Source*, procederemos a explicar cómo realizar una imagen del soporte averiado, montarla en nuestro sistema operativo, y utilizar algunas herramientas de uso común en el ámbito forense para proceder a la recuperación de datos sobre dicha imagen duplicada.

Pero antes de empezar, permítanos el avispado lector un consejo: el primer paso siempre, repetimos, *siempre*, será avisar al acongojado compañero que puede dar por perdidos los archivos causa de su angustia. Primero, porque de momento es cierto; al fin y al cabo, ha perdido esos archivos, y a nadie más que a sí mismo puede culpar, pues no fue previsor e hizo la correspondiente copia. Si después conseguimos, aunque sea en parte, restaurar parte de la información, sabrá que se debe sobre todo a la suerte, y nos deberá un favor que siempre podremos cobrarnos en el futuro. Y en segundo lugar, porque el miedo que va a pasar durante el par de horas que, como mínimo, nos ocupará este proceso de recuperación, posiblemente le concienciarán sobre futuras medidas de seguridad a tomar a la hora de hacer copias de seguridad frecuentemente, al menos, de los archivos importantes. ¿Están preparados?. Pasen y vean...

## ¿Cómo guardan la información los dispositivos de almacenamiento?

O dicho de otro modo, tanto en discos duros como en unidades de memoria flash USB, ¿por qué causa pueden perder la información que contienen?. Bueno, aunque esto depende del tipo de particionamiento que se haya aplicado, el mecanismo de almacenamiento y borrado de los datos sigue caminos parecidos:

- En el caso de particiones *ntfs (New Technology File System)*, un archivo denominado *mft (Master File Table)* contiene datos (metadatos) relativos a los archivos guardados en el volumen físico, su tamaño, nombre, y lo más importante para nosotros, las direcciones físicas inicial y final de memoria en la que dicha información está contenida.

**Figura 1.** En ningún tipo de dispositivo de almacenamiento se borra la información como tal; simplemente, el espacio físico de la memoria en la que se almacenaba vuelve a constar como disponible para volver a grabar encima.

- En el caso de particiones *fat (File Allocation Table),* el procedimiento es muy similar, salvo que la tabla con los metadatos se denomina, precisamente, *fat*)
- Las particiones de tipo *ext (extended)*, trabajan con un concepto mejorado de estas tablas, consistentes en *i-nodos*, pero que igualmente contienen información referente a cada archivo, nombres, fechas de acceso y modificación, y sobre todo direcciones físicas de memoria en las que éstos empiezan y/o acaban.

Bueno, como explicación de inicio para otros profesores esto tendría que valer. Intentar ampliar esta información (ambigua, mejorable, y, lo admitimos, modificable), equivaldría a redactar otro artículo específico. La cuestión es que si comprendemos esto, podemos comprender cómo se pueden recuperar algunos de estos archivos.

Sea por un accidente (cambios en el voltaje, desconectar una memoria USB sin desmontaje previo, un campo magnético de cierta intensidad, etc...) o por un simple borrado a causa de un error humano, a veces podemos perder acceso a uno, varios archivos, o a la totalidad de la información de nuestro disco duro, memoria USB, etc... Pero eso no quiere decir que dicha información se haya perdido aún, necesariamente.

Cuando borramos un archivo, sea cualquiera la partición que estemos utilizando para organizar nuestro volumen de almacenamiento físico, lo que estamos haciendo es indicar, en la tabla de referencia antes mencionada, que ese espacio de memoria vuelve a estar disponible. Ni más, ni menos. La próxima vez que nuestro sistema operativo necesite almacenar información, sabrá que ese espacio en particular puede ser utilizado para grabar. Que lo utilice o no, ya será cuestión de suerte.

¿Podemos entonces recuperar al menos parte de la información?. La respuesta es que hay, al menos, algunas posibilidades. Eso sí, ya habrá adivinado nuestro astuto lector que cuando estos "accidentes" sucedan, deberemos abstenernos de guardar nuevos archivos hasta que hayamos intentado recuperar dicha información. De lo contrario, nuestras posibilidades disminuirán.

## Ya me he hecho una idea del problema. ¿Por dónde empiezo?.

Bueno, está claro que estamos ante un problema. Y normalmente, un problema con mayúsculas. Así que, parafraseando a la sabiduría popular, "no la liaremos más", para empezar. No realizaremos ninguna operación directamente sobre el volumen (disco duro, memoria USB...) que ha perdido información. Lo que vamos a hacer es un volcado, byte a by-

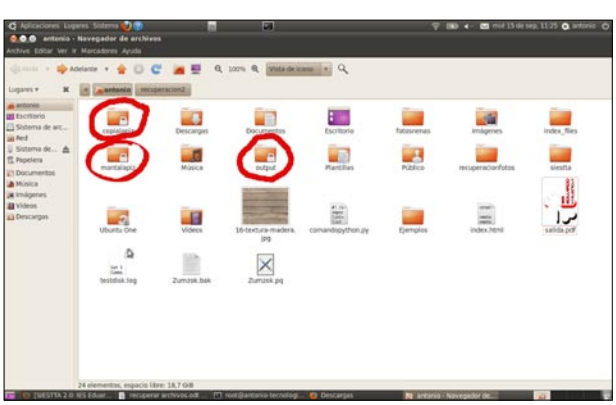

**Figura 2.** No hay que perder de vista el tipo de usuario desde el que estamos trabajando; si lo estamos haciendo como root, las carpetas con las que estamos realizando el proceso (en el ejemplo, copialapiz, montalapiz y output) tendrán los permisos de lectura y escritura reservados.

te, de toda la información recogida en dicha unidad, esté en el estado que esté. Para ello, nos valdremos de la utilísima herramienta *dd (duplicate disk)*, que nos permitirá obtener una imagen de toda esa información. A continuación, *montaremos* (esto es, daremos acceso a nuestro ordenador a la información contenida) dicha imagen en un directorio creado al efecto, con la herramienta *mount*. Para terminar, utilizaremos alguna herramienta como *foremost* (también mencionaremos las posibilidades de *testdisk*) para tratar de arreglar, al menos en parte, el caos ante el que nos encontramos. Un problema añadido que todavía no hemos mencionado, es que la mayor parte de las veces, los archivos recuperados han perdido su nombre en el proceso de borrado y recuperación, y la aplicación utilizada le otorga un nuevo nombre basado en el número de la dirección de memoria en que halló dicho archivo, así que remataremos el trabajo con el uso de *grep* desde la consola BASH; esto nos permitirá

**Listing 1.** Usamos dmesg para identificar la ruta del pendrive que acabamos de conectar

```
root@antonio-tecnologia:/home/antonio# dmesg
[11850.375786] scsi 6:0:0:0: Direct-Access     JetFlash TS2GJFV30        8.07 PQ: 0 ANSI: 2
[11850.377318] sd 6:0:0:0: Attached scsi generic sg2 type 0
[11850.380797] sd 6:0:0:0: [sdb] 4005886 512-byte logical blocks: (2.05 GB/1.90 GiB)
[11850.381829] sd 6:0:0:0: [sdb] Write Protect is off
[11850.381839] sd 6:0:0:0: [sdb] Mode Sense: 03 00 00 00
[11850.381846] sd 6:0:0:0: [sdb] Assuming drive cache: write through
[11850.387382] sd 6:0:0:0: [sdb] Assuming drive cache: write through
[11850.387392]  sdb: sdb1
[11850.390047] sd 6:0:0:0: [sdb] Assuming drive cache: write through
[11850.390055] sd 6:0:0:0: [sdb] Attached SCSI removable disk
```

**Listing 2.** Uso del comando fdisk para obtener una perspectiva de la totalidad de discos detectados por nuestro sistema operativo

```
root@antonio-tecnologia:/home/antonio# fdisk -l
Disco /dev/sda: 250.1 GB, 250059350016 bytes
255 cabezas, 63 sectores/pista, 30401 cilindros
Unidades = cilindros de 16065 * 512 = 8225280 bytes
Tamaño de sector (lógico / físico): 512 bytes / 512 bytes
Tamaño E/S (mínimo/óptimo): 512 bytes / 512 bytes
Identificador de disco: 0xebfef45b

Dispositivo Inicio    Comienzo      Fin      Bloques  Id  Sistema
/dev/sda1   *           1       24316   195312500   83  Linux
/dev/sda2           24316       30401    48883393+   5  Extendida
/dev/sda5           29273       30401     9068661   82  Linux swap / Solaris
/dev/sda6           24316       29272    39814144   83  Linux

Las entradas de la tabla de particiones no están en el orden del disco

Disco /dev/sdb: 2051 MB, 2051013632 bytes
33 cabezas, 63 sectores/pista, 1926 cilindros
Unidades = cilindros de 2079 * 512 = 1064448 bytes
Tamaño de sector (lógico / físico): 512 bytes / 512 bytes
Tamaño E/S (mínimo/óptimo): 512 bytes / 512 bytes
Identificador de disco: 0x1e654572

Dispositivo Inicio    Comienzo      Fin      Bloques  Id  Sistema
/dev/sdb1               1        1927     2002927    6  FAT16
```

**Listing 3.** Creamos un directorio al que volcamos una imagen byte a byte del dispositivo con problemas.

```
root@antonio-tecnologia:/home/antonio# mkdir copialapiz
root@antonio-tecnologia:/home/antonio# dd if=/dev/sdb1 of=/home/antonio/copialapiz/copialapiz.dd
4005886+0 registros de entrada
4005886+0 registros de salida
2051013632 bytes (2,1 GB) copiados, 341,906 s, 6,0 MB/s
```

buscar cadenas de texto dentro de los archivos recuperados, para poder identificarlos mucho más fácilmente.

## Volcado de la unidad física en una imagen. Uso de *dd*

Por si nuestro querido lector aún no lo había intuído, debemos aclarar que todo el trabajo se realiza desde consola. ¡Qué le vamos a hacer!. El usuario no avanzado (en los centros educativos, es fácil que hablemos de más del 90% de la población) es muy remiso al uso de BASH, pero su potencia y versatilidad compensan con creces la ausencia de una ventanita en la pantalla que nos de una información gráfica muchas veces insuficiente y redundante. Así que, ¡manos a la obra!. Tengamos la distribución GNU/Linux que tengamos, el acceso a estas consolas son muy sencillas. Nosotros, en particular, estamos trabajando desde Ubuntu:

```
antonio@antonio-tecnologia:~$ sudo su
[sudo] password for antonio: *****
root@antonio-tecnologia:/home/antonio#
Listado BASH 1: Nos identificamos como root.
```

En primer lugar, nos loguearemos como usuario root. A continuación, utilizaremos la herramienta dd (en caso de no estar instalada en nuestro equipo, un rápido *aptitude install dd* resolverá el problema en breves segundos).

La herramienta *dd* funciona de un modo simple y sencillo: en primer lugar, indico el dispositivo origen de la copia, y a continuación, la carpeta de destino.

¿Cómo averiguar cuál es la ruta del dispositivo a volcar?. En el caso de las memorias FLASH, nosotros utilizamos un sistema tan poco profesional que denota la mucha ignorancia que, como simples profesores de Tecnologías, y no como informáticos profesionales, aún tenemos en este mundillo: Conectamos dicho dispositivo USB y tecleamos *dmesg | tail* en consola; se nos indicará la ruta del último dispositivo conectado que se ha detectado. Así pues, nuestro dispositivo de memoria se ha montado como *sdb*, en la carpeta *dev.* Esto es, estamos en */dev/sdb*. Más concretamente, si atendemos a la tercera línea empezando por el final, vemos que hay una única partición denominada sdb1. Para obtener más información, podemos valernos de *fdisk -l* (como usuario root):

La información sobre las distintas particiones que el sistema detecta nos puede orientar sobre cuál es la ruta de dicho dispositivo, y nos confirmará qué partición estamos buscando. Como es lógico, se puede presumir que en caso de problemas con un disco duro interno, deberíamos disponer de otro dispositivo de almacenamiento, normalmente un segundo disco duro interno, de al menos su mismo tamaño, que esté libre para el volcado de datos. De lo contrario, nos dispondremos de espacio para conseguir la imagen que queremos utilizar para el experimento que nos ocupa.

En nuestro ejemplo, de todos modos, nuestra primera acción ya nos había indicado que el lápiz USB está reconocido en */dev/sdb* (actuando como un disco entero; si habláramos de particiones, tendríamos que trabajar con los términos *sdb1, sdb2...*). Así que vamos a crear un directorio en la carpeta de usuario */home/antonio* llamado *copialapiz*, y dentro de dicha carpeta situaremos la imagen *copialapiz.dd ;* indicaremos a la herramienta dd que deseamos volcar la partición */dev/sdb1* en esa carpeta.

El siguiente paso, pues, será *montar* dicho archivo para que sea tratado como una copia duplicada del dispositivo físico.

## Montando nuestra imagen.

"Montar" un dispositivo o imagen en un sistema GNU/Linux, si bien suena como algo un poco engorroso, se ha mostrado como un mecanismo de trabajo con dispositivos de almacenamiento muy útil, que permite una mayor eficacia en la utilización de recursos: todos los dispositivos detectados por el ordenador son reconocidos, y como tal, se les hace referencia desde la carpeta */dev* del sistema de archivos en las particiones de tipo *ext*. Sin embargo, parece lógico que no se haga caso de un aparato que no se necesita hasta un momento determinado. Así, se ahorran muchos recursos si no se accede al interior de estos dispositivos hasta que no es necesario, hasta que no se "montan". En consola, el comando *mount* tiene múltiples parámetros y opciones de configuración, y el correspondiente *man mount* nos dejará en pantalla el correspondiente manual. Excede el objetivo de este artículo profundizar más en este tema, así que nos limitaremos a crear una segunda carpeta denominada *montalapiz* en nuestro directorio */home* de usuario, y montaremos ahí nuestra imagen con *mount -o loop*, sin dar mayores explicaciones.

Los contenidos que originalmente estaban en nuestro dispositivo de almacenamiento, y que aún puedan leerse, aparecerían ahora en el directorio *montalapiz*. Hora es ya de probar diversas posibilidades de recuperación.

## Uso de la herramienta *foremost*

Uno de nuestros gadgets favoritos en el ámbito de la recuperación de archivos es *foremost*. Como siempre, un simple *aptitude install foremost* desde consola (como root, por supuesto) nos instalará dicha aplicación.

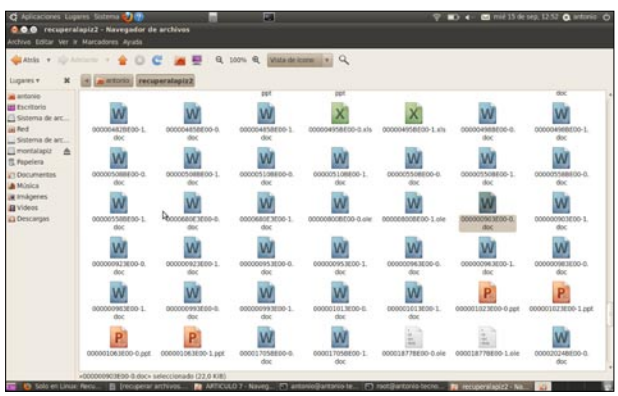

**Figura 3.** Como es habitual, estas herramientas forenses identifican los archivos recuperados con secuencias alfanuméricas, lo que dificulta su identificación a priori.

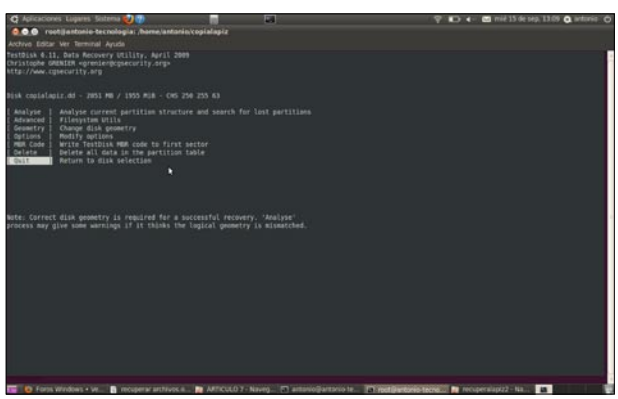

**Figura 4.** Testdisk es una excelente opción cuando el problema tiene que ver con errores en la MFT o en el MBR

El principio básico de trabajo de este tipo de herramientas forenses es muy simple, pero genial: partiendo de que cada tipo de archivo *(doc, jpg, exe,* etc...) utiliza una serie de bytes en las direcciones de memoria física donde se va a almacenar, pero se identifica con una secuencia determinada en el primer y en el último byte, se realiza un escaneado, byte a byte, de la imagen de la partición, buscando dichos patrones con las que típicamente empiezan/terminan los tipos más comunes de archivos.

El uso de *foremost* es muy simple. La documentación que proporciona un *man foremost* es lo suficientemente completa para que nos hagamos una idea de sus posibilidades, pero en un alarde de sencillez, nos ceñiremos a su uso más básico: indicaremos a foremost sobre qué carpeta queremos que trabaje.

En éste su uso más básico y simple, se le indica a *foremost* sobre qué carpeta queremos trabajar. No habiendo más indicaciones, la utilidad se dedicará a reconocer los tipos de archivo sobre los que tiene capacidad (jpg, gif, png, bmp, avi, exe, mpg, doc, pdf... y muchos más), y los que pueda ir rescatando los colocará en una carpeta de nombre *output* en la ruta desde la que damos la orden, en este caso, */home/antonio*. Dependiendo de la cantidad de información que a lo largo de su vida útil hayamos grabado sobre el dispositivo, además de su capacidad, la operación puede tardar desde unos minutos a varias horas.

Si tratamos de abrir desde el escritorio dicha carpeta, observaremos con consternación que está, en apariencia, vacía. ¿Por qué?. Téngase en cuenta que hemos acometido esta operación desde consola como *root*, y por lo tanto, la carpeta *output* generada sólo otorgaría a dicho usuario permisos de lectura (la escritura o ejecución serían aún menos concebibles, por supuesto).

Podemos, desde consola, hacer un simple *ls output* para ojear los archivos recuperados, clasificados por carpetas según el tipo de extensión que la utilidad ha juzgado que tenía dicho archivo. Por supuesto, mucho más sencillo desde el inicio sería ejecutar *foremost* como usuario normal, sin derechos de root.Hay que decir que, en ocasiones, lo que *foremost* ha identificado como un tipo en particular de archivo, en realidad no lo es, por lo que al intentar abrirlo con la aplicación correspondiente, el ordenador emitirá una señal de error, avisando sobre la posible corrupción del archivo.

Como decíamos antes, *hemos* comentado sólo la forma más básica de utilización de esta herramienta. Si lo que buscamos es tipo particular de archivos, jpg, podemos especificarlo con la opción *-t*. Si lo que deseamos es enviar todos los archivos recuperados a una carpeta específica, fuera de la carpeta *output* por defecto, utilizaremos la opción *-o rutadelacarpeta*. Un ejemplo podría ser desear recuperar específicamente las fotografías en *jpg*, que deseamos enviar a una carpeta previamente creada, denominada *fotosdemiboda*. (Antes de seguir, queremos avisar al astuto lector que no es un ejemplo ficticio; efectivamente, una compañera estuvo a punto de perder un álbum de este tipo, hace unos

**Listing 4.** Montamos nuestra imagen en un directorio previamente creado al efecto.

```
root@antonio-tecnologia:/home/antonio/copialapiz# mount -o loop copialapiz.dd /home/antonio/montalapiz
```

**Listing 5.** Subimos un nivel y empleamos foremost sobre el directorio en el que hemos montado nuestra imagen.

```
root@antonio-tecnologia:/home/antonio/copialapiz# cd ..
root@antonio-tecnologia:/home/antonio# foremost montalapiz
```

**Listing 6.** Uso de foremost como usuario no root, con salida a un directorio de nuestra elección, y filtrando la búsqueda solamente a archivos *jpg*.

```
root@antonio-tecnologia:/home/antonio# su antonio
antonio@antonio-tecnologia:~$ mkdir fotosrecuperadas
antonio@antonio-tecnologia:~$ foremost -t jpg -o /home/antonio/fotosrecuperadas/ /home/antonio/montalapiz
```

**Listing 7.** Averiguamos los patrones de comparación que magicrescue puede utilizar a la hora de recuperar archivos.

```
root@antonio-tecnologia:/usr/share/magicrescue/recipes# cd /home/antonio
root@antonio-tecnologia:/home/antonio# ls /usr/share/magicrescue/recipes/
avi        elf   gimp-xcf  gzip       jpeg-jfif  mp3-id3v2  nikon-raw  png  zip
canon-cr2  flac  gpl       jpeg-exif  mp3-id3v1  msoffice   perl       ppm
```

**Listing 8.** Ejemplo de utilización de *magicrescue* sobre una imagen de extensión dd en busca de documentos realizados con MS-Office.

```
root@antonio-tecnologia:/home/antonio# magicrescue -d /home/antonio/recuperalapiz -r  /usr/share/magicrescue/
recipes/msoffice /home/antonio/copialapiz/copialapiz.dd
```

**Listing 9.** Uso del filtro *grep* para localizar los archivos que contienen en su interior una cadena de texto que nos permita averiguar parte de su contenido, y volcado del listado obtenido a un archivo de texto.

```
antonio@antonio-tecnologia:/home/antonio# grep -l -i -r "1ª EVALUACION" recuperalapiz/
>examenes1evaluacion.txt
```

años; la anécdota es real, y puede ilustrarnos sobre hasta qué punto nos arriesgamos, a veces, al no asegurar nuestra información con copias alternativas).

Las líneas a teclear, pues, serían Listing 6.

Podríamos, incluso, rizar el rizo, si conocemos las secuencias (texto plano o hexadecimal) con las que se identifica la extensión de archivos que deseamos encontrar, e incluirlas o modificarlas en el archivo que recoge por defecto dichos patrones, */etc/foremost.conf*

## Uso de la herramienta *magicrescue*

*Magicrescue* es también un utilísimo programa que, desde consola, sigue un proceso parecido al de *foremost*, si bien exige un uso más específico (debemos indicar forzosamente qué tipo de archivo estamos buscando), y no trabaja sobre directorios. Además, los comandos de órdenes a teclear en consola son (algo) más complejos, si bien también tienen su lógica.

Se dice que *magicrescue* trabaja con *recetas*. Estas recetas son los mismos patrones cuya concordancia se busca al registrar la imagen escaneada, byte a byte. Se encuentran típicamente en la carpeta */usr/share/magicrescue/recipes*, y un simple comando de listado *ls* nos dirá los tipos de archivo que podemos buscar.

Para utilizar este programa, debemos indicar, por este orden:

- El directorio de recogida de los archivos rescatados, con el flag *-d.*
- El tipo de archivo que deseamos rescatar, con el flag *-r usr/share/magicrescue/recipes/nombredepatron* (se han visto las posibilidades en el anterior apartado).
- La ruta de la imagen (en nuestro ejemplo, copialapiz.dd) sobre la que queremos hacer el escaneado.

Para ilustrarlo con un ejemplo, recordemos que estamos refiriéndonos a un centro educativo. Supongamos que la memoria USB sobre la que estamos trabajando ha perdido, sobre todo, exámenes y listas de notas, guardadas (nos apena reconocerlo, pero suele ser así) con un programa de la suite MS-OFFICE. Bueno, pues ésta es nuestra prioridad. La orden a dar a la consola sería Listing 8.

Tras unos segundos (minutos, como mucho), la consola nos informará del final del proceso, y podremos observar el resultado de nuestras pesquisas en */home/antonio/recuperalapiz*. Por supuesto, y como ya hemos avisado, los nombres de los archivos recuperados son secuencias alfanuméricas determinadas por la posición en que dichos archivos han sido localizados. En un principio, nos tocaría ahora abrir archivo por archivo para comprobar su contenido, e irlos renombrando. Al final del artículo enunciaremos una propuesta, a título individual, que podría resolvernos, al menos en parte, el trabajo.

Como puede comprobarse, *magicrescue* es una herramienta un poco más compleja y específica, pero más rápida y potente que *foremost*, si bien presenta el inconveniente de que, en un principio, el usuario aficionado (como estos humildes profesores que escriben estas líneas) se encuentra más limitado a la hora de recuperar determinados tipos de archivos.

## Testdisk. Cuando el problema está en las tablas de particiones

Sobre todo en discos duros, y muy particularmente relacionados con apagados (accidentales o no), y casi siempre en el "sistema operativo que constituye nuestra competencia" (no haremos publicidad, ni siquiera negativa), el problema puede ser más simple, pero igualmente destructivo para el usuario: el ordenador, simplemente, no funciona. Se niega a arrancar. Un mensaje en inglés o en castellano sobre algo denominado MBR (la misión y naturaleza del sector de arranque, por más que nos empeñemos, no es algo que interese conocer al usuario no avanzado de ningún sector operativo), la pantalla en negro...Otras veces, un virus o un troyano que nos ha terminado de rendir el sistema operativo (obviamente, no de tipo GNU/Linux)... Los famosos "pantallazos azules", con o sin reinicio automático... Parafraseando a aquél personaje de Marlon Brando: "*La desesperación... la desesperación...*".

Bueno, en ese caso estamos hablando de discos duros que, en un principio, no han sufrido (aunque también podríamos solucionarlo) daños físicos de importancia, ni borrados erróneos, pero en un principio irrecuperables (ya hemos demostrado lo contrario), de archivos en particular. En estos casos, simplemente, sólo habremos de valernos de cualquier distribución *live* que incorpore la herramienta *Testdisk*.

*Testdisk* es una herramienta que puede utilizarse directamente en el ordenador averiado, vía *cd-live*, o si hemos podido volcar su imagen con *dd* a otro dispositivo de almacenamiento, desde otro ordenador se puede revisar y restaurar dicha imagen. Este programa, básicamente, lo que hace es analizar el *MBR (Master Boot Record)*, el *MFT* (*Master File Table*) o equivalente en particiones no NTFS, y la geometría física del disco duro o memoria flash a revisar.

El primer paso, si estamos en un disco duro averiado, y no estamos seguros de por dónde nos andamos, será realizar un *testdisk /list* desde consola, que nos dará un listado de los discos montados encontrados. Si efectuamos un simple *testdisk* en el BASH, el programa arrancará con tres opciones.

- Create: Crear un log para saber qué pasos hemos ido dando.
- Append: Añadir esos datos al final de otro log ya existente.
- No Log: No crear ningún log.

Escogemos la opción que más nos convenga, y a continuación se nos pedirá que escojamos la partición con la que deseamos trabajar (Nota: si, como decíamos antes, lo que deseamos es revisar la imagen de un dispositivo de almacenamiento creado con *dd*, nos ahorraremos este paso iniciando el programa con *testdisk rutadelaimagen/nombredelaimagen.dd*). Deberemos indicar la forma (que no el tipo) de partición sobre el que creemos estar trabajando (INTEL/PC, MAC, XBOX...).

A partir de ahí, el usuario puede ir explorando las posibilidades del programa, que a grandes rasgos nos permite:

- Analizar todas las particiones, buscando particiones (en su caso) perdidas.
- Herramientas avanzadas para sistemas de archivos (como por ejemplo, convertir una partición en partición de arranque).
- Cambiar la geometría del disco (sólo usuarios avezados, que ya hayan hecho todo lo posible por recuperar la información vital).
- Restaurar el Master Boot Record.
- Borrar particiones.

Testdisk es una herramienta distinta pero complementaria de *foremost* y *magicrescue*. No recupera datos de una partición como tal, pero permite devolver la funcionalidad perdida a un disco duro o partición, cuando el problema es de arranque. De hecho, en caso de avería física, suele ser necesario empezar nuestro trabajo con *Testdisk* antes de proceder a la recuperación como tal de archivos y carpetas.

## ¿Tengo que abrir todos los archivos recuperados para poder identificarlos?.

Si no hay más remedio... Pero, y sobre todo en documentos ofimáticos, esta tarea puede pasar de ser titánica a ser imposible. Recordemos que los archivos que se recuperan no son sólo los que hemos perdido en nuestro accidente o error, sino también `todos` los que alguna vez se grabaron en el dispositivo analizado, y no han sido sobreimpresionados con posterior información en la misma dirección física de memoria.

Los redactores del artículo ya hemos tenido que contender unas cuantas veces con este problema, y como es cierto que de la necesidad surge el ingenio, quisimos explorar un poco las posibilidades de BASH, concretamente de la herramienta de filtrado *grep,* para facilitar un poco este trabajo.

*Grep* es una herramienta de filtrado que nos permite buscar caracteres o grupos de caracteres en un archivo o grupo de archivos. Bien utilizada, podremos, al menos, separar los archivos que contengan palabras o líneas específicas. Supongamos, por ejemplo, que queremos localizar los exámenes de la 1ª evaluación. Utilizaremos, entonces, el comando (Listing 8).

Lo que hemos pedido a la consola es lo siguiente: "lista (-l) los archivos, indistintamente (-i) de mayúsculas y minúsculas, en la carpeta, de manera recursiva (-r) en los que te encuentres la cadena (imprescindible el entrecomillado) "1ª EVALUACION", y vuelca los nombres en un archivo denominado *examenes1evaluacion.txt*.

En dicho texto, constará un listado con los nombres de los archivos en los que se ha encontrado dicha cadenas de texto. Trabajo del usuario será ahora ir buscando dichos archivos, abrirlos, y comprobar si era lo que estaban buscando.

Por supuesto, a partir de aquí, el usuario algo más avezado puede crear su propio script a partir de esta base, modificada y mejorada, de modo que a cada ejecución, el usuario introduzca la cadena que desea localizar, se separen los archivos afectados por copia o traslado a otras carpetas, etc...

## ¿Significa todo esto que no puedo borrar nunca la información de manera definitiva?

En un centro educativo, como en cualquier organización de tipo estatal, se trabaja con alguna información sensible. No tanto como parecer *top secret*, pero está claro que algunos de los equipos, al menos los pertenecientes a los órganos directivos y el departamento de orientación trabajan con datos personales y privados.

Vivimos en la era del reciclado. No es extraño ver cómo un equipo que se queda pequeño para un cometido en particular es fácilmente sustituido, ya que los precios actuales lo permiten, por otro mucho más potente. Como tampoco lo es ver que ese equipo, que sigue siendo útil, sea reutilizado en otro ámbito del centro de trabajo. El hecho de borrar, simplemente, las carpetas con información importante, e incluso un formateado del disco duro, sobre todo si se hace el mismo tipo de partición, no será garantía absoluta de que todos los datos han desaparecido. Siempre se pueden recuperar, al menos en parte.

El problema se agudiza cuando hablamos de equipos que se han utilizado en oficinas bancarias, comerciales, e incluso en departamentos estatales o gubernativos, en los que se trabaja con multitud de datos económicos y personales de cientos, miles de ciudadanos. ¿Significa esto que los equipos en los que se ha utilizado información sensible no pueden reutilizarse, al menos en sus discos duros?. No, por supuesto. Basta con realizar un reformateo a bajo nivel.

¿A qué nos estamos refiriendo?. En el caso de GNU/Linux, a volver a utilizar la herramienta *dd*. En este caso, grabando expresamente un 0 en cada bit que conforma la totalidad de la memoria del disco duro:

```
root@antonio-tecnologia:/home/antonio# dd if=/dev/zero
of=/dev/sdb
```

También podemos grabar 1 ó 0 de manera aleatoria bit a bit con esta herramienta:

```
root@antonio-tecnologia:/home/antonio# dd if=/dev/
urandom of=/dev/sdb
```

## Conclusión

A lo largo de este artículo, hemos intentado, con mejor o peor resultado, introducir al usuario novel a las posibilidades que desde la consola, en Linux, existen a la hora de recuperar archivos que se daban por perdidos. Por nuestra naturaleza profesional, hemos utilizado como entorno de pruebas un centro educativo como es un instituto, dado que es el caldo de cultivo ideal para que se den estas situaciones: usuarios poco avanzados, ordenadores utilizados de manera comunitaria, intercambio continuo de archivos, trasiego de pendrives... Los contenidos aquí expuestos, que hemos intentado que se atengas lo más posible a la verdad, creemos que pueden ser lo suficientemente simples pero al mismo tiempo eficaces para animar a otros compañeros que se vean en este tipo de tesituras, para que al menos intenten dar pasos como éstos. Al fin y al cabo, rendirse a la evidencia y dar los archivos por perdidos siempre queda como última opción.

## Sobre los autores


María Dolores Nogueras Atance, licenciada en Ciencias Químicas, es profesora de Tecnologías en la actualidad, pero también ha pasado algunos años como profesora de Formación Profesional en Laboratorio. Su irrupción en el mundo informático ha sido algo tardío, y debido sobre todo a la estrecha relación de dicho mundo con la materia que actualmente imparte. Sin embargo, ha sabido retomar el ritmo y pone a prueba y se esfuerza por aprender toda nueva herramienta informática que caiga en sus manos y que pueda tener algo que ver con la educación.

Antonio Gómez García es Ingeniero Técnico Industrial de Formación, y lleva más de diez años dedicando su actividad profesional a la Educación Secundaria y Bachillerato en institutos. Profesor de Tecnologías y de Tecnologías de la Información, ha trabajado como asesor TIC en el Centro de Profesores de Puertollano, y dedica gran parte de su tiempo al software libre y su introducción en el sistema educativo. Desde esa filosofía, ha colaborado ya en varias actividades de formación de padres, profesores y alumnos sobre seguridad en Internet. En la actualidad, es Responsable de Medios Informáticos en el IES Eduardo Valencia, de Calzada de Calatrava (Ciudad Real). Agradecerá cualquier aporte que queráis realizar en administrador@eduardovalencia.no-ip.org