Backups

Eugenia Damonte, Ariel Fideleff y Martín Goñi

Índice

1. Backups

1.1. Que es un backup

En el mundo del IT un backup es una copia de parte o toda la información de una computadora, almacenada en una unidad de almacenamiento distinta a la de la computadora. Esta puede luego ser usada para recuperar la informacón original en caso de que ocurra una pérdida de datos. Es importante recalcar que una pérdida de datos puede ocurrir no solo debido a daño al sistema, ya sea de hardware o software, sino que también puede ser provocada por un error humano (por ejemplo, borrar un archivo importante). Para cumplir su función, un backup debe contener por lo menos una copia de toda la información que se considere que vale la pena guardar. Esto nos introduce a un dilema muy importante, ¿qué información vale la pena guardar?

En principio uno podría pensar que simplemente deberíamos hacer un backup de todo el sistema, para no tener que tomar esta decisión. Sin embargo, a medida que crece el tamaño y complejidad del sistema se vuelve cada vez mas costoso, en todo sentido, realizar backups completos. Algo que también se debe tomar en cuenta al hacer esta decisión es el valor del sistema en sí, es decir, cuánto vale el sistema operativo, sus configuraciones y ajustes. Si bien a simple vista esto puede parecer algo no muy importante, en sistemas grandes y complejos, que requieren una gran cantidad de conocimiento y experiencia para configurarlos, puede ser igual de valiosa que la información que almacena el sistema.

1.2. Tipos de backups

Antes de poder elegir qué tipo de backup hacer hay que elegir qué método utilizar para el mismo. Los dos que se usan hoy en día son:

- **Backup por archivos:** El backup por archivos es la forma original en que se hacían los backups. En este, todos los archivos y carpetas a los que se les debe realizar un backup son copiados utilizando las utilidades provistas por el sistema operativo. Este método si bien es

simple, también es lento y consume una gran cantidad de recursos.[1]

- **Backup por imágenes:** Otra opción que esta ganando popularidad es el backup por imágenes. Este método sobrepasa gran parte de las utilitades del sistema operativo, copiando bloques del disco duro de manera directa. Esto le permite ser mucho más eficiente a la hora de copiar archivos que han sido modificados, porque no es necesario copiar todo el archivo, solo los bloques que han sido modificados.

Cabe destacar que estos métodos no son mutuamente exclusivos, se pueden usar en conjunto para obtener mayor eficiencia y robustez. Por ejemplo, se puede tener un sistema que haga un backup por imágen diariamente y uno por archivos semanalmente.

Una vez que se decidió qué metodo utilizar para hacer los backups ahora hay que decidir qué método usar para los mismos. Los backups se dividen en tres tipos:

- **Backup completo:** Es el mas simple y el método original que se usaba para hacer los backups. Copia toda la información en el sistema especificado. Lo bueno de este método es que el backup es autocontenido, esto significa que no se requiere de ningún otro tipo de información o archivo para que este funcione. Por el otro lado, se necesitan grandes cantidades de espacio y pueden ser casi idénticos a backups completos anteriores.

- **Backup diferencial:** Este tipo de backup sólo copia las diferencias entre el sistema actual y el del último backup completo. La principal ventaja de este método es que es mucho mas rápido y ocupa mucho menos espacio que un backup completo. La desventaja es que para poder recuperar la informaciñ con un sistema de backup diferencial se necesita el último backup completo junto con el backup diferencial.

- **Backup incremental:** El backup incremental sólo copia diferencias entre el el sistema actual y el último backup completo, diferencial o incremental. La ventaja es que es aún mas rápido y ocupa menos espacio que un backup diferencial. El gran inconveniente con esta forma

[1] Esto se debe a que para copiar un archivo utilizando el sistema operativo se debe: Encontrar los bloques en el disco duro donde se encuentra la carpeta que contiene al archivo, leer la carpeta, buscar el archivo especificado, determinar en que bloques se encuentra y finalmente copiarlo.

de backup es que para recuperar la información se necesitan todos los backups incrementales anteriores junto con el último backup completo. Debido a esto recuperar información con este tipo de sistema puede ser un proceso largo.

2. rsync

2.1. Que es rsync

rsync es una utilidad que permite transferir y sincronizar archivos, entre otras cosas, entre una computadora y un disco duro. También es capaz de realizar estas tareas usando dispositivos de red. Su uso es muy común en sistemas basados en Unix, dada su simplicidad y facilidad de uso.

El programa inicial fue escrito por Andrew Tridgell y Paul Mackerras, en C. Su primera versión se anunció en Junio de 1996, luego en 1999 Tridgell habló sobre el diseño y la implementación de **rsync** en su tesis. Es similar a la utilidad `rdist -c` creada por Ralph Campbell en 1983. Actualmente Wayne Davison se encarga de mantener el proyecto.

2.2. Instalación de rsync

Antes de poder usar **rsync** tuvimos que instalarlo. Si bien hoy en día suele venir incluido con la gran mayoría de las distros, debido a que Debian 7 es bastante viejo no la incluye. Para instalarlo usamos el comando `sudo apt-get install rsync`.

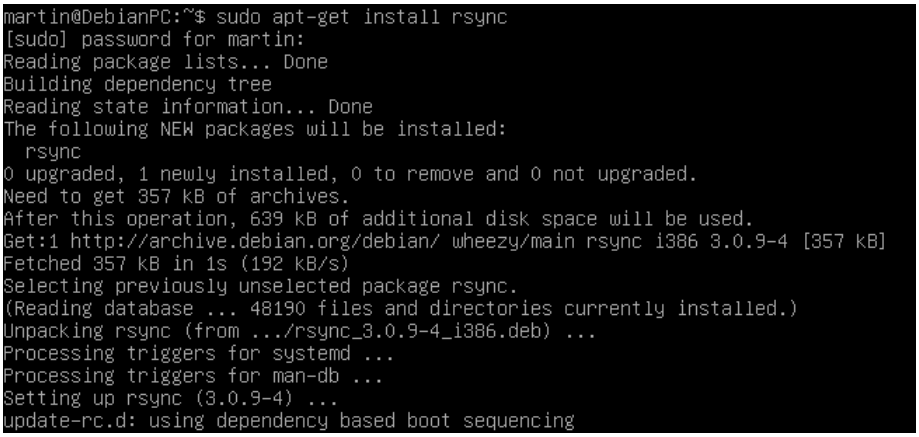


Figura 2.1: Instalamos **rsync** con `sudo apt-get install rsync`

2.3. Uso básico de rsync

2.3.1. Preparando el disco

Lo primero que necesitamos para hacer un backup de cualquier tipo es una unidad de almacenamiento para guardar el backup. En nuestro caso simplemente creamos otro disco duro y lo "conectamos" a la VM. Para hacer esto con la máquina apagada abrimos la configuración y fuimos a `Storage`. Allí apretamos el botón para añadir un disco duro, esto nos llevo a un menú donde elegimos crear un nuevo disco. Especificamos el tipo y tamaño del disco y presionamos `Create`. Finalmente montamos el disco, para lo que volvimos a apretar el botón para añadir un disco duro y seleccionamos el creado.

Es importante destacar que no se debe usar un pen drive para realizar backups. Esto es porque los transistores que almacenan información en los mismos no están hechos para soportar el número de escrituras que se necesitan para una unidad de backup. Esto a largo plazo causa que algunos de los transistores en ellos se queden "trabados"[2] en una posición haciendo imposible contiunar usándolo. Con el tiempo esto causa deterioro en la capacidad y velocidad de funcionamiento de la unidad y puede incluso causar perdida de datos.

[2]Realmente no se quedan "trabados" sino que el transistor, normalmente un FGT, pierde su capacidad de cargarse y descargarse, y por tanto de cambiar de estado.

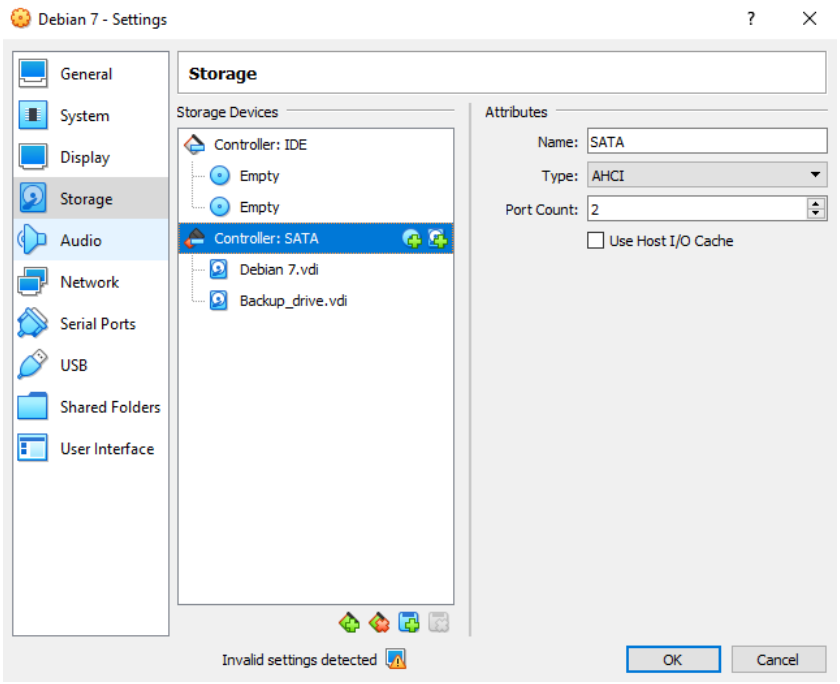


Figura 2.2: Creamos un nuevo disco y lo montamos en la VM

Ahora que nuestra VM podía ver el disco, había que montarlo y configurarlo. Primero utlizamos el comando `sudo fdisk -l` para verificar que el disco fuese detectado por el sistema. Al usar el comando, éste nos mostró que había un disco llamado `/dev/sdb` de 10GB, el cual habíamos montado.

```
martin@DebianPC:~$ sudo fdisk -l
[sudo] password for martin:

Disk /dev/sda: 16.1 GB, 16106127360 bytes
255 heads, 63 sectors/track, 1958 cylinders, total 31457280 sectors
Units = sectors of 1 * 512 = 512 bytes
Sector size (logical/physical): 512 bytes / 512 bytes
I/O size (minimum/optimal): 512 bytes / 512 bytes
Disk identifier: 0x0007252e

   Device Boot      Start         End      Blocks   Id  System
/dev/sda1   *        2048      499711      248832   83  Linux
/dev/sda2          501758    31455231    15476737    5  Extended
/dev/sda5          501760    31455231    15476736   8e  Linux LVM

Disk /dev/sdb: 10.7 GB, 10737418240 bytes
255 heads, 63 sectors/track, 1305 cylinders, total 20971520 sectors
Units = sectors of 1 * 512 = 512 bytes
Sector size (logical/physical): 512 bytes / 512 bytes
I/O size (minimum/optimal): 512 bytes / 512 bytes
Disk identifier: 0x00000000
```

Figura 2.3: Usamos el comando `sudo fdisk -l` para verificar que el disco fuese detectado por el sistema

Sabiendo el nombre del disco procedimos a particionarlo y formatearlo. Para esto usamos el comando `sudo cfdisk /dev/sdb`, que lo que hace es abrir `cfdisk`, una utilidad que permite crear particiones de discos. Primero nos aseguramos de que este fuese el disco que habíamos montado y no solo otro con esa capacidad, mirando el tipo del sistema de archivos. Éste decía `Free Space`, es decir, que no estaba formateado, era nuestro disco. Entonces seleccionamos la opción `New` para crear una nueva partición, dejamos el tamaño especificado por `cfdisk`, que es el máximo que permite la unidad. Esto nos devolvió al menú principal, en donde confirmar los cambios, usamos la opción `Write` para escribir los cambios a la tabla de discos del sistema.

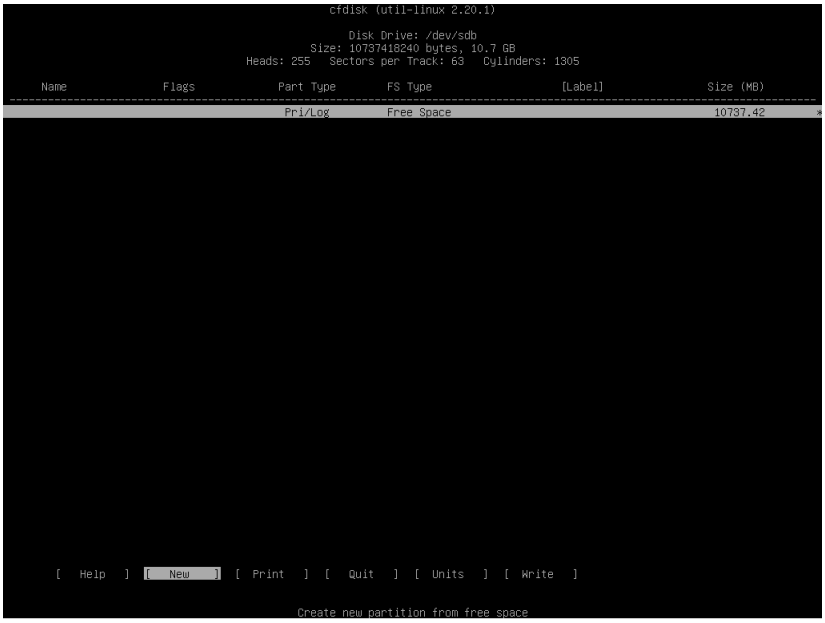


Figura 2.4: Usamos `cfdisk` para crear una partición en el disco nuevo

Finalmente confirmamos que la operación se había realizado de manera exitosa usando la opción `Print`. Ésta nos mostró que efectivamente había una partición en el disco.

```
Partition Table for /dev/sdb

        ---Starting----      ----Ending-----    Start     Number of
 # Flags Head Sect  Cyl   ID  Head Sect  Cyl    Sector    Sectors
-- ----- ---- ---- ----- ---- ---- ---- ----- ---------- ----------
 1  0x00    1    1     0 0x83  106   17  1305         63    20971457
```

Figura 2.5: Una vez creada la partición la verificamos con la opción `Print`

Ahora que ya teníamos una particion que podíamos usar, llegó el momento de formatearla. Para eso, usamos el comando `sudo mkfs.ext4 /dev/deb`. Lo que hizo fue formatear el disco con el formato `ext4`,[3] para poder así montarlo.

```
martin@DebianPC:~$ sudo mkfs.ext4 /dev/sdb
[sudo] password for martin:
mke2fs 1.42.5 (29-Jul-2012)
/dev/sdb is entire device, not just one partition!
Proceed anyway? (y,n) y
Filesystem label=
OS type: Linux
Block size=4096 (log=2)
Fragment size=4096 (log=2)
Stride=0 blocks, Stripe width=0 blocks
655360 inodes, 2621440 blocks
131072 blocks (5.00%) reserved for the super user
First data block=0
Maximum filesystem blocks=2684354560
80 block groups
32768 blocks per group, 32768 fragments per group
8192 inodes per group
Superblock backups stored on blocks:
        32768, 98304, 163840, 229376, 294912, 819200, 884736, 1605632

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

Figura 2.6: Formateamos el disco con el comando `sudo mkfs.ext4 /dev/deb`

Como último paso montamos el disco para poder usarlo. Para esto primero creamos una carpeta en la cual montar el disco, normalmente estas se encuentran en el directorio `/mnt`. Entonces creamos la carpeta `/mnt/sdb` con el comando `sudo mkdir /mnt/sdb`.

```
martin@DebianPC:~$ sudo mkdir /mnt/sdb
[sudo] password for martin:
```

Figura 2.7: Creamos el directorio donde montar el disco con `sudo mkdir /mnt/sdb`

Finalmente montamos el disco para poder usarlo con `sudo mount /dev/sdb /mnt/sdb`. El problema con solo hacer esto es que tendríamos que volver a montar el disco cada vez que iniciácemos el sistema. Para solucionar esto, cambiamos el archivo `/etc/fstab`, que almacena los discos que deben ser montados al iniciar el sistema. Abrimos el archivo

[3] `ext4` (fourth extended filesystem) es un sistema de archivos transacional que reemplazó a ext3.

para editarlo con el comando `sudo vi /etc/fstab`, y luego añadimos lo siguiente al final del mismo:

```
/dev/sdb /mnt/sdb ext4 defaults 0 0
```

```
# /etc/fstab: static file system information.
#
# Use 'blkid' to print the universally unique identifier for a
# device; this may be used with UUID= as a more robust way to name devices
# that works even if disks are added and removed. See fstab(5).
#
# <file system> <mount point>   <type>  <options>       <dump>  <pass>
/dev/mapper/DebianPC--vg-root /               ext4    errors=remount-ro 0       1
# /boot was on /dev/sda1 during installation
UUID=9cfa06ae-0230-4dd7-a9fa-4a170bcf8843 /boot           ext2    defaults        0       2
/dev/mapper/DebianPC--vg-swap_1 none            swap    sw              0       0
/dev/sr0        /media/cdrom0   udf,iso9660 user,noauto     0       0
/dev/sr1        /media/cdrom1   udf,iso9660 user,noauto     0       0
/deb/sdb        /mnt/sdb        ext4    defaults        0       0_
```

Figura 2.8: Editamos `sudo vi /etc/fstab` para que el disco se monte automaticamente al iniciar la máquina

El primer elemento es el camino del disco, el segundo a dónde debe ser montado y el tercero el sistema de archivos. Los demás los dejamos con sus valores por defecto. Habíendo hecho esto, el disco debería montarse automaticamente cada vez que iniciemos la máquina. Como una verificación usamos el comando `mount | grep "sdb"`, que lo que hace es listar todos los discos montados y buscar uno llamado `"sdb"`.

```
martin@DebianPC:~$ mount | grep "sdb"
/dev/sdb on /mnt/sdb type ext4 (rw,relatime,user_xattr,barrier=1,data=ordered)
```

Figura 2.9: Verificamos que el disco este estuviese montado correctamente con `mount | grep "sdb"`

Al ejecutar el comando vimos que efectivamente había un disco montado llamado `"sdb"`, confirmando que el disco estaba montado correctamente. Ahora que el disco estaba montado y funcionando, decidimos probar almacenar algo en el, sólo para probar. Nos movimos al mismo con `cd /mnt/sdb` y luego usamos el comando `touch Prueba` para intentar crear un archivo. Al ejecutar el comando, el sistema nos dijo que no teníamos los permisos para crear un archivo. Cuando listamos los permisos del disco con `ls -l ..`, se volvió claro porque. El dueño del disco era el usuario `root`, no nuestro usuario.

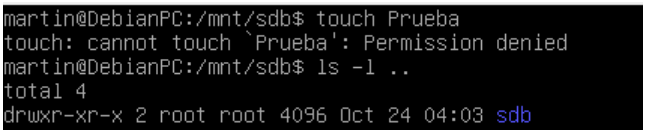


Figura 2.10: Intentamos crear un archivo con `touch Prueba`

Para solucionar esto nos movimos a la carpeta `/mnt` (donde `sdb` se encuentra) y usamos el comando `sudo chown -R martin:martin sdb` para cambiar el dueño de la carpeta de `root` a `martin`. Habiendo hecho esto repetimos la prueba con `touch` y esta fue exitosa.

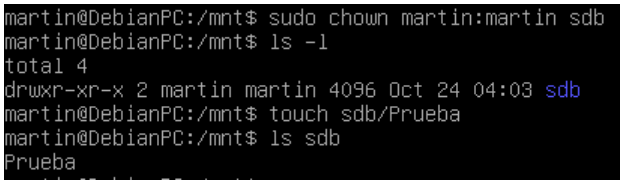


Figura 2.11: Cambiamos el dueño de `sdb` para solucionar el problema y lo verificamos

Finalmente con el disco funcionando correctamente, decidimos comenzar con la próxima parte del trabajo, hacer backups con **rsync** usando el nuevo disco.

2.3.2. Usando rsync

Habiendo llegado la hora de usar **rsync** decidimos hacer un backup completo de una carpeta en nuestro directorio propio como prueba de uso. Seleccionamos la carpeta `C`, que tenía todo el código correspondiente a el trabajo practico sobre software en formato fuente. La elegimos porque era una carpeta que a su vez tenía varias sub carpetas con archivos en ellas, lo que nos permitía probar las opciones recursivas de **rsync**. Antes de hacer el backup creamos la carpeta a donde los guardaríamos, `/mnt/sdb/Backups`, lo cual hicimos con el comando `mkdir /mnt/sdb/Backups`.

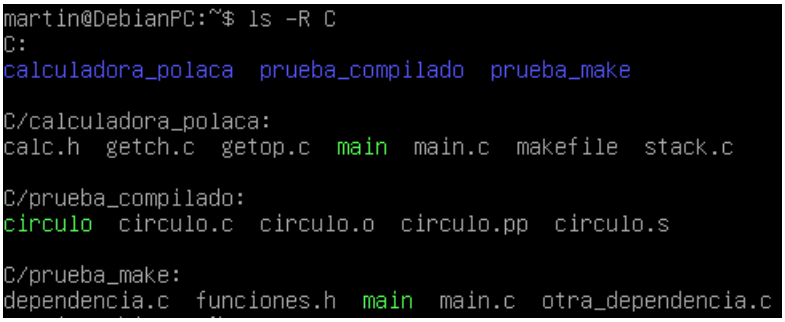


Figura 2.12: Los contenidos del directorio `C`

El uso básico de **rsync** es bastante simple, primero se pone el directorio desde el que se quiere copiar y luego hacia cual se quiere copiar. Para empezar, esto fue lo que hicimos, usamos el comando `rsync ~/C /mnt/sdb/Backup`, al hacerlo el mensaje *skipping directory C* apareció. Cuando revisamos los contenidos de `Backup` vimos que estaba vacío. Esta situación se explica si miramos los contenidos del directorio `C`, dentro del que hay sólo carpetas, no archivos. Entonces, al no encontrar archivos que copiar dentro de la carpeta especificada, **rsync** simplemente la omitió y no copió nada. Dado este problema añadimos la opción `-r` al comando, para copiar tambíen los contenidos de todas las subcarpetas. Al ejecutar el comando y luego verificar los contenidos de `Backup` vimos que ahora había una carpeta llamada `C` en ella, indicando que los archivos efectivamente se habían copiado.

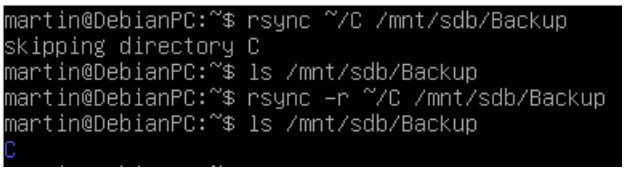


Figura 2.13: Nuestro primer intento de usar **rsync**

Para asegurarnos de que la copia hubiese efectivamente ocurrido, listamos los contenidos de `Backup` con `ls -Rl /mnt/sdb/Backup`. Esto nos reveló algo interesante, y es que todos los archivos tenían la fecha de modificación de cuando habíamos realizado el backup. Esto tenía sentido, ya que esos archivos se crearon y modificaron al momento en que nosotros realizamos el backup. Sin embargo, nos preguntamos si había alguna manera de conservar la fecha original. Entonces decidimos leer las opciones de **rsync**, para ver si

se podía, junto con tal vez encontrar otras opciones útiles.

```
martin@DebianPC:~$ ls -Rl /mnt/sdb/Backup
/mnt/sdb/Backup:
total 4
drwxr-xr-x 5 martin martin 4096 Oct 25 00:28 C

/mnt/sdb/Backup/C:
total 12
drwxr-xr-x 2 martin martin 4096 Oct 25 00:28 calculadora_polaca
drwxr-xr-x 2 martin martin 4096 Oct 25 00:28 prueba_compilado
drwxr-xr-x 2 martin martin 4096 Oct 25 00:28 prueba_make

/mnt/sdb/Backup/C/calculadora_polaca:
total 32
-rw-r--r-- 1 martin martin  239 Oct 25 00:28 calc.h
-rw-r--r-- 1 martin martin  989 Oct 25 00:28 getch.c
-rw-r--r-- 1 martin martin  727 Oct 25 00:28 getop.c
-rwxr-xr-x 1 martin martin 8063 Oct 25 00:28 main
-rw-r--r-- 1 martin martin 1486 Oct 25 00:28 main.c
-rw-r--r-- 1 martin martin  302 Oct 25 00:28 makefile
-rw-r--r-- 1 martin martin  601 Oct 25 00:28 stack.c

/mnt/sdb/Backup/C/prueba_compilado:
total 56
-rwxr-xr-x 1 martin martin  3649 Oct 25 00:28 circulo
-rw-r--r-- 1 martin martin   236 Oct 25 00:28 circulo.c
-rw-r--r-- 1 martin martin  1188 Oct 25 00:28 circulo.o
-rw-r--r-- 1 martin martin 40115 Oct 25 00:28 circulo.pp
-rw-r--r-- 1 martin martin   835 Oct 25 00:28 circulo.s

/mnt/sdb/Backup/C/prueba_make:
total 24
-rw-r--r-- 1 martin martin  181 Oct 25 00:28 dependencia.c
-rw-r--r-- 1 martin martin   58 Oct 25 00:28 funciones.h
-rwxr-xr-x 1 martin martin 5556 Oct 25 00:28 main
-rw-r--r-- 1 martin martin  288 Oct 25 00:28 main.c
-rw-r--r-- 1 martin martin  147 Oct 25 00:28 otra_dependencia.c
```

Figura 2.14: Los contenidos de `Backup`, mostrados con `ls -Rl /mnt/sdb/Backup`

Luego de leer la página de `man` de **rsync** encontramos no sólo opciones útiles, sino también información sobre como usar mejor el comando y adaptarlo a nuestras necesidades.

Lo primero que encontramos fue una sección referida a el uso de la barra(`/`). Si se pone al final del directorio desde el que se quiere copiar, cambia la forma en la que **rsync** almacena los archivos copiados. Lo que hace es evitar crear un directorio adicional, copiando directamente los contenidos del directorio especificado sin el directorio en sí. La diferencia se puede observar

mejor en la figura 2.15.

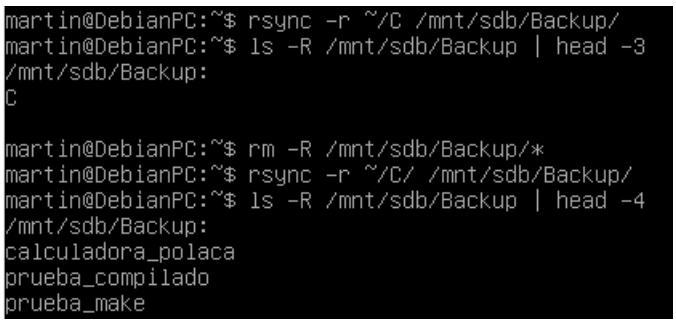


Figura 2.15: La diferencia entre usar y no usar la barra en **rsync**

En cuanto a las opciones, encontramos varias que nos parecieron bastante útiles. La primera que probamos fue `-v` (verbose), que como indica su nombre lo que hace es hacer la salida de **rsync** mucho más verbosa haciendo que muestre qué es lo que está copiando, así como cuántos bytes copia, y cuánto tiempo le lleva. Otra opción que también nos pareció útil, pero algo peligrosa, es `--delete`. Lo que hace es eliminar cualquier archivo en el directorio de destino que no esté en el directorio a copiar. La razón de su peligrosidad es su principio de funcionamiento, que si hay algun archivo en la carpeta de destino que desamos conservar pero éste no está en la carpeta a copiar, será borrado.

Luego de un poco más de búsqueda encontramos la opción `-a` (archive), la cual es equivalente a un gran número de opciones. Es la manera de decir que uno quiere preservar casi todos los atributos sobre los archivos copiados, con la unica exepción de que no preserva los enlaces duros.[4] Las opciones a las que `-a` se expande son:

- **-r:** Hace que el directorio a copiar se copie de manera recursiva, permitiendo copiar subdirectorios y sus contenidos.
- **-l:** Mantiene los enlaces símbolicos encontrados.
- **-p:** Mantiene los permisos originales de los archivos copiados. Sin esta opción, los archivos existentes (incluyendo los que se actualizan) mantienen sus permisos. Para los nuevos, los bits de permisos son iguales

[4]El motivo por el que no son preservados es porque encontrar archivos que poseen múltiples enlaces es costoso en cuanto a recursos. Para preservarlos se debe usar la opción `-H`.

```
martin@DebianPC:~$ rsync -rv ~/C/ /mnt/sdb/Backup/
sending incremental file list
.swp
calculadora_polaca/
calculadora_polaca/calc.h
calculadora_polaca/getch.c
calculadora_polaca/getop.c
calculadora_polaca/main
calculadora_polaca/main.c
calculadora_polaca/makefile
calculadora_polaca/stack.c
prueba_compilado/
prueba_compilado/circulo
prueba_compilado/circulo.c
prueba_compilado/circulo.o
prueba_compilado/circulo.pp
prueba_compilado/circulo.s
prueba_make/
prueba_make/dependencia.c
prueba_make/funciones.h
prueba_make/main
prueba_make/main.c
prueba_make/otra_dependencia.c

sent 78185 bytes  received 366 bytes  157102.00 bytes/sec
total size is 76948  speedup is 0.98
```

(a) **rsync** con la opción `-v`

```
martin@DebianPC:~$ echo "Hola" > /mnt/sdb/Backup/Prueba.txt
martin@DebianPC:~$ ls /mnt/sdb/Backup/
Prueba.txt
martin@DebianPC:~$ rsync -r --delete ~/C/ /mnt/sdb/Backup/
martin@DebianPC:~$ ls /mnt/sdb/Backup/
calculadora_polaca  prueba_compilado  prueba_make
```

(b) **rsync** con la opción `--delete`

Figura 2.16: Probamos usar las opciones `-v` y `--delete` con **rsync**

a los originales tras ser enmascarados con los bits de permiso predeterminados del directorio de destino.

- **-t:** Hace que los tiempos de modificación se transfieran de los archivos originales a los copiados. Si esta opción no se incluye, durante el próximo backup todos los archivos van a ser actualizados, es decir, copiados nuevamente.

- **-g:** Mantiene el grupo de los archivos copiados. Si esta opción no se incluye, los archivos copiados toman el grupo del usuario que haya ejecutado el comando.

- **-o:** Mantiene el dueño de los archivos copiados. Si esta opción no se incluye ,el dueño de los archivos copiados pasa a ser el usuario que haya ejecutado el comando.

- **-D:** Esta opción se expande a dos opciones más, las cuales son:
  - **--devices:** Hace que se copien los archivos de character y bloqueo de dispositivo al destino para recrear los dispositivos. Esta opción no tiene efecto si **rsync**no se ejecuta como `root`.
  - **--specials:** Hace que se copien todos los archivos especiales.

Habiendo analizado las opciones de **rsync**, decidimos que la configuración ideal, al menos en nuestro caso, sería `rsync -avH  /C/ /mnt/sdb/Backup`. Teniendo nuestra configuración, probamos usar el comando y funcionó perfectamente, haciendo un backup de todos los elementos en el directorio `C`.

```
martin@DebianPC:~$ rsync -avr ~/C/ /mnt/sdb/Backup
sending incremental file list
./
.swp
calculadora_polaca/
calculadora_polaca/calc.h
calculadora_polaca/getch.c
calculadora_polaca/getop.c
calculadora_polaca/main
calculadora_polaca/main.c
calculadora_polaca/makefile
calculadora_polaca/stack.c
prueba_compilado/
prueba_compilado/circulo
prueba_compilado/circulo.c
prueba_compilado/circulo.o
prueba_compilado/circulo.pp
prueba_compilado/circulo.s
prueba_make/
prueba_make/dependencia.c
prueba_make/funciones.h
prueba_make/main
prueba_make/main.c
prueba_make/otra_dependencia.c

sent 78207 bytes  received 369 bytes  157152.00 bytes/sec
total size is 76948  speedup is 0.98
```

Figura 2.17: Usamos **rsync** con todas las opciones que decidimos

Para verificar que el backup hubiese funcionado correctamente, listamos los contenidos del `Backup`con el comando `ls -lr /mnt/sdb/Backup/`. Efectivamente había funcionado de manera correcta, todos los archivos estaban y tenían todos los atributos correctos (dueño, permisos, última modificación, etc).

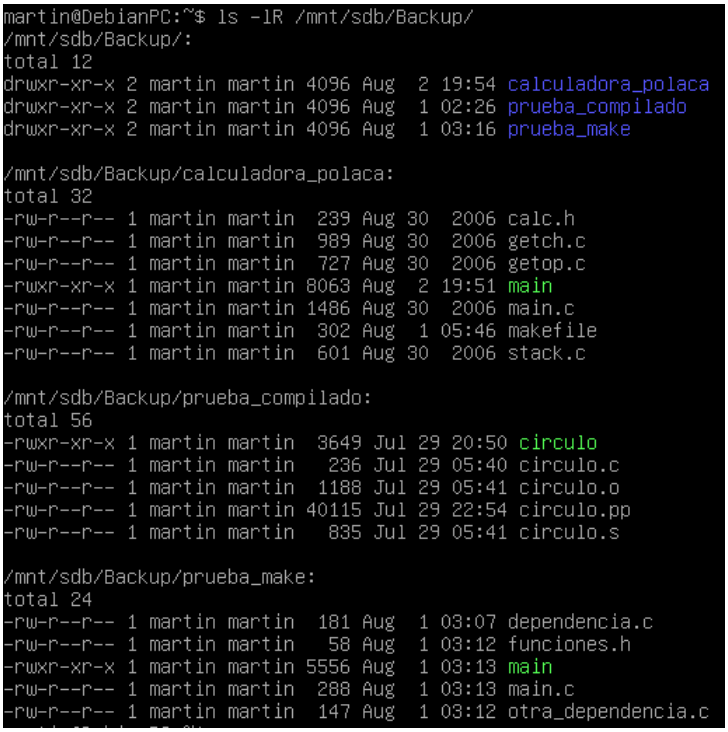


Figura 2.18: Verificamos que los resultados del backup fuesen correctos

Como nota final creemos que vale la pena aclarar que debido a nuestra configuración, **rsync** hace backups incrementales[5], dado que los archivos copian su fecha de modificación. Por lo que si intentamos hacer un backup de la misma carpeta sin cambiar nada, nignún archivo se copiará.

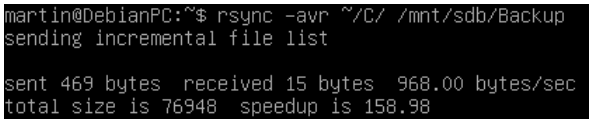


Figura 2.19: Revisamos si **rsync** hacía backups diferenciales con nuestra configuración

[5]Referirse a sección 1.2 - Tipos de backups.

3. Software comercial

Según la consigna del trabajo, ahora debemos elegir un software de backups comercial, es decir, que sea desarrollado y distrubuido por una empresa, generalmente en forma de una solución paga (que deberemos pagar). Para ello, elegimos uno muy conocido, *EaseUS Todo Backup*, de la empresa, como lo dice el nombre, *EaseUS*. Esta empresa se dedica al desarrollo de software comercial de distinto tipo, ofreciendo soluciones no sólo para backups, si no software también dedicado a recuperación de datos, administración de particiones en discos, y transferencia de datos entre computadoras, destacándose principalmente por el primero de la lista. En cuanto al programa con el que trabajaremos, mencionar que éste está disponible en dos versiones: para uso personal ("*for home*") y para uso profesional ("*for business*", también llamado "*Enterprise*") y hasta donde podemos ver, sólamente desarrollado para SOs Windows.

3.1. Instalación del software

Como dijimos que hay dos versiones, elegimos una de ellas. La especialidad y la materia tienen el objetivo de instruirnos en saber asesorar a negocios y particulares, como también manejarnos por nuestra cuenta en el ámbito correspondiente (sea hacer en este caso, backups por la seguridad de nuestros propios datos o los de nuestro propio emprendimiento, en caso de que vayamos a entablar uno), pero dado el particular enfoque del apunte de backups hacia lo profesional, sumado a que en este ámbito sea aplicable mayor variedad de opciones (y así en consecuencia, probablemente el software relacionado sea más completo) aplicando más de la teoría, optamos por probar la versión *Enterprise*.

Para instalarlo, primero nos dirigimos a la página de *EaseUS*, y de allí a la sección de su software de backups, versión profesional.[6] Debemos de descargar el ejecutable correspondiente.

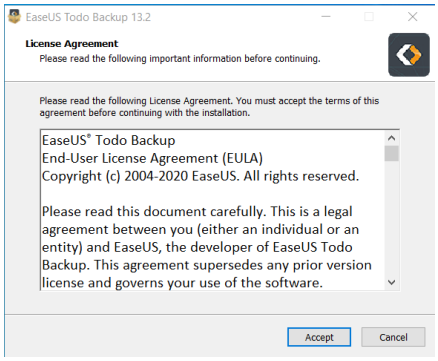
[6] `https://www.easeus.com/backup-software/tb-enterprise.html`

Figura 3.1: Página principal de *EaseUS Todo Backup Enterprise*

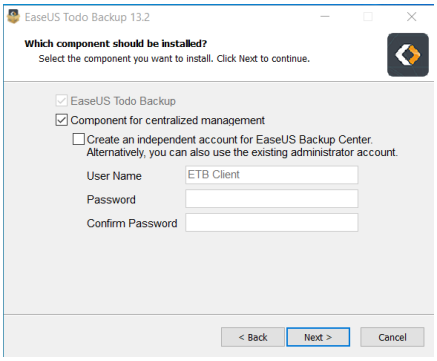
En nuestro caso, como lo usaremos nada más con propósitos demostrativos en este TP, utilizaremos la versión de prueba gratuita, *Free Trial*. Destacar que según se indica en el sitio, la versión de prueba tiene todas las funciones que la versión completa, pero nada más por un período de 30 días, para después del que luego deberemos de pagar para su uso continuado. Este tiempo será más que suficiente para ver todas sus funciones, dentro de los medios que tenemos disponibles.

Nos pide si, nuestro correo electrónico para continuar, pero supuestamente se puede optar por no recibir correos por parte de ellos, aunque dudamos de que aquello se cumpla. También es posible que use nuestro correo electrónico como un método para evitar estar continuamente volviendo a descargar el programa tras pasados los 30 días.

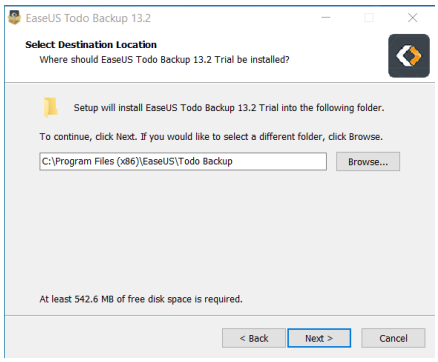
Al finalizar la descarga, tendremos un instalador de alrededor de unos 130 MB, llamado `tb_enterprise_trial.exe`. Al ejecutarlo y comenzar con la instalación, será bastante sencillo como seguir los pasos por los cuales nos guía el instalador. Al comenzar elegimos el idioma del instalador, nos dice de aceptar los términos y condiciones de uso, la ruta de instalación, y la ruta por defecto donde los backups se guardarían. Como si un dato curioso, permite manejar de forma centralizada los backups de distintos discos o unidades de almacenamiento que dispongamos a lo largo de múltiples sistemas, por un administrador, a través de otro software que parecen proveer, lo cual puede venir útil, aunque hay que tener en cuenta que ya en estos casos se optaría ya más por una solución hecha a medida, como se menciona en la teoría.



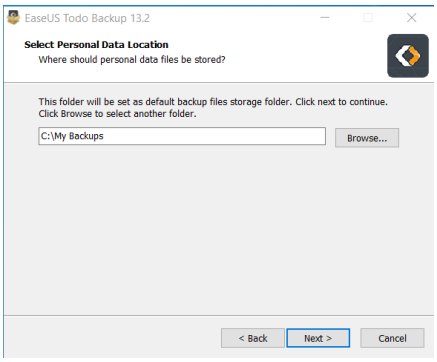
(a) Terminos y Condiciones



(b) Opción del componente de administración centralizada



(c) Ruta de instalación



(d) Ruta por defecto de backups

Figura 3.2: Etapas destacadas del proceso de instalación

3.2. Uso del software

Una vez ya instalado el programa, lo abrimos por primera vez y nos encontramos con la ventana principal desde la cual podemos acceder prácticamente a todas las distintas opciones que se ofrecen.

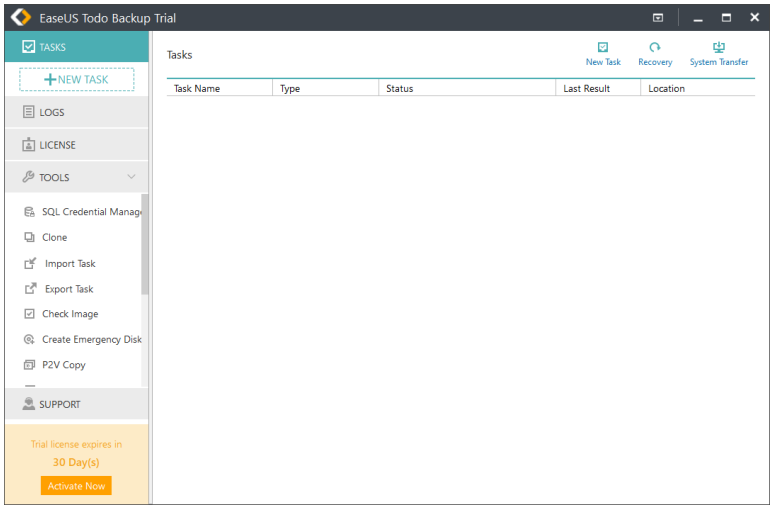


Figura 3.3: Pantalla principal del programa

3.2.1. Opciones del programa

3.2.1.1 Tasks

Podemos observar que hay una barra lateral con distintas secciones. La principal claramente es la de `TASKS`. Allí podremos encontrar las distintas tareas de backup que hayamos realizado a lo largo del tiempo, como así también poder restaurar alguna copia de seguridad realizada de ser necesario.

Para programar una nueva tareas, nos dirigimos al botón `NEW TASK`.

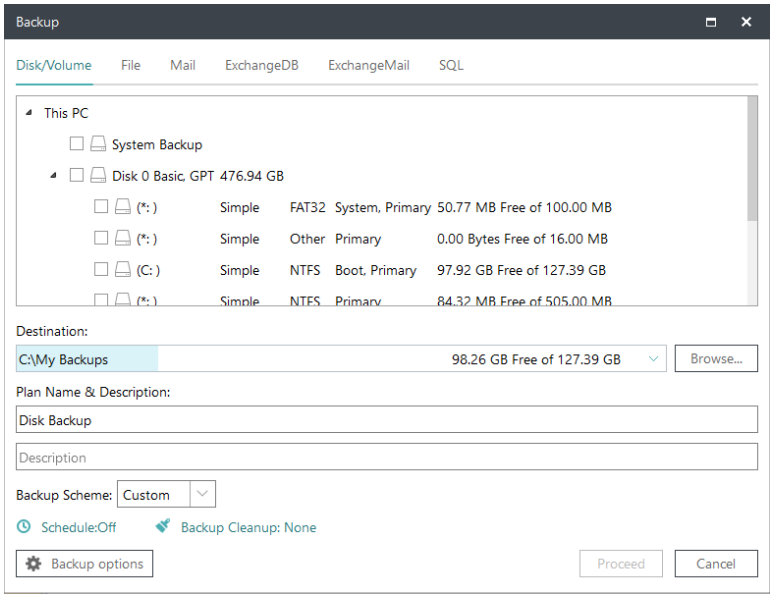


Figura 3.4: Ventana para programar una nueva *tarea*

Se nos abrirá una ventana para lo que sería, traducido al español, hacer lo que el programa llama una *Nueva Tarea*.

Nos encontramos a su vez con múltiples apartados. Primero podemos ver que nos permite hacer backup de múltiples "tipos" de información:

- **Disk/Volume** (*Disco/Volumen*): Podemos hacer el backup de un volumen[7] completo en nuestro sistema. Podemos realizarlo para uno, o varios en simultáneo.
- **File** (*Archivo*): También podemos hacer el backup de determinados directorios o archivos en nuestro sistema, de forma específica según nuestras necesidades. Esto es útil si queremos tener resguardada aparte información cuya importancia sea mayor al del resto de los archivos del volumen, o si no nos interesa directamente resguardar el resto.
- **Mail**: Se explica por sí solo. Permite hacer un backup del correo electrónico. De todas formas, según pudimos observar, parece estar limitado a aquel sincronizado con el programa de e-mail por defecto

[7]Un *volumen*, a breve, es un área de almacenamiento accesible con un sistema de archivos único, accesible por el sistema operativo.

de Windows. Como en nuestro caso no lo tenemos configurado, devuelve un mensaje de error.

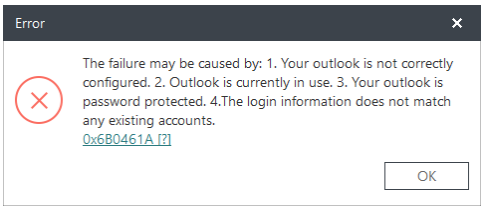


Figura 3.5: Error al intentar acceder a la opción de backup de *Mail*

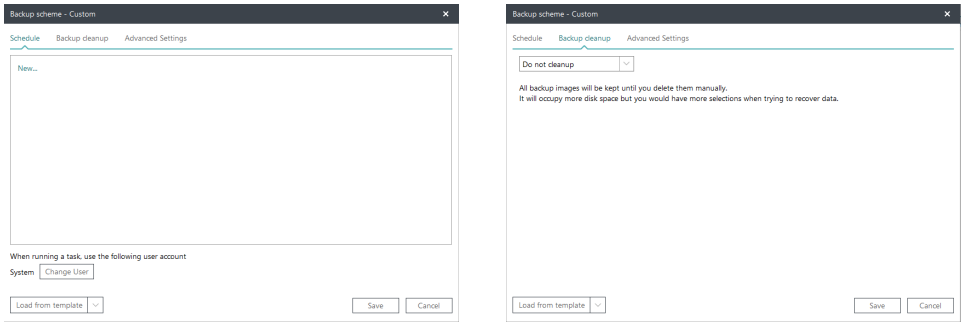
- **ExchangeDB** y **ExchangeMail**: Están relacionados al backup de información de *Exchange*, un servicio ofrecido por *Microsoft* para el manejo de correo electrónico, calendario y contactos para empresas, accesible a través de múltiples plataformas. Como claramente no tenemos acceso a este servicio, no podremos probar esta función.
- **SQL**: Permite el backup de las bases de datos de tipo *SQL* que tengamos corriendo en un servidor dentro de nuestro sistema. Este tipo de protocolo para el almacenamiento y administración de bases de datos es muy común en las distintas empresas, por lo que es bueno que el programa disponga de esta opción.

En las pestañas de la ventana que si podemos probar (*Disk/Volume* y *File*), a su vez tenemos distintas cosas que podemos configurar.

Están las cosas básicas, como el destino de la copia de seguridad, que por defecto es el establecido en la instalación (sin modificar nada es `C:\My Backups`), y tanto el nombre como la descripción del backup que estamos realizando en esta tarea, para poder identificarlo fácilmente.

Otras opciones clave, son *Schedule* (Programar), que permite establecer períodos de tiempo mediante los cuales se van a hacer backups de forma automática, y *Backup Cleanup* (Limpieza de backups) que indicaría el tiempo que tiene que pasar para que la información deje de tener validez, así el programa automáticamente borraría backups pasados esa fecha para liberar espacio.

También existen cosas más avanzadas y en detalle que se pueden configurar si entramos en *Backup options*. Iremos pestaña por pestaná con una descripción breve de las posibilidades en cada una:



(a) Schedule (b) Backup Cleanup

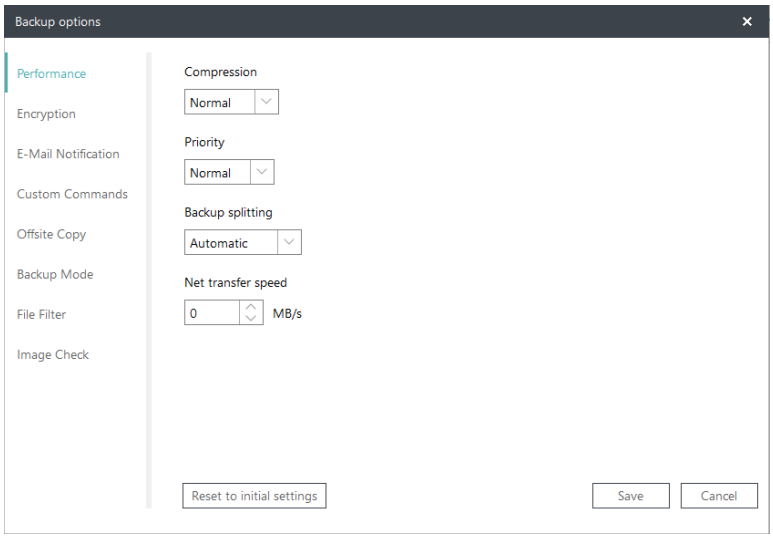
Figura 3.6: Opciones *Schedule* y *Backup Cleanup*

- *Performance* (Rendimiento): Podemos definir la prioridad del backup, así el programa puede determinar cuál hacer primero en caso de haber conflictos con varias tareas corriendo en simultáneo (por ejemplo, puede suceder con backups programados), al igual que la división del backup en partes si se lo necesita, y el nivel de compresión de la información, para reducir espacio ocupado.
- *Encryption* (Encriptación): Permite establecer una contraseña para el cifrado de los datos. Según investigamos, el programa usa el algoritmo AES256.
- *E-mail Notification* (Notificación por e-mail): Se puede establecer para que el software notifique mediante correo electrónico a direcciones designadas, cada vez que se hace un backup, tanto de forma correcta o no. Nuevamente, puede ser útil principalmente con backups programados.
- *Custom Commands* (Comandos personalizados): Si se desea, se pueden ejecutar comandos de consola tanto antes como después de un backup. Puede ser útil si se quieren realizar otras tareas antes o después de dichos, que requieran programas o aplicaciones externas.
- *Offsite Copy* (Copia fuera del lugar): Esto es clave si queremos cumplir con la estrategia 3-2-1 mencionada en la teoría, en la que se aconseja de tener una copia de los datos fuera del lugar donde originalmente están almacenados. Con esta opción, podemos ingresar las credenciales de un servidor FTP en el cual hacer una copia del backup realizado.
- *Backup Mode* (Modo del backup): Permite configurar algunas opciones varias referidas al backup. Según si estamos haciendo un backup en

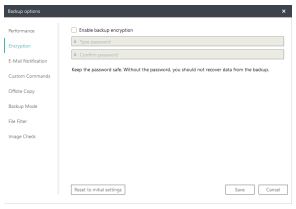
modo *Disk/Volume* podemos hacerlo sector por sector, o si hacemos en el modo *File* de mantener las opciones de seguridad que pueda tener cada archivo, entre otros detalles.

- *File Filter* (Filtro de archivos): Permite agregar excepciones a los archivos de los cuales hacer backup, como archivos de determinado formato o en cierta ruta de entre todas las incluidas.

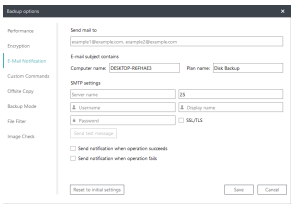
- *Image Check* (Chequeo de imagen): Otro de los puntos claves mencionados en la teoría es la revisión del funcionamiento correcto de las herramientas y de la integridad de los backups realizados para que no existan inconvenientes a la hora de restaurar alguno. Si bien sería ideal que el programa permita con regularidad esta opción, en esta sección si permite habilitar de chequear la integridad del backup realizado apenas finalizó.



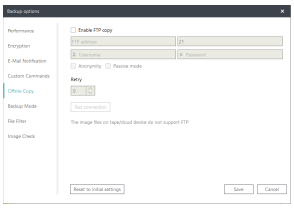
(a) Ventana de *Backup Options* (Pestaña *Performance*)



(b) *Encryption*



(c) *E-mail Notification*



(d) *Offsite Copy*

Figura 3.7: Algunas de las pestañas del apartado *Backup Options*

Cubiertas todas las opciones de la ventana `NEW TASK`, cubriremos algunas más en la barra superior de la pantalla principal presentada al comienzo del apartado.

Tenemos por supuesto que tener una forma de recuperar los archivos. Entonces uno de los botones es *Recovery*, en el cual podemos elegir desde un archivo de backup, la ubicación a la cual restaurar su contenido. Si se hace uso del botón mientras se selecciona una tarea, tendremos las opciones mencionadas, ya asumiendo que estamos intentado restaurar el backup correspondiente a dicha, pudiendo seleccionar entre los distintos backups relacionados disponibles hechos a lo largo del tiempo (se visualizará mejor al poner a prueba el programa con el backup de un pendrive en la próxima sección). Similarmente, la opción *System Transfer* permite lo mismo pero con backups hechos sobre sistemas completos.

Otra de las opciones que aparecen si seleccionamos una tarea en específico, es la de hacer un backup de la tarea, y del botón sale un menú desplegable en el cual nos permite elegir entre un backup completo, diferencial e incremental, los cuales ya definimos al comienzo del TP (también a demostrar mejor en la próxima sección). Fuera de esto, el resto de las botones nuevos que aparecen permiten visualizar detalles de la tarea, editar sus opciones y exportarla a un archivo, para si luego se desea pasarse a otra computadora con el mismo programa.

3.2.1.2 Logs

Nos muestra una lista de todas las distintas acciones realizadas con el programa, como crear tareas, ejecutar backups, etc, a lo largo del tiempo, y si se sucedieron correctamente o no. Si se desea, con el botón en la esquina superior *Export Logs* se pueden exportar los registros a un archivo `.csv`.

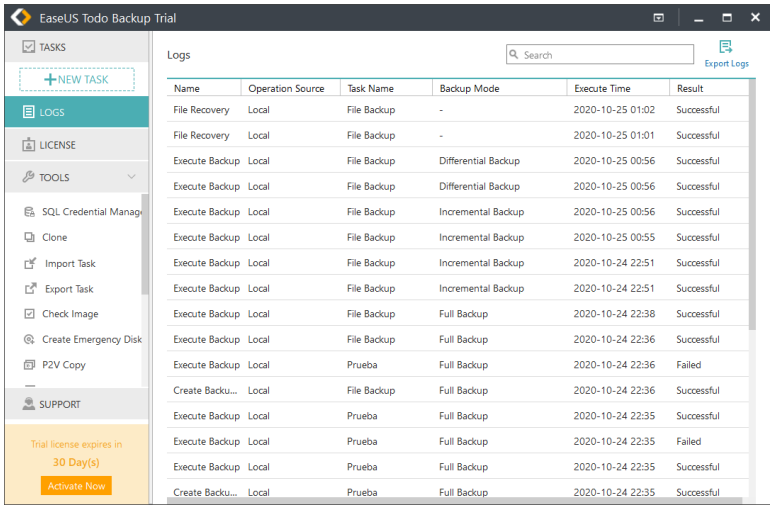


Figura 3.8: Apartado *Logs*, luego de realizar múltiples operaciones de prueba

3.2.1.3 License

Aquí se muestran las múltiples licencias de uso del software, en caso que se dispongan múltiples si se utilizan en múltiples computadoras, por ejemplo.

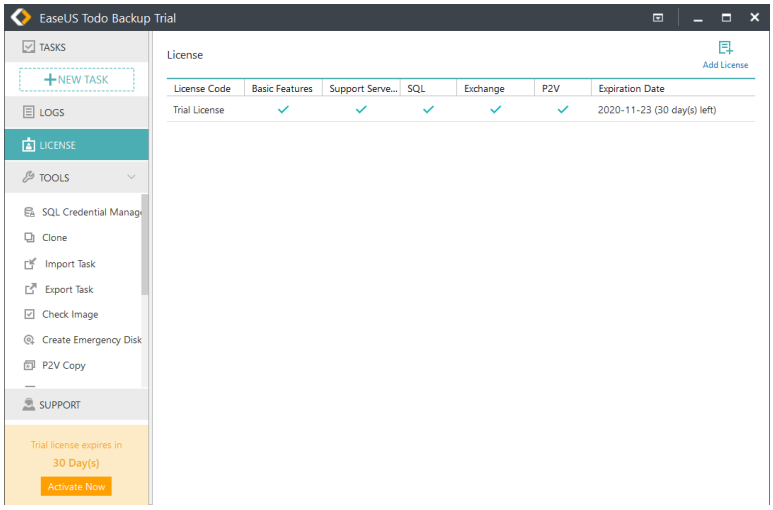


Figura 3.9: Apartado *License*

3.2.1.4 Tools

No es un apartado de por sí, si no que contiene una lista de múltiples herramientas individuales adicionales que incluye el programa. Repasaremos brevemente por ellas:

- *SQL Credential Manager* (Administrador de credenciales de SQL): Permite administrar las claves de autenticación en las bases de datos SQL en el sistema, de existir.
- *Disk/Partition Clone* (Clonar discos/particiones): Como lo dice su nombre, permite hacer copias completas de volumenes en otros accesibles en el sistema.
- *Import Task* y *Export Task*: (Importar Tarea y Exportar Tarea): Antes dijimos que podíamos exportar una tarea, por lo que a partir de estas utilidades podríamos volver a importarlas y también tener otra forma de exportarlas.
- *Check Image* (Verificar Imagen): Otra forma de verificar la imagen de un backup realizado.
- *Emergency Disk* (Disco de emergencia): Permite crear un CD/DVD o pendrive USB booteable con Windows o Linux, en caso de que se requieran recuperar backups almacenados localmente y el SO no sea capaz de bootear.
- *P2V Recovery* y *P2V Copy* (Recuperación de P2V y Copia P2V): P2V, de sus siglas en inglés, *Physical-to-Virtual* es un proceso mediante el cual se migra el sistema operativo y los datos a un entorno virtual. Esto permite correr múltiples aplicaciones en simultáneo dentro de una misma computadora, de forma aíslada. Por lo tanto, esta herramienta nso permitiría transferir los datos a un entorno virtual como también recuperarlos del mismo.
- *Tape Manager* (Administrador de cassettes): Permite copiar datos a cassettes, siendo útil para almacenamiento de información a largo plazo, dada su comprobada confiabilidad dado que son uno de los que menos se degradan a lo largo del tiempo.
- *Mount/Unmount* (Montar/Desmontar): Permite, como lo dice el nombre, montar y desmontar backups realizados (formato `.pbd` como discos, para poder verificar su validez y sus contenidos.

- *iSCSI Initiatior* (Iniciador iSCSI): El protocolo `iSCSI` permite a una computadora conectarse a un dispositivo `SCSI`[8] en la misma red.
- *Enable PreOS* (Activar PreSO): Habilita un menú de booteo cuando la compu inicia, antes que Windows, para tener una posibilidad de recuperar los datos en caso de que este último falle.
- *Enable PXE* (Activar PXE): PXE, de sus siglas en inglés, *Preboot Execution Environment*, es un entorno que permite arrancar e instalar el sistema operativo de una computadora a distancia. En este caso, permite arrancar el entorno *PreSO* hablado a través de la red.
- *Refresh Disks* (Actualizar discos): Simplemente permite actualizar para el programa la lista de discos, particiones, volumenes accesibles por la computadora. Útil si recientemente hemos conectado alguno nuevo y no está siendo reconocido por el software de backup, sin necesidad de reiniciarlo.

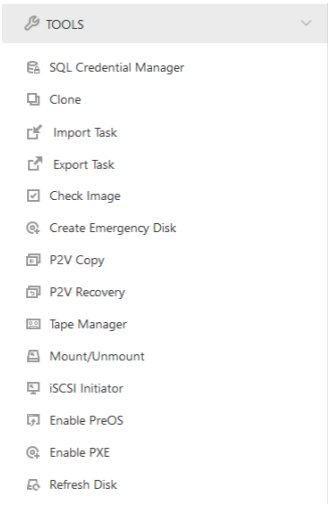


Figura 3.10: Lista de herramientas (*Tools*) completa en el menú izquierdo

[8] `SCSI`, de sus siglas en inglés, *Small Computer System Interface*, es una interfaz estándar para la transferencia de datos entre distintos dispositivos del bus de una computadora.

3.2.2. Haciendo backup de un pendrive

3.2.2.1 Diferencias entre los backups

Para probar y poner en práctica algunas de las funciones provistas por el programa, decidimos hacer un backup de un pendrive. Antes de comenzar, metimos en él archivos variados para tener algo con lo que trabajar.

Lo primero que tratamos de hacer fue un backup del pendrive tomando a este como un volumen, para lo que debimos crear una nueva tarea y, en ella, seleccionar el volumen deseado, lugar de destino y nombre de la copia. Lamentablemente este intento no funcionó. Probamos borrar la task y reiniciar el programa, crear tareas nuevas similares, pero tampoco lograron su objetivo. Hacer una copia del pendrive como volumen no dio frutos.

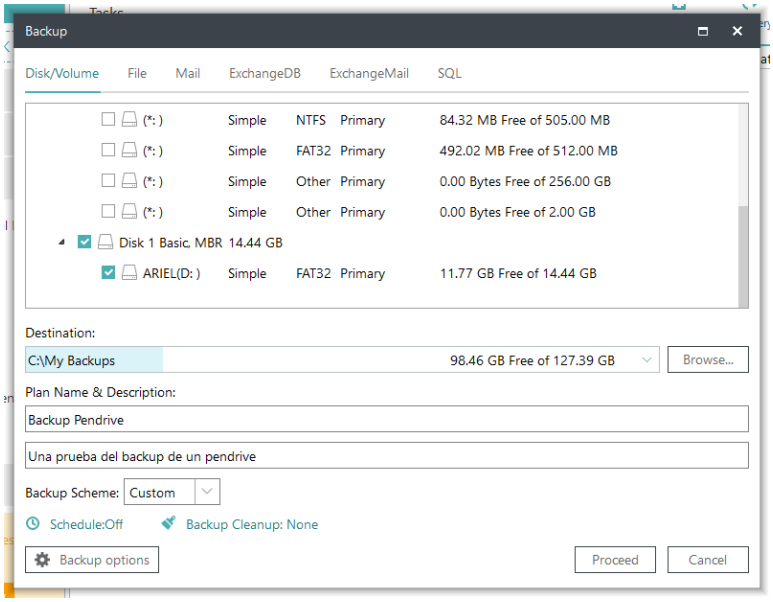


Figura 3.11: Creación de un backup del pendrive como volumen

Es por eso que se nos ocurrió hacer un backup de la unidad entera como "File". Para eso, sin borrar ninguna anterior, creamos una nueva task y nos dirigimos a la segunda pestaña, que dice "File". En ella, seleccionamos el pendrive mencionado y, debajo, en la descripción, al igual que en la tarea anterior, indicamos el destino y el nombre del nuevo backup. Esta prueba

se realizó con éxito. Para verificar que todo había salido bien, nos dirigimos a la carpeta que indicamos como destino y encontramos un archivo "`File Backup_20201024_Full_v1`", que tenía guardados los mismos archivos que el pendrive.

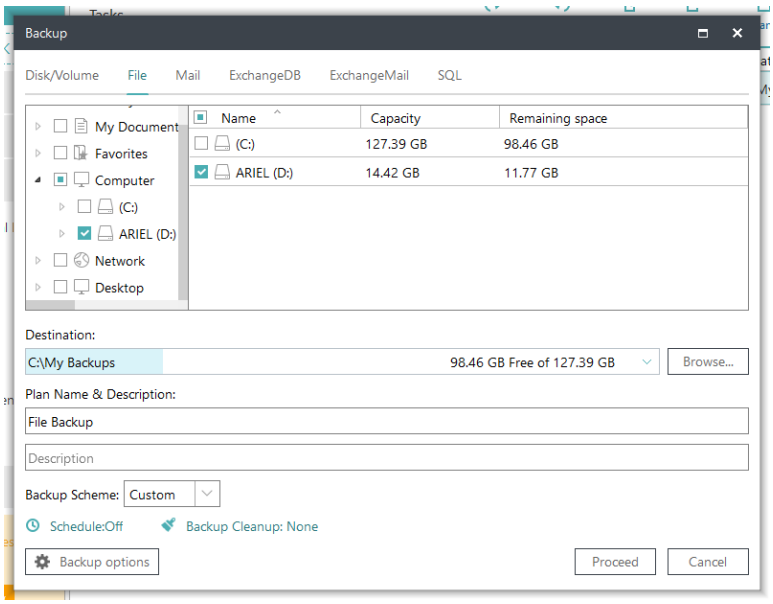
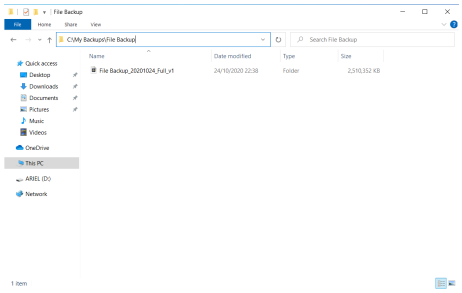
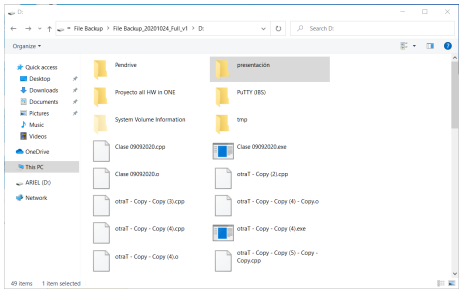


Figura 3.12: Creación de un backup del pendrive como archivo



(a) Archivo de backup



(b) Contenido del backup

Para probar que los backups se realizaran correctamente, borramos algunos archivos ejecutables que se encontraban en el pendrive. Luego realizamos, sobre este, un backup incremental y lo comparamos con el completo realizado anteriormente: el más pequeño era el incremental. Además verificamos su contenido: este último no tenía los archivos que habían sido borrados,

mientras que el completo, sí.

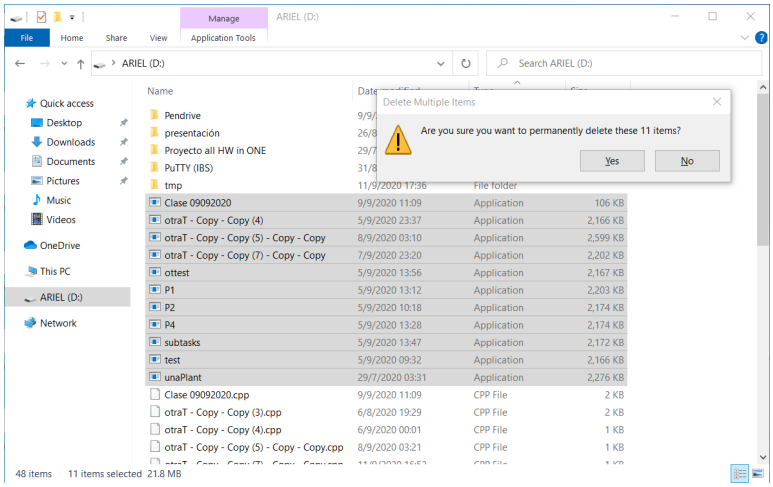


Figura 3.14: Borramos algunos archivos del pendrive

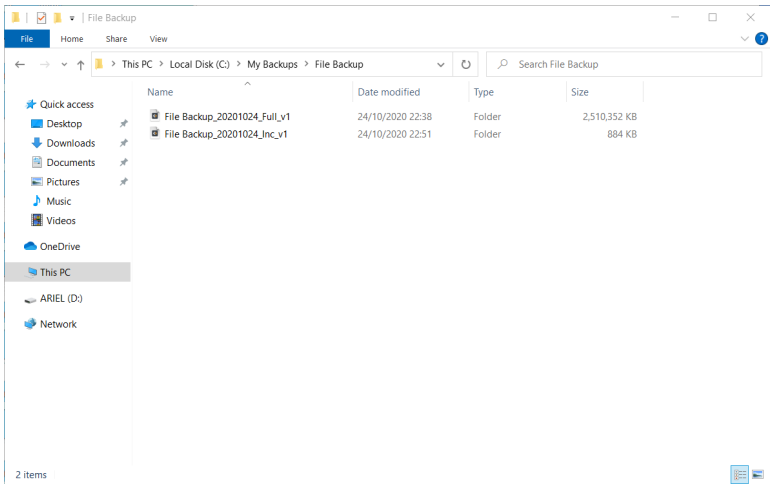
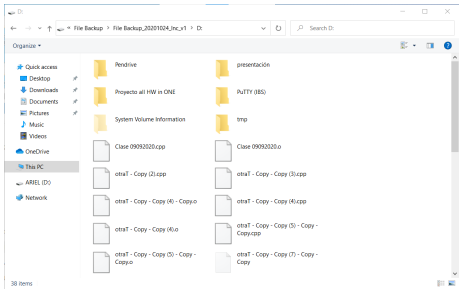
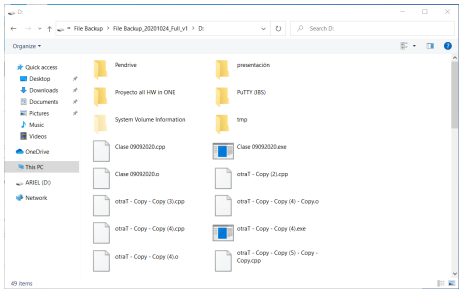


Figura 3.15: Contenido de la carpeta “File Backup”, con los archivos de copia generados



(a) Contenido del incremental



(b) Contenido del completo

Lo siguiente que hicimos fue realizar un backup diferencial. Pero antes, para poder observar su funcionamiento, creamos en el pendrive un archivo "`prueba.txt`". [9] Luego procedimos a hacer el backup diferencial y comparamos el tamaño con el de los otros, así como su contenido: tenía los mismos datos que el incremental y, además, el archivo nuevo que habíamos creado (como ya explicamos al principio de este trabajo, el diferencial copia todos los cambios desde el último completo hecho).

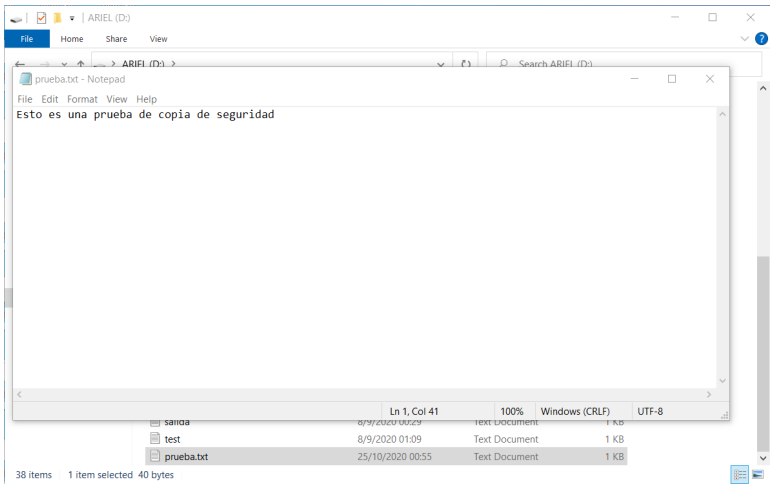


Figura 3.17: Creación de `prueba.txt`

[9] Al crear el archivo de prueba, olvidamos que no teníamos las extensiones habilitadas (es decir, se mostraba el nombre del archivo, pero no su extensión) y, por ese motivo, el archivo se terminó llamando "prueba.txt.txt".

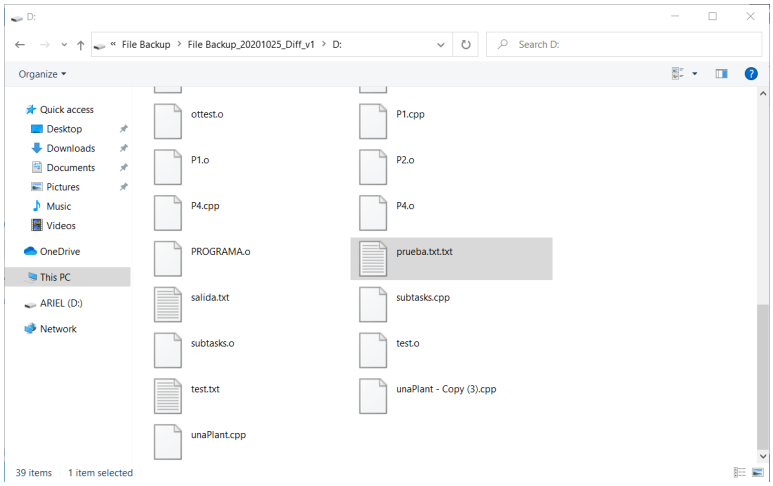


Figura 3.18: Contenido del backup diferencial

Lo próximo fue realizar otro backup incremental. No cambiamos los contenidos del pendrive para hacerlo. Una vez hecha esta copia, volvimos a observar sus características y a compararlas con las de las anteriores: el incremental era más chico que el diferencial.

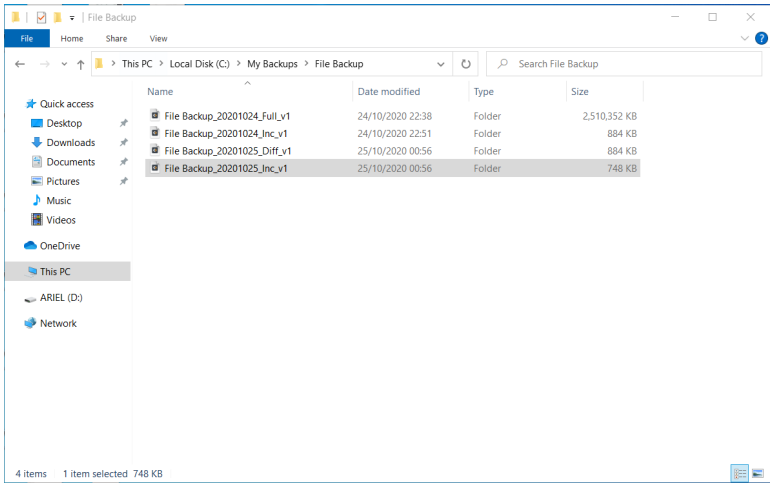
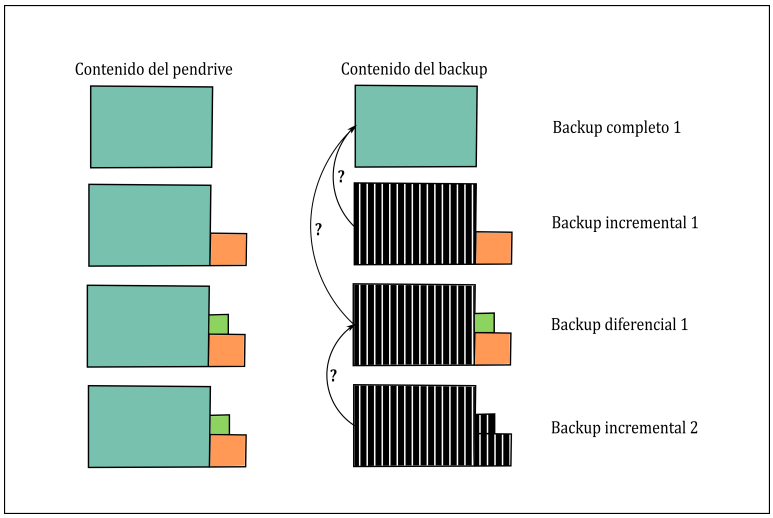


Figura 3.19: Contenido de la carpeta “File Backup”, con los archivos de copia generados

A modo de resumen y para observar mejor las diferencias entre los tipos de backups, prestemos atención a la siguiente imagen: el primer backup

que hicimos fue completo, con un tamaño de 2 510 352 KB. Esta primera copia contenía `todos` los archivos del pendrive. Luego de realizar algunos cambios en los archivos originales, hicimos un backup incremental, que pesó 884 KB y contenía `sólo los ficheros modificados`. La siguiente copia de seguridad que hicimos fue diferencial, contenía los archivos modificados (lo mismo que el incremental anterior) y, además, el archivo prueba.txt que habíamos creado. Por último, hicimos otro incremental, que no contenía nada: no se habían efectuado cambios entre esta copia y la anterior.



3.2.2.2 Recuperación de datos

Otra función que decidimos probar fue la de restaurar datos: obtener los archivos que están guardados en una copia de seguridad y reintegrarlos en nuestro sistema en el estado en que estaban al momento de hacer el backup. Para eso, hicimos click derecho sobre la tarea deseada, y, en el menú de opciones desplegado, elegimos "`Recuperación`". Esa opción abrió una nueva ventana como la de la imagen. Le pedimos que dejara los datos recuperados en una carpeta del escritorio. Luego, corroboramos que los archivos se hubieran restaurado correctamente.

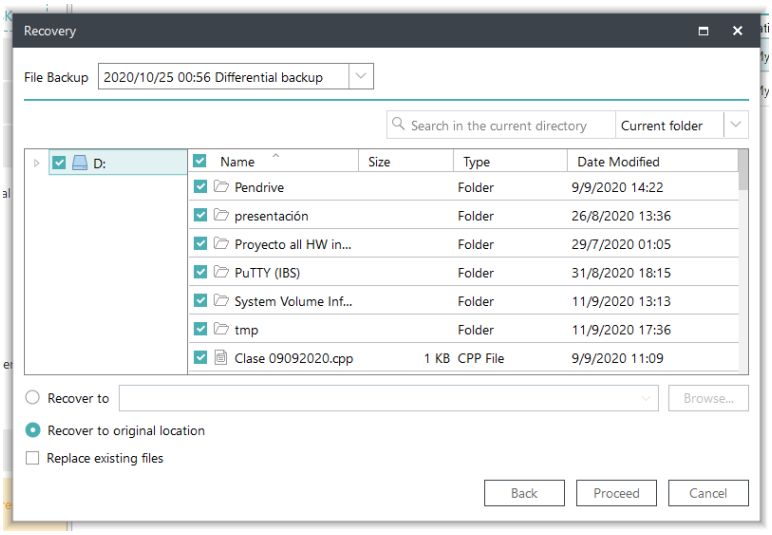


Figura 3.20: Opción para la recuperación de datos

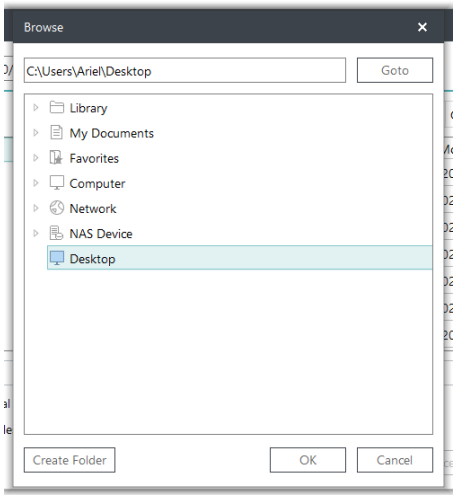
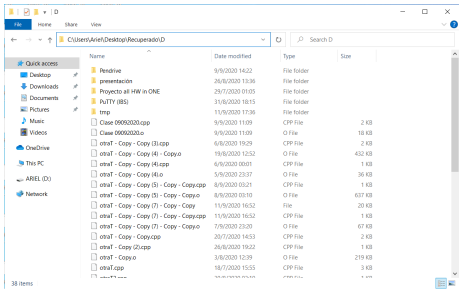
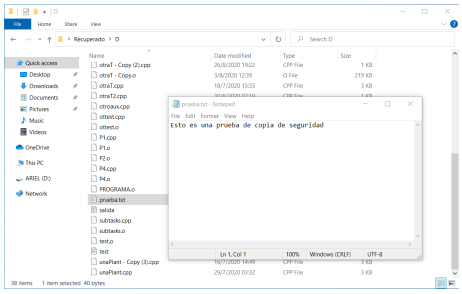


Figura 3.21: Elección de la ubicación de los datos recuperados



(a) Contenido de la carpeta con los archivos recuperados



(b) Corroboramos que los archivos no se encontraran dañados

3.2.2.3 Detalles de la tarea

Para ver los detalle de una tarea deseada, hicimos click derecho sobre ella y seleccionamos la opción "`Detalles`" del menú desplegado.

Los detalles de la tarea se organizan en tres pestañas:

- La primera, `información básica`, contiene datos como el nombre de la misma, tipo de backup (si es por volumen, por archivo, etc), dónde se guardan las imágenes y si existen programas para crear o eliminar copias.
- La segunda pestaña brinda información sobre las `imágenes de copia de seguridad`: fecha de creación, tipo (diferencial, incremental o completa), ubicación y nombre. Se pueden hacer diferentes imágenes en una misma tarea, como hicimos en este trabajo. De esta manera, las distintas copias se van comparando y complementando entre sí: la diferencial respecto a una completa y la incremental respecto a cualquier otra realizada, pero siempre de la misma tarea.
- La última pestaña, `detalles del backup`, nos informa sobre procesos que están corriendo en este momento relacionados con esta tarea.

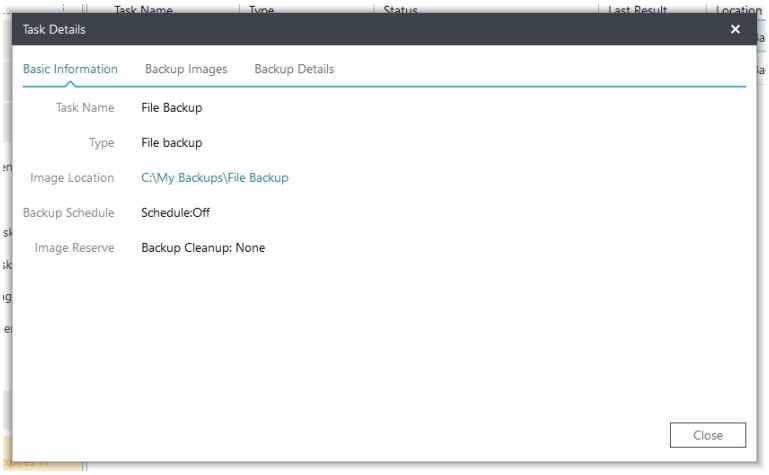


Figura 3.23: Información básica

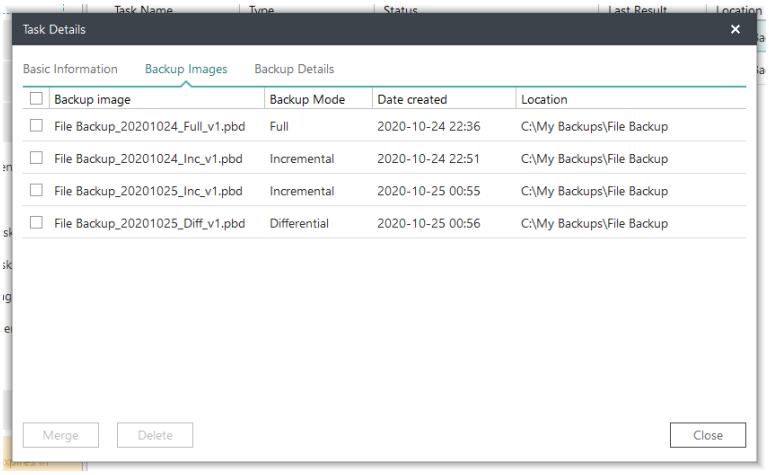


Figura 3.24: Imágenes de copia de seguridad

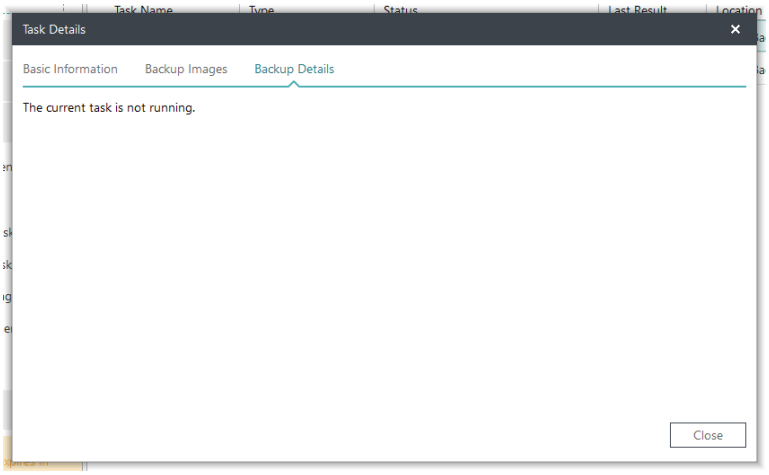


Figura 3.25: Detalles de la copia de seguridad

3.2.2.4 Comprobar imagen

Por útimo, probamos la función de comprobar la imagen. Esta utilidad nos comunica la validez de una copia de seguridad. Esta información es importante porque, si una copia se encuentra dañada, es muy probable que los datos no se puedan recuperar. Para averiguar eso, nos dirigimos a la lista de tareas, hicimos click derecho y luego seleccionamos "`Comprobar imagen`" sobre la deseada. Los resultados pueden observarse en la figura 3.27.

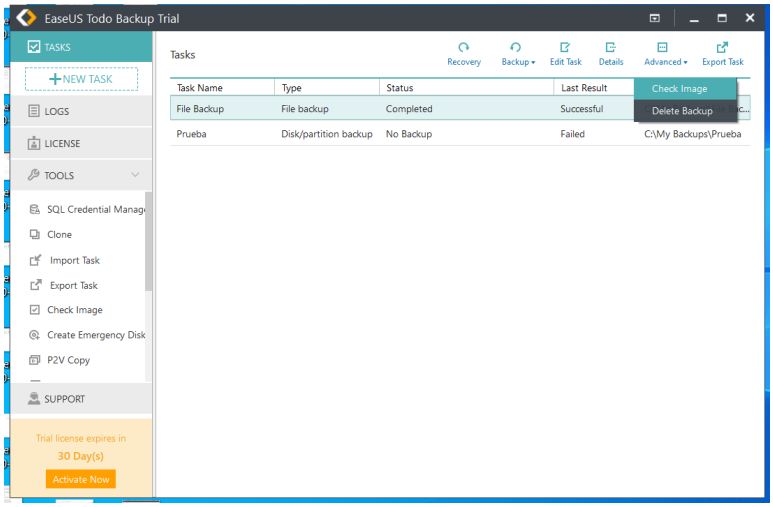


Figura 3.26: Elegimos la opción `comprobar imagen`

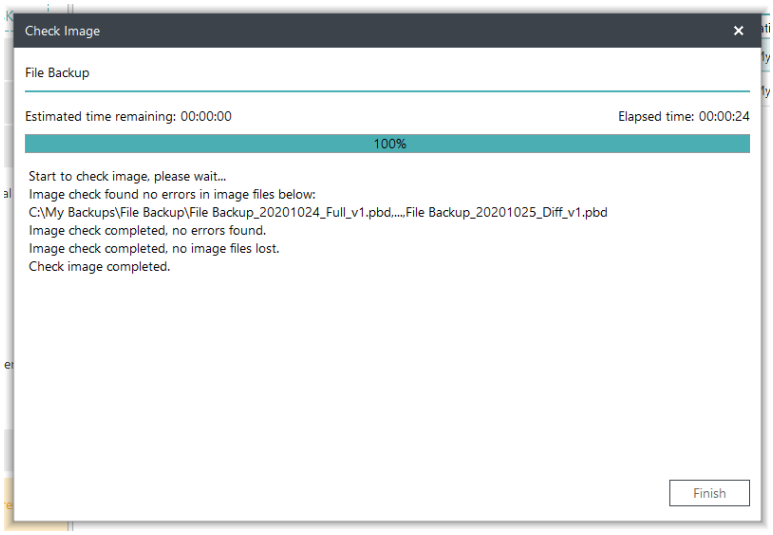


Figura 3.27: Imagen comprobada

4. Comandos usados

A continuación se encuentran todos los comandos utilizados en este trabajo, correspondientes a las imágenes presentadas.

```
sudo apt install rsync
```

[2.1]

```
sudo fdisk -l
```

[2.3]

```
sudo mkfs.ext4 /dev/sdb
```

[2.6]

```
sudo mkdir /mnt/sdb
```

[2.7]

```
mount | grep ‘‘sdb’’
```

[2.9]

```
touch Prueba
ls -l
```

[2.10]

```
sudo chown martin:martin sdb
ls -l
touch Prueba
ls sdb
```

[2.11]

```
ls -R C
```

[2.12]

```
rsync ~/C /mnt/sdb/Backup
ls /mnt/sdb/Backup
rsync -r ~/C /mnt/sdb/Backup
ls /mnt/sdb/Backup
```

[2.13]

```
ls -Rl /mnt/sdb/Backup
```

[2.14]

```
rsync -r ~/C /mnt/sdb/Backup/
ls -R /mnt/sdb/Backup | head -3
rm -R /mnt/sdb/Backup/*
rsync -r ~/C/ /mnt/sdb/Backup/
ls -R /mnt/sdb/Backup | head -4
```

[2.15]

```
rsync -rv ~/C/ /mnt/sdb/Backup/
echo ‘‘Hola’’ > /mnt/sdb/Backup/Prueba.txt
ls /mnt/sdn/Backup
rsync -r --delete ~/C/ /mnt/sdb/Backup/
ls /mnt/sdn/Backup
```

[2.16]

```
rsync -avr ~/C /mnt/sdb/Backup
```

[2.17]

```
ls -lR /mnt/sdb/Backup
```

[2.18]

```
rsync -avr ~/C /mnt/sdb/Backup
```

[2.19]