



# Curso de Ciberseguridad

Análisis Forense en Windows

Análisis Forense Informático



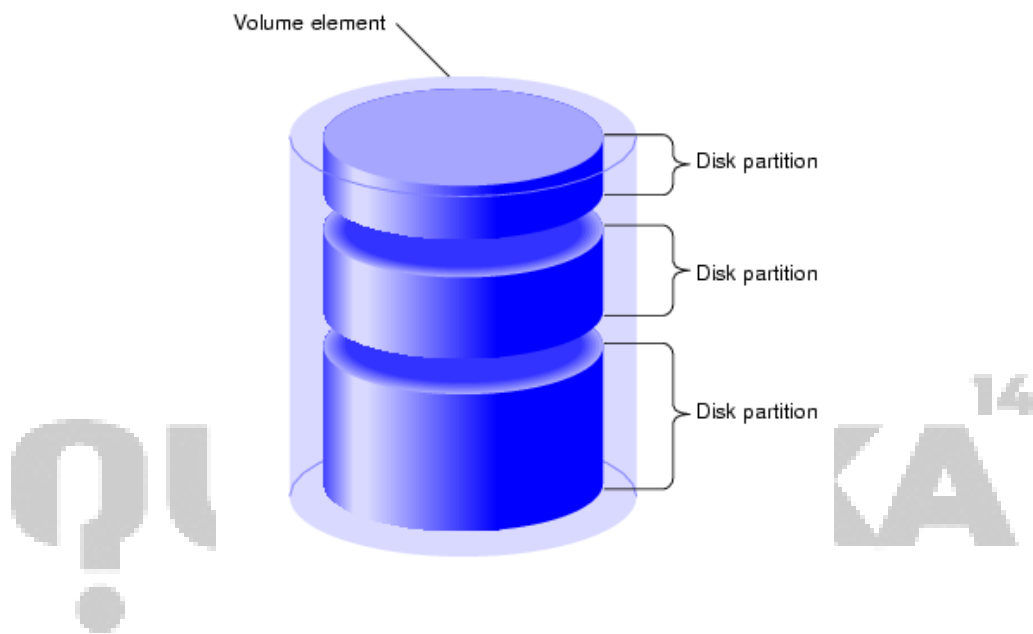
Particionado.....	3
MBR.....	4
GPT.....	7
Sistemas de archivos en Windows.....	11
FAT.....	11
NTFS.....	13
MFT .....	15
Atributos MFT .....	20
Fechas.....	24
\$LOGFILE / USNJRNL .....	31
ADS (Alternante Data Stream) .....	36
\$INDEX Records .....	40
Shadow Copies .....	42
¿Qué ocurre cuando se borra un fichero mediante el sistema operativo?.....	46
Recuperación de datos mediante el propio sistema de archivos .....	47
ReFS.....	48
Atributos.....	50
Ficheros.....	50
Papelera de Reciclaje.....	51
Recuperación de datos mediante carving.....	52
Stream Carving vs File Carving.....	55
Timeline.....	57
Metadatos de ficheros.....	58

## PARTICIONADO

Los dispositivos de almacenamiento, ya sea un disco duro o un smartphone, su almacenamiento se divide en particiones, dentro de las particiones en grupos de sectores o cilindros físicamente contiguos.

Una de las particiones primarias puede ser designada como una partición extendida, la cual puede subdividirse en particiones lógicas.

**Para un disco duro o disco ssd, la unidad mínima de información serán los sectores.**

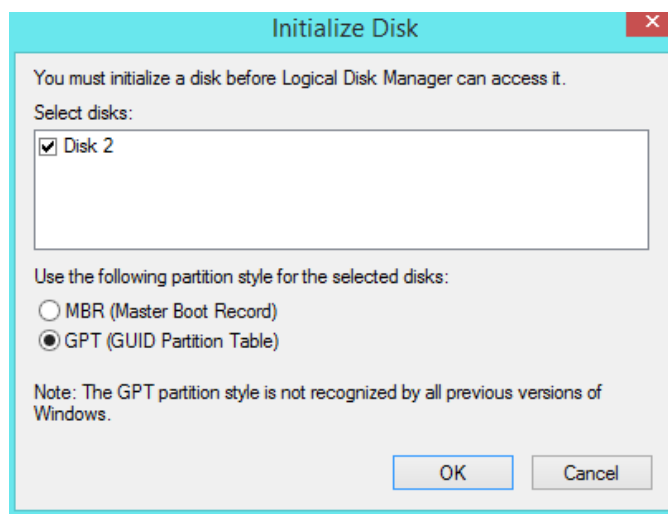


¿Por qué particionar?

- ◆ Separación: es deseable aislar los datos de las aplicaciones de los archivos del sistema operativo.
- ◆ Compartición: puede que múltiples sistemas operativos utilicen los mismos sistemas de ficheros.
- ◆ Seguridad: se desean imponer cuotas o permisos distintos en cada partición.
- ◆ Tamaño: Alguna información se mantiene constante y otras veces puede ser variable o volátil. Si una partición se llena no afectará a las demás.

Hoy en día todos los dispositivos donde se encuentre un sistema de archivos necesitan un particionamiento para saber los límites de este. Los particionados más extendidos son el MBR (Master Boot Record) y el GPT (GUID Partition Table).

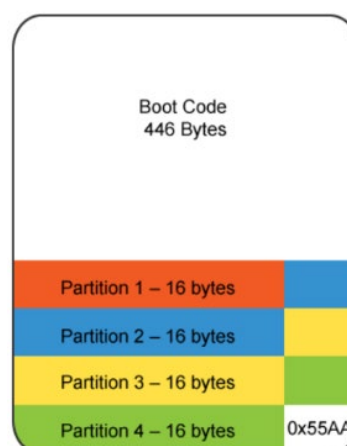
En la siguiente captura aparece el menú de Windows cuando insertamos un dispositivo de almacenamiento sin particionado.



## MBR

Máster Boot Record: alojado en el primer sector de un dispositivo y contiene tabla de particiones.

- ◆ 1 sector 512 bytes
- ◆ Un disco puede tener hasta cuatro particiones primarias.
- ◆ Tiene 64 bytes de longitud y se sitúa después de los 446 bytes del registro de arranque, y un número mágico al final (0x55AA).
- ◆ Solo una partición se marca como activa.
- ◆ Cada entrada en la tabla de particiones ocupa 16 bytes y describe las cuatro posibles particiones primarias.



Si analizamos en detenimiento como está estructurado el MBR podemos identificar:

Basic Structure of the Master Boot Record Sector			
Offsets (within sector)		Length (in bytes)	Description
<i>in Decimal</i>	<i>in Hex</i>		
000 - 445	000 - 1BD	446	Code Area
446 - 509	1BE - 1FD	64	Master Partition Table
510 - 511	1FE - 1FF	2	Boot Record Signature



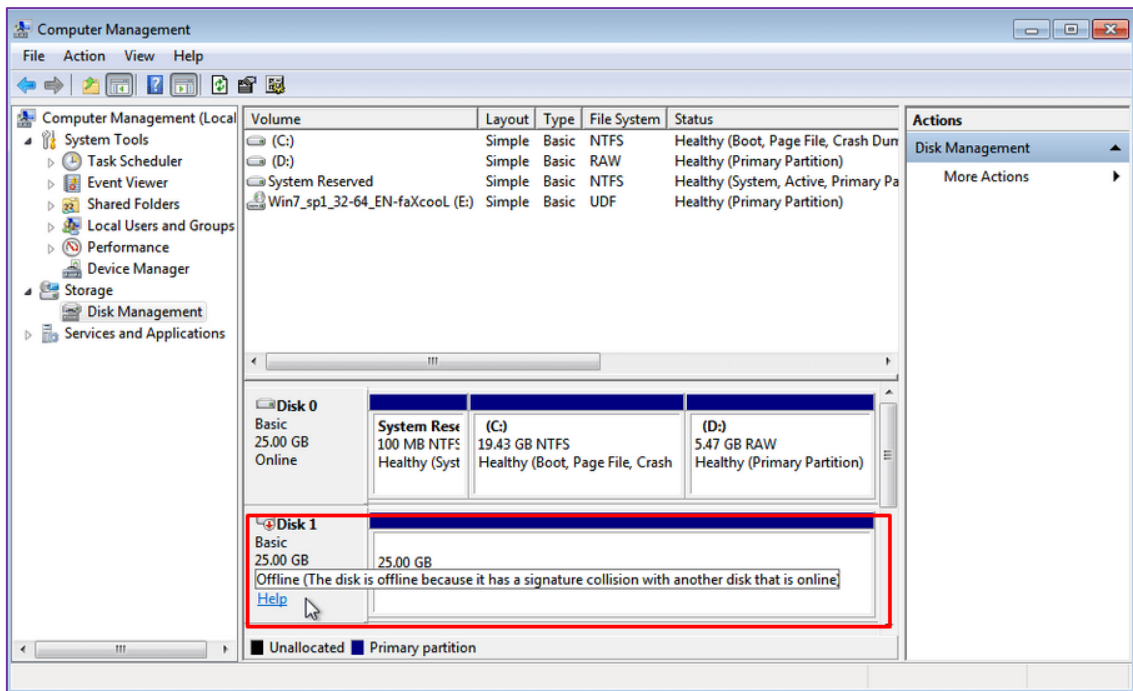
Los primeros 446 bytes son el code área, dentro de estos 446, a partir del 440 hasta el 443 aparece un número de serie o Windows Disk Signature:

33 C0 8E D0 BC 00 7C 8E	C0 8E D8 BE 00 7C BF 00	3À.Ð¼.  .À.Ø¼.  ç.
06 B9 00 02 FC F3 A4 50	68 1C 06 CB FB B9 04 00	.¹...üó«Ph...Êû¹..
BD BE 07 80 7E 00 00 7C	0B 0F 85 0E 01 83 C5 10	¾¼...~... .....Å.
E2 F1 CD 18 88 56 00 55	C6 46 11 05 C6 46 10 00	âñí...V.UÆF...ÆF..
B4 41 BB AA 55 CD 13 5D	72 0F 81 FB 55 AA 75 09	'A»ªUÍ.]r...ûUªu.
F7 C1 01 00 74 03 FE 46	10 66 60 80 7E 10 00 74	÷Á...t.þF.f`.~...t
26 66 68 00 00 00 00 66	FF 76 08 68 00 00 68 00	&fh....fÿv.h...h.
7C 68 01 00 68 10 00 B4	42 8A 56 00 8B F4 CD 13	h...h...´B.v...ôí.
9F 83 C4 10 9E EB 14 B8	01 02 BB 00 7C 8A 56 00	..Ä..ë.,...».. v.
8A 76 01 8A 4E 02 8A 6E	03 CD 13 66 61 73 1C FE	.v..N..n.í.fas.þ
4E 11 75 0C 80 7E 00 80	0F 84 8A 00 B2 80 EB 84	N.u...~.....².ë.
55 32 E4 8A 56 00 CD 13	5D EB 9E 81 3E FE 7D 55	U2ä.v.í.]ë...>þ}U
AA 75 6E FF 76 00 E8 8D	00 75 17 FA B0 D1 E6 64	ªunÿv.è...u.ú°Ñæd
E8 83 00 B0 DF E6 60 E8	7C 00 B0 FF E6 64 E8 75	è...°ßæ`è .°ÿædèu
00 FB B8 00 BB CD 1A 66	23 C0 75 3B 66 81 FB 54	.û.,»í.f#Àu;f.ûT
43 50 41 75 32 81 F9 02	01 72 2C 66 68 07 BB 00	CPAu2.ù...r,fh.».
00 66 68 00 02 00 00 66	68 08 00 00 00 66 53 66	.fh....fh....fsf
53 66 55 66 68 00 00 00	00 66 68 00 7C 00 00 66	SfUfh....fh. ..f
61 68 00 00 07 CD 1A 5A	32 F6 EA 00 7C 00 00 CD	ah...í.z2öè.. ..í
18 A0 B7 07 EB 08 A0 B6	07 EB 03 A0 B5 07 32 E4	. .ë. ¶.ë. p.2ä
05 00 07 8B F0 AC 3C 00	74 09 BB 07 00 B4 0E CD	....ð-<.t.»...´.í
10 EB F2 F4 EB FD 2B C9	E4 64 EB 00 24 02 E0 F8	.ëòôëÿ+Êädë.\$..àø
24 02 C3 49 6E 76 61 6C	69 64 20 70 61 72 74 69	\$.ÃInvalid parti
74 69 6F 6E 20 74 61 62	6C 65 00 45 72 72 6F 72	tion table.Error
20 6C 6F 61 64 69 6E 67	20 6F 70 65 72 61 74 69	loading operati
6E 67 20 73 79 73 74 65	6D 00 4D 69 73 73 69 6E	ng system.Missin
67 20 6F 70 65 72 61 74	69 6E 67 20 73 79 73 74	g operating syst
65 6D 00 00 00 63 7B 9A	AC 09 B3 3F 00 00 80 20	em...c{.¬.ª?...
21 00 07 1D 17 46 00 08	00 00 00 28 11 00 00 1D	!....F.....(...
18 46 07 FE FF FF 00 30	11 00 00 C8 6E 07 00 00	.F.þÿÿ.0...Èn...
00 00 00 00 00 00 00 00	00 00 00 00 00 00 00 00	.....
00 00 00 00 00 00 00 00	00 00 00 00 00 00 55 AA	.....Uª

Esto lo podemos verificar con la herramienta [Active Disk Editor](#), donde automáticamente nos identifica que es cada posición dentro del disco.

**Este Windows Disk Signature, es utilizado por el sistema operativo para diferenciar entre dispositivos conectados a Windows.**

¿Puede haber una colisión del Windows Disk Signature?



Sí. Como vemos en la imagen superior, esta situación se puede producir cuando se clona de manera física un dispositivo que incluye el primer sector que aloja el MBR. No se debe confundir esta firma, con el número de serie físico del disco o dispositivo de almacenamiento.

La tabla de particiones está contenida dentro del MBR y contiene registros de 16 bytes cada uno.

The standard 64-byte Primary Partition Table			
Offsets (within MBR sector)		Length (in bytes)	Contents
in Decimal	in Hex		
446 - 461	1BE - 1CD	16	Table Entry for Primary Partition # 1
462 - 477	1CE - 1DD	16	Table Entry for Primary Partition # 2
478 - 493	1DE - 1ED	16	Table Entry for Primary Partition # 3
494 - 509	1EE - 1FD	16	Table Entry for Primary Partition # 4

En los registros se especifica de que tipo es la partición.

Structure of a 16-byte Partition Table Entry		
Relative Offsets (within entry)	Length (bytes)	Contents
0	1	Boot Indicator (80h = active)
1 - 3	3	Starting CHS values
4	1	Partition-type Descriptor
5 - 7	3	Ending CHS values
8 - 11	4	Starting Sector
12 - 15	4	Partition Size (in sectors)

Tipos de particiones:

- ◆ 07: NTFS, EXFAT
- ◆ 0B/0C: FAT32
- ◆ 82: Linux Swap
- ◆ 83: Linux filesystem



La imagen anterior indica donde empieza la partición (sector), el tipo y el tamaño. El sistema de partición MBR es utilizado por las viejas BIOS.

## GPT

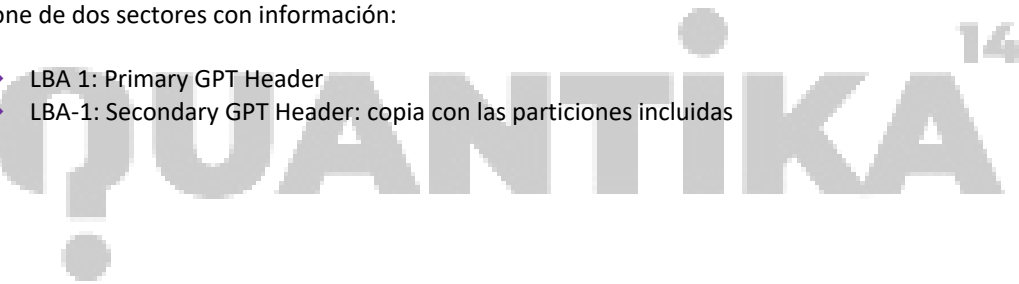
GPT viene a sustituir MBR para mantener las nuevas BIOS (UEFI). Al igual que en el MBR empieza en el sector 0 del dispositivo para mantener compatibilidad con los sistemas con BIOS.

- ◆ MBR = 32bit, GPT = 64bit
- ◆ GPT tiene una table de particiones localizada en último LBA n-1 y un backup del GPT header en el último LBA n
- ◆ MBR solo puede tener un máximo de 4 particiones, **sin embargo, GPT tiene hasta 128**
- ◆ La table de particiones en MBR permite solo para
- ◆ MBR Partition Table allows for up to 2.2 TB
- ◆ GPT allows for up to 9.4 ZB
- ◆ GPT allows for each partition to have a 36 character Unicode name
- ◆ GUIDs are stored as 128-bit values, and are displayed as 32 hexadecimal digits

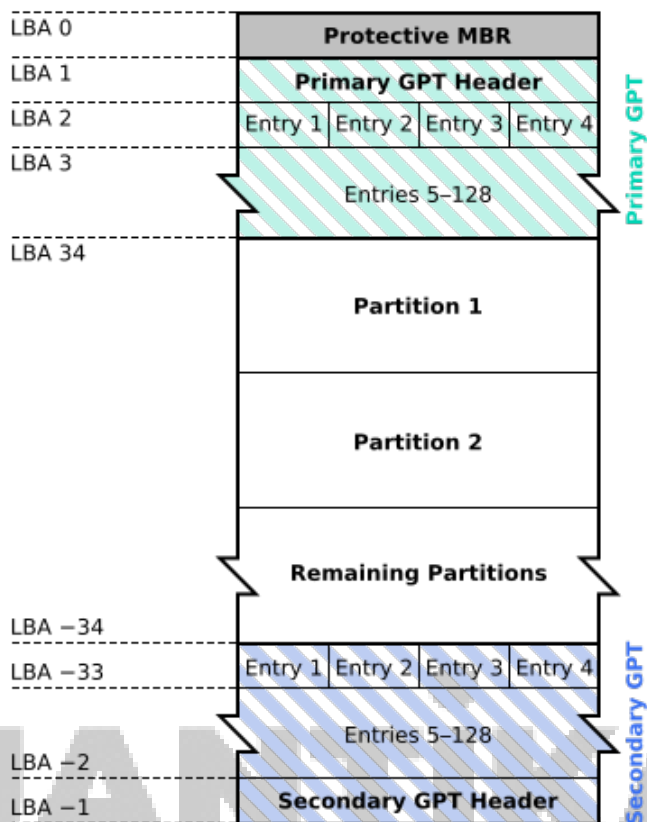
Se mantiene el MBR en el sector 0 pero conteniendo una única partición con identificador de partición 0xEE. Se le llama Protective MBR, con el área de código a cero.

Dispone de dos sectores con información:

- ◆ LBA 1: Primary GPT Header
- ◆ LBA-1: Secondary GPT Header: copia con las particiones incluidas



## GUID Partition Table Scheme





El protective MBR que aparece en el sector 0 está formado de la siguiente manera:

Offset	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
0	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
16	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
32	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
48	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
64	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
80	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
96	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
112	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
128	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
144	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
160	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
176	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
192	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
208	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
224	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
240	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
256	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
272	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
288	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
304	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
320	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
336	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
352	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
368	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
384	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
400	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
416	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
432	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
448	02	00	EE	FF	FF	FF	01	00	00	00	FF	FF	FF	FF	00	00
464	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
480	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
496	00	00	00	00	00	00	00	00	00	00	00	00	00	00	55	AA

GUID Protective MBR	
Bytes	Description
0-440	Unused by UEFI systems
440-443	Unused and set to Zero
444-445	Unused and set to Zero
446-509	MBR partition records that only have one entry pointing to the EFI Partition
510-511	Set to AA55
512	The rest of the logical block, if any, is reserved. Set to Zero

La cabecera GPT se encuentra en el LBA 1

The GPT Header data is found only in the first 92 bytes of this zero-padded sector.																
Absolute (LBA) Sector 1 (Cylinder 0, Head 0, Sector 2)																
Offset	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0000	45	46	49	20	50	41	52	54	00	00	01	00	5C	00	00	00
0010	C8	03	40	5B	00	00	00	00	01	00	00	00	00	00	00	00
0020	EF	FF	3F	00	00	00	00	00	22	00	00	00	00	00	00	00
0030	CE	FF	3F	00	00	00	00	00	5E	86	90	EF	D0	30	03	46
0040	99	3D	54	6E	B0	E7	1B	0D	02	00	00	00	00	00	00	00
0050	80	00	00	00	80	00	00	00	1E	C5	0F	B7				
	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F

Siempre empieza con el Sting "EFI PART": 45 46 49 20 50 41 52 54 -> Little endian

DISKGUID 16 Bytes : 5E 86 90 EF D0 30 03 46 99 3D 54 6E B0 E7 1B 0D

¿Cómo esta distribuidas cada entrada de partición en GPT?

Offset	Longitud	Contenido
0 (0x00)	16 bytes	Tipo de partición GUID
16 (0x10)	16 bytes	GUID único de partición
32 (0x20)	8 bytes	Inicio LBA ( <i>little endian</i> )
40 (0x28)	8 bytes	Fin LBA (inclusive, generalmente impar)
48 (0x30)	8 bytes	Indicadores (p.ej. bit 60 denota sólo lectura)
56 (0x38)	72 bytes	Nombre de partición (36 caracteres <i>UTF-16LE</i> )
	128 bytes totales	

GUIDs tipos de partición

Windows	Microsoft Reserved Partition	E3C9E316-0B5C-4DB8-817D-F92DF00215AE
	Partición de datos básica	EBD0A0A2-B9E5-4433-87C0-68B6B72699C7
	Logical Disk Manager Partición de metadatos	5808C8AA-7E8F-42E0-85D2-E1E90434CFB3
	Logical Disk Manager data partition	AF9B60A0-1431-4F62-BC68-3311714A69AD

Linux	Partición de datos	EBD0A0A2-B9E5-4433-87C0-68B6B72699C7
	Partición RAID	A19D880F-05FC-4D3B-A006-743F0F84911E
	Partición de intercambio (swap)	0657FD6D-A4AB-43C4-84E5-0933C84B4F4F
	Logical Volume Manager Partición (LVM)	E6D6D379-F507-44C2-A23C-238F2A3DF928
	Reservado	8DA63339-0007-60C0-C436-083AC8230908

Mac OS X	Hierarchical File System (HFS+) partition	48465300-0000-11AA-AA11-00306543ECAC
	Apple UFS	55465300-0000-11AA-AA11-00306543ECAC
	Apple RAID partition	52414944-0000-11AA-AA11-00306543ECAC
	Apple RAID partition, offline	52414944-5F4F-11AA-AA11-00306543ECAC
	Apple Boot partition	426F6F74-0000-11AA-AA11-00306543ECAC
	Apple Label	4C616265-6C00-11AA-AA11-00306543ECAC

¿Cómo podemos identificar el tipo de particionado sobre una Evidencia?



*\*Ver Video: 001/MÓD. 2 - Identificar Particionado*

## SISTEMAS DE ARCHIVOS EN WINDOWS

En este módulo dos de los sistemas de archivos que se utilizan en Windows: NTFS y FAT.

En la imagen inferior, podemos ver todos los sistemas de archivos que utiliza Windows hoy en día

FAT12/16: MSDOS, WIN95/98/NT/200

FAT32:95/2000/XP/2003/VISTA/7/8/10

ExFAT:2008/2012/2016/VISTA/7/8/10

NTFS:XP/2003/2008/2012/VISTA/7/8/10

ReFS:2012/2016

### FAT

El sistema de archivos FAT (File Allocation Table), es un sistema para almacenamiento pequeño. **La unidad mínima de este sistema de archivos es el clúster.**

Partition Boot Sector	FAT1	FAT2 (duplicate)	Root folder	Other folders and all files.
-----------------------------	------	---------------------	----------------	------------------------------

Como vemos en la imagen superior, el sistema de archivos está formado por:

- ◆ Sector de arranque con su firma 55AA, la localización vendrá indicada por el comienzo de la partición, ya sea GPT o MBR.
- ◆ FAT1: tabla que contiene la relación clúster-siguiente clúster.
- ◆ FAT2: backup de la FAT1, para proteger el volumen.
- ◆ Directorio raíz
- ◆ Espacio de datos

Tipo	Bytes dedicados en la FAT	Límite de Clúster
<b>FAT12</b>	1.5	4087
<b>FAT16</b>	2	Entre 4087 y 65526 clúster, incluidos
<b>FAT32</b>	4	Entre 65526 y 268,435,456 clúster, incluidos.

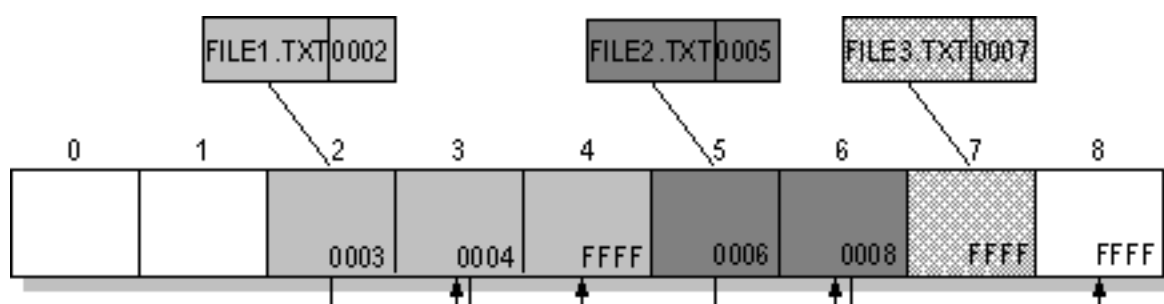
Las características de FAT son las siguientes:

- ◆ El tamaño de los bytes dedicados marcara hasta cuantos clúster es capaz de redireccionar.
- ◆ El tamaño máximo de un fichero es de 4 Gigabytes.
- ◆ El timestamp es local, es decir, lleva una zona horaria determinada
- ◆ Tiene un tamaño máximo de volumen de 32GB para FAT32

Según vemos en la imagen inferior la tabla **FAT** representa el estado del clúster y su siguiente clúster, asociado al fichero.

En esta tabla hay valores especiales para marcar:

- ◆ Clústeres erróneos (0xFFFF)
- ◆ Ultimo clúster de un fichero (0xFFFF-0xFFFF)
- ◆ Clúster usado por un archivo
- ◆ Sin usar (0x0000)



La imagen anterior muestra tres ficheros. El fichero File1.txt es un fichero que ocupa tres clúster. El fichero File2.txt, es un fichero fragmentado que ocupa tres clúster. El fichero File3.txt ocupa entero un solo clúster.

El **directorio raíz**, es un tipo especial de archivo que almacena las subcarpetas y archivos que conforman el sistema de archivos. Al ser una tabla, dispone una de una entrada para cada fichero o carpeta. La única diferencia entre este fichero y el resto es que este directorio raíz está localizado en una posición específica para FAT16 y FAT12. En FAT32 ocupa una posición como otro fichero normal.

Cada entrada puede tener:

- ◆ Nombre del fichero o carpeta (máximo 8 caracteres)
- ◆ Atributos
- ◆ Fecha de creación y hora de creación
- ◆ Fecha de modificación y hora de modificación
- ◆ Fecha de último acceso
- ◆ Dirección de la tabla FAT donde empieza el primer clúster del fichero
- ◆ Tamaño

## NTFS

El sistema de archivos NTFS fue introducido en 1993 con Windows NT 3.1 y en la actualidad se encuentra puede encontrar en distintas versiones, siendo la última la versión 5.1.

### Características de NTFS

- ◆ Journaling: guarda todos los cambios
  - \$Logfile es un fichero circular que almacena todos los cambios en cuanto a los metadatos del sistema de archivos para los ficheros. Es decir, las operaciones que hace el sistema de archivos cuando se produce una acción.
  - \$Extend\\$UsnJrnl: fichero no circular que almacena todos los cambios de ficheros y carpetas del volumen.
- ◆ Volume Shadow Copy: mantiene un histórico de los ficheros y directorios.
- ◆ Seguridad: cada fichero o directorio mantiene un descriptor de seguridad mediante listas de control de acceso.
- ◆ EFS: Encrypting File System permite cifrar ficheros o directorios
- ◆ Bitlocker: permite cifrar el volumen entero.
- ◆ Compresión
- ◆ Alternate Data Stream: añaden información extra
- ◆ Sparse Files

Un clúster es la unidad mínima de NTFS y va en función del tamaño del volumen o partición.

- ◆ Clúster 4KB (8 sectores físicos) para volúmenes de hasta 16 TB
- ◆ Clúster de 8KB para volúmenes de hasta 32 TB
- ◆ Permite hasta 16 Exabytes (16.000.000.000 Gb) por partición ( $2^{32}$  bytes).

- ◆ Los clúster son pequeños (512 a 4 Kb) poca fragmentación interna.
- ◆ Todo es tratado como atributos (incluso los datos de un fichero, el contenido de un directorio, etc.)
- ◆ Permite directamente nombres de fichero de hasta 255 caracteres.

NTFS Boot Sector	Master File Table	File System Data	Master File Table Copy
------------------------	-------------------------	------------------------	---------------------------------

El sistema de archivos NTFS está formado por:

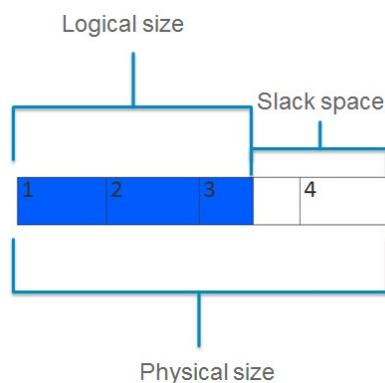
- ◆ NTFS boot sector: este sector de arranque NTFS vendrá indicado por el comienzo de la partición, ya sea GPT o MBR.
- ◆ Master File Table o \$MFT: fichero que contiene todos los ficheros y carpetas dados de alta en el sistema archivos
- ◆ Espacio clúster para almacenar archivos, este espacio puede estar usado o libre.
- ◆ Dirección de la MFT o \$MFTMirror: contiene los primeros 4 registros de la \$MFT.

En las imágenes siguientes, podemos ver cómo está organizado el NTFS boot sector, hay que tener en cuenta que el desplazamiento viene dado en clúster respecto a la partición por lo que para hacer un desplazamiento físico hay que sumar el sector actual y convertirlo a sectores.

JMP instruction	000	EB 52 90
OEM ID	003	NTFS
<b>BIOS Parameter Block</b>	<b>00B</b>	
Bytes per sector	00B	512
Sectors per cluster	00D	8
Reserved sectors	00E	0
(always zero)	010	00 00 00
(unused)	013	00 00
Media descriptor	015	248
(unused)	016	00 00
Sectors per track	018	63
Number of heads	01A	255
Hidden sectors	01C	206.848
(unused)	020	00 00 00 00
Signature	024	80 00 80 00
Total sectors	028	976.562.175
\$MFT cluster number	030	786.432
\$MFTMirr cluster number	038	2
Clusters per File Record Se...	040	246
Clusters per Index Block	044	1
Volume serial number	048	63 40 00 E4 ...
Checksum	050	0
Bootstrap code	054	FA 33 C0 8E...
Signature (55 AA)	1FE	55 AA

EB 52 90	4E 54 46 53 20	20 20 20	00 02 08 00 00	8E NTFS	.....
00 00 00 00 00 F8 00 00	3F 00 FF 00 00 28 03 00			.....ø..?..ý..(.	
00 00 00 00 00 80 00 00	FF 27 35 3A 00 00 00 00			.....ý'5:.....	
00 00 0C 00 00 00 00 00	02 00 00 00 00 00 00 00			.....	
F6 00 00 00 01 00 00 00	63 40 00 E4 4B 00 E4 C6			ó.....c@.äK..ä	
00 00 00 00 FA 33 C0 8E	D0 BC 00 7C FB 68 C0 07			...ú3Ä.ð¼.  ûhÄ.	
1F 1E 68 66 00 CB 88 16	0E 00 66 81 3E 03 00 4E			..hf.È....f.>..N	
54 46 53 75 15 B4 41 BB	AA 55 CD 13 72 0C 81 FB			TFsu.'A*Uí.r.û	
55 AA 75 06 F7 C1 01 00	75 03 E9 DD 00 1E 83 EC			U*u.+.Á..u.éý...i	
18 68 1A 00 B4 48 8A 16	0E 00 8B F4 16 1F CD 13			.h..'H.....ð..í.	
9F 83 C4 18 9E 58 1F 72	E1 3B 06 0B 00 75 DB A3			..Ä..X.rá;...u0z	
0F 00 C1 2E 0F 00 04 1E	5A 33 DB B9 00 20 2B C8			..Ä.....230+. +È	
66 FF 06 11 00 03 16 0F	00 8E C2 FF 06 16 00 E8			fý.....Äý...è	
4B 00 2B C8 77 EF B8 00	BB CD 1A 66 23 C0 75 2D			K.+Èwi, »í.f#Àu-	
66 81 FB 54 43 50 41 75	24 81 F9 02 01 72 1E 16			f.úTCPAu\$.ù..r..	
68 07 BB 16 68 70 0E 16	68 09 00 66 53 66 53 66			h.».hp..h..fsf8f	
55 16 16 16 68 B8 01 66	61 0E 07 CD 1A 33 C0 BF			U...h,.fa..í.3Ä¿	
28 10 B9 D8 0F FC F3 AA	E9 5F 01 90 90 66 60 1E			(.÷ø.úó*é....f\.	
06 66 A1 11 00 66 03 06	1C 00 1E 66 68 00 00 00			.fj...f.....fh...	
00 66 50 06 53 68 01 00	68 10 00 B4 42 8A 16 0E			..fP.Sh..h..'B...	
00 16 1F 8B F4 CD 13 66	59 5B 5A 66 59 66 59 1F			....ôí.fv[zfYfY.	
0F 82 16 00 66 FF 06 11	00 03 16 0F 00 8E C2 FF			....fý.....Äý	
0E 16 00 75 BC 07 1F 66	61 C3 A0 F8 01 E8 09 00			...u¼..faÄ ø.è..	
A0 FB 01 E8 03 00 F4 EB	FD B4 01 8B F0 AC 3C 00			û.è...ðéý'...ð<.	
74 09 B4 0E BB 07 00 CD	10 EB F2 C3 0D 0A 45 72			t.'.»...í.èðÄ..Er	
72 6F 72 20 64 65 20 64	69 73 63 6F 00 0D 0A 46			ror de disco...F	
61 6C 74 61 20 62 6F 6F	74 6D 67 72 00 0D 0A 42			alta bootmgr...B	
6F 6F 74 6D 67 72 20 63	6F 6D 70 72 69 6D 69 64			bootmgr comprimid	
6F 00 0D 0A 50 72 65 73	2E 20 43 74 72 6C 2B 41			o...Pres. Ctrl+A	
6C 74 2B 53 75 70 72 20	70 61 72 61 20 72 65 69			lt+Supr para rei	
6E 69 63 69 61 72 0D 0A	00 6C 20 74 6F 20 72 65			nciar...l to re	
73 74 61 72 74 0D 0A 00	8C 9D AD C2 00 00			start.....Ä..U	

**Slack Space:** aunque hablemos del slack space dentro de la sección de NTFS, este término se puede aplicar como veremos más adelante, también para los registros MFT.



Como sabemos, NTFS tiene la unidad mínima de información que son los clústers. Si un fichero no rellena todos los clústers que necesite para alojar su información, el espacio que sobra se llama slack space.

NTFS tiene 3 campos para indicar el tamaño de cada fichero:

- ◆ Logical
- ◆ Initialized
- ◆ Physical

Size: 122 bytes (122 bytes)

Size on disk: 4.00 KB (4,096 bytes)

## MFT

Dentro del sistema NTFS, es un fichero que contiene una tabla con todos los ficheros y directorios, incluido él mismo. Los primeros 26 registros de la tabla \$MFT son los llamados metadatos y sirven para organizar el sistema de archivos.

Name	Size	Type	Date Modified
\$Extend	1	Directory	10/14/2018 9:0...
Ficheros	1	Directory	10/14/2018 9:1...
System Volume Information	1	Directory	10/14/2018 9:0...
\$AttrDef	3	Regular File	10/14/2018 9:0...
\$BadClus	0	Regular File	10/14/2018 9:0...
\$Bitmap	120	Regular File	10/14/2018 9:0...
\$Boot	8	Regular File	10/14/2018 9:0...
\$I30	4	NTFS Index All...	10/14/2018 9:0...
\$LogFile	8,496	Regular File	10/14/2018 9:0...
\$MFT	256	Regular File	10/14/2018 9:0...
\$MFTMirr	4	Regular File	10/14/2018 9:0...
\$Secure	1	Regular File	10/14/2018 9:0...
\$TXF_DATA	1	NTFS Logged ...	10/14/2018 9:0...
\$UpCase	128	Regular File	10/14/2018 9:0...
\$Volume	0	Regular File	10/14/2018 9:0...

La imagen anterior muestra como desde FTK Imager se pueden acceder a los ficheros llamados metadatos que componen en el sistema de archivos.



Hay dos MFT, ambos indicados por el Boot NTFS sector:

- ◆ \$MFT
- ◆ \$MftMirror: al final de la partición sirve como backup, y **contiene únicamente 4 los primeros registros del MFT.**







MFT Record	Nombre	Propósito
0	\$MFT	El propio fichero MFT debe aparecer en esta tabla, ya que es un fichero más del sistema de archivos.
1	\$MFTMirror	Copia de los 4 primeros Registros
2	\$Logfile	Contiene lista de las transacciones a nivel de metadatos, realizadas a nivel del volumen NTFS
3	\$Volume	Contiene información de volumen, etiqueta y versión.
4	\$AttrDef	Tabla con los nombres de los atributos, números y descriptores
5	.	El directorio raíz
6	\$Bitmap	Indica el estado de los clúster que están en uso.
7	\$Boot	El boot sector como tal
8	\$BadClus	Clústers que el sistema de archivos como erróneos y no deben de usarse para alojar ficheros
9	\$Secure	Contiene los descriptores de seguridad para todos los ficheros del volumen
10	\$Upcase	Convierte los caracteres en minúscula en caracteres en mayúsculas de Unicode coincidentes.
11	\$Extend	Directorio que contiene cuotas, reparse point data, usnjrnl
Registro 12 a 23	Reservado	Para Futuros Usos
24	\$Quota	Fichero dentro de \$Extend que contiene límites de cuota asignados a un usuario
25	\$ObjID	Contiene los IDS de los objetos
26	\$Reparse	Contiene información acerca de ficheros y carpetas que incluyen el repase point data.

Cada fichero anteriormente enumerado ocupa un registro en la tabla MFT. Normalmente el tamaño del registro es de 1024 bytes e ira en función del tamaño del volumen.

### ¿Cómo son los registros MFT?

- Todos empiezan por el string "FILE".
- Tienen una cabecera que indica información del fichero o carpeta y tiene un tamaño de 42bytes

File Record Segment Header																
	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0	F	I	L	E	Update Seq array offset		Update Seq array size		\$LogFile Sequence Number							
1	Seq no		Hard Link Count		1 <sup>st</sup> attrib offset		Flags		Used size of file record				Allocated size of file record			
2	File reference to base file record								Next attrib ID				MFT Record No			
3	default location of update seq array (size determined by seq size)						Reserved for update sequence array?									
	Reserved for sequence array?								Common location of 1 <sup>st</sup> attrib							

Value	Description
0x00	Deleted File Entry
0x01	File Entry
0x02	Deleted Folder Entry
0x03	Folder Entry

¿Cómo podemos identificar los primeros registros de tabla MFT que son los metadatos del sistema de archivos NTFS?

*\*Ver video: 002 /MÓD. 2 - BootSector Metadatos*

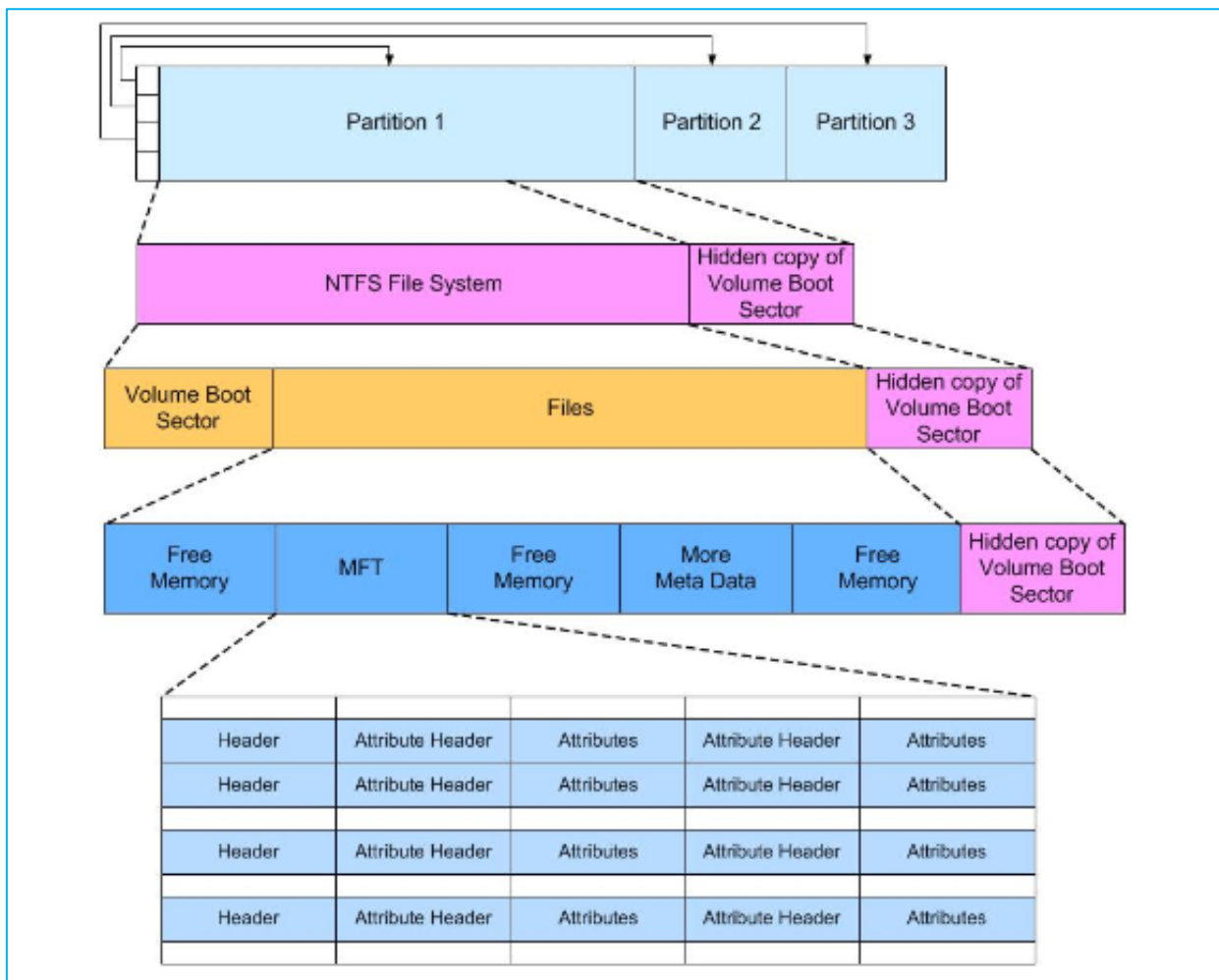
Siguiendo con la teoría hemos visto, que hay un campo dentro del registro MFT, que son los FLAGS. Estos son los que nos van a indicar si el fichero o carpeta ha sido borrada. Hay que tener en cuenta, que el sistema operativo, por el simple uso, puede sobrescribir este registro MFT y ya no tendríamos acceso a esta información. Por eso es muy importante que cuando se vaya a realizar una investigación forense, donde se sospeche que haya información borrada, el dispositivo objeto de la investigación, se haya utilizado lo menos posible después de la declaración de incidente.

¿Cómo se puede identificar a bajo nivel si un fichero está borrado? Analizando los registros MFT.

*\*Ver video: 003 /MÓD. 2 - Identificación de Registros MFT borrados con Active Disk*

*\*Ver video: 004 /MÓD. 2 - Identificación de ficheros borrados con FTK*

Resumen de cómo está compuesto un sistema NTFS en un dispositivo:





## ATRIBUTOS MFT

En la imagen anterior, donde identificábamos los flags de borrado en los registros MFT, veíamos solamente la cabecera. En la imagen de a continuación podemos ver el registro MFT completo.

Aparecen nuevos campos, los atributos:

- ◆ Cabecera del Atributo
- ◆ Atributo en si

MFT Entry Header

Attribute Header

\$FILE\_NAME

Attribute Header

\$STD\_INFO

Attribute Header

\$DATA

QUANTI

Los tipos de atributo son los siguientes:

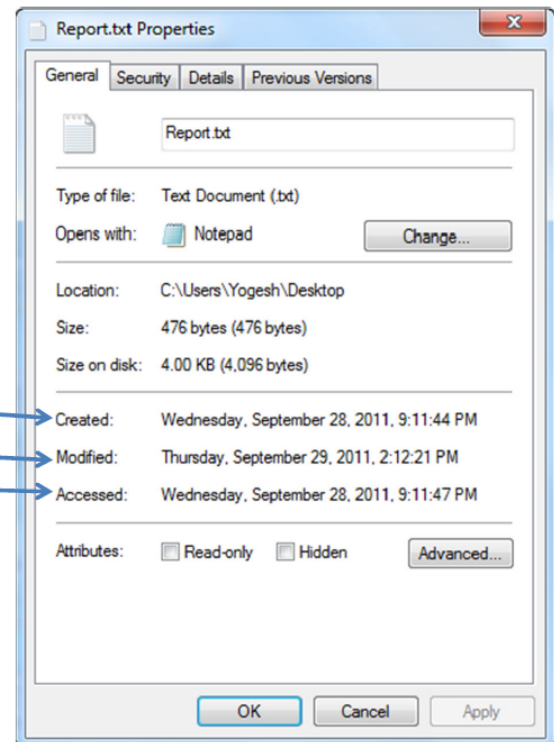
Type Identifier (Decimal)	Type Identifier (Hexadecimal)	Attribute Name
16	0x10	\$STANDARD_INFORMATION
32	0x20	\$ATTRIBUTE_LIST
48	0x30	\$FILE_NAME
64	0x40	\$VOLUME_VERSION
64	0x40	\$OBJECT_ID
80	0x50	\$SECURITY_DESCRIPTOR
96	0x60	\$VOLUME_NAME
112	0x70	\$VOLUME_INFORMATION
128	0x80	\$DATA
144	0x90	\$INDEX_ROOT
160	0xA0	\$INDEX_ALLOCATION
176	0xB0	\$BITMAP
192	0xC0	\$SYMBOLIC_LINK
192	0xD0	\$REPARSE_POINT
208	0xE0	\$EA_INFORMATION
224	0xF0	\$EA
256	0x100	\$LOGGED_UTILITY_STREAM
---	0xFFFFFFFF	End of Attributes

De acuerdo con el fichero \$AttriDef, los atributos pueden ser residentes (están dentro del registro MFT) o no residentes. Por ejemplo, el atributo \$Data podría ser residente, esto quiere decir que el contenido del propio fichero estaría dentro del MFT.

Los atributos existen para cada fichero y directorio. Los de varios tipos como vemos en la imagen superior y pueden contener desde el nombre, el propietario y permisos del fichero/carpeta, e incluso la información como tal.

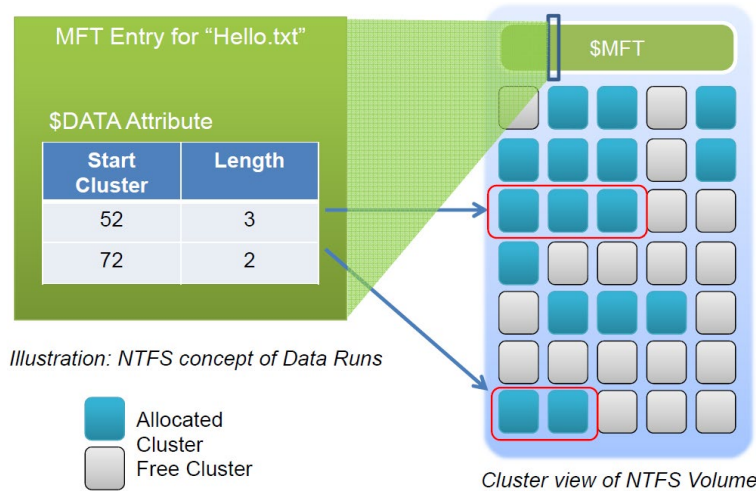
- ◆ \$Standar Information: contiene los timestamps que nosotros vemos en el explorador de Windows.
  - Fecha de Creación
  - Fecha de Modificación
  - Fecha de Acceso
  - Fecha de cuando se actualizado algún de las fechas del registro MFT
  - Flags
  - Identificador de Seguridad

- 64 bit Timestamp
  - Number of 100 Nanosecond intervals since 1<sup>st</sup> January 1601
    - 1 second = 0x989680
- 4 Timestamps
  - Created
  - Modified
  - Accessed
  - MFT Entry Modified - ?

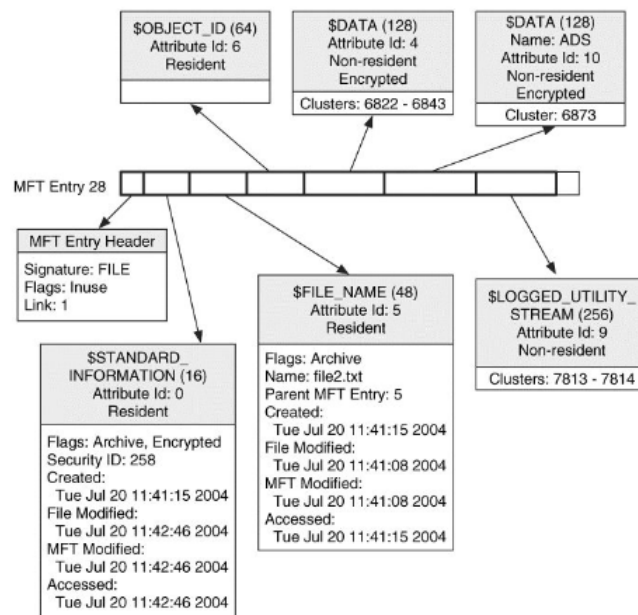


# QUANTIKA<sup>14</sup>

- ◆ \$File Name: contiene:
  - Nombre de archivo
  - Flags
  - Parent MFT Entry: el registro del MFT de la carpeta que contiene dicho archivo o carpeta.
  - Fecha de Creación
  - Fecha de Modificación
  - Fecha de Acceso
  - Fecha de cuando se ha actualizado al registro MFT en cuanto a fechas
- ◆ \$Data: contiene la información del fichero. En la imagen de a continuación podemos cómo funciona el atributo \$DATA:



Un resumen mucho más detallado de los atributos en NTFS lo podemos ver en la siguiente imagen:



\*Ver Video: 005/MÓD. 2 - Identificación de Atributos a bajo nivel



## FECHAS

NTFS dispone de 8 timestamps los del \$STANDARD\_INFORMATION y los de \$FILE\_NAME, nosotros solo vemos en Windows Explorer los del \$STANDARD\_INFORMATION

- Modificado
- Accedido
- Cambiado (El \$MFT ha sido cambiado)
- Creado (Birth o creación de ficheros)

Estos son atributos como los que hemos visto anteriormente.

La siguiente tabla muestra como las fechas de \$STANDARD\_INFORMATION y de \$FILE\_NAME son actualizadas en función de la operación que se haga con el fichero en Windows 10.

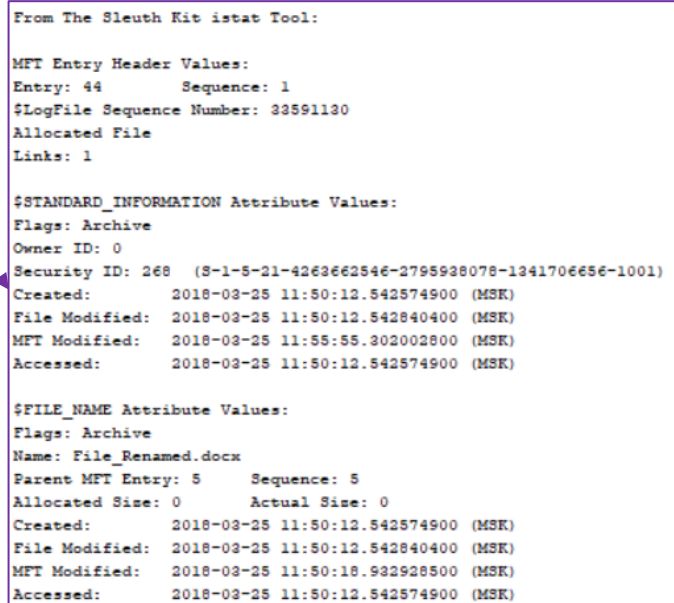
\$STANDARD_INFO							
File Rename	Local File Move	Volume File Move	File Copy	File Access	File Modify	File Creation	File Deletion
Modified - No Change	Modified - No Change	Modified - No Change	Modified - No Change	Modified - No Change	Modified - Changed	Modified - Changed	Modified - No Change
Access - No Change	Access - No Change	Access - Changed	Access - Changed	Access - No Change	Access - Changed	Access - Changed	Access - No Change
Creation - No Change	Creation - No Change	Creation - Changed	Creation - Changed	Creation - No Change	Creation - No Change	Creation - Changed	Creation - No Change
Metadata - Changed	Metadata - Changed	Metadata - No change	Metadata - No change	Metadata - No Change	Metadata - Changed	Metadata - Changed	Metadata - No Change
\$FILE_NAME							
File Rename	Local File Move	Volume File Move	File Copy	File Access	File Modify	File Creation	File Deletion
Modified - No Change	Modified - No Change	Modified - Changed	Modified - Changed	Modified - No Change	Modified - Changed	Modified - Changed	Modified - No Change
Access - No Change	Access - No Change	Access - Changed	Access - Changed	Access - No Change	Access - Changed	Access - Changed	Access - No Change
Creation - No Change	Creation - No Change	Creation - Changed	Creation - Changed	Creation - No Change	Creation - No Change	Creation - Changed	Creation - No Change
Metadata - No change	Metadata - No change	Metadata - Changed	Metadata - Changed	Metadata - No Change	Metadata - Changed	Metadata - Changed	Metadata - No Change



### Renombrar el fichero

Si renombramos el fichero "File\_Rename.docx" a "File\_Renamed.docx" y esto lo que ocurre con los atributos: **solo la fecha de cambio del registro del MFT \$STANDARD\_INFO MFT se modifica.**

- 25-03-2018 11:55:55



```
From The Sleuth Kit istat Tool:

MFT Entry Header Values:
Entry: 44          Sequence: 1
$LogFile Sequence Number: 33591130
Allocated File
Links: 1

$STANDARD_INFORMATION Attribute Values:
Flags: Archive
Owner ID: 0
Security ID: 268 (S-1-5-21-4263662546-2795938078-1341706656-1001)
Created:      2018-03-25 11:50:12.542574900 (MSK)
File Modified: 2018-03-25 11:50:12.542840400 (MSK)
MFT Modified:  2018-03-25 11:55:55.302002800 (MSK)
Accessed:      2018-03-25 11:50:12.542574900 (MSK)

$FILE_NAME Attribute Values:
Flags: Archive
Name: File_Renamed.docx
Parent MFT Entry: 5      Sequence: 5
Allocated Size: 0        Actual Size: 0
Created:      2018-03-25 11:50:12.542574900 (MSK)
File Modified: 2018-03-25 11:50:12.542840400 (MSK)
MFT Modified:  2018-03-25 11:50:18.932928500 (MSK)
Accessed:      2018-03-25 11:50:12.542574900 (MSK)
```

### Mover un fichero local dentro volumen:

Movemos el fichero "Local\_Move.docx" en el mismo volumen. De nuevo **solamente la fecha de cambio del registro del MFT \$STANDARD\_INFO MFT se modifica.**

- 25-03-2018 11:56:06

```

From The Sleuth Kit istat Tool:

MFT Entry Header Values:
Entry: 47          Sequence: 1
$LogFile Sequence Number: 33591955
Allocated File
Links: 1

$STANDARD_INFORMATION Attribute Values:
Flags: Archive
Owner ID: 0
Security ID: 268 (S-1-5-21-4263662546-2795938078-1341706656-1001)
Created:      2018-03-25 11:51:04.734437900 (MSK)
File Modified: 2018-03-25 11:51:04.734437900 (MSK)
MFT Modified:  2018-03-25 11:56:06.759833600 (MSK)
Accessed:      2018-03-25 11:51:04.734437900 (MSK)

$FILE_NAME Attribute Values:
Flags: Archive
Name: Local_Move.docx
Parent MFT Entry: 54   Sequence: 1
Allocated Size: 0      Actual Size: 0
Created:      2018-03-25 11:51:04.734437900 (MSK)
File Modified: 2018-03-25 11:51:04.734437900 (MSK)
MFT Modified:  2018-03-25 11:51:09.211645200 (MSK)
Accessed:      2018-03-25 11:51:04.734437900 (MSK)

```

Movemos el fichero a otro Volumen NTFS

Movemos el fichero a otro volumen NTFS con el nombre "Volume\_Move.docx" y esto es lo que ocurre: cambian todos excepto:

- STANDAR\_INFO Modificación
- STANDARD\_INFO MFT

```

From The Sleuth Kit istat Tool:

MFT Entry Header Values:
Entry: 44          Sequence: 1
$LogFile Sequence Number: 33577125
Allocated File
Links: 1

$STANDARD_INFORMATION Attribute Values:
Flags: Archive
Owner ID: 0
Security ID: 268 (S-1-5-21-4263662546-2795938078-1341706656-1001)
Created:      2018-03-25 11:56:15.548242400 (MSK)
File Modified: 2018-03-25 11:51:36.610832400 (MSK)
MFT Modified:  2018-03-25 11:51:43.063572400 (MSK)
Accessed:      2018-03-25 11:56:15.548242400 (MSK)

$FILE_NAME Attribute Values:
Flags: Archive
Name: Volume_Move.docx
Parent MFT Entry: 5   Sequence: 5
Allocated Size: 0      Actual Size: 0
Created:      2018-03-25 11:56:15.548242400 (MSK)
File Modified: 2018-03-25 11:56:15.548242400 (MSK)
MFT Modified:  2018-03-25 11:56:15.548242400 (MSK)
Accessed:      2018-03-25 11:56:15.548242400 (MSK)

```

Copiado de Ficheros

- Copiamos en el mismo volumen y esto es lo que ocurre: cambian todos excepto:  
**STANDARD\_INFO Modificación**
- **STANDARD\_INFO MFT**

```
From The Sleuth Kit istat Tool:

MFT Entry Header Values:
Entry: 55          Sequence: 1
LogFile Sequence Number: 33592740
Allocated File
Links: 1

$STANDARD_INFORMATION Attribute Values:
Flags: Archive
Owner ID: 0
Security ID: 268 (S-1-5-21-4263662546-2795938078-1341706656-1001)
Created:          2018-03-25 11:56:39.830218900 (MSK)
File Modified:    2018-03-25 11:52:53.262297100 (MSK)
MFT Modified:     2018-03-25 11:52:55.865906700 (MSK)
Accessed:         2018-03-25 11:56:39.830218900 (MSK)

$FILE_NAME Attribute Values:
Flags: Archive
Name: Copy.docx
Parent MFT Entry: 54      Sequence: 1
Allocated Size: 0         Actual Size: 0
Created:          2018-03-25 11:56:39.830218900 (MSK)
File Modified:    2018-03-25 11:56:39.830218900 (MSK)
MFT Modified:     2018-03-25 11:56:39.830218900 (MSK)
Accessed:         2018-03-25 11:56:39.830218900 (MSK)
```

QUANTIKA<sup>14</sup>

Acceso a los ficheros

No cambia ningún timestamp, en sistemas Windows, viene por defecto deshabilitado esta opción.

```

From The Sleuth Kit istat Tool:

MFT Entry Header Values:
Entry: 50          Sequence: 1
$LogFile Sequence Number: 32585065
Allocated File
Links: 1

$STANDARD_INFORMATION Attribute Values:
Flags: Archive
Owner ID: 0
Security ID: 268 (S-1-5-21-4263662546-2795938078-1341706656-1001)
Created:          2018-03-25 11:53:44.869496700 (MSK)
File Modified:    2018-03-25 11:53:44.869496700 (MSK)
MFT Modified:     2018-03-25 11:53:47.321143200 (MSK)
Accessed:         2018-03-25 11:53:44.869496700 (MSK)

$FILE_NAME Attribute Values:
Flags: Archive
Name: Access.docx
Parent MFT Entry: 5      Sequence: 5
Allocated Size: 0        Actual Size: 0
Created:          2018-03-25 11:53:44.869496700 (MSK)
File Modified:    2018-03-25 11:53:44.869496700 (MSK)
MFT Modified:     2018-03-25 11:53:44.869496700 (MSK)
Accessed:         2018-03-25 11:53:44.869496700 (MSK)

```

Modificación de fichero

Se modifica un fichero: solo se mantienen la fecha de creación en el \$STANDARD\_INFO y \$FILE\_NAME. El resto de timestamps cambia.

```

From The Sleuth Kit istat Tool:

MFT Entry Header Values:
Entry: 58          Sequence: 1
$LogFile Sequence Number: 32598208
Allocated File
Links: 1

$STANDARD_INFORMATION Attribute Values:
Flags: Archive
Owner ID: 0
Security ID: 272 (S-1-5-21-4263662546-2795938078-1341706656-1001)
Created:          2018-03-25 11:54:02.390699600 (MSK)
File Modified:    2018-03-25 11:57:13.817538200 (MSK)
MFT Modified:     2018-03-25 11:57:13.839329800 (MSK)
Accessed:         2018-03-25 11:57:13.806276200 (MSK)

$FILE_NAME Attribute Values:
Flags: Archive
Name: Modification.docx
Parent MFT Entry: 5      Sequence: 5
Allocated Size: 12288    Actual Size: 11414
Created:          2018-03-25 11:54:02.390699600 (MSK)
File Modified:    2018-03-25 11:57:13.817538200 (MSK)
MFT Modified:     2018-03-25 11:57:13.838326700 (MSK)
Accessed:         2018-03-25 11:57:13.806276200 (MSK)

```

### Creación de Fichero

Creamos un nuevo fichero con el botón derecho del ratón: **Por su puesto todos los timestamps se cambian.**

```
MFT Entry Header Values:
Entry: 52          Sequence: 1
LogFile Sequence Number: 33587313
Allocated File
Links: 1

$STANDARD_INFORMATION Attribute Values:
Flags: Archive
Owner ID: 0
Security ID: 268 (S-1-5-21-4263662546-2795938078-1341706656-1001)
Created:          2018-03-25 11:54:16.739109900 (MSK)
File Modified:    2018-03-25 11:54:16.842875100 (MSK)
MFT Modified:     2018-03-25 11:54:19.122337700 (MSK)
Accessed:         2018-03-25 11:54:16.739109900 (MSK)

$FILE_NAME Attribute Values:
Flags: Archive
Name: Creation.docx
Parent MFT Entry: 5      Sequence: 5
Allocated Size: 0        Actual Size: 0
Created:                2018-03-25 11:54:16.739109900 (MSK)
File Modified:          2018-03-25 11:54:16.842875100 (MSK)
MFT Modified:           2018-03-25 11:54:16.842875100 (MSK)
Accessed:               2018-03-25 11:54:16.739109900 (MSK)
```

### Borrado del fichero

A nivel de timestamps no cambian.

```

From The Sleuth Kit istat Tool:

MFT Entry Header Values:
Entry: 53          Sequence: 2
$LogFile Sequence Number: 33599259
Not Allocated File
Links: 1

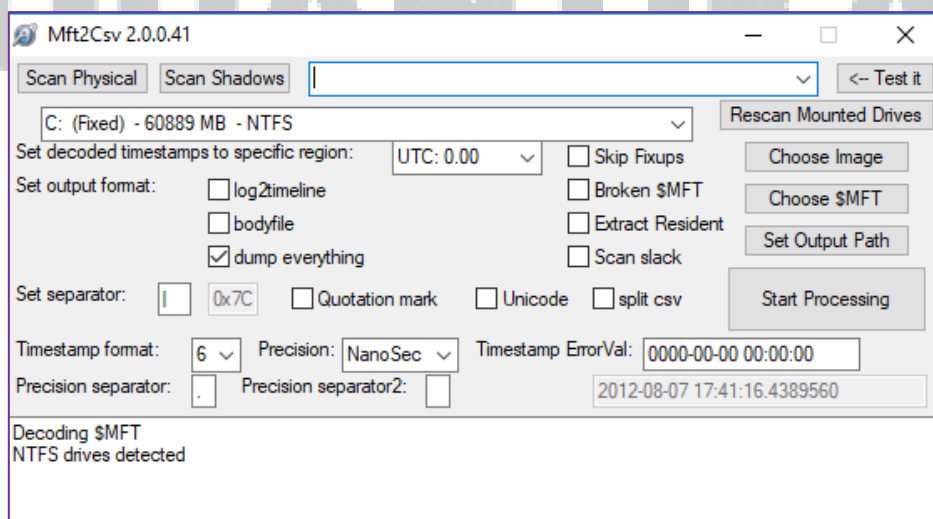
$STANDARD_INFORMATION Attribute Values:
Flags: Archive
Owner ID: 0
Security ID: 268 (S-1-5-21-4263662546-2795938078-1341706656-1001)
Created:          2018-03-25 11:54:33.318813900 (MSK)
File Modified:    2018-03-25 11:54:33.318813900 (MSK)
MFT Modified:     2018-03-25 11:54:27.402380300 (MSK)
Accessed:         2018-03-25 11:54:33.318813900 (MSK)

$FILE_NAME Attribute Values:
Flags: Archive
Name: Deletion.docx
Parent MFT Entry: 5          Sequence: 5
Allocated Size: 0           Actual Size: 0
Created:          2018-03-25 11:54:33.318813900 (MSK)
File Modified:    2018-03-25 11:54:33.318813900 (MSK)
MFT Modified:     2018-03-25 11:54:33.318813900 (MSK)
Accessed:         2018-03-25 11:54:33.318813900 (MSK)

```

¿Qué herramienta podemos utilizar para analizar el MFT?

MFT2CSV: <https://github.com/jschicht/Mft2Csv>



Permite analizar:

- ◆ Leer un disco o imagen en formato RAW con MBR y GPT
- ◆ Leer una partición en formato RAW
- ◆ Leer directamente el fichero \$MFT (previa extracción de un post mortem)
- ◆ Leer directamente el fichero \$MFT de un sistema Live
- ◆ Leer \$MFT de una shadow copy
- ◆ MFT y sus atributos.
- ◆ ADS
- ◆ Extraer Ficheros residentes en el MFT

- ◆ Atributos \$INDEX
- ◆ Timestamps de \$STANDARD\_INFORMATION y \$FILE\_NAME
- ◆ Ficheros que se hayan borrado

*\*Ver Video: 006/MÓD. 2 - Análisis de MFT*

## \$LOGFILE / USNJRNL

El **\$Logfile** contiene información usada por NTFS para recuperarse de manera rápida para volver al estado anterior. Es decir que todas las transacciones que se realizan sobre \$LogFile. Tiene un tamaño máximo de 65536KB. Esta localizado en el registro segundo del MFT.

¿Qué tipo de transacciones?

- ◆ Creación de Fichero
- ◆ Borrado de Ficheros
- ◆ Renombrar el fichero
- ◆ Copia de Ficheros

If_LSN	If_LSNPrevious	If_RedoOperation	If_UndoOperation
4210114	4210102	AddIndexEntryRoot	DeleteIndexEntryRoot
4210139	4210114	InitializeFileRecordSegment	Noop
4210215	4210196	UpdateNonResidentValue	Noop
4210267	0	UpdateResidentValue	UpdateResidentValue
4210278	4210267	UpdateResidentValue	UpdateResidentValue
4210310	4210291	UpdateNonResidentValue	Noop
4210417	4210398	UpdateNonResidentValue	Noop
4210469	0	UpdateResidentValue	UpdateResidentValue
4210515	4210496	UpdateNonResidentValue	Noop
4210622	4210603	UpdateNonResidentValue	Noop
4210663	4210644	UpdateResidentValue	UpdateResidentValue
4216877	0	DeleteIndexEntryAllocation	AddIndexEntryAllocation
4216902	4216877	DeallocateFileRecordSegment	InitializeFileRecordSegment
4216947	4216928	UpdateNonResidentValue	Noop

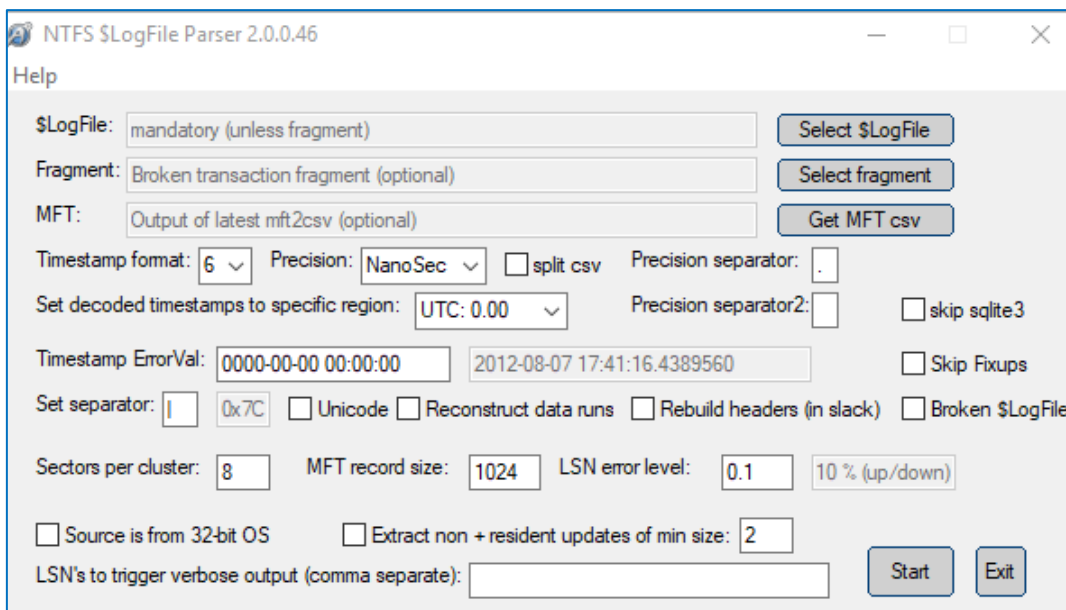
En la imagen superior podemos ver los siguientes campos:

- ◆ LSN: número de transacción dentro del \$LogFile
- ◆ Operación tanto REDO como UNDO

Como se puede observar se encarga de restablecer las operaciones que realiza a nivel del sistema de archivos, es decir de todos los atributos NTFS que hemos visto anteriormente, incluso las operaciones que hace sobre el \$USNJRNL.

La herramienta que nos permite analizarlo se llama LogfileParser:

<https://github.com/jschicht/LogFileParser>



Para hacerla funcionar, es necesario copiar el ZIP sobre una ruta más simple, y se le debe proporcionar el CSV que ha sido generado anteriormente con el MFT2CSV.

*\*Ver video: 007/MÓD. 2 - Análisis de LogFile*

El fichero \$EXTEND\\$\\$UsnJrnl\\$\$ es un alternate data stream del fichero \$EXTEND\\$\\$UsnJrnl Este artefacto contiene un registro con muchísimos cambios que se producen en un volumen NTFS. Se interpreta mucho mejor que el \$logfile.

¿Qué operaciones aparecen en este fichero?

- ◆ Ficheros añadidos
- ◆ Ficheros borrados
- ◆ Ficheros Modificados

El fichero \$\$J tiene una estructura como la que se ve en la siguiente imagen.



Record Length

Reason

File Attributes

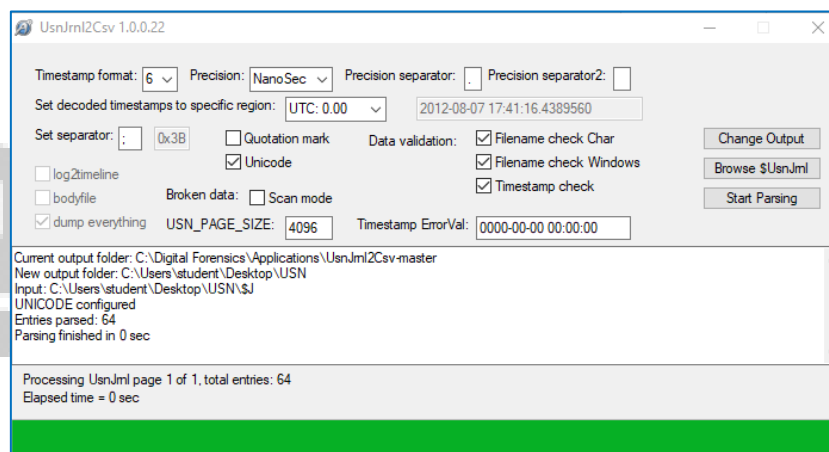
TimeStamp

File name

0280	A6 00 00 00	02 00 00 00	23 00	...	#
028A	00 00 00 00	02 00 05 00	00 00	...	...
0294	00 00 05 00	80 02 00 00	00 00	...	...
0298	00 00 17 FE	81 A7 1E 5C	CC 51	...	...
02A8	00 02 00 80	00 00 00 00	00 00	...	...
02B2	00 00 20 00	00 00 6C 00	3C 00	...	...
02BC	41 00 6C 00	69 00 5F 00	41 00	A.l.i._.A.	
02C6	6B 00 62 00	61 00 72 00	5F 00	k.b.a.r._.	
02D0	4B 00 68 00	61 00 6E 00	5F 00	W.h.a.n._.	
02DA	26 00 5F 00	4C 00 5F 00	53 00	&._.L._.S.	
02E4	75 00 62 00	72 00 61 00	6D 00	u.b.r.a.m.	
02EE	61 00 6E 00	69 00 61 00	6E 00	a.n.i.a.n.	
02F8	5F 00 2D 00	5F 00 41 00	6C 00	._._.A.l.	
0302	61 00 70 00	5F 00 69 00	6E 00	a.p._.i.	
030C	5F 00 44 00	61 00 64 00	72 00	._.D.a.d.r.	
0316	61 00 5F 00	54 00 61 00	6C 00	a._.T.a.l.	
0320	2E 00 6D 00	70 00 33 00	68 00	._.m.p.3.h.	

¿Qué herramienta vamos a utilizar? UsnJrnl2Csv que se encuentra en la url:

<https://github.com/jschicht/UsnJrnl2Csv>



*\*Ver video: 008/MÓD. 2 - Análisis de USNJOURNAL*

El resultado de analizar el USNJRNL de la evidencia USB nos indica que el fichero fue borrado en el momento marcado en rojo. Tener en cuenta que este borrado se produjo desde el CMD y nunca paso por la papelera de reciclaje. Cuando pasa por la papelera de reciclaje, no se borra el fichero, sino que cambia de carpeta. En el momento del “vaciado” es cuando se borra.

B	C	D	E
FileName	USN	Timestamp	Reason
Deleted_n.txt	1624	2018-10-14 21:07:29.3443920	FILE_CREATE
Deleted_n.txt	1712	2018-10-14 21:07:29.3443920	DATA_EXTEND+FILE_CREATE
Deleted_n.txt	1800	2018-10-14 21:07:29.3443920	DATA_EXTEND+DATA_OVERWRITE+FILE_CREATE
Deleted_n.txt	1888	2018-10-14 21:07:29.3443920	BASIC_INFO_CHANGE+DATA_EXTEND+DATA_OVERWRITE+FILE_CREATE
Deleted_n.txt	1976	2018-10-14 21:07:29.3443920	BASIC_INFO_CHANGE+CLOSE+DATA_EXTEND+DATA_OVERWRITE+FILE_CREATE
Deleted_n.txt	5784	2018-10-14 21:08:47.9794029	CLOSE+FILE_DELETE



¿Qué diferencias hay entre el \$usnrnl y el \$logfile? Por resumir, en el \$logfile encontramos las transacciones sobre los atributos del fichero que se han realizado. Mientras que en el usnjournal podemos ver el resultado de dichas acciones.

```
USN      : 4832
File name: password.txt
Timestamp: 2016-01-29 21:29:12.795932
Reason   : FILE_DELETE|CLOSE

$LogFile transaction number: 38

LSN      Redo operation                               Undo operation
1085650 Delete Index Entry Allocation                Add Index Entry Allocation
1085675 Delete Index Entry Root                      Add Index Entry Root
1085697 Deallocate File Record Segment               Initialize File Record Segment
1085711 Clear Bits in Nonresident Bitmap              Set Bits in Nonresident Bitmap
1085723 Set New Attribute Sizes                       Set New Attribute Sizes
1085742 Update Nonresident Value                     No-Operation
1085764 Set New Attribute Sizes                       Set New Attribute Sizes
1085783 Forget Transaction                           Compensation Log Record
```





Una herramienta que analiza \$UsnJrnl y \$LogFile a la vez, es [NTFS LogTracker](#), también permite realizar carving en busca de registros borrados de \$UsnJrnl

The screenshot shows the NTFS Log Tracker v1.41 application window. It features several input fields for file paths, including \$LogFile File Path, \$UsnJrnl:\$J File Path, and an Unallocated Dump Path. There are 'Clear' and 'Parse' buttons. Below these, there are sections for 'Option' (with \$MFT File Path) and 'Open SQLite DB File' (with SQLite DB File Path and an 'Open' button). A 'Search' button and a 'CSV Export' button are also present. At the bottom, there is a table with columns: LSN, Event Time, Event, Detail, and File Name. The table is currently empty.

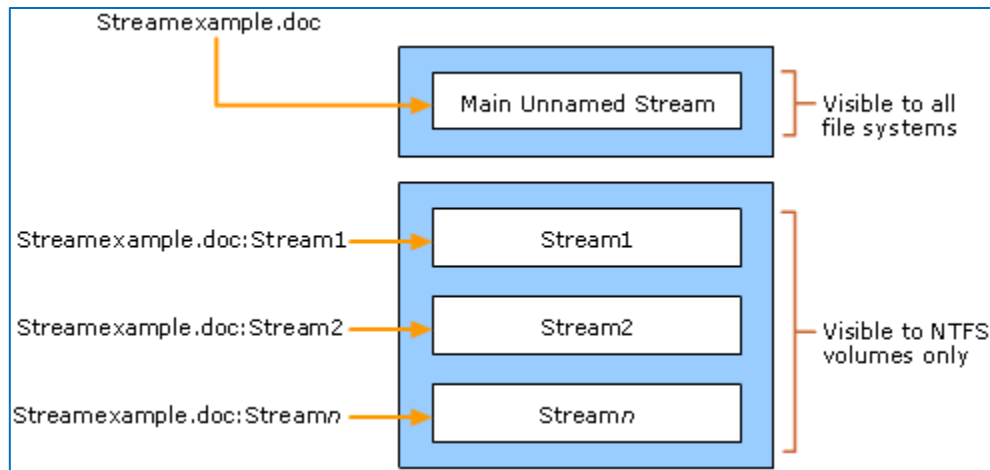
Otra herramienta muy parecida es ANJP de <https://www.gettriforce.com/> que permite tener una visión conjunta de todos los artefactos \$UsnJrnl, \$Logfile y \$MFT

The screenshot shows the ANJP 3.11.07 application window. It has a menu bar with 'File' and 'Help'. Below the menu bar, there are tabs for 'Process' and 'Reports'. The main area contains input fields for Case Name, Case Path, MFT, LogFile, and USN, each with a 'Browse' button. There is a checkbox labeled 'Process Events After Parsing'. At the bottom, there are buttons for 'Parse', 'Options', and 'Event Selection'.

*\*Ver video: 009/MÓD. 2 - ANJP*

## ADS (ALTERNANTE DATA STREAM)

Desde Windows 2000, el sistema de archivos NTFS permite el uso de metadatos conocido como ALTERNATE DATA STREAMS (ADS), que le permiten almacenar datos dentro de un propio fichero. No es detectable durante la navegación por el sistema de archivos, o en cualquier lugar dentro de Windows



Esta característica solo funciona con sistemas basados en NTFS, perdiendo esta particularidad si se copia o mueve a otros formatos de ficheros.

```

Administrator: Command Prompt

C:\Users\Sam\ads>echo a > little.txt

C:\Users\Sam\ads>echo "BBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBB" > little.txt:big.txt

C:\Users\Sam\ads>dir
Volume in drive C is Win7RTM
Volume Serial Number is C2E4-15E3

Directory of C:\Users\Sam\ads

09/27/2010  01:04 PM    <DIR>          .
09/27/2010  01:04 PM    <DIR>          ..
09/27/2010  01:04 PM                4 little.txt
               1 File(s)                4 bytes
               2 Dir(s)  14,264,012,800 bytes free
  
```

En la imagen anterior vemos cómo podemos añadir un ADS al fichero Little.txt, el fichero big.txt, con el contenido "BBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBB". Al navegar mediante Windows Explorer y con el CMD de Windows, no podríamos detectarlo.

¿Por qué es útil? Todos los ficheros descargados de Internet, Navegadores y Powershell incluidos le añaden un ADS al fichero descargado.

- ◆ WinXP: solo los ejecutables
- ◆ Win7-Win10: cualquier fichero descargado le añade el ADS

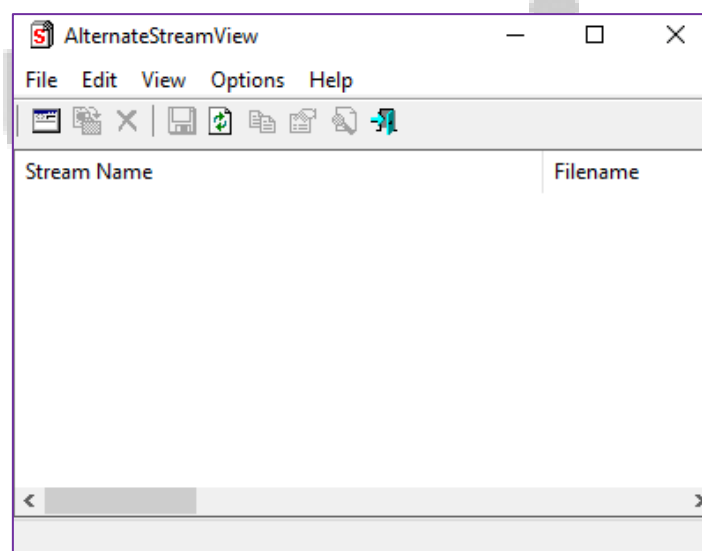
El ADS no es más que un fichero “oculto”, ¿Cómo podemos identificar cuales los ficheros que tienen un ADS que indica que ha sido descargado de Internet?

- ◆ Un ADS que tiene el nombre de fichero Zone.Identifier

Dentro de este fichero podemos encontrar distintas Zonas de Internet que irán en función de lo que se tenga configurado en Internet Explorer o Edge. Esta información se encuentra en el registro de Windows

- ◆ Zone ID = 3 -> Internet
- ◆ Zone ID = 4 -> Untrusted
- ◆ Zone ID = 2 -> Trusted
- ◆ Zone ID = 1 -> Intranet
- ◆ Zone ID = 0 -> Mycomputer

EL ADS lo sacábamos como un atributo más de los ficheros, por lo cual se encontraba en el MFT. Con la herramienta MFT2CSV nos indicaba los ficheros que tiene un ADS mediante el campo “ADS”. Otra herramienta muy útil, es la herramienta [AlternateStreamView de Nirsoft](#).



Para hacerla funcionar, conectaremos nuestra evidencia con posibilidad de escritura temporal, ya que otorgaremos permisos sobre ciertas carpetas para que puedan ser analizadas en busca de ADS.

*\*Ver Video: 010/MÓD. 2 - Análisis de ADS*

```
[ZoneTransfer]
ZoneId=3
ReferrerUrl=https://www.jonrajewski.com/resources/
HostUrl=http://www.jonrajewski.com/data/for270/timestomp.exe
```

En el video aparece la captura de la imagen anterior, del fichero

C:\users\ismis\Download\timestomp.exe donde se puede apreciar incluso la URL completa de donde se descargó, gracias a que lo hizo desde Chrome.

A continuación, una lista de los principales browsers o aplicaciones que descargan ficheros de internet y que dejan este ADS:

Google Chrome:

- ◆ [ZoneTransfer]
- ◆ ZoneId=3
- ◆ ReferrerUrl
- ◆ HostUrl

Microsoft Edge:

- ◆ [ZoneTransfer]
- ◆ LastWriterPackageFamilyName=Microsoft.MicrosoftEdge\_8wekyb3d8bbwe
- ◆ ZoneId=3

Firefox:

- ◆ [ZoneTransfer]
- ◆ ZoneId=3

Opera:

- ◆ [ZoneTransfer]
- ◆ ZoneId=3
- ◆ ReferrerUrl
- ◆ HostUrl

Tor Browser:

- ◆ [ZoneTransfer]
- ◆ ZoneId=3
- ◆ Tor está basado en Firefox

Vivaldi:

- ◆ [ZoneTransfer]
- ◆ ZoneId=3
- ◆ ReferrerUrl
- ◆ HostUrl

Microsoft Outlook 2016:

- ◆ [ZoneTransfer]
- ◆ ZoneId=3

Mozilla Thunderbird too.



- ◆ [ZoneTransfer]
- ◆ Zoneld=3

Windows Mail:

- ◆ [ZoneTransfer]
- ◆ Zoneld=3

µTorrent

- ◆ Zoneld=3
- ◆ HostUrl=about:internet

QUANTIKA<sup>14</sup>



## BitTorrent

- ◆ Zone.Identifier:
- ◆ ZoneId=3
- ◆ HostUrl=about:internet

## Telegram Desktop

- ◆ [ZoneTransfer]
- ◆ ZoneId=3

## Skype:

- ◆ [ZoneTransfer]
- ◆ ZoneId=3

## \$INDEX RECORDS

NTFS indexa la información de cada directorio en un B+ Tree:

File List									
Name	Size	Type	Date Modified						
My Music	1	Reparse Point	10/3/2018 5:27:...						
My Pictures	1	Reparse Point	10/3/2018 5:27:...						
My Videos	1	Reparse Point	10/3/2018 5:27:...						
\$I30	4	NTFS Index Allocation	10/3/2018 5:36:...						

000	49	4E	44	58	28	00	09	00-A6	56	53	10	00	00	00	00	INDX ( ...;VS .....
010	00	00	00	00	00	00	00	00-28	00	00	00	A8	02	00	00	..... ( ..... ..
020	E8	0F	00	00	00	00	00	00-03	00	D4	01	00	00	00	00	è ..... ò .....
030	00	00	00	00	00	00	00	00-00	00	00	00	00	00	00	00	.....
040	AC	3C	01	00	00	00	01	00-68	00	52	00	00	00	00	00	←< ..... h·R .....
050	49	3C	01	00	00	00	01	00-9D	F6	8C	69	3E	5B	D4	01	I< ..... ò·i>[ô·
060	9D	F6	8C	69	3E	5B	D4	01-9D	F6	8C	69	3E	5B	D4	01	·ò·i>[ô·ò·i>[ô·
070	9D	F6	8C	69	3E	5B	D4	01-00	00	00	00	00	00	00	00	·ò·i>[ô·.....
080	00	00	00	00	00	00	00	00-06	24	00	10	03	00	00	A0	..... \$ .....
090	08	01	4D	00	79	00	20	00-4D	00	75	00	73	00	69	00	·M·y· ·M·u·s·i·
0a0	63	00	00	00	00	00	00	00-AD	3C	01	00	00	00	01	00	c ..... < .....
0b0	68	00	58	00	00	00	00	00-49	3C	01	00	00	00	01	00	h·X ..... I< .....
0c0	9D	F6	8C	69	3E	5B	D4	01-9D	F6	8C	69	3E	5B	D4	01	·ò·i>[ô·ò·i>[ô·
0d0	9D	F6	8C	69	3E	5B	D4	01-9D	F6	8C	69	3E	5B	D4	01	·ò·i>[ô·ò·i>[ô·
0e0	00	00	00	00	00	00	00	00-00	00	00	00	00	00	00	00	.....
0f0	06	24	00	10	03	00	00	A0-0B	01	4D	00	79	00	20	00	·\$ ..... ·M·y·
100	50	00	69	00	63	00	74	00-75	00	72	00	65	00	73	00	P·i·c·t·u·r·e·s·
110	AE	3C	01	00	00	00	01	00-68	00	54	00	00	00	00	00	è< ..... h·I .....
120	49	3C	01	00	00	00	01	00-9D	F6	8C	69	3E	5B	D4	01	I< ..... ò·i>[ô·
130	9D	F6	8C	69	3E	5B	D4	01-9D	F6	8C	69	3E	5B	D4	01	·ò·i>[ô·ò·i>[ô·
140	9D	F6	8C	69	3E	5B	D4	01-00	00	00	00	00	00	00	00	·ò·i>[ô·.....
150	00	00	00	00	00	00	00	00-06	24	00	10	03	00	00	00	.....



FTK analiza el sistema de archivos y lo representa como un fichero más (\$I30), según vemos en la imagen superior.

Este atributo contiene información sobre los nombres de archivo y directorios que hay almacenados y que hubo dentro de él. Este atributo es también conocido como \$INDEX y contiene:

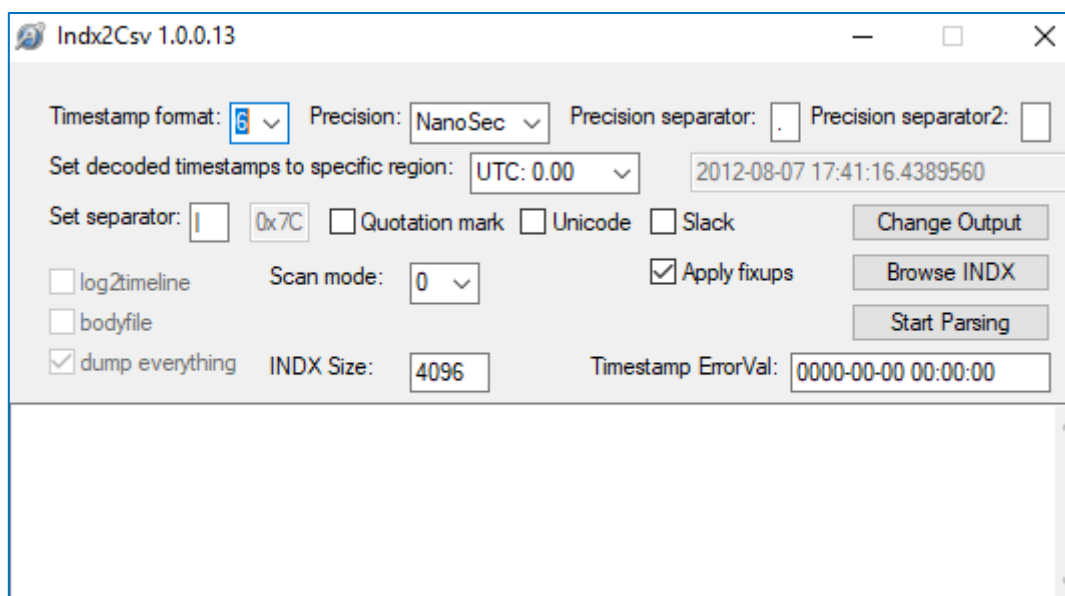
- ◆ \$INDEX\_ROOT = 0x90
- ◆ \$INDEX\_ALLOCATION = 0xA0
- ◆ \$BITMAP = 0XB0

Cuando borras un fichero de un directorio, el atributo \$I30 queda de manera de slack space y se puede encontrar evidencias de que existió. Es decir, dentro del propio fichero \$I30 tenemos slack space.

Name	Size	Type	Date Modified
\$I30	4	NTFS Index Allocation	10/14/2018 9:10:08 PM
CS89_Final_Writeup.pdf	8,722	Regular File	8/18/2017 10:35:56 AM
datawiz2014_02.pdf	1,220	Regular File	8/7/2017 12:02:02 PM
Deleted_n.txt	1	Regular File	10/14/2018 8:21:44 PM
Deleted_p.txt	1	Regular File	10/14/2018 8:21:57 PM
largescaleimageforensics-icrmfse...	899	Regular File	8/7/2017 11:41:37 AM
MFSEC17.pdf	3,057	Regular File	9/24/2017 6:21:14 PM
Nicole_Beebe.pdf	2,134	Regular File	8/7/2017 11:45:17 AM
S5212-Massimo-Bernaschi.pdf	1,380	Regular File	8/7/2017 11:42:56 AM
TIKA.pdf	3,061	Regular File	8/7/2017 11:29:37 AM
Uforia.pdf	128	Regular File	8/7/2017 11:46:23 AM
Uforia.pdf		\$I30 INDX Entry	

En la captura se identifica el fichero Uforia.pdf como \$I30 INDX Entry. Es decir, esta entrada está dentro del \$I30. ¿Qué herramienta podemos utilizar para analizar el fichero \$I30?

IndexCSV



Muy importante marcar la opción de Slack para que pueda obtener dentro del slack space del fichero I30 los registros que hubo.

*\*Ver video: 011 /MÓD. 2 - Análisis de INDX*

¿Qué podemos obtener una vez hayamos analizado el fichero \$I30?

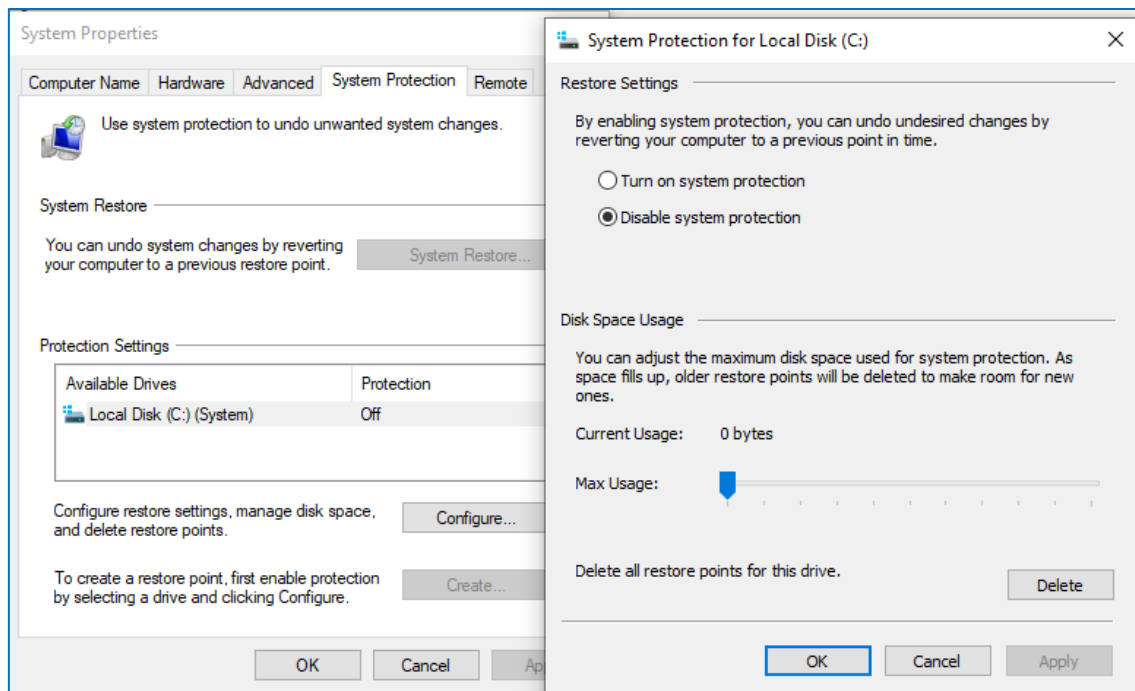
- ◆ Fecha de Creación del Fichero que hay en la carpeta
- ◆ Fecha de modificación del fichero que hay en la carpeta
- ◆ Fecha de acceso del fichero que hay en la carpeta
- ◆ Fecha de cambio de registro MFT que hay en la carpeta
- ◆ Tamaño físico, lógico.
- ◆ Registro MFT

FileName	MFTReference	MF	Indx	MFT	MFT	CTime	ATime
CS89_Final_Writeup.pdf	41	1	0	40	1	2018-10-14 21:07:26.9532593	2017-08-18 10:35:56.3255886
datawiz2014_02.pdf	42	1	0	40	1	2018-10-14 21:07:29.0784003	2017-08-07 12:02:02.7929221
largescaleimageforensics-icrmfsec2017-170608221906.pdf	45	1	0	40	1	2018-10-14 21:07:29.4844162	2017-08-07 11:41:37.9425525
MFSEC17.pdf	46	1	0	40	1	2018-10-14 21:07:30.2811488	2017-09-24 18:21:14.9218029
S5212-Massimo-Bernaschi.pdf	48	1	0	40	1	2018-10-14 21:07:32.5315100	2017-08-07 11:42:56.0933668
Uforia.pdf	0	0	2	40	1	2018-10-14 21:07:33.5155800	2017-08-07 11:46:23.6181172
Uforia.pdf	0	0	2	40	1	2018-10-14 21:07:33.5155800	2017-08-07 11:46:23.6181172
Uforia.pdf	50	1	0	40	1	2018-10-14 21:07:33.5155800	2017-08-07 11:46:23.6181172
Uforia.pdf	8589934608	0	0	40	1	2018-10-14 21:07:33.5155800	2017-08-07 11:46:23.6181172
Uforia.pdf	8589934608	0	0	40	1	2018-10-14 21:07:33.5155800	2017-08-07 11:46:23.6181172

Nunca obtendremos la fecha del fichero borrado que hubo en la carpeta.

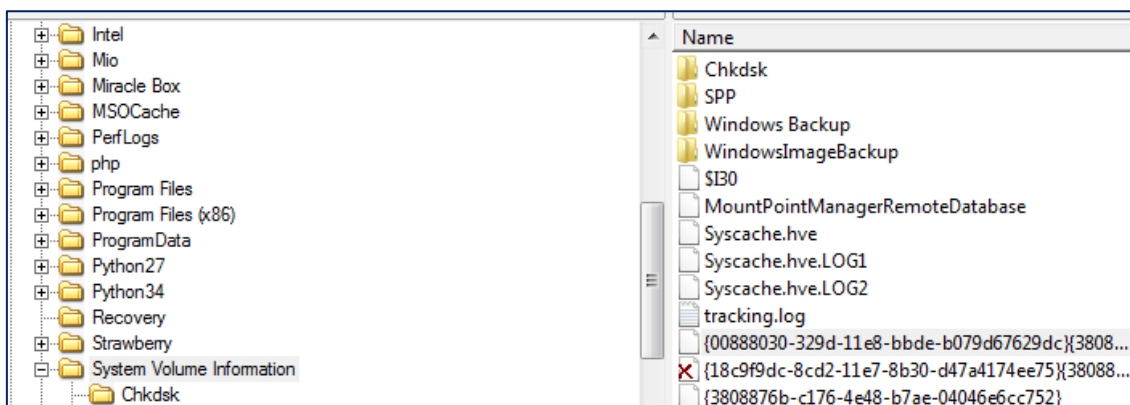
## SHADOW COPIES

Las shadow copies también conocidas como el Servicio Volume Snapshot o Volume Shadow Copy (VSS), es una tecnología que se incluyó en los sistemas operativos a partir de Windows 7. Es un servicio de Backup automático y transparente para el usuario que esté utilizando el sistema.

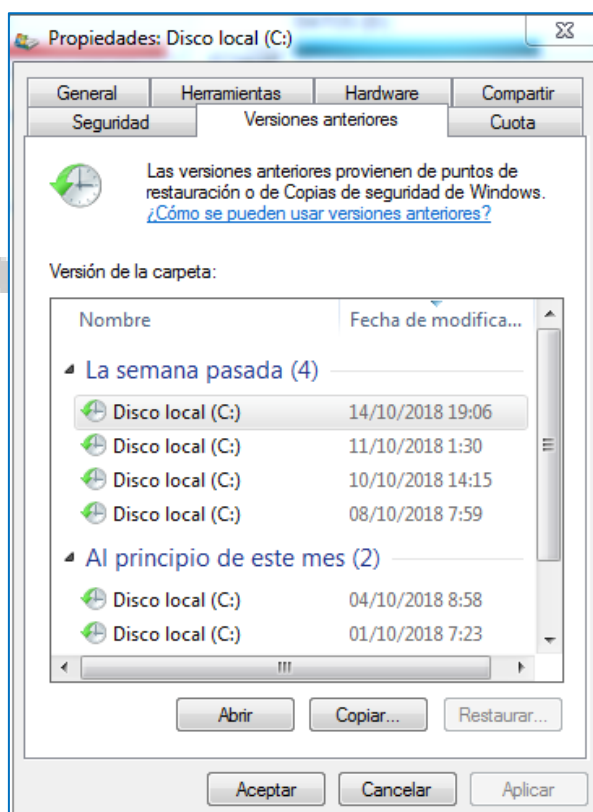


# QUANTIKA<sup>14</sup>

Físicamente las shadows copies se encuentran en carpeta de la raíz: System Volume Information:



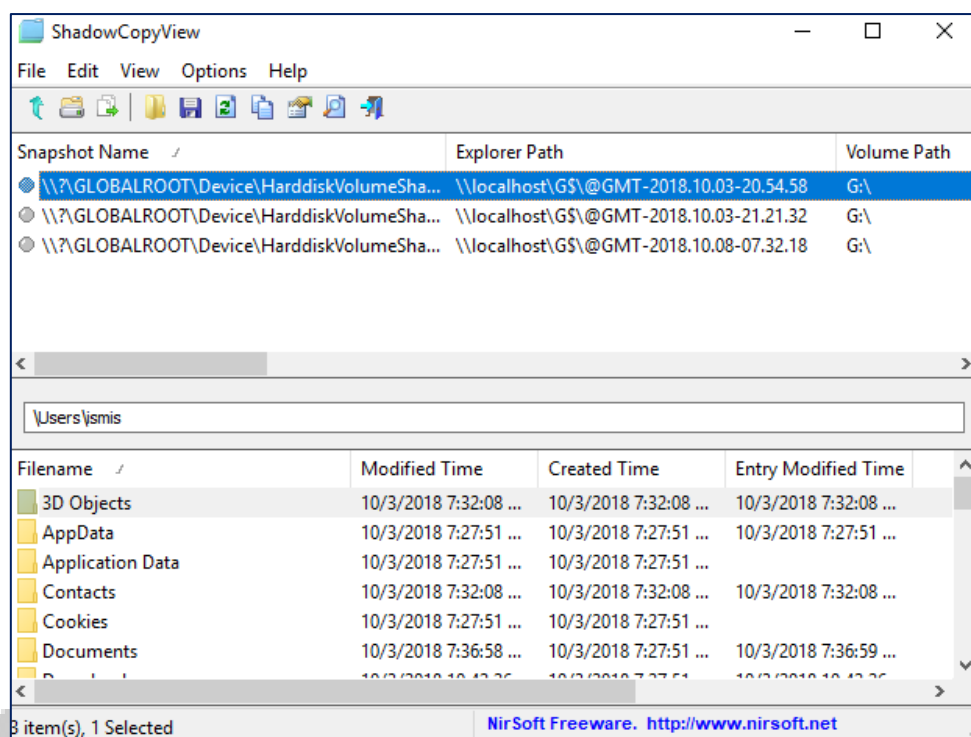
En un equipo encendido o Live, es fácil acceder a dichas shadows copies:



¿cómo podríamos acceder a las shadows copies de una imagen forense?

*\*Ver Video: 012/MÓD. 2 - Análisis de Shadows*

Junto con **Arsenal Image Mounter** y la herramienta de [Nirsoft ShadowCopy View](#) podríamos extraer los ficheros que hay en cada snapshot.



**Para analizar las Shadows Copies de un Windows 10, hay que lanzarlos siempre desde un Windows 10. Así para todas las versiones de Windows.**

¿De qué ficheros se realiza backup mediante las shadows copies?

Para ello habría que analizar el registro de Windows, en especial el fichero SYSTEM e ir a la siguiente ruta:

- ◆ HKEY\_LOCAL\_MACHINE\SYSTEM\CurrentControlSet\Services\VSS

Y a esta ruta:

- ◆ HKEY\_LOCAL\_MACHINE\SYSTEM\CurrentControlSet\Control\BackupRestore

Dentro de esta última clave de registro, podremos encontrar:

- ◆ **FilesNotToBackup** -> ficheros que no se deben copiar
- ◆ **KeysNotToRestore** -> claves de registro de Windows que no se deben de copiar.

## ¿QUÉ OCURRE CUANDO SE BORRA UN FICHERO MEDIANTE EL SISTEMA OPERATIVO?

Cuando vacía la papelera en un Sistema Windows, se borra presionando la tecla “SHIFT” o se borra desde el CMD, es cuando de verdad se produce un borrado el fichero. A continuación, identificamos todo el proceso a nivel forense:

1. Se cambia el atributo flag, en la cabecera del registro MFT del fichero o carpeta que se ha borrado:
  - a. Archivos: 0x01 to 0x00
  - b. Directorios: 0x03 to 0x02.
2. El atributo \$BITMAP del registro del MFT es procesado y establecido a 0
3. Los atributos no residentes del registro del MFT asociado al fichero en cuestión, son procesados y sus clúster son establecidos como espacio libre en el fichero \$BITMAP.

**¿Todo el proceso anterior impide que se pueda recuperar la información?** No. Como ya hemos visto anteriormente se podrá recuperar siempre y cuando no se sobrescriba el registro MFT, debido al uso del sistema.

Los ransomware cuando deshabilitan las Shadows copies, lo que hacen es utilizar una llamada al sistema para la lanzar el comando pertinente. Este comando es propio del sistema operativo

PowerShell

```
vssadmin delete shadows /for=<ForVolumeSpec> [/oldest | /all | /shadow=<ShadowID>] [/quiet]
```

Para recuperarlas veremos en la parte de carving otra herramienta.

**¿Las herramientas de borrado seguro cómo funcionan? Estas herramientas pueden eliminar:**

- ◆ El registro de MFT completo del fichero asociado
- ◆ Logfile
- ◆ Shadows copies

A parte de realizar las operaciones anteriores, los clusters que son utilizados para contener el propio fichero, se sobrescriben a ceros o siguiendo un estándar de borrado que va en función del número de pasadas.



## RECUPERACIÓN DE DATOS MEDIANTE EL PROPIO SISTEMA DE ARCHIVOS

¿Cómo funcionan los programas de recuperación de ficheros que analizan el MFT?

1. Buscan en el registro \$MFT del fichero en cuestión.
2. Analizan su atributo \$DATA del registro.
3. Sí el atributo \$DATA es residente, el contenido como tal está en el propio registro.
4. Sí el atributo \$DATA es no residente, el contenido del fichero está en un external clúster. A continuación, se parsea el RunList. (Runlist contiene los clústeres asociados al fichero)
5. Verificar si los clústers del Runlist están allocated o no. Si todos los clúster están allocated, la recuperación de datos es posible.
6. Si algunos clústers tienen de estado de allocated a 1, **se puede realizar una recuperación de datos parcial**. Los que están allocated cambiarlos a 0.
7. Si todos los clústers estuviesen a 1, significa que ya hay un fichero que utiliza todos los clúster

**Este tipo de técnicas si permite recuperar el nombre del fichero, ya que se obtiene del atributo \$FILE\_NAME.**

Otras técnicas de recuperación de archivos utilizando artefactos del sistema de archivos, serían:

- ◆ Carving de registros MFT en el espacio libre
- ◆ Búsqueda de registros MFT en el propio registro \$MFT, en el slack space.

## REFS

El sistema de archivos ReFS (Sistema de archivos resiliente) es el último sistema de archivos de Microsoft, diseñado para optimizar la disponibilidad de los datos, administrar de manera eficiente la escalabilidad para grandes cantidades de datos y garantizar la integridad de los datos mediante la llamada “resiliencia” a la corrupción de archivos.

ReFS fue diseñado para hacer frente a los nuevos escenarios de crecimiento de datos y como base para futuras innovaciones. ReFS se introdujo con Windows Server 2012, y luego también para Windows 8 y las últimas versiones de Windows 10. Desde su primera versión, se han introducido otras características importantes, especialmente con Windows Server 2016 y Windows Server 2019.

En comparación con NTFS, ReFS presenta características clave para mejorar la resistencia a la corrupción, el rendimiento y la escalabilidad de los datos. Para entrar en la práctica, debe tenerse en cuenta que, en todos los sistemas operativos más recientes de Windows, especialmente en servidores, podemos crear fácilmente unidades y particiones con formato ReFS.

Estos son algunos de los beneficios clave del sistema de archivos ReFS:

### Resistencia

ReFS presenta nuevas funciones que pueden detectar con precisión la corrupción e incluso corregirla mientras permanece en línea, lo que ayuda a proporcionar una mayor integridad y disponibilidad de los datos:

- ◆ **Integrity-stream:** ReFS utiliza sumas de comprobación para metadatos y, opcionalmente, para datos de archivos, lo que permite a ReFS detectar de manera confiable la corrupción del sistema de archivos.
- ◆ **Integración con la función de espacios de almacenamiento:** cuando se usa junto con un espejo o espacio de paridad, ReFS puede reparar automáticamente los daños detectados utilizando la copia alternativa de los datos proporcionados por los espacios de almacenamiento.
- ◆ **Corrección de errores proactiva:** además de validar los datos antes de las operaciones de lectura y escritura, ReFS introduce un escáner de integridad de datos, conocido como depurador. Este depurador analiza periódicamente el volumen, identificando corrupciones latentes y activando proactivamente una reparación de datos corruptos.

### Rendimiento

ReFS introduce nuevas características para cargas de trabajo virtualizadas y sensibles al rendimiento. La optimización de niveles en tiempo real, la clonación de bloques y la VDL dispersa son buenos ejemplos de las capacidades en evolución de ReFS, diseñadas para soportar cargas de trabajo dinámicas y diversas:

- ◆ **Paridad acelerada por duplicación:** esta característica ofrece un alto rendimiento y, en conjunto, un almacenamiento de datos más eficiente. Para hacer esto, ReFS divide un volumen en dos grupos de almacenamiento lógico, conocidos como niveles. Estos niveles pueden tener sus propias unidades y tipos de resistencia, lo que permite que cada nivel optimice el rendimiento o la capacidad.
- ◆ **Mejora del rendimiento para máquinas virtuales Hyper-V:** ReFS presenta nuevas características específicamente diseñadas para mejorar el rendimiento de las cargas de trabajo virtualizadas



- ◆ Clonación de bloques: la clonación de bloques acelera las operaciones de copia, lo que permite operaciones de combinación de puntos de control de máquinas virtuales de menor impacto y más rápido.
- ◆ VDL disperso: ReFS le permite cero archivos rápidamente (relleno cero), lo que reduce el tiempo que lleva crear VHD fijos de minutos a segundos.
- ◆ Tamaño de clúster variable: ReFS admite tamaños de clúster de 4K y 64K. 4K es el tamaño de clúster recomendado para la mayoría de las distribuciones, pero los clústeres de 64 K son adecuados para grandes cargas de trabajo de E / S secuenciales.

### Escalabilidad

ReFS está diseñado para admitir conjuntos de datos extremadamente grandes, millones de terabytes, sin afectar el rendimiento, lo que resulta en una mayor escalabilidad que los sistemas de archivos anteriores.

Si comparamos las características con NTFS podemos ver que comparten algunas de las características de NTFS que hemos visto anteriormente.

Característica	ReFS	NTFS
<b>Cifrado BitLocker</b>	<b>Si</b>	<b>Si</b>
Duplicación de datos	Si	Si
Cluster Shared Volume (CSV) soporte	Si	Si
Soft links	Si	Si
Failover cluster support	Si	Si
Listas de control de acceso	Si	Si
<b>USN journal</b>	<b>Si</b>	<b>Si</b>
Notificaciones de cambios	Si	Si
Junction points	Si	Si
<b>Mount points</b>	<b>Si</b>	<b>Si</b>
<b>Reparse points</b>	<b>Si</b>	<b>Si</b>
<b>Instantáneas de volumen</b>	<b>Si</b>	<b>Si</b>
File IDs	Si	Si
Oplocks	Si	Si
Archivos dispersos	Si	Si
Named streams	Si	Si
Thin Provisioning	Si	Si
Offloaded Data Transfer (ODX)	No	Si
Trim/Unmap	Si	Si

En ReFS no hay tabla MFT como tal.

## ATRIBUTOS

Los atributos son los mismos que en NTFS, la información de cada fichero esta guardada en los atributos, incluido \$FILE\_NAME y \$SECURITY\_DESCRIPTOR:

Offset(h)	00	01	02	03	04	05	06	07	08	09	0A	0B	0C	0D	0E	0F	10	11	12	13	14	15	16	17	18	19	1A	1B	1C	1D	1E	1F	
000123000	24	00	53	00	54	00	41	00	4E	00	44	00	41	00	52	00	44	00	5F	00	49	00	4E	00	46	00	4F	00	52	00	4D	00	\$S.T.A.N.D.A.R.D._I.N.F.O.R.M.
000123020	41	00	54	00	49	00	4F	00	4E	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	A.T.I.O.N.
000123040	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	
000123060	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	
000123080	10	00	00	00	00	00	00	00	00	00	00	00	00	40	00	00	30	00	00	00	00	00	00	00	30	00	00	00	00	00	00	00	0
0001230A0	24	00	41	00	54	00	54	00	52	00	49	00	42	00	55	00	54	00	45	00	5F	00	4C	00	49	00	53	00	54	00	00	00	\$A.T.T.R.I.B.U.T.E._L.I.S.T.
0001230C0	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	
0001230E0	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	
000123100	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	
000123120	20	00	00	00	00	00	00	00	00	00	00	00	00	80	00	00	00	00	00	00	00	00	00	00	FF	FF	FF	FF	FF	FF	FF	FF	c
000123140	24	00	46	00	49	00	4C	00	45	00	5F	00	4E	00	41	00	4D	00	45	00	00	00	00	00	00	00	00	00	00	00	00	00	\$F.I.L.E._N.A.M.E.
000123160	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	
000123180	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	
0001231A0	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	
0001231C0	30	00	00	00	00	00	00	00	00	00	00	00	00	42	00	00	00	44	00	00	00	00	00	00	42	02	00	00	00	00	00	00	0
0001231E0	24	00	56	00	4F	00	4C	00	55	00	4D	00	45	00	5F	00	56	00	45	00	52	00	53	00	49	00	4F	00	4E	00	00	00	\$V.O.L.U.M.E._V.E.R.S.I.O.N.

## FICHEROS

A la hora de almacenar los ficheros, ReFS tiene más similitudes con FAT32 que con NTFS. ReFS almacena el contenido de los ficheros en unos offsets especiales donde ninguna otra información puede ir ahí, este mismo funcionamiento es como en FAT32.

NTFS drive:																																				
Offset(h)	00	01	02	03	04	05	06	07	08	09	0A	0B	0C	0D	0E	0F	10	11	12	13	14	15	16	17	18	19	1A	1B	1C	1D	1E	1F				
04010A000	46	49	4C	45	30	00	03	00	B9	48	80	01	00	00	00	00	02	00	01	00	38	00	01	00	A0	01	00	00	00	04	00	00	FILE000...HE.....8.....			
04010A020	00	00	00	00	00	00	00	00	04	00	00	00	00	28	00	00	00	04	00	00	00	00	00	00	00	10	00	00	00	60	00	00	00	.....(.....)		
04010A040	00	00	00	00	00	00	00	00	48	00	00	00	18	00	00	00	CE	E5	1D	83	7F	2B	D0	01	B4	D2	56	A1	65	41	D0	01	00	00	.....H.....iA.f.+D.ÓV;eAD.	
04010A060	B4	D2	56	A1	65	41	D0	01	CE	E5	1D	83	7F	2B	D0	01	20	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	.....ÓV;eAD.iA.f.+D. ....		
04010A080	00	00	00	00	0C	01	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	30	00	00	00	78	00	00	00	.....0.....x.....		
04010A0A0	00	00	00	00	00	00	02	00	5E	00	00	00	18	00	01	00	05	00	00	00	00	00	00	05	00	CE	E5	1D	83	7F	2B	D0	01	00	00	.....^.....iA.f.+D.
04010A0C0	CE	E5	1D	83	7F	2B	D0	01	CE	E5	1D	83	7F	2B	D0	01	CE	E5	1D	83	7F	2B	D0	01	00	00	00	00	00	00	00	00	00	00	.....iA.f.+D.iA.f.+D.iA.f.+D.	
04010A0E0	00	00	00	00	00	00	00	00	20	00	00	00	00	00	00	00	0E	00	68	00	65	00	6C	00	6C	00	6F	00	77	00	6F	00	00	00	.....h.e.l.l.o.w.o.	
04010A100	72	00	6C	00	64	00	2E	00	64	00	6F	00	63	00	00	40	00	00	00	28	00	00	00	00	00	00	00	00	00	00	00	00	00	00	.....r.l.d...d.o.c...@...()	
04010A120	10	00	00	00	18	00	00	16	77	09	C1	55	AD	E4	11	94	17	E4	66	86	94	ED	AA	80	00	00	00	00	60	00	00	00	00	00	.....w.áU.Ä."Äf"*E.....	
04010A140	00	00	18	00	00	00	01	00	43	00	00	00	18	00	00	00	49	20	6C	69	6B	65	20	64	69	6E	6F	73	61	75	72	73	00	00	.....C.....I like dinosaurs	
04010A160	2E	20	4D	79	20	66	61	76	6F	75	72	69	74	65	20	69	73	20	61	20	56	65	6C	6F	63	69	72	61	70	74	6F	72	00	00	.....My favourite is a Velociraptor	
04010A180	20	61	6E	64	20	61	20	70	74	65	72	6F	64	61	63	74	79	6C	2E	00	00	00	00	00	FF	FF	FF	FF	82	79	47	11	00	00	.....and a pterodactyl.....YYYY,yG.	

En FAT y REFS no aparece ningún indicador del fichero:

[illegible]

## PAPELERA DE RECICLAJE

La papelera de reciclaje sigue existiendo como en NTFS, pero ya no dispone del registro FILE0:

Offset(h)	00	01	02	03	04	05	06	07	08	09	0A	0B	0C	0D	0E	0F	10	11	12	13	14	15	16	17	18	19	1A	1B	1C	1D	1E	1F				
000758540	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	.....			
000758560	78	00	00	00	00	00	1C	00	00	00	00	30	00	48	00	00	00	00	00	30	02	00	24	00	52	00	45	00	43	00	59	00	43	00	x.....0.H...0...S.R.E.C.Y.C.	
000758580	4C	00	45	00	2E	00	42	00	49	00	4E	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	L.E...B.I.N		
0007585A0	86	98	ED	46	7D	2B	D0	01	6D	36	F0	46	7D	2B	D0	01	6D	36	F0	46	7D	2B	D0	01	6D	36	F0	46	7D	2B	D0	01	6D	36	F0	+F)+D.m6DF)+D.m6DF)+D.m6DF)+D.m6DF)+D.
0007585C0	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	06	00	00	00	10	00	00	00	00	78	00	00	00	10	00	1A	00	00	.....x		
0007585E0	00	00	30	00	48	00	00	00	30	00	02	68	00	65	00	6C	00	6C	00	6F	00	66	00	6F	00	6C	00	64	00	65	00	00	00	...0.H...0...h.e.l.l.o.f.o.l.d.e.		
000758600	72	00	00	00	00	00	00	00	03	07	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	10	CB	15	9F	7D	2B	D0	01	00	E.....E.Y)+D.	
000758620	10	CB	15	9F	7D	2B	D0	01	10	CB	15	9F	7D	2B	D0	01	10	CB	15	9F	7D	2B	D0	01	00	00	00	00	00	00	00	00	00	00	E.Y)+D...E.Y)+D...E.Y)+D...	

Como veremos mas adelante, NTFS guarda la información de un fichero borrado mediante nombres que empiezan por \$R que contiene el contenido del fichero y mediante \$I que contiene la información de donde estaba el fichero. Este comportamiento es el mismo en ReFS:

[illegible]

Uno de los programas que en la actualidad soporta la recuperación de ficheros, es R-Studio.

## RECUPERACIÓN DE DATOS MEDIANTE CARVING

La recuperación de datos mediante carving, se basa en ir analizando clúster a clúster del dispositivo de almacenamiento en busca de cabeceras o formas conocidas. Esta técnica NO permite recuperar el nombre del fichero. No depende del sistema de archivos, sino de conocer previamente como está formado el fichero que queremos recuperar

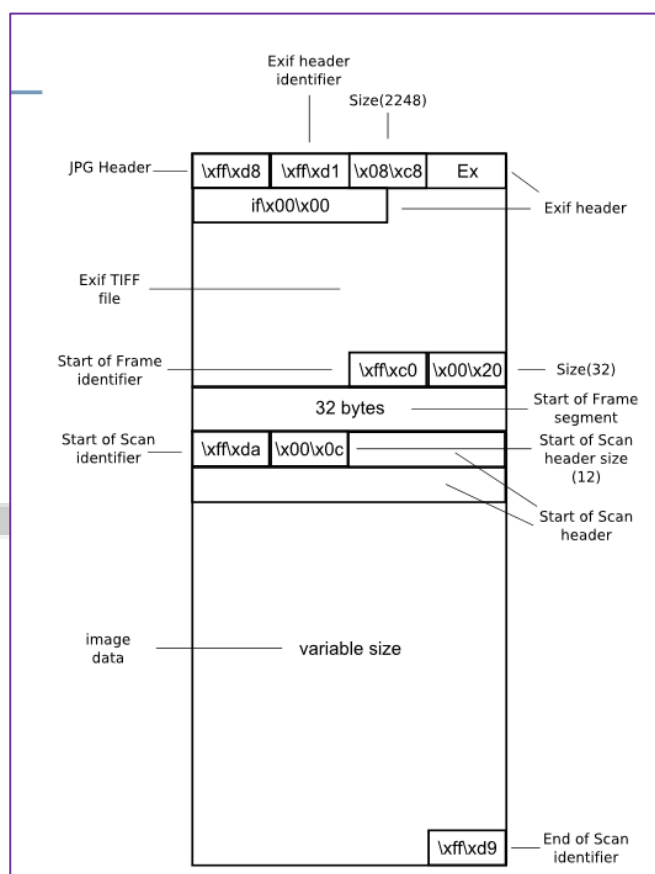
Tres tipos de técnicas:

- ◆ Basadas en cabeceras y cierre del fichero: Header-Footer con tamaño máximo del fichero
- ◆ Basadas en cómo está el fichero construido: File structure based carving
- ◆ Content based Carving -> basadas en el contenido en sí

### Técnica Header - Footer

Se utiliza un header conocido y un footer:

- ◆ Fichero JPEG Header: 0xFF,0xD8 -> FFD8
- ◆ Fichero JPEG Footer: 0xFF,0xD9 -> FFD9



Un listado de todas las cabeceras o firmas lo podemos encontrar en la siguiente URL:

[https://www.garykessler.net/library/file\\_sigs.html](https://www.garykessler.net/library/file_sigs.html)

¿Qué herramienta podemos utilizar para aplicar esta técnica? **Photorec**

¿Sirve con los programas de borrado seguro? No, ya que no podemos identificar headers/footers conocidos al haber sido sobrescritos por ciertos patrones grupos de ceros.

Photorec no funciona con formato de imagen Encase, solo en imágenes en RAW o sobre el dispositivo físico.

Dispone de muchísimas cabeceras y tipos de ficheros:

[https://www.cgsecurity.org/wiki/File\\_Formats\\_Recovered\\_By\\_PhotoRec](https://www.cgsecurity.org/wiki/File_Formats_Recovered_By_PhotoRec)

```
Administrator: Command Prompt - photorec_win.exe
PhotoRec 7.1-WIP, Data Recovery Utility, March 2018
Christophe GRENIER <grenier@cgsecurity.org>
https://www.cgsecurity.org

PhotoRec is free software, and
comes with ABSOLUTELY NO WARRANTY.

Select a media (use Arrow keys, then press Enter):
>Disk /dev/sda - 64 GB / 60 GiB (RO) - VMware, VMware Virtual S
Drive D: - 4243 MB / 4046 MiB (RO) - NECVMWar VMware SATA CD01

>[Proceed] [Quit]

Note:
Disk capacity must be correctly detected for a successful recovery.
If a disk listed above has an incorrect size, check HD jumper settings and BIOS
detection, and install the latest OS patches and disk drivers.
```

*\*Ver video: 013/MÓD. 2 – Photorec*

La técnica de carving no es válida para ficheros que estén desfragmentados ya que los clústers que ocupan el fichero no están contiguos.

¿Qué podríamos recuperar mediante técnicas de carving?

- ◆ Registros MFT
- ◆ \$Logfile
- ◆ \$Usnjrnl
- ◆ Shadows Copies
- ◆ \$I30

Las shadows se pueden recuperar gracias a la siguiente herramienta:

[https://github.com/mnrkbys/vss\\_carver](https://github.com/mnrkbys/vss_carver)

En el github aparece como y qué comandos son necesarios para poder utilizarla:

#### vss\_carver.py

- It can carve Store data from a disk image.
- It can regenerate Catalog data from carved Store data.
- If there is a Catalog in a disk image, that is merged with carved information (Catalog takes precedence).

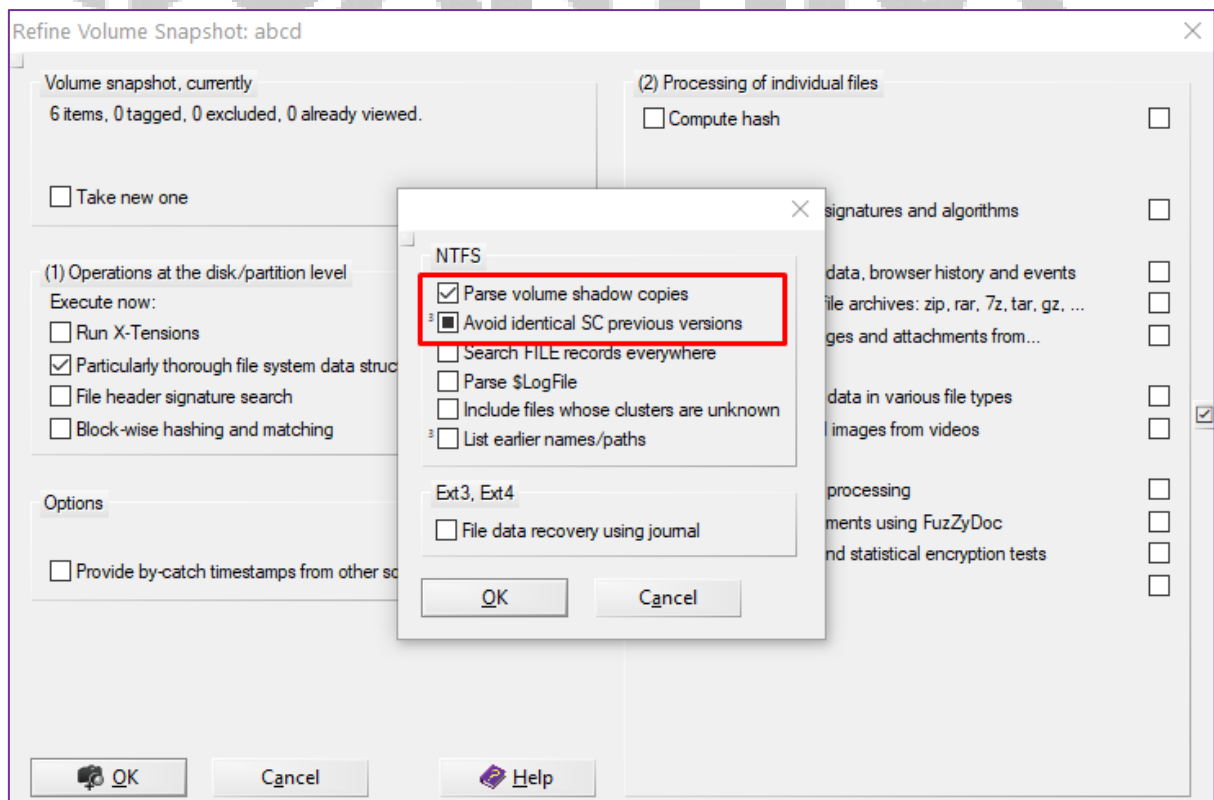
#### vss\_catalog\_manipulator.py

- It can manipulate the Catalog entries (change the order of entries, delete entries, and so on.)

#### extended-vshadowmount (based on libvshadow-20170902)

- We added two new options for reading reconstructed Catalog and carved Store.

Existe otra herramienta de pago que permite realizar la recuperación de shadows copies borradas, así como interpretarlas, [X-Ways Forensics](#):





## STREAM CARVING VS FILE CARVING

El data Stream Carving se basa en extraer fragmentos de las siguientes fuentes de información:

- ◆ Memoria / PageFile
- ◆ Espacio libre
- ◆ Ficheros de bases de datos

Ejemplos:

- ◆ URLs (navegación privada)
- ◆ Sesiones de Chat
- ◆ E-mails
- ◆ Claves de Cifrado

El **data stream carving** está basado en la localización de pequeños fragmentos y no en la búsqueda del fichero entero. Los pequeños fragmentos pueden tener un gran significado, pero normalmente son parte de un fichero. Por ejemplo, las bases de datos como el fichero index.dat puede contener cientos de URL que han sido visitadas cuando se borraron. Si no eres capaz de recuperar el fichero completo se podrían extraer fragmentos.

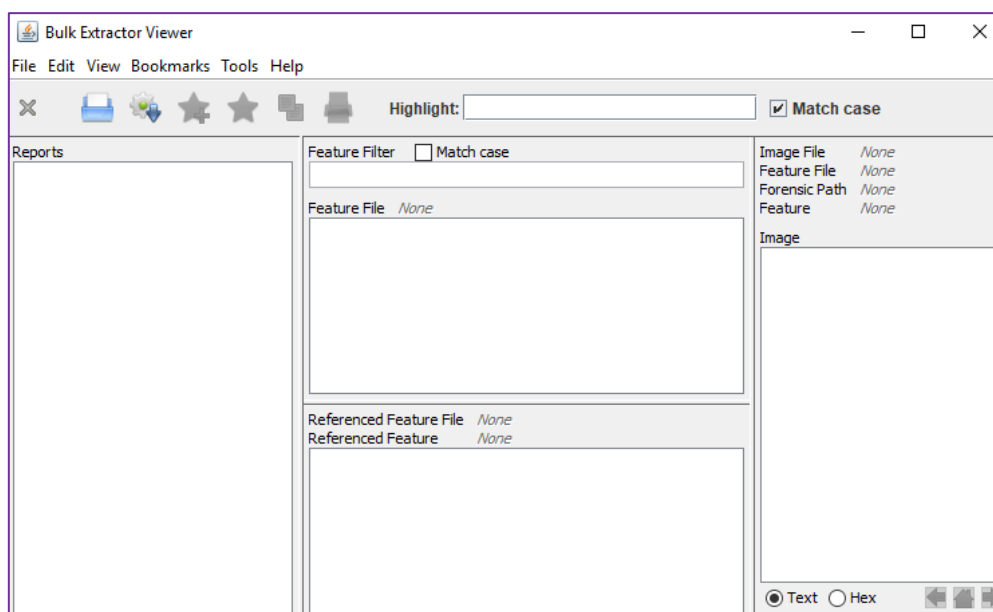
El file Carving o carving a secas, puede extraer ficheros de:

- ◆ Memoria / PageFile
- ◆ Espacio libre

Ejemplos:

- ◆ Word (.doc/.docx)
- ◆ Imágenes (.jpg,.png,.gif)
- ◆ Archivos (.zip /.rar)

Una herramienta que permite hacer ambos tipos de recuperación es **Bulk Extractor**, soporta imágenes forenses en formato E01:

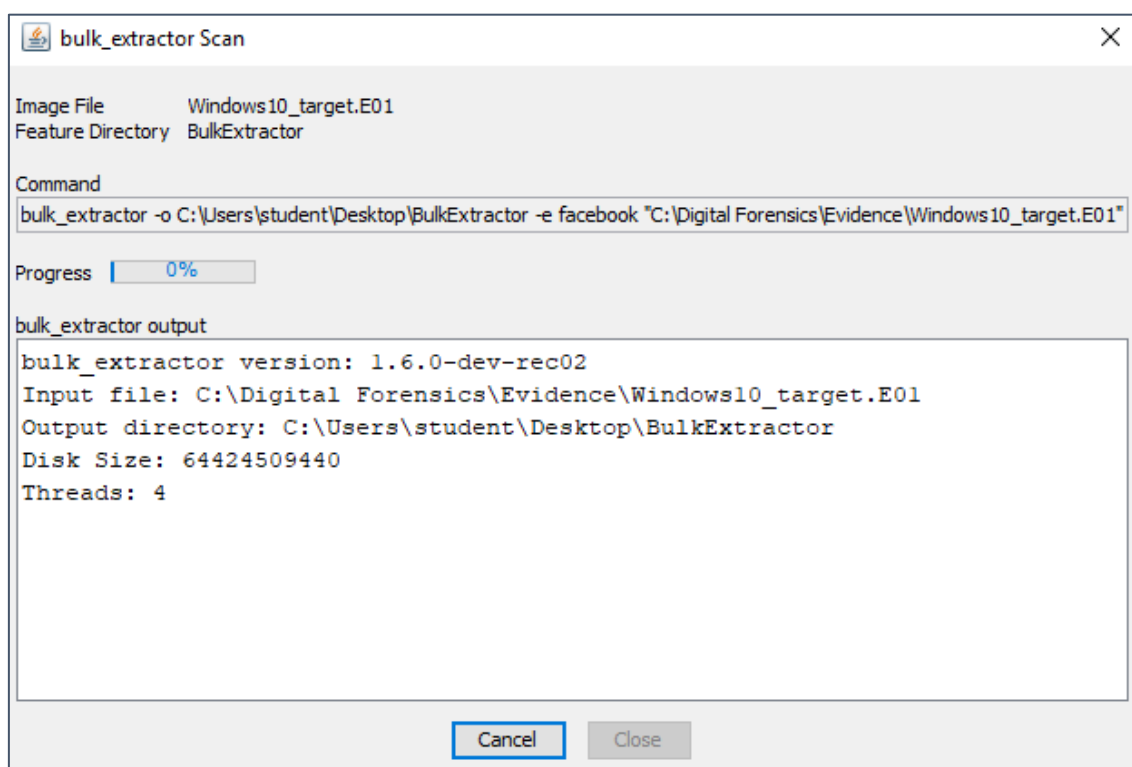


También permite realizar carving de:

- ◆ Registros MFT
- ◆ \$Logfile
- ◆ \$Usnjrnl
- ◆ \$I30

*\*Ver video: 014/MÓD. 2 - Bulk Extractor*

Una vez seleccionemos con Bulk Extractor la imagen a procesar, podemos ver el estado de cómo va avanzado:



Todos los resultados los guardará sobre la carpeta que hemos indicado.



## TIMELINE

El timeline o línea del tiempo nos permite obtener acotar en tiempo la investigación de una evidencia, por ejemplo: nos indican que la investigación está focalizada a raíz de la comunicación a un trabajador que va a ser despedido. La compañía a raíz de esta comunicación ha visto un comportamiento sospechoso.

- ◆ Fecha de comunicación de despido: mínima fecha a investigar
- ◆ Fecha de salida de la compañía: máxima fecha a investigar

Nuestro timeline estará basado únicamente en los timestamps del sistema de archivos, es decir, en las fechas de:

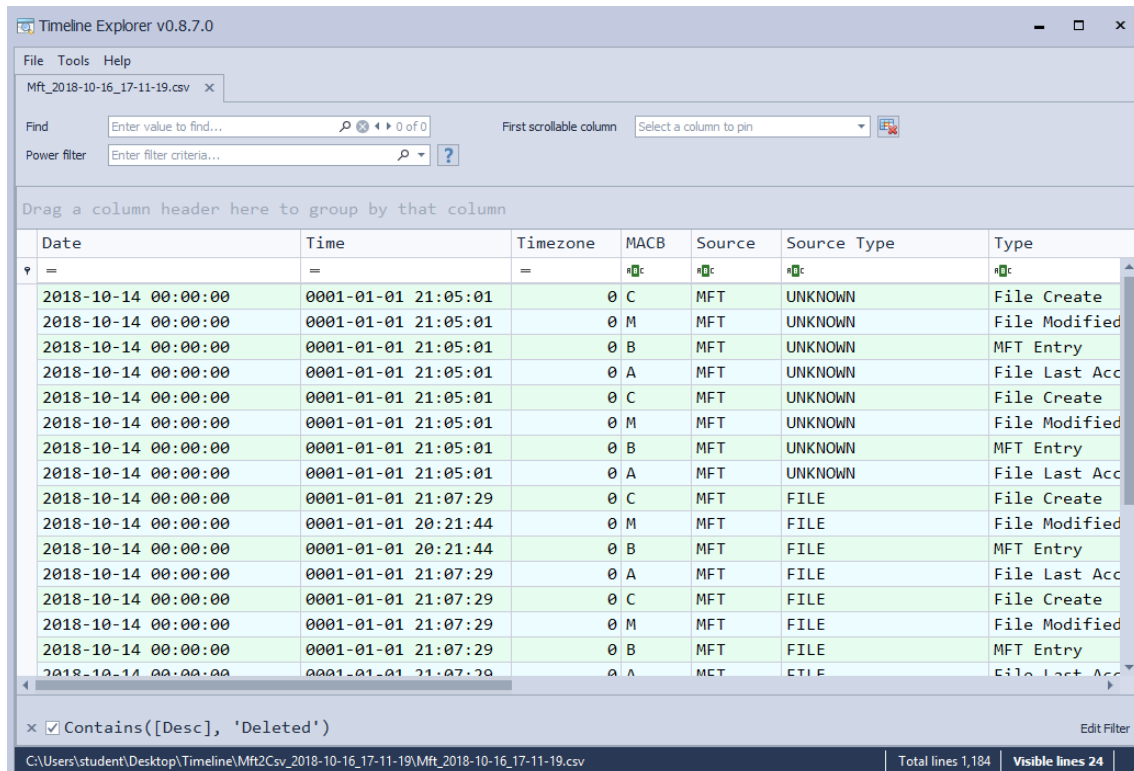
- ◆ Creación de fichero -> C
- ◆ Modificación de fichero -> M
- ◆ Cambio de registro MFT ->B
- ◆ Acceso -> A

Estas fechas se le denominan MACB timestamp, para crear el Timeline podemos utilizar la herramienta Autopsy:



*\*Ver video: 015/MÓD. 2 - Autopsy*

Otra manera de generar un Timeline sería con la herramienta MFT2CSV, pero seleccionando el formato de salida log2timeline. Una vez se haya generado el CSV, deberemos de abrirlo con la herramienta Timeline Explorer.



*\*Ver Video: 016/MÓD. 2 - Timeline*

Los metadatos están definidos como estructura interna del propio fichero en sí. No tiene nada que ver con los metadatos que hemos visto del sistema de archivos. ¿Qué tipos de ficheros pueden contener metadatos?

- ◆ Imágenes
- ◆ Documentos ofimáticos
- ◆ Documentos de Audio
- ◆ Documentos de video
- ◆ Ejecutables

¿Qué metadatos apodemos encontrar en un fichero ofimático?

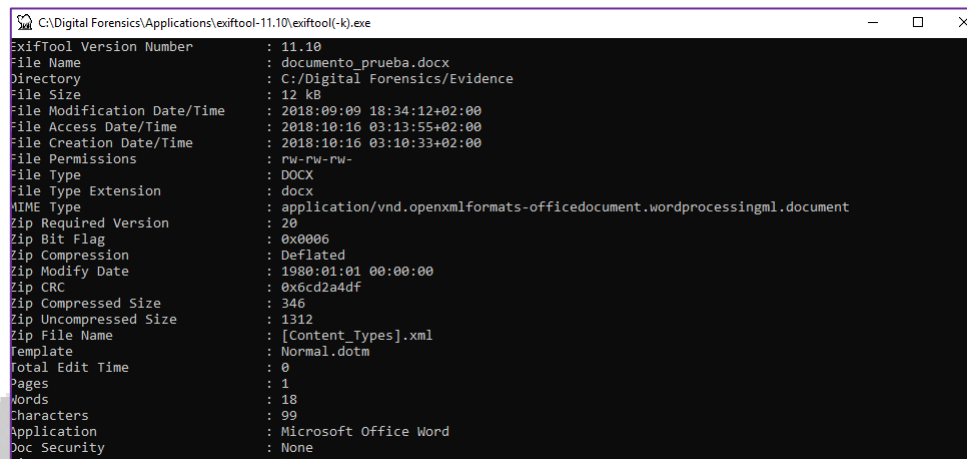
- ◆ Título
- ◆ Versión de MS Office Utilizada
- ◆ Último Autor
- ◆ Creación del documento /hora
- ◆ Ultima fecha de guardado

- ◆ Última fecha de impresión

¿Qué metadatos Podemos encontrar en una imagen?

- ◆ EXIF Data
- ◆ Marca Cámara / Modelo
- ◆ Hora de creación
- ◆ Hora de modificación
- ◆ Información de la imagen
- ◆ Coordenadas GPS

Nota: estos metadatos son muy fáciles de manipular. Para realizar una investigación correcta, se deben utilizar más artefactos y verificar dicha información.



```
C:\Digital Forensics\Applications\exiftool-11.10\exiftool(-k).exe
exiftool Version Number      : 11.10
File Name                    : documento_prueba.docx
Directory                    : C:/Digital Forensics/Evidence
File Size                    : 12 kB
File Modification Date/Time  : 2018:09:09 18:34:12+02:00
File Access Date/Time       : 2018:10:16 03:13:55+02:00
File Creation Date/Time     : 2018:10:16 03:10:33+02:00
File Permissions             : rw-rw-rw-
File Type                    : DOCX
File Type Extension          : docx
MIME Type                    : application/vnd.openxmlformats-officedocument.wordprocessingml.document
Zip Required Version         : 20
Zip Bit Flag                  : 0x0006
Zip Compression              : Deflated
Zip Modify Date              : 1980:01:01 00:00:00
Zip CRC                      : 0x6cd2a4df
Zip Compressed Size          : 346
Zip Uncompressed Size        : 1312
Zip File Name                 : [Content_Types].xml
Template                     : Normal.dotm
Total Edit Time               : 0
Pages                        : 1
Words                        : 18
Characters                   : 99
Application                  : Microsoft Office Word
Doc Security                  : None
```

*\*Ver Video: 017/MÓD. 2 - Exif tool*

Exiftool también permite analizar de manera recursiva un directorio:

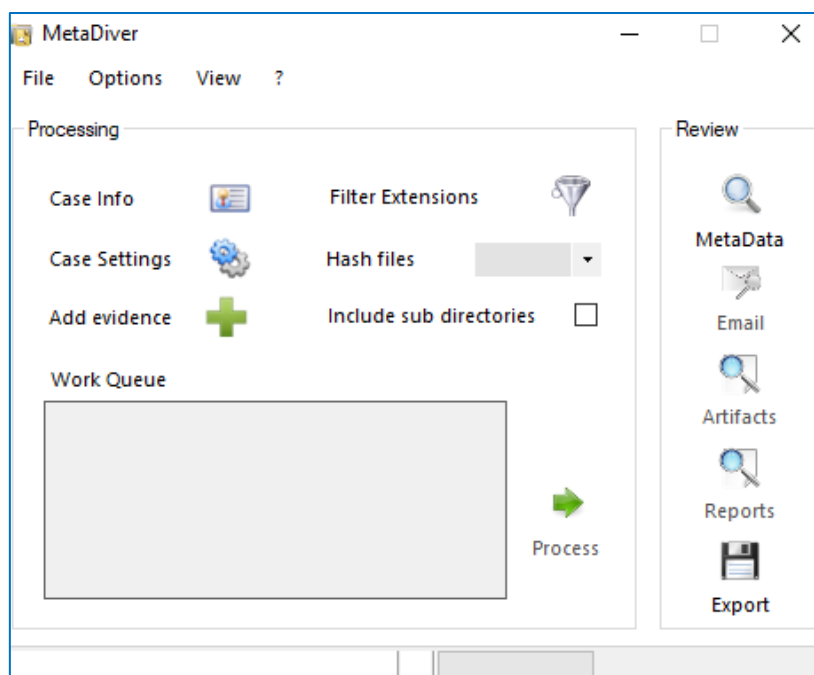
<https://owl.phy.queensu.ca/~phil/exiftool/examples.html>

Las fechas que aparecen como metadatos en el documento ofimático son introducidas por la herramienta ofimática. No confundir con las fechas del sistema de archivos.

Un documento como tal nunca contendrá fechas a no ser que disponga de algún metadato dentro él.

Otra herramienta muy buena para metadatos es Metadiver:

<https://www.easymetadata.com/metadiver-2/>



Características:

MetaDiver	Supported	OpenXML	ExifLib
Review metadata	✓	Apache Tika	ElasticSearch
Review document strings	✓	CDO/RDO - Email	SQLite
LNK and Jumplist parsing	✓		
Email (PST, MSG and RFC822)	✓	<b>Screenshots</b>	
Document metadata	✓		
Audio and Video metadata	✓	Processing files	
Image metadata	✓		
Excel and CSV output	✓	Reviewing results	
CSV and TSV Output	✓		
Case Info	✓		
Elastic Search	✓		
Tika file processing	✓		
Hashing	✓		
File Info	✓		