AUBURN

UNIVERSITY

FAT and NTFS image recovery

Jordan Sosnowski, Demarcus Campbell, Grant Pinkert

March 1, 2019

Executive Summary

We were tasked with analyzing two different disk image files: FAT.dd and FTFS.dd. With these images, we found the number of partitions and attempted to find existing and deleted files. For each file, we specified the hexadecimal address while also generating the FATs for the FAT image and we found the file attributes of each file in the NTFS image. We were able to recover several JPEG images as well as several PDFs while identifying several deleted files and subsequently recovering those as well.

Table of Contents

List of Figures

List of Tables

1 Introduction

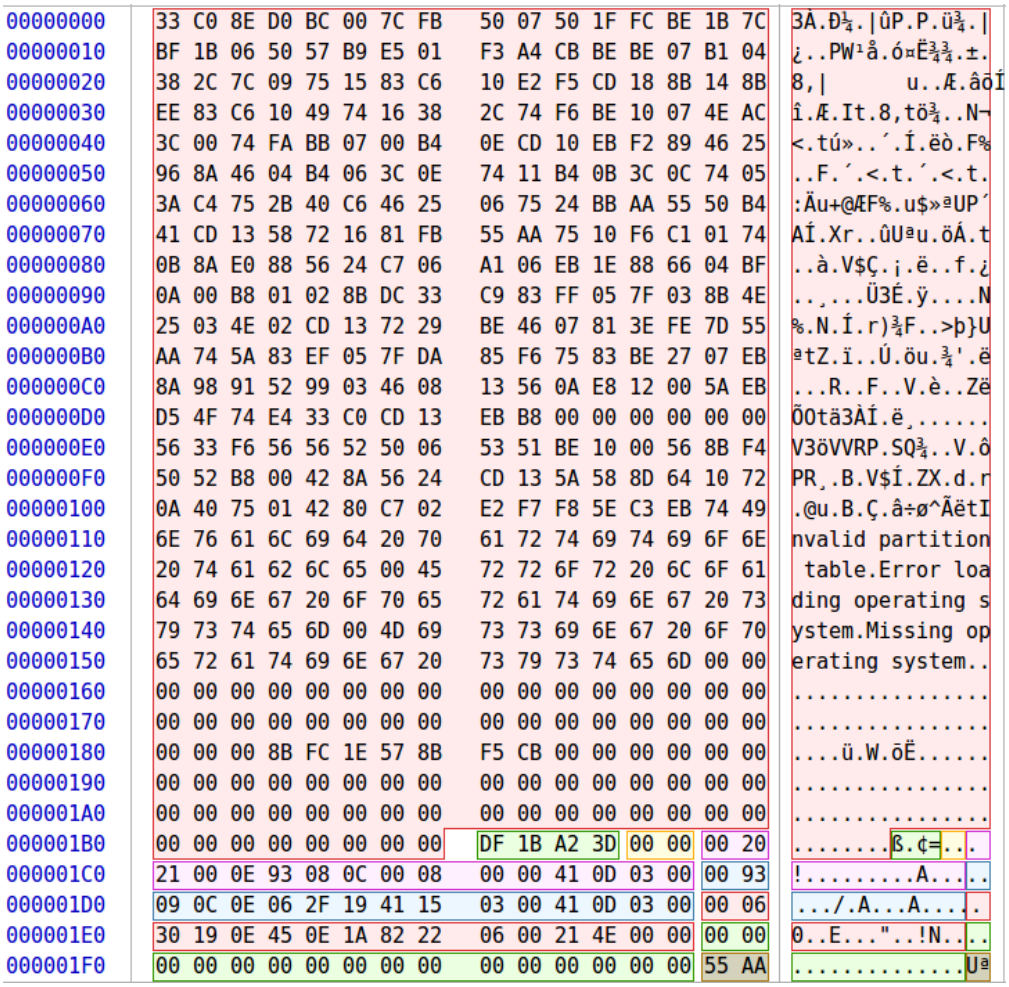
For this project we were given two image files, FAT.dd and NTFS.dd. With these files, we were tasked with finding out the details of each image. For both the FAT image and the NTFS image we were to find out the starting hexadecimal storage address. This address is needed because with each image, there is a small offset in the beginning with some image information, such as the partitions, the partition types, etc. which is needed to find out where each partition starts.

For the FAT image, we needed to generate the File Allocation Table. This table is comprised of a map showing where the clusters in the data area is located as well as how many active files are in the partition. After that we can go to the root directory, here we will find information about the file's name, size, if its deleted, etc. From the information found in the FAT and root directory we can locate the file in the data area and attempt to recover it.

For the NTFS image, we will need to find the Master File Table start sector. At the beginning of the MFT there are a few system files, each MFT entry is 1024 bytes. If we jump past a few entries, we will eventually find user files. Here, just like other MFT entry the entry is made up of file attributes. Using the values of the file attributes we can find the file name, file creation time, and even the file data itself.

After grabbing all these values for the various partitions and images we can finally attempt to recover the files. Deleted files are recovered just like the normal files. However, some deleted and active files are fragmented and are unable to be recovered using the normal process we would use for DD, to recover them it is easier to use icat.

2 FAT Image



00000000	33 C0 8E D0 BC 00 7C FB 50 07 50 1F FC BE 1B 7C	3À.Ð¼.|ûP.P.ü¾.|
00000010	BF 1B 06 50 57 B9 E5 01 F3 A4 CB BE BE 07 B1 04	¿..PW¹å.ó¤Ë¾¾.±.
00000020	38 2C 7C 09 75 15 83 C6 10 E2 F5 CD 18 8B 14 8B	8,| u..Æ.âõÍ..
00000030	EE 83 C6 10 49 74 16 38 2C 74 F6 BE 10 07 4E AC	î.Æ.It.8,tö¾..N¬
00000040	3C 00 74 FA BB 07 00 B4 0E CD 10 EB F2 89 46 25	<.tú»..´.Í.ëò.F%
00000050	96 8A 46 04 B4 06 3C 0E 74 11 B4 0B 3C 0C 74 05	..F.´.<.t.´.<.t.
00000060	3A C4 75 2B 40 C6 46 25 06 75 24 BB AA 55 50 B4	:Äu+@ÆF%.u$»ªUP´
00000070	41 CD 13 58 72 16 81 FB 55 AA 75 10 F6 C1 01 74	AÍ.Xr..ûUªu.öÁ.t
00000080	0B 8A E0 88 56 24 C7 06 A1 06 EB 1E 88 66 04 BF	..à.V$Ç.¡.ë..f.¿
00000090	0A 00 B8 01 02 8B DC 33 C9 83 FF 05 7F 03 8B 4E	..¸...Ü3É.ÿ....N
000000A0	25 03 4E 02 CD 13 72 29 BE 46 07 81 3E FE 7D 55	%.N.Í.r)¾F..>þ}U
000000B0	AA 74 5A 83 EF 05 7F DA 85 F6 75 83 BE 27 07 EB	ªtZ.ï..Ú.öu.¾'.ë
000000C0	8A 98 91 52 99 03 46 08 13 56 0A E8 12 00 5A EB	...R..F..V.è..Zë
000000D0	D5 4F 74 E4 33 C0 CD 13 EB B8 00 00 00 00 00 00	ÕOtä3ÀÍ.ë¸......
000000E0	56 33 F6 56 56 52 50 06 53 51 BE 10 00 56 8B F4	V3öVVRP.SQ¾..V.ô
000000F0	50 52 B8 00 42 8A 56 24 CD 13 5A 58 8D 64 10 72	PR¸.B.V$Í.ZX.d.r
00000100	0A 40 75 01 42 80 C7 02 E2 F7 F8 5E C3 EB 74 49	.@u.B.Ç.â÷ø^ÃëtI
00000110	6E 76 61 6C 69 64 20 70 61 72 74 69 74 69 6F 6E	nvalid partition
00000120	20 74 61 62 6C 65 00 45 72 72 6F 72 20 6C 6F 61	table.Error loa
00000130	64 69 6E 67 20 6F 70 65 72 61 74 69 6E 67 20 73	ding operating s
00000140	79 73 74 65 6D 00 4D 69 73 73 69 6E 67 20 6F 70	ystem.Missing op
00000150	65 72 61 74 69 6E 67 20 73 79 73 74 65 6D 00 00	erating system..
00000160	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00	....
00000170	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00	....
00000180	00 00 00 8B FC 1E 57 8B F5 CB 00 00 00 00 00 00	....ü.W.õË......
00000190	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00	....
000001A0	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00	....
000001B0	00 00 00 00 00 00 00 00 DF 1B A2 3D 00 00 00 20	........ß.¢=...
000001C0	21 00 0E 93 08 0C 00 08 00 00 41 0D 03 00 00 93	!.........A....
000001D0	09 0C 0E 06 2F 19 41 15 03 00 41 0D 03 00 00 06	.../.A...A.....
000001E0	30 19 0E 45 0E 1A 82 22 06 00 21 4E 00 00 00 00	0..E..."..!N....
000001F0	00 00 00 00 00 00 00 00 00 00 00 00 00 00 55 AA	..............Uª

Figure 1 – Active Disk Editor Partition Records Master Boot Record Example

Using Active Disk Editor, we can find the start of each partition. Partition I's first sector is located at 2048 (offset 0x1C6 from the beginning of the image). Partition II's first sector is located at 202,049 (offset 0x1D6). Partition III's first sector is located at 402,050 (offset 0x1E6). We can also see each partition file system ID's located at 0x01C2, 0x01D2, 0x01E2 respectively. They are all 0x0E which symbolizes FAT16.

Now that we have their sector locations, we need the offset in bytes. To calculate this, you need to multiply the sector by the value of bytes per sectors, we will assume the sector per byte count is 512[1]. Partitions in this image are located at the following address:

- Partition I: 2048 * 512 = 1,048,576
- Partition II: 202,049 * 512 = 103,449,088
- Partition III: 402,050 * 512 = 205,849,600

```
$ fdisk -l fat.dd
Disk fat.dd: 244 MiB, 255852544 bytes, 499712 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
Disklabel type: dos
Disk identifier: 0x3da21bdf

Device     Boot  Start    End Sectors  Size Id Type
fat.dd1          2048 202048  200001 97.7M  e W95 FAT16 (LBA)
fat.dd2        202049 402049  200001 97.7M  e W95 FAT16 (LBA)
fat.dd3        402050 422050   20001  9.8M  e W95 FAT16 (LBA)
```

Figure 2 - Fdisk Example

We can corroborate this information by using the tool fdisk. Using fdisk with the switch 'l' and the image's path it will list the disks partitions and information about it. As you can see each sector is sized 512 bytes and that the partitions start at the values that we got from the hex dump.

2.1 SEC PICS

We will not actually view the raw image in Active Disk Editor, but instead view each partition individually. This will make the address smaller due to the fact that the partitions offsets are not being added. We will have to add each partition's offset back when recovering the files however. That aside we will go to the first FAT Partition SEC PICS. This will bring us to the first reserved sector of the FAT partition. Here we can grab: bytes/sector, sectors/cluster, reserved sectors, number of fats, root entries, and total sectors.

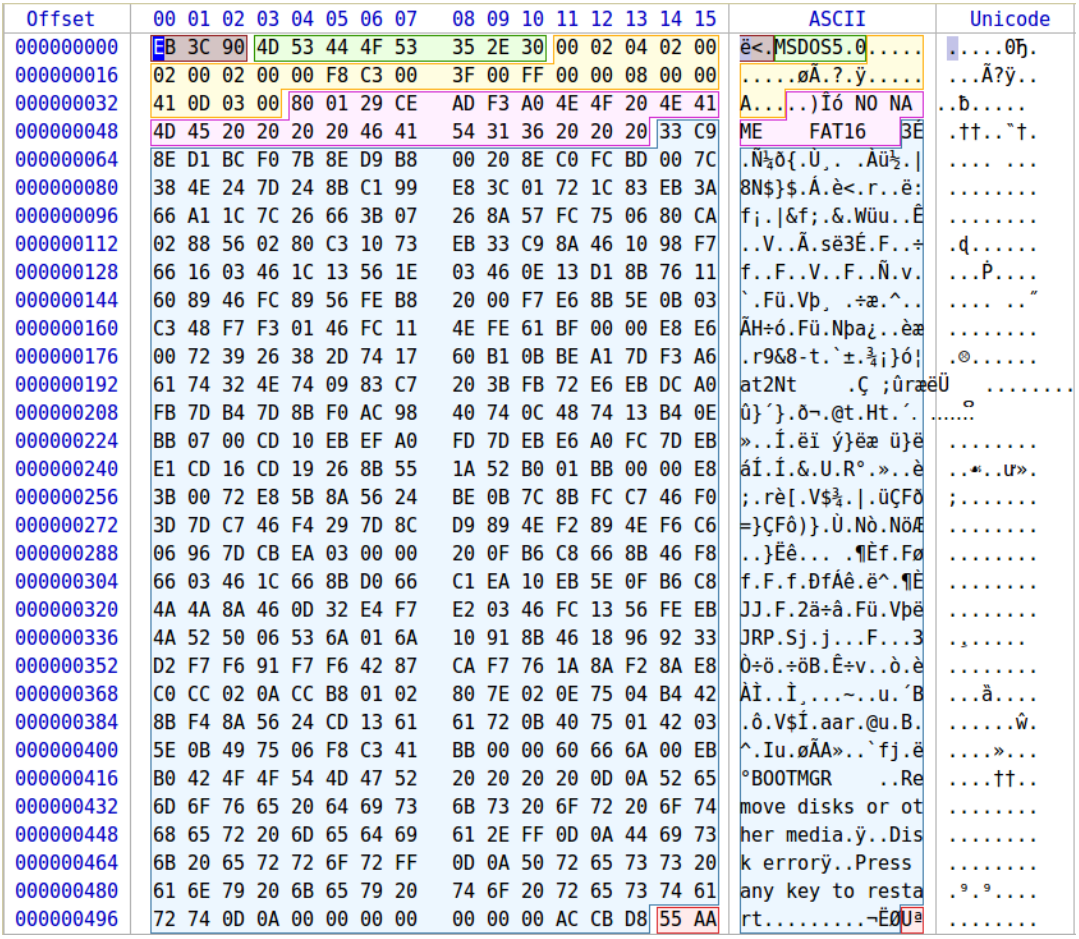
[1] Sectors are usually 512 bytes, sometimes 4096. One can see exactly how big a sector is in the partition's reserved area

Table 1 - FAT Information

Name	Offset	Value
Bytes/sector	0x00B	512
Sectors/cluster	0x00D	4
Number of FATs	0x010	2
Root Entries	0x011	512

Table 2 - FAT Offsets

Name	Offset	Value
Total Sectors [small]	0x013	0
Sectors Per FAT	0x016	195
Total Sectors [large]	0x020	200,001



Offset	00 01 02 03 04 05 06 07	08 09 10 11 12 13 14 15	ASCII	Unicode
000000000	EB 3C 90	4D 53 44 4F 53	35 2E 30	00 02 04 02 00	ë<.MSDOS5.0.....	.....0ђ.
000000016	02 00 02 00 00 F8 C3 00	3F 00 FF 00 00 08 00 00	.....øÃ.?.ÿ.....	...Ã?ÿ..
000000032	41 0D 03 00	80 01 29 CE	AD F3 A0 4E 4F 20 4E 41	A...€.)Íó NO NA	..Ђ.....
000000048	4D 45 20 20 20 20 46 41	54 31 36 20 20 20	33 C9	ME FAT16 3É	.††..`†.
000000064	8E D1 BC F0 7B 8E D9 B8	00 20 8E C0 FC BD 00 7C	.Ñ¼ð{.Ù¸. .Àü½.|	....
000000080	38 4E 24 7D 24 8B C1 99	E8 3C 01 72 1C 83 EB 3A	8N$}$.Á.è<.r..ë:	....
000000096	66 A1 1C 7C 26 66 3B 07	26 8A 57 FC 75 06 80 CA	f¡.|&f;.&.Wüu..Ê	....
000000112	02 88 56 02 80 C3 10 73	EB 33 C9 8A 46 10 98 F7	..V..Ã.së3É.F..÷	.ą.....
000000128	66 16 03 46 1C 13 56 1E	03 46 0E 13 D1 8B 76 11	f..F..V..F..Ñ.v.	...Ṗ....
000000144	60 89 46 FC 89 56 FE B8	20 00 F7 E6 8B 5E 0B 03	`.Fü.Vþ¸ .÷æ.^..	.... ..˝
000000160	C3 48 F7 F3 01 46 FC 11	4E FE 61 BF 00 00 E8 E6	ÃH÷ó.Fü.Nþa¿..èæ	....
000000176	00 72 39 26 38 2D 74 17	60 B1 0B BE A1 7D F3 A6	.r9&8-t.`±.¾¡}ó¦	.⊗.....
000000192	61 74 32 4E 74 09 83 C7	20 3B FB 72 E6 EB DC A0	at2Nt .Ç ;ûræëÜ	....
000000208	FB 7D B4 7D 8B F0 AC 98	40 74 0C 48 74 13 B4 0E	û}´}.ð¬.@t.Ht.´.	.....°
000000224	BB 07 00 CD 10 EB EF A0	FD 7D EB E6 A0 FC 7D EB	»..Í.ëï ý}ëæ ü}ë	....
000000240	E1 CD 16 CD 19 26 8B 55	1A 52 B0 01 BB 00 00 E8	áÍ.Í.&.U.R°.»..è	..◄..u».
000000256	3B 00 72 E8 5B 8A 56 24	BE 0B 7C 8B FC C7 46 F0	;.rè[.V$¾.|.üÇFð	;.......
000000272	3D 7D C7 46 F4 29 7D 8C	D9 89 4E F2 89 4E F6 C6	=}ÇFô)}.Ù.Nò.NöÆ	....
000000288	06 96 7D CB EA 03 00 00	20 0F B6 C8 66 8B 46 F8	..}Ëê... .¶Èf.Fø	....
000000304	66 03 46 1C 66 8B D0 66	C1 EA 10 EB 5E 0F B6 C8	f.F.f.ÐfÁê.ë^.¶È	....
000000320	4A 4A 8A 46 0D 32 E4 F7	E2 03 46 FC 13 56 FE EB	JJ.F.2ä÷â.Fü.Vþë	....
000000336	4A 52 50 06 53 6A 01 6A	10 91 8B 46 18 96 92 33	JRP.Sj.j...F...3	.ˌ.....
000000352	D2 F7 F6 91 F7 F6 42 87	CA F7 76 1A 8A F2 8A E8	Ò÷ö.÷öB.Ê÷v..ò.è	....
000000368	C0 CC 02 0A CC B8 01 02	80 7E 02 0E 75 04 B4 42	ÀÌ..Ì¸...€~..u.´B	...ȁ....
000000384	8B F4 8A 56 24 CD 13 61	61 72 0B 40 75 01 42 03	.ô.V$Í.aar.@u.B.	......ŵ.
000000400	5E 0B 49 75 06 F8 C3 41	BB 00 00 60 66 6A 00 EB	^.Iu.øÃA»..`fj.ë	....»...
000000416	B0 42 4F 4F 54 4D 47 52	20 20 20 20 0D 0A 52 65	°BOOTMGR ..Re	....††..
000000432	6D 6F 76 65 20 64 69 73	6B 73 20 6F 72 20 6F 74	move disks or ot	....
000000448	68 65 72 20 6D 65 64 69	61 2E FF 0D 0A 44 69 73	her media.ÿ..Dis	....
000000464	6B 20 65 72 72 6F 72 FF	0D 0A 50 72 65 73 73 20	k errorÿ..Press	....
000000480	61 6E 79 20 6B 65 79 20	74 6F 20 72 65 73 74 61	any key to resta	.⁹.⁹....
000000496	72 74 0D 0A 00 00 00 00	00 00 00 AC CB D8	55 AA	rt.........¬ËØU²	....

Figure 3 - Active Disk Editor FAT Reserved Sector

With the information we gathered above we can create a layout for the FAT structure. We know that the reserved sectors consist of 2 sectors, each FAT is 195. However, we do not know inherently how big the root directory is. However, we do have how many entry's we have in the root and we know that each entry is always 32 bytes. We know that the root directory is 16,384 total bytes (512 entries * 32 bytes) and from that we know that its 32 sectors long (16,384 bytes / 512 bytes/sector). We also know that there is a total of 200,001 sectors and we know where the root directory ends (reserved sector + 1st FAT + 2nd FAT + Root) which is 424. To get the size of the data area we can subtract 200,001 from 424 which gives us 199,579 sectors for the Data Area, and it starts at 424. From this information we can generate the image below.

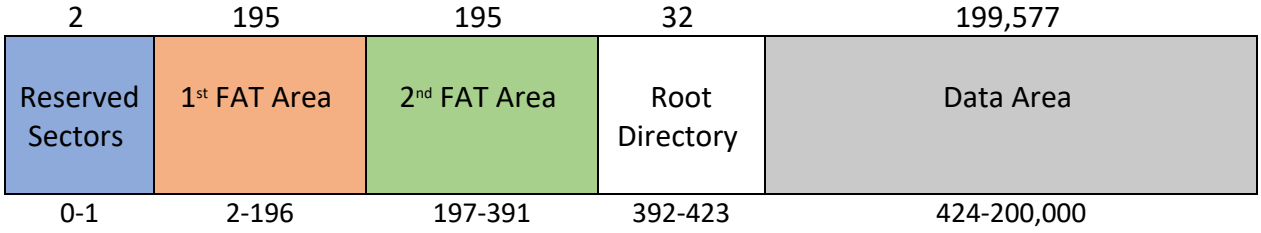
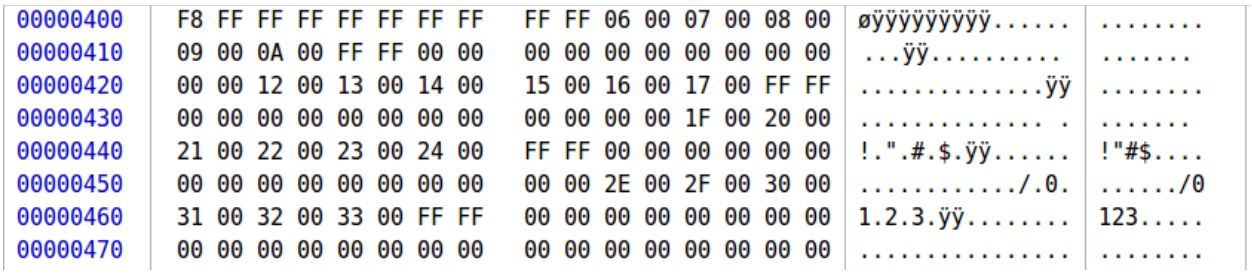


Figure 4 - FAT Format

After generating the layout of the partition, we will go to the 1st FAT area to see how the clusters are allocated. The first FAT is located at sector 2, which in bytes is 1024 (0x400). At that location we see the below image. For FAT16 each cluster takes up two bytes in the FAT. The first two entries are special values. The first entry in the FAT is the FAT ID, 0xF0 indicates a volume on a non-partition supper floppy drive, and 0xF8 for partitioned disks [1]. The second entry is the end of chain indicator, so for this FAT 0xFFFF marks the end of a chain.



00000400	F8 FF FF FF FF FF FF FF FF FF 06 00 07 00 08 00	øÿÿÿÿÿÿÿÿÿÿ.....	....
00000410	09 00 0A 00 FF FF 00 00 00 00 00 00 00 00 00 00	...ÿÿ..........	....
00000420	00 00 12 00 13 00 14 00 15 00 16 00 17 00 FF FF	..............ÿÿ	....
00000430	00 00 00 00 00 00 00 00 00 00 00 00 1F 00 20 00	....	....
00000440	21 00 22 00 23 00 24 00 FF FF 00 00 00 00 00 00	!."#$.ÿÿ......	!"#$....
00000450	00 00 00 00 00 00 00 00 00 00 2E 00 2F 00 30 00	............./.0.	....../0
00000460	31 00 32 00 33 00 FF FF 00 00 00 00 00 00 00 00	1.2.3.ÿÿ........	123.....
00000470	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00	....	....

Figure 5 - FAT Entry

After these special entries start actual files. The first few files on a partition are usually system files but we cannot always assume that. However, we can create the chart listed below from FAT. The green entries are user files that are active. The blue entries are empty clusters, which may or may not contain deleted files. The black entries are the special entries, and the orange ones are assumed to be system files. Which we can validate later in the root directory. After analyzing the FAT, we can go to the root directory.

Table 3 List of clusters and the cluster entry

Cluster	Next Cluster
0x0000	0xFFF8
0x0001	0xFFFF
0x0002	0xFFFF
0x0003	0xFFFF
0x0004	0xFFFF
0x0005	0x0006
0x0006	0x0007
0x0007	0x0008
0x0008	0x0009
0x0009	0x000A
0x000A	0xFFFF
0x000B	0x0000
0x000C	0x0000
0x000D	0x0000
0x000E	0x0000
0x000F	0x0000
0x0010	0x0000
0x0011	0x0012
0x0012	0x0013
0x0013	0x0014
0x0014	0x0015
0x0015	0x0016
0x0016	0x0017
0x0017	0xFFFF
0x0018	0x0000
0x0019	0x0000

Cluster	Next Cluster
0x001A	0x0000
0x001B	0x0000
0x001C	0x0000
0x001D	0x0000
0x001E	0x001F
0x001F	0x0020
0x0020	0x0021
0x0021	0x0022
0x0022	0x0023
0x0023	0x0024
0x0024	0xFFFF
0x0025	0x0000
0x0026	0x0000
0x0027	0x0000
0x0028	0x0000
0x0029	0x0000
0x002A	0x0000
0x002B	0x0000
0x002C	0x0000
0x002D	0x002E
0x002E	0x002F
0x002F	0x0030
0x0030	0x0031
0x0031	0x0032
0x0032	0x0033
0x0033	0xFFFF

Based on the table we created earlier the root directory starts at sector 392, or byte 200,704 (0x31000). The root table shows us directory entries, as well as all files, even those that are marked as deleted. Here, we find the filename, extension, attribute, modify date, cluster, and the file size, all of which is to be used to extract the file. At the root directory we see this. As stated early each listing in the root directory is 32 bytes long. As we assumed earlier the first 3 entries are system entries. At the fourth entry we can see the user file Tennesse.JPG

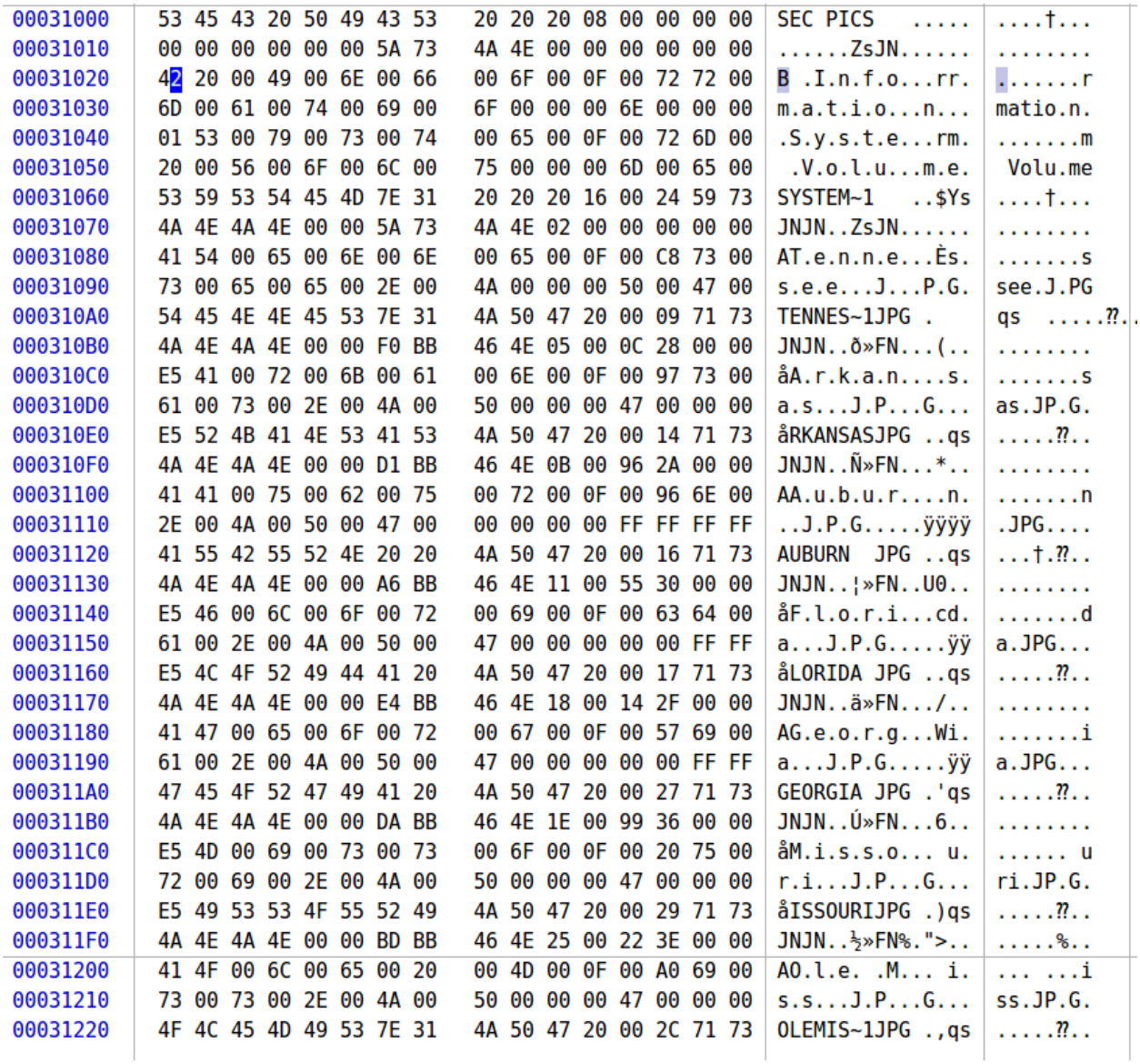


Figure 6 – Root Directory Entry

Each short entry will follow the same format. The entries prior to the short entries are long entries, which contain the full length of the filename if the short entry cannot hold it. From the short entries we can gather the following: file status, filename, file extension, attribute, modified date/time, starting cluster, and file size. With this information we can attempt to locate the individual files in the data area and hopefully recover them.

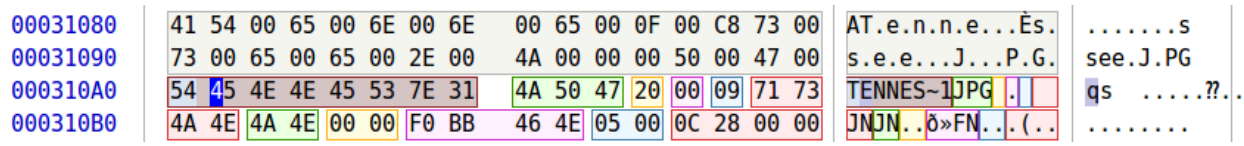


Figure 7 - FAT Directory Entry

1. Tennessee
    a. Status: 0x41 – Normal
    b. Filename: Tennessee
    c. Extension: JPG
    d. Attribute: 0x20 – Archive
    e. Modified date/time: 2/6/19 11:31 PM
    f. Cluster: 0x0005 – 5
    g. File Size: 0x0000280C – 10,252 bytes

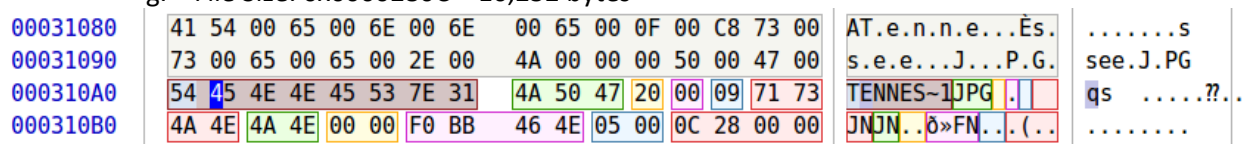


Figure 8 - FAT Directory Entry

2. Arkansas
    a. Status: 0xE5 – Deleted
    b. Filename: Arkansas
    c. Extension: JPG
    d. Attribute: 0x20 – Archive
    e. Modified date/time: 2/6/19 11:30 PM
    f. Cluster: 0x000B – 11
    g. File Size: 0x00002A96 – 10,902 bytes

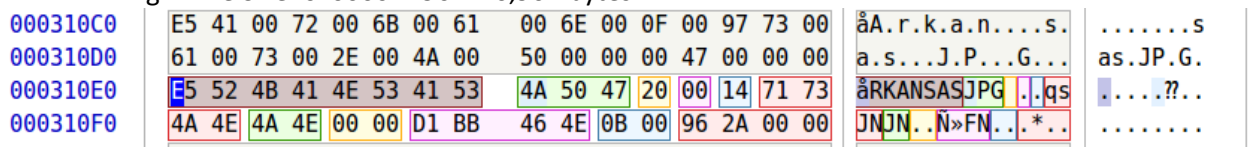


Figure 9 - FAT Directory Entry

3. Auburn
    a. Status: 0x41 – Normal File
    b. Filename: Auburn
    c. Extension: JPG
    d. Attribute: 0x20 – Archive

e. Modified data/time: 2/6/19 11:29 PM
f. Cluster: 0x0011 – 17
g. File Size: 0x00003055 – 12,373 bytes

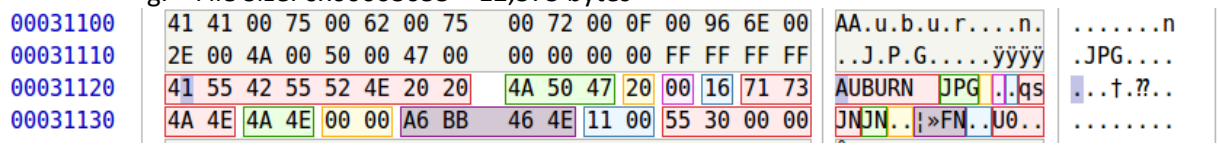


Figure 10 - FAT Directory Entry

4. Florida
   a. Status: 0xE5 – Deleted
   b. Filename: Florida
   c. Extension: JPG
   d. Attribute: 0x20 – Archive
   e. Modified date/time: 2/6/19 11:31 PM
   f. Cluster: 0x0018 – 24
   g. File Size: 0x00002F14 – 12,052

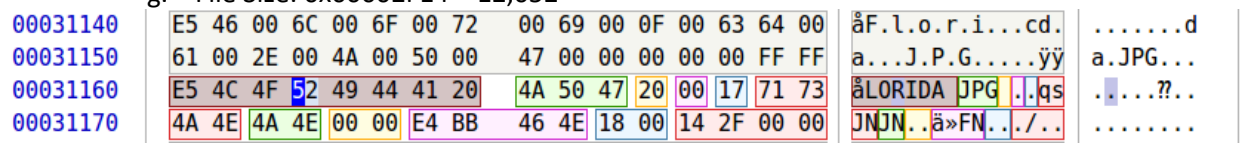


Figure 11 - FAT Directory Entry

5. Georgia
   a. Status: 0x41 – Normal File
   b. Filename: Georgia
   c. Extension: JPG
   d. Attribute: 0x20 – Archive
   e. Modified date/time: 2/6/19 11:30 PM
   f. Cluster: 0x001E – 30
   g. File size: 0x00003699 – 13,977 bytes

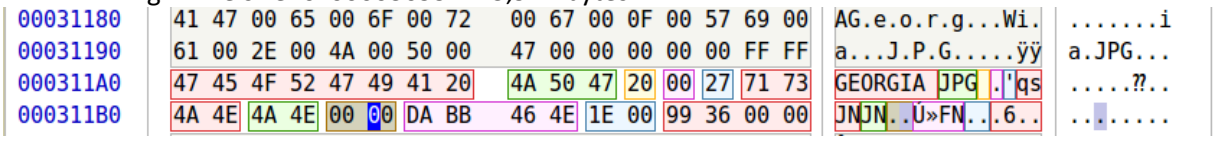


Figure 12 - FAT Directory Entry

6. Missouri
   a. Status: 0xE5 – Deleted
   b. Filename: Missouri
   c. Extension: JPG
   d. Attribute: 0x20 – Archive
   e. Modified date/time: 2/6/19 11:29 PM
   f. Cluster: 0x0025 – 37
   g. File size: 0x00003E22 – 15,906 bytes

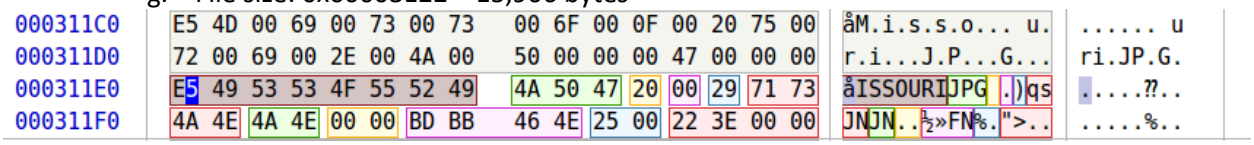


Figure 13 - FAT Directory Entry

7. Ole Miss
    a. Status: 0x41 – Normal File
    b. Filename: Ole Miss
    c. Extension: JPG
    d. Attribute: 0x20 – Archive
    e. Modified date/time: 2/6/19 11:30 PM
    f. Cluster: 0x002D – 45
    g. File size: 0x0000339B – 13,211 bytes

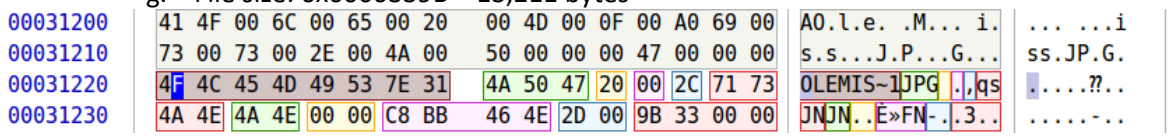


Figure 14 - FAT Directory Entry

Using all the information, now it is possible to recover the files. To get a file, we use the dd command as follows:

sudo dd if=image_name of=filename bs=block_size skip=blocks_to_skip count=blocks_to_grab

The skip value is the starting position of the file. This is found by finding the cluster number for whichever file, subtracting two from it, multiplying that value by the value of bytes per cluster (cluster offset), then adding the cluster offset to data offset (file offset), and then adding that to partition offset (actual offset). Using the root directory entry, we can then find the length of the file of bytes, which is used for the count argument. Block size is set to 1 in case a file does not end up using full sectors or clusters. Using the above information, we can get this table. If the commands were run in a terminal with the current directory hosting the fat.dd image the files would be correctly retrieved.

Table 4 - FAT File Recovery

Filename	Cluster	File Length	Cluster Offset	File Offset	Actual Offset	Command to Retrieve File
Tennessee.JPG	5	10,252	6,144	223,232	1,271,808	sudo dd if=fat.dd of='Tennessee.JPG' bs=1 skip=1271808 count=10252 status=progress
Arkansas.JPG	11	10,902	18,432	235,520	1,284,096	sudo dd if=fat.dd of='Arkansas.JPG' bs=1 skip=1284096 count=10902 status=progress
Auburn.JPG	17	12,373	30,720	247,808	1,296,384	sudo dd if=fat.dd of='Auburn.JPG' bs=1 skip=1296384 count=12373 status=progress
Florida.JPG	24	12,502	45,056	262,144	1,310,720	sudo dd if=fat.dd of='Florida.JPG' bs=1 skip=1310720 count=12502 status=progress
Georgia.JPG	30	13,977	57,344	274,432	1,323,008	sudo dd if=fat.dd of='Georgia.JPG' bs=1 skip=1323008 count=13977 status=progress
Missouri.JPG	37	15,906	71,680	288,768	1,337,344	sudo dd if=fat.dd of='Missouri.JPG' bs=1 skip=1337344 count=15906 status=progress
Ole Miss.JPG	45	13,211	88,064	305,152	1,353,728	sudo dd if=fat.dd of='Ole Miss.JPG' bs=1 skip=1353728 count=13211 status=progress

We can corroborate all the information we found with a Sleuth Kit, which is a collection of digital forensics tools, that essentially do all the work we did in a few lines. The first command we will want to use is mmls <image>. This will give us information about the volume's partitions. However we have already ran an equivalent command to that with fdisk -l, so we will skip it. The next command we will want to run is fsstat which gives us information about the file system. Running fsstat -o 2048 fat.dd will give us information about the first partition.

```
$ fsstat -o 2048 fat.dd
FILE SYSTEM INFORMATION
--------------------------------------------
File System Type: FAT16

OEM Name: MSDOS5.0
Volume ID: 0xa0f3adce
Volume Label (Boot Sector): NO NAME
Volume Label (Root Directory): SEC PICS
File System Type Label: FAT16

Sectors before file system: 2048

File System Layout (in sectors)
Total Range: 0 - 200000
* Reserved: 0 - 1
** Boot Sector: 0
* FAT 0: 2 - 196
* FAT 1: 197 - 391
* Data Area: 392 - 200000
** Root Directory: 392 - 423
** Cluster Area: 424 - 199999
** Non-clustered: 200000 - 200000

METADATA INFORMATION
--------------------------------------------
Range: 2 - 3193750
Root Directory: 2

CONTENT INFORMATION
--------------------------------------------
Sector Size: 512
Cluster Size: 2048
Total Cluster Range: 2 - 49895

FAT CONTENTS (in sectors)
--------------------------------------------
424-427 (4) -> EOF
428-431 (4) -> EOF
432-435 (4) -> EOF
436-459 (24) -> EOF
484-511 (28) -> EOF
536-563 (28) -> EOF
596-623 (28) -> EOF
```

Figure 15 - fsstat Example

Here we can see that our findings about the first FAT partition, SEC PICS, match up exactly.

The next command we will want to run is fls. This command will list file and directory names in the disk image. Note just like for fsstat we will have to give an offset to the correct partition.

Running fls -o 2048 fat.dd we get the following information

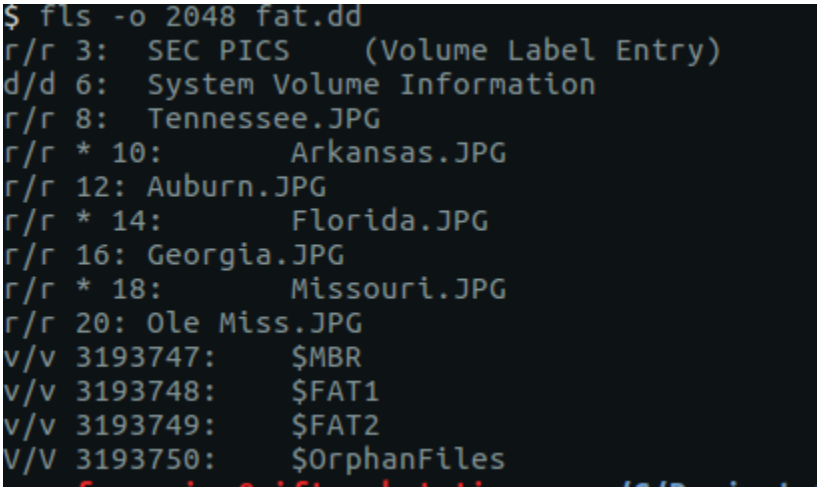


Figure 16 - fls Example

Here we can see the information about the file's names, if they are deleted, and their inodes. Using istat, which gives us details about meta-data structures (i.e. inode), we can print out information about the specific file. Using istat -o 2048 fat.dd 10, we can preview information about Arkansas.JPG.

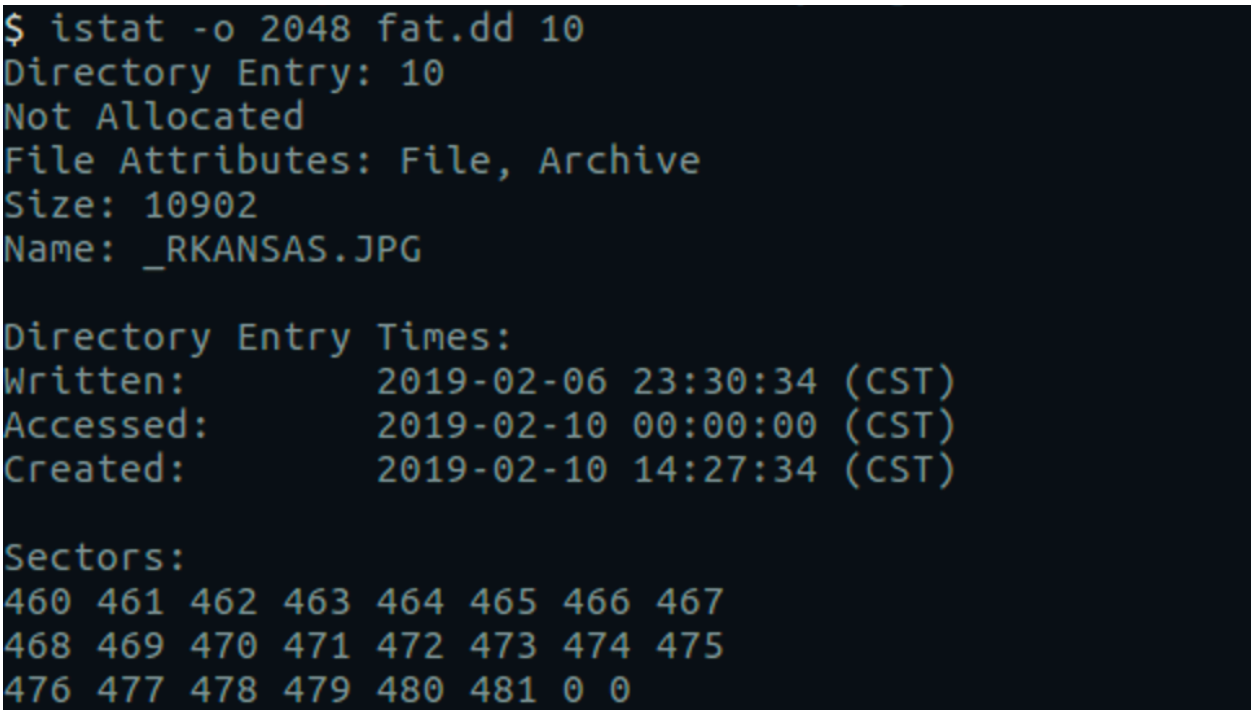


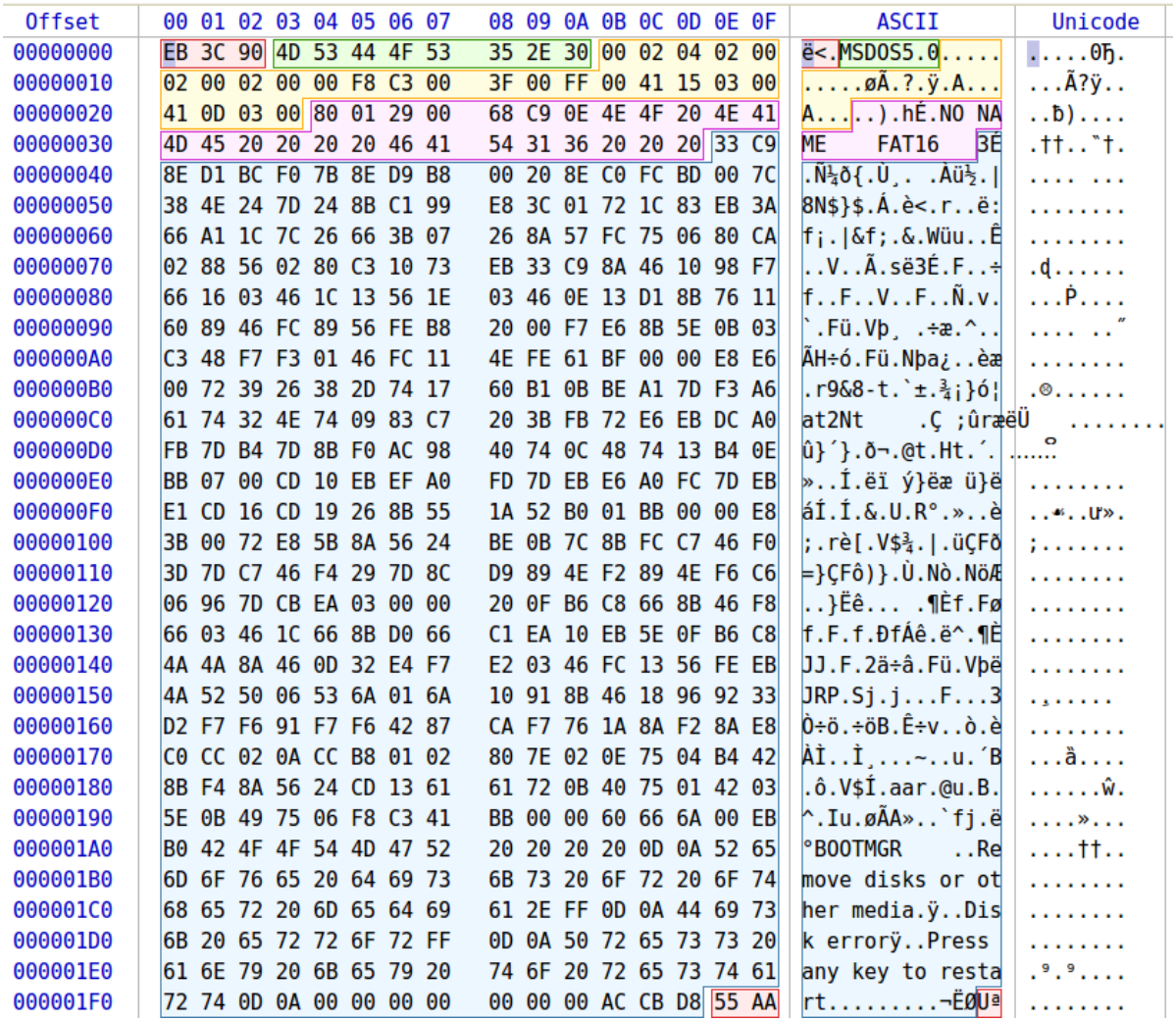
Figure 17 - istat Example

2.2 CLASSICS

The same methodology can be applied to other FAT Partitions. What follows will just be a quick rundown of the values found in the classics partition instead of a detailed analysis.

1. Structure Size Calculations
    - **a.** Partition Offset = 0x62A8200 = **103,449,088** bytes
    - b. Bytes / Sector
        - i. Offset 0x0B = 0x0200 = **512**
    - c. Sectors / Cluster
        - i. Offset 0x0D = 0x4 = **4**
    - d. Bytes / Cluster
        - i. 4 Sectors / Cluster * 512 Bytes / Sector = **2048**
    - e. Reserved Sectors
        - i. Offset 0x000E = 0x0002 = **2**
    - f. Number of FATs
        - i. Offset 0x0010 = 0x02 = **2**
    - g. 1st FAT Area Size
        - i. Offset 0x0016 = 0x00C3 = **195** sectors
    - h. 2nd FAT Area Size
        - i. Offset 0x0016 = 0x00C3 = **195** sectors
    - i. Root Directory
        - i. Directory Entries
            1. Offset 0x0011 = 0x0200 = 512 entries
        - ii. 512 * 32 bytes = 16,384 bytes / 512 (bytes/sector) = **32** sectors
    - j. Data Area
        - i. Total Sectors
            1. Offset 0x0020 = 0x00030D41 = 200,001 Sectors
        - ii. Data Area = Total Sectors – End of Root
            1. 200,001 – 424 = **199579** Sectors

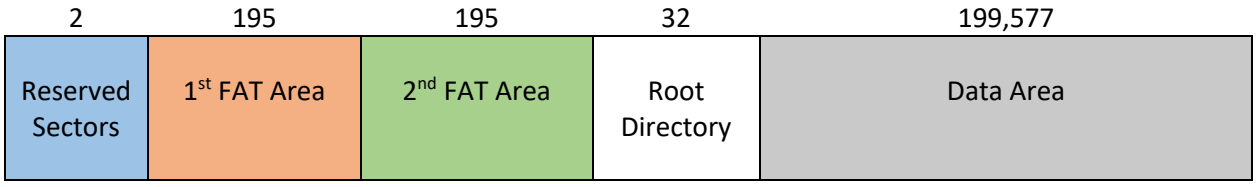
Figure 18 - Disk Editor



Offset	00 01 02 03 04 05 06 07	08 09 0A 0B 0C 0D 0E 0F	ASCII	Unicode
00000000	EB 3C 90 4D 53 44 4F 53	35 2E 30 00 02 04 02 00	ë<.MSDOS5.0.....	.....0Ђ.
00000010	02 00 02 00 00 F8 C3 00	3F 00 FF 00 41 15 03 00	.....øÃ.?.ÿ.A...	...Ã?ÿ..
00000020	41 0D 03 00 80 01 29 00	68 C9 0E 4E 4F 20 4E 41	A...€.).hÉ.NO NA	..Ђ)....
00000030	4D 45 20 20 20 20 46 41	54 31 36 20 20 20 33 C9	ME FAT16 3É	.††.‶†.
00000040	8E D1 BC F0 7B 8E D9 B8	00 20 8E C0 FC BD 00 7C	.Ñ¼ð{.Ù¸. .Àü½.|	....
00000050	38 4E 24 7D 24 8B C1 99	E8 3C 01 72 1C 83 EB 3A	8N$}$.Á.è<.r..ë:	....
00000060	66 A1 1C 7C 26 66 3B 07	26 8A 57 FC 75 06 80 CA	f¡.|&f;.&.Wüu..Ê	....
00000070	02 88 56 02 80 C3 10 73	EB 33 C9 8A 46 10 98 F7	.^V.€Ã.së3É.F..÷	.ą......
00000080	66 16 03 46 1C 13 56 1E	03 46 0E 13 D1 8B 76 11	f..F..V..F..Ñ.v.	...Ṗ....
00000090	60 89 46 FC 89 56 FE B8	20 00 F7 E6 8B 5E 0B 03	`.Fü.Vþ¸ .÷æ.^..	.... ..″
000000A0	C3 48 F7 F3 01 46 FC 11	4E FE 61 BF 00 00 E8 E6	ÃH÷ó.Fü.Nþa¿..èæ	....
000000B0	00 72 39 26 38 2D 74 17	60 B1 0B BE A1 7D F3 A6	.r9&8-t.`±.¾¡}ó¦	.®.....
000000C0	61 74 32 4E 74 09 83 C7	20 3B FB 72 E6 EB DC A0	at2Nt .Ç ;ûræëÜ	....
000000D0	FB 7D B4 7D 8B F0 AC 98	40 74 0C 48 74 13 B4 0E	û}´}.ð¬.@t.Ht.´.	.....ᵒ
000000E0	BB 07 00 CD 10 EB EF A0	FD 7D EB E6 A0 FC 7D EB	»..Í.ëï ý}ëæ ü}ë	....
000000F0	E1 CD 16 CD 19 26 8B 55	1A 52 B0 01 BB 00 00 E8	áÍ.Í.&.U.R°.»..è	..ᵃ..ư».
00000100	3B 00 72 E8 5B 8A 56 24	BE 0B 7C 8B FC C7 46 F0	;.rè[.V$¾.|.üÇFð	;.......
00000110	3D 7D C7 46 F4 29 7D 8C	D9 89 4E F2 89 4E F6 C6	=}ÇFô)}.Ù.Nò.NöÆ	....
00000120	06 96 7D CB EA 03 00 00	20 0F B6 C8 66 8B 46 F8	..}Ëê... .¶Èf.Fø	....
00000130	66 03 46 1C 66 8B D0 66	C1 EA 10 EB 5E 0F B6 C8	f.F.f.ÐfÁê.ë^.¶È	....
00000140	4A 4A 8A 46 0D 32 E4 F7	E2 03 46 FC 13 56 FE EB	JJ.F.2ä÷â.Fü.Vþë	....
00000150	4A 52 50 06 53 6A 01 6A	10 91 8B 46 18 96 92 33	JRP.Sj.j...F...3	.¸.....
00000160	D2 F7 F6 91 F7 F6 42 87	CA F7 76 1A 8A F2 8A E8	Ò÷ö.÷öB.Ê÷v..ò.è	....
00000170	C0 CC 02 0A CC B8 01 02	80 7E 02 0E 75 04 B4 42	ÀÌ..Ì¸...€~..u.´B	...ȁ....
00000180	8B F4 8A 56 24 CD 13 61	61 72 0B 40 75 01 42 03	.ô.V$Í.aar.@u.B.	......ŵ.
00000190	5E 0B 49 75 06 F8 C3 41	BB 00 00 60 66 6A 00 EB	^.Iu.øÃA»..`fj.ë	....»...
000001A0	B0 42 4F 4F 54 4D 47 52	20 20 20 20 0D 0A 52 65	°BOOTMGR ..Re	....††..
000001B0	6D 6F 76 65 20 64 69 73	6B 73 20 6F 72 20 6F 74	move disks or ot	....
000001C0	68 65 72 20 6D 65 64 69	61 2E FF 0D 0A 44 69 73	her media.ÿ..Dis	....
000001D0	6B 20 65 72 72 6F 72 FF	0D 0A 50 72 65 73 73 20	k errorÿ..Press	....
000001E0	61 6E 79 20 6B 65 79 20	74 6F 20 72 65 73 74 61	any key to resta	.⁹.⁹....
000001F0	72 74 0D 0A 00 00 00 00	00 00 00 AC CB D8 55 AA	rt.........¬ËØUª	....

2. FAT Structure

Figure 19 - FAT Structure



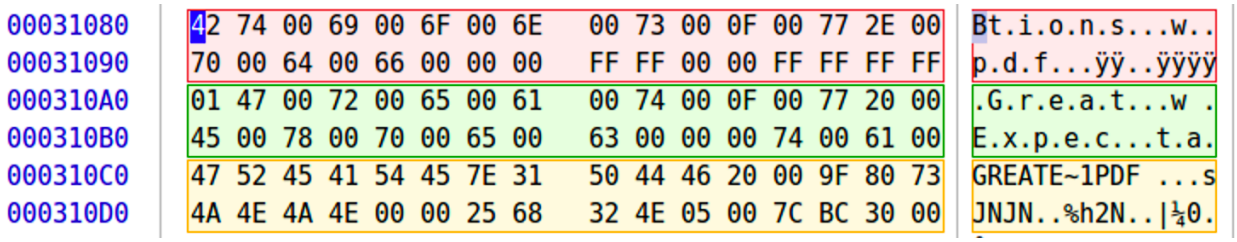
3. File Allocation Table
    a. Located from sector 2 – 196
    b. Bytes 1024 – 100,352
    c. Address 0x400 – 0x18800

Cluster	Next Cluster
0x0000	0xFFF8
0x0001	0xFFFF
0x0002	0xFFFF
0x0003	0xFFFF
0x0004	0xFFFF
0x0005	0x0006
0x0006	0x0007
0x0007	0x0008
0x0009	0x000A
…	
0x061B	0x061C
0x061C	0xFFFF
0x0000	0x000
…	
0x000	0x000

As you can see this FAT table is slightly different than the one earlier. This one only has one linked list while the other had four separated by three lists of zeros. This table however has one list of allocated clusters and a large list of zeros. What this means is that there is one allocated file, and the rest of it is unallocated space. There could be one massive deleted file in that list of zeros or multiple deleted files, or no deleted files at all. All we know at this point is there is one large allocated file, and a large set of unallocated clusters.

4. Root Directory Structure
    a. Sector 392 – 423
    b. Byte 200,704 – 216,576
    c. Address 0x31000 – 0x34E00
5. Root Directory Entries
    a. Great Expectations
        i. Status: 0x42
        ii. Filename: Great Expectations
        iii. Extension: PDF
        iv. Attribute: 0x20 – Archive
        v. Modified date/time: 2/10/19 2:28 PM
        vi. Cluster: 0x0005 – 5
        vii. File Size: 0x0000280C – 3,193,980 bytes

Figure 20 - Great Expectations



    b. Pride and Prejudice
        i. Status: 0xE5 – Deleted

ii. Filename: Pride and Prejudice
iii. Extension: PDF
iv. Attribute: 0x20 – Archive
v. Modified date/time: 1/18/19 1:01 PM
vi. Cluster: 0x061D – 1,565
vii. File Size: 0x000C303B – 798,779 bytes

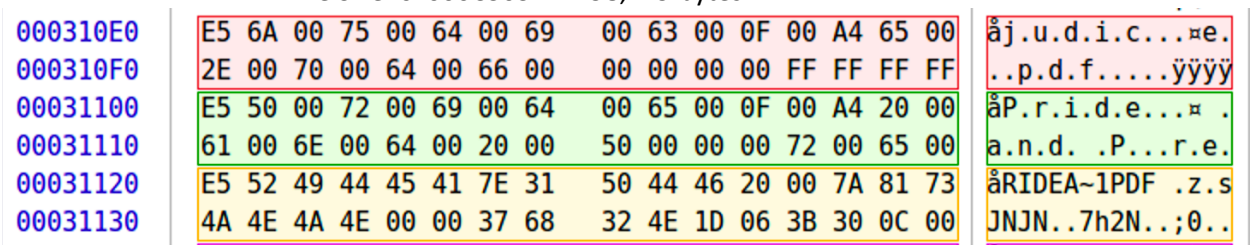


Figure 21 - Pride and Predjudice

c. War and Peace
i. Status: 0xE5 – Deleted
ii. Filename: War and Peace
iii. Extension: PDF
iv. Attribute: 0x20 – Archive
v. Modified date/time: 2/10/19 11:15 AM
vi. Cluster: 0x07A4 – 1956
vii. File Size: 0x009BC0F7 – 10,207,479

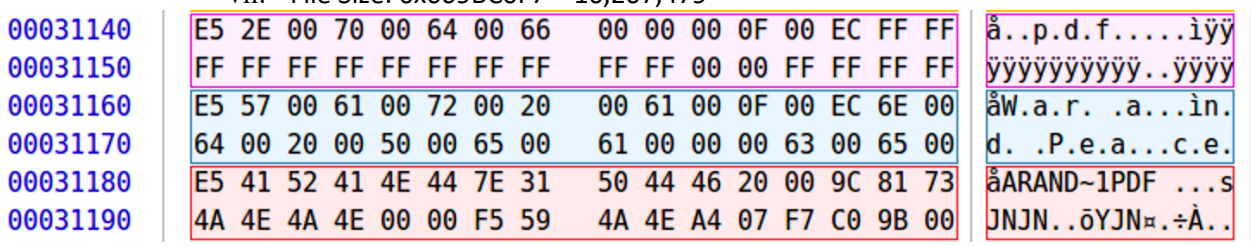


Figure 22 - War and Peace

d. Tale of Two Cities
i. Status: 0xE5 – Deleted
ii. Filename: Tale of Two Cities
iii. Extensions: PDF
iv. Attribute: 0x20 – Archive
v. Modified date/time: 1/18/19 12:58 PM
vi. Cluster: 0x1B1D – 6,941
vii. File Size: 1,316,140 bytes

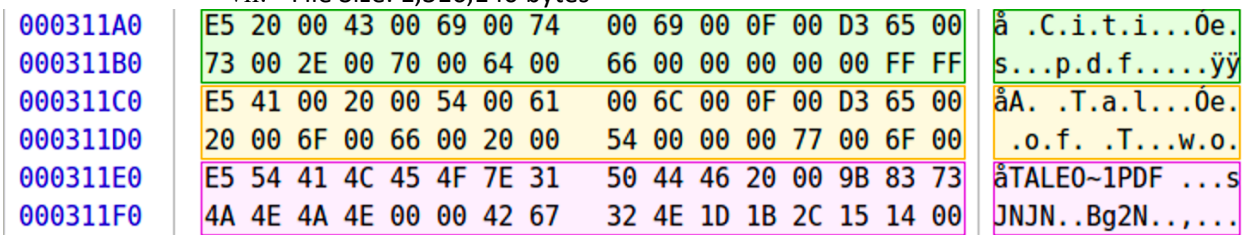


Figure 23 - Tale of Two Cities

6. File Recovery

Table 5 - FAT Partition 2 Recovery

Filename	Cluster	File Length	Cluster Offset	File Offset	Actual Offset	Command To Retrieve File
Great Expectations.pdf	5	3,193,980	6144	223,232	103,672,320	sudo dd if=fat.dd of= 'Great Expectations.pdf' bs=1 skip=103672320 count=3193980 status=progress
Pride and Prejudice.pdf	1,565	798,779	3201024	3,418,112	106,867,200	sudo dd if=fat.dd of= 'Pride and Prejudice.pdf ' bs=1 skip=106867200 count=798779 status=progress
War and Peace.pdf	1,956	10,207,479	4001792	4,218,880	107,667,968	sudo dd if=fat.dd of=' War and Peace.pdf' bs=1 skip=107667968 count=10207479 status=progress
Tale of Two Cities.pdf	6,941	1,316,140	14211072	14,428,160	117,877,248	sudo dd if=fat.dd of=' Tale of Two Cities.pdf' bs=1 skip=117877248 count=1316140 status=progress

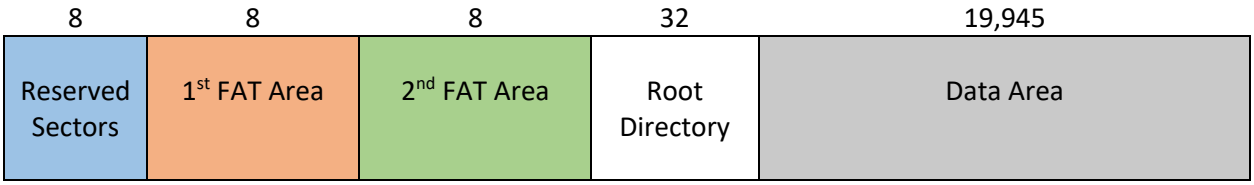
3.3 GIFS

GIFS is a FAT partition similar to SEC PICS and CLASSICS. However, GIFS is a FAT12 partition. Which does not change much other than the max file size and how it organizes it's clusters in the FAT area.

1. Structure Size Calculations
    a. Partition Offset
        i. 0xC450400 = **205,849,600** bytes
    b. Bytes / Sector
        i. Offset 0x0B = 0x0200 = **512**
    c. Sectors / Cluster
        i. Offset 0x0D = 0x8 = **8**
    d. Bytes / Cluster
        i. 8 Sectors / Cluster * 512 Bytes / Sector = **4096**
    e. Reserved Sectors
        i. Offset 0x000E = 0x0008= **8** sectors
    f. Number of FATs
        i. Offset 0x0010 = 0x2 = **2**

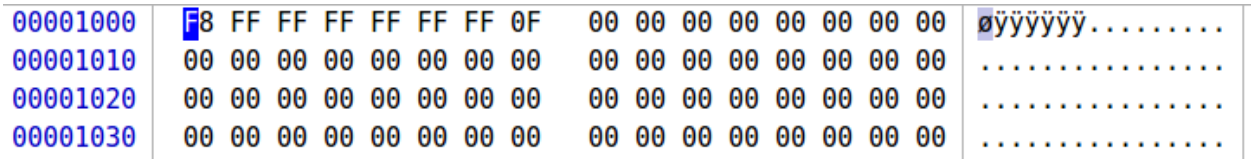
g. 1st FAT Area Size
    i. Offset 0x0016 = 0x8 = **8** sectors
h. 2nd FAT Area Size
    i. Offset 0x0016 = 0x8 = **8** sectors
i. Root Directory
    i. Directory Entries
        1. Offset 0x0011 = 0x0200 = 512 entries
    ii. 512 * 32 bytes = 16,384 bytes / 512 (bytes/sector) = **32** sectors
j. Data Area
    i. Total Sectors
        1. Offset 0x0013= 0x4E21 = 20,001 Sectors
    ii. Data Area = Total Sectors – End of Root
        1. 20,001 – 56 = **19,945** Sectors

2. FAT Structure



3. File Analysis
    a. File Allocation Table
        i. Located from sector 8 – 15
        ii. Bytes 4096 – 7,680
        iii. Address 0x1000 – 0x1E00

Note that for FAT12 each FAT entry is 12 bits. Therefore, the two special entries for the FAT table below are 0xFF8 and 0xFFF. After that there are three system files allocated (0xFFF, 0xFFF, 0xFFF). However, after those there are just zeros, which would indicate that there are no active user files in this partition. However, there still could be recoverable deleted files.



```
00001000  F8 FF FF FF FF FF FF 0F   00 00 00 00 00 00 00 00  øÿÿÿÿÿÿ.........
00001010  00 00 00 00 00 00 00 00   00 00 00 00 00 00 00 00  ................
00001020  00 00 00 00 00 00 00 00   00 00 00 00 00 00 00 00  ................
00001030  00 00 00 00 00 00 00 00   00 00 00 00 00 00 00 00  ................
```

Cluster	Next Cluster
0x000	0xFF8
0x001	0xFFF
0x002	0xFFF
0x003	0xFFF
0x004	0xFFF
0x005	0xFFF

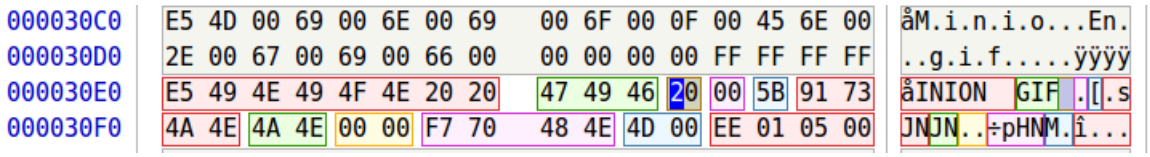
4. Root Directory Structure
    a. Sector 24 – 55
    b. Byte 12,288 – 28,160
    c. Address 0x3000 – 0x6E00

```
00003000  47 49 46 53 20 20 20 20   20 20 20 08 00 00 00 00  GIFS       .....
00003010  00 00 00 00 00 00 42 73   4A 4E 00 00 00 00 00 00  ......BsJN......
00003020  42 20 00 49 00 6E 00 66   00 6F 00 0F 00 72 72 00  B .I.n.f.o...rr.
00003030  6D 00 61 00 74 00 69 00   6F 00 00 00 6E 00 00 00  m.a.t.i.o...n...
00003040  01 53 00 79 00 73 00 74   00 65 00 0F 00 72 6D 00  .S.y.s.t.e...rm.
00003050  20 00 56 00 6F 00 6C 00   75 00 00 00 6D 00 65 00   .V.o.l.u...m.e.
00003060  53 59 53 54 45 4D 7E 31   20 20 20 16 00 50 41 73  SYSTEM~1   ..PAs
00003070  4A 4E 4A 4E 00 00 42 73   4A 4E 02 00 00 00 00 00  JNJN..BsJN......
00003080  E5 42 00 61 00 6E 00 61   00 6E 00 0F 00 05 61 00  åB.a.n.a.n....a.
00003090  2E 00 67 00 69 00 66 00   00 00 00 00 FF FF FF FF  ..g.i.f.....ÿÿÿÿ
000030A0  E5 41 4E 41 4E 41 20 20   47 49 46 20 00 2A 91 73  åANANA  GIF .*.s
000030B0  4A 4E 4A 4E 00 00 C0 59   4A 4E 05 00 3F 73 04 00  JNJN..ÀYJN..?s..
000030C0  E5 4D 00 69 00 6E 00 69   00 6F 00 0F 00 45 6E 00  åM.i.n.i.o...En.
000030D0  2E 00 67 00 69 00 66 00   00 00 00 00 FF FF FF FF  ..g.i.f.....ÿÿÿÿ
000030E0  E5 49 4E 49 4F 4E 20 20   47 49 46 20 00 5B 91 73  åINION  GIF .[.s
000030F0  4A 4E 4A 4E 00 00 F7 70   48 4E 4D 00 EE 01 05 00  JNJN..÷pHNM.î...
```

5. Root Directory Entries
    a. Banana
        i. Status: 0xE5 – Deleted
        ii. Filename: Banana
        iii. Extension: GIF
        iv. Attribute: 0x20 – Archive
        v. Modified date/time: 2/10/19 11:14 AM
        vi. Cluster: 0x0005 – 5
        vii. File Size: 0x0000280C – 291,647 bytes

```
00003080  E5 42 00 61 00 6E 00 61   00 6E 00 0F 00 05 61 00  åB.a.n.a.n....a.
00003090  2E 00 67 00 69 00 66 00   00 00 00 00 FF FF FF FF  ..g.i.f.....ÿÿÿÿ
000030A0  E5 41 4E 41 4E 41 20 20   47 49 46 20 00 2A 91 73  åANANA  GIF .*.s
000030B0  4A 4E 4A 4E 00 00 C0 59   4A 4E 05 00 3F 73 04 00  JNJN..ÀYJN..?s..
```

    b. Minions
        i. Status: 0xE5 – Deleted
        ii. Filename: Minion
        iii. Extension: GIF
        iv. Attribute: 0x20 – Archive
        v. Modified date/time: 2/8/19 2:07 PM
        vi. Cluster: 0x007A – 77
        vii. File Size: 0x0000280C – 328,174 bytes



6. File Recovery

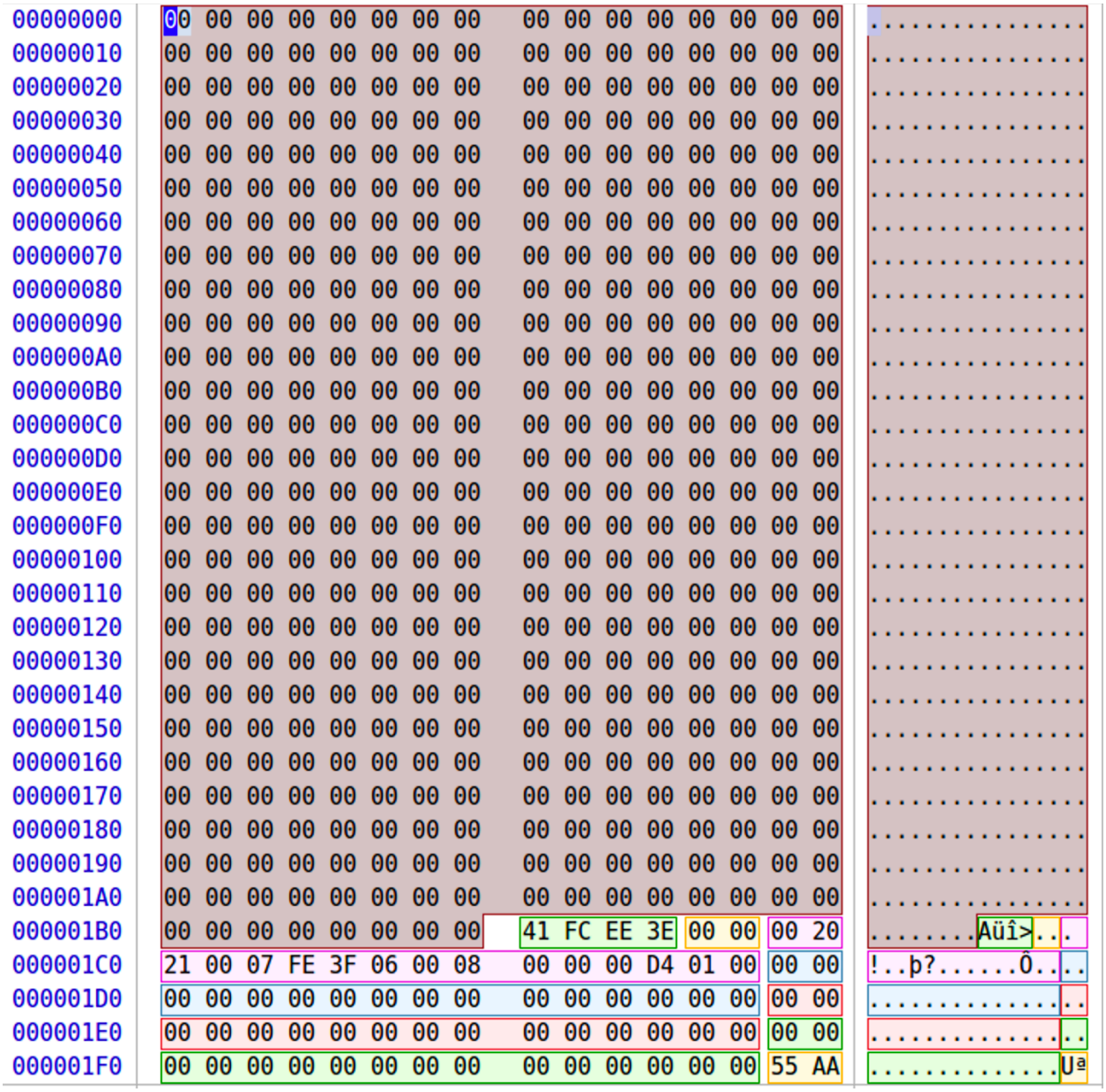
Table 6 - FAT Partition 3 File Recovery

Filename	Cluster	File Length	Cluster Offset	File Offset	Actual Offset	Command To Retrieve File
Banana.GIF	5	291647	12288	40960	205890560	sudo dd if=fat.dd of='Banana.GIF' bs=1 skip=205890560 count=291647 status=progress
Minions.GIF	77	328174	307200	335872	206185472	sudo dd if=fat.dd of='Minions.GIF' bs=1 skip=206185472 count=328174 status=progress

3 NTFS Image

Using the same method, we used on the FAT volume we can find the partition sector offset and what partition type it is.

1. First Sector: Offset 0x01C6 = 0x0800 = 2048
2. First Sector Address: 512 * 2048 = 1,048,576 = 0x100000
3. Partition ID: Offset 0x01C2 = 0x07 = 7 = NTFS



We now know the NTFS partition offset. We will use it later for recovering the files. However, we will not use the raw image in disk editor, due to the large offsets, we will use the partition view instead. Here we can grab the following information:

a. Bytes/Sector: Offset 0x0B = 0x0200 = 512
b. Sectors/Cluster: Offset 0x0D = 0x08 = 8
c. Bytes/Cluster: 8 * 512 = 4096 bytes / cluster
d. $MFT Cluster Number: Offset 0x30 = 0x1380 = 4,992
e. $MFT Address: 0x13800000 = 20,447,232

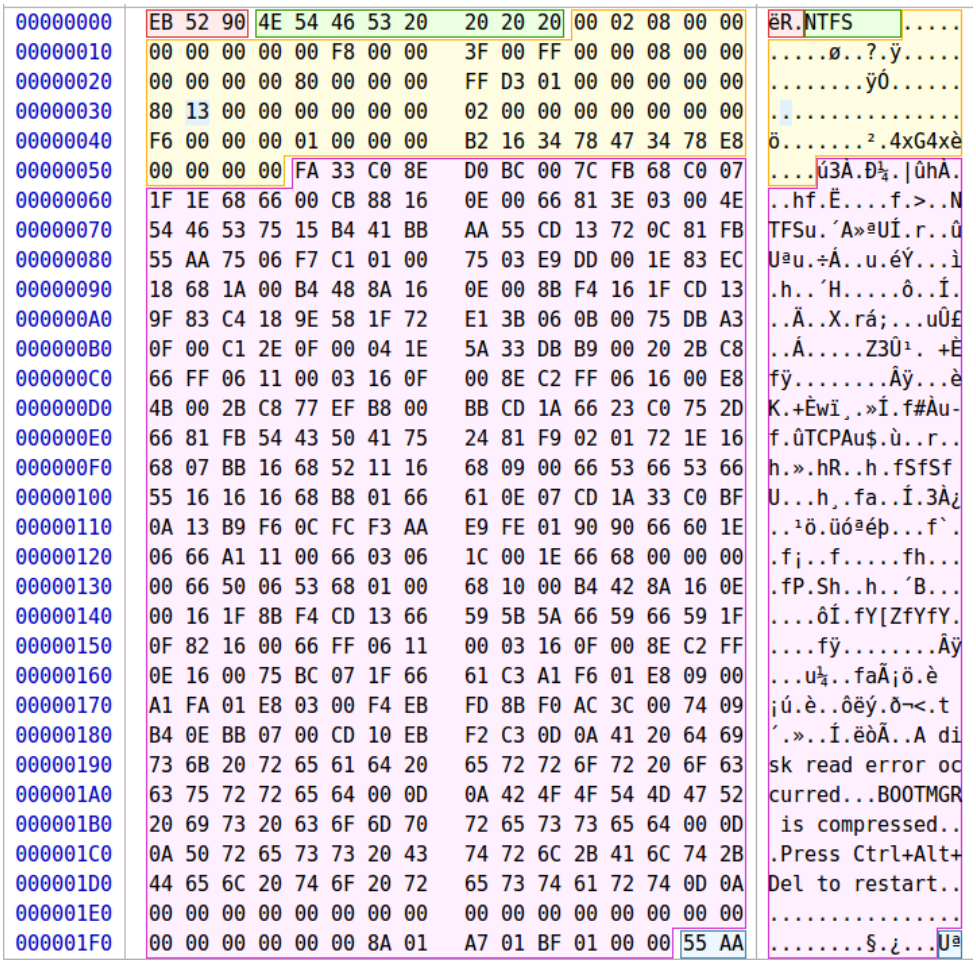
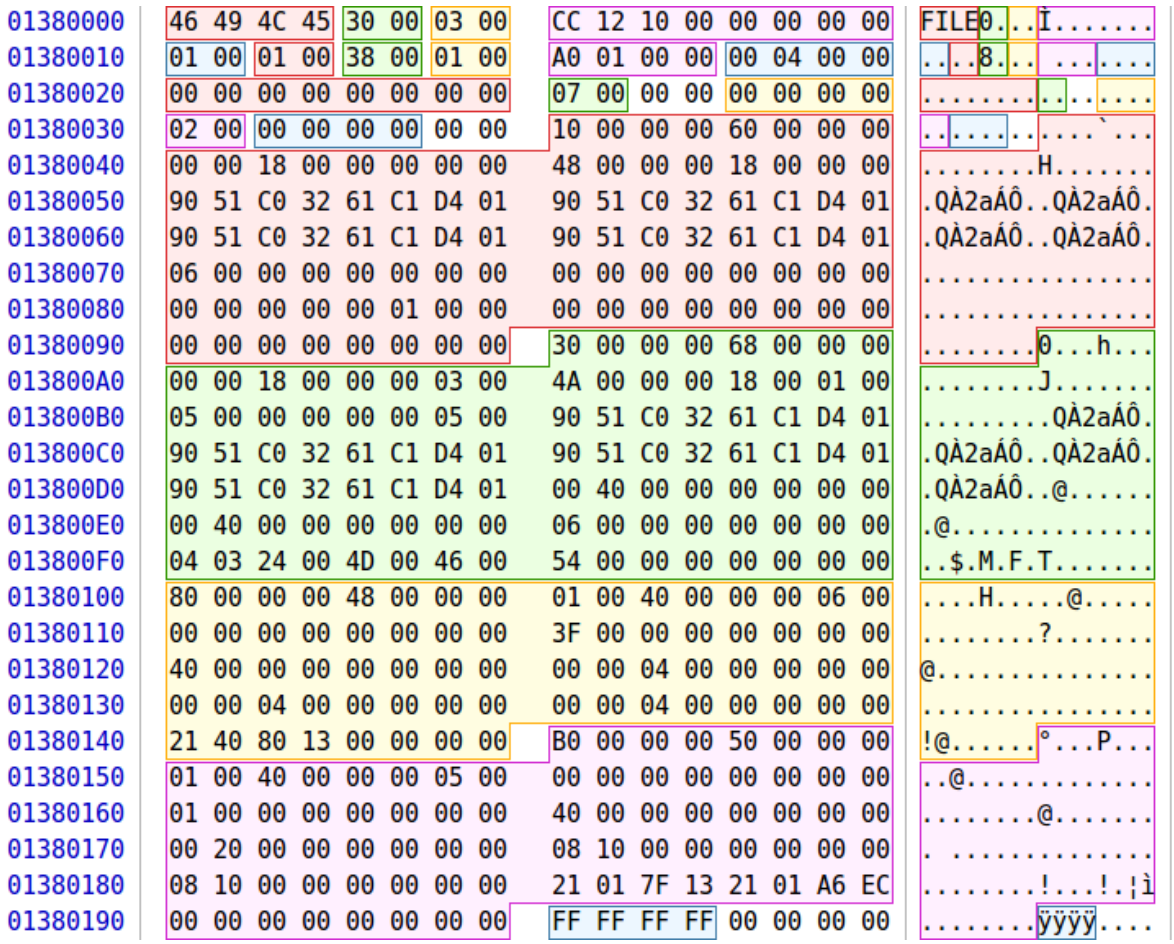


Figure 24 - NTFS MBR using FTK Imager

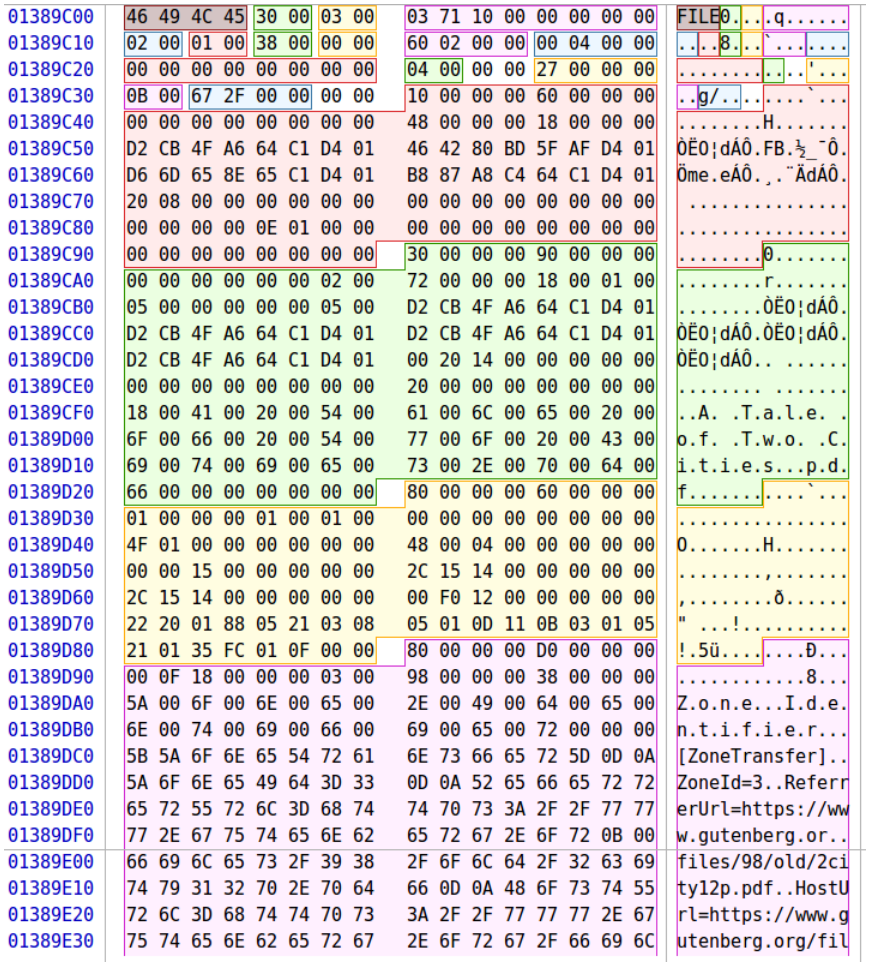
Now that we know where the Master File Table is located, we can find the number of system files, the system files length, and the start of the user files. To find the start of the user files, we simply add the $MFT Address to the system files length. We know that there are normally thirty-nine system files, so if we calculate the number of bytes for those system files we get that the system files take up 39,936 bytes (0x9C00) since each MFT entry is 1024 bytes. Adding that to the $MFT offset, 20,447,232 (0x1380000) we get that the user files start at **20,487,168 (0x1389C00)**. Each file entry should have at least the following file attributes

- $STANDARD_INFORMATION (0x10)
- $FILE_NAME (0x30)
- $DATA (0x80)



Jumping to offset 0x1389C00 we get the first user file.

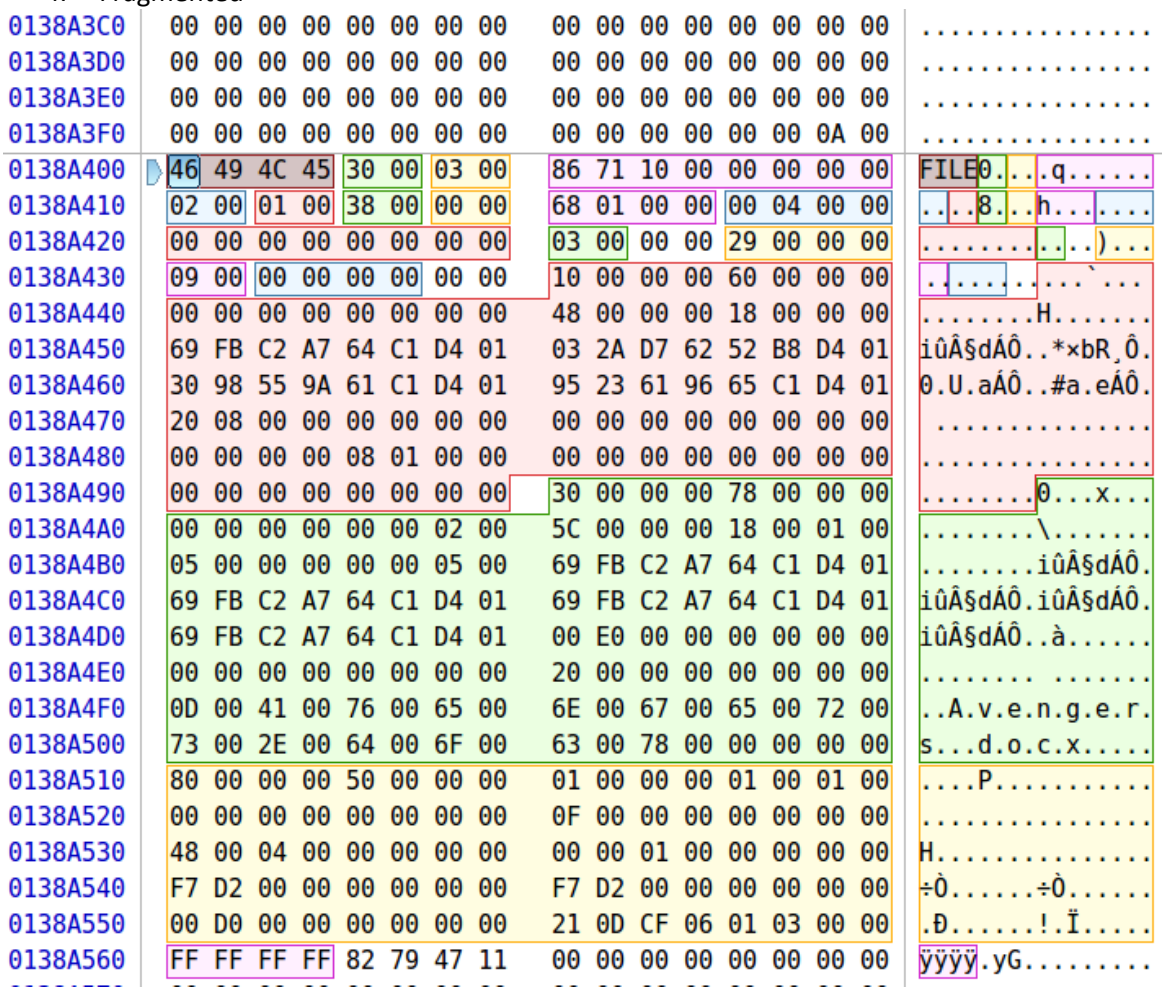
1. A Tale of Two Cities
    a. List File Attributes:
        i. 0x10 - $STANDARD_INFORMATION
        ii. 0x30 - $FILE_NAME
        iii. 0x80 - $DATA
        iv. 0xB0 - $BITMAP
    b. Filename: Offset 0x0F2 = A Tale of Two Cities.pdf
    c. Allocated Size: Offset 0x150= 0x150000 = 1,376,256 bytes
    d. Deleted
    e. Data Start Cluster: Offset 0x170 = 0x0588 = 1,416
        i. Data Start Address
            1. 1,416 * 4,096 = 5,799,936
        ii. Actual Offset
            1. Partition Offset + Start Address
            2. 1,048,576 + 5,799,936= 6,848,512
    f. However, this file is fragmented, as seen by the multiple starting clusters. It cannot be recovered using the standard recovery method that we have been using
    g. Fragmented



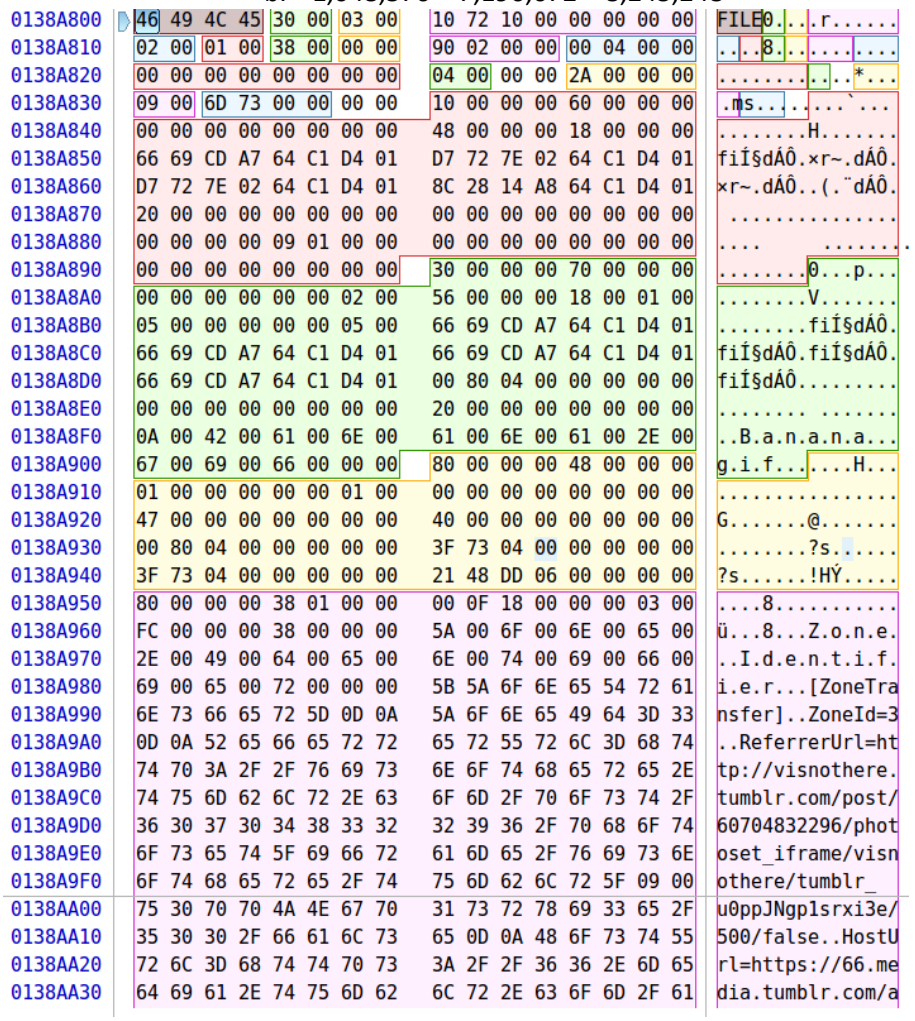
2. Auburn
    a. List File Attributes:
        i. 0x10 - $STANDARD_INFORMATION
        ii. 0x30 - $FILE_NAME
        iii. 0x80 - $DATA
    b. Filename: Offset 0x0F2 = Auburn.jpg
    c. Active
    d. Allocated Size: Offset 0x130 = 0x4000 = 16,384 bytes
    e. Data Cluster Information
        i. 21 04 CB 06
            1. 2 bytes required for the first cluster address
            2. Number of continuous clusters for this file – 1 * 4 = 4
            3. Start Cluster for Data: 0x06CB = 1,739
            4. Start Address for Data:
                a. 1,739 * 4096 = 7,122,944
            5. Actual Offset
                a. Partition Offset + Start Address
                b. 1,048,576 + 7,122,944 = 8,171,520

```
0138A000  46 49 4C 45 30 00 03 00  CB 54 10 00 00 00 00 00  FILE0...ËT......
0138A010  01 00 01 00 38 00 01 00  58 01 00 00 00 04 00 00  ....8...X.......
0138A020  00 00 00 00 00 00 00 00  03 00 00 00 28 00 00 00  ............(...
0138A030  0A 00 00 00 00 00 00 00  10 00 00 00 60 00 00 00  ............`...
0138A040  00 00 00 00 00 00 00 00  48 00 00 00 18 00 00 00  ........H.......
0138A050  F9 29 A4 A7 64 C1 D4 01  92 32 72 0E A6 BE D4 01  ù)¤§dÁÔ..2r.¦¾Ô.
0138A060  BD 9F 44 94 65 C1 D4 01  CB C7 4A DC 65 C1 D4 01  ½.D.eÁÔ.ËÇJÜeÁÔ.
0138A070  22 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  "...............
0138A080  00 00 00 00 08 01 00 00  00 00 00 00 00 00 00 00  ................
0138A090  00 00 00 00 00 00 00 00  30 00 00 00 70 00 00 00  ........0...p...
0138A0A0  00 00 00 00 00 00 02 00  56 00 00 00 18 00 01 00  ........V.......
0138A0B0  05 00 00 00 00 00 05 00  F9 29 A4 A7 64 C1 D4 01  ........ù)¤§dÁÔ.
0138A0C0  F9 29 A4 A7 64 C1 D4 01  F9 29 A4 A7 64 C1 D4 01  ù)¤§dÁÔ.ù)¤§dÁÔ.
0138A0D0  F9 29 A4 A7 64 C1 D4 01  00 40 00 00 00 00 00 00  ù)¤§dÁÔ..@......
0138A0E0  00 00 00 00 00 00 00 00  20 00 00 00 00 00 00 00  ........ .......
0138A0F0  0A 00 41 00 75 00 62 00  75 00 72 00 6E 00 2E 00  ..A.u.b.u.r.n...
0138A100  4A 00 50 00 47 00 00 00  80 00 00 00 48 00 00 00  J.P.G.......H...
0138A110  01 00 00 00 00 00 01 00  00 00 00 00 00 00 00 00  ................
0138A120  03 00 00 00 00 00 00 00  40 00 00 00 00 00 00 00  ........@.......
0138A130  00 40 00 00 00 00 00 00  55 30 00 00 00 00 00 00  .@......U0......
0138A140  55 30 00 00 00 00 00 00  21 04 CB 06 00 00 00 00  U0......!.Ë.....
0138A150  FF FF FF FF 82 79 47 11  00 00 00 00 00 00 00 00  ÿÿÿÿ.yG.........
```

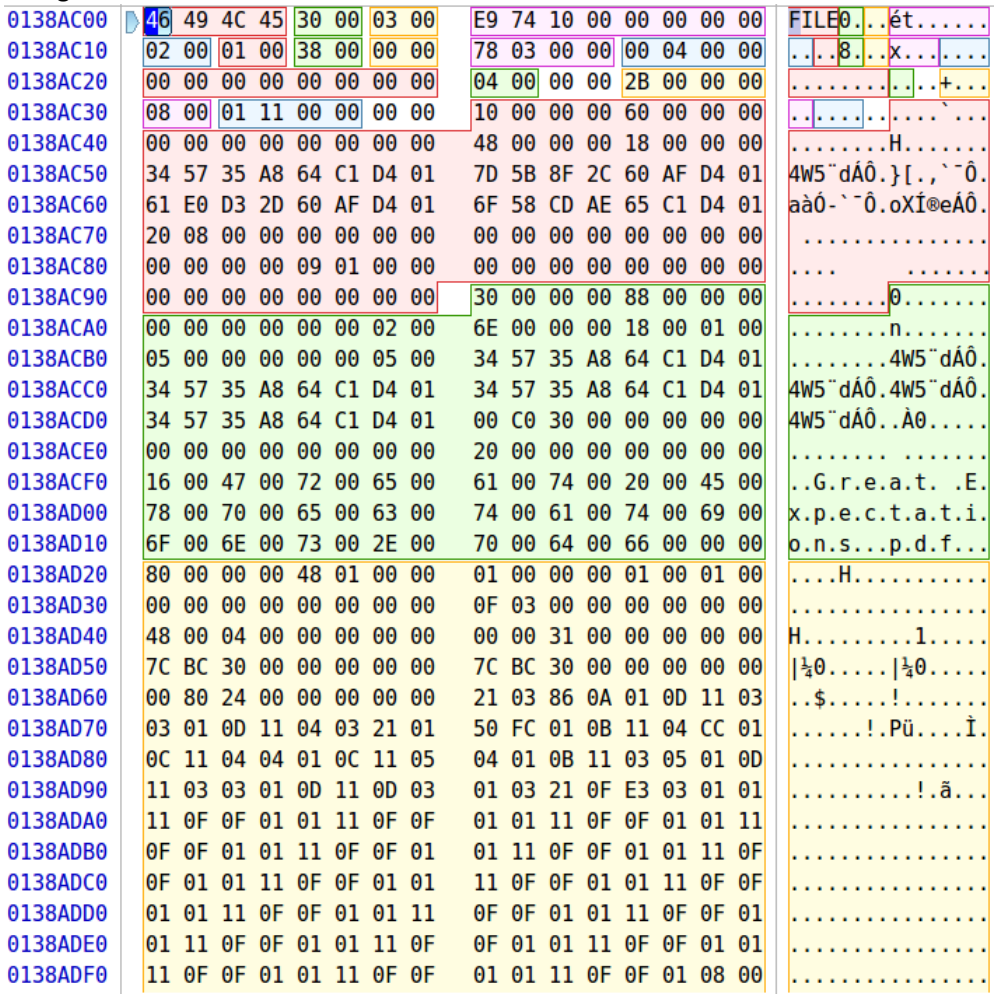
3. Avengers – Offset 0x138A3C0 (20,489,216)
    a. List File Attributes:
        i. 0x10 - $STANDARD_INFORMATION
        ii. 0x30 - $FILE_NAME
        iii. 0x80 - $DATA
    b. Filename: Offset 0x0F2 = Avengers.docx
    c. Deleted
    d. Allocated Size: Offset 0x138 = 0x100000 = 65,536 bytes
    e. Data Cluster Information
        i. 21 0D CF 06
            1. 2 bytes required for the first cluster address
            2. Number of continuous clusters for this file – 1 * D = 13
            3. Start Cluster for Data: 0x06CF = 1,743
            4. Start Address for Data:
                a. 1,739 * 4096 = 7,139,328
            5. Actual Offset
                a. Partition Offset + Start Address
                b. 1,048,576 + 7,139,328 = 8,187,904
    f. Fragmented



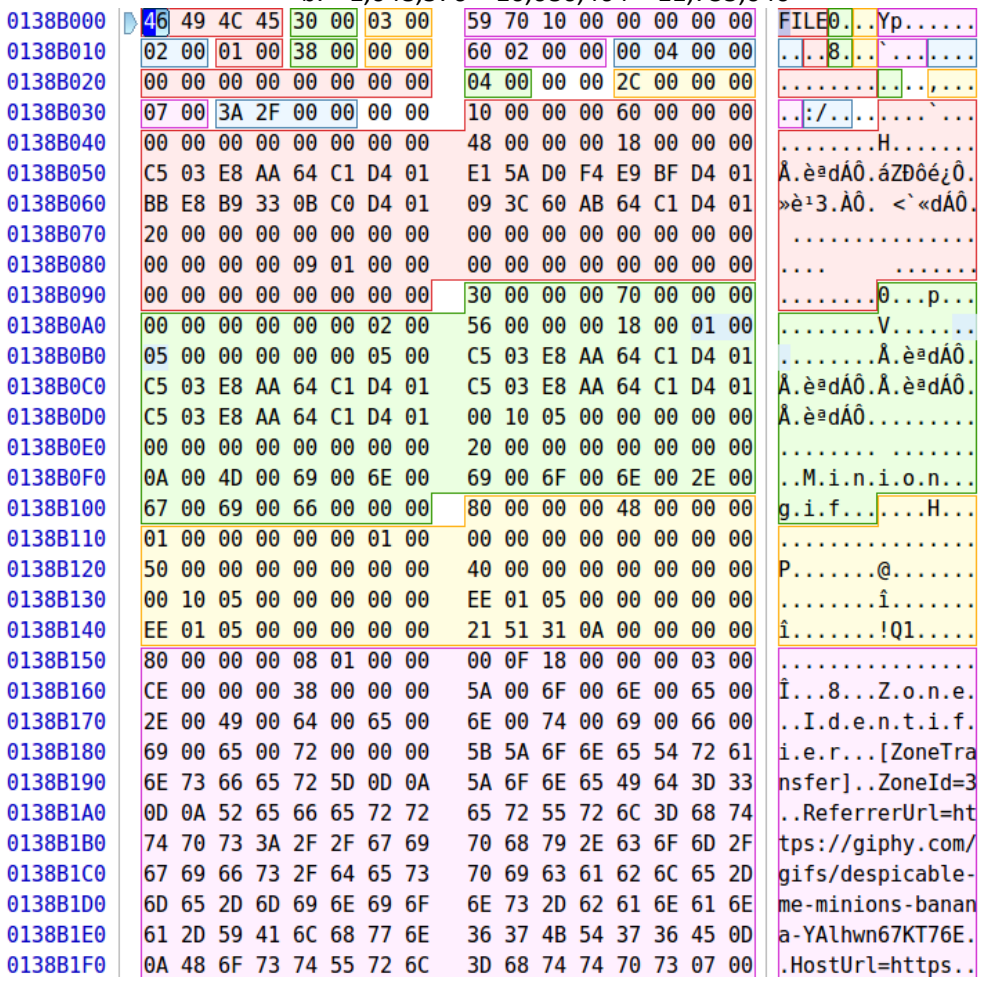
4. Banana – Offset 0x138A800 (20,490,240)
    a. List File Attributes:
        i. 0x10 - $STANDARD_INFORMATION
        ii. 0x30 - $FILE_NAME
        iii. 0x80 - $DATA
        iv. 0xB0 - $BITMAP
    b. Filename: Offset 0x0F2 = Banana.gif
    c. Deleted
    d. Allocated Size: Offset 0x130 = 0x048000= 294,912 bytes
    e. Data Cluster Information
        i. 21 48 DD 06
            1. 2 bytes required for the first cluster address
            2. Number of continuous clusters for this file – 0x1 * 0x48 = 72
            3. Start Cluster for Data: 0x06DD = 1,757
            4. Start Address for Data
                a. 1,757 * 4096 = 7,196,672
            5. Actual Offset
                a. Partition Offset + Start Address
                b. 1,048,576 + 7,196,672 = 8,245,248



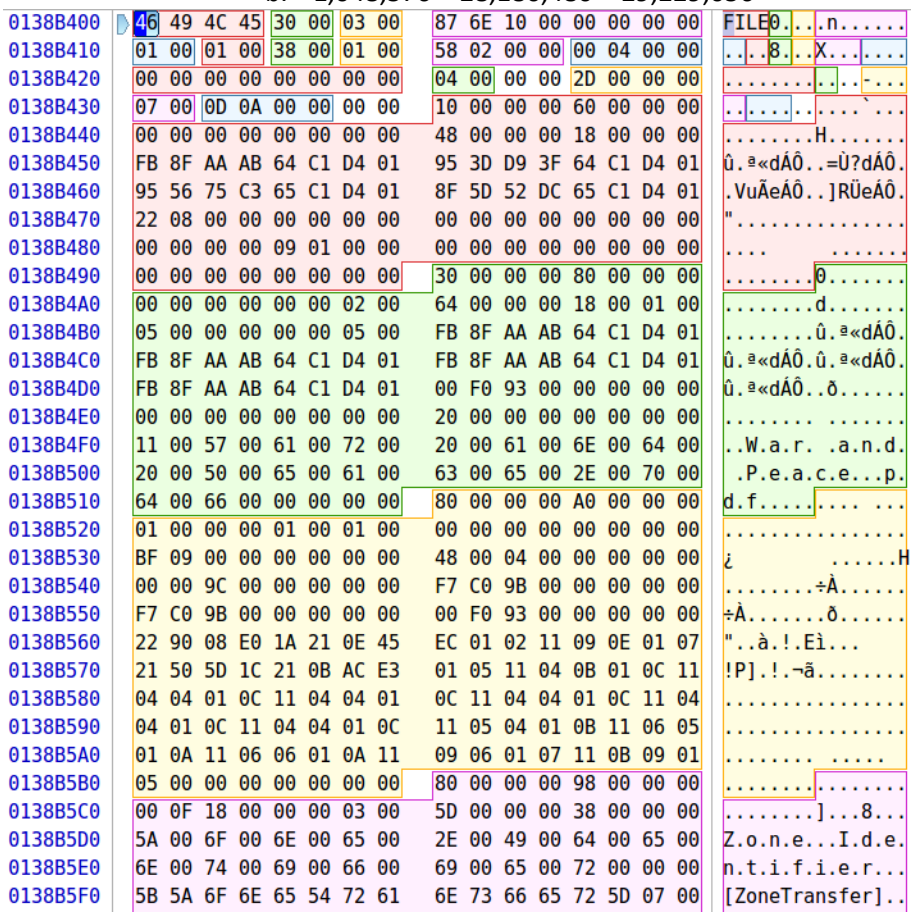
4. Great Expectations – Offset 0x138AC00 (20,491,264)
    a. List File Attributes:
        i. 0x10 - $STANDARD_INFORMATION
        ii. 0x30 - $FILE_NAME
        iii. 0x80 - $DATA
        iv. 0xB0 - $BITMAP
    b. Filename: Offset 0x0F2 = Great Expectations.pdf
    c. Deleted
    d. Allocated Size: Offset 0x148 = 0x310000= 3,211,264 bytes
    e. Data Cluster Information
        i. 21 03 86 0A
            1. 2 bytes required for the first cluster address
            2. Number of continuous clusters for this file – 0x1 * 0x3 = 3
            3. Start Cluster for Data: 0x0A86 = 2,694
            4. Start Address for Data:
                a. 2,694 * 4096 = 11,034,624
            5. Actual Offset
                a. Partition Offset + Start Address
                b. 1,048,576 + 11,034,624 = 12,083,200
    f. Fragmented



5. Minion – Offset 0x138B000 (20,492,288)
    a. List File Attributes:
        i. 0x10 - $STANDARD_INFORMATION
        ii. 0x30 - $FILE_NAME
        iii. 0x80 - $DATA
        iv. 0x80 - $BITMAP
    b. Filename: Offset 0x0F2 = Minion.gif
    c. Allocated Size: Offset 0x130 = 0x051000 = 331,776 bytes
    d. Deleted
    e. Data Cluster Information
        i. 21 05 31 0A
            1. 2 bytes required for the first cluster address
            2. Number of continuous clusters for this file – 0x1 * 0x5 = 5
            3. Start Cluster for Data: 0x0A31 = 2,609
            4. Start Address for Data:
                a. 2,609 * 4096 = 10,686,464
            5. Actual Offset
                a. Partition Offset + Start Address
                b. 1,048,576 + 10,686,464 = 11,735,040



6. War and Peace – Offset 0x138B000 (20,492,288)
    a. List File Attributes:
        i. 0x10 - $STANDARD_INFORMATION
        ii. 0x30 - $FILE_NAME
        iii. 0x80 - $DATA
        iv. 0x80 - $BITMAP
    b. Filename: Offset 0x0F2 = War and Peace.pdf
    c. Active
    d. Allocated Size: Offset 0x140 = 0x9C0000 = 10,223,616 bytes
    e. Data Cluster Information
        i. 22 90 08 E0 1A
            1. 2 bytes required for the first cluster address
            2. 2 bytes required for cluster counter
                a. 0x0890 = 2,192
            3. Start Cluster for Data: 0x1AE0 = 6,880
            4. Start Address for Data:
                a. 6,880 * 4096 = 28,180,480
            5. Actual Offset
                a. Partition Offset + Start Address
                b. 1,048,576 + 28,180,480 = 29,229,056



Using the information gathered we can make a table similar to the ones we used for the FAT partitions. However, this time clusters are not subtracted by two. Note that some of these files are fragmented and cannot be recovered in the same process we have been using so we will have to use sleuth kit instead.

Filename	Cluster	File Length	Cluster Offset	Actual Offset	Command To Retrieve File	Recoverable with DD
A Tale of Two Cities.pdf	1,416	1,376,256	5,799,936	6,848,512	sudo dd if=fat.dd of='A Tale of Two Cities.pdf' bs=1 skip=6848512 count=1376256 status=progress	
Auburn.jpg	1,739	16,384	7,122,944	8,171,520	sudo dd if=fat.dd of='Auburn.jpg' bs=1 skip=8171520 count=16384 status=progress	X
Avengers.docx	1,743	65,536	7,139,328	8,187,904	sudo dd if=fat.dd of='Avengers.docx' bs=1 skip=8187904 count=65536 status=progress	
Banana.gif	1,757	294,912	7,196,672	8,245,248	sudo dd if=fat.dd of='Banana.gif' bs=1 skip=8245248 count=294912 status=progress	X
Great Expectations.pdf	2,694	3,211,264	11,034,624	12,083,200	sudo dd if=fat.dd of='Great Expectations.pdf' bs=1 skip=12083200 count=3211264 status=progress	
Minion.gif	2,609	331,776	10,686,464	11,735,040	sudo dd if=fat.dd of='Minion.gif' bs=1 skip=11735040 count=331776 status=progress	X
War and Peace.pdf	6,880	10,223,616	28,180,480	29,229,056	sudo dd if=fat.dd of='War and Peace.pdf' bs=1 skip=29229056 count=10223616 status=progress	X

Table 7 - NTFS File Recovery

```
$ fls -o 2048 ntfs.dd
r/r 4-128-1:    $AttrDef
r/r 8-128-2:    $BadClus
r/r 8-128-1:    $BadClus:$Bad
r/r 6-128-4:    $Bitmap
r/r 7-128-1:    $Boot
d/d 11-144-4:   $Extend
r/r 2-128-1:    $LogFile
r/r 0-128-6:    $MFT
r/r 1-128-1:    $MFTMirr
r/r 9-128-8:    $Secure:$SDS
r/r 9-144-11:   $Secure:$SDH
r/r 9-144-14:   $Secure:$SII
r/r 10-128-1:   $UpCase
r/r 10-128-4:   $UpCase:$Info
r/r 3-128-3:    $Volume
r/r 40-128-1:   Auburn.JPG
d/d 36-144-1:   System Volume Information
r/r 45-128-1:   War and Peace.pdf
r/r 45-128-3:   War and Peace.pdf:Zone.Identifier
-/r * 39-128-1: A Tale of Two Cities.pdf
-/r * 39-128-3: A Tale of Two Cities.pdf:Zone.Identifier
-/r * 41-128-1: Avengers.docx
-/r * 42-128-1: Banana.gif
-/r * 42-128-3: Banana.gif:Zone.Identifier
-/r * 43-128-1: Great Expectations.pdf
-/r * 43-128-3: Great Expectations.pdf:Zone.Identifier
-/r * 44-128-1: Minion.gif
-/r * 44-128-3: Minion.gif:Zone.Identifier
d/d 256:        $OrphanFiles
```

Figure 25 - File Fragmentation Recovery

Using sleuth kit we can recover the files. However, similar to before we will need the inodes of these files first. To gather them we will need to run fls -o 2048 ntfs.dd. With the inodes we can now attempt to recover the files. We can use icat to recover these files.

icat -o 2048 -r ntfs.dd [inode] > [filename]

- icat -o 2048 -r ntfs.dd 39-128-1 > 'A Tale of Two Cities.pdf'
- icat -o 2048 -r ntfs.dd 41-128-1 > 'Avengers.docx'
- icat -o 2048 -r ntfs.dd 43-128-1 > 'Great Expectations.pdf'

5 Conclusion

This project allowed us to delve into a FAT and a NTFS file system. In doing so, we gained a better understanding of how the formats are structured, where data is stored, and how the file systems take care of deleted files.

Number of FAT Partitions: 3

Number of Active and Deleted Files per Partition (FAT):

SEC Pics: 4 Active and 3 Deleted

Classics: 1 Active and 3 Deleted

GIFs: 2 Deleted Gifs

Number of NTFS Image Partitions: 1

Number of NTFS Active and Deleted files: 2 Active and 5 Deleted

6 References

[1] https://en.wikipedia.org/wiki/Design_of_the_FAT_file_system