Digital Forensics

Federico Conti

2024/25

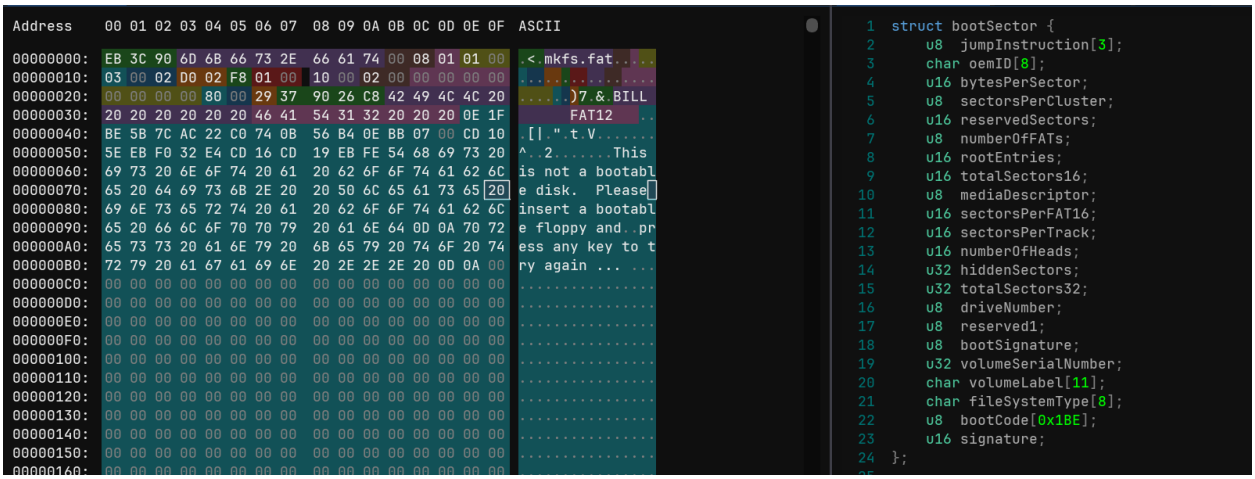
Contents

corrupted.dd

This report details the forensic analysis conducted on the disk image `corrupted.dd`. The objective was to investigate the file system structure, identify signs of corruption, recover inaccessible data, and locate specific string patterns within the image.

Initial Setup

```
file corrupted.dd
# Output:
    DOS/MBR boot sector, code offset 0x3c+2, OEM-ID "mkfs.fat", Bytes/sector 2048, FATs 3, root entries

fsstat corrupted.dd
# Output:
    ...
    File System Type: FAT12
    OEM Name: mkfs.fat
    Volume ID: 0xc8269037
    Volume Label (Boot Sector): BILL
    Volume Label (Root Directory): BILL
    File System Type Label: FAT12
    ...
    File System Layout (in sectors)
    Total Range: 0 - 719
    * Reserved: 0 - 0
    ** Boot Sector: 0
    * FAT 0: 1 - 1
    * FAT 1: 2 - 2
    * FAT 2: 3 - 3
    * Data Area: 4 - 719
    ** Root Directory: 4 - 11
    ** Cluster Area: 12 - 719

    METADATA INFORMATION
    ------------------------------------------
    Range: 2 - 45831
    Root Directory: 2

    CONTENT INFORMATION
    ------------------------------------------
    Sector Size: 2048
    Cluster Size: 2048
```



A sector 0 directly contains a FAT12 Boot Sector, so there is no MBR/GPT table listing partitions, and it is not a bootable image.

- **Volume Label**: `BILL`
- **Sector Size**: 2048
- **Cluster Size**: 2048
- **Num FATs**: 3

Analysis Process

Using The Sleuth Kit (TSK):

```
fls -r -p corrupted.dd | grep '\.TXT'
# Output:
    r/r 45: HOMEWORK.TXT
    r/r 32: NETWORKS.TXT
    r/r * 36:       _EADME.TXT
```

1. **HOMEWORK.TXT** (pseudo-inode 45) > Status: Allocated
   > Size: 6 bytes
   > Sector: 619
   > Readability:
   Can read it both by using `icat` and by mounting the image.

2. **NETWORKS.TXT** (pseudo-inode 32) > Status: Allocated
   > Size: 17,465 bytes
   > Starting Sector: 345
   > Readability:
   Cannot read it by mounting the image, but you can read it using `icat`.
   Explanation: This indicates that the mounted file system has issues following the cluster chain, likely due to corruption in the FAT.

3. **EADME.TXT** (pseudo-inode 36) > Status: Deleted
   > Size: 60,646 bytes
   > Sectors: 457 to 486
   > Readability:
   Since this file is deleted, it is expected not to appear in the mounted file system. However, you can recover it using `icat` if the data has not been overwritten.

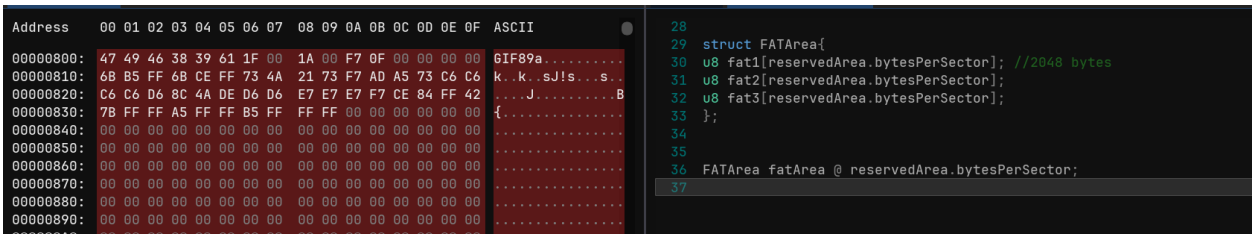
```
istat corrupted.dd 32
# Output:
    Directory Entry: 32 #pseudo inode by TSK
    Allocated
    File Attributes: File, Archive
    Size: 17465
    Name: NETWORKS.TXT
    ...
    Sectors: 345
```

Fix FAT Table

Objective: Rebuild the cluster chain in the FAT for corrupted files, particularly for `NETWORKS.TXT`.

Analyzing the first FAT it was discovered that FAT0 was overwritten with non-FAT data, likely a fragment of a GIF file.

```
xxd -s $((2048)) -l 2048 corrupted.dd | less
```



A known good copy of FAT2 was used to restore FAT0:

```
cp corrupted.dd corrupted_fixed.dd
dd if=corrupted_fixed.dd of=corrupted_fixed.dd bs=2048 skip=3 seek=1 count=1 conv=notrunc
```

After fixing:

- The cluster chain has been rebuilt.
- Both `.TXT` files can now be mounted correctly, allowing the recovery of the hash for `NETWORKS.TXT`.

```
sha256sum *.TXT
9b4a458763b06fefc65ba3d36dd0e1f8b5292e137e3db5dea9b1de67dc361311  HOMEWORK.TXT
e9207be4a1dde2c2f3efa3aeb9942858b6aaa65e82a9d69a8e6a71357eb2d03c  NETWORKS.TXT
```

zxgio

Inside the file `corrupted.dd`, there are some occurrences of the string `zxgio` (without quotes). Below is the analysis:

```
strings -t d corrupted.dd | grep -E zxgio
    512 zxgio
   2832 zxgio
 724025 zxgio
1267712 zxgio
```

From fsstat on intact image:

Offset (byte)	Sector	File System Area
512	0	Boot Sector
2832	1	FAT 0 (corrupted)

Offset (byte)	Sector	File System Area
724025	353	Cluster Area (slack space)
1267712	619	Cluster Area (inside `HOMEWORK.TXT` 619-619 (1) -> EOF)

1. **Verify sector 353**:
   - Sector 353 contained the string in slack space (confirmed via dd and xxd).
   - Offset 1,267,712 confirmed within HOMEWORK.TXT.
   - No occurrence found within actual data of NETWORKS.TXT.

```
istat corrupted_fixed.dd 32
# Output:
    Directory Entry: 32
    Size: 17465
    Name: NETWORKS.TXT
    Sectors: 345 346 347 348 349 350 351 352 353

icat corrupted_fixed.dd 32 | grep zxgio
# Output:
    NULL
```

2. **Slack Space Inspection**:
   - It can be confirmed that the string is contained in the slack space of cluster 353

```
dd if=corrupted_fixed.dd bs=2048 skip=353 count=1 of=sector353.bin
xxd sector353.bin | less
# Output:
    tware....zxgio..
```

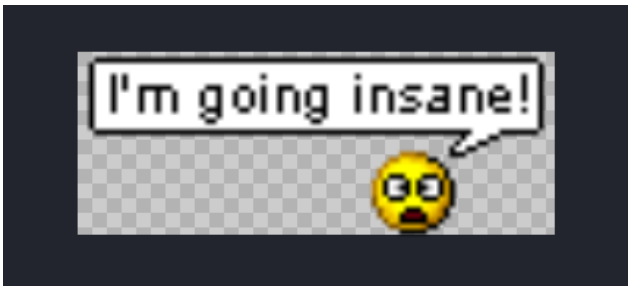
Unlocated Space

The image `corrupted.dd` has a size of 721 sectors, while the FAT12 file system only uses sectors 0-719. Sector 720, being outside the file system, was extracted and analyzed.

```
dd if=corrupted.dd bs=2048 skip=720 count=1 of=unused_sector720.bin

file unused_sector720.bin
# Output:
  ASCII text

echo "asci text" | base64 -d > hidden_file

file hidden_file
# Output:
    GIF image data, version 89a, 86 x 33
```

Results:

- The sector contains a long ASCII string without line terminators.
- Analysis revealed it to be Base64 encoding.
- Decoding the string produced a file recognized as a GIF image.


I'm going insane!