File-System assignment

Verifying hashes

Before starting, let's verify the SHA256 hashes of the uncompressed images.

Note: I'm using macOS .

```
shasum -a 256 *.dd

7f9ef6a650c3eda6d6272efa37bb9d87c234caf27ef1095d89bab93dba44a611  console.dd
44261edf3d078ade9529e995aeec4614b637bde53c9acb9a0ba070fb73c4b994  corrupted.dd
a1ee435924f28e958078824ae8cb4e8595fffde8539c04b1c6ac68cc0059d207b  strange.dd
```

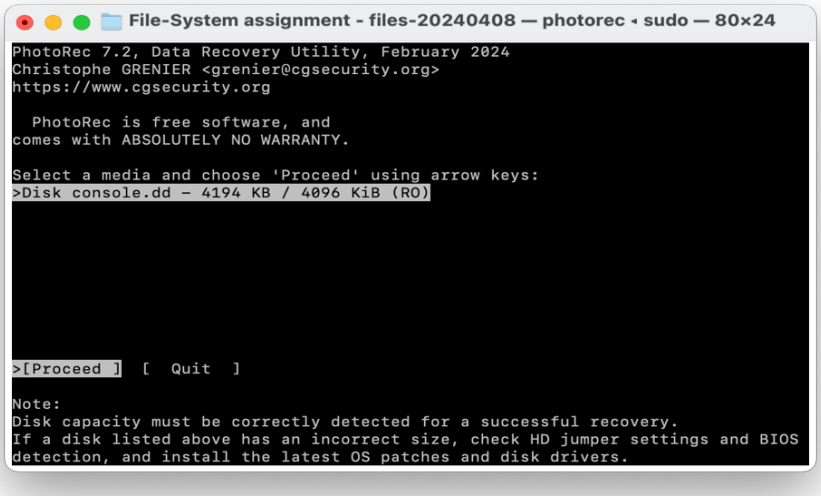
```
shasum -a 256 -c *.sha256

console.dd: OK
corrupted.dd: OK
strange.dd: OK
```

console.dd

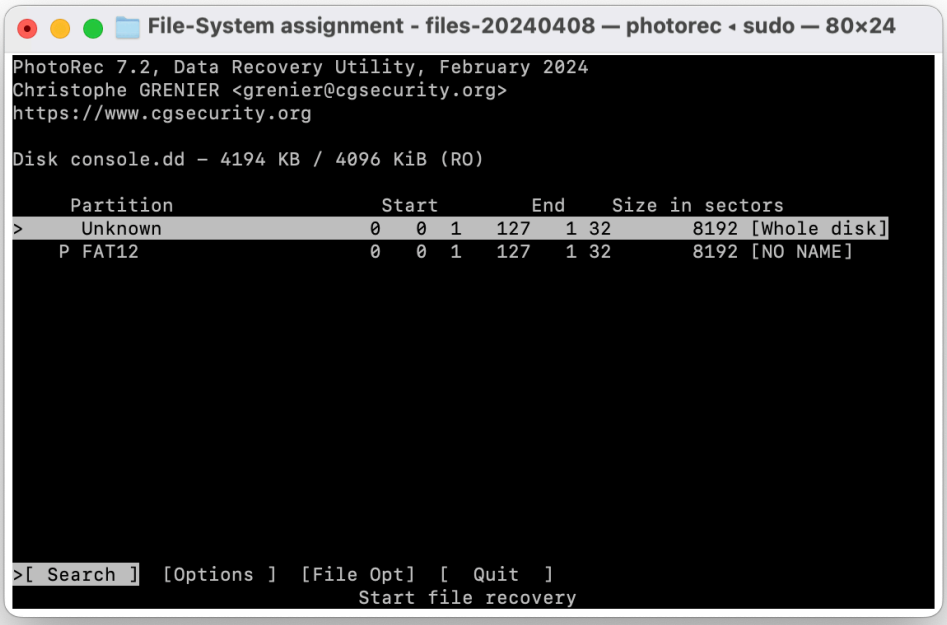
I used **PhotoRec** which is a file data recovery software designed to recover lost pictures from digital camera memory or even hard disks.

```
sudo testdisk-7.2/photorec console.dd
```

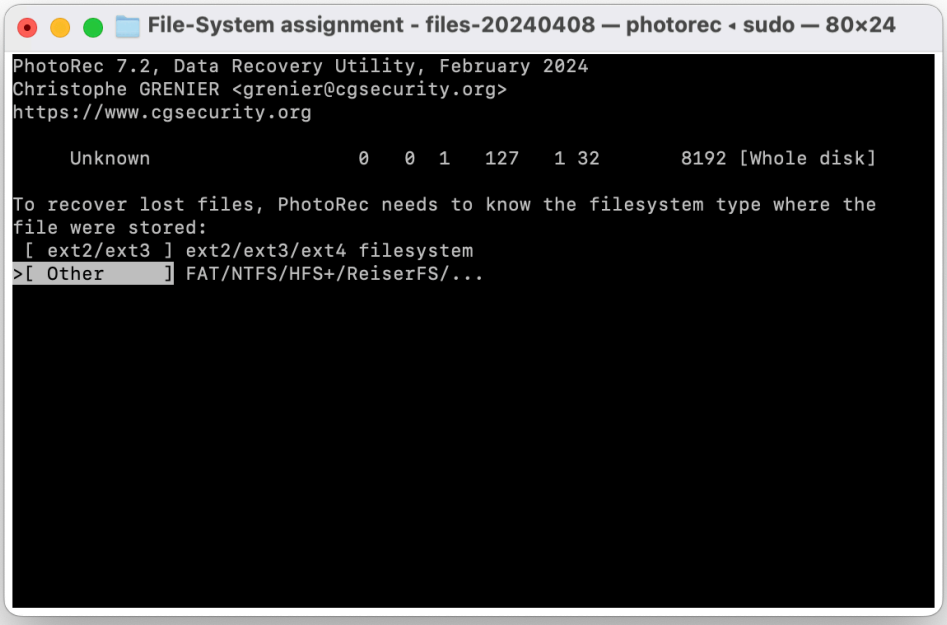
you shoud run the command as a root user. now we select the .dd image file.



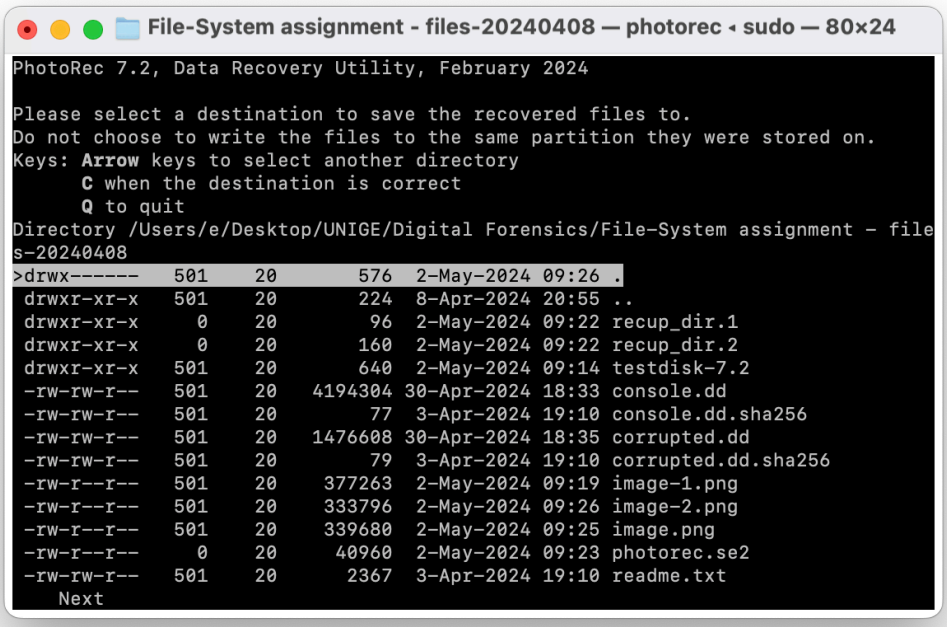
then we choose the whole disk.

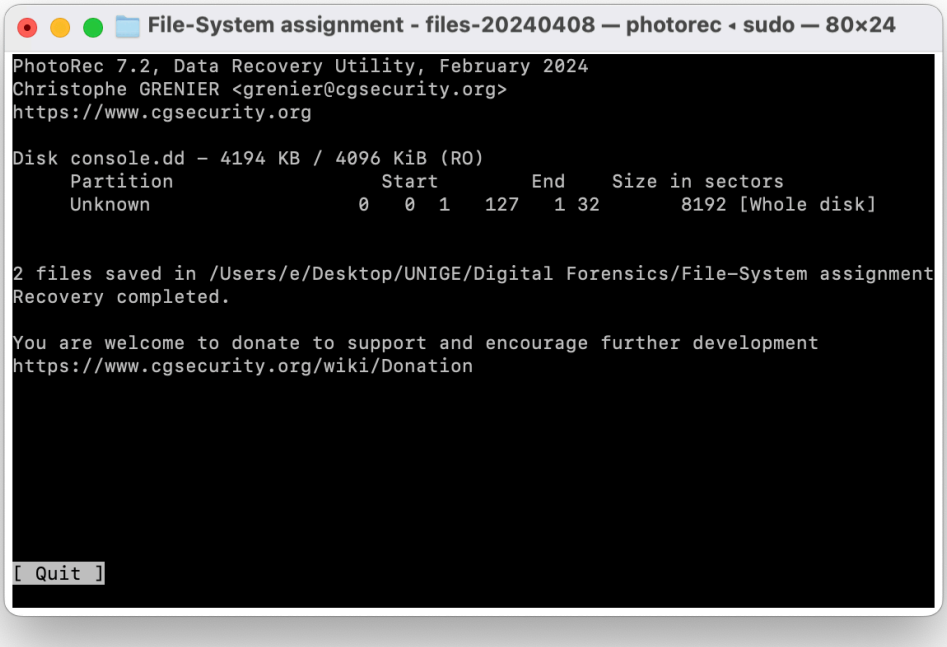


select other as we know it was in fat format.



go to the directory you want to save the found images and click **'c'** button on your keyboard to save them.



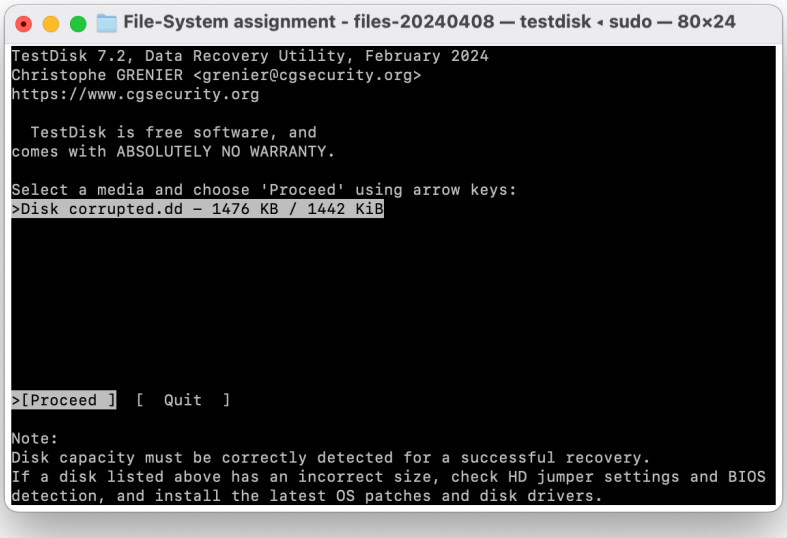


We found two images:

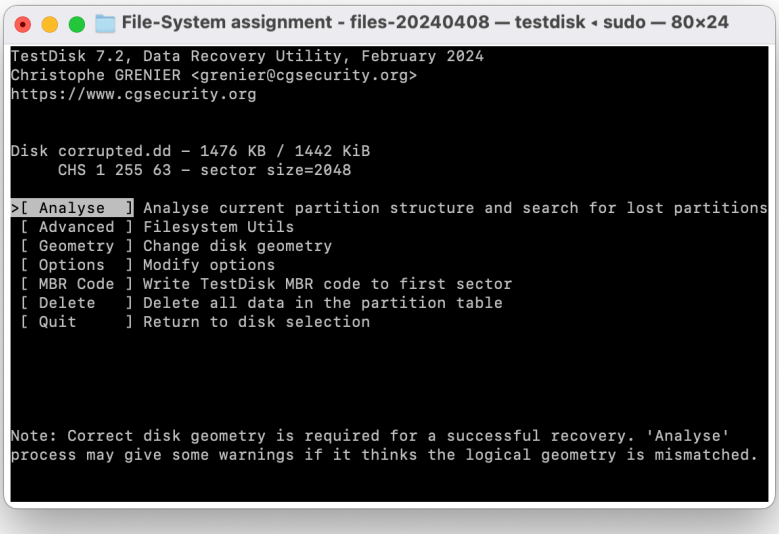
corrupted.dd

We use **TestDisk** to open this file. TestDisk checks the partition and boot sectors of your disks. It is very useful in recovering lost partitions.

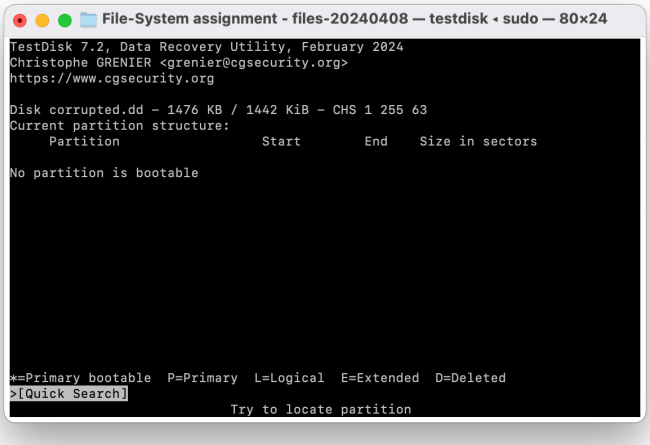
Command: sudo testdisk-7.2/testdisk corrupted.dd



Proceed with the disk, select Intel and then select Analyse:



TestDisk will show if there are any partitions.



There is no prtition found. Now for a detailed analysis of the FAT file system, we use **The Sleuth Kit (TSK)**. It does not require mounting the disk image and works directly with the disk data, which can also prevent accidental writes or changes.

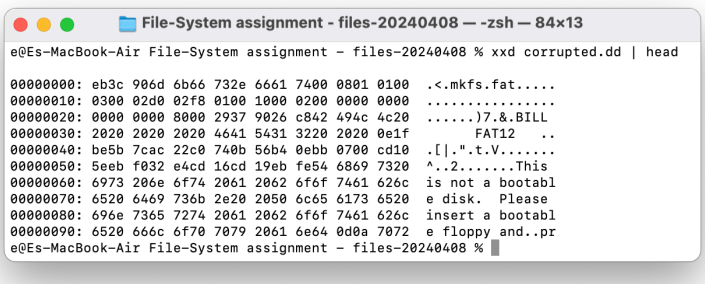
Command: mmls corrupted.dd

This command lists the partition layout of the disk image.

However, we don’t get any results. this suggests that either the disk image doesn't contain any partitions or the partition table itself may be damaged.

Let’s investigate our file some more:

Command: xxd corrupted.dd | head



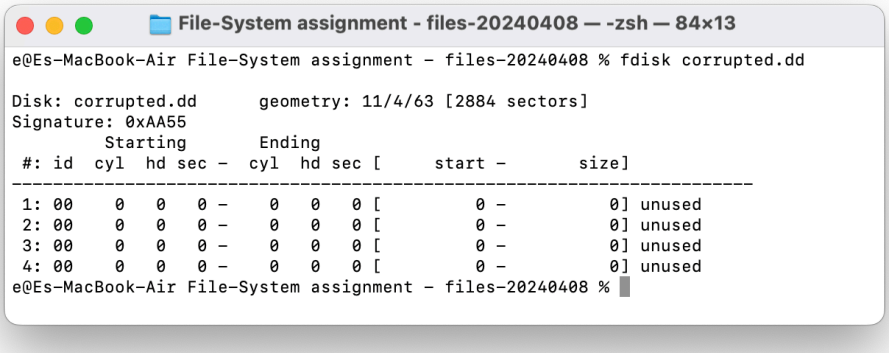
We used **xxd** to view the first few bytes of the image and found some clues!

1. **Volume Name**: The text "....7.8.BILL" might be part of the volume label. Volume labels can be up to 11 characters long in FAT file systems and typically start at byte offset 43 in the boot sector.

2. **File System Type**: The string "FAT12" is indicating that the file system type of the disk image is FAT12.

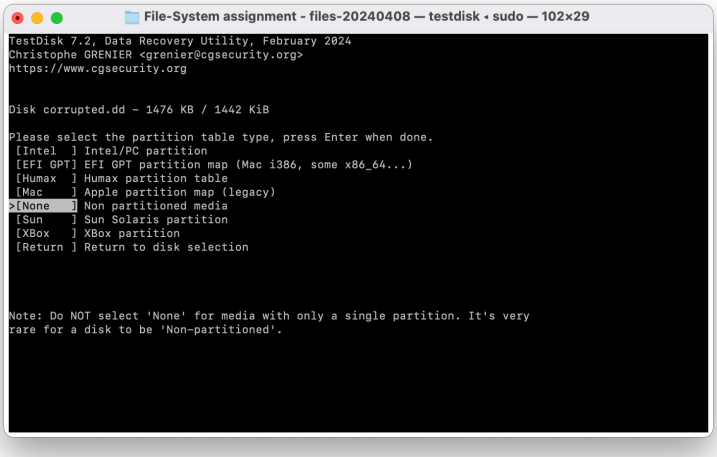
3. **Non-Executable Warning**: The message "This is not a bootable disk. Please insert a bootable floppy and press any key to try again." is common in disk images that are formatted but not made bootable.

Command: `fdisk corrupted.dd`

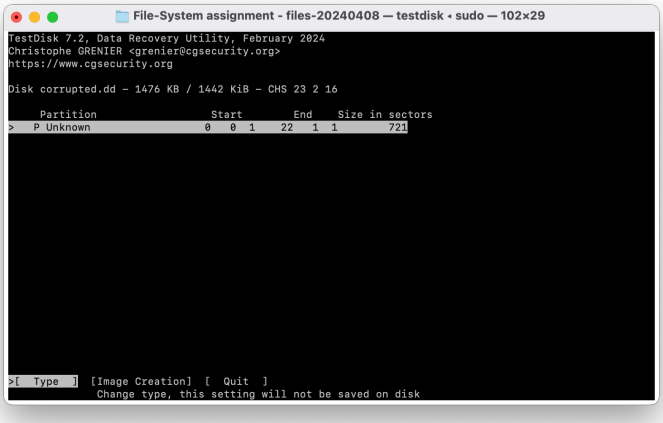


Looks like the file is damaged so let's try and fix it using TestDisk.

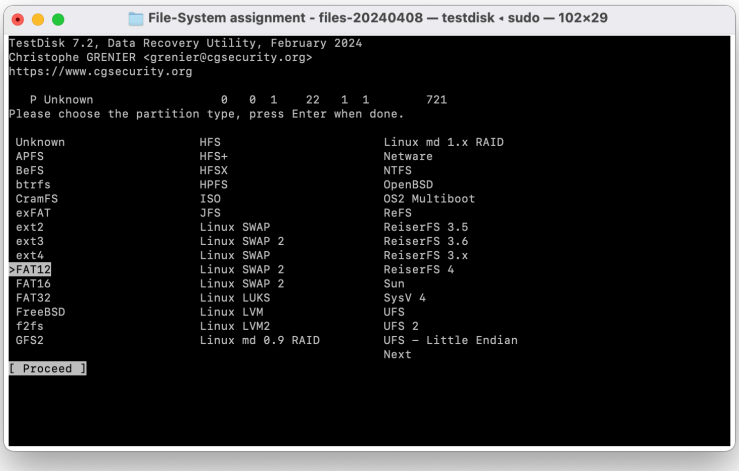
Select None in the picture below.



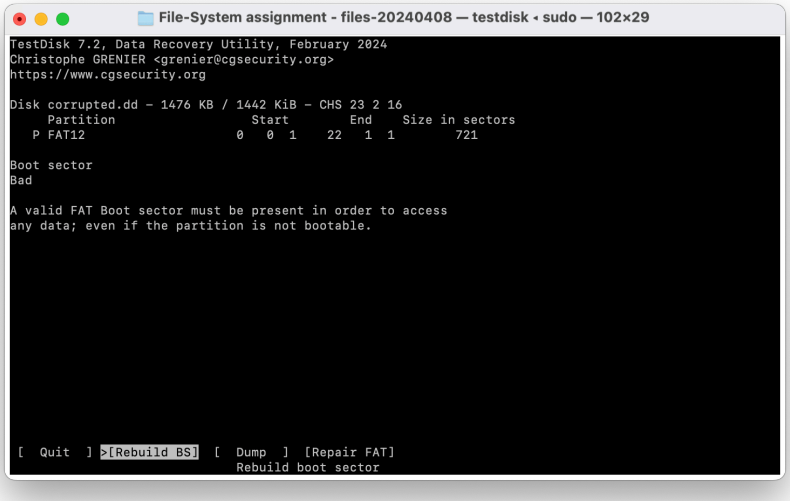
And then press enter on your screen to move on:



Choose FAT12 and proceed:

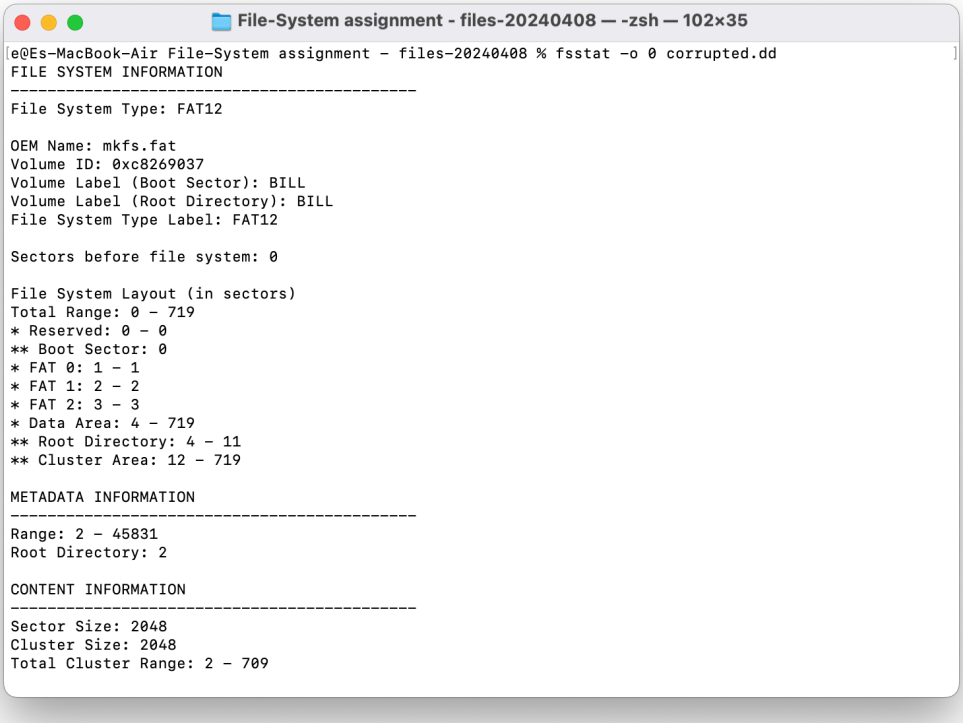


We tried repairing FAT and rebuilding BS but didn't get any result.



We use another command called **fsstat**. This will display the details associated with a file system.

Command: fsstat -o 0 corrupted.dd



So we finally have some answers:

1. **What is the volume label?** The volume label is **BILL** in the Boot Sector and the Root Directory.

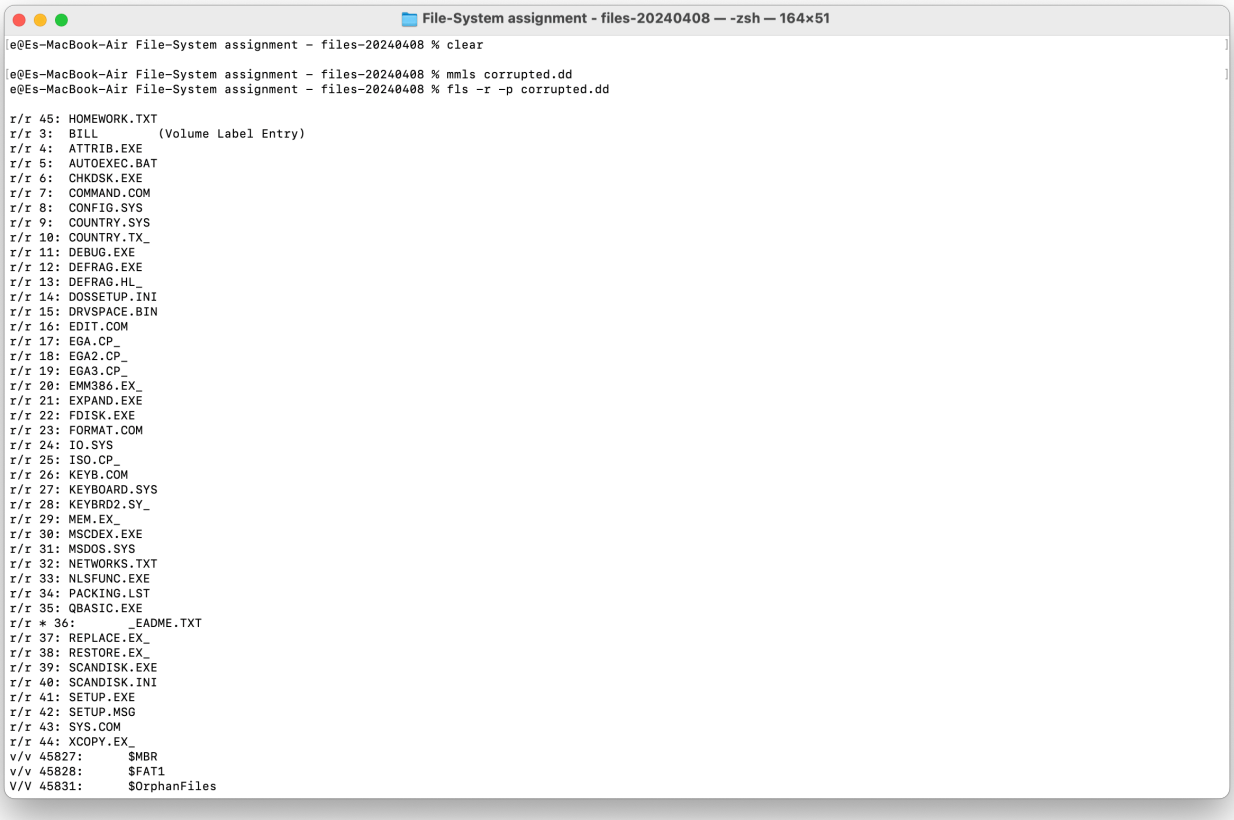
2. **What is the sector size?** The sector size is **2048 bytes**.

3. **What is the cluster size?** The cluster size is also **2048 bytes**.

4. **How many FAT tables are present?** There are **3 FAT tables** present (FAT 0, FAT 1, and FAT 2)

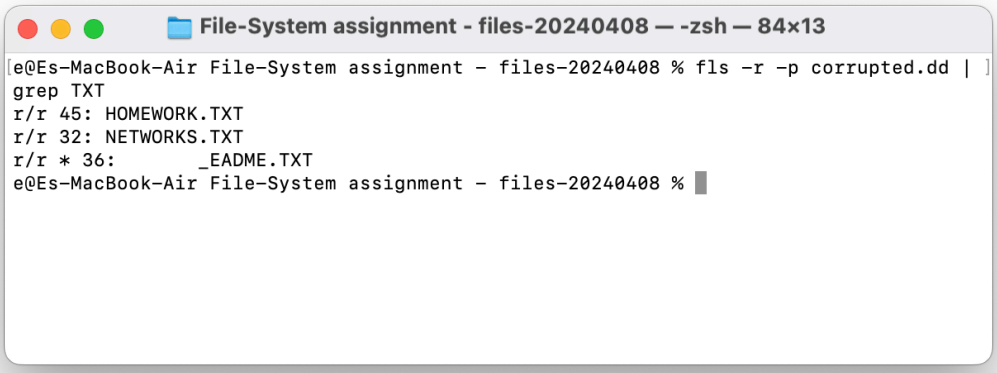
Command: fls -r -p corrupted.dd

This command recursively lists files and paths, including deleted files (**-r** for recursive, **-p** to display full path).



We use grep to only get the files with .txt extention.

Command: fls -r -p corrupted.dd | grep TXT



Extracting HOMEWORK.TXT:

Command: icat -o 0 corrupted.dd 45 > HOMEWORK.TXT

This command extracts the file with inode number 45 and redirects the output to a file named HOMEWORK.TXT.

This file contains **'zxgio'** string!

Extracting NETWORKS.TXT:

Command: icat -o 0 corrupted.dd 32 > NETWORKS.TXT

Extracting _EADME.TXT (Deleted File):

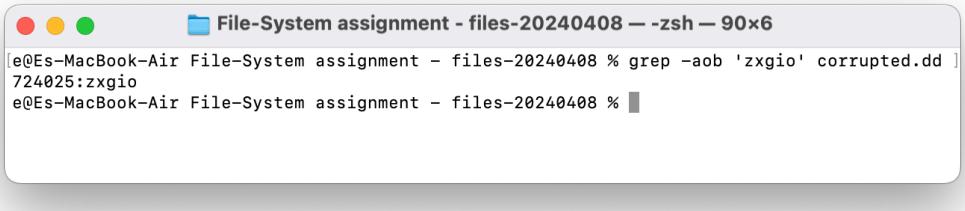
Command: icat -o 0 corrupted.dd 36 > _EADME.TXT

Finding 'zxgio' String Occurrences

we can use the **grep** command to search for strings within the disk image:

Command: grep -aob 'zxgio' corrupted.dd

- **-a** treats the binary file as text.
- **-o** shows only the matched parts.
- **-b** outputs the byte offset.



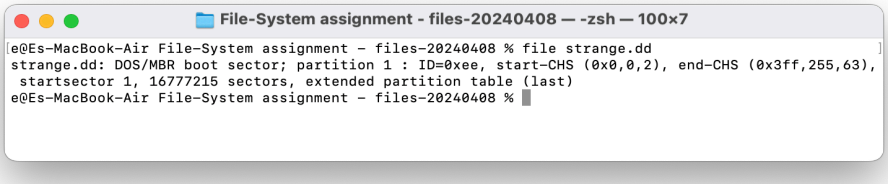
For each occurrence, use **istat** in TSK to determine if the byte offset falls within a file, unused space, or slack space.

---

strange.dd

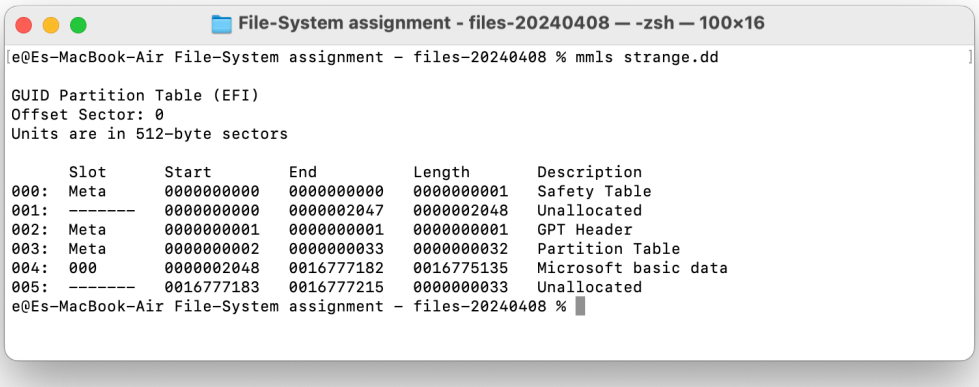
Partition Scheme:

Command: file strange.dd



Using **file** command, we notice the MBR boot sector.

Command: `mmls strange.dd`



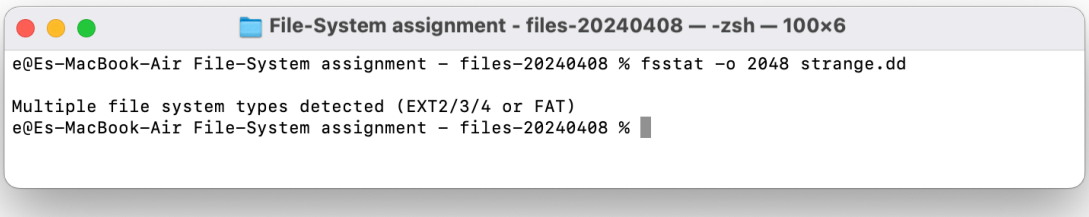
The output indicates that the disk image uses a **GUID Partition Table (GPT)**.

Also, Slot 4 is the First Real Partition. From the **mmls** output, it appears that slot 4 is the first partition designated for storing user data, formatted with a file system recognizable by Windows ("Microsoft basic data").

Identify the file system type:

Command: fsstat -o 2048 strange.dd

Note: The starting sector for slot 4 is 2048 which is why we set it in our command.



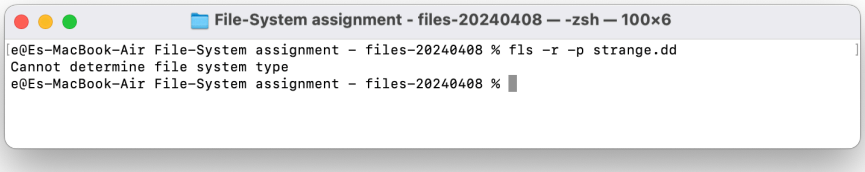
What is Strange About This Image?

- Large blocks of unallocated space before the start and after the GPT table entries and between defined partitions can be intriguing. This could be wasted space or could contain hidden data.

- The mix of a DOS/MBR boot sector signature along with a GPT partition table can be considered unusual. Typically, GPT does not use a boot sector like MBR.

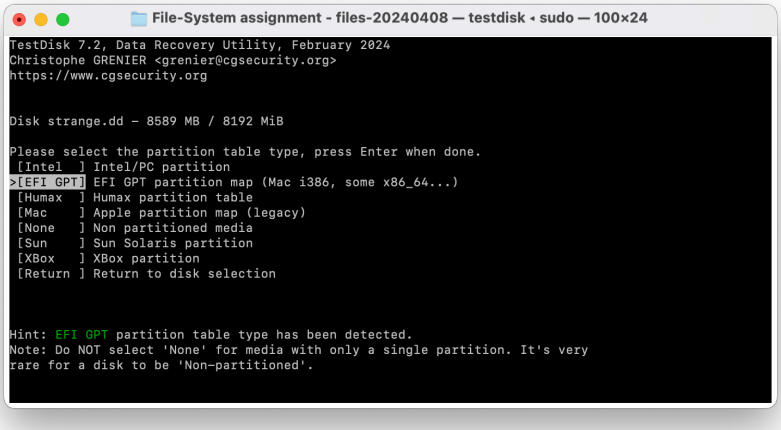
- Multiple file system types are detected which is strange!

We try to list all the files inside the partition:
Command: fls -r -p strange.dd

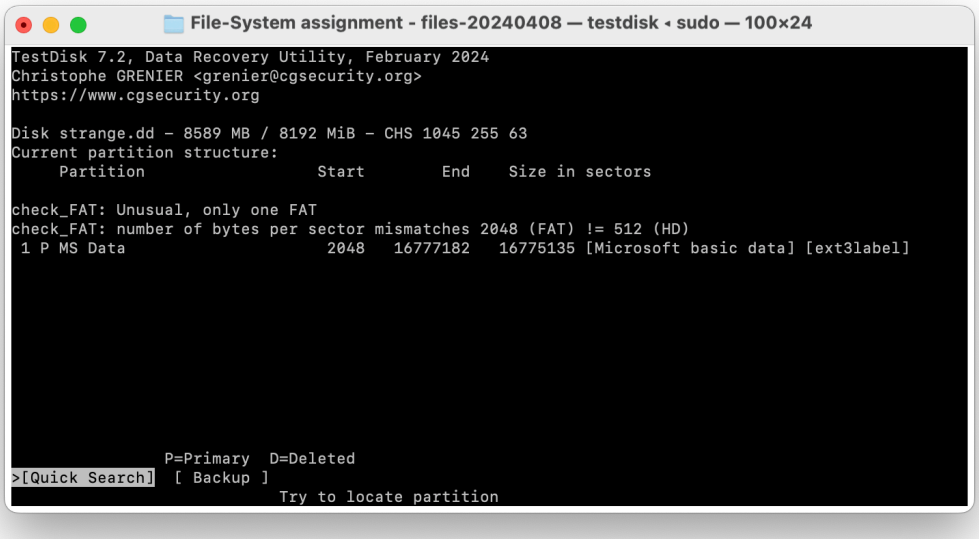


Having multiple types is strange and we can't extract files using previuos commands such as **fls** and **icat**. So let's try a different method and open the image using **TestDisk**.

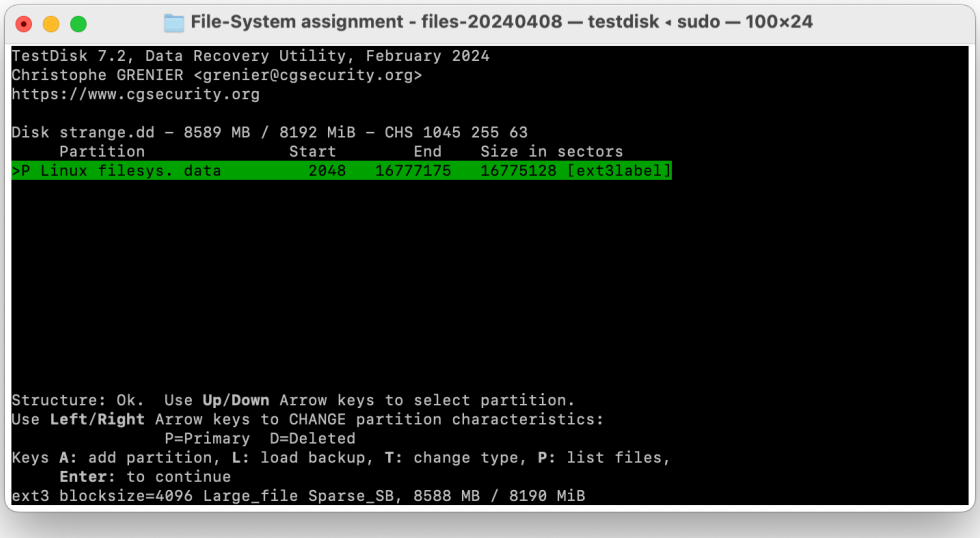
Command: sudo ./testdisk-7.2/testdisk strange.dd



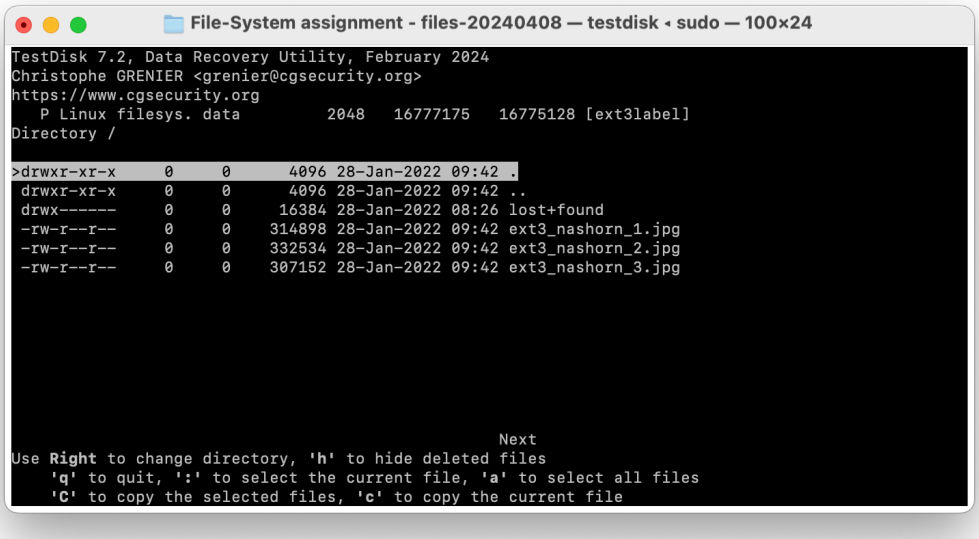
Choose GPT and then Analyse.



Select Quick Search and press **Enter**. it recognizes only one partition:



press **P** button to see the files:



We found 3 images. Let’s extract them!