

Guide

# **File Carving Simulation**


---


Initial Date of the Creation


4 February 2026, 18:44 UTC+7


## Folders

1. Dataset Downloader - Crime Plan
2. Disk Generator - Crime Scene
3. File Carver - Detective
4. Validator - The Judge

 1- Dataset Downloader - Crime Plan

 2- Disk Generator - Crime Scene

 3 - File Carver - Detective

 4 - Validator - The Judge

## Step by Steps

### Step 1 - Creation of the Datasets

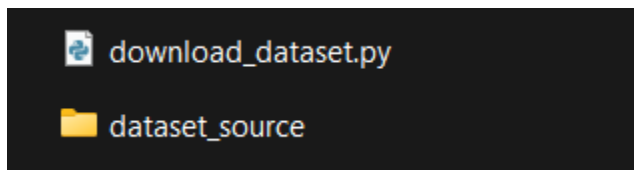
Before we start, we need to make the mock photos, at least 50-100 of them with a mix of .png and .jpg files

Instead of downloading manually or using personal stuff, we can use the python script available on the first folder, **1 - Dataset Downloader - Crime Plan**

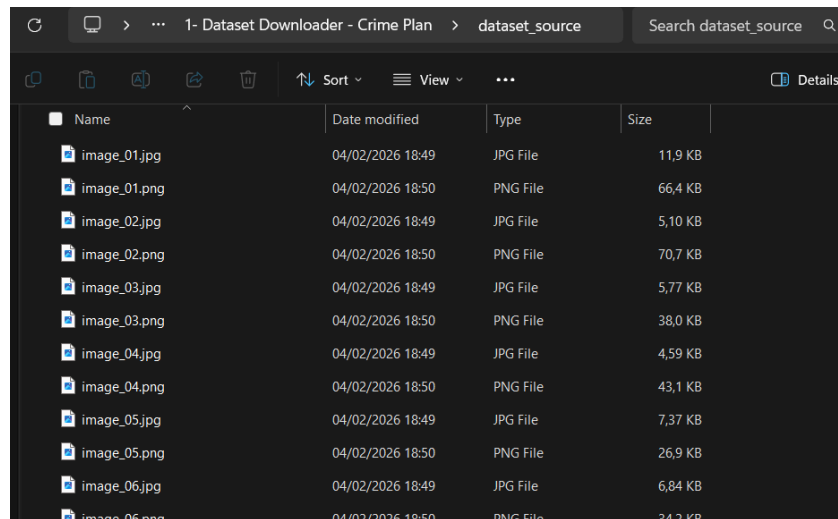
Run the following command in the folder :

```
python download_dataset.py
```

After running the command, you should have a new folder called **dataset\_source**



Inside the folders should be .png and .jpg files that has been downloaded by the script



## Step 2 - Generating the disk with the datasets to simulate deleted files inside a disk

Move the **dataset\_source** folder to the **2 - Disk Generator - Crime Scene** folder

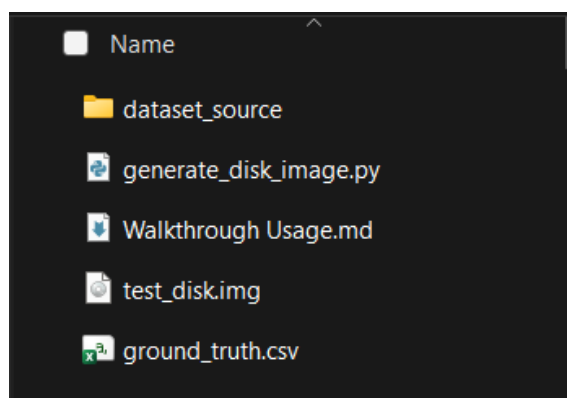
In this step, we will create simulated deleted files on a disk image. The resulting disk image will appear corrupted when you try to open it by double-clicking, because it does not contain a "Table of Contents."

This simulates an unallocated disk space explaining that the file system metadata is missing. However, the raw bytes of the **dataset\_source** photos have already been manually inserted into the disk image.

Run the following command in the folder :

```
python generate_disk_image.py
```

After running the command, you should have a new file called **test\_disk.img** and **ground\_truth.csv**



**test\_disk.img** is the image that simulates the deleted files in a disk that doesn't have a table of contents and it is a fully unallocated disk, we will carve this to get back our photos, you can confirm it is fully raw bytes using [HxD](#) software.

**ground\_truth.csv** are the md5 hashes, start offset decimal, and file size bytes of our dataset or photos. We will use it later on the fourth step **The Judge** to confirm our recovered files to then separate the true positives and the false positives.

### Step 3 - Carving the disk and recovering the deleted files in the image

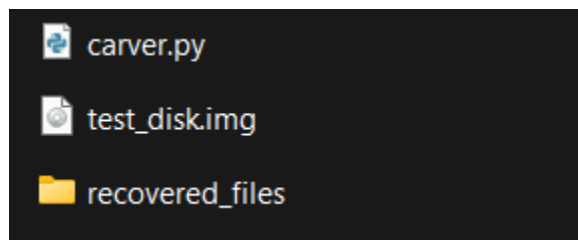
Move the **test\_disk.img** file to the **3 - File Carver - Detective** folder

Now we will start carving the file using a simple algorithm, where the script will scan the disk image for headers and footers of png/jpg [file signatures](#). Then the script will create a folder for the recovered files, expect many noises (false positives and corrupt files) since this is a naive algorithm. That's why next we need a validator to validate the recovered files and sort them out.

**Run the following command in the folder :**

```
python carver.py
```

After running the command, you should have a new folder called **recovered\_files**



## Step 4 - Validating the recovered files and sorting out the false positive

Move the **recovered\_files** folder to the **4 - Validator - The Judge** folder

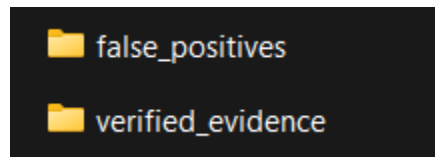
Move the **ground\_truth.csv** file from **2 - Disk Generator - Crime Scene** folder to the **4 - Validator - The Judge** folder

This final step will run a script that will match the files inside the **recovered\_files** folder hashes with the hash in **ground\_truth.csv**. Then it will sort the true positive (The actual original photo files) and the false positives in their own respective folders.

Run the following command in the folder :

```
python validator.py
```

After running the command, you should have a new folder called inside **recovered\_files**



**Verified\_evidence** folder consist of the true positive (original jpg/png files) that has be recovered.

And thats it!