

Presented with xmind

## Własny pomysł

### Przygotowanie środowiska

1. Stworzenie kopii binarnej karty SD, która była w połowie zapełniona zdjęciami i filmami - karta podłączona do komputera za pomocą adaptera
2. Następnie tworzę kopię binarną pamięci aparatu - który jest podłączony do

1  
2  
2

komputera za pomocą kabla micro USB - w taki sposób chcę sprawdzić, czy jest jakakolwiek różnica w zawartości otwartej pamięci	2
3. Formatuję kartę za pomocą oprogramowania aparatu (DMC gx80 Ver. 1.3), a następnie tworzę jej kopię binarną	2
4. Nadpisuję wcześniej sformatowaną kartę innymi zdjęciami (naturalny proces w pracy fotografa) a następnie robię ostatnią już kopię binarną	2
<b>Analiza</b>	3
Foremost	3
RescuePro Deluxe	11
Porównanie ilości odzyskanych poszczególnych plików - RescuePro vs Foremost	
12	
Pakiet Adobe	17
Podsumowanie	17
<b>Przygotowanie środowiska</b>	19
Narzędzia	19
Zmiana ustawień systemowych	19
<b>Autopsy</b>	19
Utworzenie nowej sprawy w programie autopsy oraz wstępna konfiguracja	19
Z jakiego systemu korzystał nasz użytkownik?	20
Informacje o samych użytkownikach?	20
Urządzenia	21
Analiza metadanych EXIF	22
Historia wyszukiwarki	27
Web History	28
Inne ciekawe pliki	29
<b>Zrzut pamięci RAM oraz jej analiza</b>	29
Narzędzia	29
Przygotowanie maszyny wirtualnej	29
Volatility	31
windows.info	31
pslist	31
pstree	32
malfind	34
filescan	35
Sygnatura pliku	36
hivelist	37
netscan	37
Podsumowanie	40
<b>Wykorzystaniu narzędzia ExifTool do wyciągnięcia metadanych z plików graficznych oraz przedstawienia metod odzyskiwania straconych danych.</b>	41
analiza odzyskiwania plików obrazu systemu maszyny wirtualnej	41
foremost	41
JPG	41
PNG	43
MOV	43

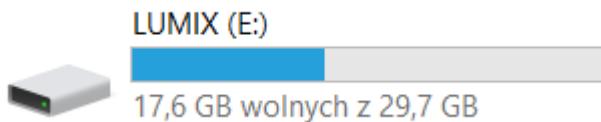
MP4	44
Recoverjpeg	46
Bulk_extractor	47
GPS	49
JPG	52
Podsumowanie	53
<b>Wyciągnięciu danych ze smartfona. Analiza plików (PLIST i SQLite).</b>	<b>54</b>
Analiza sqlite	54
Przygotowanie środowiska	56
Analiza Autopsy	56
TimeLine	56
Call Logs	57
Lokalizacja GPS	58
1) Zdjęcia zrobione telefonem wraz z zostawioną, włączoną lokalizacją GPS	
59	59
2) Współrzędne lokalizacyjne sieci wi-fi	59
3) Lokalizacja zaparkowania samochodu	60
Urządzenia BlueTooth	62
Calendar Entries	64
Oprogramowanie	65
Messages	66
Events	68
Web Search i Web History	68
Podsumowanie	69
<b>Utworzenie kopii binarnej nośnika</b>	<b>70</b>
ewfacquire	70
hash	70
zamontowanie obrazu	71
System plików	71
Analiza grup bloków	71
Analiza plików	71
Odzyskanie pliku	71
Exif	72
fls	72
Odzyskiwanie hasła	72

## Własny pomysł

Jako iż w pewnej części zajmuję się fotografią i filmowaniem, sprawdzę jak efektywne jest

- 1) odzyskiwanie zdjęć z profesjonalnych kart SD Sandiska za pomocą dostępnych narzędzi open source (**foremost**) jak i komercyjnych - **RescuePRO Deluxe**
- 2) jak bezpieczne jest formatowanie kart za pomocą oprogramowania aparatu

## Przygotowanie środowiska



1. Stworzenie kopii binarnej karty SD, która była w połowie zapełniona zdjęciami i filmami - karta podłączona do komputera za pomocą adaptera

The screenshot shows the PassMark imageUSB V1.5 Build 1003 software interface. The main title is "Create and write an image of a USB drive".

**Step 1: Select the USB drive(s) to be processed**

A checkbox is selected for "Generic STORAGE DEVICE (Serial: 000000001532 Disk: 2, Part. Type: MBR, Size: 29.71 GB, Volumes: E)". Below it, details show "E: Label: LUMIX, FileSystem: FAT32, Size: 29.71 GB", "Start Time: 09:30:09", and "Create Progress: 76.8% (57.3 MB/sec)".

Buttons at the bottom of this section include "Select All", "Unselect All", "Drives Selected: 1", and "Refresh Drives".

**Step 2: Select the action to be performed on the selected USB drive(s)**

Radio buttons for actions are shown: "Write image to USB drive" (unchecked), "Create image from USB drive" (checked), "Zero USB drive" (unchecked), and "Reformat USB drive (Windows Vista or later)" (unchecked). To the right, under "Available Options", are checkboxes for "Post Image Verification" (checked), "Extend/Add Partition (NTFS Only)", "Boot Sector(s) Only", and "Beep on Completion". A dropdown menu for "Format Option" is set to "NTFS".

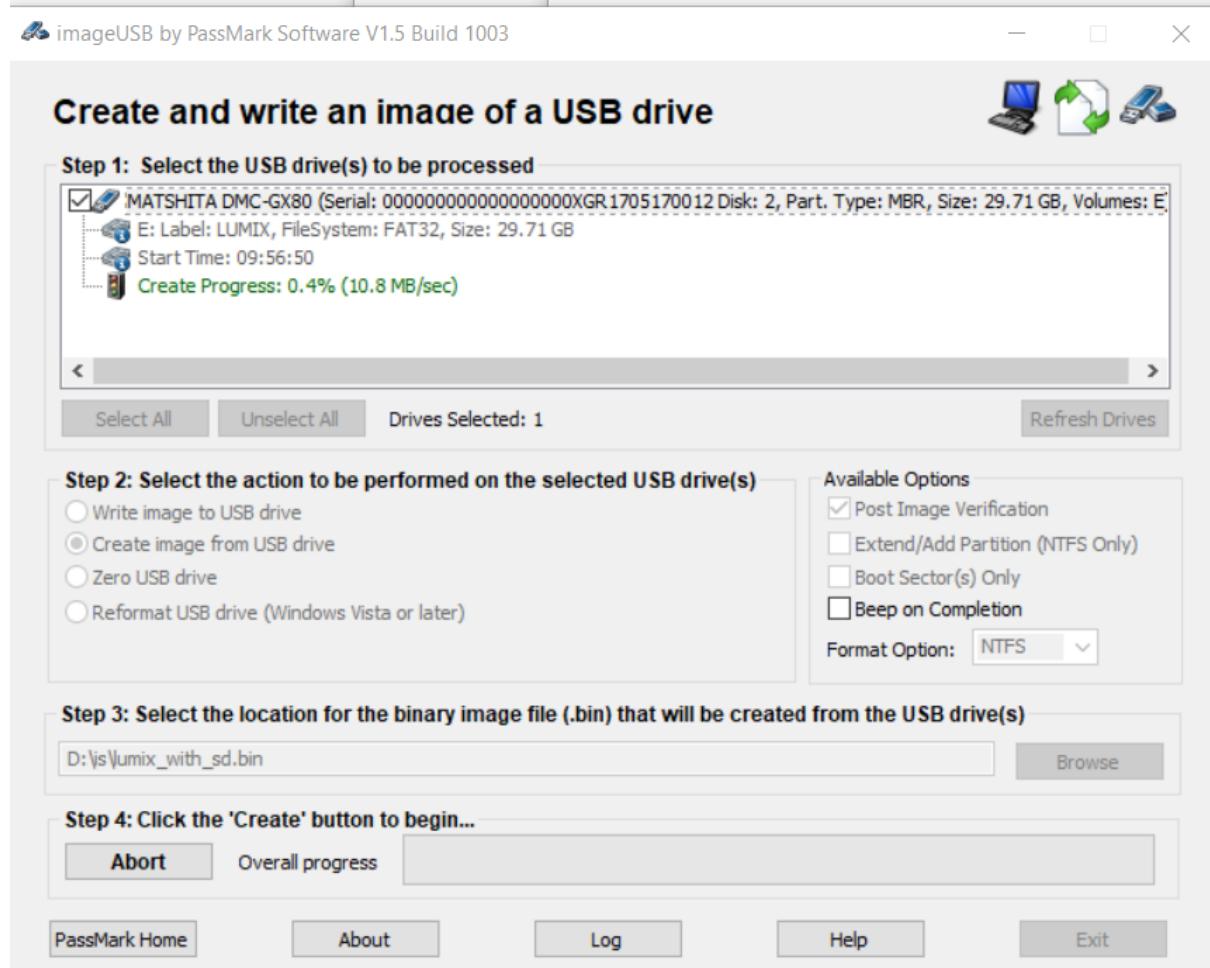
**Step 3: Select the location for the binary image file (.bin) that will be created from the USB drive(s)**

A text input field contains "D:\ls\sandisk\_sd.bin" with a "Browse" button to its right.

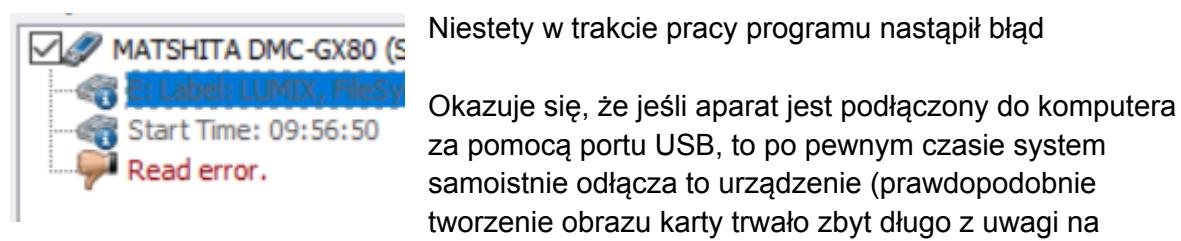
**Step 4: Click the 'Create' button to begin...**

A progress bar at the bottom shows "Overall progress" at 100% completion. Buttons at the bottom of the software window include "Abort", "About", "Log", "Help", and "Exit".

2. Następnie tworzę kopię binarną pamięci aparatu - który jest podłączony do komputera za pomocą kabla micro USB - w taki sposób chcę sprawdzić, czy jest jakakolwiek różnica w zawartości otwartej pamięci

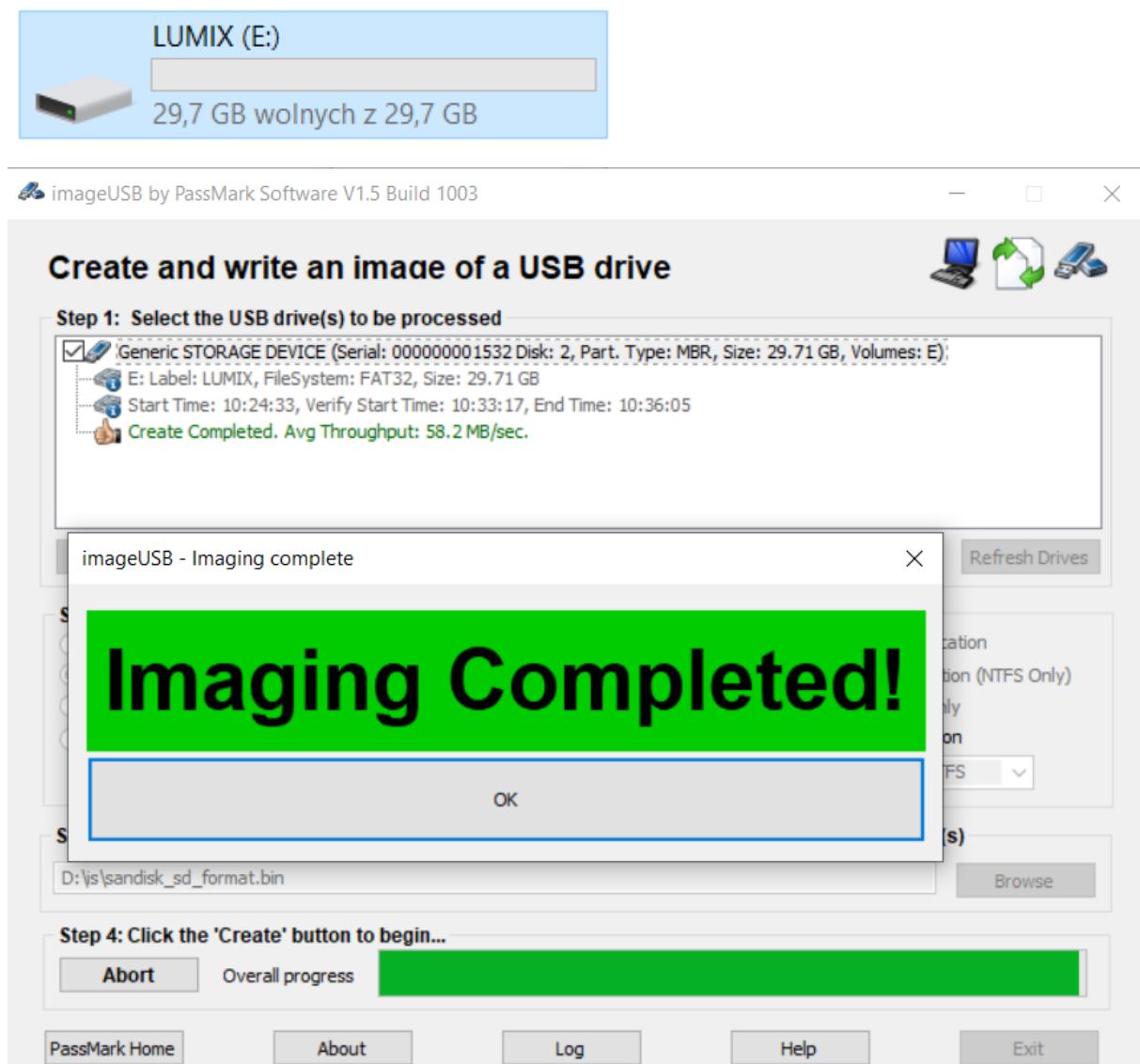


Spostrzeżenie: na pierwszy rzut oka, po podłączeniu aparatu do komputera zawartość karty SD się nie zmieniła - jednakże znacząco spadła prędkość odczytu, ciężko jest odtworzyć jakikolwiek film w sposób *bez zobaczenia artefaktów*. Podczas tworzenia kopii binarnej również znacząco spadła prędkość - z 60/160 MB/sec do około 11 MB/sec



W takim razie analiza powyższego podpunktu przyjmie trochę inną formę - **ile danych można uzyskać z uszkodzonej karty/uszkodzonego obrazu?**

3. Formatuję kartę za pomocą oprogramowania aparatu (DMC gx80 Ver. 1.3), a następnie tworzę jej kopię binarną



4. Nadpisuję wcześniej sformatowaną kartę innymi zdjęciami (naturalny proces w pracy fotografa) a następnie robię ostatnią już kopię binarną

## Analiza

### Foremost

z doświadczenia mogę stwierdzić, że jest to naprawdę skuteczne narzędzie. Problemem jest dostępność rozszerzeń, a raczej ich brak

Formaty, które będą mnie interesowały to:

- .JPG
- .RW2 (domyślne rozszerzenie do zapisywania surowych zdjęć firmy Panasonic)
- .MOV

Następnie w pliku konfiguracyjnym dodaję rozszerzenia - .rw2 oraz .mov. Warto tutaj wspomnieć, że aby to zrobić należy znaleźć tak zwany *header* rozszerzenia. Można to zrobić za pomocą analizowania kolejnych plików komendą *hexdump -C* oraz porównywania części wspólnych

```
[arek@DESKTOP-8V3EBJB 133_PANA] $ hexdump -C P1330794.RW2 | head -20
00000000  49 49 55 00 18 00 00 00  88 e7 74 d8 f8 25 1d 4d  | IIU.....t..%.M|
00000010  94 7a 6e 77 82 2b 5d 6a  35 00 01 00 07 00 04 00  | .znw.+]j5.....|
00000020  00 00 30 33 36 30 02 00  03 00 01 00 00 00 d0 12  | ..0360.....|
00000030  00 00 03 00 03 00 01 00  00 00 88 0d 00 00 04 00  | .....|
00000040  03 00 01 00 00 00 08 00  00 00 05 00 03 00 01 00  | .....|
00000050  00 00 08 00 00 00 06 00  03 00 01 00 00 00 80 0d  | .....|
00000060  00 00 07 00 03 00 01 00  00 00 f8 11 00 00 08 00  | .....|
00000070  03 00 01 00 00 00 01 00  00 00 09 00 03 00 01 00  | .....|
00000080  00 00 04 00 00 00 0a 00  03 00 01 00 00 00 0c 00  | .....|
00000090  00 00 0b 00 03 00 01 00  00 00 0c 86 00 00 0d 00  | .....|
000000a0  03 00 01 00 00 00 01 00  00 00 0e 00 03 00 01 00  | .....|
000000b0  00 00 ff 0f 00 00 0f 00  03 00 01 00 00 00 ff 0f  | .....|
000000c0  00 00 10 00 03 00 01 00  00 00 ff 0f 00 00 17 00  | .....|
000000d0  03 00 01 00 00 00 d0 07  00 00 18 00 03 00 01 00  | .....|
000000e0  00 00 00 01 00 00 19 00  03 00 01 00 00 00 00 01  | .....|
000000f0  00 00 1a 00 03 00 01 00  00 00 00 01 00 00 1b 00  | .....|
00000100  07 00 2a 00 00 00 9a 02  00 00 1c 00 03 00 01 00  | ...*....|
00000110  00 00 80 00 00 00 1d 00  03 00 01 00 00 00 80 00  | .....|
00000120  00 00 1e 00 03 00 01 00  00 00 80 00 00 00 24 00  | .....$..|
00000130  03 00 01 00 00 00 f7 01  00 00 25 00 03 00 01 00  | .....%....|
```

Zauważylem, że w wszystkich plikach z rozszerzeniem .rw2 pojawia się powyższe znaki

```
73 #
74 # GIF and JPG files (very common)
75 # (NOTE THESE FORMATS HAVE BUILTIN EXTRACTION FUNCTION)
76 # gif y 155000000 \x47\x49\x46\x38\x37\x61 \x00\x3b
77 # gif y 155000000 \x47\x49\x46\x38\x39\x61 \x00\x00\x3b
78 # jpg n 20000000 \xff\xd8\xff\xe0\x00\x10 \xff\xd9
79 # jpg n 20000000 \xff\xd8\xff\xe1 \xff\xd9
80 # jpg n 20000000 \xff\xd8 \xff\xd9
81 # rw2 n 200000000 \x49\x49\x55\x00\x18
```

#### Plik `/etc/foremost.conf`

Uruchamiam program foremost na każdym z plików

```
[arek@DESKTOP-8V3EBJB własne] $ foremost sandisk_sd_format
Processing: sandisk_sd_format
```

Powstały 4 osobne foldery wraz z wyekstraktowanymi zdjęciami, w każdym znajdują się jakieś odzyskane zdjęcia, które podzielę na następujące kategorie:



`output_lumix_with_sd`



`output_sandisk_sd`



`output_sandisk_sd_form`

`at`



`output_sandisk_sd_form`

`at_overwritten`

- 1) W pełni odzyskane zdjęcia, w maksymalnej rozdzielczości, rozmiarze oraz zachowaną resztą - dane exif. Bez problemu można je przeglądać a nawet dalej użyć
- 2) Zdjęcia z nie zachowaną pełną rozdzielczością, jednakże z zachowaną całą resztą (w tym metadane). Również bezproblemowe jeśli chodzi o ich sam podgląd
- 3) Obrócone zdjęcia z możliwością podglądu, w zmniejszonej rozdzielczości (w tym dziwnych proporcjach), bez zachowanych metadanych (punkt 2) jednakże bez metadanych)
- 4) Uszkodzone zdjęcia - w niektórych nawet pozostały metadane
- 5) Podglądy miniaturek filmów - w postaci zdjęć w formacie .jpg z naprawdę małą rozdzielczością (416 × 240 pixeli) i brakiem jakichkolwiek metadanych

Properties	Properties	Properties
Size 5184 × 3888 pixels	Size 1440 × 1920 pixels	Size 1440 × 1080 pixels
Type JPEG image	Type JPEG image	Type JPEG image
File Size 6.5 MB	File Size 681.7 kB	File Size 956.0 kB
Folder <a href="#">jpg</a>	Folder <a href="#">jpg</a>	Folder <a href="#">jpg</a>
Aperture f/1.7	Aperture f/5.0	Aperture
Exposure 1/100 sec.	Exposure 1/200 sec.	Exposure
Focal Length 50.0 (35mm film), 25.0 (lens)	Focal Length 26.0 (35mm film), 13.0 (lens)	Focal Length
ISO 3200	ISO 400	ISO
Metering Pattern	Metering Pattern	Metering
Camera DC-G9	Camera DMC-GX80	Camera
Date Fri, 24 December 2021	Date Thu, 11 November 2021	Date
Time 12:57:54 PM	Time 04:12:25 PM	Time
<a href="#">Show Details</a>		



Moja hipoteza a propos punktu **2)** - mimo próby zidentyfikowania charakterystycznych headerów surowych plików z rozszerzeniem **.rw2**, nie udało się żadnego odzyskać w pierwotnej formie, jednakże powyższe zdjęcia o zmniejszonej rozdzielczości mogą być z nich odzyskane. Niestety nie zawsze skutecznie - stąd powstaje problem obracania zdjęcia z uwagi na brak metadanych (jak punkt **3)**). Co ciekawe, takie zachowanie wydaje się być częściowo normalne, z mojego doświadczenia - próbując przeglądać surowe zdjęcia prosto z karty SD, bez odpowiedniego kodeka, Windows ma problemy z odtwarzaniem takich plików, niepotrzebnie je obraca i nie wyświetla żadnych metadanych.

Los zdjęć typu **4)** to najprawdopodobniej następująca sytuacja: zdjęcie zostało usunięte w trakcie sesji i następnie nadpisane przez inne - dlatego pozostała część danych.

Dostępne zdjęcia w folderach typu *output* najczęściej zostały odzyskane w następujących kombinacjach:

- 1) i 2)
- 1), 2) i 3)
- 1) i 3)
- 2) i 3)
- 2)

- 5)
- 4)

#### Spostrzeżenia:

- Zdjęcia o zmniejszonej rozdzielczości zachowują dokładne proporcje wymiaru pixeli, oraz mają 10 krotnie mniejszą rozdzielczość
- Po głębszej analizie, zauważałem, że podczas sesji gdzie zapisywałem same zdjęcia w formacie RAW, można odzyskać 1) i 2) lub 3) - w takim razie zdjęcia o zmniejszonej rozdzielczości mogą być zarówno miniaturkami plików .jpg jak i .rw2 - warto zaznaczyć iż nie jest to reguła
- sposób oraz warunki utworzenia zrzutu obrazu karty SD nie mają wpływu na powyższe własności (kategorie i kombinacje)

Terminologia, której będę używał w następującej analizie:

- *sandisk\_sd* - odzyskane pliki z kopii binarnej karty SD, która była w połowie zapełniona zdjęciami i filmami - karta podłączona do komputera za pomocą adaptera
- *sandisk\_sd\_format* - odzyskane pliki z kopii binarnej karty SD, która była sformatowana za pomocą oprogramowania aparatu
- *lumix\_with\_sd* - odzyskane pliki z kopii binarnej pamięci aparatu - który jest podłączony do komputera za pomocą kabla micro USB
- *sandisk\_sd\_format\_overwritten* - odzyskane pliki z kopii binarnej karty SD najpierw sformatowanej za pomocą oprogramowania aparatu, a następnie w połowie nadpisana innymi plikami (zdjęciami)

własność	<i>sandisk_sd</i>	<i>sandisk_sd_format</i>	<i>lumix_with_sd</i>	<i>sandisk_sd_format_overwritten</i>
Ilość odzyskanych plików	1295	1294	741	1360
występujące kombinacje	wszystkie	wszystkie	wszystkie	wszystkie
Ilość odzyskanych zdjęć w wysokiej rozdzielczości	400	399	242	447
Ilość odzyskanych zdjęć wraz z dużą ilością metadanych	800	799	483	839
Ilość	494	494	257	520

odzyskanych zdjęć bez metadanych				
Ilość odzyskanych miniatur filmów	85	85	12	70

Na potrzeby powyższych statystyk napisałem proste skrypty::

```
^C[arek@fedora jpg]$ exiftool * | grep "Image Size" | awk -F":\" '{print $2}' | awk -F"x" '{ if ($1 > 3000) {print} }' | wc -l
242
```

```
1 []
2 result=0
3
4 for file in /home/arek/astudia/is/projekt/wlasne/output_lumix_with_sd/jpg/*
5 do
6     num=$(exiftool $file | wc -l)
7     #echo num $num
8     if [[ $num -lt 25 ]];
9     then
10         result=$((result+1))
11         #echo result $result
12     fi
13     num=0
14 done
15
16 echo $result
```

Co ciekawe, analizując dane exif, znalazłem model obiektywu, którego nie posiadam, a mimo to dzięki niemu zostało wykonane zdjęcie

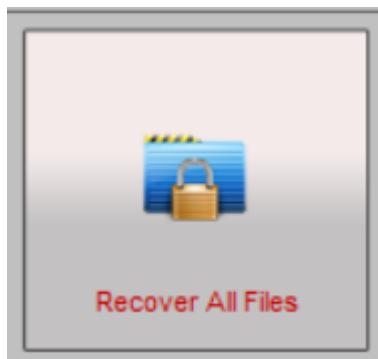
```
Lens Type : Leica D Vario Elmar 14-150mm F3.5-5.6
Advanced Scene Mode : Off
Circle Of Confusion : 0.015 mm
Field Of View : 61.9 deg
Focal Length : 15.0 mm (35 mm equivalent: 30.0 mm)
Hyperfocal Distance : 12.48 m
Lens ID : Leica D Vario Elmar 14-150mm F3.5-5.6
```

Nie zgadza się również przesłona - zdjęcie zostało wykonane z wartością równą f/1,2 a minimalna dla domniemanego obiektywu to f/3,5

## RescuePro Deluxe

Na sam początek warto zaznaczyć iż program współpracuje z bezpośrednio zamontowanymi nośnikami lub zrzutami obrazu z rozszerzeniem *.img*  
Dlatego muszę skonwertować wcześniejszy obraz kart z rozszerzenia *.bin* do *.img*

Rozpoczynam skanowanie:



Wybieram opcję Recover All Files

# RescuePRO® Deluxe

## File Recovery Utility

# SanDisk®

SanDisk, RescuePRO and the SanDisk logo are registered trademarks of SanDisk Corporation, registered in the United States and other countries. Copyright © LC Technology International, Inc. All Rights Reserved.

### Select Device

-  [UDS:BusType.btNVME] Fixed Disk WDC PC SN520 SDAP  
Total size 256 GB
-  Mounted file D:\is\lumix\_with\_sd.img  
Total size 7.5 GB
-  Backup Media File (.IMG)

Enable fragmented recovery



Back

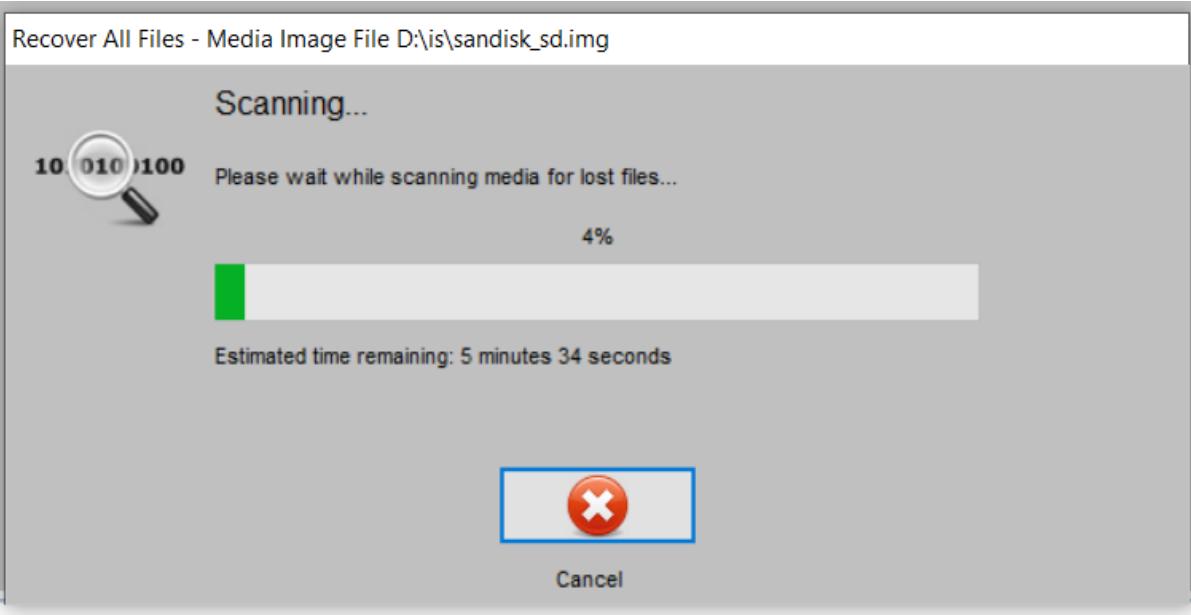


Start



Help

Zrzut procesu działania programu



Sector 2877440, 154.7 MB/sec

RescuePRO 7.0.2.2

Przykładowy wygląd programu zaraz po zakończeniu skanowania

RescuePRO 7.0.2.2 - Media Image File D:\is\lumix\_with\_sd.img

File View Help

New Scan Preview Properties Open Output Folder Update Help

Information

File009.rw2

Size	19649024
Resolution	4816x3464
Depth	
Device	Panasonic DMC-GX80
ISO	200
F-Value	4.0
Exposure Time	10/32000
Time	2021:09:03 11:02:15

OK

< >

983 files, 0 read errors

Sector 0, 0 B/sec

RescuePRO 7.0.2.2

Warto zaznaczyć, że proces skanowania programu trwał zauważalnie krócej niż w przypadku foremost

Już na pierwszy rzut oka można zobaczyć jeden z największych plusów tego programu - ilość dostępnych rozszerzeń.

W tym przypadku udało się odzyskać surowe zdjęcia w formacie **.rw2** oraz filmy **.mp4** - czego się nie udało dla oprogramowania **foremost**, mimo próby przeanalizowania i dodania nagłówka do reguł

Już na pierwszy rzut oka widać, że zostało odzyskanych znacznie więcej zdjęć - w tym wiele w wysokiej rozdzielczości i z metadanymi - których nie udało się odzyskać foremost'em

Co najważniejsze, działa odtwarzanie plików wideo - nawet w oryginalnej rozdzielczości oraz pełną ścieżką audio (fullhd a nawet 4k), jednakże mam wrażenie, że w pomniejszonej jakości

Odzyskane surowe zdjęcia są pełnoprawnymi plikami nadającymi się do dalszego użytku - np. obróbki graficznej

Program RescuePro również odnalazł uszkodzone pliki, dokładnie te same co foremost.

Porównanie ilości odzyskanych poszczególnych plików - RescuePro vs Foremost

**Na czerwono dane z Foremost**

własność	sandisk_sd	sandisk_sd_for mat	lumix_with_sd	sandisk_sd_for mat_overwritten
Ilość odzyskanych plików	1729 <b>1295</b>	1728 <b>1294</b>	982 <b>741</b>	1779 <b>1360</b>
występujące kombinacje	wszystkie <b>wszystkie</b>	wszystkie <b>wszystkie</b>	wszystkie <b>wszystkie</b>	wszystkie <b>wszystkie</b>
Ilość odzyskanych zdjęć w wysokiej rozdzielczości	850 <b>400</b>	849 <b>399</b>	485 <b>242</b>	883 <b>447</b>
Ilość odzyskanych zdjęć wraz z dużą ilością metadanych	1315 <b>800</b>	1314 <b>799</b>	736 <b>483</b>	1324 <b>839</b>
Ilość odzyskanych zdjęć bez	410 <b>494</b>	410 <b>494</b>	245 <b>257</b>	451 <b>520</b>

metadanych				
Ilość odzyskanych plików video (miniatur)	70 85	70 85	10 12	52 70

Liczby nie kłamią - jak widać w powyższej tabeli, ilość odzyskanych plików i szczegółów dzięki **RescuePro** jest znacznie większe. Jest to najbardziej widocznie dla zdjęć w wysokiej rozdzielcości i metadanych - uzyskana liczba plików jest prawie dwukrotnie większa. Najciekawsze dla mnie są wyniki odzyskanych filmów vs samych miniaturek - udało się odzyskać więcej miniatur niż samych plików video.

Program **RescuePro** działał zdecydowanie szybciej oraz udało mu się odnaleźć znacznie więcej istotnych informacji - które zostały pominięte na przykładzie analizowanego oprogramowania Open Source.

Myślę, że jeśli ktoś byłby bardzo zdesperowany do odzyskania dużej ilości plików a tym samym jak najdokładniej, to warto zapłacić za skorzystanie z **RescuePro**, jednakże dla osoby niewymagającej zbyt wiele - foremost w zupełności wystarczy - przypadkowo usunięte cenne zdjęcia z wakacji nie są pod wielkim zagrożeniem 😊

W każdym razie można godnie polecić oba programy.

## Pakiet Adobe

Przeanalizowałem również pliki wyeksportowane za pomocą oprogramowania:

- Adobe Photoshop CC
- Adobe Premiere Pro CC
- Adobe Lightroom CC

I programy te zostawiają po sobie naprawdę sporo metadanych, większość oczywiście potrzebnych i *normalnych*, aczkolwiek jest też parę zastanawiających:

np. dokładną wersję oprogramowania, dokładną historię działania programu, czas zapisania samych metadanych (ochrona integralności), XMP toolkit z datą, ścieżkę do pliku na systemie użytkownika!

zhashowane historia instancji i ID użytkownika i dokumentu (pliku) albo coś, co wygląda jak hash licencji

History Software Agent	:	Adobe Photoshop 21.0 (Windows), Adobe Photoshop 21.0 (Windows), Adobe Photoshop 21.0 (Windows), Adobe Photoshop 21.0 (Windows)
History Changed	:	/, /, /
History Parameters	:	from Image/jpeg to application/vnd.adobe.photoshop, converted from image/jpeg to application/vnd.adobe.photoshop, from application/vnd.adobe.photoshop to image/jpeg, converted from application/vnd.adobe.photoshop to image/jpeg
Create Date	:	2022:12:09 18:32:41+01:00
Metadata Date	:	2022:12:09 18:40:21+01:00
History Action	:	saved, converted, derived, saved, converted, derived, saved

Pantry Instance ID	:	aa49d6df-d998-6aca-e547-2acf00000070
Pantry Document ID	:	f9ceab39-a226-cc6a-9be6-7b5400000043
Pantry Original Document ID	:	xmp.did:b456c9e3-1201-e944-91e0-bf48b38b2ae2
Pantry Duration Value	:	26611200
Pantry Duration Scale	:	5.55555555555556e-006
Pantry Alt Timecode Time Value	:	19:02:30:23
Pantry Alt Timecode Time Format	:	25 fps
Pantry History Action	:	saved
Pantry History Instance ID	:	aa49d6df-d998-6aca-e547-2acf00000070
Pantry History When	:	2022-02-16 19:09:42.03 -00

Moim zdaniem, programy te zostawiają po sobie zbyt dużo metadanych, co może doprowadzić do negatywnych konsekwencji - np. do dopasowania konkretnej osoby (ID instancji) z danym plikiem lub w przypadku próby sfabrykowania podstawowych danych exif (np. daty utworzenia), jeśli osoba nie jest czujna, to nadal idzie wykryć tą prawdziwą datę - dostępną pod metadaną np. *History When*.

I właśnie tak się zdarzyło w historii - Donieckiej Republiki ludowej, kiedy filmik wyszedł niby dnia X i wtedy był aktualny, a tak naprawdę został zarejestrowany dwa dni wcześniej (wykryte za pomocą dokładnej analizy metadanych)

## Podsumowanie

Z każdej z powyższych opcji analizy, udało się coś ciekawego uzyskać (*sandisk\_sd*, *sandisk\_sd\_format*, *lumix\_with\_sd*, *sd\_format\_overwritten*)

Formatowanie kart SD w normalny sposób nie jest bezpieczne. Za pomocą dostępnych narzędzi (zarówno darmowych jak i płatnych) można odzyskać naprawdę wiele plików i metadanych - a tym samym dowiedzieć się sporo o osobie, która z takowego nośnika korzystała. Dlatego sprzedawanie używanych kart SD lub np. pendrive'ów niesie za sobą co najmniej ryzyko skompromitowania prywatnych zdjęć użytkownika, ale również pozyskania cennych informacji o jego zachowaniach, zwyczajach, a nawet lokalizacji (dane geolokalizacyjne exif).

Nie ma znaczenia w jaki sposób karta SD została sformatowana - czy to z poziomu komputera (GUI), czy za pomocą oprogramowania aparatu - **nie jest to bezpieczne**

Można również uzyskać informację z zepsutego nośnika (w tym przypadku skorumpowanego zrzutu pamięci), co może czasem okazać się przydatne.

Moim zdaniem, dostawcy sprzętu fotograficznego powinni w oprogramowaniach urządzeń umożliwiać dokładne formatowanie kart - wraz z wielokrotnym nadpisywaniem - tak, aby uniemożliwić odzyskanie wcześniej zapisanych danych. Oczywiście, takie rozwiązanie nie jest idealne, czasem zdarzy się sytuacja, gdy ktoś formuluje kartę przez przypadek i wtedy program do odzyskiwania danych okazuje się być zbawiennym; a sam proces dodatkowego nadpisywania danych pochłania zasoby oraz przede wszystkim czas. Jednakże, najlepiej aby użytkownik miał możliwość - czy chce sformatować kartę w sposób szybki (ale nie do końca bezpieczny) czy bezpieczny (aczkolwiek wolny).

Z tego co mi wiadomo i udało się znaleźć, to takie rozwiązanie zostało zaprezentowane w latach około 2010 jedynie w firmie **Leica**. Żaden inny wielki producent nie poszedł tą drogą,

a nawet sama Leica od tego odchodzi z uwagi na negatywne opinie użytkowników. Niestety, lata mijają a producenci aparatów nie idą w stronę bezpieczeństwa

Jednakże takowa opcja jest dostępna w systemie Windows - przy resetowania komputera do ustawień fabrycznych

Na koniec dnia, umiejętność wyważenia jednocześnie wygody i bezpieczeństwa (które niestety nie idą w parze) jest naprawdę pożądana.

# Przygotowanie środowiska

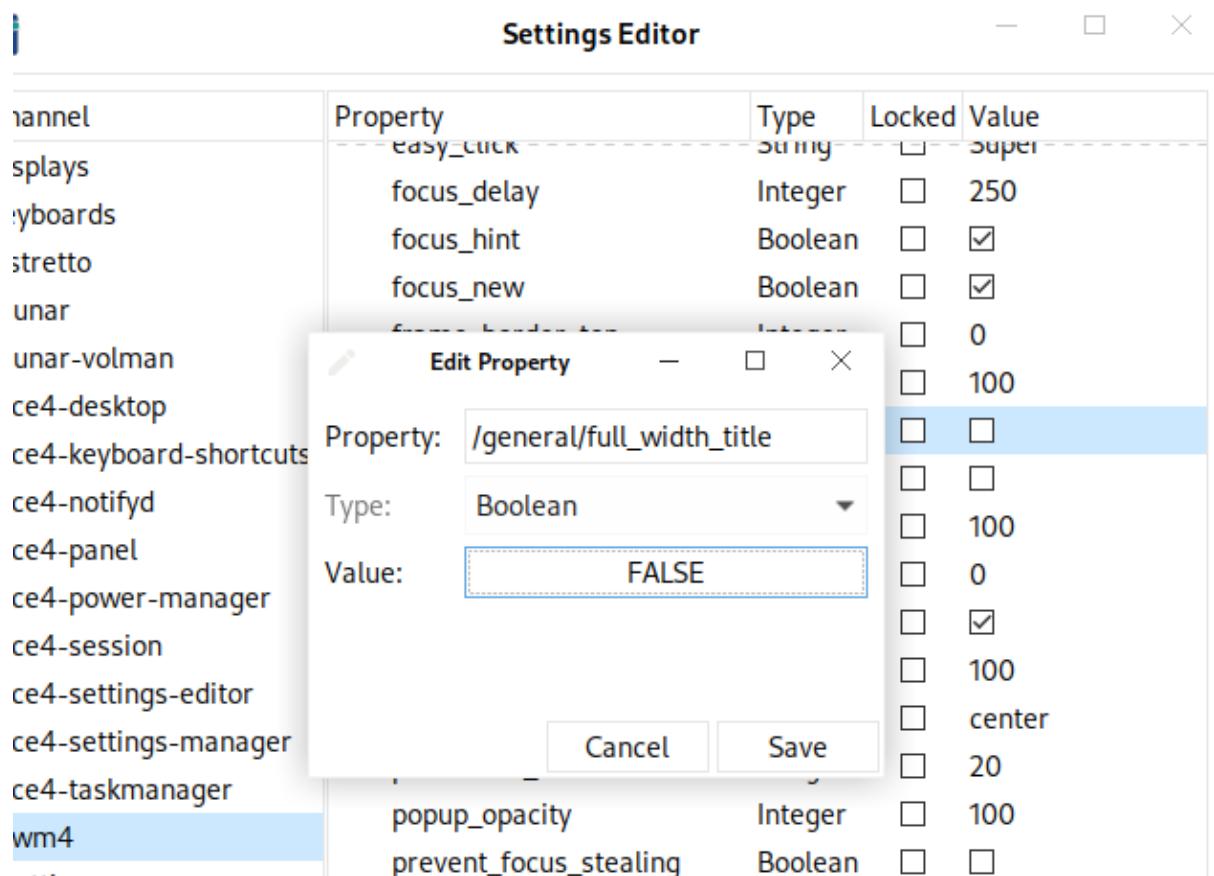
## Narzędzia

Na potrzeby projektu skorzystam z następujących narzędzi:

- Oracle Vm VirtualBox z zainstalowanym systemem Kali Linux/Linux Mint
- obraz systemu Windows użyty od kolegi z kierunku

## Zmiana ustawień systemowych

- zmiana motywu graficznego, tapety, window manager'a
- zmiana paru ustawień w rejestrze (xfwm4: wrap\_cycle, box\_move, zoom\_desktop; xfce4-panel: position-locked, enable-keyboard-shortcuts)

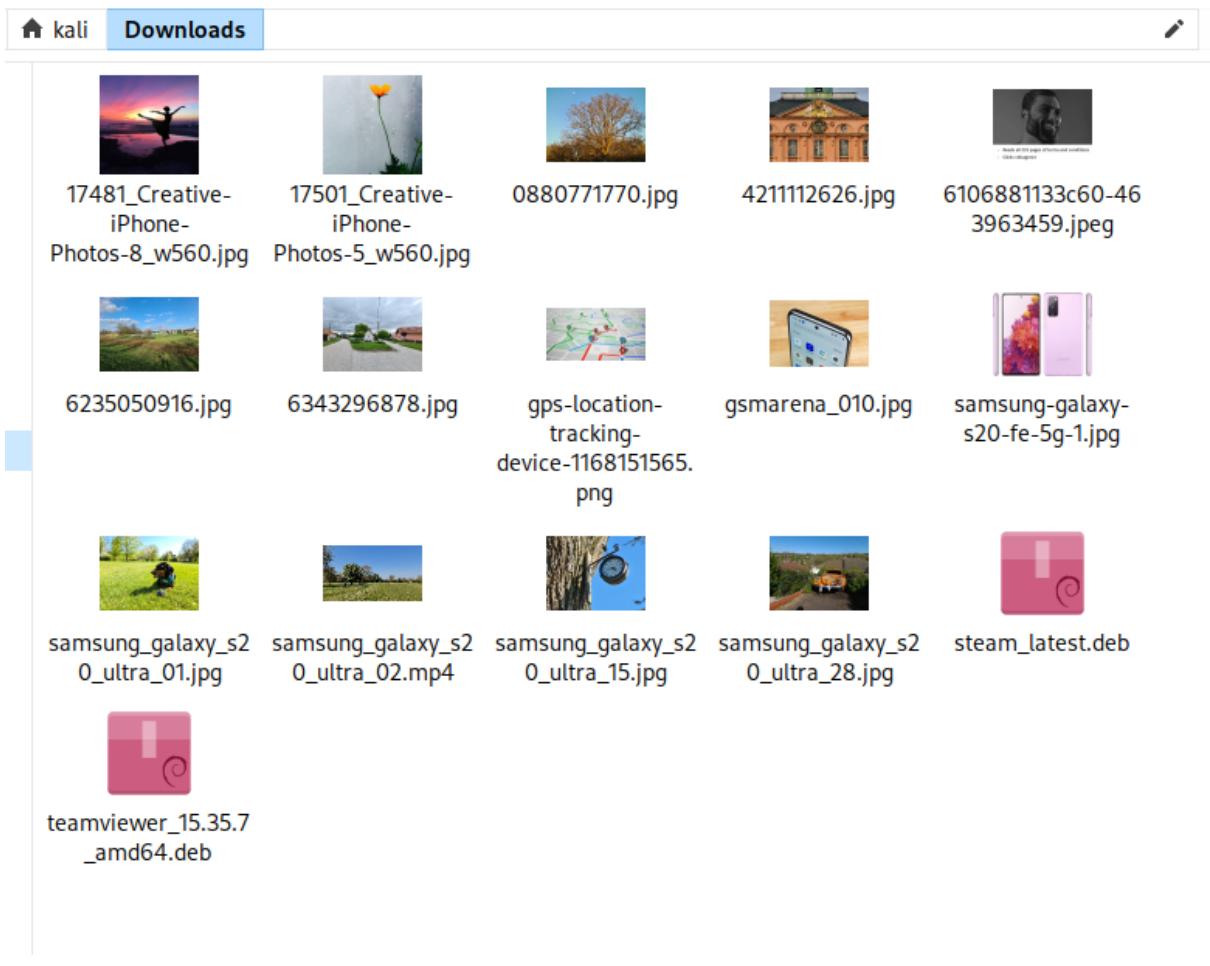


- zmiana ustawień firewalla

```
(kali㉿kali)-[~]
$ sudo ufw enable
Firewall is active and enabled on system startup

(kali㉿kali)-[~]
$ sudo ufw status verbose
Status: active
Logging: on (low)
Default: deny (incoming), allow (outgoing), disabled (routed)
New profiles: skip
```

- instalacja zewnętrznego oprogramowania (steam, vlc, kleopatra, obs, teamviewer)
- pobrania plików graficznych



- oraz utworzyłem parę plików w formatach docx, pdf, txt za pomocą różnych metod (terminal, LibreOffice Writer, Google docs)
- utworzyłem przykładową sesję użytkownika internetu
- wrzuciłem zdjęcia wraz z lokalizacją GPS wykonane telefonem do systemu plików maszyny wirtualnej
- poprzenosiłem, pousuwałem parę plików, próbowałem „nadpisać” miejsca gdzie pliki zostały usunięte za pomocą plików o dużym rozmiarze

**3.**

```
qemu-img: Must specify image file name
[arek@fedora IS] $ qemu-img convert -f vdi kali-linux-2022.3-virtualbox-amd64\ C
lone.vdi -O raw example_image.raw
```

Utworzenie pliku .raw

## Autopsy

Utworzenie nowej sprawy w programie autopsy oraz wstępna konfiguracja

Dodaje domyślne moduły:

## Add Data Source

### Steps

1. Select Host
2. Select Data Source Type
3. Select Data Source
- 4. Configure Ingest**
5. Add Data Source

### Configure Ingest

Run ingest modules on:

All Files, Directories, and Unallocated Space

- | <input checked="" type="checkbox"/> | Recent Activity              |
|-------------------------------------|------------------------------|
| <input checked="" type="checkbox"/> | Hash Lookup                  |
| <input checked="" type="checkbox"/> | File Type Identification     |
| <input checked="" type="checkbox"/> | Extension Mismatch Detector  |
| <input checked="" type="checkbox"/> | Embedded File Extractor      |
| <input checked="" type="checkbox"/> | Picture Analyzer             |
| <input checked="" type="checkbox"/> | Keyword Search               |
| <input checked="" type="checkbox"/> | Email Parser                 |
| <input checked="" type="checkbox"/> | Encryption Detection         |
| <input checked="" type="checkbox"/> | Interesting Files Identifier |
| <input checked="" type="checkbox"/> | Central Repository           |
| <input checked="" type="checkbox"/> | PhotoRec Carver              |
| <input checked="" type="checkbox"/> | Virtual Machine Extractor    |
| <input checked="" type="checkbox"/> | Data Source Integrity        |

Select All

Deselect All

History

## Add Data Source

### Steps

1. Select Host
2. Select Data Source Type
3. Select Data Source
4. Configure Ingest
- 5. Add Data Source**

### Add Data Source

Processing data source and adding it to a local database. File analysis will start when this finishes.

Status  
Adding: /usr/share/exploitdb/exploits/php/webapps/

\*This process may take some time for large data sources.

< Back

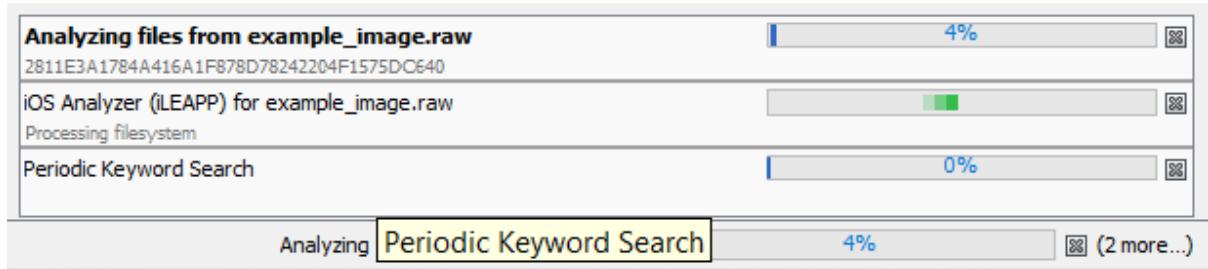
Next >

Finish

Cancel

Help

Warto zauważyć iż program zaraz po otwarciu rozpoczyna skanowanie w poszukiwaniu wszelkiego rodzaju plików



Będzie to dluuuie skanowaanie...

Domyślnie obraz maszyny wirtualnej kąiego zajmuje 80 GB (dzieje się tak, ponieważ system plików był zaznaczony jako dynamiczny, a przecież maszyna i tak musi zgoła znać jakąś dolną granicę - analogia do pamięci VSZ i RSS), dlatego do dalszej analizy w programie autopsy użyję obrazu systemu Windows utworzonego przez kolegę z kierunku.

Na wstępnie pragnę zaznaczyć, iż postaram się zdobyć jak najwięcej informacji o samym użytkowniku i jego aktywnościach, a następnie porównam owe informacje z tym, co faktycznie zrobił kolega, który użyczył mi obrazu systemu.

Już przy pierwszych chwilach jesteśmy witani errorem

Error encountered while calculating the hash value for  
 /Users/Cyberniebezpieczny/AppData/Local/Microsoft/OneDrive/logs/Personal/SyncEngine-2022-12-03.2119.5116.1.odl (Deleted File).

Prawdopodobnie świadczy to o tym, iż plik który powinien był się tam znajdować, został nadpisany, dlatego trzeba mieć na uwadze, że nie wszystkie pliki pójdziesz odzyskać

## Z jakiego systemu korzystał nasz użytkownik?

Po samej strukturze plików łatwo jest stwierdzić, że jest to windows, tylko należy zadać pytanie jaki dokładnie?

Udało mi się odnaleźć plik odpowiadający na to pytanie

Operating System Information	
2022-12-03 20:14:27	Windows 10 Home
Name	DESKTOP-DUAOS4Q
Domain	
Version	Windows_NT
Processor Architecture	AMD64
Temporary Files Directory	%SystemRoot%\TEMP
Source File Path	/img_obrazek.img/vol_vol3/Windows/System32/config/SYSTEM
Artifact ID	-9223372036854775678

## Informacje o samych użytkownikach?

Dlaczego liczba mnoga?

	Ponieważ w C:\windows\system32\config\SAM\Domains\Account\Users\Names możemy znaleźć iż faktycznie było ich więcej niż sam użytkownik o nazwie Cyberniebezpieczny
	Widnieje też <i>Sasiad Zbigniew</i> oraz <i>Hektor</i>

Hex Text Application File Metadata OS Account Data Artifacts Analysis

### Basic Properties

Login:	Sasiad Zbigniew
Full Name:	
Address:	S-1-5-21-2480793805-441931286-2866870202-1002
Type:	
Creation Date:	2022-12-03 20:59:30 CET

### obrazek.img\_1 Host Details

Login Count:	0
Password Hint:	null
Password Settings:	Password does not expire, Password not required
Flag:	Normal user account

Potwierdza to również inna zakładka **OS Accounts**

Gdzie mamy również mamy informacje o tym, ile razy dany użytkownik się zalogował, oraz czy używała hasła

## Urządzenia

### Data Artifacts > USB Device Attached:

W zakładce tej możemy wyczytać iż użytkownik korzystał z jakiegoś urządzenia o nazwie *ROOT\_HUB30*, po lekkim researchu można stwierdzić iż jest to po prostu sterownik intel'a

Source Name	S	C	O	Date/Time	Device Make	Device Model	Device ID	Data Source
SYSTEM			0	2022-12-03 22:20:55 CET		ROOT_HUB30	4&24054718&0&0	obrazek.img
SYSTEM			0	2022-12-03 22:20:55 CET	VirtualBox	USB Tablet	5&12c8f4c0&0&1	obrazek.img

Inną ciekawą rzeczą w tej zakładce jest urządzenie o nazwie *USB Tablet*, którego twórcą był VirtualBox - i na tej podstawie możemy określić iż dołączone urządzenie nie było fizyczne - tylko wirtualne/logiczne. Daje nam to podstawy do przypuszczania, że nasz główny użytkownik korzystał właśnie z oprogramowania do wirtualizacji

## Analiza metadanych EXIF

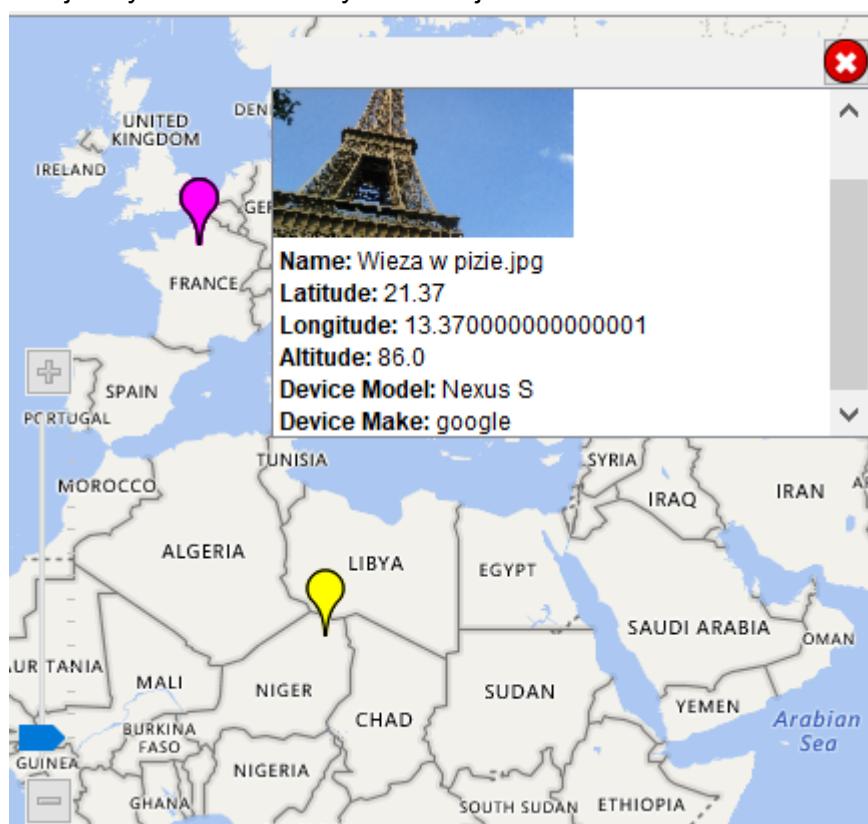
### EXIF Metadata (34)

W podglądzie zakładki widnieje liczba plików 34, w rzeczywistości liczba unikalnych zdjęć to 17.

IMG_20120724_172958.jpg	1	File	Not Notable	2012-07-24 17:29:57 CEST	/img_obrazek.img/vol_vo3/Users/Cyberniebezpieczny/Pict...
IMG_20120724_172958.jpg	1	File	Not Notable	2012-07-24 17:29:57 CEST	/img_obrazek.img/vol_vo3/#Recycle.Bin/5-1-5-21-248079...

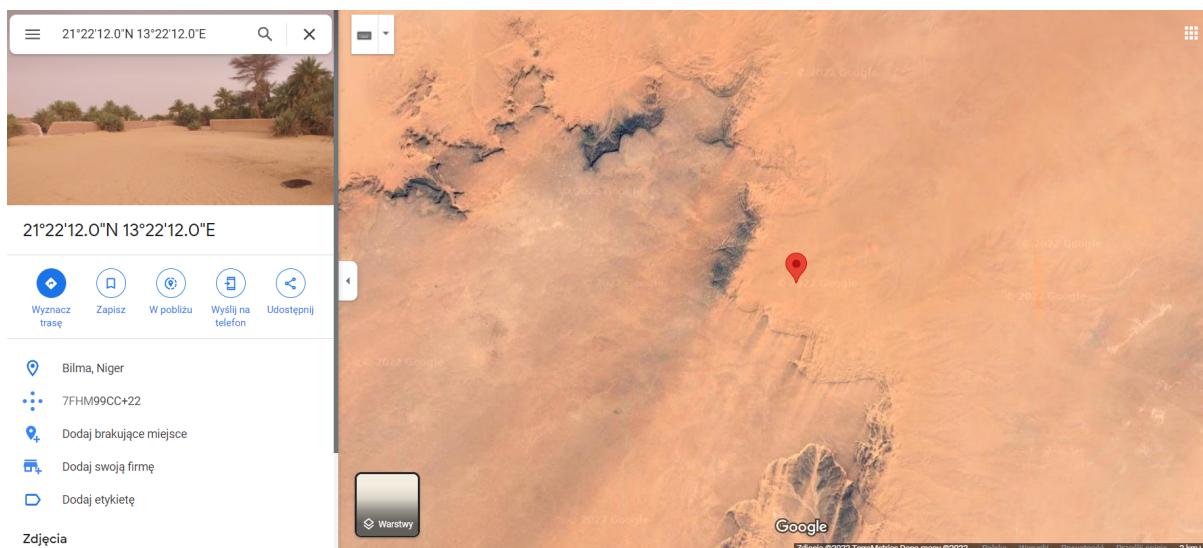
Dzieje się tak, ponieważ użytkownik przechowywał te pliki w folderze User/Pictures, a następnie usunął je do kosza (a mimo wszystko udało się je stamtąd odzyskać)

Przejdzmy teraz do analizy lokalizacji



Udało się odnaleźć 2 znaczaco odległe od siebie miejsca.

- 1) zdjęcie *Wieza w pizie.jpg* gdzie faktycznie jest ukazana wieża Eiffla. Co może zaskakiwać, jest fakt, że znaczniki lokalizacji zostały zmanipulowane (najprawdopodobniej programem typu ExifTools) - wskazują one nie na paryż, a środek pustyni w Nigrze.



- 2) Drugim obrazem jest jakaś świątynia/monument. Po odrobinie analizy OSINT jesteśmy w stanie stwierdzić, że jest to Panteon znajdujący się w Paryżu. Warto zauważać, że znaczniki GPS faktycznie wskazują na to miejsce

## Panteon w Paryżu

4.6 ★★★★☆

Pomnik



Większość zdjęć zrobionych smartfonem Nexus znajduje się w tej samej lokalizacji - Francji, Paryżu

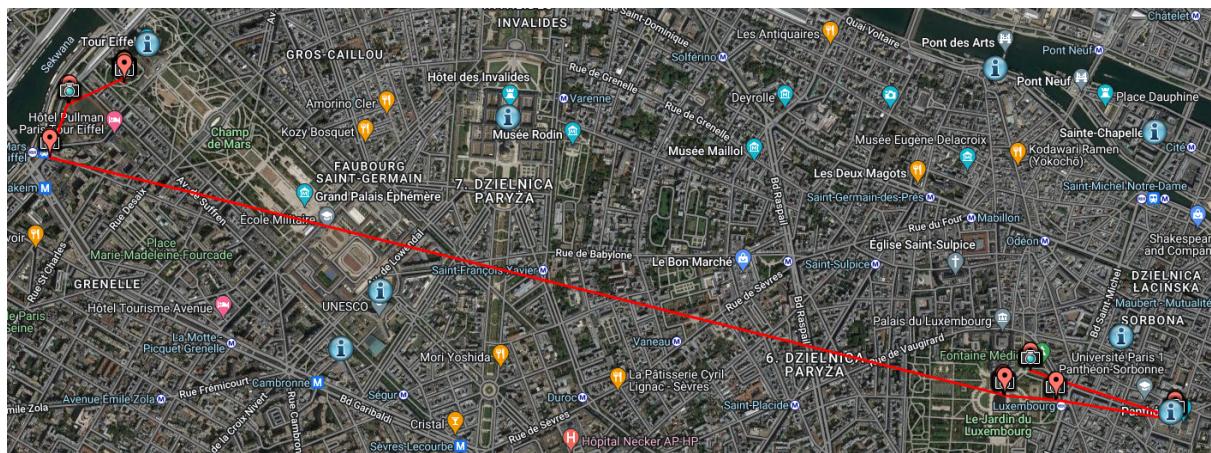
Można przypuścić iż użytkownik był tam na dłużej na jakimś wyjeździe/wakacjach, gdzie zajmował się zwiedzaniem miasta. Przedział datowy to 2012.07.24-26 a więc 3 dni



Co ciekawe, program autopsy nie znalazł nam lokalizacji zdjęcia o metadanych znajdującego się w Anglii

Okazuje się, że jest to tylko tapeta (prawdopodobnie autor fotografii zapomniał usunąć wrażliwych metadanych). Może o tym fakt iż fotografia była w folderze *pobrane* oraz miała inną datę

Korzystając z darmowej aplikacji javowej **GeoTag** oraz analizując czas wykonania zdjęcia oraz same lokalizacje, jesteśmy w stanie utworzyć trasę, po której poruszała się ta osoba



Jak widać, są to jedne z najbardziej charakterystycznych miejsc paryża  
W ramach ciekawostki mogę dodać iż program ten pozwala nam na głębszą analizę typu OSINT - przykładowo zaznaczając, w którą stronę zostało wykonane zdjęcie na podstawie informacji charakterystycznych elementów otoczenia (ikona i), czy cieni

Ciekawostka #2 - **Geoguessr** istnieje aplikacja/gra w której losujemy dowolne miejsce na świecie - i na podstawie przemieszczania się po okolicy za pomocą widoku google maps - musimy odgadnąć dokładną lokalizację. Przydatne mogą się okazać wtedy np. charakterystyczne budynki, język na bilbordach, zabudowa, obecność wody itd.

Może być przydatna w ćwiczeniu umiejętności analizy OSINT'owej oraz zakresu informatyki śledczej

Niestety, cała powyższa aktywność użytkownika Cyberniebezpiecznika (związana z robieniem zdjęć) nie została stworzona przez niego - a pobrana z internetu  
Świadczyć może o tym fakt, że w zakładce Web Downloads są ślady właśnie tych zdjęć

Web Downloads					
Source Name	S	C	O	Path	
IMG_20120725_111015.jpg:Zone.Identifier				/Users/Cyberniebezpieczny/Pictures/luxembourg-parc/IMG...	
IMG_20120725_112345.jpg:Zone.Identifier				/Users/Cyberniebezpieczny/Pictures/luxembourg-parc/IMG...	
IMG_20120725_112417.jpg:Zone.Identifier				/Users/Cyberniebezpieczny/Pictures/luxembourg-parc/IMG...	
IMG_20120725_112847.jpg:Zone.Identifier				/Users/Cyberniebezpieczny/Pictures/luxembourg-parc/IMG...	
IMG_20120725_120424.jpg:Zone.Identifier				/Users/Cyberniebezpieczny/Pictures/luxembourg-parc/IMG...	
IMG_20120724_172958.jpg:Zone.Identifier				/Users/Cyberniebezpieczny/Pictures/museum-d_orsay/IMG...	

Kolejne potwierdzenie - w usuniętych plikach z folderu Downloads było archiwum .zip o świadczące o paczce obrazków wraz z metadanymi gps

File Metadata	OS Account	Data Artifacts	Analysis Results	Context	Annotations	Other
images_gps.zip			2022-12-03 20:52:57 CET	2022-12-03 20:52:57 CET	2022-12-03	
PineTools.com_2022-			2022-12-03 20:58:45 CET	2022-12-03 20:58:45 CET	2022-12-03	
python-3.11.0-amd64			2022-12-03 20:52:57 CET	2022-12-03 20:52:57 CET	2022-12-03	
Hex	Text	Application	File Metadata	OS Account	Data Artifacts	Analysis Results
Page: 1 of 1476	Page	← →	Go to Page: 1	Jump to Offset		
0x0000000000: 50 4B 03 04 14 03 00 00 00 00 64 17 86 51 00 00 PK.....d..Q..						
0x00000010: 00 00 00 00 00 00 00 00 00 00 0B 00 00 00 69 6D ..im.....im						
0x00000020: 61 67 65 73 5F 67 70 73 2F 50 4B 03 04 14 03 00 ages_gps/PK.....						
0x00000030: 00 00 00 64 17 86 51 00 00 00 00 00 00 00 00 00 ...d..Q.....						
0x00000040: 00 00 00 18 00 00 00 69 6D 61 67 65 73 5F 67 70 .....images_gp						
0x00000050: 73 2F 65 69 66 66 65 6C 2D 74 6F 77 65 72 2F 50 s/eiffel-tower/P						
0x00000060: 4B 03 04 14 03 00 00 08 00 64 17 86 51 81 4D D2 K.....d..Q.M.						
0x00000070: 4F 93 2D 13 00 E5 75 13 00 2F 00 00 00 69 6D 61 O.-..u.../...ima						
0x00000080: 67 65 73 5F 67 70 73 2F 65 69 66 66 65 6C 2D 74 ges_gps/eiffel-t						
0x00000090: 6F 77 65 72 2F 49 4D 47 5F 32 30 31 32 30 37 32 over/IMG_2012072						
0x000000a0: 35 5F 31 38 32 30 34 34 2E 6A 70 67 EC 5C 77 50 5_182044.jpg.\wP						
0x000000b0: 53 5F 16 7E 88 80 22 8A A2 A8 14 81 1F A0 28 08 S_.~..".....(.						
0x000000c0: 48 EF A0 22 4D 29 4A AF 22 2D 40 30 74 88 94 10 H.. "M) J."-@0t...						

## Historia wyszukiwarki

**Co możemy dalej powiedzieć o użytkowniku?** Pokaż mi historię wyszukiwarki a powiem ci kim jesteś

Source Name	S	C	O	Domain	Text	Program Name	Date Accessed	Old Source
History				bing.com	how to become a frontend developer	Microsoft Edge	2022-12-03 11:19:41 CET	obrazek.img
History				bing.com	how to become a frontend developer	Microsoft Edge	2022-12-03 11:19:41 CET	obrazek.img
History				bing.com	how to center div horizontally	Microsoft Edge	2022-12-03 11:20:13 CET	obrazek.img
History				bing.com	how to center div horizontally	Microsoft Edge	2022-12-03 11:20:13 CET	obrazek.img
History				bing.com	bieszczady wypasanie owiec	Microsoft Edge	2022-12-03 11:20:22 CET	obrazek.img
History				bing.com	is my password secure	Microsoft Edge	2022-12-03 20:44:40 CET	obrazek.img
History				bing.com	is my password secure	Microsoft Edge	2022-12-03 20:44:40 CET	obrazek.img
History				bing.com	is password admin secure	Microsoft Edge	2022-12-03 20:44:50 CET	obrazek.img
History				bing.com	why do i need password	Microsoft Edge	2022-12-03 20:44:59 CET	obrazek.img
History				bing.com	bardzo interesuje się gotowaniem makaronu	Microsoft Edge	2022-12-03 21:01:59 CET	obrazek.img

Na tej podstawie można najprawdopodobniej stwierdzić że użytkownik

- 1) korzystał z wyszukiwarki bing i przeglądarki Microsoft Edge
- 2) bardzo interesuje się gotowaniem makaronu
- 3) sprawdzał, czy jego hasło jest bezpieczne
- 4) szukał jak wypasa się owce w bieszczadach
- 5) pracuje albo chce pracować we frontendzie

## Web History

Znaleziony plik WebCacheV01.dat posiada ślady próby logowania do usług microsoft'u, z zapytania w przeglądarce jesteśmy w stanie wydobyć id użytkownika

WebCacheV01.dat - Properties	
Properties	
Source Name	WebCacheV01.dat
S	(No Property Editor)
C	NO_COMMENT
O	0
URL	https://login.live.com/oauth20_authorize.srf?client_id=00000000480728C5&scop...
Date Accessed	2022-12-03 21:21:33 CET

Również w WebHistory udało mi się znaleźć ślady o pliku *notAScript.jpg.py* (ślady znalezione dzięki plikom z rozszerzeniem .lnk), ciężko było go pierwotnie znaleźć w

systemie plików - aczkolwiek udało się. Plik udomowił się w katalogu /System32/bg-BG/DoNotDelete/

```
notascript.jpg.py import socket
if __name__ == "__main__":
    while True:
        for p in range(1, 4098):
            sock = socket.socket(socket.AF_INET, socket.SOCK_DGRAM, socket.IPPROTO_UDP)
            sock.setsockopt(socket.SOL_SOCKET, socket.SO_BROADCAST, 1)
            sock.sendto(b"Completely harmless, no need to inspect", ("255.255.255.255", p))
```

Jest to niegroźny malware?

Najprawdopodobniej nasz nieświadomy użytkownik pobrał taki plik z internetu

Z tego co udało mi się wyczytać, pliki z rozszerzeniem .lnk są automatycznie tworzone przez system windows w trakcie otwarcia pliku/programu. Może się to okazać kluczowa funkcjonalność dla późniejszej analizy obrazu systemu (lub analizy szkodliwego oprogramowania)

## Inne ciekawe pliki

Z zakładka File Views > File Types > By MIME Type > znajdziemy plik, który pierwotnie znajdował się w katalogu Cyberniebezpieczny/Music

Name	Size	Modified time	Create time	Access time	Created time	Size	Allocated	Filepath	Unknown	Last write
Cat meowing.mp3	0	2022-12-03 20:51:37 CET	2022-12-03 20:51:59 CET	2022-12-03 22:25:24 CET	2022-12-03 20:51:37 CET	89998	Allocated	Allocated	unknown	/img_obra
AchievementUnlocked.mp3	0	2019-12-07 10:55:38 CET	2022-12-03 20:12:38 CET	2022-12-03 20:12:38 CET	2019-12-07 10:55:38 CET	12268	Allocated	Allocated	unknown	/img_obra

Zdecydowanie jest to ślad związany z życiem prywatnym użytkownika

Parę wydobytych zrzutów ekranu:

Name	Size	Modified time	Create time	Access time	Created time	Size	Allocated	Filepath	Unknown	Last write
(6E73DABE-345E-445D-A8E8-0B6D3283E2FE).png	0	2022-12-03 20:43:38 CET	2022-12-03 20:43:38 CET	2022-12-03 20:43:38 CET	2022-12-03 20:43:38 CET	55632	Allocated	Allocated	unknown	/img_obra

## Usuniętych programów

Name	Size	Modified time	Create time	Access time	Created time	Size	Allocated	Filepath	Unknown	Last write
PineTools.com_2022-12-03_1th57m25s.zip	0	2022-12-03 20:58:45 CET	2022-12-03 20:58:45 CET	2022-12-03 20:58:22 CET	2022-12-03 20:57:26 CET	524301922	Unallocated	Unallocated	unknown	/img_obra
libwireshark.dll	0	2022-10-26 20:00:38 CEST	2022-12-03 11:22:19 CET							
msedge.dll	0	2021-08-05 08:46:16 CEST	2022-12-03 11:19:25 CET							
MicrosoftEdge_X64_107.0.1418.62.exe	0	2022-12-03 20:36:49 CET	2022-12-03 20:36:54 CET							

oraz plików tekstowych i archiwum .gz

```
no preferred path found.lnk desktop-duaos4q
?hh
7C:\Windows\System32\bg-BG\DoNotDelete\notAScript.jpg.py
*_VL
desktop-duaos4q
;C:\Users\Cyberniebezpieczny\Videos\Super funny cats.mp4.vbs
desktop-duaos4q
3C:\Users\Cyberniebezpieczny\Downloads\README.md.txt
desktop-duaos4q
=C:\Users\Cyberniebezpieczny\Desktop\tamtenplikniebylfajny.txt
desktop-duaos4q
2C:\Users\Cyberniebezpieczny\Desktop\fajny plik.txt
```

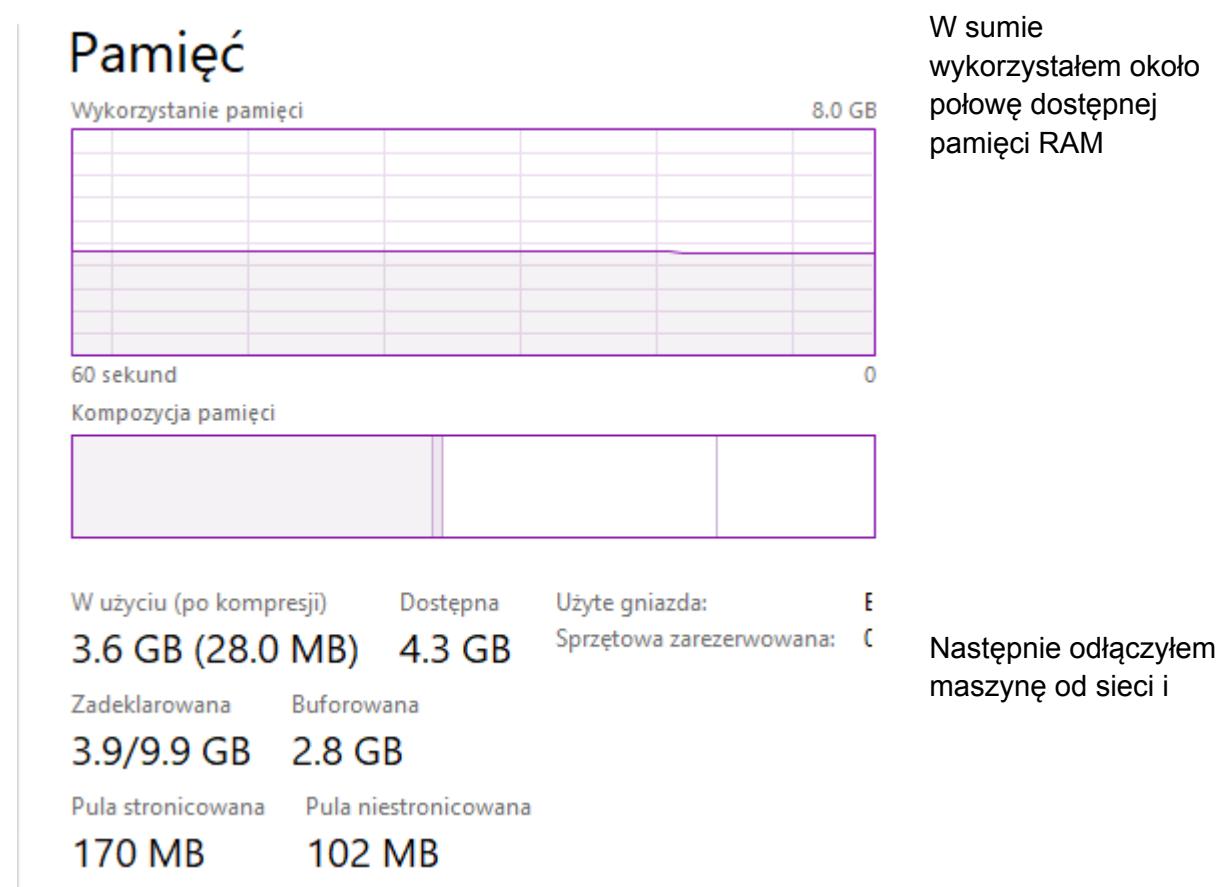
# Zrzut pamięci RAM oraz jej analiza

## Narzędzia

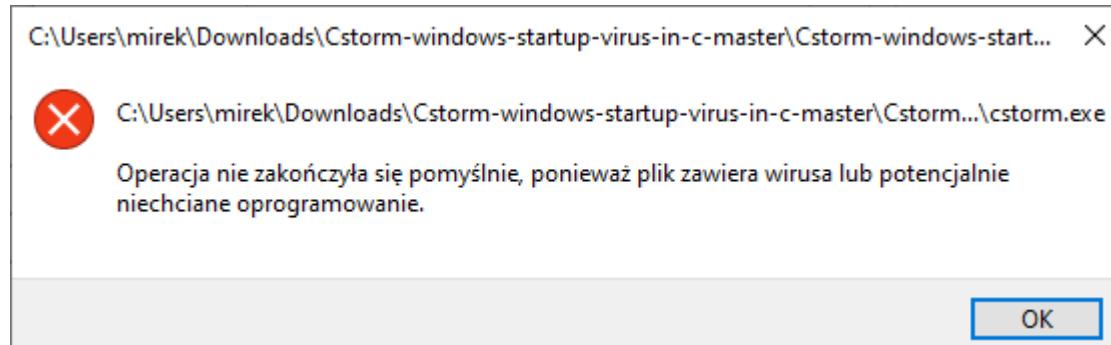
Na potrzeby powyższego zadania skorzystam z maszyny wirtualnej (na VirtualBox'ie) + programu **Volatility** do analizy

## Przygotowanie maszyny wirtualnej

Najpierw stworzyłem zwykłą sesję użytkownika - otwartych parę kart przeglądarki, uruchomiony odtwarzacz zdjęć i filmów, eksplorator plików, procesy w terminalach, uruchomione pobieranie w tle



odpaliłem [malware](#) - backdoora + keyloggera



Najpierw tworzę zrzut pamięci RAM uruchomionej maszyny wirtualnej za pomocą debuggera programu Virtualbox - musiałem to zrobić naprawdę szybko, ponieważ od razu po odpaleniu malwer'a system postanowił się wyłączyć, dlatego jednym z celów tego zadanie będzie:

Jak wiele można odzyskać/wyczytać z analizy pamięci RAM z wyłączającego się systemu?

## Volatility

windows.info

Podstawowe informacje komendy parametru **windows.info** (używam symboli windowsowych):

```
[arek@fedora volatility3-1.0.0] $ python vol.py -f ~/astudia/is/projekt/ram/windows_ram.bin windows.info
```

```

variable      Value

Kernel Base    0xf80069800000


```

Możemy się stąd dowiedzieć trochę o samym systemie i sprzęcie - przykładowo, że jest 64 bitowy, używa procesora intel, jaką ma aktualną godzinę, wersję systemu operacyjnego (Windows 10)

## pslist

Przejdźmy teraz do nieco ciekawszej komendy - listy wszystkich procesów **pslist**

Volatility 3 Framework 1.0.0											
PID	PPID	ImageFileName	Offset(V)	Threads	Handles	SessionId	Wow64	CreateTime	ExitTime	File output	
4	0	System	0xbff810d086080	152	-	N/A	False	2022-12-10 14:42:41.000000	N/A	Disabled	
124	4	Registry	0xbff810d1d6040	4	-	N/A	False	2022-12-10 14:42:38.000000	N/A	Disabled	
376	4	smss.exe	0xbff811021040	2	-	N/A	False	2022-12-10 14:42:41.000000	N/A	Disabled	
480	464	csrss.exe	0xbff811344a080	11	-	0	False	2022-12-10 14:42:45.000000	N/A	Disabled	
556	464	wininit.exe	0xbff8113c9d080	5	-	0	False	2022-12-10 14:42:45.000000	N/A	Disabled	
576	548	csrss.exe	0xbff8113ca080	14	-	1	False	2022-12-10 14:42:45.000000	N/A	Disabled	
656	548	winlogon.exe	0xbff8113ccc080	6	-	1	False	2022-12-10 14:42:45.000000	N/A	Disabled	
700	556	services.exe	0xbff8113cd2080	7	-	0	False	2022-12-10 14:42:45.000000	N/A	Disabled	
720	556	lsass.exe	0xbff8113cda080	10	-	0	False	2022-12-10 14:42:45.000000	N/A	Disabled	
836	700	svchost.exe	0xbff8113d6f200	40	-	0	False	2022-12-10 14:42:45.000000	N/A	Disabled	
856	556	fontdrvhost.ex	0xbff8113d76140	5	-	0	False	2022-12-10 14:42:45.000000	N/A	Disabled	
864	656	fontdrvhost.ex	0xbff8113d74140	5	-	1	False	2022-12-10 14:42:45.000000	N/A	Disabled	
960	700	svchost.exe	0xbff811444d280	17	-	0	False	2022-12-10 14:42:45.000000	N/A	Disabled	
1008	700	svchost.exe	0xbff811446c240	5	-	0	False	2022-12-10 14:42:45.000000	N/A	Disabled	

Przykładowy zrzut ekranu (nie uda się zamieścić całości wyjścia tej komendy, ponieważ jest zbyt długie). Można znaleźć tutaj takie informacje jak: numer procesu i jego rodzica, nazwa procesu, miejsce w pamięci (zapisane szesnastkowo), liczbę wątków, data utworzenia

Sprawdźmy ile różnych procesów było uruchomionych:

```

[arek@fedora ram] $ awk '{print $3}' pslist.txt | sort | uniq | wc -l
42

```

42 - z czego znacząca większość to procesy systemowe

```
[arek@fedora ram] $ awk '{print $3}' pslist.txt | sort | uniq

ApplicationFra
audiogd.exe
csrss.exe
ctfmon.exe
dwm.exe
explorer.exe
fontdrvhost.ex
Framework
ImageFileName
LogonUI.exe
lsass.exe
MemCompression
msedge.exe
MsMpEng.exe
Music.UI.exe
Registry
RuntimeBroker.
SearchFilterHo
SearchIndexer.
SearchProtocol
SecurityHealth
services.exe
SgrmBroker.exe
sihost.exe
smss.exe
spoolsv.exe
svchost.exe
System
taskhostw.exe
Taskmgr.exe
TextInputHost.
TiWorker.exe
TrustedInstall
userinit.exe
VBoxService.ex
VBoxTray.exe
wininit.exe
winlogon.exe
WinStore.App.e
wlrmrdr.exe
WmiPrvSE.exe
```

Są to między innymi: odtwarzacz muzyki, przeglądarka explorer i edge, eksplorator plików, menadżer zadań, sklep microsoft oraz wiele więcej *tajemniczych* procesów

## VBoxService.exe VBoxTray.exe

Dlaczego są tam procesy virtualboxa?

Proces [VBoxTray](#) oraz [VBoxService](#) tak naprawdę nie ma nic wspólnego z programem VirtualBox, jest to program do zarządzania innymi programami, obsługą klawiatury oraz myszki

pstree

Sprawdźmy drzewo procesów **pstree**:

PID	PPID	ImageFileName	Offset(V)	Threads	Handles	SessionId	Wow64	CreateTime	ExitTime
4	0	System	0xbff81178d7080	152	-	N/A	False	2022-12-10 14:42:41.000000	N/A
* 376	4	smss.exe	0xbff81178d7080	2	-	N/A	False	2022-12-10 14:42:41.000000	N/A
* 124	4	Registry	0xbff81178d7080	4	-	N/A	False	2022-12-10 14:42:38.000000	N/A
* 1928	4	MemCompression	0xbff81178d7080	22	-	N/A	False	2022-12-10 14:42:46.000000	N/A
480	464	csrss.exe	0xbff81178d7080	11	-	0	False	2022-12-10 14:42:45.000000	N/A
556	464	wininit.exe	0xbff81178d7080	5	-	0	False	2022-12-10 14:42:45.000000	N/A
* 720	556	lsass.exe	0xbff81178d7080	10	-	0	False	2022-12-10 14:42:45.000000	N/A
* 856	556	fontdrvhost.exe	0xbff81178d7080	5	-	0	False	2022-12-10 14:42:45.000000	N/A
* 700	556	services.exe	0xbff81178d7080	7	-	0	False	2022-12-10 14:42:45.000000	N/A
** 6152	700	svchost.exe	0xbff81178d7080	11	-	0	False	2022-12-10 14:42:51.000000	N/A
** 1528	700	svchost.exe	0xbff81178d7080	5	-	0	False	2022-12-10 14:42:46.000000	N/A
** 5136	700	svchost.exe	0xbff81178d7080	5	-	0	False	2022-12-10 14:52:09.000000	N/A
** 7700	700	svchost.exe	0xbff81178d7080	5	-	0	False	2022-12-10 14:44:29.000000	N/A
** 3624	700	svchost.exe	0xbff81178d7080	6	-	0	False	2022-12-10 14:42:47.000000	N/A
** 3120	700	MsMpEng.exe	0xbff81178d7080	29	-	0	False	2022-12-10 14:52:09.000000	N/A
** 7220	700	svchost.exe	0xbff81178d7080	1	-	0	False	2022-12-10 15:00:58.000000	N/A
** 4156	700	svchost.exe	0xbff81178d7080	3	-	0	False	2022-12-10 14:42:47.000000	N/A
*** 4240	4156	ctfmon.exe	0xbff81178d7080	12	-	1	False	2022-12-10 14:42:47.000000	N/A
** 1088	700	svchost.exe	0xbff81178d7080	2	-	0	False	2022-12-10 14:42:46.000000	N/A
** 1100	700	svchost.exe	0xbff81178d7080	6	-	0	False	2022-12-10 14:42:46.000000	N/A

```
~ 15:06:53.000000 WmiPrvSE.exe
~ 15:07:10.000000 svchost.exe
~ 15:07:55.000000 TiWorker.exe
~ 15:07:55.000000 TrustedInstall
~ 15:07:55.000000 WmiPrvSE.exe
~ 15:07:57.000000 wlrmldr.exe
~ 15:08:02.000000 LogonUI.exe
~ N/A 0xbff81178d7080
```

Sortując procesy po czasie można dokładnie określić co i w jakiej kolejności użytkownik robił.

Jesteśmy w stanie prześledzić proces instalacji malware

**TiWorker** - proces odpowiedzialny za instalacje programu > **TrustedInstall** - nadanie uprawnień instalacji

Dwa powyższe procesy są to dzieci procesu systemowego - **svchost.exe** oraz **services.exe**. Można to łatwo zobaczyć dzięki strukturze drzewa komendy **pstree**

		svchost.exe	0xbf811
**	836 700	svchost.exe	0xbf811
***	8896	836	RuntimeBroker.
***	6368	836	RuntimeBroker.
***	9560	836	RuntimeBroker.
***	7300	836	RuntimeBroker.
***	4536	836	WmiPrvSE.exe
***	4488	836	WinStore.App.e
***	968 836	RuntimeBroker.	0xbf811
***	5164	836	RuntimeBroker.
***	7692	836	TextInputHost.
***	7088	836	RuntimeBroker.
***	7568	836	ApplicationFra
***	4272	836	RuntimeBroker.
***	3928	836	WmiPrvSE.exe
***	2772	836	TiWorker.exe

**wlrmdr.exe** najprawdopodobniej jest to właśnie interesujący nasz malware, ponieważ od razu po nim nastąpiło wylogowanie się systemu windows (przypadłość, którą wcześniej opisałem) oraz jest to pojedynczy proces. Poza tym, udało mi się znaleźć [informacje](#) iż czasem pod taką nazwą pliku mogą kryć się groźne programy. Jest to wstępne założenie, które być może później uda się potwierdzić/obalić

Również dzięki takiemu ułożeniu graficznemu można zobaczyć jak uruchamiają się karty przeglądarki

***	6916	4524	msedge.exe
****	5988	6916	msedge.exe
****	8484	6916	msedge.exe
****	8772	6916	msedge.exe
****	6980	6916	msedge.exe
****	8964	6916	msedge.exe
****	3716	6916	msedge.exe
****	10052	6916	msedge.exe
****	3596	6916	msedge.exe
****	7724	6916	msedge.exe
****	6252	6916	msedge.exe
****	3564	6916	msedge.exe
****	5712	6916	msedge.exe
****	8364	6916	msedge.exe
****	7952	6916	msedge.exe
****	564	6916	msedge.exe

malfind

Przejdzmy do wbudowanego narzędzia do analizy malware - **malfind**

```
[Iarek@fedora volatility3-1.0.0] $ python vol.py -f ~/astudia/is/projekt/ram/windows_ram.bin windows.malfind
Volatility 3 Framework 1.0.0
```

```
3120 MsMpEng.exe 0x1a51d860000 0x1a51d861fff VadS PAGE_EXECUTE_READWRITE 2 1 Disabled
55 48 8d 2c 24 48 83 ec UH.,SH..
20 48 8b 01 48 8b 49 08 .H..H.I.
ff d0 48 8d 65 00 5d c3 ..H.e.].
cc cc cc cc cc cc cc cc ..... .
cc cc cc cc cc cc cc cc ..... .
cc cc cc cc cc cc cc cc ..... .
cc cc cc cc cc cc cc cc ..... .
cc cc cc cc cc cc cc cc ..... .
0x1a51d860000: push rbp
0x1a51d860001: lea rbp, [rsp]
0x1a51d860005: sub rsp, 0x20
0x1a51d860009: mov rax, qword ptr [rcx]
0x1a51d86000c: mov rcx, qword ptr [rcx + 8]
0x1a51d860010: call rax
0x1a51d860012: lea rsp, [rbp]
0x1a51d860016: pop rbp
```

Wyszukuje on procesów które mają wszystkie 3 uprawnienia - read, write, execute - ponieważ działający malware z reguły takowe posiada

```
[arek@fedora volatility3-1.0.0] $ cat ~/astudia/is/projekt/ram/malfind.txt | grep .exe
3120 MsMpEng.exe 0x1a519260000 0x1a51936cff VadS PAGE_EXECUTE_READWRITE 269 1 Disabled
3120 MsMpEng.exe 0x1a51a610000 0x1a51a71cff VadS PAGE_EXECUTE_READWRITE 269 1 Disabled
3120 MsMpEng.exe 0x1a51aaa0000 0x1a51abacfff VadS PAGE_EXECUTE_READWRITE 269 1 Disabled
3120 MsMpEng.exe 0x1a51c180000 0x1a51c180fff VadS PAGE_EXECUTE_READWRITE 1 1 Disabled
3120 MsMpEng.exe 0x1a51d850000 0x1a51d850fff VadS PAGE_EXECUTE_READWRITE 1 1 Disabled
3120 MsMpEng.exe 0x1a51d840000 0x1a51d840fff VadS PAGE_EXECUTE_READWRITE 1 1 Disabled
3120 MsMpEng.exe 0x1a51d860000 0x1a51d861fff VadS PAGE_EXECUTE_READWRITE 2 1 Disabled
3120 MsMpEng.exe 0x1a51d8b0000 0x1a51d8b1fff VadS PAGE_EXECUTE_READWRITE 2 1 Disabled
3120 MsMpEng.exe 0x1a51d8c0000 0x1a51d8c0fff VadS PAGE_EXECUTE_READWRITE 1 1 Disabled
3120 MsMpEng.exe 0x1a51eb40000 0x1a51eb40fff VadS PAGE_EXECUTE_READWRITE 1 1 Disabled
3120 MsMpEng.exe 0x1a51eb90000 0x1a51eb90fff VadS PAGE_EXECUTE_READWRITE 1 1 Disabled
3120 MsMpEng.exe 0x1a51ebd0000 0x1a51ebd5fff VadS PAGE_EXECUTE_READWRITE 6 1 Disabled
3120 MsMpEng.exe 0x1a51f0c0000 0x1a51f1bffff VadS PAGE_EXECUTE_READWRITE 256 1 Disabled
3120 MsMpEng.exe 0x1a520290000 0x1a52048ffff VadS PAGE_EXECUTE_READWRITE 512 1 Disabled
```

O dziwo, jedyne co udało się znaleźć to procesy o nazwie - *MsMpEng* - rozwinięcie tego skrótu to Microsoft Malware Protection Engine, a więc jest to Windows Defender  
W takim razie, w tej kategorii nie udało się odnaleźć potencjalnego procesu z malwarem

## filescan

Sprawdźmy teraz ogólne pliki, które udało się znaleźć

```
[arek@fedora volatility3-1.0.0] $ python3 vol.py -f ~/astudia/is/projekt/ram/windows_ram.bin filescan > ~/astudia/is/projekt/ram/filescan.txt
```

Jest ich naprawdę sporo - około 9 tysięcy

```
[arek@fedora ram] $ wc -l filescan.txt
9098 filescan.txt
[arek@fedora ram] $
```

```
[arek@fedora ram] $ cat filescan.txt | head -15
Volatility 3 Framework 1.0.0

Offset Name Size
0xb810da0e00 \Windows\System32\CatRoot\{F750E6C3-38EE-11D1-85E5-00C04FC295EE}\Microsoft-Windows-Client-Desktop-Required-Package094-31bf3856ad364e35-amd64--10.0.19041.200
0xb810da0e210 \Windows\System32\drivers\Wif.sys 216
0xb810da0e4f0 \Windows\System32\drivers\ndiscap.sys 216
0xb810da0e7d0 \Windows\System32\CatRoot\{F750E6C3-38EE-11D1-85E5-00C04FC295EE}\Microsoft-Windows-Client-Features-Package0316-31bf3856ad364e35-amd64--10.0.19041.1949.cat2
16
0xb810da0e0c20 \Windows\System32\drivers\BoxSF.sys 216
0xb810da0f1e0 \Windows\System32\drivers\nethios.sys 216
0xb810da0f350 \Windows\System32\drivers\Wid.sys 216
0xb810da0f910 \Windows\System32\drivers\wdmhv.sys 216
0xb810da31210 \Windows\System32\drivers\dam.sys 216
0xb810da31380 \Windows\System32\drivers\npsvcvtrig.sys 216
0xb810da31660 \Windows\System32\drivers\mssmbios.sys 216
```

Można tutaj znaleźć przykładowo sterowniki oraz pliki systemowe oraz generalną strukturę plików użytkownika

Znając nazwę wirusa, sprawdzam, co udało się programowi Volatility odzyskać

```
[arek@fedora ram] $ cat filescan.txt | grep Cstorm
0xb8117118df0 \Users\mirek\Downloads\cstorm-windows-startup-virus-in-c-master\cstorm-windows-startup-virus-in-c-master\Binaries\cstorm Remover.exe.sig 216
0xb811711bbe0 \Users\mirek\Downloads\cstorm-windows-startup-virus-in-c-master\cstorm-windows-startup-virus-in-c-master\Custom Variants\README.md 216
0xb811712bafe0 \Users\mirek\Downloads\cstorm-windows-startup-virus-in-c-master\cstorm-windows-startup-virus-in-c-master\Custom Variants\Mstools Remover.c.sig 216
0xb8117129980 \Users\mirek\Downloads\cstorm-windows-startup-virus-in-c-master\cstorm-windows-startup-virus-in-c-master\Source Code\cstorm Remover.c.sig 216
0xb8117129790 \Users\mirek\Downloads\cstorm-windows-startup-virus-in-c-master 216
0xb811712b540 \Users\mirek\Downloads\cstorm-windows-startup-virus-in-c-master\cstorm-windows-startup-virus-in-c-master\Source Code\cstorm.c.sig 216
0xb811712d2f0 \Users\mirek\Downloads\cstorm-windows-startup-virus-in-c-master\cstorm-windows-startup-virus-in-c-master\Source Code\cstorm Remover.exe 216
0xb811712d610 \Users\mirek\Downloads\cstorm-windows-startup-virus-in-c-master\cstorm-windows-startup-virus-in-c-master\Binaries 216
0xb8117133ba0 \Users\mirek\Downloads\cstorm-windows-startup-virus-in-c-master\cstorm-windows-startup-virus-in-c-master\Binaries\cstorm Remover.exe 216
0xb8117134370 \Users\mirek\Downloads\cstorm-windows-startup-virus-in-c-master\cstorm-windows-startup-virus-in-c-master\Binaries\cstorm Remover.exe.sig 216
0xb8117134820 \Users\mirek\Downloads\cstorm-windows-startup-virus-in-c-master\cstorm-windows-startup-virus-in-c-master\Binaries\cstorm Remover.exe 216
0xb8117134b40 \Users\mirek\Downloads\cstorm-windows-startup-virus-in-c-master\cstorm-windows-startup-virus-in-c-master\Binaries\cstorm Remover.exe.sig 216
0xb8117136a00 \Users\mirek\Downloads\cstorm-windows-startup-virus-in-c-master\cstorm-windows-startup-virus-in-c-master\Binaries\cstorm Remover.exe 216
0xb81171759a170 \Users\mirek\Downloads\cstorm-windows-startup-virus-in-c-master\cstorm-windows-startup-virus-in-c-master\Binaries\cstorm Remover.exe 216
0xb81171759a050 \Users\mirek\Downloads\cstorm-windows-startup-virus-in-c-master\cstorm-windows-startup-virus-in-c-master\Binaries\cstorm Remover.exe.sig 216
0xb81171759b5c0 \Users\mirek\Downloads\cstorm-windows-startup-virus-in-c-master\cstorm-windows-startup-virus-in-c-master\Custom Variants\Mstools Remover.c.sig 216
0xb8117175ad540 \Users\mirek\Downloads\cstorm-windows-startup-virus-in-c-master\cstorm-windows-startup-virus-in-c-master\Custom Variants\Mstools.c.sig 216
0xb8117175adea0 \Users\mirek\Downloads\cstorm-windows-startup-virus-in-c-master\cstorm-windows-startup-virus-in-c-master\Custom Variants\README.md 216
0xb8117175ae1c0 \Users\mirek\Downloads\cstorm-windows-startup-virus-in-c-master\cstorm-windows-startup-virus-in-c-master\Custom Variants\README.sig 216
0xb8117175ae800 \Users\mirek\Downloads\cstorm-windows-startup-virus-in-c-master\cstorm-windows-startup-virus-in-c-master\Source Code\README.md 216
0xb81175aae40 \Users\mirek\Downloads\cstorm-windows-startup-virus-in-c-master\cstorm-windows-startup-virus-in-c-master\Source Code\cstorm Remover.c 216
```

Możemy tutaj odnaleźć parę plików wykonywalnych z rozszerzeniem .exe  
Udało się odnaleźć wcześniej wspomniany plik, podejrzany o bycie malwarem

```
[arek@fedora ram] $ cat filescan.txt | grep wlrmrdr
0xbf8115d4d2d0  \Windows\System32\pl-PL\wlrmrdr.exe.mui  216
0xbf811644f160  \Windows\System32\wlrmrdr.exe      216
```

Jest on udomowiony w katalogu \System32, co znacząco zmniejsza szanse na bycie plikiem złośliwym - a tylko procesem systemowym

<https://fileinfo.com/extension/mui>

```
[arek@fedora ram] $ cat filescan.txt | grep cstorm.exe
0xb8117134b40  \Users\mirek\Downloads\Cstorm-windows-startup-virus-in-c-master\Cstorm-windows-startup-virus-in-c-master\Binaries\cstorm.exe      216
0xb811782fb90  \Users\mirek\Downloads\Cstorm-windows-startup-virus-in-c-master\Cstorm-windows-startup-virus-in-c-master\Binaries\cstorm.exe.sig      216
```

W tym plik instalacyjny złośliwego oprogramowania + jego sygnatura

### Sygnatura pliku

Przejdzmy zatem do tejże sygnatury pliku

```
Volatility 3 Framework 1.0.0

Cache   FileObject       FileName        Result
DataSectionObject    0xbf811782fb90  cstorm.exe.sig  Error dumping file
```

Niestety nie udało się odzyskać pliku

### hivelist

```
[arek@fedora volatility3-1.0.0] $ python3 vol.py -f ~/astudia/is/projekt/ram/windows_ram.bin hivelist
Volatility 3 Framework 1.0.0
Progress: 100.00          PDB scanning finished
Offset  FileFullPath      File output

0x8783b7a64000  Disabled
0x8783b7a56000  \REGISTRY\MACHINE\SYSTEM      Disabled
0x8783b7aec000  \REGISTRY\MACHINE\HARDWARE    Disabled
0x8783b9357000  \SystemRoot\System32\Config\SAM Disabled
0x8783b9359000  \SystemRoot\System32\Config\SECURITY  Disabled
0x8783b930d000  \SystemRoot\System32\Config\DEFAULT   Disabled
0x8783b930f000  \SystemRoot\System32\Config\SOFTWARE   Disabled
0x8783bb19e000  \Device\HarddiskVolume1\Boot\BCD     Disabled
0x8783bb417000  \??\C:\Windows\ServiceProfiles\NetworkService\NTUSER.DAT      Disabled
0x8783bb575000  \SystemRoot\System32\Config\BBI Disabled
0x8783bb577000  \??\C:\Windows\ServiceProfiles\LocalService\NTUSER.DAT  Disabled
0x8783bc25000  \??\C:\Users\mirek\ntuser.dat      Disabled
0x8783bcada000  \??\C:\Users\mirek\AppData\Local\Microsoft\Windows\UsrClass.datDisabled
0x8783bd2e9000  \??\C:\Windows\AppCompat\Programs\Amcache.hve  Disabled
0x8783bda76000  \??\C:\Windows\ServiceProfiles\NetworkService\AppData\Local\Microsoft\Windows\DeliveryOptimization\State\dosvcState.dat Dis-
abled
0x8783be483000  \??\C:\ProgramData\Microsoft\Windows\AppRepository\Packages\MicrosoftWindows.Client.CBS_120.2212.4180.0_x64_cw5n1h2txyewy\A-
ctivationStore.dat Disabled
0x8783be4e4000  \??\C:\Users\mirek\AppData\Local\Packages\MicrosoftWindows.Client.CBS_cw5n1h2txyewy\Settings\settings.dat      Disabled
```

Komenda ta pozwala na wydobycie logów z rejestru systemu. Można tu znaleźć wiele interesujących ustawień i konfiguracji.

Jest tu między innymi plik odpowiedzialny za ustawienia użytkownika - a w nim hash hasła

Potrzebny będzie adres komórki pamięci, gdzie znajduje się interesujący plik - w tym przypadku 0x8783bc25000

Niestety komendzie nie udało się nic znaleźć. Może być to związane z tym, że nie istnieją wskaźniki tego pliku, lub są uszkodzone

```
[arek@fedora volatility3-1.0.0] $ python3 vol.py -f ~/astudia/is/projekt/ram/windows_ram.bin windows.dumpfiles --virtaddr 0x8783bc25000
Volatility 3 Framework 1.0.0
Progress: 100.00          PDB scanning finished
Cache   FileObject      FileName       Result
```

Tak samo nie działa komenda **dumpfiles**

```
[arek@fedora volatility3-1.0.0] $ python3 vol.py -f ~/astudia/is/projekt/ram/windows_ram.bin windows.dumpfiles --virtaddr 0xbff811644f160
Volatility 3 Framework 1.0.0
Progress: 100.00          PDB scanning finished
Cache   FileObject      FileName       Result

ImageSectionObject 0xbff811644f160 wlrmdr.exe Error dumping file
```

Myślę, że niedziałający charakter poszczególnych komend ma związek z tym, że zrzut pamięci RAM stworzyłem w trakcie wyłączania się systemu

**netscan**

sprawdźmy, czy złośliwy proces próbował się połączyć z siecią (według działania jego kodu powinien) za pomocą komendy **windows.netscan**

```
[arek@fedora volatility3-1.0.0] $ python3 vol.py -f ~/astudia/is/projekt/ram/windows_ram.bin windows.netscan > ~/astudia/is/projekt/ram/netscan.txt
[arek@fedora volatility3-1.0.0] $ B scanning finished
```

Warto zauważyć, że powyższa komenda wykonywała się naprawdę długo

Volatility 3 Framework 1.0.0										
Offset	Proto	LocalAddr	LocalPort	ForeignAddr	ForeignPort	State	PID	Owner	Created	
0xbff810d076cb0	TCPv4	0.0.0.0	445	0.0.0.0 0	LISTENING	4	System	2022-12-10 14:42:46.000000		
0xbff810d076cb0	TCPv6	::	445	:: 0	LISTENING	4	System	2022-12-10 14:42:46.000000		
0xbff810d0c0010	TCPv4	-	50091	51.104.136.2	443	CLOSED	8716	Cortana.exe	2022-12-10 14:50:12.000000	
0xbff81111675d0	TCPv4	0.0.0.0	49664	0.0.0.0 0	LISTENING	720	lsass.exe	2022-12-10 14:42:45.000000		
0xbff8111167730	TCPv4	0.0.0.0	49664	0.0.0.0 0	LISTENING	720	lsass.exe	2022-12-10 14:42:45.000000		
0xbff8111167730	TCPv6	::	49664	:: 0	LISTENING	720	lsass.exe	2022-12-10 14:42:45.000000		
0xbff8111167e10	TCPv4	0.0.0.0	135	0.0.0.0 0	LISTENING	960	svchost.exe	2022-12-10 14:42:45.000000		
0xbff81111680d0	TCPv4	0.0.0.0	135	0.0.0.0 0	LISTENING	960	svchost.exe	2022-12-10 14:42:45.000000		
0xbff81111680d0	TCPv6	::	135	:: 0	LISTENING	960	svchost.exe	2022-12-10 14:42:45.000000		
0xbff8111168230	TCPv4	0.0.0.0	49665	0.0.0.0 0	LISTENING	556	wininit.exe	2022-12-10 14:42:45.000000		
0xbff8111168390	TCPv4	0.0.0.0	49665	0.0.0.0 0	LISTENING	556	wininit.exe	2022-12-10 14:42:45.000000		
0xbff8111168390	TCPv6	::	49665	:: 0	LISTENING	556	wininit.exe	2022-12-10 14:42:45.000000		
0xbff8111168910	TCPv4	0.0.0.0	49666	0.0.0.0 0	LISTENING	1100	svchost.exe	2022-12-10 14:42:46.000000		
0xbff8112b051b0	TCPv4	0.0.0.0	49668	0.0.0.0 0	LISTENING	2656	spoolsv.exe	2022-12-10 14:42:46.000000		
0xbff8112b051b0	TCPv6	::	49668	:: 0	LISTENING	2656	spoolsv.exe	2022-12-10 14:42:46.000000		
0xbff8112b05310	TCPv4	0.0.0.0	49670	0.0.0.0 0	LISTENING	700	services.exe	2022-12-10 14:42:46.000000		

Przykładowy output

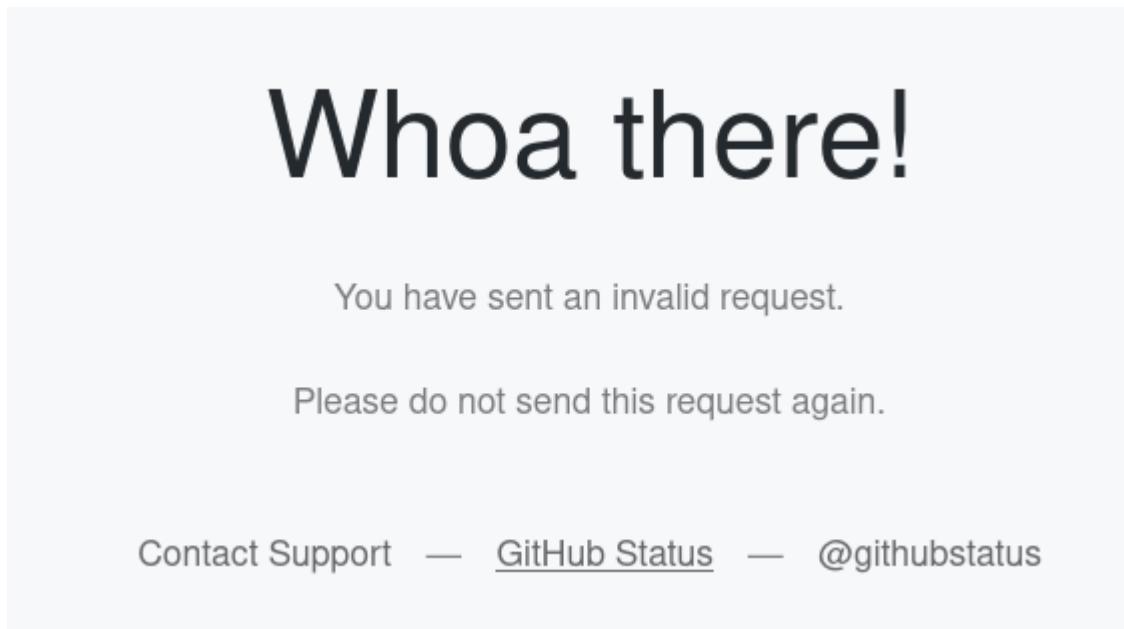
Są tutaj dostępne parametry takie jak: miejsce w pamięci, protokół sieciowy, adres źródła i docelowy, port, data wykonania oraz informacje o procesie.

Oczywiście, jest dużo informacji o samych procesach systemowych Windows'a - svchost, cortana, wininit

Na podstawie powyższego output'u można stwierdzić z jakimi serwerami (stronami) próbował połączyć się użytkownik za pomocą przeglądarki

0xbff8115277a50	TCPv4	-	50355	140.82.121.9	443	CLOSED	5988	msedge.exe	2022-12-10 14:59:03.000000
0xbff81154894e0	TCPv4	-	50365	13.107.21.200	443	CLOSED	4440	SystemSettings	2022-12-10 15:01:02.000000
0xbff8115e61010	TCPv4	-	50367	212.191.241.11	443	CLOSED	5988	msedge.exe	2022-12-10 15:02:23.000000
0xbff8115ff4010	TCPv4	-	50271	23.48.1.104	443	CLOSED	5988	msedge.exe	2022-12-10 14:58:02.000000

Przeanalizowałem parę dostępnych adresów i udało mi się ustalić, że użytkownik wysyłał zapytania na adres ip serwerów GitHuba (co jest prawdą)



Na podstawie samych adresów można stwierdzić, skąd pochodzi użytkownik (o ile nie używa serwera proxy/vpn'a)

**212.191.241.11**

The public IP address **212.191.241.11** is located in *Poland*. It is assigned to the ISP *Institute of Bioorganic Chemistry Polish Academy o.* The address belongs to ASN 8501 which is delegated to *Institute of Bioorganic Chemistry Polish Academy of Science, Poznan Supercomputing and Network*.

W tym przypadku zapytania przechodziły przez polskie serwery

```
[arek@fedora ram] $ cat netscan.txt | grep wlrmrdr
[arek@fedora ram] $
```

Niestety nie udało się znaleźć

próby połączenia tego procesu z siecią (problemem może być to że nie zdążył się on wykonać - albo nie nastąpiło ustanowienie sesji z powodu odłączenia maszyny wirtualnej od internetu)

## Podsumowanie

Łącząc wszystkie powyższe fakty, dochodzę do wniosku, że faktycznie złośliwa część kodu programu, który został podejrzewany o bycie malware'm, nie została wykonana poza samym zainstalowaniem, jakakolwiek część działania programu nie została zainicjalizowana

Wykorzystaniu narzędzia ExifTool do wyciągnięcia metadanych z plików graficznych oraz przedstawienia metod odzyskiwania straconych danych.

analiza odzyskiwania plików obrazu systemu maszyny wirtualnej

foremost

rozpoczynam skanowanie za pomocą komendy *foremost windows*

```
[arek@fedora projekt] $ foremost windows
```

Powyższy program układają output w sposób zorganizowany, dzieląc na podfoldery w zależności od rozszerzenia



audit.txt



avi



bmp



dll



exe



gif



htm



jpg



mov



mp4



ole



png



wav



wmv



xlsx



zip

### 33043 FILES EXTRACTED

```
jpg:= 457
gif:= 685
bmp:= 2126
wmv:= 1
mov:= 52
mp4:= 10
rif:= 303
htm:= 480
ole:= 193
zip:= 56
exe:= 7891
png:= 20789
```

Udało się znaleźć trochę plików w różnych rozszerzeniach

W poniższej analizie zajmę się głównie plikami graficznymi - jpg, mov, mp4, png.

Na starcie warto zaznaczyć iż plików jest tak dużo, ponieważ zeskanowany został cały obraz systemu Windows - a w nim jest dużo plików graficznych potrzebnych, aby oprogramowanie działało sprawnie (ikony, animacje itp.)

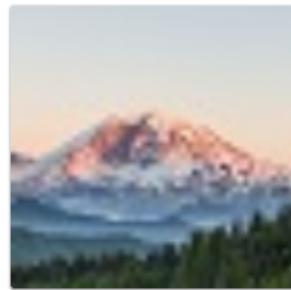
## JPG

Łącznie Udało się znaleźć 457 plików. Jest to całkiem spora ilość, dzięki której jest możliwość analizy indywidualnej sesji użytkownika

Przykładowe odnalezione zdjęcia:



03194784.jpg



03194864.jpg



03194912.jpg



03195096.jpg



03195352.jpg



03195792.jpg

28414840.jpg



28414840.jpg

28415144.jpg



28415144.jpg

28415800.jpg



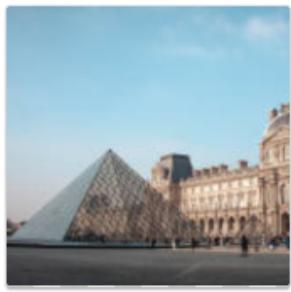
28415800.jpg



28417624.jpg



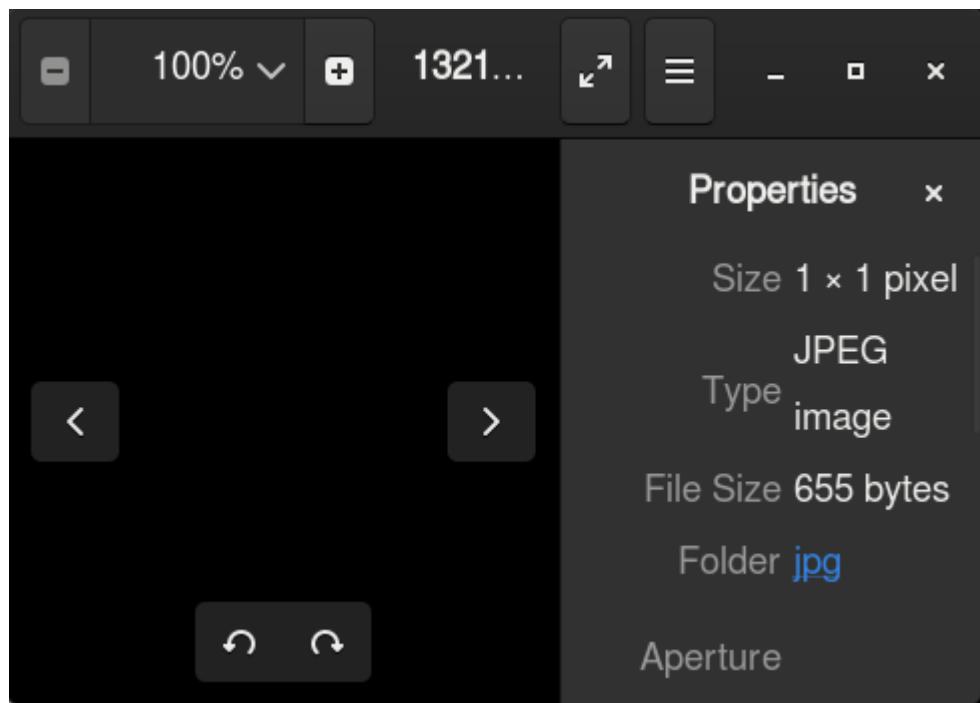
28417656.jpg



28417720.jpg

Na pierwszy rzut oka zdjęcia pochodzą z:

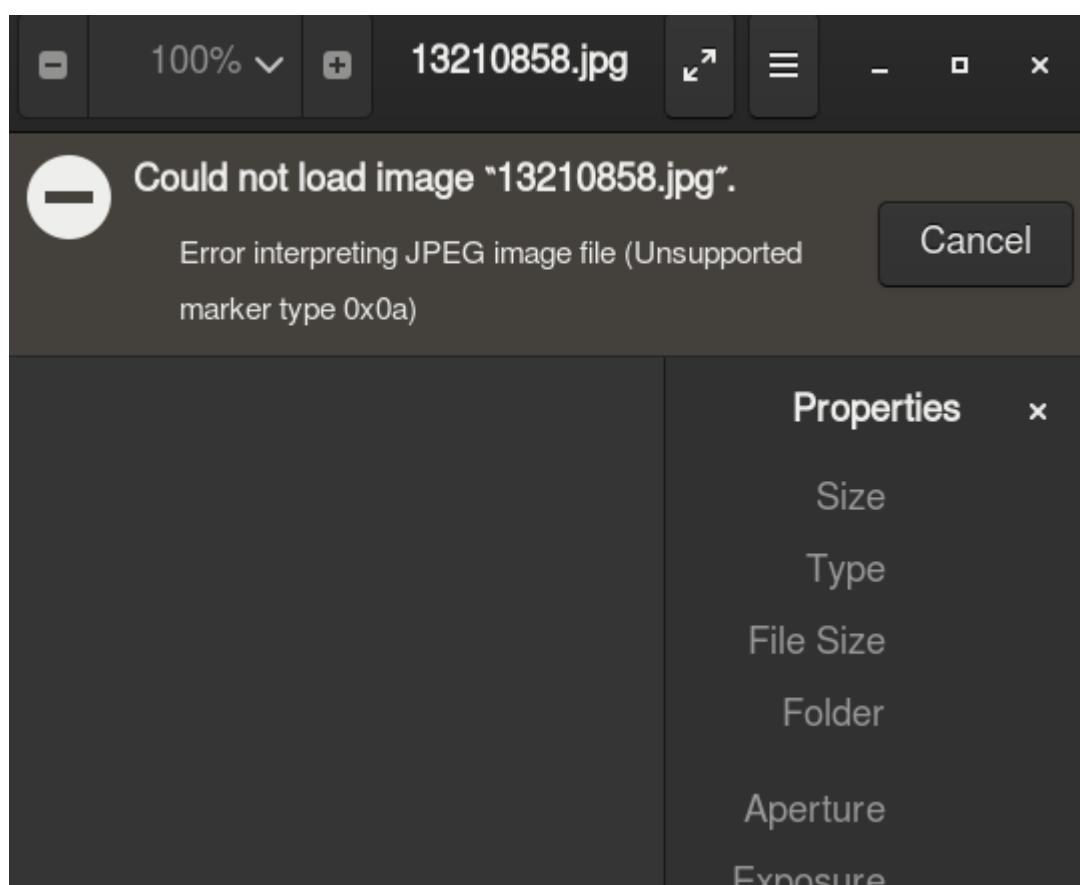
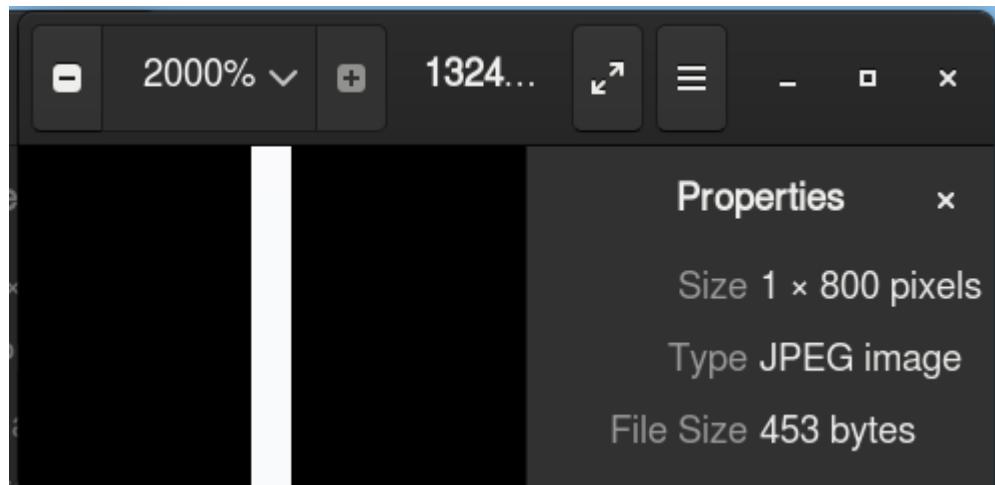
- 1) systemu windows
- 2) artykułów
- 3) indywidualnych preferencji użytkownika
- 4) galerii tapet



Co dziwne, udało się znaleźć parę plików o rozmiarze 1 x 1 pixel. Trochę nie widzę sensu, dlaczego

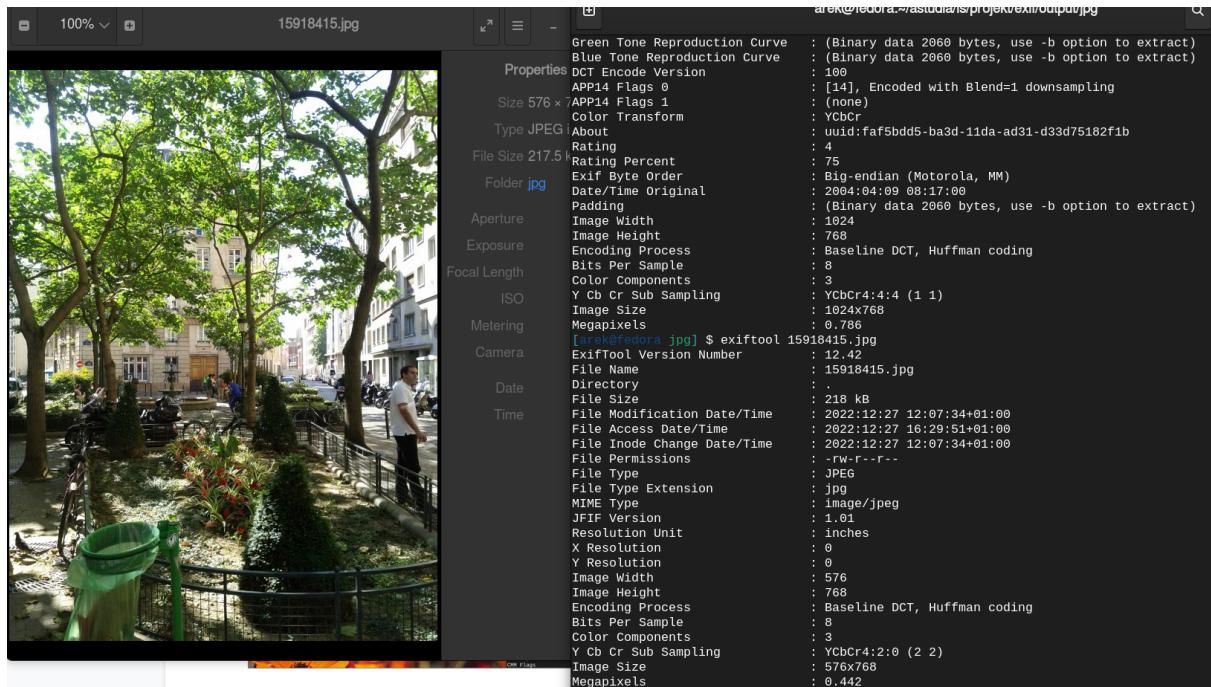
system operacyjny miałby przetrzymywać takie pliki. Prawdopodobnie jest to obrazek pobrany automatycznie w wyniku śledzenia strony internetowej/maila - tak zwany *pixel tracking* lub *web beacon*

Poniższy obraz o rozmiarze 1 x 800 pixeli, również jest trochę dziwny, aczkolwiek jestem w stanie wyobrazić sobie jego zastosowanie w systemie



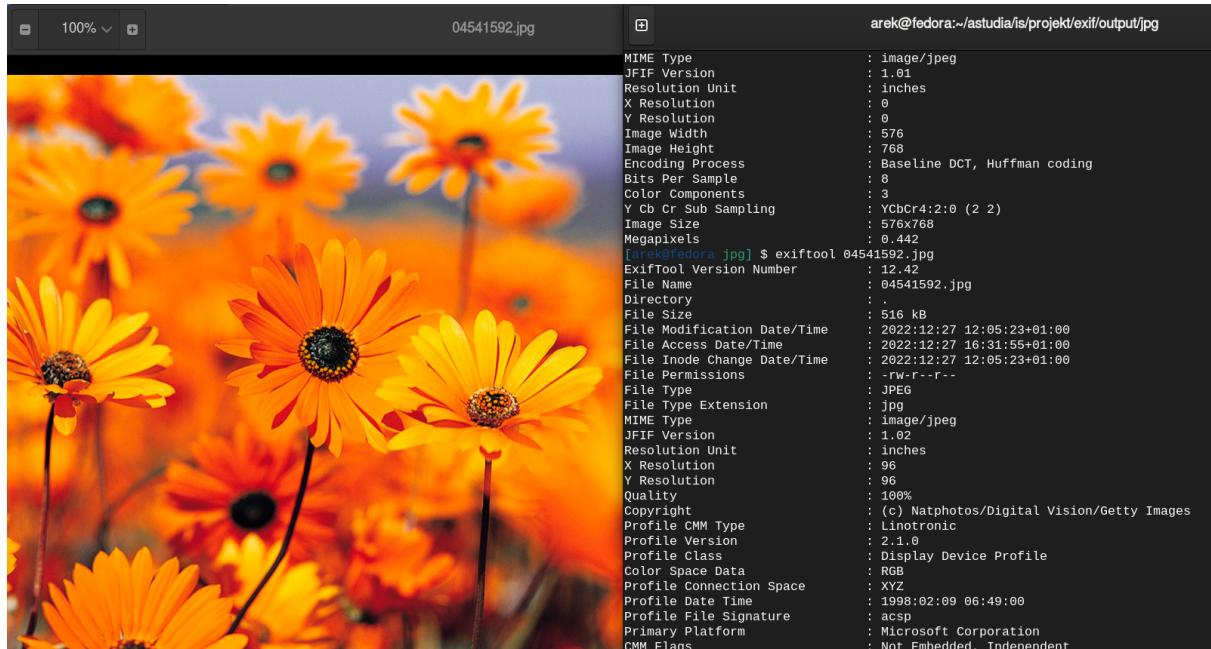
Jest też parę uszkodzonych plików

Posortowałem odnalezione pliki według rozmiaru - ponieważ zdjęcia zrobione przez użytkownika powinny mieć większy rozmiar niż przykładowo ikony systemowe



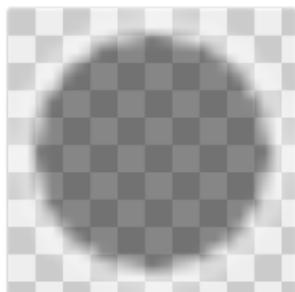
Niestety, owych zdjęć jest naprawdę mało, a w żadnym z nich nie zostały zachowane dane exif

Jedynie udało się odzyskać trochę danych z podstawowej tapety systemu Windows



## PNG

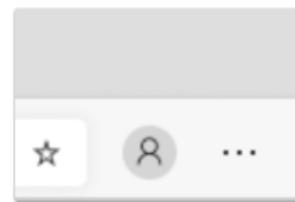
Przykładowe pliki:



28398872\_1.png



28398872\_2.png



28402134.png



28402141.png



28402256.png



28402257.png



28402271.png



28402317.png



28416216.png

Tutaj będziemy mieli do czynienia ze znacznie większą ilością plików systemowych



17596440.png



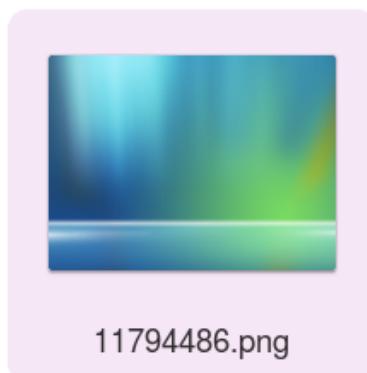
11590146.png



17564104.png



10990656.png



11794486.png



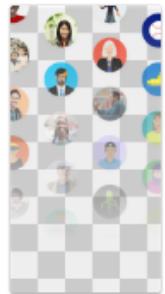
14808900.png



14776729.png



02740176.png

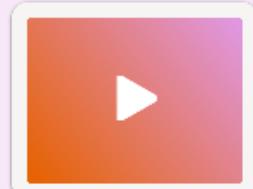


14775309.png

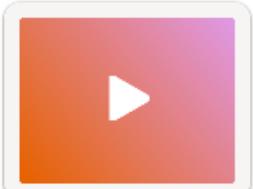
Podobnie jak w przypadku plików z rozszerzeniem .JPG - nie udało się odnaleźć praktycznie żadnych danych exif, nawet w plikach wyglądających na zdjęcia użytkownika

## MOV

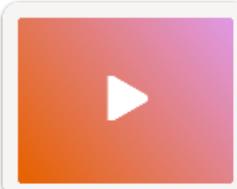
---



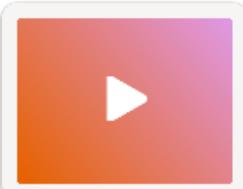
02407288.mov



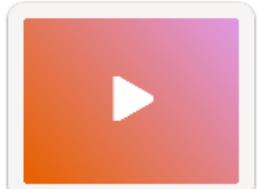
02405016.mov



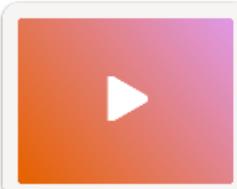
02404576.mov



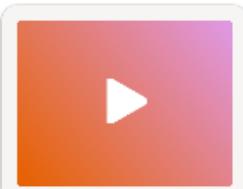
02403312.mov



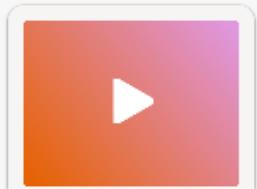
02402992.mov



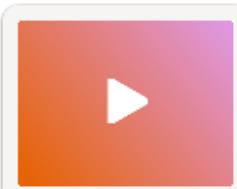
02404064.mov



02404320.mov



02249432.mov



02410016.mov

Udało się odnaleźć aż 52 elementy, jednakże kompletnie wszystkie są uszkodzone - brak miniaturek, video się nie odtwarza

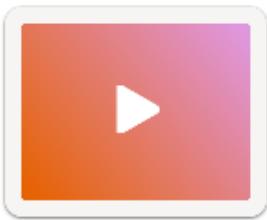
Natomiast zachowało się trochę metadanych - m.in. dostawca, data utworzenia, software

```
Track Volume : 100.00%
Matrix Structure : 1 0 0 0 1 0 0 0 1
Media Header Version : 0
Media Create Date : 2015:06:16 09:59:30
Media Modify Date : 2015:06:16 09:59:30
Media Time Scale : 44100
Media Duration : 0.70 s
Media Language Code : und
Balance : 0
Audio Format : mp4a
Audio Channels : 2
Audio Bits Per Sample : 16
Audio Sample Rate : 44100
Handler Type : Metadata
Handler Vendor ID : Apple
Title : Skype_Shutter_Master_v2_11JUN15_mono
Compilation : No
Play Gap : Insert Gap
Beats Per Minute : 0
Encoder : iTunes 12.0.1.26
iTunSMPB : 0 840 26B 6D55 0 0 0 0 0 0 0 0 0
```

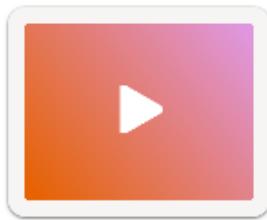
Choć wnioskując po tytule - *Skype\_Shutter\_Master\_v2\_11JUN15\_mono* - są to video pochodzące z domyślne dostępnego w systemie Windows programu do rozmów **Skype**

## MP4

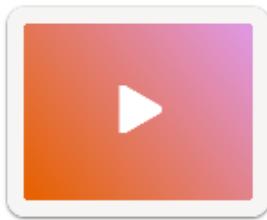
Odnalezione pliki nie posiadają wcale obrazu - jest tylko czarny ekran.  
Z drugiej strony dźwięk w nich został zachowany. Jednakże większość dźwięków tych plików brzmi jak nie z tej ziemi, dodatkowo są przesterowane



02402992.mp4



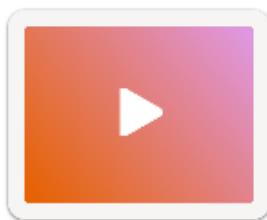
02403312.mp4



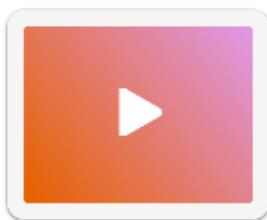
02404064.mp4



02404320.mp4



02404576.mp4



02405016.mp4

Jednakże udało mi się odnaleźć 2 filmiki, w których występuje dzwonek programu skype (choć też z przesterami), jest to o tyle dziwne, ponieważ jest to w rozszerzeniu .mp4, czyli typowo filmowym a nie dźwiękowym

```
[arek@fedora mp4] $ md5sum 02404576.mp4
0f0aecfc4b7b09b2a7538871c3ff6eed6 02404576.mp4
```

Sprawdziłem wartość hash powyższego pliku

File distributed by Microsoft

b373812d62c84787f78695741ef0a9033f66bcbe6daca35345b6fc5911fb9e90  
Skype\_Call\_Ringing.m4a

204.72 KB | 2022-06-08 09:19:37 UTC | 6 months ago

mp4 | known-distributor | legit

Community Score: 0 / 56

DETECTION DETAILS COMMUNITY

**Basic Properties**

MD5	0f0aecfc4b7b09b2a7538871c3ff6eed6
SHA-1	e9880df53cd6f5bc3ae85fa6d6c97e22d4c04a
SHA-256	b373812d62c84787f78695741ef0a9033f66bcbe6daca35345b6fc5911fb9e90
SSDEEP	3072:nbIKUYnqXMqmWSIKIUay+IZJKjbOTKbrBxQuf/D9jVhXUL:nkSqjmNurBJKbjxmKVJEL
TSLH	T1CA24122B50AA3DE2F64AD2B152D930939B110F7B1182B2E5E19E449FE32C1FF376C156
File type	MP4
Magic	ISO Media, MPEG v4 system, iTunes AAC-LC
TrID	AAC Audio in MP4 container (50%)   ISO base media container (33.3%)   Adobe PhotoShop Brush (16.6%)
File size	204.72 KB (209631 bytes)

I faktycznie jest to plik należący do Microsoftu - służący jako dzwonek

Większość metadanych plików została zachowana

```
Track Modify Date : 2017:04:26 21:07:31
Track ID : 1
Track Duration : 0:00:30
Track Layer : 0
Track Volume : 100.00%
Matrix Structure : 1 0 0 0 1 0 0 0 1
Media Header Version : 0
Media Create Date : 2017:04:26 21:07:31
Media Modify Date : 2017:04:26 21:07:31
Media Time Scale : 44100
Media Duration : 0:00:30
Balance : 0
Audio Format : mp4a
Audio Channels : 2
Audio Bits Per Sample : 16
Audio Sample Rate : 44100
Handler Type : Metadata
Handler Vendor ID : Apple
iTunSMPB : 0 840 3C8 142FF8 0 0 0 0 0 0 0 0 0 0
Media Data Size : 1202732
Media Data Offset : 57344
Avg Bitrate : 320 kbps
```

Można znaleźć takie informacje jak np. - data utworzenia, dane dostawcy czy bitrate

## Recoverjpeg

Rozpoczynam skanowanie za pomocą komendy

```
└─(kali㉿kali)-[~/media/sf_Kali_shared_folder]
$ recoverjpeg obrazek.img
Restored 307 pictures
```

Udało się odnaleźć 307 plików, jest to znaczco mniejsza liczba niż w przypadku programu **foremost**

Przykładowy wynik



image00268.jpg



image00221.jpg



image00222.jpg



image00209.jpg



image00210.jpg



image00204.jpg

Niestety również metadane nie zostały zachowane, narzędzie to nie pomogło

## Bulk\_extractor

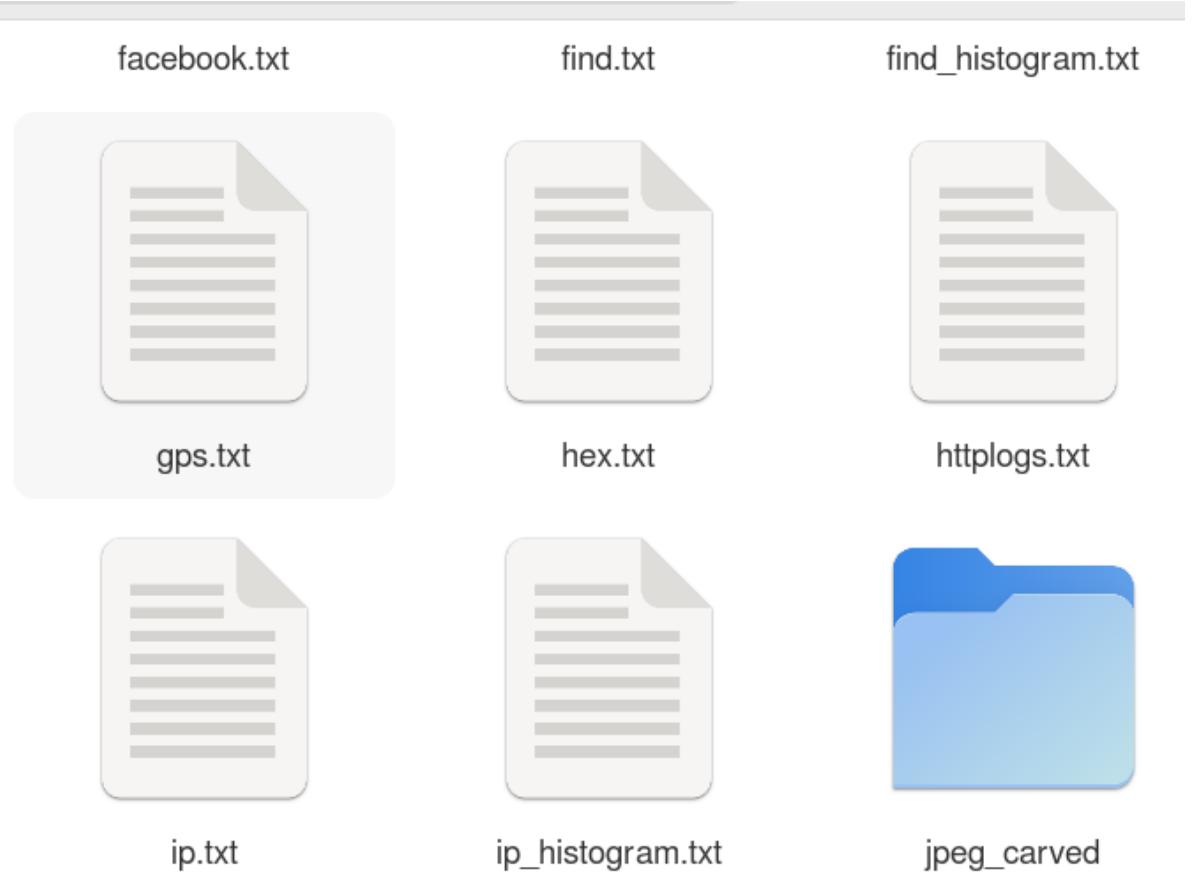
Przyszła teraz pora na program, który jest znany z wydobywania metadanych - **bulk\_extractor**. Zobaczmy, czy pomoże rozwiązać problem ich braku w analizowanym obrazie

Uruchamiam program komendą

```
(kali㉿kali)-[/media/sf_Kali_shared_folder]
$ sudo bulk_extractor -e all obrazek.img -o bulk_extractor
mkdir "bulk_extractor"
bulk_extractor version: 2.0.0
Input file: "obrazek.img"
Output directory: "bulk_extractor"
Disk Size: 32212254720
Scanners: aes base64 elf evtx exif facebook find gzip hiberfile httplogs json kml_carved msxml net ntfsindx ntfslog
file ntfsmft ntfsusn outlook pdf rar sqlite utmp vcard_carved windirs winlnk winpe winprefetch wordlist xor zip bas
e16 accts email gps
```

Generalnie efektywność bulk\_extractora można opisać jako dość biedną - jednakże jego siła kryje się w wszelkich innych typach plików i sygnaturach

Potrafi wydobyć przykładowo linki url zawarte w systemie, numery telefonów, adresy email, zapytania DNS



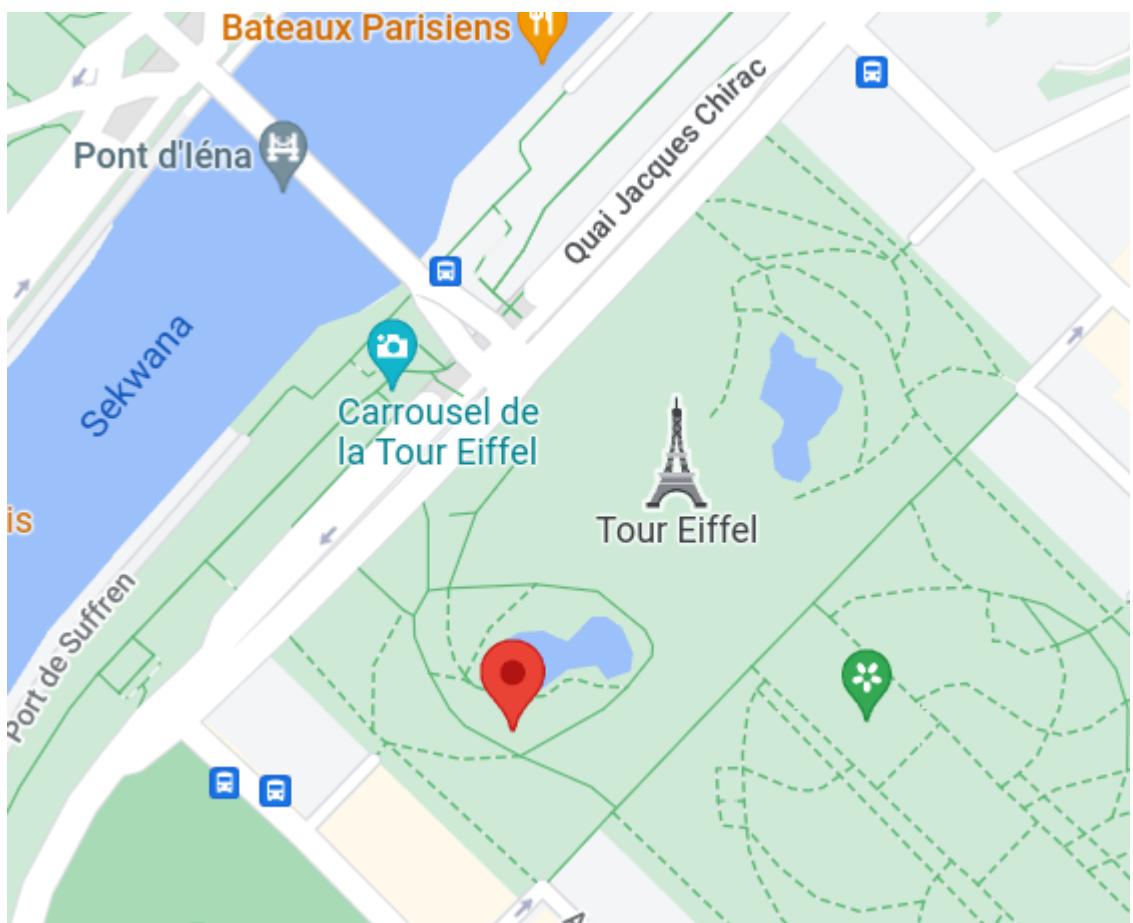
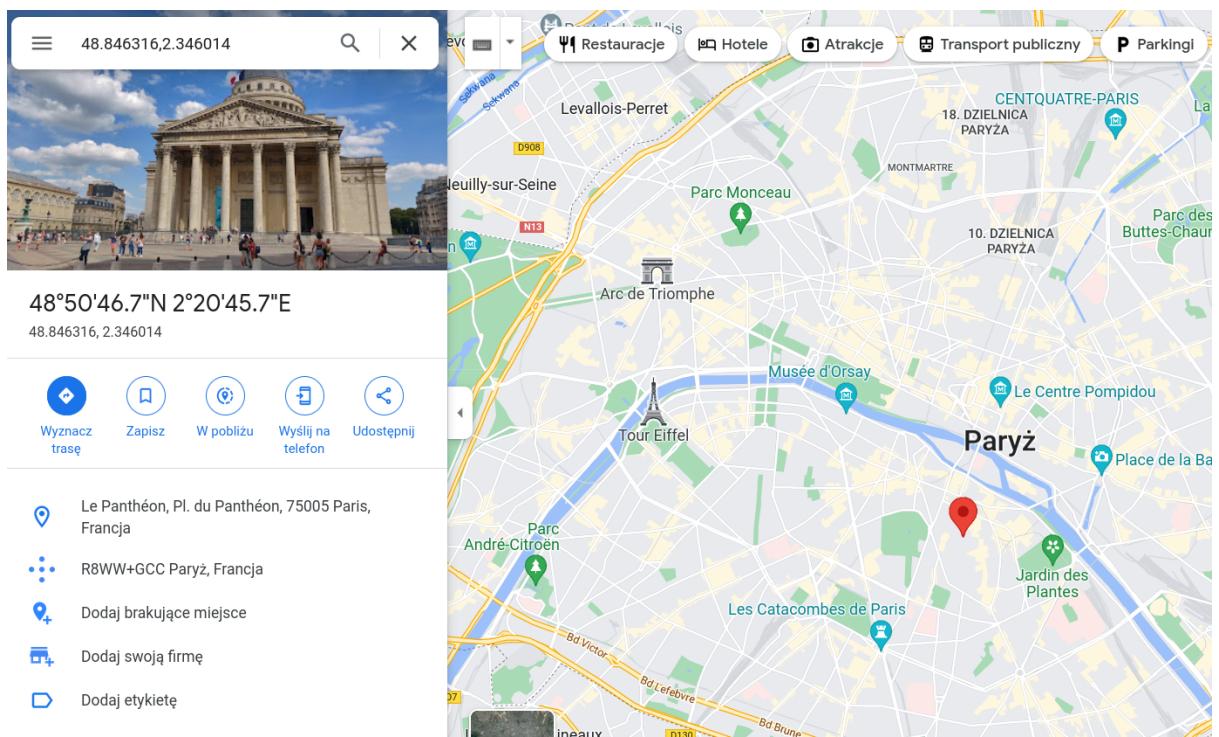
Program ten zbiera odnalezione dane i dzieli je na pliki tekstowe - zawierające szczegółowe informacje - oraz foldery z odzyskanymi plikami

## GPS

Programowi udało się wydobyć trochę danych lokalizacyjnych - czego nie udało się reszcie (mimo przeszukania wszystkich metadanych plików za pomocą komend powłoki bash)

```
# BANNER FILE NOT PROVIDED (-b option)
# BULK_EXTRACTOR-Version: 2.0.0
# Feature-Recoder: gps
# Filename: obrazek.img
# Feature-File-Version: 1.1
7812300800 12af1b7464d392cf54a9e1907ca15361429cedbf
7818002432 c0948bbef8db75c21ce80ea08d17b547accc12f3
7824109568 ed277774a2d9c4fbf4d02c999cdf844bee8a61f4
7824953344 7b20d9c8e92b8b2b68b725d218969df8229dd7ceb
7935184896 cb36286bea4b92ae5396483d0ee8567298ab73e5
7944548352 ea5b5a967e0dbca49861b2773a28f1aad42c8ccb
8112488448 574f94d029d1dad9bd007cc559fac3a2f9700035
8121319424 3c63214503fab2c44a32d45fee45e4cedaba2c5e
8228851712 b0fbda5f1863e294a9df7dc408fbe1b666ec393e
13364305920 eee65e602723ecec7bab04f05baf8d2e5dae424a
13555433472 1bd21131b0ffc680cd42a122c7e771a9ba2860a
13841334272 c49f1700cc9f4247f8ba9bb5e92762e8879ef8f0
14414909440 5587c988d014d2f8b691b297cc8d585003dd53ab
14576250880 5214315d157143373020756f24070545183a45a3
2012-07-25T17/1 52/1 14/1,48.857496,2.293340,85.000000,,,
2012-07-25T8/1 35/1 52/1,48.846316,2.346014,0.000000,,,
2012-07-25T15/1 42/1 3/1,48.846316,2.346014,0.000000,,,
2012-07-25T17/1 52/1 14/1,48.857496,2.293340,85.000000,,,
2012-07-25T16/1 20/1 44/1,48.854970,2.289638,90.000000,,,
2012-07-25T16/1 24/1 12/1,48.856687,2.290600,91.000000,,,
2012-07-25T9/1 24/1 16/1,48.847136,2.337402,87.000000,,,
2012-07-25T9/1 23/1 45/1,48.847089,2.337463,100.000000,,,
2012-07-25T17/1 52/1 14/1,48.857496,2.293340,85.000000,,,
2012-07-25T17/1 42/1 54/1,48.857382,2.293374,86.000000,,,
2012-07-25T9/1 24/1 46/1,48.847170,2.337398,90.000000,,,
2012-07-25T17/1 47/1 55/1,48.857470,2.293360,84.000000,,,
2012-07-25T10/1 4/1 24/1,48.847906,2.338748,84.000000,,,
2012-07-25T9/1 16/1 14/1,48.846952,2.346007,85.000000,,,
```

Sprawdźmy więc przykładowe dane:



Lokalizacja wskazuje typowo na paryż, co jest zgodne z analiza przeprowadzoną z użyciem programu autopsy

Dostępne są również daty utworzenia danego zdjęcia - wskazują na rok 2012, co również się zgadza

Co więcej, udało się odzyskać metadane z wszystkich zdjęć podróży użytkownika - jednakże z małym problemem, który opiszę poniżej

JPG



14462982144-GZIP-0.jpg

14548045824-GZIP-0.jpg

Niestety, w folderze *jpg\_carved* udało się znaleźć zaledwie 2 pliki, co jest naprawdę małą ilością zważając na inne programy (np. ponad 400)

## Podsumowanie

Generalnie każde z powyższych narzędzi w jaki sposób spełniło swoją rolę - z tym, że istnieją narzędzia, które łączą plusy każdego programu a nawet są bardziej efektywne (mowa tutaj o np. **autopsy**, **RescuePro**). W przypadku analizy obrazu systemu można znaleźć wiele plików należących do systemu operacyjnego, co znacznie utrudnia pracę i ciężko jest przypisać konkretne dane do użytkownika

# Wyciągnięciu danych ze smartfona. Analiza plików (PLIST i SQLite).

Na potrzeby następującej analizy skorzystam z gotowego obrazu pobranego z MS Teams oraz narzędzia polecanego przez NIST: modułu **Autopsy Android/iOS**  
Również przeanalizuję bazę danych - plik accounts3.sqlite

## Analiza sqlite

Sprawdź dostępne *tables*:

```
sqlite> .tables
ZACCESSOPTIONSKEY          Z_2ENABLEDDATACLASSES
ZACCOUNT                   Z_2PROVISIONEDDATACLASSES
ZACCOUNTPROPERTY           Z_4SUPPORTEDDATACLASSES
ZACCOUNTTYPE                Z_4SYNCABLEDATACLASSES
ZAUTHORIZATION              Z_METADATA
ZCREDENTIALITEM             Z_MODELCACHE
ZDATACLASS                  Z_PRIMARYKEY
Z_1OWNINGACCOUNTTYPES
```

Za pomocą komendy SELECT ZUSERNAME, ZACCOUNTDESCRIPTION FROM ZACCOUNT:

```
sqlite> SELECT ZUSERNAME, ZACCOUNTDESCRIPTION FROM ZACCOUNT;  
|Local  
thisisdfir@gmail.com|  
thisisdfir@gmail.com|  
thisisdfir@gmail.com|iCloud  
thisisdfir@gmail.com|  
thisisdfir@gmail.com|  
thisisdfir@gmail.com|
```

Można wydobyć adres email użytkownika - [thisisdfir@gmail.com](mailto>thisisdfir@gmail.com)

Na sam start warto zaznaczyć iż użytkownik korzystał z telefonem systemem iOS

```
sqlite> SELECT * FROM ZACCOUNTPROPERTY;
1|3|49|1|cookies|bplist000
2|3|6|1|storefrontID|bplist000
X$versionY$archiverT$topX$objects
3|3|1|2|dsid|bplist000
X$versionY$archiverT$topX$objects
4|3|1|2|isManagedAppleID|bplist000
X$versionY$archiverT$topX$objects
5|3|1|2|lastName|bplist000
X$versionY$archiverT$topX$objects
```

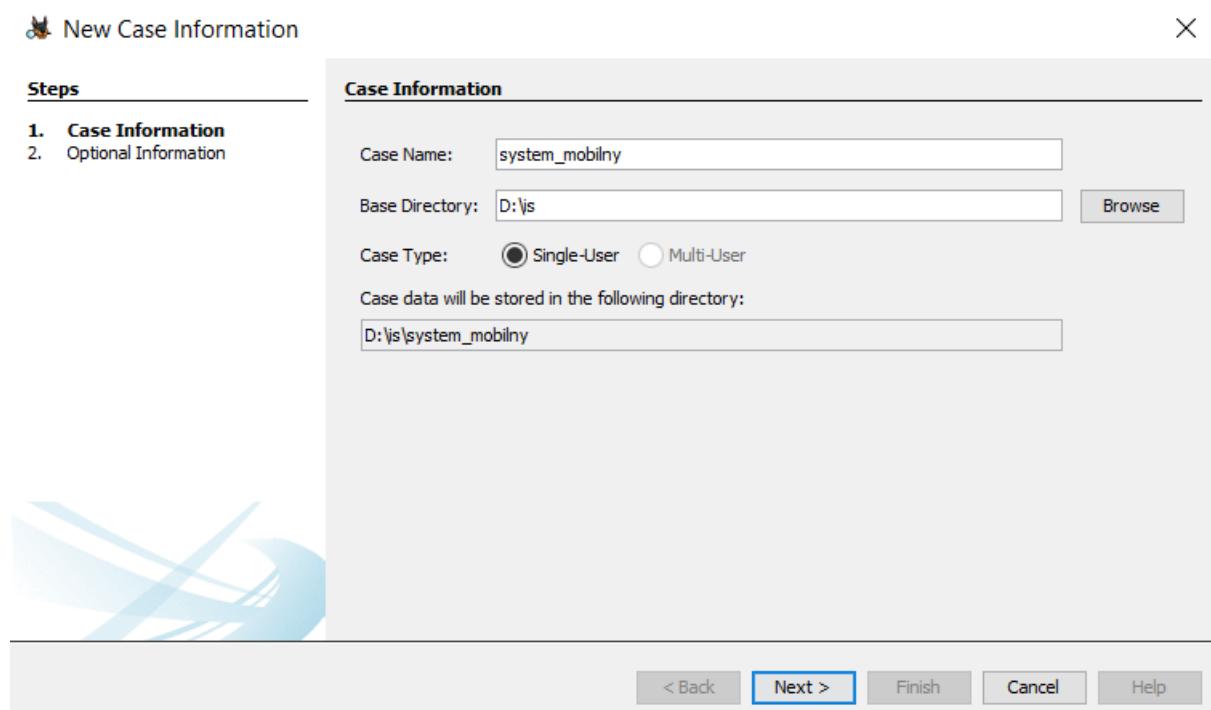
```
sqlite> SELECT * FROM Z_2ENABLEDDATACLASSES;
4|30
4|36
4|20
4|14
4|8
4|27
4|19
4|2
```

Następnie sprawdzam każdą tabelę

Jednakże nie udaje mi się odnaleźć nic ciekawego, a sama metodologia jest czasochłonna i mało efektywna

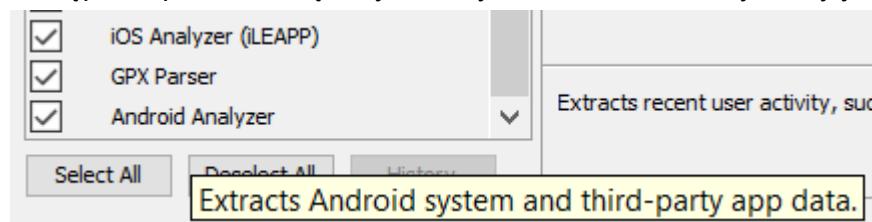
Dlatego przechodzę do następnego narzędzia - Autopsy z modułem iOS analyzer:

## Przygotowanie środowiska



Dodaję nową sprawę do programu

Następnie upewniam się, aby moduły iOS i Android Analyzer były zaznaczone



Dodaję folder do programu Autopsy

Rozpoczyna się skanowanie plików



## Analiza Autopsy

Po skończonym skanowaniu, struktura programu wygląda w następujący sposób:

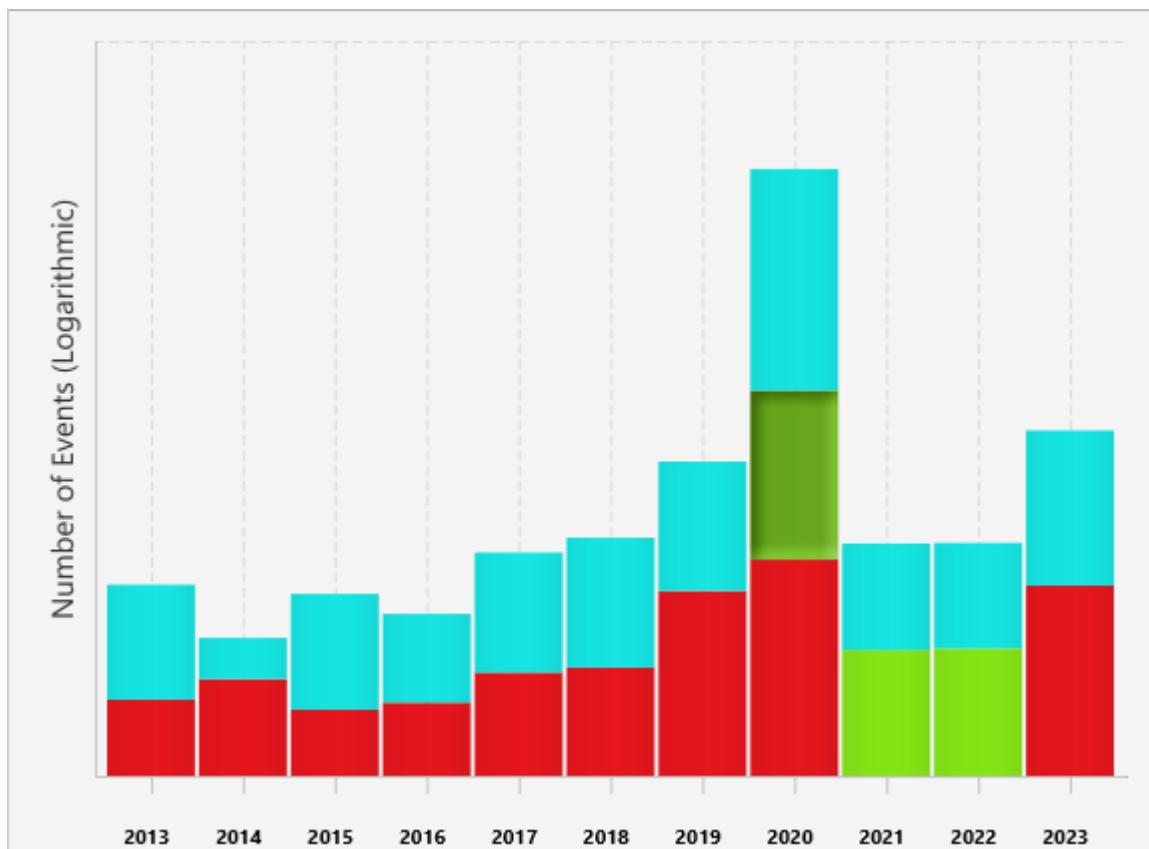
# system\_mobilny - Autopsy 4.19.3

Case View Tools Window Help

The screenshot shows the left sidebar of the Autopsy 4.19.3 interface. The sidebar contains several sections:

- File Types**
- Deleted Files**
- MB File Size**
- Data Artifacts** (selected)
  - Backup Events (14)
  - BlueTooth Pairings (54)
  - Calendar Entries (119)
  - Call Logs (32)
  - Communication Accounts (2)
  - Deleted Programs (1)
  - DHCP Information (6)
  - GPS Last Known Location (619)
  - Installed Programs (434)
  - Messages (42)
  - Metadata (307)
  - Program Notifications (1126)
  - Recent Documents (16)
  - Run Programs (5865)
  - Screenshots (574)
  - USB Device Attached (1)
  - User Device Events (5799)
  - Web Cookies (736)
  - Web History (101)
  - Web Search (18)
  - Wireless Networks (3172)
- Analysis Results**
  - Encryption Suspected (2)
  - EXIF Metadata (16)
  - Extension Mismatch Detected (152)
  - Keyword Hits (9060)
  - User Content Suspected (16)
  - Web Categories (7)

## TimeLine



Analizując powyższy wykres można stwierdzić, iż aktywność użytkownika przypada na okres lat 2020-2022

Czerwone - system plików - metadane z nich niekoniecznie mogły zostać stworzone w czasie kiedy użytkownik korzystał z systemu

Zielone - Web Activity, głównie korzystanie z przeglądarki i mediów społecznościowych (reddit, instagram, mewe, tiktok)

Niebieskie - Inne

Szczególnie intensywny okres przypada na marzec/kwiecień 2020

## Call Logs

Source Name	0	0	0	Start Date/Time	Phone Number	Comment	Data Source
CallHistory.storedata	0	0	0	2020-03-23 20:02:52 CET	+14082560700	Call Logs	LogicalFileSet3
CallHistory.storedata	0	0	0	2020-03-24 17:37:18 CET	+14082560700	Call Logs	LogicalFileSet3
CallHistory.storedata	0	0	0	2020-03-26 17:51:45 CET	+14082560700	Call Logs	LogicalFileSet3
CallHistory.storedata	0	0	0	2020-03-27 16:25:36 CET	+14082560700	Call Logs	LogicalFileSet3
CallHistory.storedata	0	0	0	2020-03-27 19:55:03 CET	+14082560700	Call Logs	LogicalFileSet3
CallHistory.storedata	0	0	0	2020-04-01 20:06:38 CEST	+14082560700	Call Logs	LogicalFileSet3
CallHistory.storedata	0	0	0	2020-04-03 16:10:54 CEST	+14082560700	Call Logs	LogicalFileSet3
CallHistory.storedata	0	0	0	2020-04-05 20:42:18 CEST	9197627808	Call Logs	LogicalFileSet3
CallHistory.storedata	0	0	0	2020-04-06 20:34:33 CEST	+14082560700	Call Logs	LogicalFileSet3
CallHistory.storedata	0	0	0	2020-04-06 21:48:08 CEST	+14082560700	Call Logs	LogicalFileSet3

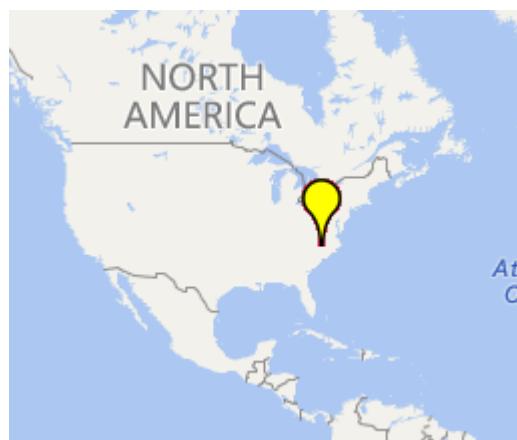
Użytkownik wykonywał połączenia z następującymi numerami:

- +14082560700
- +19193643164
- +17042751134
- 9197627808
- 9192853680

Wszystkie te połączenia były wykonane na przełomie 9-12 kwietnia 2020

## Lokalizacja GPS

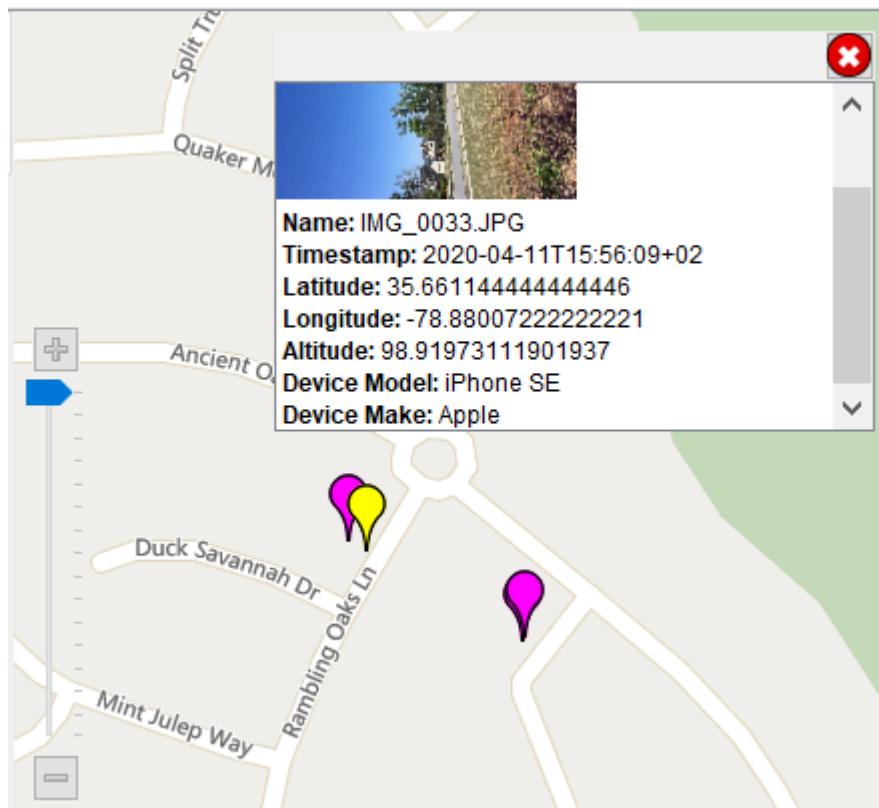
Przechodzę do zakładki *Geolocation*



Cała aktywność okresu korzystania użytkownika z telefonu mieści się w jednym miasteczku w Stanach Zjednoczonych -> Holly Springs

Znalezione dane można pogrupować na 3 kategorie:

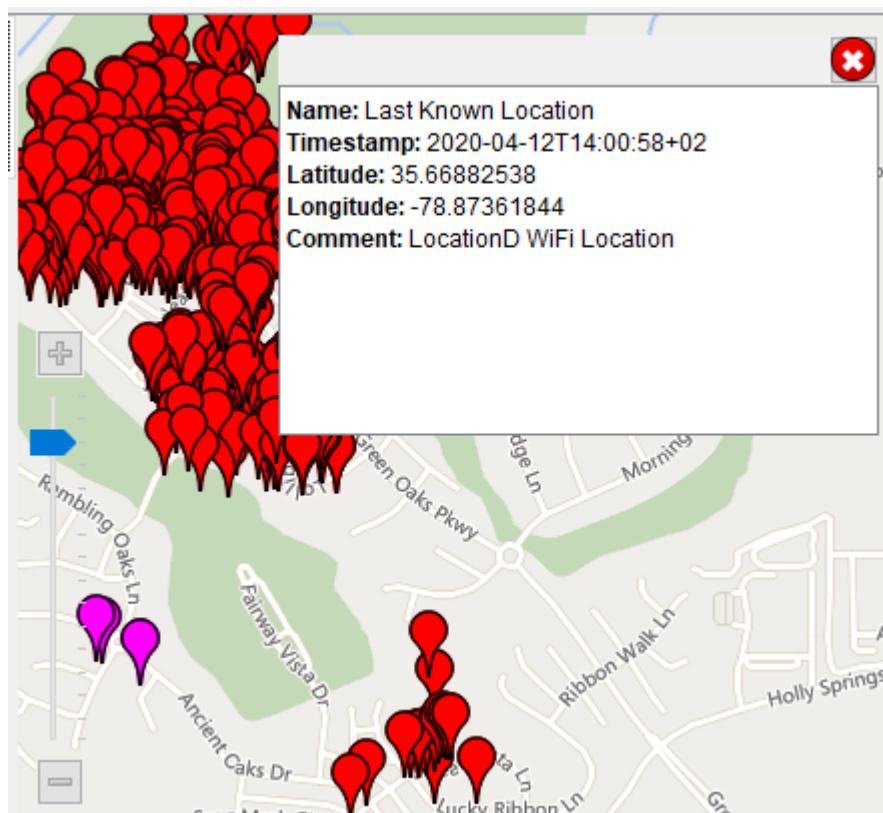
1) Zdjęcia zrobione telefonem wraz z zostawioną, włączoną lokalizacją GPS



Dostępne są tylko 4 takie pliki, co nie jest zbyt dużą liczbą jak na telefon - niektórzy użytkownicy trzymają ich tysiące w pamięci

Na powyższych zdjęciach można zobaczyć osiedle znajdujące się w tym miejscu na mapie - i faktycznie tam jest - porównując ze Street View map Google

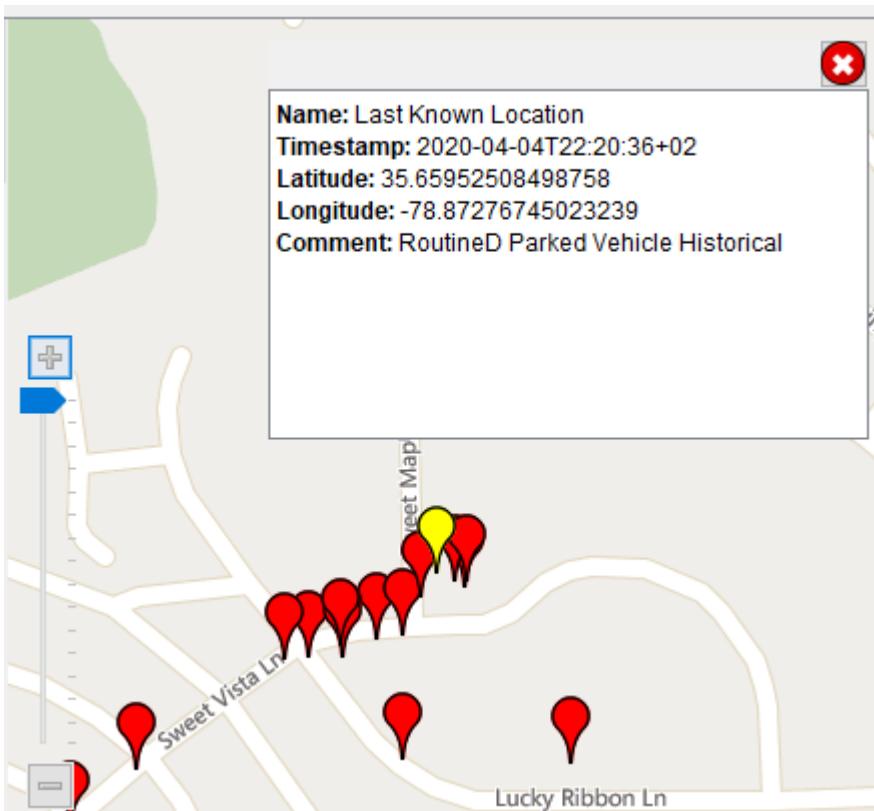
2) Współrzędne lokalizacyjne sieci wi-fi



A jest ich naprawdę dużo, obejmują zasięg połowy miasteczka.  
Interesujący jest fakt, że telefon zapisuje te dane w takiej ilości i robi to często (można wywnioskować po odstępach *Timestamp'u* kolejnych plików)  
Na podstawie gęstości występowania znaczników, można stwierdzić w jakich miejscach użytkownik przebywał najczęściej - a tym samym określić miejsce jego zamieszkania - prawdopodobnie okolice 228 Whisk Fern Way, Holly Springs, NC 27540, Stany Zjednoczone



### 3) Lokalizacja zaparkowania samochodu



Jest to ślad działania usługi [map Apple](#), która zostawia znacznik lokalizacyjny samochodu, kiedy telefon rozłączy się z usługą Bluetooth z Apple CarPlay

LogicalFileSet3			2020-04-12 14:27:43 CEST	35.65949216...	-78.87260528...	RoutineD Parked Vehic...	LogicalFileSet3
LogicalFileSet3			2020-04-12 14:27:43 CEST	35.65949216...	-78.87260528...	RoutineD Parked Vehic...	LogicalFileSet3
LogicalFileSet3			2020-04-12 14:27:43 CEST	35.65949216...	-78.87260528...	RoutineD Parked Vehic...	LogicalFileSet3
LogicalFileSet3			2020-04-12 14:27:43 CEST	35.65949216...	-78.87260528...	RoutineD Vehicle Location	LogicalFileSet3

Co ciekawe, oprócz zaparkowanego samochodu można również znaleźć informację o lokalizacji auta na żywo/w ruchu

Na podstawie tych danych można stwierdzić zarówno miejsce zamieszkania użytkownika jak i siedzibę firmy, w której pracuje ta osoba

### Urządzenia BlueTooth

W zakładce *BlueTooth Pairings* dostępnych jest ponad 50 rekordów i jedyne widoczne urządzenia to słuchawki AirPods, które mają 2 różne nazwy:

- Josh's AirPods
- This Is's AirPods

Josh's AirPods
This Is's AirPods Pro

Na tej podstawie można stwierdzić iż imię użytkownika telefonu to najprawdopodobniej **Josh**

USB Device Attached									1 Result	
Table		Thumbnail		Summary						
Source Name		S	C	O	Date/Time	Device ID	Device Make	Device Model	Comment	Data Source
LogicalFileSet3		0			2020-04-06 14:56:46 CEST	175811	Watch4,3	N141sAP	Powerlog Paired Device Configuration	LogicalFileSet3

Hex	Text	Application	Source File Metadata	OS Account	Data Artifacts	Analysis Results	Context	Annotations	Other Occurrences
Result: 14619 of 17875	Result								USB Device Attached
<hr/>									
Result: 14619 of 17875									
Type	Value							Source(s)	
Date/Time	2020-04-06 14:56:46 CEST							iOS Analyzer (IEA)	
Device ID	175811							iOS Analyzer (IEA)	
Device Make	Watch4,3							iOS Analyzer (IEA)	
Device Model	N141sAP							iOS Analyzer (IEA)	
Comment	Powerlog Paired Device Configuration							iOS Analyzer (IEA)	

Chociaż w zakładce USB Device Attached można znaleźć informację, że użytkownik korzystał z Apple Watcha - 40mm GPS + Cellular Apple Watch Series 4 (Watch4,3 model)

## Calendar Entries

LogicalFileSet3	2019-09-30 00:00:00 CEST	2019-09-30 23:59:59 CEST	Rosh Hashanah
LogicalFileSet3	2019-10-09 00:00:00 CEST	2019-10-09 23:59:59 CEST	Yom Kippur
LogicalFileSet3	2019-10-27 00:00:00 CEST	2019-10-27 23:59:59 CET	Diwali
LogicalFileSet3	2019-11-28 00:00:00 CET	2019-11-28 23:59:59 CET	Thanksgiving
LogicalFileSet3	2019-12-23 00:00:00 CET	2019-12-23 23:59:59 CET	Hanukkah
LogicalFileSet3	2020-01-25 00:00:00 CET	2020-01-25 23:59:59 CET	Lunar New Year
LogicalFileSet3	2020-03-10 00:00:00 CET	2020-03-10 23:59:59 CET	Holi
LogicalFileSet3	2020-04-05 00:00:00 CEST	2020-04-05 23:59:59 CEST	Palm Sunday
LogicalFileSet3	2020-04-08 00:00:00 CEST	2020-04-08 23:59:59 CEST	Passover
LogicalFileSet3	2020-04-10 00:00:00 CEST	2020-04-10 23:59:59 CEST	Good Friday
LogicalFileSet3	2020-04-12 00:00:00 CEST	2020-04-12 23:59:59 CEST	Easter
LogicalFileSet3	2020-04-12 14:00:00 CEST	2020-04-12 15:00:00 CEST	Pick up lunch
LogicalFileSet3	2020-04-15 00:00:00 CEST	2020-04-15 23:59:59 CEST	Tax Day
LogicalFileSet3	2020-04-19 00:00:00 CEST	2020-04-19 23:59:59 CEST	Orthodox Easter
LogicalFileSet3	2020-04-24 00:00:00 CEST	2020-04-24 23:59:59 CEST	Beginning of Ramadan
LogicalFileSet3	2020-05-23 00:00:00 CEST	2020-05-23 23:59:59 CEST	Eid al-Fitr

Można stąd wydobyć historię zdarzeń, na które najprawdopodobniej poszedł nasz użytkownik

Większość z dostępnych wpisów to niestety bardzo ogólne, święta narodowe/dni wolne. Jedynym prywatnym wpisem, który udało mi się znaleźć, jest *Pick up lunch* - jako jedyny ma ustawioną normalną godzinę (15:00), zamiast 23:59:59

## Oprogramowanie

Według programu, użytkownik ma zainstalowane 434 programy - jest to naprawdę duża liczba. Myślę że liczba jest tak wielka z powodu również wliczania się aplikacji systemowych (np. HomeUIService.app) lub niektóre są powielone

Program Name	Path	Przykładowa lista programów użytkownika
Houseparty	com.herzick.houseparty	
Snapchat	com.toyopagroup.picaboo	
Signal - Private Messenger	org.whispersystems.signal	
Signal - Private Messenger	org.whispersystems.signal	
Venmo	net.kortina.labs.Venmo	
Sync Solver for Fitbit	com.redshiftdev.Fitbit-Health-Sync	
	com.apple.DocumentsApp	
WhatsApp Messenger	net.whatsapp.WhatsApp	
Viber Messenger: Chats & Calls	com.viber	
TikTok - Make Your Day	com.zhiliaoapp.musically	
Firefox Focus: Privacy browser	org.mozilla.ios.Focus	
Telegram Messenger	ph.telegra.Telegraph	

Znajdziemy tutaj:

- różne komunikatory np. messenger, signal, whatsapp, discord
- media społecznościowe np. tiktok, facebook, instagram
- aplikacje systemowe typu mapy, eksplorator plików, przeglądarka
- skrzynkę pocztową ProtonMail

## Messages

Chyba jest to najciekawsza zakładka, ponieważ SMS'y nie są szyfrowane i można zobaczyć ich zawartość plaintextem

Here is another one. ~s9LlsgAIo/D1mAg  
It is. But there was some weird stuff at the end.  
Hopefully this is a green message. ~A4eIvwAGEzyYAgg  
Here is the sysdiagnose paper. Enjoy!  
Here is Heather's paper from DFIR Review.  
It is! We will still need to generate a green bubble at some...  
Test iMessage. I hope this is a blue bubble!  
889557 is your verification code for Dust - a safer place to ...

Można tutaj zobaczyć konwersacje użytkownika z inną osobą - pisząc o słynnym *green bubble* w przypadku urządzeń z iOS

Warto zaznaczyć, że w tej zakładce nie da się wydobyć załączników - które użytkownik najprawdopodobniej przesłał

Text	Można również odnaleźć kody weryfikacyjne do usług wymagających 2FA (Google, Tiktok, Signal, Viber, Telegram).
Your Viber code: 276340 You can also tap this link to finish ...	
? ThisIs, you have 5 new notifications on Facebook: https://...	
Your Skout verification code is: 46710~ZA1ZVgAJzBzxTgg	
Your Skout verification code is 3079,~8DWNaAAjpQzYcg	
Your WhatsApp code: 203-591 You can also tap on this link ...	
[TikTok] 8494 is your verification code, valid for 5 minutes. ...	
Telegram code: 95037 You can also tap on this link to log in: ...	Jest to idealny przykład, dlaczego uwierzytelnianie dwuskładnikowe w postaci kodów SMS nie jest do końca bezpieczne - wiadomości są relatywnie łatwe do przechwycenia (atak typu <i>man in the middle</i> - np. używane przez NSA tzw <i>jaskółki</i> ) a fakt, że nie są szyfrowane pozwala na ich łatwe odczytanie
? ThisIs, you have 2 new notifications on Facebook: https://...	
Your Signal verification code: 903-394 Or tap: sgnl://verify/...	
Please enter this code in the app to reset your MeWe pass...	
Please enter 737950 into LINE within the next 30 mins.~iczs...	
imo code: 258465~kVFkSwBQWNytllgg	
G-773293 is your Google verification code.~FB9YQAAFDQ...	

## Events

Jest to zakładka do przeznaczona do wyświetlania powiadomień systemowych:

- o poziomie baterii
- podłączeniu urządzenia do ładowania
- włączeniu trybu *do not disturb*
- zablokowaniu urządzenia
- zmianie orientacji ekranu
- używaniu Siri

Jednakże najdziwniejsze wydaje się powiadomienie o obudzeniu się użytkownika wraz z dokładną datą! Fakt ten może być dyskusyjny jeśli chodzi o prywatność

2020-04-12 13:54:03 CEST	2020-04-12 16:54:32 CEST	User Waking	LogicalFileSet3
2020-03-21 21:44:26 CET	2020-03-21 22:57:25 CET	User Waking	LogicalFileSet3
2020-04-03 23:01:08 CEST	2020-04-04 16:13:36 CEST	User Waking	LogicalFileSet3
2020-04-06 21:51:14 CEST	2020-04-07 00:02:37 CEST	User Waking	LogicalFileSet3
2020-04-08 00:37:16 CEST	2020-04-12 13:54:03 CEST	User Waking	LogicalFileSet3
2020-04-12 17:38:21 CEST	2020-04-12 17:38:36 CEST	User Waking	LogicalFileSet3

## Web Search i Web History

Source Name	S	C	O	URL	Date Accessed
History.db			1	https://www.google.com/search?q=when+does+mlb+star...	2020-03-28 02:02:05 CET
History.db			1	https://www.google.com/search?q=Is+the+NHL+going+t...	2020-03-28 02:02:44 CET
History.db			1	https://www.google.com/search?q=Is+the+NHL+going+t...	2020-03-28 02:02:44 CET
History.db			1	https://www.nhl.com/news/commissioner-gary-bettman-nh...	2020-03-28 02:03:53 CET
History.db			1	https://www.nhl.com/news/commissioner-gary-bettman-nh...	2020-03-28 02:03:51 CET
History.db			1	https://www.apple.com/	2020-03-28 02:06:32 CET
History.db			1	https://www.apple.com/ipad-pro/	2020-03-28 02:07:41 CET
History.db			1	https://www.cultofmac.com/	2020-03-28 02:38:48 CET
History.db			1	https://dfir.pubpub.org/	2020-03-28 02:43:49 CET
History.db			1	https://dfir.pubpub.org/pub/e5xlbw88	2020-03-28 02:44:37 CET

Można tutaj zobaczyć strony, z którymi łączył się użytkownik wraz z dokładną datą.

Na tej podstawie można stwierdzić preferencję użytkownika

Użytkownik wyszukiwał:

- strony apple.com wraz z urządzeniem Ipad Pro
- kiedy zaczyna się *mlb 2020* oraz informacje o sędziim
- czy NHL będzie kontynuowane
- serwis dfir.pubpub.org

## Podsumowanie

Z urządzenia mobilnego można znacznie więcej zyskać niż z tradycyjnego komputera. Geolokalizacja urządzenia pozwala na odtworzenie dokładnej ścieżki po której poruszał się użytkownik, określenia miejsca zamieszkania/pracy, odtworzeniu gdzie znajdują się sieci wi-fi. Poza tym można łatwo przechwycić wrażliwe wiadomości w postaci SMS'ów. Również łatwo jest zobaczyć ważne informację o zdrowiu fizycznym użytkownika - np. kiedy się budził.

Generalnie rzecz biorąc, telefon zbiera znaczne ilości danych i jeśli takowe dane trafią w ręce osób niepowołanych, to mogą wywołać niemałe szkody - a co dopiero nie mówiąc o naruszeniu prywatności.

## Utworzenie kopii binarnej nośnika

ewfacquire

Na potrzeby późniejszej analizy konwertuje plik .raw na format czytany przez narzędzia ewf -  
ewfacquire

```
[arek@fedora projekt] $ ewfacquire -f raw example_image.raw
ewfacquire 20140608

Storage media information:
Type: RAW image
Media size: 86 GB (86000000000 bytes)
Bytes per sector: 512
```

```
wipe_sectors_on_read_error (mimic EnCase-like behavior) (yes, no) [no]

The following acquisition parameters were provided:
Image path and filename:                      my_case.E01
Case number:                                    1
Description:                                   Projekt Informatyka Sledcza
Evidence number:                             12
Examiner name:                                Arek
Notes:                                         None
Media type:                                    fixed disk
Is physical:                                 yes
EWF file format:                            EnCase 6 (.E01)
Compression method:                          deflate
Compression level:                           none
Acquisition start offset:                   0
Number of bytes to acquire:                 80 GiB (86000000000 bytes)
Evidence segment file size:                86 GiB (92341796864 bytes)
Bytes per sector:                            512
Block size:                                    64 sectors
Error granularity:                           64 sectors
Retries on read error:                      2
Zero sectors on read error:                 no
```

Ustawianie podstawowych parametrów

Warto zaznaczyć iż blok ma **64 sektory**, a każdy sektor ma **512 bajty**

```
Written: 80 GiB (86000001316 bytes) in 7 minute(s) and 47 second(s) with 175 MiB/s (184154178 bytes/second).
MD5 hash calculated over data:           d40647c0fd182404e81d4b91eb3a074b
ewfacquire: SUCCESS
```

Jak widać konwersja przeszła pomyślnie

## hash

hash (może przydać się później przy sprawdzaniu integralności danych):  
d40647c0fd182404e81d4b91eb3a074b

## zamontowanie obrazu

Przechodzę do zamontowania obrazu w folderze /mnt/ tak, aby miał on formę fizycznego (lub logicznego) obrazu dysku

```
[arek@fedora projekt] $ sudo mkdir /mnt/ewf
[sudo] password for arek:
[arek@fedora projekt] $ ewfmount my_case.E01 /mnt/ewf
ewfmount 20140608

fusermount: user has no write access to mountpoint /mnt/ewf
Unable to create fuse channel.
[arek@fedora projekt] $ sudo chown arek /mnt/ewf
[arek@fedora projekt] $ ewfmount my_case.E01 /mnt/ewf
ewfmount 20140608

[arek@fedora projekt] $ cd /mnt/ewf/
[arek@fedora ewf] $ ll
total 0
-r--r--r--. 1 root root 86000000000 Nov 19 16:46 ewf1
[arek@fedora ewf] $
```

## System plików

Następnie trzeba ustalić system plików, który jest używany w naszym obrazie dysku. W przypadku systemów unixo-pochodnych najczęściej jest to .ext4

```
[arek@fedora projekt] $ fsck -N my_case.E01
fsck from util-linux 2.38.1
[/usr/sbin/fsck.ext4 (1) -- /boot] fsck.ext4 my_case.E01 /dev/nvme0n1p4
[/usr/sbin/fsck.vfat (1) -- /boot/efi] fsck.vfat my_case.E01 /dev/nvme0n1p1
[arek@fedora projekt] $
```

I rzeczywiście, interesujący nas system plików to ext4

```
[arek@fedora projekt] $ mmcls -i ewf my_case.E01
DOS Partition Table
Offset Sector: 0
Units are in 512-byte sectors

      Slot      Start          End          Length        Description
000: Meta    0000000000  0000000000  0000000001 Primary Table (#0)
001: -----  0000000000  0000002047  0000002048 Unallocated
002: 000:000  0000002048  0167968749  0167966702 Linux (0x83)
[arek@fedora projekt] $ mmcls -c 512 -t DOS my_case.E01
```

Mając powyższe informacje (offset sector, block size) możemy przejść do analizy grup bloków

## Analiza grup bloków

```
arek@fedora:/mnt/ewf — fsstat -b 512 -o 2048 ewf1

FILE SYSTEM INFORMATION
-----
File System Type: Ext4
Volume Name: root
Volume ID: 37ca020a1fe9b2bea645802c7deade12

Last Written at: 2022-11-18 09:53:54 (CET)
Last Checked at: 2022-08-08 12:26:53 (CEST)

Last Mounted at: 2022-11-18 09:53:54 (CET)
Unmounted properly
Last mounted on: /

Source OS: Linux
Dynamic Structure
Compat Features: Journal, Ext Attributes, Resize Inode, Dir Index
InCompat Features: Filetype, Extents, 64bit, Flexible Block Groups,
Read Only Compat Features: Sparse Super, Large File, Huge File, Extra Inode Size

Journal ID: 00
Journal Inode: 8

METADATA INFORMATION
-----
Inode Range: 1 - 5251073
Root Directory: 2
Free Inodes: 4842434
Inode Size: 256

CONTENT INFORMATION
-----
Block Groups Per Flex Group: 16
Block Range: 0 - 20995836
Block Size: 4096
Free Blocks: 16420819

BLOCK GROUP INFORMATION
-----
Number of Block Groups: 641
Inodes per group: 8192
Blocks per group: 32768

Group: 0:
  Block Group Flags: [INODE_ZEROED]
  Inode Range: 1 - 8192
  Block Range: 0 - 32767
  Layout:
:
```

Mamy dostępnych **641 grup bloków** - co wydaje się naprawdę sporą liczbą - a samych bloków na grupę: **32768**  
Liczba l-węzłów na grupę: **8192**

## Analiza plików

Przejdźmy więc do analizy znajdujących się w środku plików za pomocą komendy **fls**

```
[arek@fedora ewf] $ fls -b 512 -o 2048 ewf1
d/d 1966081:    home
d/d 11: lost+found
d/d 3538945:    opt
d/d 3276801:    usr
d/d 1703937:    proc
d/d 1835009:    root
l/l 12: lib64
d/d 2228225:    media
d/d 2097153:    sys
r/r 13: 0
d/d 2359297:    dev
l/l 14: bin
d/d 2490369:    srv
l/l 15: sbin
d/d 2752513:    tmp
l/l 16: lib
d/d 2883585:    boot
l/l 17: lib32
d/d 3014657:    mnt
l/l 18: libx32
d/d 3145729:    run
d/d 1703938:    var
d/d 2097154:    etc
r/r 19: swapfile
l/l 21: initrd.img.old
l/l 20: vmlinuz.old
l/l 23: initrd.img
l/l 22: vmlinuz
V/V 5251073:    $OrphanFiles
```

Idąc ścieżką /home/Downloads możemy trafić na parę interesujących plików

```
[arek@fedora ewf] $ fls -b 512 -o 2048 ewf1 1966124
r/r 2752521:      steam_latest.deb
r/r 2100387:      IMG_20221118_124742.jpg
d/d 2100380:      sledcaz
r/r 2100389:      IMG_20221118_125210.jpg
r/r 1972655:      6106881133c60-463963459.jpeg
r/r 2100388:      IMG_20221118_125642.jpg
r/r 1978317:      17501_Creative-iPhone-Photos-5_w560.jpg
r/r 1978330:      gsmarena_010.jpg
r/r 2100386:      IMG_20221118_130515.jpg
r/r 1978598:      samsung_galaxy_s20_ultra_01.jpg
r/r 1978402:      samsung_galaxy_s20_ultra_15.jpg
r/r 1978669:      samsung_galaxy_s20_ultra_28.jpg
r/r 2752620:      samsung_galaxy_s20_ultra_02.mp4
r/r 1978680:      4211112626.jpg
r/r 1978624:      6343296878.jpg
r/r 1978935:      6235050916.jpg
r/r 1978941:      0880771770.jpg
r/r 1966120:      pliktekstowy2.txt
r/r 2752619:      D19970553.pdf
r/r 1979239:      plik1.txt
r/r 2100384:      sample.pdf
r/r 2752636:      D20221692Lj.pdf
r/r 2752637:      szkolni_klasa2_podrecznik4.pdf
r/r 2752638:      LibreOffice_7.4.2_Linux_x86-64_rpm.tar.gz
d/d 2100236:      LibreOffice_7.4.2.3_Linux_x86-64_rpm
r/r 2752639:      PDFStudio_linux64.sh
r/r 2100382:      SarmataFlag.png
r/r 2100383:      Untitled 1.docx
r/r 2100385:      Untitled 1 in all.docx
[arek@fedora ewf] $
```

## Odzyskanie pliku

Spróbujmy teraz odzyskać przykładowy plik: SarmataFlag.png

Wiemy iż ma on następujący numer inode: 2100382

Za pomocą komendy icat jesteśmy w stanie uzyskać w pełni powyższy plik:

```

[arek@fedora ewf] $ icat -o 2048 -b 512 ewf1 2100382
PONG
IHDR 001 pHYs

/00&Q0 00IDATx00}0000{00w000000
00C0HP00P0400Xm*0FPE`00_0 b000?00000ZorX060000000vnf0003700<03p
p0
&00\09\0s0k000[Mc|lT0+/0000E00I0[00
F2C0#00h\xp
01I!/0r00F2g_000w

E. X]f0#g0v3]&. 0=0缶 000u0M0000=0V<0BS4t0
G0t000vlç0J000P0!(0
T0'$"q0x"00{0P00B0b00?000\00000=00z000e0009Q000C0k0870]003000?
Ry00HS04
00b!0b0n00;0U80█]000Z00*0*0
*0000N004;0000K
0X00s00█]001
h0U000000Z0"003YMjC0_00s00(_v0:MRa00:000Y0-0bb0LX000]0
0J00"0a0-n0L00Yd0EIp0000;Rbqt00%)$^0Vw!ō
?7900a800.0N0000ü00GR00
I0q
00]0, .0 @Rk/o02n4u00P0F0a-fEb%0`F
l0:D$0KE,@Ā000@0@0y00CJ0w0X00r00)0|040708 ED 4m060008h00b
0.0v_>000030>0iU0`0200}0L0000010$0070y0^h00<v00p0000000\f0,0m0
F0000o00>0c0000U0L0\]@00T|0000|x00@0j0q00, ;00000&000"Z}0u0q0t09m
009]00A00`000w0r00;00Va

0R000&0ZR00š 030^0000500000X!Uj;K<00

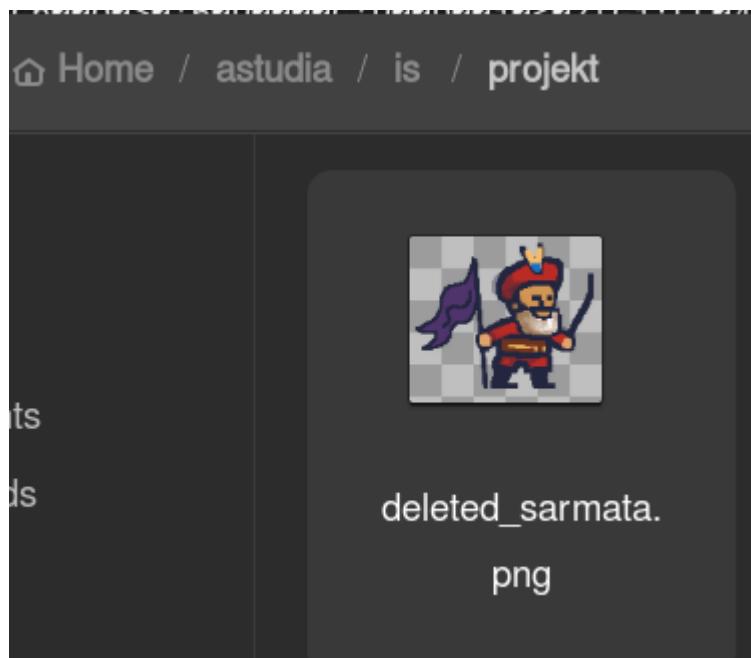
```

Jednak jest to format binarny, nieczytelny dla człowieka. W takim razie skierujmy output do pliku .png

```

[arek@fedora ewf] $ icat -o 2048 -b 512 ewf1 2100382 > ~/astudia/is/projekt/deleted_sarmata.png

```



Plik został odzyskany 😎

## Exif

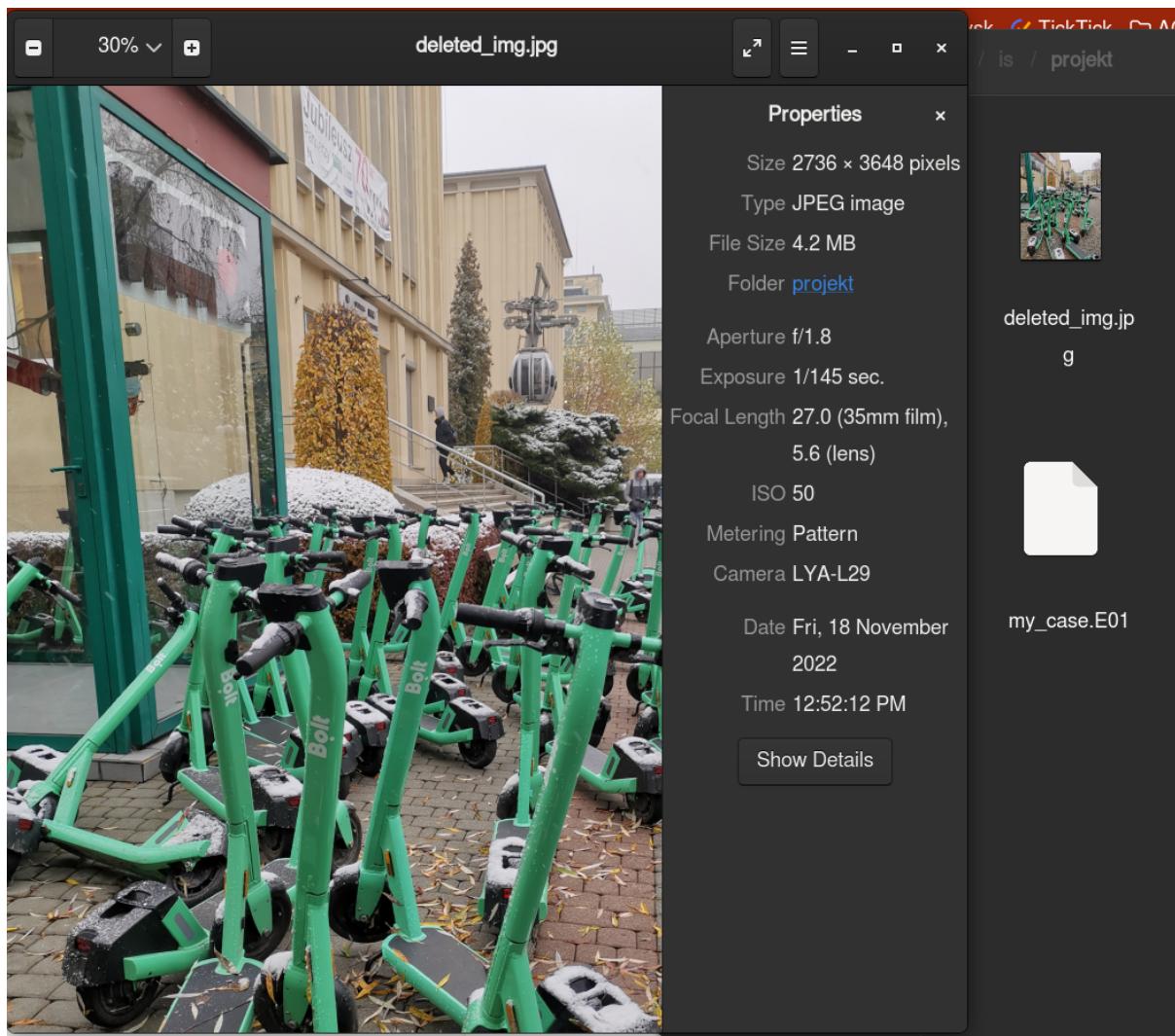
Można następnie przejść następnie do analizy danych exif (już lokalnie)

```
[arek@fedora projekt] $ exiftool deleted_sarmata.png
ExifTool Version Number : 12.42
File Name               : deleted_sarmata.png
Directory               :
File Size                : 3.7 kB
File Modification Date/Time : 2022:11:19 17:15:11+01:00
File Access Date/Time   : 2022:11:19 17:15:22+01:00
File Inode Change Date/Time : 2022:11:19 17:15:11+01:00
File Permissions        : -rw-r--r--
File Type                : PNG
File Type Extension     : png
MIME Type                : image/png
Image Width              : 140
Image Height             : 120
Bit Depth                : 8
Color Type                : RGB with Alpha
Compression              : Deflate/Inflate
Filter                   : Adaptive
Interlace                 : Noninterlaced
Pixels Per Unit X       : 2835
Pixels Per Unit Y       : 2835
Pixel Units              : meters
Image Size                : 140x120
Megapixels               : 0.017
```

Niestety nie ma tu nic ciekawego

Znajdźmy plik, który jest zdjęciem ze smartfona (najlepiej z danymi lokalizacji GPS)

```
[arek@fedora ewf] $ fls -b 512 -o 2048 ewf1 1966124
r/r 2752521: steam_latest.deb
r/r 2100387: IMG_20221118_124742.jpg
d/d 2100380: sledcaz
r/r 2100389: IMG_20221118_125210.jpg
r/r 1972655: 6106881133c60-463963459.jpeg
r/r 2100388: IMG_20221118_125642.jpg
r/r 1978317: 17501_Creative-iPhone-Photos-5_w560.jpg
r/r 1978330: gsmarena_010.jpg
r/r 2100386: IMG_20221118_130515.jpg
r/r 1978598: samsung_galaxy_s20_ultra_01.jpg
r/r 1978402: samsung_galaxy_s20_ultra_15.jpg
r/r 1978669: samsung_galaxy_s20_ultra_28.jpg
r/r 2752620: samsung_galaxy_s20_ultra_02.mp4
r/r 1978680: 4211112626.jpg
r/r 1978624: 6343296878.jpg
r/r 1978935: 6235050916.jpg
r/r 1978941: 0880771770.jpg
r/r 1966120: pliktekstowy2.txt
r/r 2752619: D19970553.pdf
r/r 1979239: plik1.txt
r/r 2100384: sample.pdf
r/r 2752636: D20221692Lj.pdf
r/r 2752637: szkolni_klasa2_podrecznik4.pdf
r/r 2752638: LibreOffice_7.4.2_Linux_x86-64_rpm.tar.gz
d/d 2100236: LibreOffice_7.4.2.3_Linux_x86-64_rpm
r/r 2752639: PDFStudio_linux64.sh
r/r 2100382: SarmataFlag.png
r/r 2100383: Untitled 1.docx
r/r 2100385: Untitled 1 in all.docx
[arek@fedora ewf] $ icat -o 2048 -b 512 ewf1 2100389 > ~/astudia/is/projekt/deleted_img.jpg
[arek@fedora ewf] $
```



Udało się pomyślnie odzyskać zdjęcie w formacie .jpg

## fIs

Jeśli chodzi o dalsze przeszukiwanie systemu plików za pomocą komendy **fIs** - ciekawy może okazać się folder /media. Znajdują się tam podfoldery używane do zamontowania obrazów np. pendrive'ów, cd rom'ów, a nawet dysków chmurowych

```
[arek@fedora ewf] $ fIs -b 512 -o 2048 ewf1 2228225
d/d 2228226:    sf_is
```

Tutaj zostało znalezione medium "sf\_is"

Niestety, wchodząc w jego numer i\_node nie dostajemy żadnego outputu

```
[arek@fedora ewf] $ fIs -b 512 -o 2048 ewf1 2228226
[arek@fedora ewf] $
```

Może być to spowodowane niepoprawnym skonfigurowaniem ów dysku przez użytkownika

## Odzyskiwanie hasła

Spróbujmy teraz odzyskać hasło, którego używały użytkownicy systemu  
Hash hasła znajdziemy w /etc/shadow

```
[arek@fedora ewf] $ icat -b 512 -o 2048 ewf1 2098403 > ~/astudia/is/projekt/shadow.txt
[arek@fedora ewf] $ cat shadow.txt
king-phisher:!:19212::::::
kali:$y$j9T$syJ4c33f2G3t4qhVR/geu.$0RGhUWfVibVvPWIP3hcZD.b859AGmMtdPyTvmc5tLxC:
19212:0:99999:7:::
debian-tor:!:19215::::::
```

oraz plik /etc/passwd

```
king-phisher:x:133:142::/var/lib/king-phisher:/usr/sbin/nologin
kali:x:1000:1000:,,,:/home/kali:/usr/bin/zsh
debian-tor:x:134:145::/var/lib/tor:/bin/false
```

```
(kali㉿kali)-[~]
$ grep -rn ENCRYPT_METHOD /etc/login.defs
277:# This variable is deprecated. You should use ENCRYPT_METHOD.
294:ENCRYPT_METHOD SHA512
297:# Only works if ENCRYPT_METHOD is set to SHA256 or SHA512.
314:# Only works if ENCRYPT_METHOD is set to YESCRYPT.
```

Trzeba jeszcze sprawdzić jakiego algorytmu jaki był algorytm haszujący użytkownika

Rozpoczynam działanie programu **hashcat**

```
[arek@fedora hashcat-6.2.6] $ ./hashcat.bin -m 1800 -a 0 -o ~/astudia/is/projekt/obraz/output.txt
~/astudia/is/projekt/obraz/hash.hash ~/wordlists/rockyou.txt
hashcat (v6.2.6) starting

Successfully initialized the NVIDIA main driver CUDA runtime library.
```

I za pomocą metody brute force + słownika udało mi się odzyskać jakże nieskomplikowane hasło - **kali**