

MICT1

EXERCISE WEEK 3

Joeri van Grimbergen (1244825) | Nursize Bilen (1260235) | Rick van Gorp (1328417)
ZUYD HOGESCHOOL | HEERLEN | GROEP 4

1. What we found and where we found it

Voor dit onderzoek is een script gebouwd in Python die in staat is om te bepalen of er sprake is van foutieve blocks en in staat is om toegevoegde random data blocks, zoals de nullen, te verwijderen uit het bestand. Alle handelingen die onderstaand zijn beschreven dienen in volgorde te worden uitgevoerd om te voorkomen dat de waardes van de grenzen niet langer kloppen. Het doel van dit onderzoek is om zoveel mogelijk data terug te halen uit het GIF-bestand.

Het resultaat bestaat uit het GIF-bestand *e_r.gif*, waarbij 11 frames hersteld zijn en het GIF-bestand *f_z.gif*, waarbij 7 frames hersteld zijn. Aan de hand van de gedefinieerde “magic value” GIF89a die als prefix is weergegeven in beide bestanden is vastgesteld dat het om twee GIF-bestanden gaat.

Onderstaande resultaten hebben betrekking op het bestand *e_r.data*.

```
Verified Values:
[['APPLIC', 781, 800], ['GRAPHIC', 800, 808], ['IMG', 808, 67132], ['GRAPHIC', 67132, 67140], ['IMG', 67140, 82000], ['GRAPHIC', 147254, 147262], ['IMG', 147262, 163914], ['GRAPHIC', 390449, 390457], ['IMG', 390457, 409669], ['GRAPHIC', 466416, 466424], ['IMG', 466424, 505449], ['GRAPHIC', 505449, 505457], ['IMG', 505457, 532605], ['GRAPHIC', 581464, 581472], ['IMG', 581472, 625730], ['GRAPHIC', 625730, 625738], ['IMG', 625738, 663565], ['GRAPHIC', 663565, 663573], ['IMG', 663573, 709300], ['GRAPHIC', 709300, 709308], ['IMG', 709308, 747861], ['GRAPHIC', 747861, 747869], ['IMG', 747869, 796348], ['GRAPHIC', 796348, 796356], ['IMG', 796356, 843366], ['GRAPHIC', 843366, 843374], ['IMG', 843374, 849197]]

Unallocated spaces (Boundaries):
[[82000, 147254], [163914, 390449], [409669, 466416], [532605, 581464]]

Broken Values (Boundaries):
[]

Problems found in IMG Blocks:
[]

More specific unallocated spaces: [[81919, 112640], [163839, 225280], [245759, 276480], [307199, 358400], [409599, 450560], [532479, 563200]]
Above spaces are filled by: b'\x00'

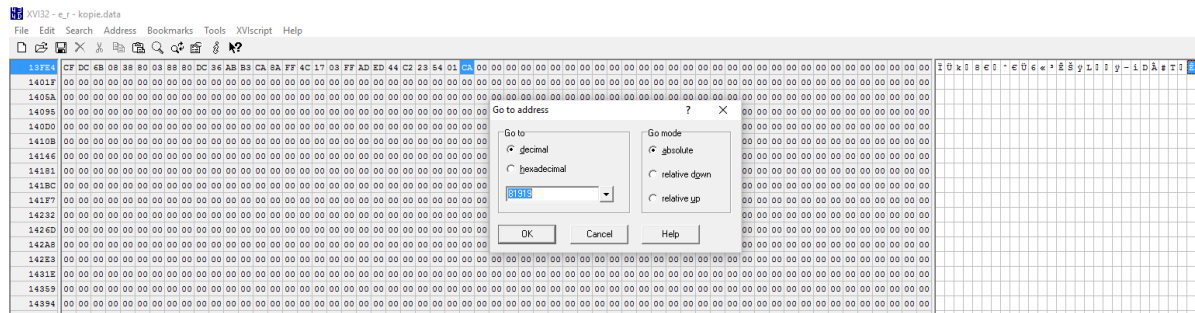
Start of block: [81919, 133119, 153599, 184319, 235519, 317439]

Press Enter to check again...
```

In bovenstaande afbeelding zijn de eerste resultaten van het Python script te zien. Onderstaand staan de betekenissen van de waardes beschreven:

- Verified values: De datablocks die zijn gevalideerd door de parser in het script;
- Unallocated spaces: De datablocks die zijn overgebleven en dus nog geen waarde toegewezen hebben gekregen;
- Broken Values: Alle datablocks die corrupte gegevens bevatten die tot nu toe zijn vastgesteld;
- Problems found in IMG Blocks: De grenzen van de corrupte datablocks in de image descriptor van een GIF-bestand;
- More specific unallocated spaces: Dit zijn de datablocks, waarbij is vastgesteld dat deze dezelfde data hebben. In “Above spaces are filled by” staat welke waarde dit is.
- Start of block: Dit zijn de beginpunten van de datablocks in het nieuwe bestand. Dit houdt in dat dit de beginpunten zijn geweest van de nulblokken. De waardes zijn aangepast naar het formaat van het nieuwe bestand.

In bovenstaande afbeelding worden offsets in decimalen weergegeven. Dit houdt dus in dat in het bestand op de genoemde locaties bij “More specific unallocated spaces” de grenzen te vinden zijn van de datablocks die gevuld zijn met byteobject \x00. Zie onderstaande afbeelding als voorbeeld:



Dit houdt dus specifiek in dat de volgende handelingen zijn uitgevoerd door het script:

Verwijderde datablocks:

Gevonden Data	Links block (offset dec)	Rechts block (offset dec)	Lengte (dec)
\x00	81919	112640	30721
\x00	163839	225280	61441
\x00	245759	276480	30721
\x00	307199	358400	51201
\x00	409599	450560	40961
\x00	532479	563200	30721

Aangezien de inhoud van deze datablocks gelijk was aan \x00, voldoet deze niet aan een structuur dat gedefinieerd is in de GIF-specificaties. Met deze reden zijn deze datablocks verwijderd.

Na de eerste bewerking om de nulblokken te verwijderen is het script nog eens uitgevoerd om te controleren of er nog fouten in het bestand staan:

```

Verified Values:
[['APPLIC', 781, 800], ['GRAPHIC', 800, 808], ['IMG', 808, 67132], ['GRAPHIC', 67132, 67140], ['IMG', 67140, 116534], ['GRAPHIC', 116534, 116542], ['IMG', 116542, 191276], ['GRAPHIC', 216369, 216377], ['IMG', 216377, 251376], ['GRAPHIC', 251376, 251384], ['IMG', 251384, 290409], ['GRAPHIC', 290409, 290417], ['IMG', 290417, 335704], ['GRAPHIC', 335704, 335712], ['IMG', 335712, 379970], ['GRAPHIC', 379970, 379978], ['IMG', 379978, 417805], ['GRAPHIC', 417805, 417813], ['IMG', 417813, 463540], ['GRAPHIC', 463540, 463548], ['IMG', 463548, 502101], ['GRAPHIC', 502101, 502109], ['IMG', 502109, 550588], ['GRAPHIC', 550588, 550596], ['IMG', 550596, 597606], ['GRAPHIC', 597606, 597614], ['IMG', 597614, 603437]]

Unallocated spaces (Boundaries):
[[191276, 216369]]

Broken Values (Boundaries):
[]

Problems found in IMG Blocks:
[[134930, 153618], [156980, 184372]]

Press Enter to check again...

```

Uit bovenstaande resultaten is op te maken dat er specifieke structuurproblemen zijn in een image descriptor block of meerdere image descriptor blocks nog structuurfouten zitten. Aan de hand van de offsets wordt bepaald waar de problematische datablokken zich bevinden.

De waarden die bovenstaand worden weergegeven zijn waarden die een indicatie aangeven waar het probleem zich ongeveer bevindt. Deze zijn gebaseerd op de doorbroken Data Block Size reeks die zich voordoet. Door te zoeken naar de grootste Data Block Size die in de buurt ligt (FF) kan middels de functie *Go To* (Jump width = 255 instellen of FF) worden bepaald waar de Data Blocks verbroken worden.

Op hex offset: 0x20812 is de laatste Data Block Size te vinden die voldoet aan de rest van de Data Block sizes in het datablock eronder.

Op hex offset: 0x20749 is de laatste Data Block Size te vinden die voldoet aan de rest van de Data Block sizes in het datablock erboven.

In dit onderdeel is het mogelijk dat óf de header van de image descriptor mist of er is data midden in het block overschreven. Uit de eerder weergegeven afbeeldingen kan worden opgemaakt dat de start van een nieuw block, namelijk bij decimale offset 133119 in de buurt ligt van dit foutieve block. Het is dus mogelijk dat er data mist of data overschreven is. Om de reeks van data block sizes kloppend te maken is de volgende bewerking uitgevoerd:

Verwijderde datablocks:

Gevonden Data	Links block (offset hex)	Rechts block (offset hex)	Lengte (dec)
Zie onderstaande afbeelding	20849	20911	200

2073F	64 E6 C1 07 0F 0C 76 B0 59 30 FF C7 4A 82 40 09 06 11 C0 61 03 C1 B2 02 6E A4 2D 78 45 40 16 01 3F 7E 14 BC F1 63 CA 14 82 6D DC E0 C1 A3 07 8F 19 37 59 DC 5A 71 D3 A7 CF 23 50 d
2077A	02 1E 11 7C C3 C5 4B 63 2F 02 8E 28 AD 3A 99 72 65 12 29 8E A0 39 44 89 D2 A4 41 83 26 51 E2 C4 69 04 AD 5A A3 38 75 1E F5 AA 96 B3 78 DD 9C DD 02 C4 E9 D6 AF 5A 5E E2 C2 25 B8
207B5	43 00 EF 9B 08 BB 08 40 23 80 09 90 2D 05 27 B1 FA 25 A0 D8 40 32 78 10 D6 22 28 4E 00 35 83 5D 0F 36 38 A8 4B 00 30 EE 79 CA 8A 10 61 C1 42 89 F2 E5 55 0A 68 A6 AE 9C C1 8A 05 C
207F0	2B AA 22 18 D2 20 2F 82 AB 58 57 16 30 0C 56 9E 41 04 74 90 52 98 E3 6C EE E8 07 3C A0 21 00 73 14 23 FF 1F F6 30 C1 0C 12 C2 12 65 1D C4 07 AC 43 61 58 68 B0 C2 27 88 A4 59 13 +
2082B	B0 C0 0B 70 64 10 12 08 20 81 3F F0 19 60 AA 50 05 23 94 21 2F 3F 38 D7 0E CB D0 05 3F A0 77 0B 5D 60 C2 11 FF 30 08 4A DC 6B 10 48 5A 62 91 10 E4 BA 14 92 45 31 88 01 CB ED E4 *
20866	60 99 CB 08 40 61 B7 53 08 82 04 20 46 26 55 86 61 05 F1 C2 20 8E 73 1C 01 9C 09 12 64 32 C8 1F FC 90 88 55 24 42 0F 7D 10 00 D0 FC A0 87 99 F4 66 17 B4 A0 49 41 48 A1 09 45 BC ^
208A1	C2 4F 9A 18 8E 22 3C B1 0B 5E 88 CF 91 38 83 81 43 04 A0 85 2B 58 E1 20 A0 C8 44 26 80 81 28 8F 78 0C 0B 69 30 84 41 EC D0 89 30 6C AA 08 8B B0 84 25 38 75 81 0C 04 50 00 04 38 Å
208DC	80 02 C1 42 81 83 44 A0 03 E1 F2 4F 3D EA A1 0D 01 6C 22 12 47 01 44 29 8A 01 0B 01 F0 01 1A D6 28 05 2C E4 50 82 52 30 47 10 50 98 83 0D A0 20 CD 12 22 24 42 55 FF C4 66 42 68 e
20917	C0 BA 12 88 A0 04 25 70 60 41 7E 30 4E 20 54 81 09 55 C8 41 19 78 78 06 23 FC F0 2F E8 64 42 19 E8 15 88 79 71 41 2F 5D E8 02 99 12 F1 C4 2E 78 C1 32 02 08 04 6A 4A F3 15 24 1C Å
20952	84 7C C5 02 4A 17 84 B0 03 21 18 44 8C 08 69 82 11 0A 1A BB 2D 9E 4E 62 B0 43 48 96 40 43 B0 31 FE EE 77 6A 7C CA 2D 7A 71 11 84 8C 89 0B 02 E0 80 0D B3 A0 07 3D EC 71 8F 7F 80 u
2098F	04 28 29 21 A7 41 DA 0F 38 5F 73 18 0B C3 88 4E 4F 73 17 8E 38 8C 1C 6F 08 BA 24 80 3A 20 64 49 14 77 84 07 2F 5F 78 43 0A 00 C8 18 0F EF 50 A8 8B 15 24 0A C1 10 A0 14 0A 83

Om de nieuwe locaties te bepalen wordt het script nogmaals uitgevoerd:

```
Verified Values:
[['APPLIC', 781, 800], ['GRAPHIC', 800, 808], ['IMG', 808, 67132], ['GRAPHIC', 67132, 67140], ['IMG', 67140, 116534], ['GRAPHIC', 116534, 116542], ['IMG', 116542, 191075], ['GRAPHIC', 216168, 216176], ['IMG', 216176, 251175], ['GRAPHIC', 251175, 251183], ['IMG', 251183, 290208], ['GRAPHIC', 290208, 290216], ['IMG', 290216, 335503], ['GRAPHIC', 335503, 335511], ['IMG', 335511, 379769], ['GRAPHIC', 379769, 379777], ['IMG', 379777, 417604], ['GRAPHIC', 417604, 417612], ['IMG', 417612, 463339], ['GRAPHIC', 463339, 463347], ['IMG', 463347, 501900], ['GRAPHIC', 501900, 501908], ['IMG', 501908, 550387], ['GRAPHIC', 550387, 550395], ['IMG', 550395, 597405], ['GRAPHIC', 597405, 597413], ['IMG', 597413, 603236]]

Unallocated spaces (Boundaries):
[[191075, 216168]]

Broken Values (Boundaries):
[]

Problems found in IMG Blocks:
[[156779, 184171]]

Press Enter to check again...
```

Ln: 57 Col: 0

Voor dit foutieve block geldt dezelfde werkwijze als hiervoor beschreven.

Op hex offset: 0x2576B is de laatste Data Block Size te vinden die voldoet aan de rest van de Data Block sizes in het datablock eronder.

Op hex offset: 0x25649 is de laatste Data Block Size te vinden die voldoet aan de rest van de Data Block sizes in het datablock erboven.

Ook in dit onderdeel is het mogelijk dat óf de header van image descriptor mist of er is data midden in het block overschreven. Om de reeks van data block sizes kloppend te maken is de volgende bewerking uitgevoerd:

Verwijderde datablocks:

Gevonden Data	Links block (offset hex)	Rechts block (offset hex)	Lengte (dec)
Zie onderstaande afbeelding	25749	2576A	33

2564C	CA 04 FA 5A 10 5F C0 C7 0D 61 0A 71 D5 63 99 C8 0C 6B F0 01 07 5E 10 40 EA 10 10 10 07 DC 10 0E E6 F2 09 9D 50 EB A6 D0 70 D2 8B C9 99 88 8C D4 A0 D5 CA 20 0A 39 45 7F FB A7
25687	AE 11 1B 1B 50 06 C4 50 0B 7C 10 67 E9 17 0A 1E 82 10 C8 50 31 5A 35 0B 7D E7 07 13 A3 38 22 1A 2B 2F A5 ED 78 25 0E C5 50 06 2E 23 55 80 50 05 28 C1 09 BF 30 09 D9 74 17 28
256C2	E4 66 51 DA 63 22 26 6C B1 E4 34 27 B5 9F 81 C1 24 30 CE 82 2B E5 82 AB 40 89 72 E5 0F CF 25 C9 6D 01 C0 15 DA 02 60 A9 C4 85 00 0F 20 5C 6F 01 17 51 B3 10 CD B9 17 27 20 5D
256FD	65 E6 22 4F 18 9B C6 17 39 A1 27 6D C0 5D 72 7B 1D 92 A1 DB 53 90 07 75 9E 07 6D 30 BB 77 94 EA B7 E6 46 94 D0 2A A3 A1 04 DB AC 78 CB 9D 02 9E 2D 02 5E D0 D2 AD 12 C5 AB 42
25748	5A 50 86 69 18 85 81 8E 2C 16 9B 35 3C 88 12 01 80 A1 54 E8 56 01 78 84 1A 82 11 7A DD 05 15 E2 C4 90 E2 57 78 2C 81 14 10 80 4F F0 A0 88 1E 02 33 3E D2 FF 24 45 08 78 3C 4C
25773	D8 00 58 20 07 74 D0 06 5D 00 69 84 C8 B4 95 B8 B4 95 E8 D9 1F 50 03 41 50 A3 55 3B 08 CA 75 CD 9A BE E9 79 DD E9 A6 D0 DE 84 70 9E 59 7B C7 12 08 DF A0 36 02 4A FD 01 20 48
257AE	E9 3C 02 49 85 A3 FE EE 64 C3 83 17 4F 86 EA 62 23 36 62 83 6F A2 40 81 1B 18 8A A1 28 AE 0D 00 02 49 82 0A 44 BA 9B 2E 4C 04 58 8E 8A 2E 28 E0 B4 3A 02 21 50 60 37 E5 C8 32
257E9	02 F3 DC 65 A8 40 82 DE 51 C9 8A 98 0A E8 11 D2 85 D9 D8 84 F0 84 41 00 BD 57 58 DB 76 9B 3D 4E D8 D5 59 08 85 AE A3 66 01 40 06 62 18 06 64 E8 05 5B 00 05 3D B0 82 62 DD 02

Na nogmaals uitvoeren van het script blijkt dat er nog een stuk unallocated space te vinden is:

```
Verified Values:
[['APPLIC', 781, 800], ['GRAPHIC', 800, 808], ['IMG', 808, 67132], ['GRAPHIC', 67132, 67140], ['IMG', 67140, 116534], ['GRAPHIC', 116534, 116542], ['IMG', 116542, 191041], ['GRAPHIC', 216134, 216142], ['IMG', 216142, 251141], ['GRAPHIC', 251141, 251149], ['IMG', 251149, 290174], ['GRAPHIC', 290174, 290182], ['IMG', 290182, 335469], ['GRAPHIC', 335469, 335477], ['IMG', 335477, 379735], ['GRAPHIC', 379735, 379743], ['IMG', 379743, 417570], ['GRAPHIC', 417570, 417578], ['IMG', 417578, 463305], ['GRAPHIC', 463305, 463313], ['IMG', 463313, 501866], ['GRAPHIC', 501866, 501874], ['IMG', 501874, 550353], ['GRAPHIC', 550353, 550361], ['IMG', 550361, 597371], ['GRAPHIC', 597371, 597379], ['IMG', 597379, 603202]]

Unallocated spaces (Boundaries):
[[191041, 216134]]

Broken Values (Boundaries):
[]

Problems found in IMG Blocks:
[]

Press Enter to check again...
```

Naar aanleiding van de hierboven gegeven informatie wordt de cursor verplaatst naar de eerder bepaalde locatie van de voormalige start van een nulblock. De waarde die het dichtst bij de beginwaarde van unallocated spaces ligt wordt gekozen. Dit is offset: 184319. Deze klopt echter niet meer aangezien er reeds waardes zijn verwijderd. Er moet dus boven deze offset worden gezocht naar veranderde Data Block Sizes. Hiervoor geldt dezelfde werkwijze als voor de andere twee vondsten.

Om de reeks van data block sizes kloppend te maken is de volgende bewerking uitgevoerd:

Verwijderde datablocks:

Gevonden Data	Links block (offset hex)	Rechts block (offset hex)	Lengte (dec)
Zie onderstaande afbeelding	2CF49	2CF61	24

F8	83	20	AC	02	02	05	30	C6	31	C0	37	5D	03	BC	AD	A7	AE	8F	22	91	14	8F	64	C6	85	FC	56	22	EB	38	C8	D2	D4	E4	28	B7	62	2D	92	5F	3F	B0	CE	3B	39	97	05	70	00	06	78	1F	FC	58	B6	CA	CB	C9
7F	9C	2E	7F	19	CC	5C	16	BF	55	88	DE	73	F7	97	B6	31	F4	61	A6	37	DF	97	66	32	97	F9	B2	00	64	6D	D3	E3	DA	BE	61	79	C6	61	92	09	86	B4	80	BC	24	AD	9D	72	AB	D1	DC	29	EC	50	82	D2	E4	09
D3	46	61	67	54	44	D4	0A	CB	2A	3E	C8	B9	20	E4	0A	AE	E0	0D	34	30	03	BB	2B	3E	F8	0E	08	E2	C4	EF	F8	44	0E	B0	26	BF	38	E1	81	2C	8A	13	30	EA	23	BA	80	00	83	A2	0B	A0	06	08	BE	80	2A	D6
E6	4A	0E	61	14	5E	E1	6D	9E	ED	D7	B6	22	4D	EE	C4	13	0E	A1	28	36	43	78	3A	40	04	76	A0	30	6E	E1	4F	52	6F	2D	AC	FF	01	72	C8	21	DC	28	0C	1C	4A	01	16	C4	02	DE	D4	62	C3	28	C5	A9	E0	C2
73	E8	C2	2A	0E	00	07	3C	40	00	08	E0	00	0E	20	76	A6	8E	79	D8	2A	5B	76	AA	12	12	50	00	70	30	AC	48	41	F9	96	6F	C7	6E	E3	AE	86	AC	36	EA	0A	37	E8	2C	E4	B0	CF	0F	89	63	01	E8	67	D4	8E
80	04	36	8B	E4	BE	6C	FC	04	20	95	A0	E9	FD	E6	03	01	10	A0	E7	F6	10	FE	EA	67	2D	06	04	E8	EC	03	40	08	46	E9	C4	A2	76	4C	E6	01	3A	A3	5A	AC	0E	D1	A4	86	00	BB	20	28	C4	22	B7	18	CD	0D
15	30	EC	B6	02	02	E5	50	45	94	40	D4	66	80	07	AC	6C	04	AC	E2	60	6E	A0	BB	3C	F0	80	84	E7	04	44	60	04	B6	2D	04	85	00	0D	9A	84	44	2C	6A	12	20	C8	A2	A4	82	23	B8	00	6A	9E	5E	EF	64	63
A7	28	E1	30	59	1E	5C	5A	EE	40	09	74	C0	02	34	04	5C	C6	65	07	50	48	15	98	41	35	DA	42	CB	D8	90	35	EA	25	5F	5E	5D	23	14	02	2A	82	82	26	68	63	CB	54	09	62	26	A4	6B	62	C8	43	4D	25	
50	C2	20	7C	81	86	40	CC	AF	31	61	84	AE	06	65	5E	26	AA	75	A6	67	4E	FF	21	11	B5	1E	8F	61	47	1A	AA	21	E2	5D	10	E2	2D	1F	D2	24	0D	2B	D0	02	08	39	83	33	D4	A6	57	A4	E1	84	65	C5	6E	DA
C4	9D	F9	A6	49	5C	C1	65	BC	0C	4D	24	A7	3B	E9	02	70	45	07	1F	EC	19	4D	3C	82	49	D6	44	A0	3D	27	76	9C	81	24	A8	12	A2	7C	C3	37	CC	E4	77	86	0A	4D	EC	A1	49	50	5A	4C	38	D9	2A	D4	42	32

Om te bepalen of er nog nieuwe fouten zijn opgetreden in het bestand wordt het script nogmaals uitgevoerd.

```
Verified Values:
[['APPLIC', 781, 800], ['GRAPHIC', 800, 808], ['IMG', 808, 67132], ['GRAPHIC', 67132, 67140], ['IMG', 67140, 116534], ['GRAPHIC', 116534, 116542], ['IMG', 116542, 216109], ['GRAPHIC', 216109, 216117], ['IMG', 216117, 251116], ['GRAPHIC', 251116, 251124], ['IMG', 251124, 290149], ['GRAPHIC', 290149, 290157], ['IMG', 290157, 335444], ['GRAPHIC', 335444, 335452], ['IMG', 335452, 379710], ['GRAPHIC', 379710, 379718], ['IMG', 379718, 417545], ['GRAPHIC', 417545, 417553], ['IMG', 417553, 463280], ['GRAPHIC', 463280, 463288], ['IMG', 463288, 501841], ['GRAPHIC', 501841, 501849], ['IMG', 501849, 550328], ['GRAPHIC', 550328, 550336], ['IMG', 550336, 597346], ['GRAPHIC', 597346, 597354], ['IMG', 597354, 603177]]

Unallocated spaces (Boundaries):
[]

Broken Values (Boundaries):
[]

Problems found in IMG Blocks:
[]
```

Hieruit blijkt dat er geen foutieve waardes meer zijn gevonden en dat het script geen unallocated spaces meer heeft vastgesteld in het bestand. Daarbij worden de lengtes van het begin van het bestand tot eerste waarde en laatste waarde tot einde van bestand niet meegenomen.

Het bestand is gecontroleerd op validiteit door meerdere tools zoals “Jeffrey's Exif Viewer” op <http://regex.info/exif.cgi> en <http://www.imagemagick.org/MagickStudio/scripts/MagickStudio.cgi>. Doordat ImageMagick is een bibliotheek die parsers en interpreters bevat om afbeeldingen te kunnen verwerken. Bij de eerste tool werd geen fout weergegeven, maar bij gebruik van de tweede tool werd de fout “425: Corrupt Image” weergegeven. Naar aanleiding hiervan is in Jeffrey's Exif Viewer gecontroleerd of er corrupte data op enkele frames stond. Er was geen duidelijke foutmelding zichtbaar in de Exif viewer, maar er stonden wel enkele frames tussen die image data misten. Als hypothese voor het vervolgonderzoek werd gesteld: ImageMagick is niet in staat zelf opvulling te geven aan missende image data. Om het bestand valide te maken voor alle parsers en interpreters zijn de frames 3 en 4 van bestand *e_r.data* verwijderd. Frames zijn te herkennen aan een Graphic Control Extension gevolgd door een Image Descriptor met Block Terminator.

Om deze wijzigingen door te voeren zijn de volgende handelingen in volgorde uitgevoerd:

1. Gezocht naar hex string: 00 21 F9 met XVI32.

Verwijderde datablocks:

Gevonden Data	Links block (offset hex)	Rechts block (offset hex)	Lengte (dec)
Zie bijlage: e_r_removed.txt (Block 1)	1C736	34C2C	99574
Zie bijlage: e_r_removed.txt (Block 2)	1C736	24FF4	35006

Na het verwijderen van beide frames bleek de GIF-afbeelding wel valide te zijn. Aangezien het format voldeed aan de specificatie van GIF, werden er geen fouten meer weergegeven door het python

script. De foutmelding die eerder via imagemagick werd weergegeven kwam niet meer voor. De herstellde afbeelding bestaat uit 11 functionerende frames.

Het bestand is te openen in een photo viewer en er wordt een animatie weergegeven. Er bevindt zich echter nog wat ruis in de afbeelding. In de uitvoer van het script onder “Verified Values” is te zien dat er één IMG block (Image Descriptor Block) veel groter is dan de rest van de Image Descriptors [*‘IMG’, 116542, 216109*]. Dit zou kunnen betekenen dat er een header en een einde van een Image Descriptor zijn weggevallen. Dit kan resulteren in het missen van bijvoorbeeld een frame of andere image data. Nadere uitleg staat in hoofdstuk 2. Zie de bijlage op github voor het resultaat.



Onderstaande resultaten hebben betrekking op het bestand *f_z.data*.

```
Verified Values:
[['GRAPHIC', 781, 789], ['APPLIC', 789, 808], ['IMG', 808, 76339], ['GRAPHIC', 76339, 76347], ['IMG', 76347, 102624], ['GRAPHIC', 172734, 172742], ['IMG', 172742, 204884], ['GRAPHIC', 273641, 273649], ['IMG', 273649, 307321], ['GRAPHIC', 374013, 374021], ['IMG', 374021, 409733], ['GRAPHIC', 449311, 449319], ['IMG', 449319, 471261], ['GRAPHIC', 545263, 545271], ['IMG', 545271, 593988], ['GRAPHIC', 640098, 640106], ['IMG', 640106, 715151], ['GRAPHIC', 715151, 715159], ['IMG', 715159, 791819], ['GRAPHIC', 791819, 791827], ['IMG', 791827, 866593], ['GRAPHIC', 866593, 866601], ['IMG', 866601, 942432]]

Unallocated spaces (Boundaries):
[[102624, 172734], [204884, 273641], [307321, 374013], [409733, 449311], [471261, 545263], [593988, 640098]]

Broken Values (Boundaries):
[]

Problems found in IMG Blocks:
[]

More specific unallocated spaces: [[102399, 122880], [204799, 245760], [307199, 327680], [348159, 368640], [409599, 430080], [471039, 491520], [511999, 532480], [593919, 614400]]
Above spaces are filled by: b'\x00'

Start of block: [102399, 184319, 245759, 266239, 307199, 348159, 368639, 430079]
```

De betekenissen van de waardes die hierboven zijn weergegeven zijn beschreven in het gedeelte van *e_r.data*. In bovenstaande afbeelding worden offsets in decimalen weergegeven. Dit houdt dus in dat in het bestand op de genoemde locaties bij “More specific unallocated spaces” de grenzen te vinden zijn van de datablocks die gevuld zijn met byteobject `\x00`.

Dit houdt dus specifiek in dat de volgende handelingen zijn uitgevoerd door het script:

Verwijderde datablocks:

Gevonden Data	Links block (offset dec)	Rechts block (offset dec)	Lengte (dec)
\x00	102399	122880	20481
\x00	204799	245760	40961
\x00	307199	327680	20481
\x00	348159	368640	20481
\x00	409599	430080	20481
\x00	471039	491520	20481

Aangezien de inhoud van deze datablocks gelijk was aan `\x00`, voldoet deze niet aan een structuur dat gedefinieerd is in de GIF-specificaties. Met deze reden zijn deze datablocks verwijderd.

Na de eerste bewerking om de nulblokken te verwijderen is het script nog eens uitgevoerd om te controleren of er nog fouten in het bestand staan:

```
Verified Values:
[['GRAPHIC', 781, 789], ['APPLIC', 789, 808], ['IMG', 808, 76339], ['GRAPHIC', 76339, 76347], ['IMG', 76347, 152254], ['GRAPHIC', 152254, 152262], ['IMG', 152262, 186646], ['GRAPHIC', 212201, 212209], ['IMG', 212209, 264713], ['GRAPHIC', 271613, 271621], ['GRAPHIC', 326431, 326439], ['IMG', 271621, 381423], ['IMG', 326439, 381423], ['GRAPHIC', 381423, 381431], ['IMG', 381431, 455778], ['GRAPHIC', 455778, 455786], ['IMG', 455786, 530831], ['GRAPHIC', 530831, 530839], ['IMG', 530839, 607499], ['GRAPHIC', 607499, 607507], ['IMG', 607507, 682273], ['GRAPHIC', 682273, 682281], ['IMG', 682281, 758112]]

Unallocated spaces (Boundaries):
[[186646, 212201], [264713, 271613], [271621, 326431]]

Broken Values (Boundaries):
[]

Problems found in IMG Blocks:
[[351440, 368780], [379410, 381195], [379410, 381195]]

Press Enter to check again...|
```

Uit bovenstaande resultaten is op te maken dat er specifieke structuurproblemen zijn in een image descriptor block of meerdere image descriptor blocks nog structuurfouten zitten. Aan de hand van de offsets wordt bepaald waar de problematische datablokken zich bevinden.

Om te bepalen of er sprake is van afbraak van Data Block Sizes is dezelfde werkwijze gehanteerd als bij *e_r.data*. Het verschil is echter dat in bestand *f_z.data* gebruik wordt gemaakt van *FE* als data block size en niet *FF*. Er zijn nog zes blocks waar potentiële fouten in zitten. Ook hier wordt de dichtstbijzijnde waarde gezocht per block om het begin vast te stellen. Deze wordt uit de eerste afbeelding gehaald van *f_z.data* bij “start of block”.

Voor unallocated space offset 186646, geldt dat offset 184319 de positie aangeeft van de voorheen door nullen gesplitste blocks. De resultaten bij het zoeken naar de juiste Data Block Size zijn de volgende:

Op hex offset: 0x2D003 is de laatste Data Block Size te vinden die voldoet aan de rest van de Data Block sizes in het datablock eronder.

Op hex offset: 0x2CF54 is de laatste Data Block Size te vinden die voldoet aan de rest van de Data Block sizes in het datablock erboven.

Om de reeks van data block sizes kloppend te maken is de volgende bewerking uitgevoerd:

Verwijderde datablocks:

Gevonden Data	Links block (offset hex)	Rechts block (offset hex)	Lengte (dec)
Zie onderstaande afbeelding	2D053	2D101	174

2CF86	0A 78 00 3C E8 72 AF 16 87 38 4E 4C B1 06 28 38 3E 07 40 A3 E3 64 88 F2 B2 6E 31 D2 68 DA D0 50 D4 D0 90 44 4C FD 9C 46 6C 01 71 D0 23 3D 03 0D BA F2 8C D5 90 B5 51 D5 1C E9 0A
2CF87	2C D9 86 64 E6 5A 86 1F 70 DE 9A 99 36 3C 20 83 6B 5D 05 12 A8 99 0C 98 03 93 09 81 39 70 3E 18 20 85 2C 90 5A E7 AB 2C 93 D1 36 D9 F5 CF 64 5D E5 66 FC CF E9 FC BD 10 19 01 E2
2CF88	00 B5 55 38 7B 51 5B FE 7B A1 4B 1F 48 C3 45 E9 4B 3A 5C 02 3E 08 83 BA AD 54 22 F8 12 88 FC 91 B8 0B 01 93 1C 03 07 A4 D4 CB FC 04 CC 8C CC 30 18 SC C3 E5 54 4C AD D4 C2 AD 4C
2D037	F2 02 12 0B E5 5B 12 80 01 52 98 2A AA 12 22 B2 02 8D 56 6D 55 1A A0 01 34 08 02 CC 1B 04 69 0C 5D 90 A1 9B DF 28 90 8C 71 0D 2B F0 00 0F 78 80 DB 48 86 61 13 8E E6 6C 99 DF 80
2D072	BD E8 18 15 2D 6D CE DA AB B4 50 3B 95 BB 12 B5 ED 98 00 68 ED 0E A5 01 CB F0 70 DA 6C F4 AF 76 4A 0F D9 88 90 08 A9 AC 45 BA EF FC D0 8F B2 C9 03 06 68 CF 51 10 04 0E 40 19 45
2D0AD	4B B6 63 43 A4 2A 3A 25 55 2A 27 0A 69 22 D2 08 1C 56 CA 10 0F 29 9C 0E 61 BF 17 A1 91 E4 E2 03 5A 50 9D 08 65 8C 7C 7B 48 85 65 8A AD C8 82 38 E0 01 76 6B 82 24 28 8B B2 E0 81
2D0B8	23 63 83 93 D4 DB 71 9B 2E 07 44 01 C8 90 9C 38 88 D0 20 E1 DB 16 E5 83 6B 0A FE 11 93 FD 1D 94 85 80 E2 39 90 99 EC 13 97 85 1E 89 03 30 45 38 4A 1A B4 C1 EC D1 C1 04 C3 15 8C
2D123	35 CA ED 01 5A 85 EA 00 A2 D4 27 29 B8 C1 18 6C 00 05 B8 99 55 51 E5 DF AD 4E 34 1D 4B 16 08 C3 2A 60 01 05 18 1A 05 18 8F AC 4D D3 27 F0 DA 25 70 CF 51 A0 C3 4B 83 C3 B1 75 05
2D15E	35 A4 D3 45 88 C3 61 20 D4 1D B9 E2 22 88 86 69 D8 82 54 60 BA 6C 90 85 3F 60 05 56 80 06 DC B1 81 BA 15 DC 7B C3 01 2A 79 60 21 D1 05 84 61 40 C4 AD DB 39 A6 E3 3A A6 63 47 9C
2D199	E3 CB 9C E3 C1 BD CC 38 A8 C4 BD BD 3B 31 20 BC D5 C4 2A 52 10 22 89 71 01 CA 2B 2B B3 7A 83 37 38 86 52 B2 02 5B 94 64 4F 58 5D 2B 88 9A DC F0 2C 0F 38 83 45 B3 82 68 E4 B4 D4

Om te bepalen of er nog nieuwe fouten zijn opgetreden in het bestand wordt het script nogmaals uitgevoerd.

```
Verified Values:
[['GRAPHIC', 781, 789], ['APPLIC', 789, 808], ['IMG', 808, 76339], ['GRAPHIC', 76339, 76
347], ['IMG', 76347, 152254], ['GRAPHIC', 152254, 152262], ['IMG', 152262, 212026], ['GR
APHIC', 212026, 212034], ['IMG', 212034, 264538], ['GRAPHIC', 271438, 271446], ['GRAPHIC
', 326256, 326264], ['IMG', 271446, 381248], ['IMG', 326264, 381248], ['GRAPHIC', 381248
, 381256], ['IMG', 381256, 455603], ['GRAPHIC', 455603, 455611], ['IMG', 455611, 530656]
, ['GRAPHIC', 530656, 530664], ['IMG', 530664, 607324], ['GRAPHIC', 607324, 607332], ['I
MG', 607332, 682098], ['GRAPHIC', 682098, 682106], ['IMG', 682106, 757937]]

Unallocated spaces (Boundaries):
[[264538, 271438], [271446, 326256]]

Broken Values (Boundaries):
[]

Problems found in IMG Blocks:
[[351265, 368605], [379235, 381020], [379235, 381020]]

Press Enter to check again...|
```

Voor unallocated space offset 264538, geldt dat offset 266239 de positie aangeeft van de voorheen door nullen gesplitste blocks. Deze waarde is inmiddels kleiner, omdat de data ervoor is aangepast. Het geeft echter een indicatie om in de buurt te zoeken. De resultaten bij het zoeken naar de juiste Data Block Size zijn de volgende:

Op hex offset: 0x42461 is de laatste Data Block Size te vinden die voldoet aan de rest van de Data Block sizes in het datablock eronder.

Op hex offset: 0x42315 is de laatste Data Block Size te vinden die voldoet aan de rest van de Data Block sizes in het datablock erboven.

Om de reeks van data block sizes kloppend te maken is de volgende bewerking uitgevoerd:

Verwijderde datablocks:

Gevonden Data	Links block (offset hex)	Rechts block (offset hex)	Lengte (dec)
Zie onderstaande afbeelding	42414	42460	76

423B2	4E 1F 28 84 EC A9 C0 19 9C C1 00 DC 83 15 6C 40 08 B4 00 04 40 6E 8A AC 1F E5 92 91 7F 58 4D 4B B4 44 CD B6 C4 7E B5 AC 95 02 D8 5A 6E C4 22 CA 86 1C 01 8C 8C 02 5A 6E 84 87 25
423BD	78 21 BB 62 C7 CC F8 6B C0 1E AC 5D AD 2B EB 8E C6 C0 B6 2E 76 80 6B 77 38 0C 76 F4 86 25 18 AC C3 04 AC EE BE EE BD 38 0E AF BD 4A DD 38 50 64 1B 80 C0 4D 34 6F 20 88 AC E5 71
42428	80 2F 60 AC 7E 31 98 F3 42 EF 04 E9 C6 C0 1E EF 75 F8 9C 6B 30 42 46 C6 B6 C7 26 59 93 05 2C F0 5E 47 2F 04 04 00 21 F9 04 04 08 00 00 00 2C 00 00 00 00 F4 01 DF 00 00 00 F2 00

Om te bepalen of er nog nieuwe fouten zijn opgetreden in het bestand wordt het script nogmaals uitgevoerd.

```
-----
Verified Values:
[['GRAPHIC', 781, 789], ['APPLIC', 789, 808], ['IMG', 808, 76339], ['GRAPHIC', 76339, 76347], ['IMG', 76347, 152254], ['GRAPHIC', 152254, 152262], ['IMG', 152262, 212026], ['GRAPHIC', 212026, 212034], ['IMG', 212034, 264538], ['GRAPHIC', 326179, 326187], ['IMG', 326187, 381171], ['GRAPHIC', 381171, 381179], ['IMG', 381179, 455526], ['GRAPHIC', 455526, 455534], ['IMG', 455534, 530579], ['GRAPHIC', 530579, 530587], ['IMG', 530587, 607247], ['GRAPHIC', 607247, 607255], ['IMG', 607255, 682021], ['GRAPHIC', 682021, 682029], ['IMG', 682029, 757860]]

Unallocated spaces (Boundaries):
[[264538, 326179]]

Broken Values (Boundaries):
[]

Problems found in IMG Blocks:
[[379158, 380943]]

Press Enter to check again...
```

Uit bovenstaande afbeelding blijkt dat dezelfde unallocated space nog steeds is opgenomen in het bestand. Dit houdt in dat er een andere unallocated space is opgelost in plaats van deze. Om de fout van deze unallocated space alsnog te vinden moet worden teruggezocht vanaf offset 245759 of de FE-reeks meerdere malen doorbroken wordt. De resultaten bij het zoeken naar de juiste Data Block Size zijn de volgende:

Op hex offset: 0x3BF79 is de laatste Data Block Size te vinden die voldoet aan de rest van de Data Block sizes in het datablock eronder.

Op hex offset: 0x3BECA is de laatste Data Block Size te vinden die voldoet aan de rest van de Data Block sizes in het datablock erboven.

Om de reeks van data block sizes kloppend te maken is de volgende bewerking uitgevoerd:

Verwijderde datablocks:

Gevonden Data	Links block (offset hex)	Rechts block (offset hex)	Lengte (dec)
Zie onderstaande afbeelding	3BFC9	3C077	174

3BF36	A7 E0 0A 58 C6 62 96 6A B5 A1 13 6F A0 C2 5A D8 52 07 53 50 01 0D 1E F0 00 05 9A E3 AC 8E 0F F8 E8 E2 A3 47 52 F8 B8 94 29 D5 9D 4C E9 AF F5 F0 2F 8E 53 DB B5 4E 46 ED 51 0E E2
3BF71	88 82 10 29 39 07 C8 1D FE 71 B0 02 0F C0 DB 5D 41 82 34 88 F3 5D 11 64 06 50 A8 29 88 04 59 59 89 22 AA D3 26 B4 83 5F EE 03 20 18 26 69 31 8A 46 0D 0C 36 38 13 52 F0 8B CB 2E
3BFAC	01 30 EC 21 FF D1 21 83 FC 97 B0 6B B2 1B 86 92 07 1D 55 C7 10 98 C8 B4 F4 81 C1 85 18 68 82 55 5D 0B 50 1D 75 CB 51 8A CD 40 48 54 D7 65 DE F5 ED 04 70 93 D1 59 C4 46 7C 44 DA
3BFE7	D8 00 63 E6 AA 64 CE A7 91 28 3C E1 D8 44 29 B0 CD E4 A0 9A C6 5B 2B E7 D0 DE F1 48 45 9E D1 8E E3 04 67 72 C6 EE E9 4E 2C 0D 89 10 BB 1A 87 EB 54 34 3A 7A 34 75 D8 2B E6 78 07
3C022	4C C3 6E AD D9 9A 09 81 24 74 50 10 4D 03 A4 B2 81 2C 04 07 FF EA 0F 9B 0D F9 10 C5 D2 90 71 28 80 53 28 78 BF A7 68 5C E5 70 D4 D0 DC 1A 1E 03 97 B2 78 50 70 F0 8B C7 55 F6 1F
3C05D	4D 83 5F FF 12 F7 FF 8E 07 08 1B 09 12 D8 48 14 E1 E0 8F 55 64 0C 91 A9 D0 B0 78 FE 05 42 84 20 12 A2 41 23 45 10 2B 03 20 0C 18 50 A3 06 27 4F 1F 6B 0C 68 E1 C0 41 8B 16 56 AC
3C098	3C 58 99 B2 C5 85 64 3A 1A 9C BB D0 40 C6 83 78 08 C2 DD 5C 31 41 DC 85 00 C9 10 08 45 90 2C C0 B7 6F C9 BE 21 F8 56 53 E7 B7 71 42 CF F1 4C D7 80 43 83 9B 32 14 88 13 A7 00 A7
3C0D3	0C 71 32 D2 88 4D B3 C2 C2 D9 34 E1 C0 AD 79 A7 28 0A 38 B8 BF 1C 34 D0 A1 43 9C 87 15 91 18 20 61 30 28 D2 52 35 7B 19 30 E0 36 E5 0B B7 7B 1B 54 A8 18 50 C0 91 9C 02 05 06 4A

```
Verified Values:
[['GRAPHIC', 781, 789], ['APPLIC', 789, 808], ['IMG', 808, 76339], ['GRAPHIC', 76339, 76347], ['IMG', 76347, 152254], ['GRAPHIC', 152254, 152262], ['IMG', 152262, 212026], ['GRAPHIC', 212026, 212034], ['GRAPHIC', 326004, 326012], ['IMG', 212034, 380996], ['IMG', 326012, 380996], ['GRAPHIC', 380996, 381004], ['IMG', 381004, 455351], ['GRAPHIC', 455351, 455359], ['IMG', 455359, 530404], ['GRAPHIC', 530404, 530412], ['IMG', 530412, 607072], ['GRAPHIC', 607072, 607080], ['IMG', 607080, 681846], ['GRAPHIC', 681846, 681854], ['IMG', 681854, 757685]]

Unallocated spaces (Boundaries):
[[212034, 326004]]

Broken Values (Boundaries):
[]

Problems found in IMG Blocks:
[[351013, 368353], [378983, 380768], [378983, 380768]]

Press Enter to check again...
```

Op hex offset: 0x4AE89 is de laatste Data Block Size te vinden die voldoet aan de rest van de Data Block sizes in het datablock eronder.

Om de reeks van data block sizes kloppend te maken is de volgende bewerking uitgevoerd:

Gevonden Data	Links block (offset hex)	Rechts block (offset hex)	Lengte (dec)
Zie onderstaande afbeelding	4AED9	4AF87	174

[illegible]

In bovenstaande bewerking wordt 31 **niet** meegenomen.

Om te bepalen of er nog nieuwe fouten zijn opgetreden in het bestand wordt het script nogmaals uitgevoerd.

```
Verified Values:
[['GRAPHIC', 781, 789], ['APPLIC', 789, 808], ['IMG', 808, 76339], ['GRAPHIC', 76339, 76
347], ['IMG', 76347, 152254], ['GRAPHIC', 152254, 152262], ['IMG', 152262, 212026], ['GR
APHIC', 212026, 212034], ['IMG', 212034, 325829], ['GRAPHIC', 325829, 325837], ['IMG', 3
25837, 380821], ['GRAPHIC', 380821, 380829], ['IMG', 380829, 455176], ['GRAPHIC', 455176
, 455184], ['IMG', 455184, 530229], ['GRAPHIC', 530229, 530237], ['IMG', 530237, 606897]
, ['GRAPHIC', 606897, 606905], ['IMG', 606905, 681671], ['GRAPHIC', 681671, 681679], ['I
MG', 681679, 757510]]

Unallocated spaces (Boundaries):
[]

Broken Values (Boundaries):
[]

Problems found in IMG Blocks:
[[378808, 380593]]

Press Enter to check again...
```

In een van de Image Descriptors zit nog een fout. Dezelfde onderzoekswijze wordt toegepast zoals hierboven beschreven met offset 368639.

Op hex offset: 0x59DE2 is de laatste Data Block Size te vinden die voldoet aan de rest van de Data Block sizes in het datablock eronder.

Op hex offset: 0x59D33 is de laatste Data Block Size te vinden die voldoet aan de rest van de Data Block sizes in het datablock erboven.

Om de reeks van data block sizes kloppend te maken is de volgende bewerking uitgevoerd:

Verwijderde datablocks:

Gevonden Data	Links block (offset hex)	Rechts block (offset hex)	Lengte (dec)
Zie onderstaande afbeelding	59E32	59EE0	174

59C0D	BB	BE	D5	16	9E	53	3F	39	8E	01	86	27	12	9E	42	5D	35	20	0D	44	14	BC	20	60	38	2F	60	51	74	F2	B3	78	50	5F	C1	65	58	30	E2	FE	26	2B	14	51	48	90	32	EA	F5	79	57	32	07	69	32	33	46	12	24
59C48	37	63	38	B7	77	82	3A	53	1E	DA	65	00	EE	21	7C	7B	CC	45	C5	CE	00	12	41	84	9A	90	5F	BE	CE	86	5E	34	2B	48	C8	E2	FC	C5	0E	68	21	02	EC	80	5B	AF	32	0C	C3	D0	62	46	AA	B8	F4	55	AA	42	66
59C83	58	06	58	58	30	CC	3F	86	C5	09	C0	E3	CC	FE	EE	3C	00	6F	1D	A8	B6	F0	C0	34	3C	BA	C3	2C	50	AF	3B	C0	34	2D	9A	B6	69	65	C6	4D	D7	B4	F5	CE	14	66	12	73	6C	F2	03	3E	A2	56	1D	A6	B6	11	20
59CB2	AB	1D	18	41	11	FF	40	16	AA	80	1A	F4	20	1B	64	41	0F	C2	E1	45	4E	11	87	BB	C6	1D	6E	AF	4E	9F	44	70	54	64	B6	3A	40	0A	BE	A4	23	E0	C4	42	2F	A0	47	5C	4D	89	E7	66	B5	52	00	1F	A0	E0	01
59CF9	C6	41	1D	AC	C8	1B	CA	C1	8A	B3	64	D9	8C	0D	FA	A0	71	36	77	0B	37	BB	E4	53	13	27	0A	2E	C0	E0	0E	CE	13	56	00	09	3C	41	96	C8	8D	72	AD	00	0E	B2	20	08	88	07	3F	27	92	A5	DE	AF	1B	93	FE
59D34	73	75	56	C2	BB	C0	4B	1C	C4	A1	22	B0	A0	73	D3	4B	9A	90	15	59	95	15	8C	1E	00	41	EA	49	CE	CC	73	7C	58	37	0D	04	34	12	D4	D3	A4	74	49	82	26	A8	DC	62	22	08	D0	C0	39	2B	60	04	DE	40	02
59D6F	F0	B3	FD	64	25	16	A6	20	0F	1E	60	38	89	37	4F	F0	44	61	17	16	01	2C	C3	9F	FE	C9	2D	D6	60	79	53	D0	B3	0A	23	1C	0E	63	06	59	CE	01	4A	F0	A0	0E	56	27	31	E3	32	6A	2C	24	45	3D	B8	40	13
59DAA	35	D1	00	9A	09	B2	3A	4D	B2	F6	E3	14	3A	01	02	4C	B1	13	A8	6C	12	44	2F	EB	BA	31	18	14	E1	17	4E	8F	34	09	C2	15	1B	72	9D	60	EF	24	56	EB	44	B2	C7	44	58	F3	45	0A	4E	43	50	44	FE	6E	62
59DE5	E4	6E	56	84	18	39	C4	B7	E0	66	1A	92	CF	19	E7	46	19	0D	25	37	0D	85	47	BE	61	AA	40	A3	0D	DF	50	08	78	EE	DB	3A	C3	33	A4	0A	32	0A	AC	C0	EC	6B	33	10	81	9D	D6	86	47	98	11	F9	92	01	BD
59E20	A0	29	1D	D3	F1	61	68	AA	D4	20	20	FD	C4	4B	EB	96	6B	76	18	8E	A0	A2	C7	52	DC	F3	22	D8	C9	B4	34	C4	F9	00	A5	F5	BE	67	51	F8	8F	4F	A6	6D	0D	78	AD	16	F1	F1	3E	55	C5	23	5D	22	00	6A	C2
59E5B	1C	6A	42	55	00	93	A0	30	52	1E	1E	2C	A0	3C	42	D2	E7	C3	6A	C5	7D	DE	A7	56	D2	81	56	94	C1	17	4A	50	25	15	60	05	9B	E2	90	D2	60	7F	FA	87	05	AC	A2	0A	64	F0	58	F4	61	06	96	C0	06	29	
59E96	41	27	B9	21	10	0A	C8	0A	CC	02	59	A6	A0	29	6B	B4	0B	32	41	0E	98	B2	E9	B8	01	0D	64	8A	81	8A	2A	BC	48	0F	FB	BE	2C	33	C4	2B	FB	BC	E2	CA	54	C7	0C	05	24	EC	4E	C7	FC	0E	26	0D	86	ED	
59ED1	D8	4E	08	66	61	AC	B4	20	0E	E9	90	16	F8	21	04	37	FE	C0	4B	F1	EE	AD	98	43	0C	FA	AC	CF	48	A0	CF	40	A0	CF	06	11	16	62	A0	4D	79	E0	AF	FC	D2	3B	B0	03	00	08	F3	05	32	81	66	8E	80	04	E8
59F0C	D2	39	E8	32	04	BC	D4	4F	2D	CD	63	80	43	13	39	51	32	ED	E8	8E	86	C6	B9	36	40	12	8C	06	F4	4C	51	BC	1C	40	12	A2	0C	DB	60	F1	F4	24	64	34	0D	50	42	64	A7	54	02	A0	FB	FE	4B	42	60	CF	B4
59F47	4C	64	D7	F8	53	6C	E4	B3	E0	2E	09	D8	5C	C4	35	8F	D1	B6	96	2F	19	7B	64	6E	E8	C6	18	7D	D1	44	1C	C0	0A	E0	A0	0F	B0	74	AC	EC	60	35	BA	4A	3C	A1	29	35	AC	EB	84	DA	AD	2E	92	E1	22	7C	51

Om te bepalen of er nog nieuwe fouten zijn opgetreden in het bestand wordt het script nogmaals uitgevoerd.

```
Verified Values:
[['GRAPHIC', 781, 789], ['APPLIC', 789, 808], ['IMG', 808, 76339], ['GRAPHIC', 76339, 76347], ['IMG', 76347, 152254], ['GRAPHIC', 152254, 152262], ['IMG', 152262, 212026], ['GRAPHIC', 212026, 212034], ['IMG', 212034, 325829], ['GRAPHIC', 325829, 325837], ['IMG', 325837, 380646], ['GRAPHIC', 380646, 380654], ['IMG', 380654, 455001], ['GRAPHIC', 455001, 455009], ['IMG', 455009, 530054], ['GRAPHIC', 530054, 530062], ['IMG', 530062, 606722], ['GRAPHIC', 606722, 606730], ['IMG', 606730, 681496], ['GRAPHIC', 681496, 681504], ['IMG', 681504, 757335]]

Unallocated spaces (Boundaries):
[]

Broken Values (Boundaries):
[]

Problems found in IMG Blocks:
[]
```

Hieruit blijkt dat er geen foutieve waardes meer zijn gevonden en dat het script geen unallocated spaces meer heeft vastgesteld in het bestand. Daarbij worden de lengtes van het begin van het bestand tot eerste waarde en laatste waarde tot einde van bestand niet meegenomen.

Er bevindt zich echter nog wat ruis in de afbeelding. In de uitvoer van het script onder “Verified Values” is te zien dat er één IMG block (Image Descriptor Block) veel groter is dan de rest van de Image Descriptors *['IMG', 212034, 325829]*. Dit zou kunnen betekenen dat er een header en een einde van een Image Descriptor zijn weggevallen. Dit kan resulteren in het missen van bijvoorbeeld een frame of andere image data. Nadere uitleg staat in hoofdstuk 2.

Het bestand is gecontroleerd op validiteit door meerdere tools zoals “Jeffrey's Exif Viewer” op <http://regex.info/exif.cgi> en <http://www.imagemagick.org/MagickStudio/scripts/MagickStudio.cgi>. Doordat ImageMagick is een bibliotheek die parsers en interpreters bevat om afbeeldingen te kunnen verwerken. Bij de eerste tool werd geen fout weergegeven, maar bij gebruik van de tweede tool werd de fout “425: Corrupt Image” weergegeven. Naar aanleiding hiervan is in Jeffrey's Exif Viewer gecontroleerd of er corrupte data op enkele frames stond. Er was geen duidelijke foutmelding zichtbaar in de Exif viewer, maar er stonden wel enkele frames tussen die image data misten. Om het bestand valide te maken voor alle parsers en interpreters zijn de frames 3, 4 en 5 van bestand *f_z.data* verwijderd. Daarnaast is de Graphic Control Header op positie 781 (decimaal) omgewisseld met de Application Extension op positie 789 (decimaal), omdat de Graphic Control Header invloed heeft op image data. De instellingen in de Graphic Control Header hebben invloed op het eerstvolgende block. Frames zijn te herkennen aan een Graphic Control Extension gevolgd door een Image Descriptor met Block Terminator.

Om deze wijzigingen door te voeren zijn de volgende handelingen in volgorde uitgevoerd:

1. Gezocht naar hex string: 00 21 F9 met XVI32.

Verwijderde datablocks:

Gevonden Data	Links block (offset hex)	Rechts block (offset hex)	Lengte (dec)
Zie bijlage: f_z_removed.txt (Block 1)	4F8C5	5CEE5	99574
Zie bijlage: f_z_removed.txt (Block 2)	33C39	4F8C4	35006
Zie bijlage: f_z_removed.txt (Block 3)	252BE	33C39	59771

Na het verwijderen van beide frames bleek de GIF-afbeelding wel valide te zijn. Aangezien het format voldeed aan de specificatie van GIF, werden er geen fouten meer weergegeven door het python script. De foutmelding die eerder via imagemagick werd weergegeven kwam niet meer voor. De herstelde afbeelding bestaat uit 11 functionerende frames.

Het bestand is te openen in een photo viewer en er wordt een animatie weergegeven. Zie de bijlage op github voor het resultaat.



2. How we found it

Om tot de hierboven beschreven resultaten te komen is er een Python script ontworpen door groep 4. In dit gedeelte zal beschreven worden hoe dit Python script tot stand is gekomen en het zal een korte handleiding bevatten om het Python script te bedienen.

2.1 Handelingen voor resultaten

In dit gedeelte worden de uitgevoerde handelingen beschreven om tot de hierboven beschreven resultaten te komen.

2.1.1 Python script

Om de resultaten in het Python script weer te geven is het Python script uitgevoerd. De bestanden *f_z.data* en *e_r.data* moet in dezelfde map staan als het Python script. In 2.3 Uitleg Python script wordt beschreven hoe het Python script is opgebouwd. In 2.2 Controle uitvoer Python script wordt de controle van de uitvoer van het Python script beschreven. Ter bediening van het script hoeft de gebruiker alleen gebruik te maken van de Enter toets en op meerdere plekken in het bestand de naam van het input-bestand in het script te wijzigen.

2.2 Controle uitvoer Python script & zoeken naar afwijkende data block sizes

De uitvoer van het Python script wordt gecontroleerd met een hex-editor (XVI32). In de hex-editor zijn de bestanden *f_z.data* en *e_r.data* geopend.

In het python script worden boundaries als resultaten gegenereerd. Deze boundaries worden gebruikt om de locaties vast te stellen van de problematische image descriptor blocks. De kans is groot dat er problemen optreden met de data in het bestand op de plekken waar de nulblocks verwijderd zijn. Daarom wordt de start waarde van deze blocks opgeslagen en wordt deze vergeleken met de waarde waar ongeveer de fout in zou moeten zitten. In de buurt van deze waardes is gezocht naar een afwijking in de data block size.

Bij afbeelding *e_r.data* werd voornamelijk gebruik gemaakt van *FF* als data block size, terwijl bij afbeelding *f_z.data* voornamelijk gebruik werd gemaakt van *FE* als data block size. Aangezien deze twee waardes bekend zijn, is er gezocht naar deze waardes in het door het script gespecificeerde block. Om te controleren of de data block sizes overeenkomen, kan gebruik worden gemaakt van de *Jump/Goto* functie in XVI32. Deze functie is in staat om vanaf de offset van één *FF* te springen naar de volgende, volgens GIF-specificaties gedefinieerde, data size block. Als deze meermaals afwijkt van de waarde *FF* dan kan het zo zijn dat er foutieve data in staat. Daarom wordt rond het punt waar de laatste kloppende *FF* is gevonden gezocht naar een andere *FF* die wel klopt in het logische block dat volgt. Dit was het geval en dus kon er een stuk van de data tussen deze *FF*'s verwijderd worden om de lengte tussen deze specificaties kloppend te maken.

Uit enkele sizes van image descriptors was op te maken dat deze veel groter waren dan de rest van de image descriptors. Binnen deze image descriptors waren dan ook fouten opgetreden. Om deze fout te corrigeren is er een block data verwijderd. Aangezien deze image descriptors veel groter waren, bestaat de mogelijkheid dat de image descriptor header van een andere image descriptor is overschreven, waardoor het block dat volgt niet meer wordt herkend als image descriptor. In deze gevallen is de image data dan ook samen gevoegd.

2.3 Uitleg Python script

Het Python script is te vinden in de bijbehorende Github-repository van groep 4. Onderstaand worden delen van het script kort uitgelegd.

```
from struct import *
import sys
import os
import binascii
import re
import random
```

In bovenstaande afbeeldingen zijn enkele libraries weergegeven waarvan gebruik wordt gemaakt in het script. Struct is een library om binaire data te herkennen als unsigned integer, unsigned short etc. Sys wordt gebruikt om bestandspaden nader te definiëren. Os wordt gebruikt om de grootte van een bestand te kunnen bepalen. Binascii wordt gebruikt om te converteren van binaire data naar hexadecimaal of omgekeerd. Re wordt gebruikt om data in een bestand te zoeken op basis van een regex string. Random wordt op dit moment niet gebruikt, het was een module die gebruikt zou worden om data te ordenen op basis van random gegenereerde nummers.

```
#GIF Header definitie
intDefinition = 6 #Signature + Version = Length 6 Bytes

#Logical screen descriptor chunk
#Should be read as little endian
intWidth = 2
intHeight = 2
intPacked = 1
intBackgroundColorandAspect = 2
```

In bovenstaande afbeelding zijn voor enkele waardes zoals de GIF header en de Logical Screen Descriptor beschreven uit welke elementen ze bestaan en wat de grootte in bytes is van deze elementen.

```
#Function for finding left boundary for specific data block
def CheckBlockLeft(f, currentPos, strBinary):
    blnEndOfData = False
    count = 1
    while blnEndOfData == False:
        f.seek(currentPos - count)
        if file.read(1) == strBinary:
            count = count + 1
        else:
            blnEndOfData = True
            if count == 1:
                return False
            else:
                return (f.tell()-1)
```

In bovenstaande afbeelding wordt de functie *CheckBlockLeft* weergegeven. Dit is een functie die aan de hand van ingevoerde data de linkergrens van een block met dezelfde ingevoerde data kan vaststellen en terugsturen. De functie *CheckBlockRight* wordt gebruikt voor dezelfde doeleinden, maar dan voor de rechtergrens.

```

#Function to check the validity of the image descriptor block
def CheckIMGDescriptor(f, currentPos, MaxWidth, MaxHeight):
    f.seek(currentPos)
    f.read(5)
    arrWrongBoundaries = []
    intWidth = unpack("<H", f.read(2))[0]
    intHeight = unpack("<H", f.read(2))[0]
    if (intWidth > MaxWidth or intHeight > MaxHeight) or (intWidth == 0 and intHeight == 0):
        return 0, f.tell(), arrWrongBoundaries
    intPacked = str(bin(unpack("<B", f.read(1))[0]))
    arrPacked = list(intPacked[2:len(intPacked)])
    if arrPacked[0] == "1":
        intColorTable = int(intPacked[len(intPacked)-3:len(intPacked)], 2)
        intSizeLocalColor = 3*2**(intColorTable+1)
        LocalColor = f.read(intSizeLocalColor)
    f.read(1)
    blnTerminated = False
    count = 0
    arrDBSize = [] #Used to define biggest size used, mostly only a few blocks differ from this
    while blnTerminated == False:
        blockPos = f.tell()
        intDBSize = unpack("B", f.read(1))[0]
        if intDBSize <= 255 and intDBSize > 0: #It is very unlikely for image data to be 0.
            f.read(intDBSize)
            arrDBSize.append(intDBSize)
            if intDBSize < max(arrDBSize):
                arrWrongBoundaries.append(blockPos)
                arrWrongBoundaries.append(f.tell())
            intBlockTerm = unpack("B", f.read(1))[0]
            if intBlockTerm == 0:
                blnTerminated = True
                return 2, f.tell(), arrWrongBoundaries
            else:
                f.seek(f.tell()-1)
        else:
            return 2, f.tell(), arrWrongBoundaries

```

In bovenstaande afbeelding wordt de functie *CheckIMGDescriptor* weergegeven. Deze functie wordt gebruikt om de validiteit van de Image Descriptor te controleren. Er wordt gecontroleerd op de volgende factoren:

- Zijn de Width en Height groter dan de Width en Height gedefinieerd in de Logical Screen Descriptor?
- Is er sprake van een local color table?
- Is de gespecificeerde data block size kleiner of gelijk aan 255 of groter dan 0?
- Is de gespecificeerde data block size in overeenkomst met de grootste waarde die voorkomt in de lijst vorige data block sizes?

```

#Function to check the validity of the Application Extension header
def CheckApplicChunk(f, currentPos):
    f.seek(currentPos+2)
    intBlockSize = unpack("B", f.read(1))[0]
    f.read(11)
    intDBSize = unpack("B", f.read(1))[0]
    strData = f.read(intDBSize)
    intBlockTerm = unpack("B", f.read(1))[0]
    if intBlockSize == 11 and intBlockTerm == 0:
        return 2, f.tell()
    else:
        return 0, f.tell()

```

In bovenstaande afbeelding wordt de functie *CheckApplicChunk* weergegeven. In deze functie wordt gecontroleerd of de Application Extension Chunk valide is. Dit wordt gedaan aan de hand van de *fixed* blocksize die altijd 11 moet zijn en te controleren of de Block terminator op de juiste

locatie staat. Hetzelfde wordt ook uitgevoerd in de functies *CheckPlainChunk*, *CheckGraphicChunk* en *CheckCommentChunk* met fixed blocksizes.

```
blnSolved = False
arrNew = []

while blnSolved == False:
    file = open("f_z.data", "rb")
    content = file.read()

    file.seek(0)
    magic_val = file.read(6)
    strWidth = unpack("<H",file.read(2))[0]
    strHeight = unpack("<H",file.read(2))[0]

    arrKnownValues = [{"APPLIC", b"\x21\xff"}, {"GRAPHIC", b"\x21\xf9"}, {"COMMENT", b"\x21xfe"}, {"PLAIN", b"\x21\x01"}, {"IMG", b"\x00\x2c"}, {"TRAILER", b"\x3b"}]
    arrPossibleValues = []
    arrVerifiedValues = []
    arrBrokenValues = []
    arrUnallocated = []
    arrDataBlockBounds = []
```

Uit bovenstaande afbeelding blijkt dat er een *While-Loop* wordt uitgevoerd na de functie definities. Dit geldt voor de rest van de data. Deze *While-Loop* wordt gebruikt om bijvoorbeeld het bestand nogmaals te controleren op fouten. In dit gedeelte wordt ook gedefinieerd welk bestand in binaire leesmodus geopend moet worden. Vervolgens worden de eerste paar waardes van de GIF-header gelezen en opgeslagen. In het array *arrKnownValues* worden de fixed identifiers voor alle bekende blocks gedefinieerd. Deze komen uit de tabel op <https://www.w3.org/Graphics/GIF/spec-gif89a.txt>. Vervolgens worden er nog vier arrays gemaakt om waardes in op te slaan.

```
#Determine possible locations of specific headers
for i in range(0, len(arrKnownValues)):
    for m in re.finditer(arrKnownValues[i][1], content):
        arrPossibleValues.append([arrKnownValues[i][0], m.start()])
```

Aan de hand van de bekende data uit *arrKnownValues* wordt in het geopende bestand gezocht naar deze specifieke prefixes. Deze waardes worden toegewezen aan het array *arrPossibleValues*.

```

#Take action based on the found data
for i in range(0,len(arrPossibleValues)):
    if arrPossibleValues[i][0] == "APPLIC":
        test = CheckApplicChunk(file, arrPossibleValues[i][1])
        if test[0] != 0:
            # = Application block
            arrVerifiedValues.append([arrPossibleValues[i][0], arrPossibleValues[i][1], test[1]])
    elif arrPossibleValues[i][0] == "GRAPHIC":
        test = CheckGraphicChunk(file, arrPossibleValues[i][1])
        if test[0] != 0:
            # = Graphic Control Extension
            arrVerifiedValues.append([arrPossibleValues[i][0], arrPossibleValues[i][1], test[1]])
    elif arrPossibleValues[i][0] == "COMMENT":
        test = CheckCommentChunk(file, arrPossibleValues[i][1])
        if test[0] != 0:
            # = Comment Extension
            arrVerifiedValues.append([arrPossibleValues[i][0], arrPossibleValues[i][1], test[1]])
    elif arrPossibleValues[i][0] == "PLAIN":
        test = CheckPlainChunk(file, arrPossibleValues[i][1])
        if test[0] != 0:
            # = Plain Text Extension
            arrVerifiedValues.append([arrPossibleValues[i][0], arrPossibleValues[i][1], test[1]])
    elif arrPossibleValues[i][0] == "IMG":
        arrPossibleValues[i][1] = arrPossibleValues[i][1] + 1
        test = CheckIMGDescriptor(file, arrPossibleValues[i][1], strWidth, strHeight)
        if test[0] == 1:
            # = Broken Image Descriptor
            arrBrokenValues.append(arrPossibleValues[i])
        elif test[0] == 2:
            # = Image Descriptor
            arrVerifiedValues.append([arrPossibleValues[i][0], arrPossibleValues[i][1], test[1]])

    if isinstance(test[2], list):
        if len(test[2]) > 5:
            i = 1
            while i < len(test[2])-1:
                if test[2][i+1]-test[2][i] > 0:
                    arrDataBlockBounds.append([test[2][i],test[2][i+1]])
                i = i + 2

```

Voor alle waarden die zijn toegewezen aan *arrPossibleValues* wordt gecontroleerd of het block dat erachter zit daadwerkelijk het type block is die hoort bij de gespecificeerde identifier. Als dit het geval is worden de naam, linkergrens en rechtergrens van het datablock toegewezen aan het array *arrVerifiedValues*. Nadat dit is uitgevoerd wordt specifiek voor het image descriptor block extra gecontroleerd of de teruggegeven waarde van de functie *CheckIMGDescriptor* een list is. Als deze ook groter dan 5 is worden de linkergrens en rechtergrens van de foutieve data in de image descriptor toegewezen aan het array *arrDataBlockBounds*.

```

arrVerifiedValues = sorted(arrVerifiedValues, key=lambda x: x[2])
print("Verified Values:\n" + str(arrVerifiedValues))

```

Alle geverifieerde waarden worden gesorteerd op basis van de tweede waarde in het 2D-array. Dit is de rechtergrens (van klein naar groot). Deze waarden worden vervolgens weergegeven aan de gebruiker.

```

for i in range(0, len(arrVerifiedValues)-1):
    if (int(arrVerifiedValues[i+1][1]) - int(arrVerifiedValues[i][2])) > 0:
        arrUnallocated.append([arrVerifiedValues[i][2], arrVerifiedValues[i+1][1]])
print("\nUnallocated spaces (Boundaries):\n" + str(arrUnallocated))
print("\nBroken Values (Boundaries):\n" + str(arrBrokenValues))
print("\nProblems found in IMG Blocks:\n" + str(arrDataBlockBounds))

```

Uit bovenstaande afbeelding blijkt dat de grenzen van de unallocated spaces worden vastgesteld. Deze worden weergegeven aan de gebruiker, samen met de broken values en grenzen van de foutieve blocks in de image descriptor.

```

if not arrUnallocated and not arrBrokenValues and not arrDataBlockBounds:
    blnSolved = True
elif arrNew:
    #if arrBrokenValues:
    #    strResult = Reorder(file, arrNew, brokenVal = arrBrokenValues)
    #else:
    #    strResult = Reorder(file, arrNew)
    #file.close()
    #file = open("e_r.data", "r+b")
    #file.seek(arrNew[0])
    #file.write(strResult)
    #file.close()
else:
    #Search for same character (filled with zeroes) and determine blocks
    arrSameChar = []
    for i in range(0, len(arrUnallocated)):
        file.seek(arrUnallocated[i][0])
        strTest = file.read(1)
        blnEndOfBlock = False
        count = 0
        while blnEndOfBlock == False:
            intLeftPos = CheckBlockLeft(file, arrUnallocated[i][0] + count, strTest)
            if intLeftPos != False:
                intRightPos = CheckBlockRight(file, arrUnallocated[i][0] + count, strTest)
                if intRightPos != False:
                    if [intLeftPos, intRightPos] not in arrSameChar:
                        arrSameChar.append([intLeftPos, intRightPos])
            count=count+10000 #check with steps of 10000 whether there is another block with same value in the same unallocated block
        if count > (300000):
            blnEndOfBlock = True

```

In bovenstaande afbeelding is een *if-statement* weergegeven. Deze statement bepaald of er waardes zijn in de arrays *arrUnallocated*, *arrBrokenValues* of *arrDataBlockBounds*. Als dit niet het geval is wordt de Boolean *blnSolved* die de *While-Loop* aan de gang houdt veranderd naar true. Als dit wel het geval is wordt bepaald of er grote blocks data uit het array *arrUnallocated* gelijk zijn aan elkaar. Hierbij worden de functies *CheckBlockLeft* en *CheckBlockRight* gebruikt. Deze waardes worden opgeslagen in het array *arrSameChar*.

```

#If blocks with same character found, continue to write edited data to file
if arrSameChar != []:
    file.seek(0)
    strFileSize = os.path.getsize("f_z.data")
    data = file.read(arrSameChar[0][0]+1)
    arrBlockMarkers = []
    if len(arrSameChar)==1:
        file.seek(arrSameChar[0][1])
    else:
        for i in range(0,len(arrSameChar)-1):
            file.seek(arrSameChar[i][1])
            data = data + file.read(arrSameChar[i+1][0]-(arrSameChar[i][1]-1))
            arrBlockMarkers.append(arrSameChar[i+1][0]-(arrSameChar[i][1]-1))
    arrNew.append(arrSameChar[0][0])
    for i in range(0,len(arrBlockMarkers)):
        arrNew.append(arrNew[i] + arrBlockMarkers[i])
    print("\nMore specific unallocated spaces: " + str(arrSameChar))
    print("Above spaces are filled by: " + str(strTest))
    print("\nStart of block: " + str(arrNew))

    try:
        file.seek(arrSameChar[len(arrSameChar)-1][1])
        data = data + file.read(strFileSize)
    except:
        print("EOF")
    file.close()
    f = open("f_z.data", "wb")
    f.write(data)
    f.close()
    file.close()
    x = input("\nPress Enter to check again...")

```

Als er data is gespecificeerd in het array *arrSameChar* wordt de betreffende data in één variabele geplaatst (*data*). Daarnaast worden de specifiekere unallocated spaces aangegeven uit *arrSameChar*. Deze heeft namelijk door de functies *CheckBlockRight* en *CheckBlockLeft* de nauwkeurige grenzen van de nulblokken kunnen vaststellen. Daarnaast worden ook de waarden weergegeven van de starts van de blokken (met nulwaarden) met behulp van *arrNew*.

De data die weggelaten moet worden wordt niet opgenomen in de variabele *data*. De gegevens die gedefinieerd zijn in variabele *data* worden vervolgens naar hetzelfde bestand als het input bestand geschreven. Vervolgens wordt de optie aangeboden om het script nogmaals uit te voeren door op Enter te drukken. Dit kan bijvoorbeeld gebruikt worden om te controleren of er meer fouten zitten in de blocks van een afbeelding.