Reversing Undocumented File Formats

Using a Hex Editor and your Brain

Reverse What?

00000000	0F13	0DDA	32A2	AB23	342E	D49E	EE85	0AFA	B2BC	1404	9108	ADB4	B947	1C59	0509	A400
00000020	92AB	62AC	2D6D	6C2E	90C4	D13B	2B07	7CA4	396B	49DA	6293	DD18	AB82	9A62	5701	AD96
00000040	FB47	E985	4648	0353	6206	4041	02D8	D85E	B3F4	6CD8	323C	6368	89EE	0FD2	AB9E	957F
00000060	B92F	4F65	85B3	2747	CAB0	237E	177D	5F9C	28C6	7C9A	2601	D45E	BD3D	A6E9	EAB7	E738
00000080	73C1	A955	EA03	776D	0F49	A9E4	D8E7	A7BB	05F1	6DB9	E6A2	DAF6	88C9	5538	8329	91D4
000000A0	4020	33B6	739E	DBFE	AA93	83CF	7C14	4D56	1855	E77F	D156	40A3	8138	7192	98DE	3D1E
000000C0	98CC	D5BD	FB3F	0A7C	D406	BE58	C2AA	9005	E51A	BE18	FA58	058D	B1ED	CA8D	AE28	E3F3
000000E0	65CC	485B	06F1	57EF	F1ED	AD97	910F	F507	AE29	F0A9	96B4	ACB3	443B	44FC	67BD	BA8C
00000100	ABD6	E1CC	4204	7551	0322	E878	1F4B	7C9A	EE11	3197	C45A	1BB1	24D8	2462	220C	30A4
00000120	58BE	46FD	A46D	3C07	98A1	C980	9DCD	6437	A356	2771	DED6	2EF7	94EA	9A8A	8F7D	F664
00000140	4972	6FB7	D1BC	F8EE	C0F1	1C86	43E5	3BBE	E72C	66BE	E5E7	07D4	7A2D	A139	6C95	89EC
00000160	9965	722E	20D8	86D0	8D61	8052	1C10	7FB8	E9E4	C9A7	B240	E015	C1C7	0462	C9D7	645C
00000180	91E8	50CA	7EF1	3457	F17B	D141	7438	C400	1D0C	6422	3C02	AAEE	01ED	711E	F224	FB9C
000001A0	2D73	1965	DAF0	663D	0584	BAD7	735D	6171	066A	77A0	27C6	1700	8293	62BE	2A65	5927
000001C0	FF72	A340	D392	34C6	6C25	EB64	F75C	2424	7C64	C0F5	1E96	9F3A	675F	E07A	ABBF	84E5
000001E0	753B	A846	74E6	AC4F	9436	EC10	AAA7	0F1F	E9CC	3A49	6BB1	F757	4B9F	219E	5F0A	A54C
00000200	71D0	EF98	534D	A5BE	8B69	9280	3E5A	9C60	A40E	969C	72CF	BB84	E4E7	28A5	A1A0	90EB
00000220	2599	1875	A675	9233	C33A	E7CA	9152	3657	185C	D585	24BB	42FC	5CFF	F8A5	2667	B045
00000240	B7A5	FAA6	5B77	6B66	026E	01FD	0E3F	ADF5	B86A	569E	93D0	A901	A931	BFE3	878D	ECBD
00000260	5AC2	BFD5	89B6	10B5	6B3C	9C1F	3CBD	BB30	94A1	C2BE	262F	EEA4	7A80	3CE4	2A71	20C0
00000280	827C	0903	5643	09DF	3582	394D	8A6D	6952	B8A7	D32A	FEDD	3CD0	74B5	2CF8	1C8E	C0C8
000002A0	65AC	83CF	BFDC	861D	93D6	EC9B	7CB9	4DBE	0391	0432	2821	ECB9	30A9	BC76	B49F	9657
000002C0	FE77	1445	CA17	060E	EDCE	3112	8CE3	3738	B459	D54D	F63E	D0DB	2CD7	9EE9	B3F2	F508
000002E0	1023	1926	512D	3479	70E7	5FC2	C560	7A38	9E6C	1975	E284	2C42	309A	E02C	1DF6	1189
00000300	2379	CCA6	544D	976F	DED3	2AA8	0F5A	B14F	AF19	EF79	6D19	FE10	C294	75F0	A842	AF23
00000320	62C1	8D2E	8031	D8EC	1877	0AEB	2BD4	C158	62BF	0661	13AF	AFE8	4DCD	DFF0	A283	87BD
00000340	2A60	0F6C	90B3	A356	71A1	1602	1B8D	48AB	6F45	CF7B	B3B3	2C5B	B3E0	0B95	145C	BA26
00000360	7025	A565	D9B1	2B09	1060	B414	598B	DF3F	BDAC	814C	D5C5	9B91	1301	2331	06FF	B411

Why use a Hex Editor and not a Disassembler?

- When the program that reads the File is not accessible(i.e. Firmware)
- When the program is in your possession, but is too hard to reverse(packed, protected etc.)
- When you don't know how to reverse executable files, but still want to reverse a file format
- Sometimes, reversing a file itself is faster than reversing the executable file(Game Archives)

Methodology

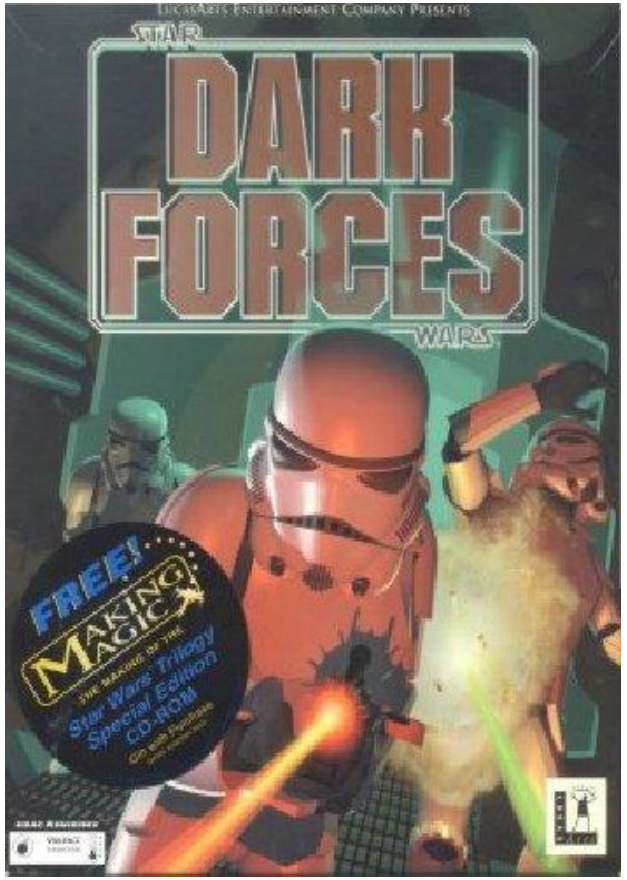
- Search Wotsit.org
- Study the general structures of File Formats
- Using a Hex Editor, try to match the structure of the data with the structures you already know
- "Look" at the Hex Bytes and try to find patterns
- Make Assumptions and verify
- If the theory proves to be correct, make more assumptions based on the new facts until the whole(or part) of the file format is revealed

Real World Reversing

- Case I : Cannot write but readable
- Case II : readable/writable
- Case III : Only 1 sample, cannot read/write

("can read/write" means the program that reads/writes to the file is accessible)

Case I : Reversing a Game Archive File Format



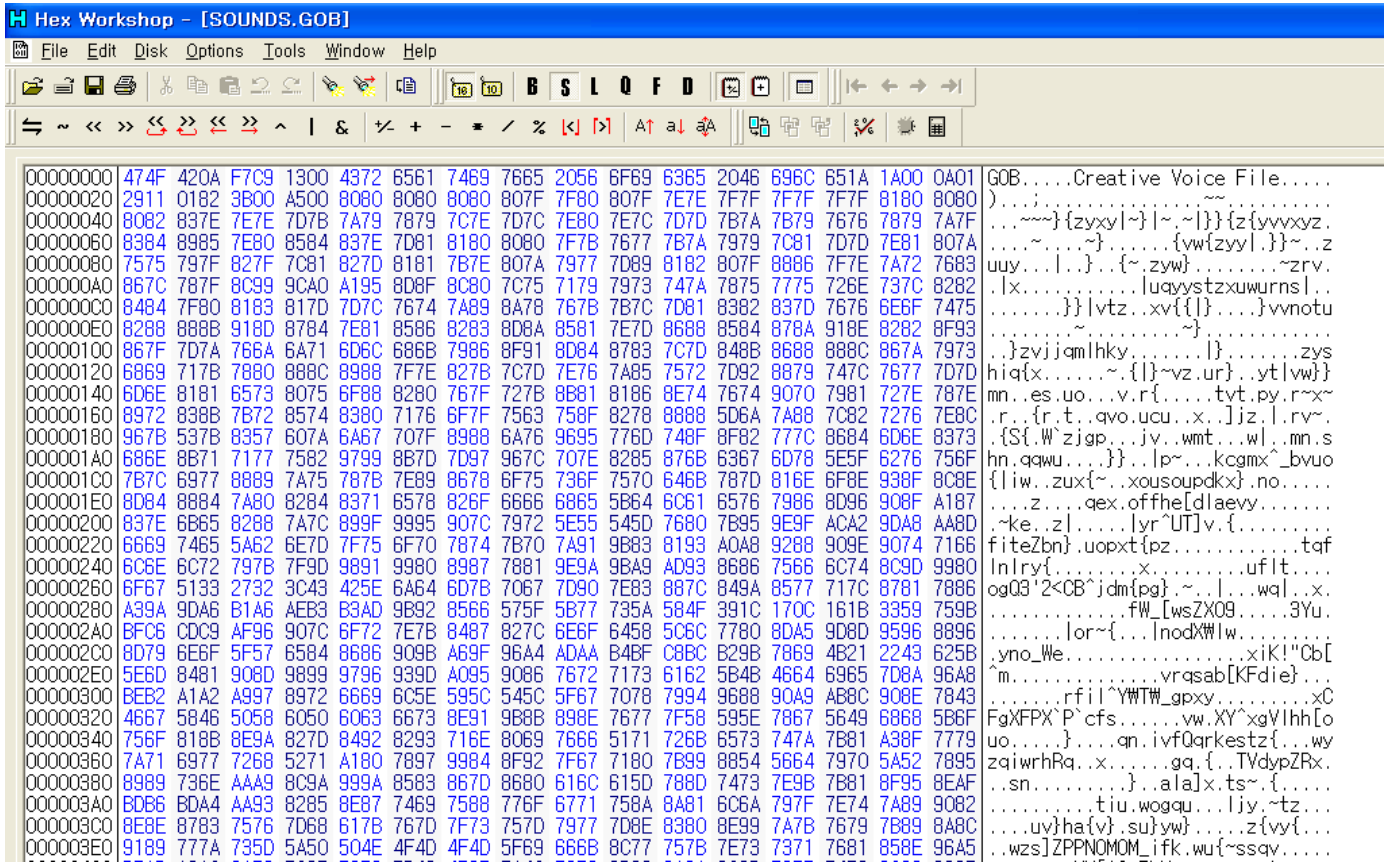
Things to look out for

- GRAIS(Game Resource Archive Identity Strings)
- Offsets
- Size Fields
- Number of Files
- Filenames

Basic Facts

- The file that contains the resources are bigger than the other files
- Most values are stored in a 4 byte field
- There is usually at least 1 meaningful field in the beginning of the file
- Strings can be null-terminated, or fixed-length in which case the string is padded with nulls

*.GOB File



OK... So I'm looking at a bunch of Hex Bytes... Now What???

First Things First

- File Size = 1299033
- File size fits in 3 bytes(0~16777216)
- Most fields are stored as 4 bytes
- All offset and size fields will end with a NULL byte

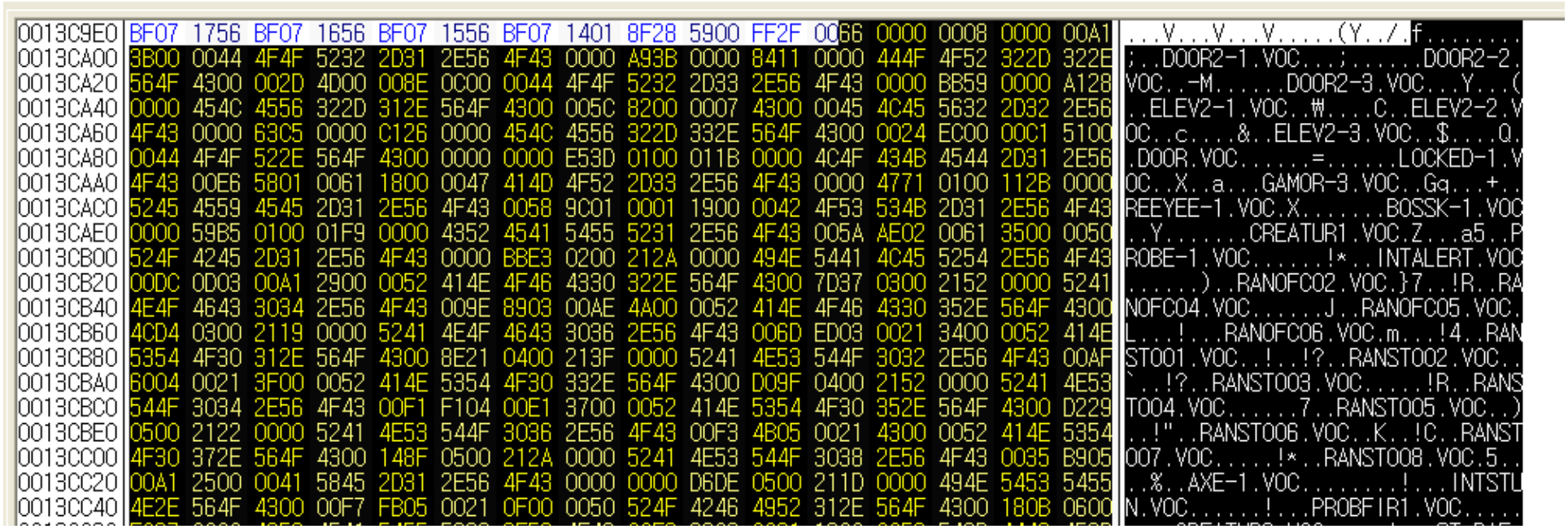
Identifying the Header

GRAIF

Possibly an offset ???

```
00000000 474F 420A F7C9 1300 4372 6561 7469 7665 2056 6F69 6365 2046 696C 651A 1A00 0A01 GOB.....Creative Voice File.....
00000020 2911 0182 3B00 A500 8080 8080 8080 807F 7F80 807F 7E7E 7F7F 7F7F 7F7F 8180 8080 )...;.............~~..........
00000040 8082 837E 7E7E 7D7B 7A79 7879 7C7E 7D7C 7E80 7E7C 7D7D 7B7A 7B79 7676 7879 7A7F ...~~~}{zyxy|~}|~.~|}}{z{yvvxyz.
00000060 8384 8985 7E80 8584 837E 7D81 8180 8080 7F7B 7677 7B7A 7979 7C81 7D7D 7E81 807A ....~....~}.....{vw{zyy|.}}~..z
00000080 7575 797F 827F 7C81 827D 8181 7B7E 807A 7977 7D89 8182 807F 8886 7F7E 7A72 7683 uuy...|..}..{~.zyw}........~zrv.
000000A0 867C 787F 8C99 9CA0 A195 8D8F 8C80 7C75 7179 7973 747A 7875 7775 726E 737C 8282 .|x..........|uqyystzxuwurns|..
000000C0 8484 7F80 8183 817D 7D7C 7674 7A89 8A78 767B 7B7C 7D81 8382 837D 7676 6E6F 7475 .......}}|vtz..xv{{|}....}vvnotu
000000E0 8288 888B 918D 8784 7E81 8586 8283 8D8A 8581 7E7D 8688 8584 878A 918E 8282 8F93 ........~.........~}............
00000100 867F 7D7A 766A 6A71 6D6C 686B 7986 8F91 8D84 8783 7C7D 848B 8688 888C 867A 7973 ..}zvjjqmlhky.......|}.......zys
00000120 6869 717B 7880 888C 8988 7F7E 827B 7C7D 7E76 7A85 7572 7D92 8879 747C 7677 7D7D hiq{x.....~.{|}~vz.ur}..yt|vw}}
00000140 6D6E 8181 6573 8075 6F88 8280 767F 727B 8B81 8186 8E74 7674 9070 7981 727E 787E mn..es.uo...v.r{.....tvt.py.r~x~
```

Verifying

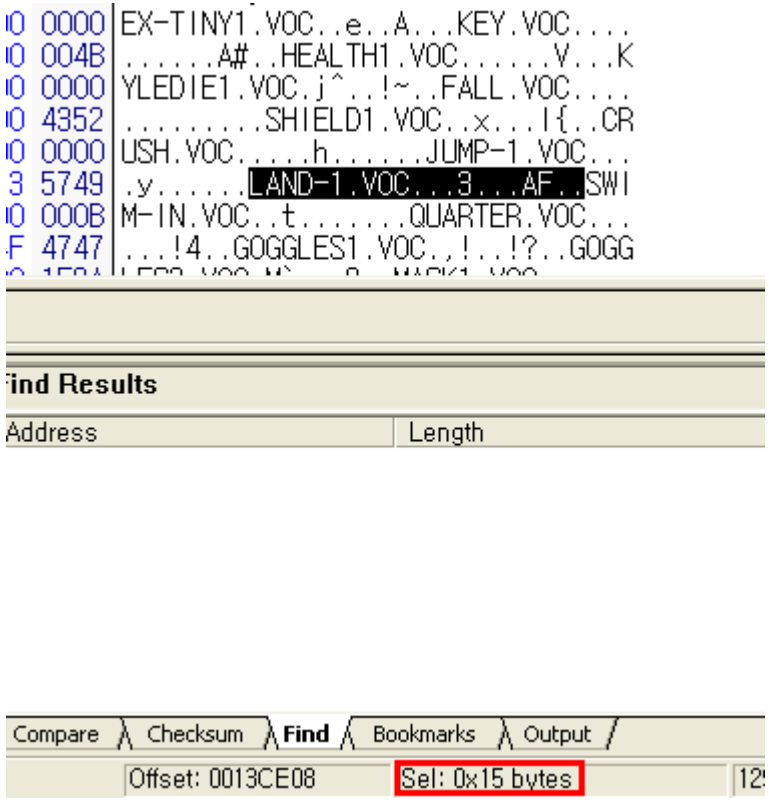


```
0013C9E0 BF07 1756 BF07 1656 BF07 1556 BF07 1401 8F28 5900 FF2F 0066 0000 0008 0000 00A1  ...V...V...V.....(Y../.f........
0013CA00 3B00 0044 4F4F 5232 2D31 2E56 4F43 0000 A93B 0000 8411 0000 444F 4F52 322D 322E  ;..DOOR2-1.VOC...;.....DOOR2-2.
0013CA20 564F 4300 002D 4D00 008E 0C00 0044 4F4F 5232 2D33 2E56 4F43 0000 BB59 0000 A128  VOC..-M.....DOOR2-3.VOC...Y...(
0013CA40 0000 454C 4556 322D 312E 564F 4300 005C 8200 0007 4300 0045 4C45 5632 2D32 2E56  ..ELEV2-1.VOC..₩....C..ELEV2-2.V
0013CA60 4F43 0000 63C5 0000 C126 0000 454C 4556 322D 332E 564F 4300 0024 EC00 00C1 5100  OC..c....&..ELEV2-3.VOC..$....Q.
0013CA80 0044 4F4F 522E 564F 4300 0000 0000 E53D 0100 011B 0000 4C4F 434B 4544 2D31 2E56  .DOOR.VOC.....=.....LOCKED-1.V
0013CAA0 4F43 00E6 5801 0061 1800 0047 414D 4F52 2D33 2E56 4F43 0000 4771 0100 112B 0000  OC..X..a...GAMOR-3.VOC..Gq...+..
0013CAC0 5245 4559 4545 2D31 2E56 4F43 0058 9C01 0001 1900 0042 4F53 534B 2D31 2E56 4F43  REEYEE-1.VOC.X.....BOSSK-1.VOC
0013CAE0 0000 59B5 0100 01F9 0000 4352 4541 5455 5231 2E56 4F43 005A AE02 0061 3500 0050  ..Y.....CREATUR1.VOC.Z...a5..P
0013CB00 524F 4245 2D31 2E56 4F43 0000 BBE3 0200 212A 0000 494E 5441 4C45 5254 2E56 4F43  ROBE-1.VOC.....!*..INTALERT.VOC
0013CB20 00DC 0D03 00A1 2900 0052 414E 4F46 4330 322E 564F 4300 7D37 0300 2152 0000 5241  ......)..RANOFCO2.VOC.}7..!R..RA
0013CB40 4E4F 4643 3034 2E56 4F43 009E 8903 00AE 4A00 0052 414E 4F46 4330 352E 564F 4300  NOFCO4.VOC.....J..RANOFCO5.VOC.
0013CB60 4CD4 0300 2119 0000 5241 4E4F 4643 3036 2E56 4F43 006D ED03 0021 3400 0052 414E  L...!...RANOFCO6.VOC.m...!4..RAN
0013CB80 5354 4F30 312E 564F 4300 8E21 0400 213F 0000 5241 4E53 544F 3032 2E56 4F43 00AF  STO01.VOC..!..!?..RANSTO02.VOC..
0013CBA0 6004 0021 3F00 0052 414E 5354 4F30 332E 564F 4300 D09F 0400 2152 0000 5241 4E53  `..!?..RANSTO03.VOC.....!R..RANS
0013CBC0 544F 3034 2E56 4F43 00F1 F104 00E1 3700 0052 414E 5354 4F30 352E 564F 4300 D229  TO04.VOC.....7..RANSTO05.VOC..)
0013CBE0 0500 2122 0000 5241 4E53 544F 3036 2E56 4F43 00F3 4B05 0021 4300 0052 414E 5354  ..!"..RANSTO06.VOC..K..!C..RANST
0013CC00 4F30 372E 564F 4300 148F 0500 212A 0000 5241 4E53 544F 3038 2E56 4F43 0035 B905  O07.VOC.....!*..RANSTO08.VOC.5..
0013CC20 00A1 2500 0041 5845 2D31 2E56 4F43 0000 0000 D6DE 0500 211D 0000 494E 5453 5455  ..%..AXE-1.VOC.....!...INTSTU
0013CC40 4E2E 564F 4300 00F7 FB05 0021 0F00 0050 524F 4246 4952 312E 564F 4300 180B 0600  N.VOC.....!...PROBFIR1.VOC.....
```

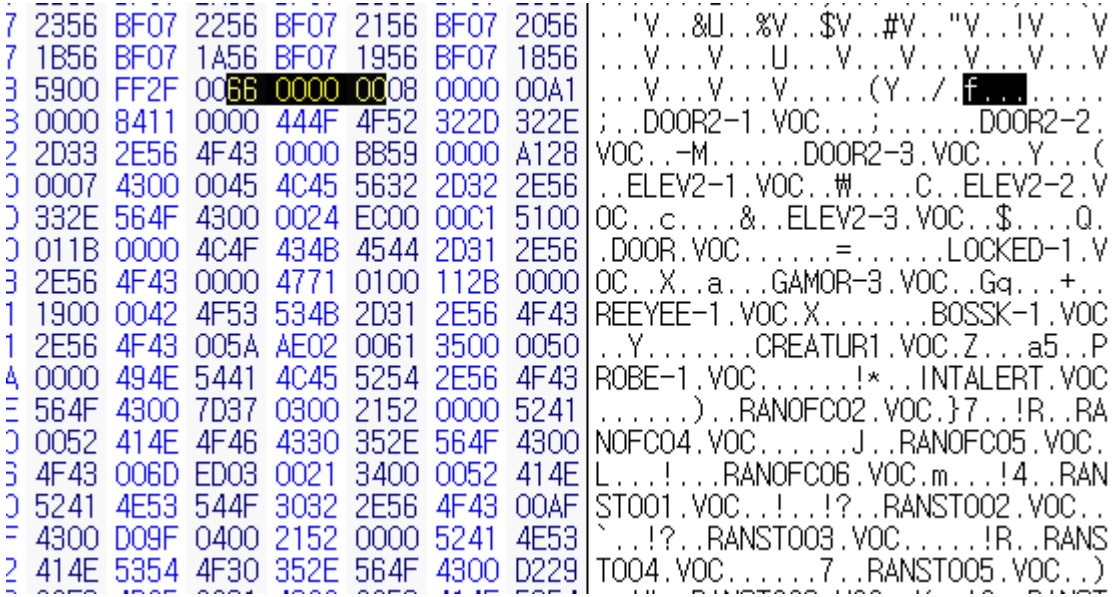
Is indeed an offset **J**

Looks like a directory…

Guessing the Directory entry length

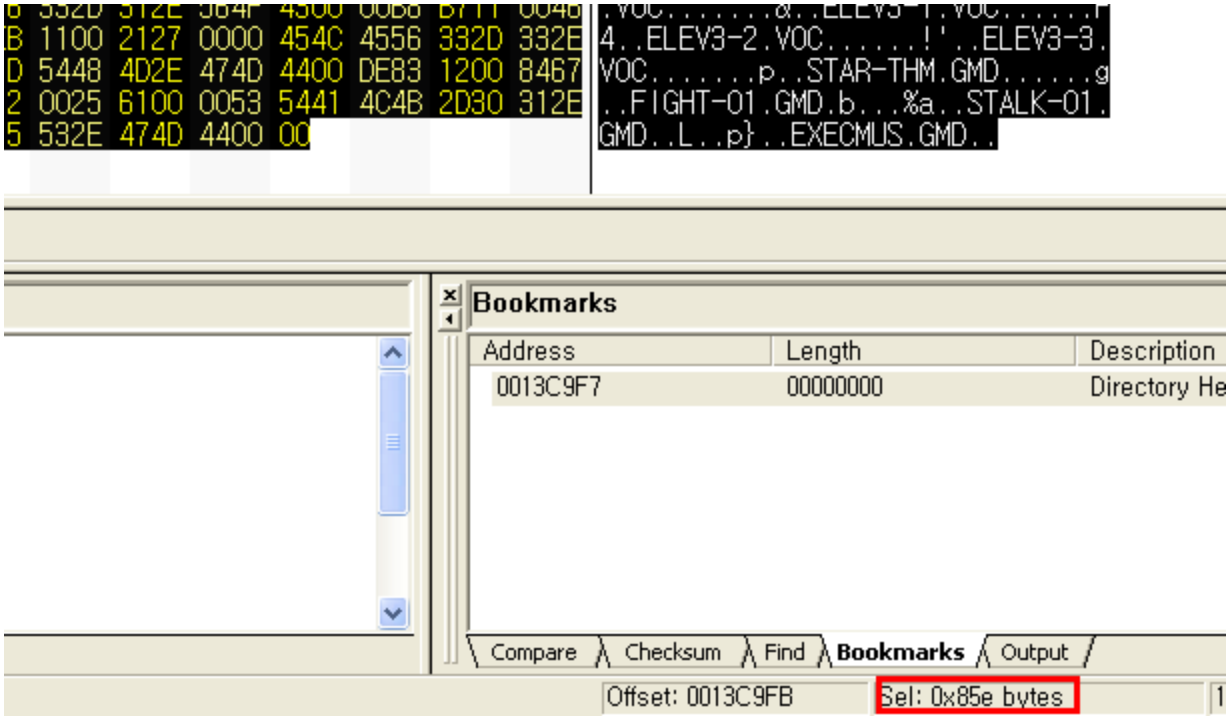


The excess data



Possibly the number of Directory entries in the Directory?

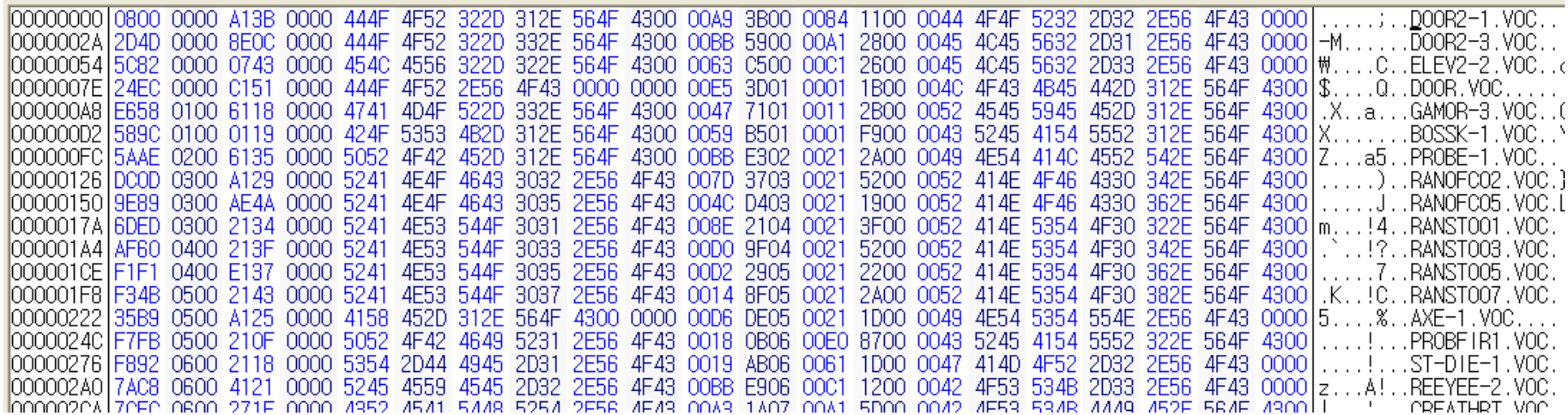
Verifying



0x85E / 0x15 = 0x66

Verified! **J**

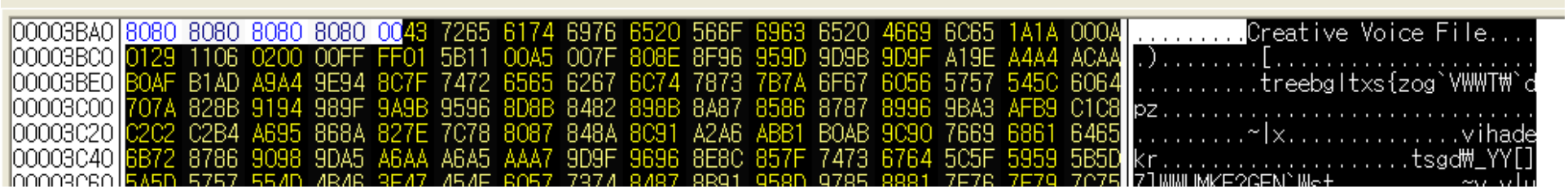
Guessing the fields in the Directory Entry



```
00000000 0800 0000 A13B 0000 444F 4F52 322D 312E 564F 4300 00A9 3B00 0084 1100 0044 4F4F 5232 2D32 2E56 4F43 0000 .....;..DOOR2-1.VOC..
0000002A 2D4D 0000 8E0C 0000 444F 4F52 322D 332E 564F 4300 00BB 5900 00A1 2800 0045 4C45 5632 2D31 2E56 4F43 0000 -M......DOOR2-3.VOC..
00000054 5C82 0000 0743 0000 454C 4556 322D 322E 564F 4300 0063 C500 00C1 2600 0045 4C45 5632 2D33 2E56 4F43 0000 ₩....C..ELEV2-2.VOC..
0000007E 24EC 0000 C151 0000 444F 4F52 2E56 4F43 0000 0000 00E5 3D01 0001 1B00 004C 4F43 4B45 442D 312E 564F 4300 $....Q..DOOR.VOC.....
000000A8 E658 0100 6118 0000 4741 4D4F 522D 332E 564F 4300 0047 7101 0011 2B00 0052 4545 5945 452D 312E 564F 4300 .X..a...GAMOR-3.VOC..
000000D2 589C 0100 0119 0000 424F 5353 4B2D 312E 564F 4300 0059 B501 0001 F900 0043 5245 4154 5552 312E 564F 4300 X.......BOSSK-1.VOC..
000000FC 5AAE 0200 6135 0000 5052 4F42 452D 312E 564F 4300 00BB E302 0021 2A00 0049 4E54 414C 4552 542E 564F 4300 Z...a5..PROBE-1.VOC..
00000126 DC0D 0300 A129 0000 5241 4E4F 4643 3032 2E56 4F43 007D 3703 0021 5200 0052 414E 4F46 4330 342E 564F 4300 .....)..RANOFCO2.VOC.}
00000150 9E89 0300 AE4A 0000 5241 4E4F 4643 3035 2E56 4F43 004C D403 0021 1900 0052 414E 4F46 4330 362E 564F 4300 .....J..RANOFCO5.VOC.L
0000017A 6DED 0300 2134 0000 5241 4E53 544F 3031 2E56 4F43 008E 2104 0021 3F00 0052 414E 5354 4F30 322E 564F 4300 m...!4..RANSTO01.VOC.
000001A4 AF60 0400 213F 0000 5241 4E53 544F 3033 2E56 4F43 00D0 9F04 0021 5200 0052 414E 5354 4F30 342E 564F 4300 .`..!?..RANSTO03.VOC.
000001CE F1F1 0400 E137 0000 5241 4E53 544F 3035 2E56 4F43 00D2 2905 0021 2200 0052 414E 5354 4F30 362E 564F 4300 .....7..RANSTO05.VOC.
000001F8 F34B 0500 2143 0000 5241 4E53 544F 3037 2E56 4F43 0014 8F05 0021 2A00 0052 414E 5354 4F30 382E 564F 4300 .K..!C..RANSTO07.VOC.
00000222 35B9 0500 A125 0000 4158 452D 312E 564F 4300 0000 00D6 DE05 0021 1D00 0049 4E54 5354 554E 2E56 4F43 0000 5....%..AXE-1.VOC....
0000024C F7FB 0500 210F 0000 5052 4F42 4649 5231 2E56 4F43 0018 0B06 00E0 8700 0043 5245 4154 5552 322E 564F 4300 ....!...PROBFIR1.VOC.
00000276 F892 0600 2118 0000 5354 2D44 4945 2D31 2E56 4F43 0019 AB06 0061 1D00 0047 414D 4F52 2D32 2E56 4F43 0000 ....!...ST-DIE-1.VOC.
000002A0 7AC8 0600 4121 0000 5245 4559 4545 2D32 2E56 4F43 00BB E906 00C1 1200 0042 4F53 534B 2D33 2E56 4F43 0000 z...A!..REEYEE-2.VOC.
000002CA 7CEC 0600 271F 0000 4352 4541 5448 5254 2E56 4F43 00A3 1A07 00A1 5D00 0042 4F53 534B 4449 452E 564F 4300 |...'...CREATHRT.VOC.
```

- First field is increasing in each successive entries → Possibly a File Offset?
- If first theory is correct, second field has a high chance to be the File Size

Verifying



Confirmed! **J**

Extracting the resources using a custom script

```
import struct
import os
import sys
import string

if len(sys.argv) < 2:
    print "usage : swdf_fileextract filename"
    exit(1)

data = open(sys.argv[1],"rb").read()
gaif = struct.unpack("4s",data[0:4])[0]
directory_offset = struct.unpack("L",data[4:8])[0]
filenum = struct.unpack("L",data[directory_offset:directory_offset+4])[0]
newdirname = "./"+sys.argv[1][:sys.argv[1].index('.')]
try:
    os.makedirs(newdirname)
except OSError:
    pass
print "Game Archive Identification String :",gaif
print "Directory Offset :",hex(directory_offset)
print "Number of Files :",filenum

for i in range(filenum):
    directory_index = directory_offset+4+(i*0x15)
    file_offset = struct.unpack("L",data[directory_index:directory_index+4])[0]
    file_size = struct.unpack("L",data[directory_index+4:directory_index+8])[0]
    file_name = string.join(struct.unpack("13s",data[directory_index+8:directory_index+8+13]))
    file_data = struct.unpack("%ds"%file_size,data[file_offset:file_offset+file_size])[0]
    file_name = file_name.strip('\0')
    outfile = open("./%s/%s"%(newdirname,file_name),"wb")
```

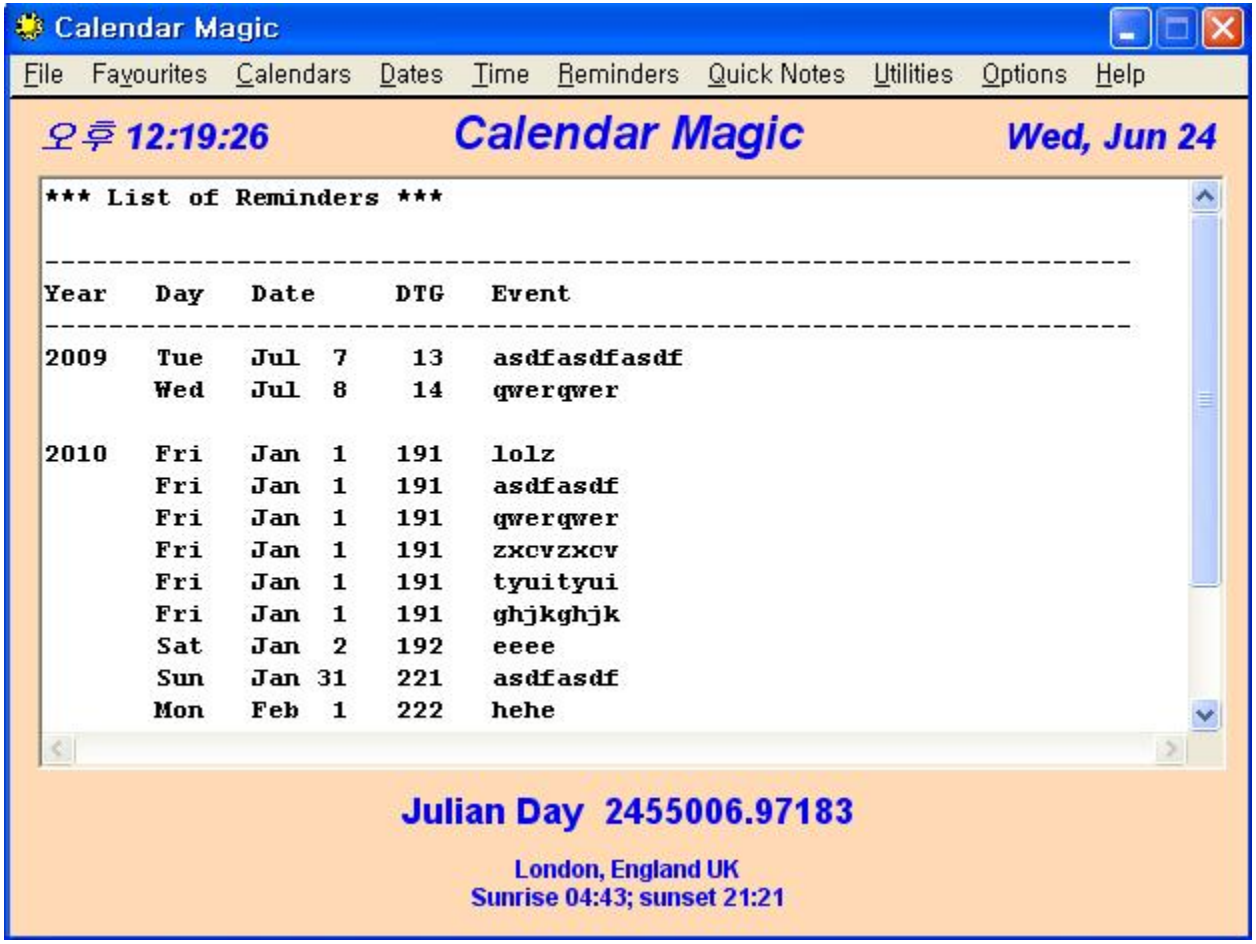
The extracted Resources

Files extracted! **J** Moving on…

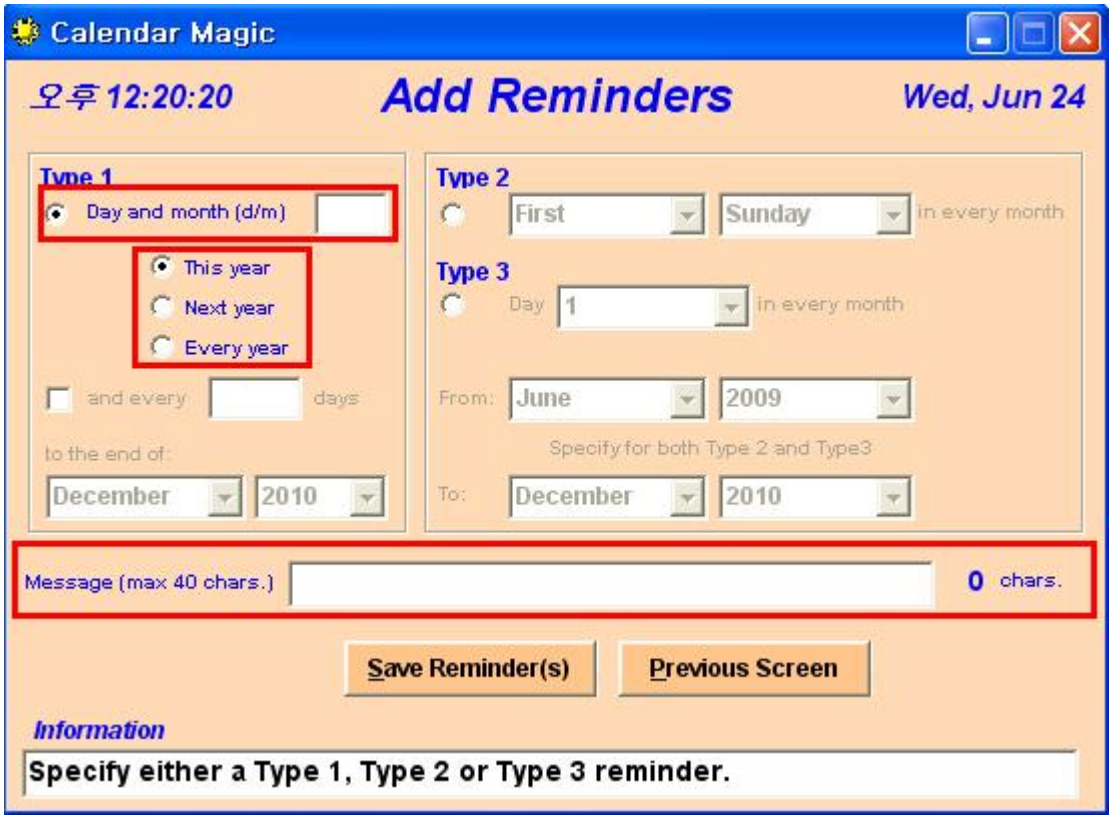
Other things to look out for when reversing Game Archive Files

- First File Offset
- Archive Name
- Filename Offset
- Filename Directory Offset
- Total File Data Size
- Total Directory Size
- Archive Size
- Number Of Directories
- Directory Offset
- File Extension / Type
- File ID
- Archive Version
- Filename Length
- Decompressed File Size
- Checksum
- Timestamp

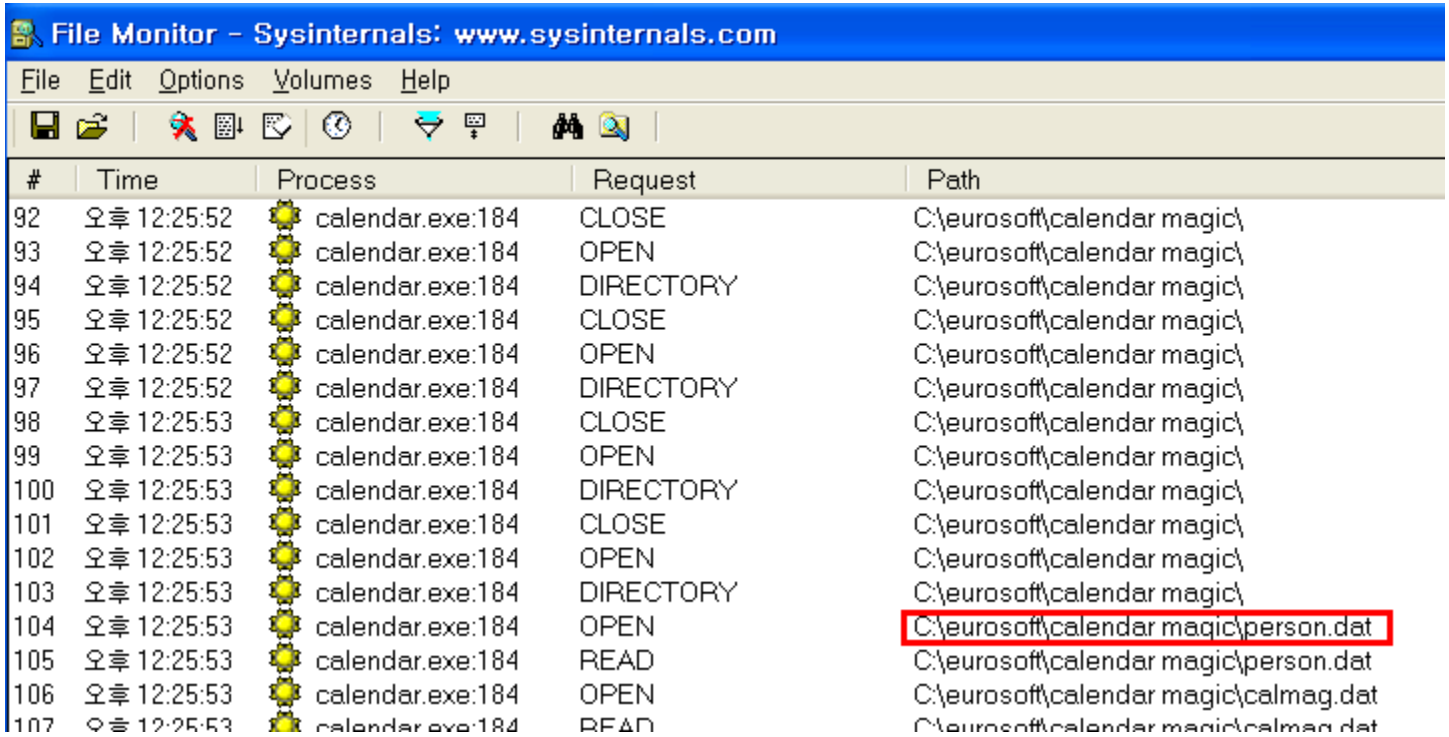
Case II : Reversing a Calendar Data File



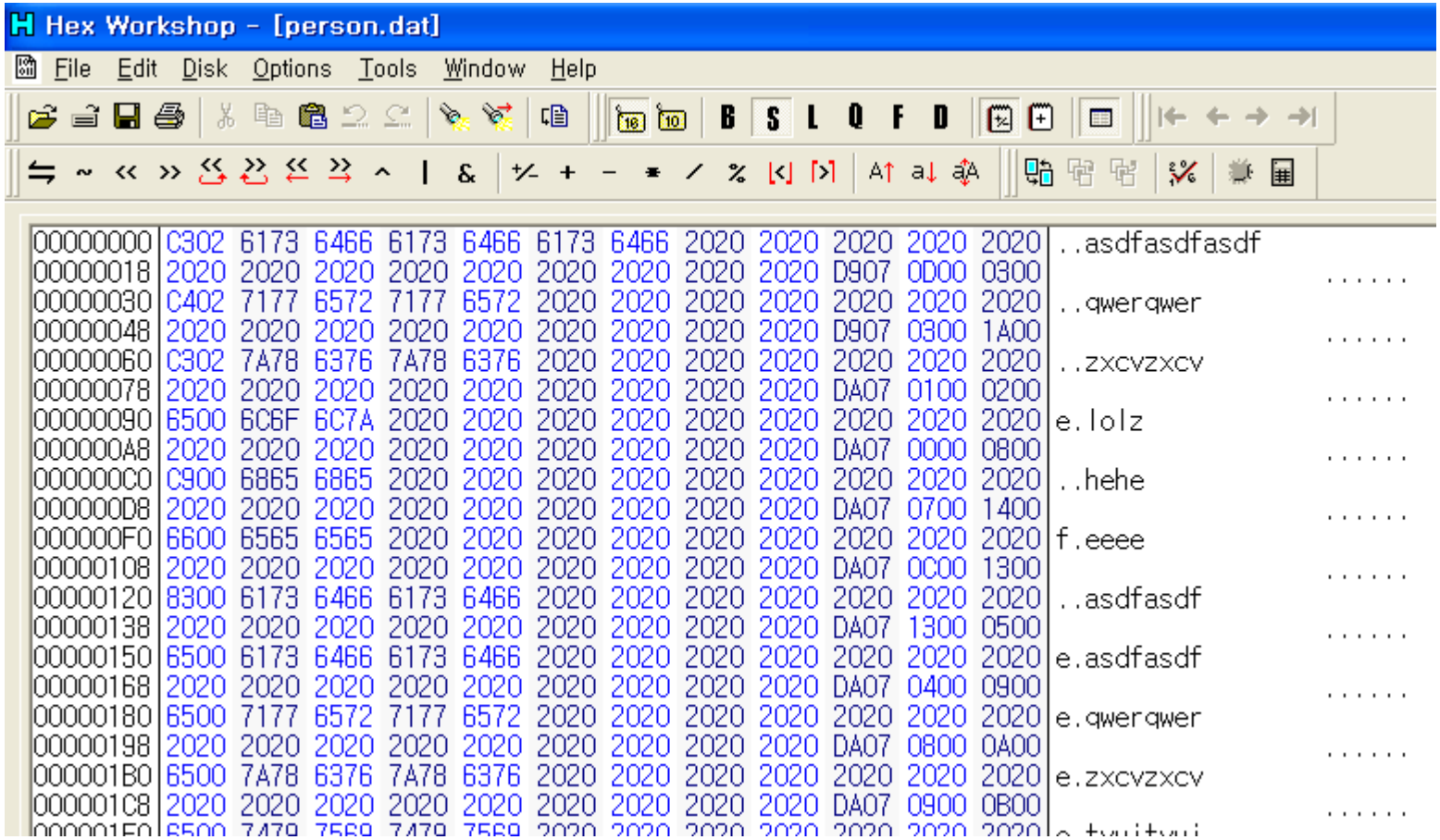
Guess what's going to be stored in the Data File



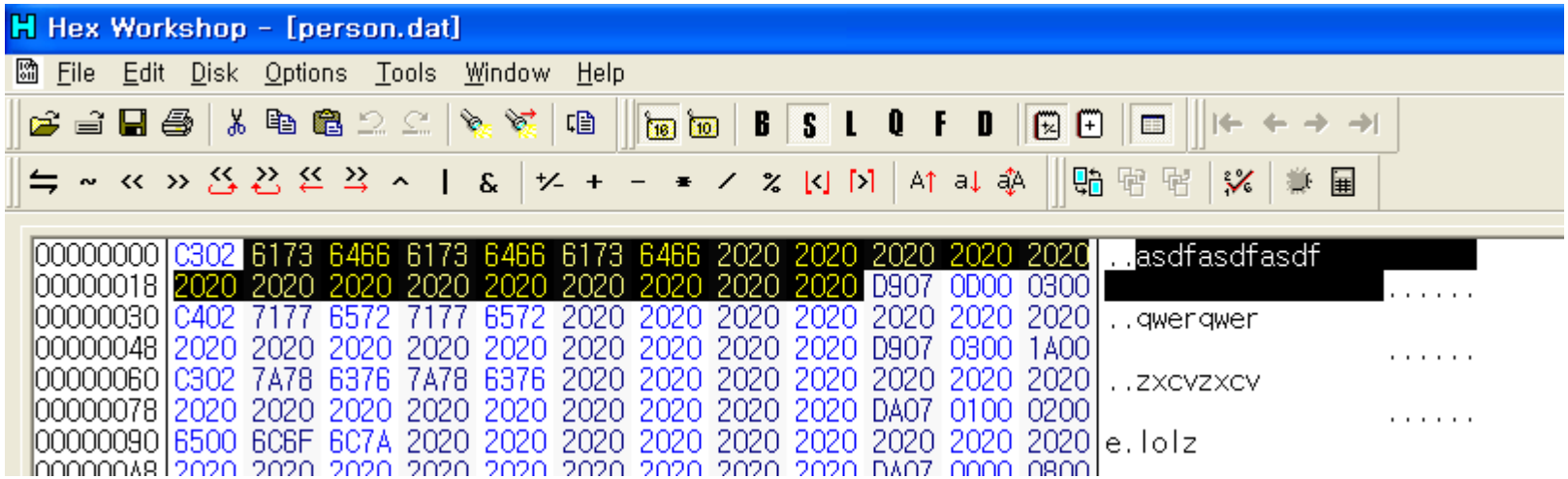
Find the Data File



Start Digging! ☺



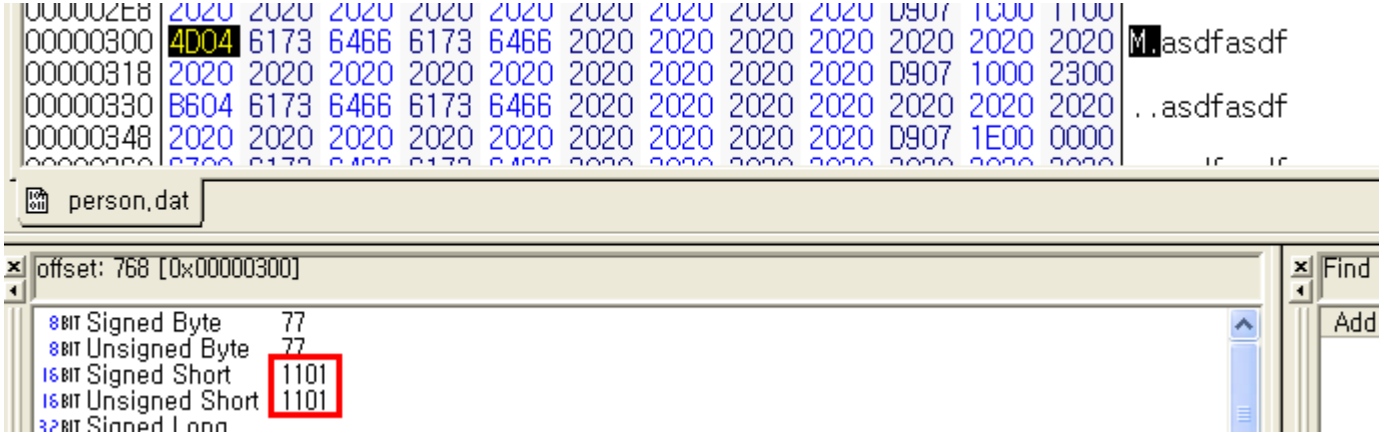
Message Part



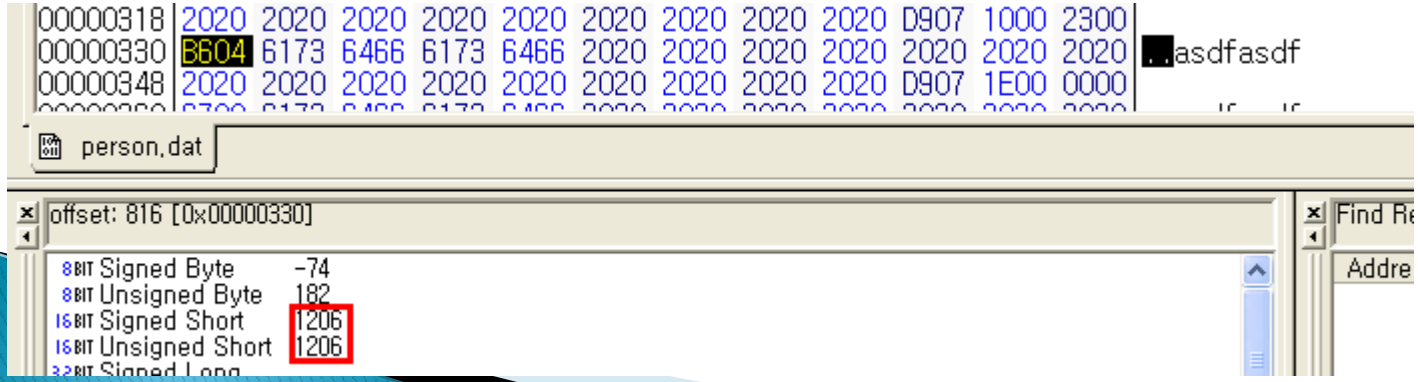
- Max 40 Chars

Date Field

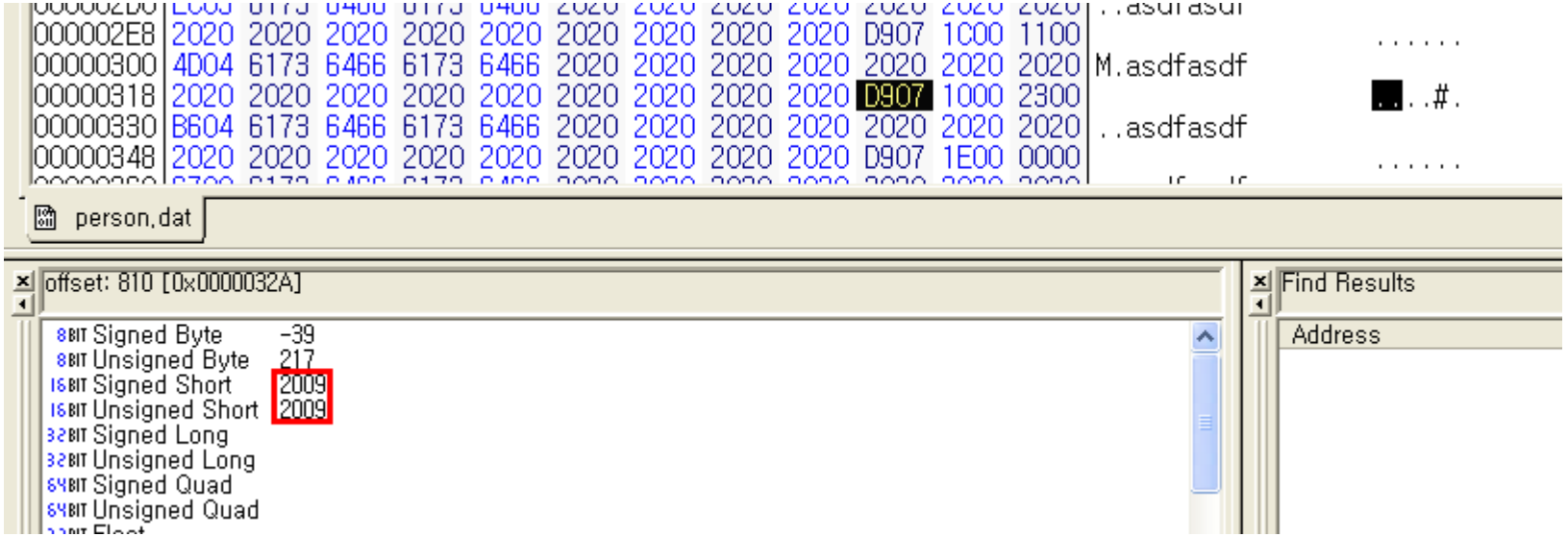
- 11/1



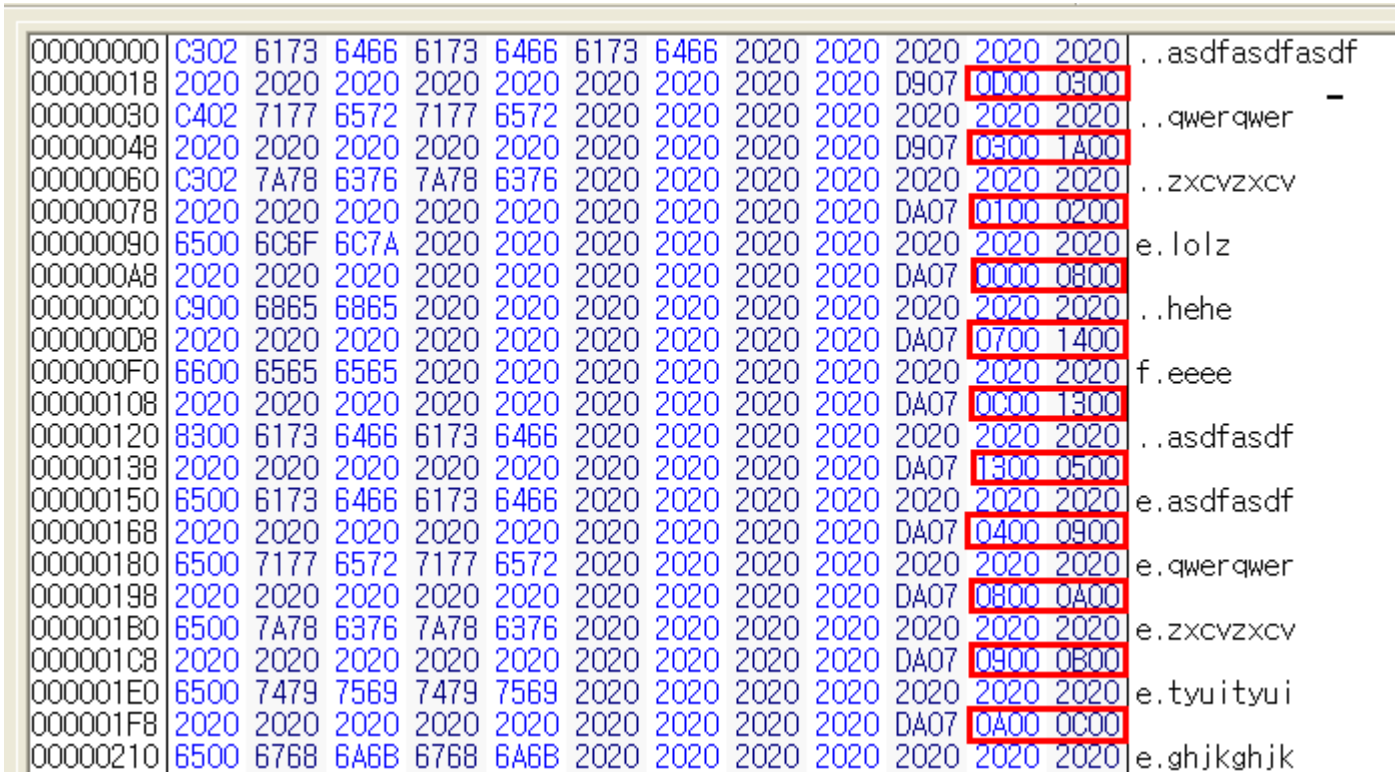
- 12/6



Year Field


000002E8 2020 2020 2020 2020 2020 2020 2020 2020 2020 D907 1C00 1100
00000300 4D04 6173 6466 6173 6466 2020 2020 2020 2020 2020 2020 2020
00000318 2020 2020 2020 2020 2020 2020 2020 2020 2020 D907 1000 2300
00000330 B604 6173 6466 6173 6466 2020 2020 2020 2020 2020 2020 2020
00000348 2020 2020 2020 2020 2020 2020 2020 2020 2020 D907 1E00 0000
M.asdfasdf
..asdfasdf
person.dat
offset: 810 [0x0000032A]
8BIT Signed Byte -39
8BIT Unsigned Byte 217
16BIT Signed Short 2009
16BIT Unsigned Short 2009
32BIT Signed Long
32BIT Unsigned Long
64BIT Signed Quad
64BIT Unsigned Quad
Find Results
Address

The Excess Data



```
00000000 C302 6173 6466 6173 6466 6173 6466 2020 2020 2020 2020 2020 ..asdfasdfasdf
00000018 2020 2020 2020 2020 2020 2020 2020 2020 2020 D907 0D00 0300
00000030 C402 7177 6572 7177 6572 2020 2020 2020 2020 2020 2020 2020 ..qwerqwer
00000048 2020 2020 2020 2020 2020 2020 2020 2020 2020 D907 0300 1A00
00000060 C302 7A78 6376 7A78 6376 2020 2020 2020 2020 2020 2020 2020 ..zxcvzxcv
00000078 2020 2020 2020 2020 2020 2020 2020 2020 2020 DA07 0100 0200
00000090 6500 6C6F 6C7A 2020 2020 2020 2020 2020 2020 2020 2020 2020 e.lolz
000000A8 2020 2020 2020 2020 2020 2020 2020 2020 2020 DA07 0000 0800
000000C0 C900 6865 6865 2020 2020 2020 2020 2020 2020 2020 2020 2020 ..hehe
000000D8 2020 2020 2020 2020 2020 2020 2020 2020 2020 DA07 0700 1400
000000F0 6600 6565 6565 2020 2020 2020 2020 2020 2020 2020 2020 2020 f.eeee
00000108 2020 2020 2020 2020 2020 2020 2020 2020 2020 DA07 0C00 1300
00000120 8300 6173 6466 6173 6466 2020 2020 2020 2020 2020 2020 2020 ..asdfasdf
00000138 2020 2020 2020 2020 2020 2020 2020 2020 2020 DA07 1300 0500
00000150 6500 6173 6466 6173 6466 2020 2020 2020 2020 2020 2020 2020 e.asdfasdf
00000168 2020 2020 2020 2020 2020 2020 2020 2020 2020 DA07 0400 0900
00000180 6500 7177 6572 7177 6572 2020 2020 2020 2020 2020 2020 2020 e.qwerqwer
00000198 2020 2020 2020 2020 2020 2020 2020 2020 2020 DA07 0800 0A00
000001B0 6500 7A78 6376 7A78 6376 2020 2020 2020 2020 2020 2020 2020 e.zxcvzxcv
000001C8 2020 2020 2020 2020 2020 2020 2020 2020 2020 DA07 0900 0B00
000001E0 6500 7479 7569 7479 7569 2020 2020 2020 2020 2020 2020 2020 e.tyuityui
000001F8 2020 2020 2020 2020 2020 2020 2020 2020 2020 DA07 0A00 0C00
00000210 6500 6768 6A6B 6768 6A6B 2020 2020 2020 2020 2020 2020 2020 e.ghjkghjk
```

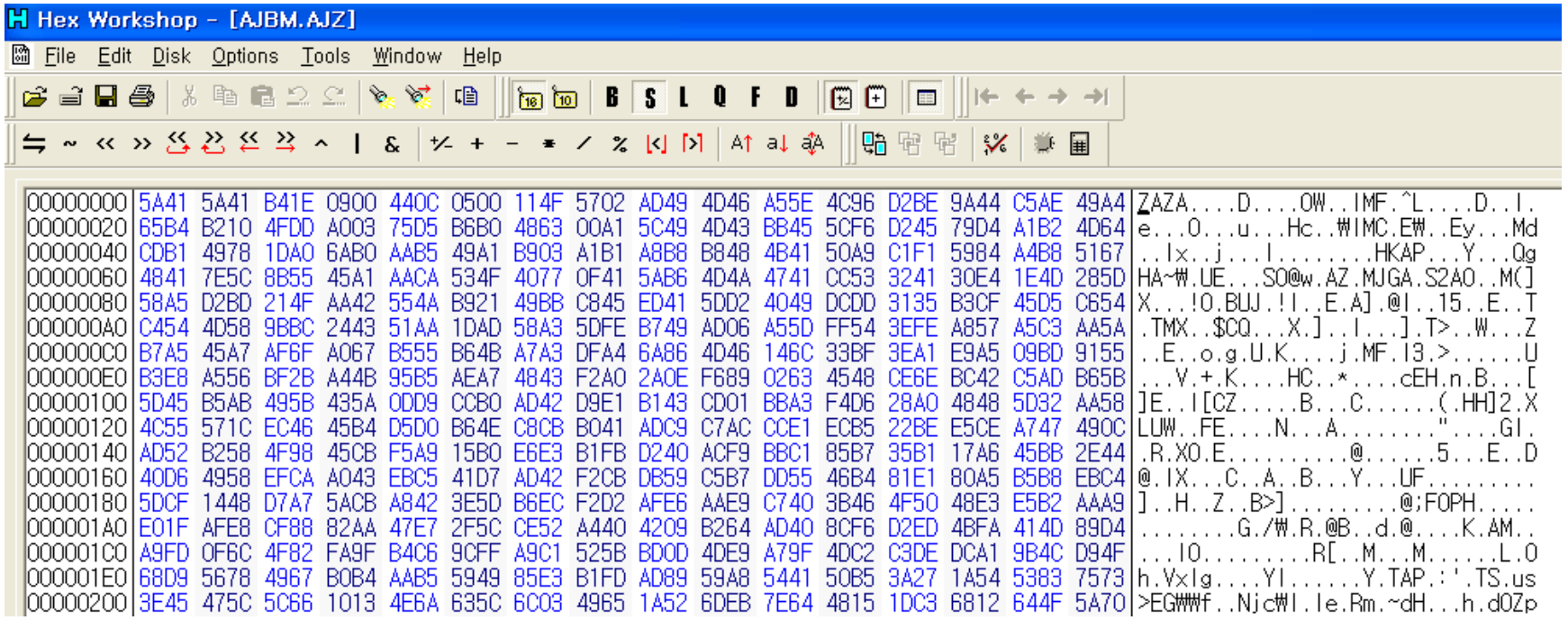
- Possibly some kind of unique identifier

Person.dat File Format

Chunk Structure

Size	Meaning
2	Date
40	Message
2	Year
4	Identifier

Case III : Reversing Firmware



You know what to do next! **J**

Try to identify the Header

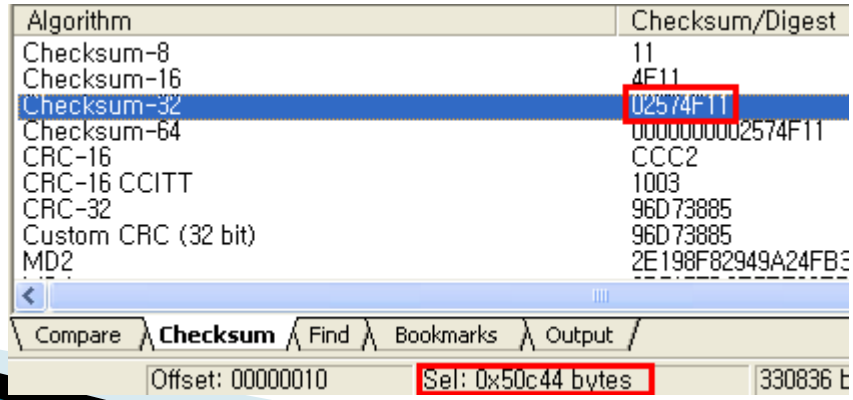
FileType

Unknown

Data Size

Checksum

00000000	5A41	5A41	B41E	0900	440C	0500	114F	5702	AD49	4D46	A55E	4C96	D2BE	9A44	C5AE	49A4	ZAZA
00000020	65B4	B210	4FDD	A003	75D5	B6B0	4863	00A1	5C49	4D43	BB45	5CF6	D245	79D4	A1B2	4D64	e...O.
00000040	CDB1	4978	1DA0	6AB0	AAB5	49A1	B903	A1B1	A8B8	B848	4B41	50A9	C1F1	5984	A4B8	5167	..Ix..
00000060	4841	7E5C	8B55	45A1	AACA	534F	4077	0F41	5AB6	4D4A	4741	CC53	3241	30E4	1E4D	285D	HA~₩.UE
00000080	58A5	D2BD	214F	AA42	554A	B921	49BB	C845	ED41	5DD2	4049	DCDD	3135	B3CF	45D5	C654	X...!O.
000000A0	C454	4D58	9BBC	2443	51AA	1DAD	58A3	5DFE	B749	ADO6	A55D	FF54	3EFE	A857	A5C3	AA5A	.TMX..S
000000C0	B7A5	45A7	AF6F	A067	B555	B64B	A7A3	DFA4	6A86	4D46	146C	33BF	3EA1	E9A5	09BD	9155	..E..o.
000000E0	B3E8	A556	BF2B	A44B	95B5	AEA7	4843	F2A0	2A0E	F689	0263	4548	CE6E	BC42	C5AD	B65B	...V.+.
00000100	5D45	B5AB	495B	435A	ODD9	CCB0	AD42	D9E1	B143	CDO1	BBA3	F4D6	28A0	4848	5D32	AA58	]E..I[C
00000120	4C55	571C	EC46	45B4	D5D0	B64E	C8CB	BO41	ADC9	C7AC	CCE1	ECB5	22BE	E5CE	A747	490C	LUW..FE
00000140	AD52	B258	4F98	45CB	F5A9	15B0	E6E3	B1FB	D240	ACF9	BBC1	85B7	35B1	17A6	45BB	2E44	.R.XO.E
00000160	40D6	4958	EFCA	A043	EBC5	41D7	AD42	F2CB	DB59	C5B7	DD55	46B4	81E1	80A5	B5B8	EBC4	@.IX...
00000180	5DCF	1448	D7A7	5ACB	A842	3E5D	B6EC	F2D2	AFE6	AAE9	C740	3B46	4F50	48E3	E5B2	AAA9	]..H..Z



Algorithm	Checksum/Digest
Checksum-8	11
Checksum-16	4F11
Checksum-32	02574F11
Checksum-64	0000000002574F11
CRC-16	CCC2
CRC-16 CCITT	1003
CRC-32	96D73885
Custom CRC (32 bit)	96D73885
MD2	2E198F82949A24FB3

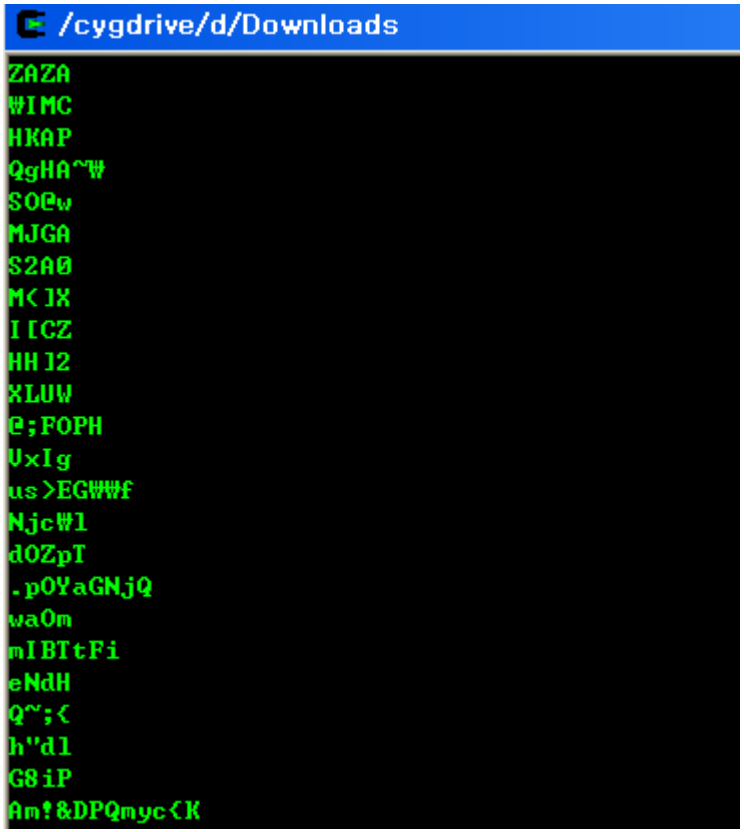
Compare | Checksum | Find | Bookmarks | Output

Offset: 00000010 | Sel: 0x50c44 bytes | 330836 b

Copying the data excluding the Header to a separate File

00000000	AD49	4D46	A55E	4C96	D2BE	9A44	C5AE	49A4	65B4	B210	4FDD	A003	75D5	B6B0	4863	00A1
00000020	5C49	4D43	BB45	5CF6	D245	79D4	A1B2	4D64	CDB1	4978	1DA0	6AB0	AAB5	49A1	B903	A1B1
00000040	A8B8	B848	4B41	50A9	C1F1	5984	A4B8	5167	4841	7E5C	8B55	45A1	AACA	534F	4077	0F41
00000060	5AB6	4D4A	4741	CC53	3241	30E4	1E4D	285D	58A5	D2BD	214F	AA42	554A	B921	49BB	C845
00000080	ED41	5DD2	4049	DCDD	3135	B3CF	45D5	C654	C454	4D58	9BBC	2443	51AA	1DAD	58A3	5DFE
000000A0	B749	AD06	A55D	FF54	3EFE	A857	A5C3	AA5A	B7A5	45A7	AF6F	A067	B555	B64B	A7A3	DFA4
000000C0	6A86	4D46	146C	33BF	3EA1	E9A5	09BD	9155	B3E8	A556	BF2B	A44B	95B5	AEA7	4843	F2A0
000000E0	2A0E	F689	0263	4548	CE6E	BC42	C5AD	B65B	5D45	B5AB	495B	435A	0DD9	CCB0	AD42	D9E1
00000100	B143	CD01	BBA3	F4D6	28A0	4848	5D32	AA58	4C55	571C	EC46	45B4	D5D0	B64E	C8CB	B041
00000120	ADC9	C7AC	CCE1	ECB5	22BE	E5CE	A747	490C	AD52	B258	4F98	45CB	F5A9	15B0	E6E3	B1FB
00000140	D240	ACF9	BBC1	85B7	35B1	17A6	45BB	2E44	40D6	4958	EFCA	A043	EBC5	41D7	AD42	F2CB
00000160	DB59	C5B7	DD55	46B4	81E1	80A5	B5B8	EBC4	5DCF	1448	D7A7	5ACB	A842	3E5D	B6EC	F2D2
00000180	AFE6	AAE9	C740	3B46	4F50	48E3	E5B2	AAA9	E01F	AFE8	CF88	82AA	47E7	2F5C	CE52	A440
000001A0	4209	B264	AD40	8CF6	D2ED	4BFA	414D	89D4	A9FD	0F6C	4F82	FA9F	B4C6	9CFF	A9C1	525B
000001C0	BD0D	4DE9	A79F	4DC2	C3DE	DCA1	9B4C	D94F	68D9	5678	4967	B0B4	AAB5	5949	85E3	B1FD
000001E0	AD89	59A8	5441	50B5	3A27	1A54	5383	7573	3E45	475C	5C66	1013	4E6A	635C	6C03	4965
00000200	1A52	6DEB	7E64	4815	1DC3	6812	644F	5A70	54BF	2E70	4F59	6147	4E6A	5183	E85C	B1C3
00000220	9205	AF77	614F	6DEC	1094	EBFB	6D49	4254	7446	699C	BE50	5968	81A1	73F5	6163	956E
00000240	537B	9360	61C9	A772	ADB2	654E	6448	8278	4674	F386	4F53	31BA	CA5F	E966	4A13	517E
00000260	3B7B	9368	2264	6CBD	157D	BA68	D5CF	689E	61D7	6CDA	76A6	2A4B	8B4B	57C9	BB6F	FE86
00000280	5E6A	8360	A670	9277	088B	5A60	ED86	7D98	2355	1362	455E	151A	4738	6950	0815	7300
000002A0	416D	2126	4450	516D	7963	7B4B	09CD	FF6B	2917	9379	9A55	566F	6581	671F	2743	4201

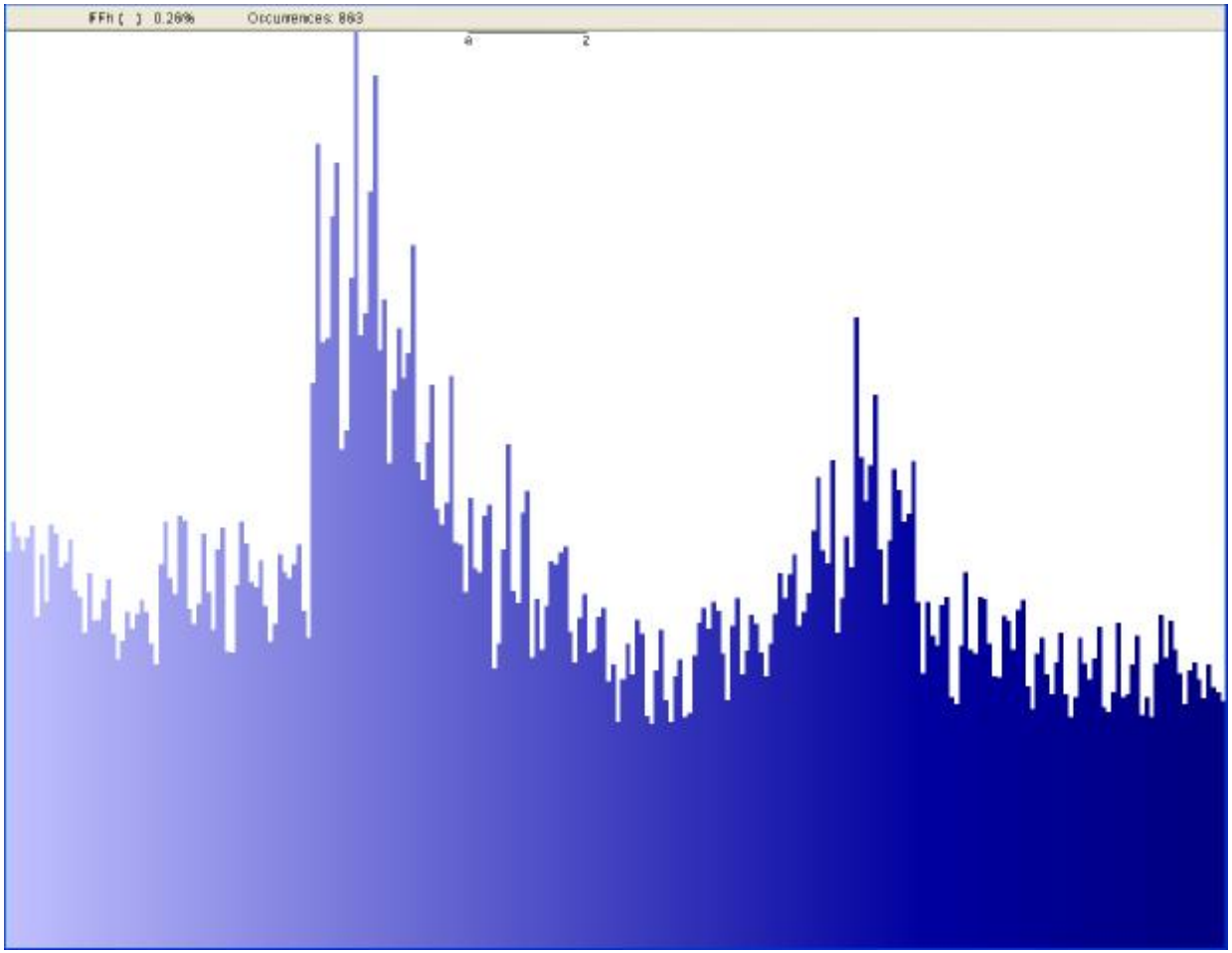
Search for readible strings



- All strings are absolutely Incomprehensible
- → Possibly Compressed, Encrypted
- What to do next?

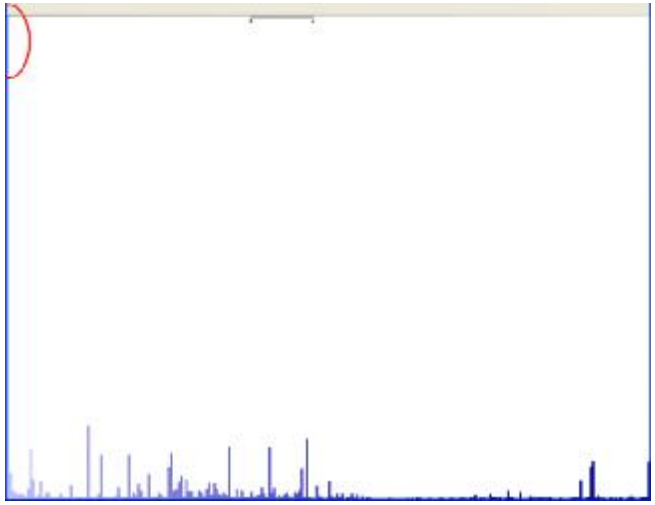
→ Frequency Analysis! **J**

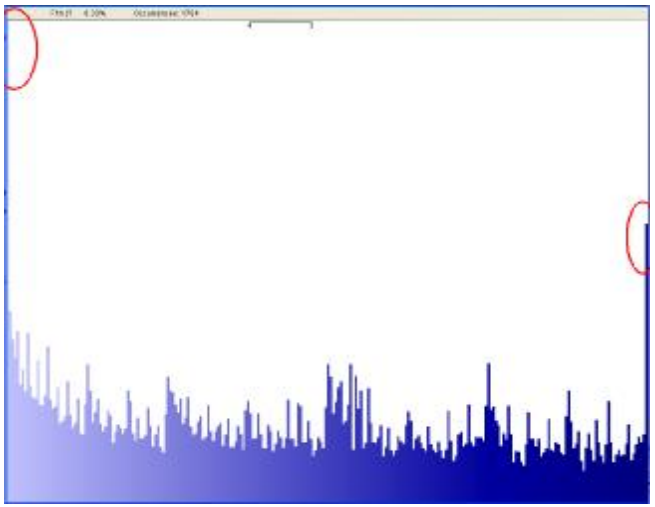
Frequency Analysis of Individual Bytes

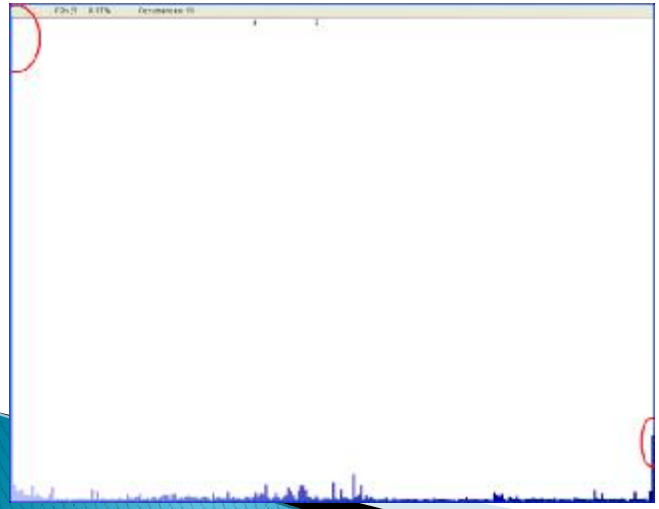


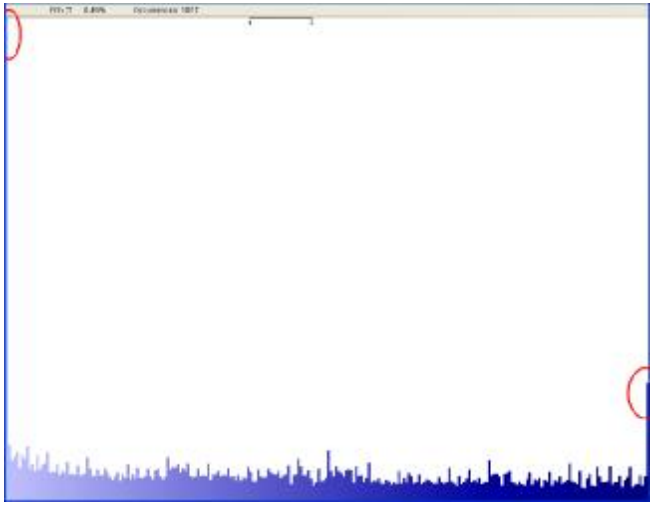
What can we understand from this?

Frequency Analysis of Normal Executable Files

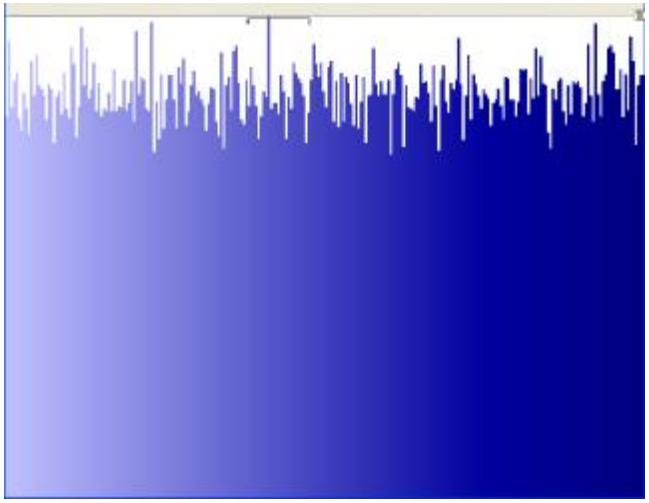


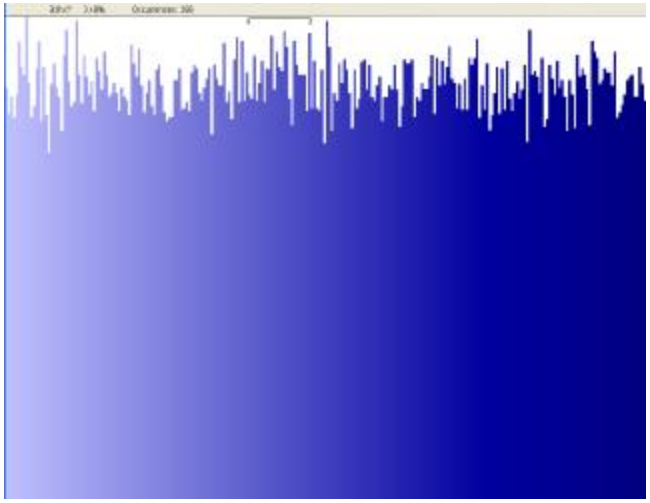


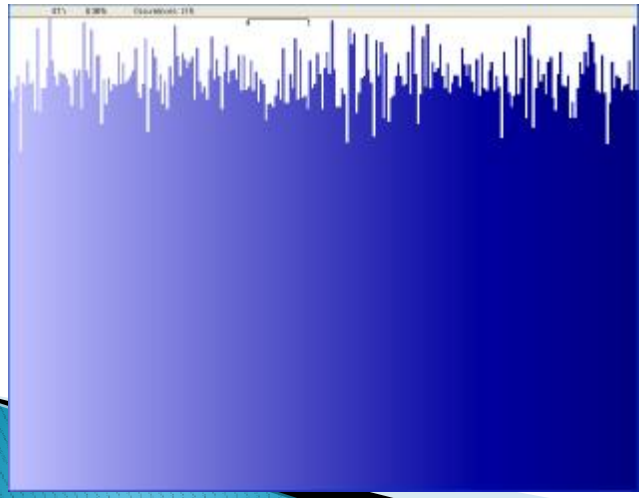


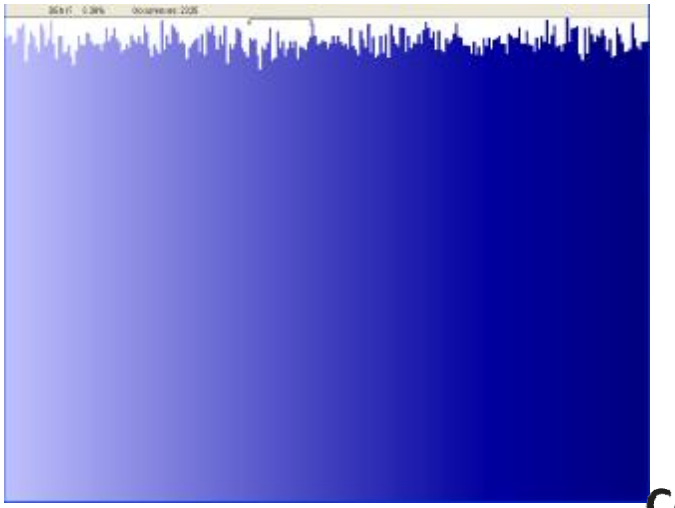


Frequency Analysis of Encrypted Files(Modern Crypto)

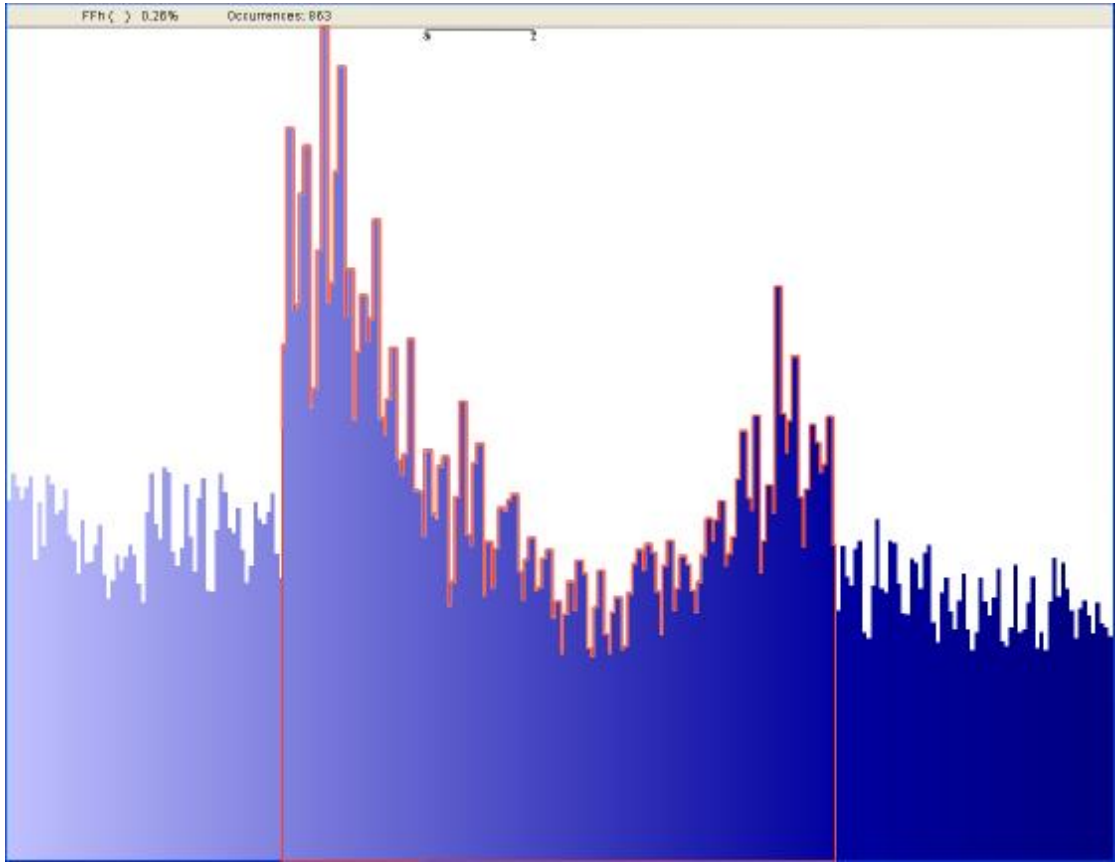








The Symmetry of the Frequency Graph



Highly Symmetric → Pattern

→ Sign of a Weak Cipher !!!

Guessing the Encryption Scheme

- Monoalphabetic
- Polyalphabetic
- Transposition

- ~~Monoalphabetic~~
(Frequency Doesn't Change)
- Polyalphabetic
- ~~Transposition~~
(Frequency Doesn't Change)

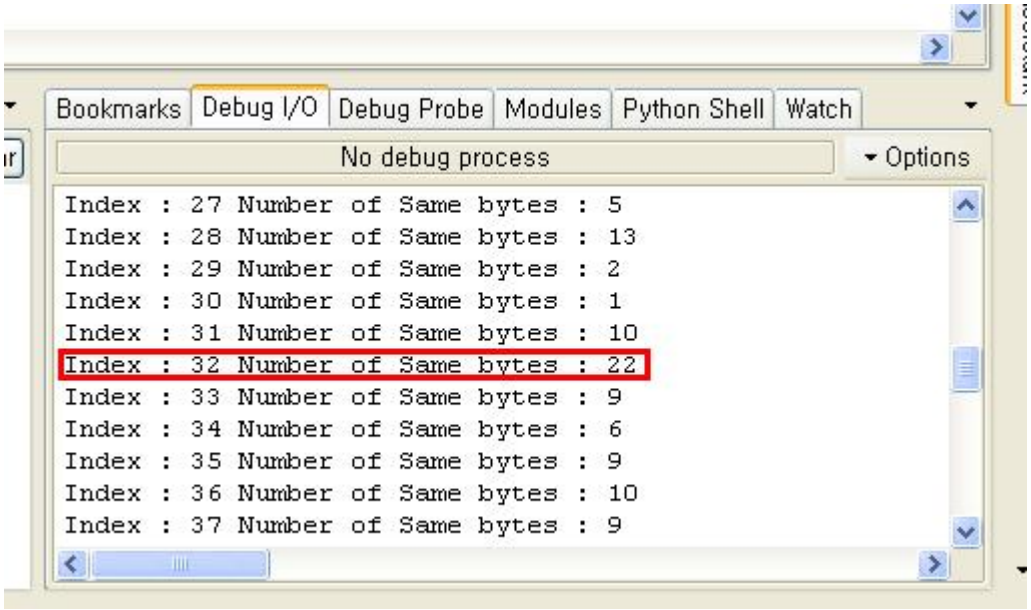
Possible Source code of the Encryption Scheme

```
char[] key = "1234567890"

for(int i=0; i<filesize; i++){
        encrypted_data[i] = data[i]^key[i % strlen(key)]
}
```

- Why? Because it is the simplest encryption algorithm with a key to implement! **J**
- It also matches our theory of Polyalphabetic Cipher, or more like, a Vigenère cipher

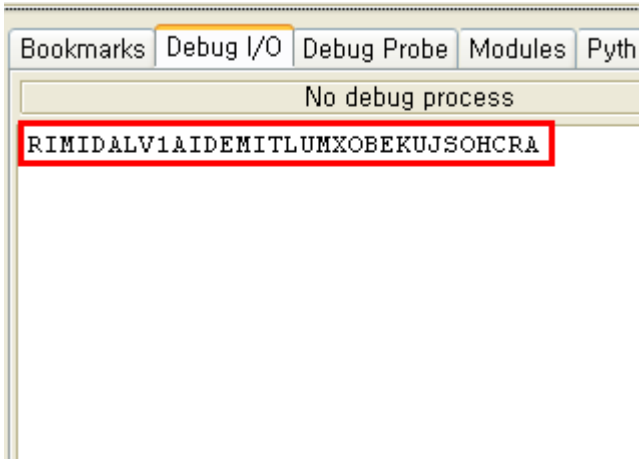
Attacking Vigenère Ciphers - Coincidence Counting -



Key Length : 32 bytes

Making Assumptions

- Key is XORed
- Key is Added
- Key is Subtracted
- ~~Key is RORed~~
- ~~Key is ROLed~~
- …
- …



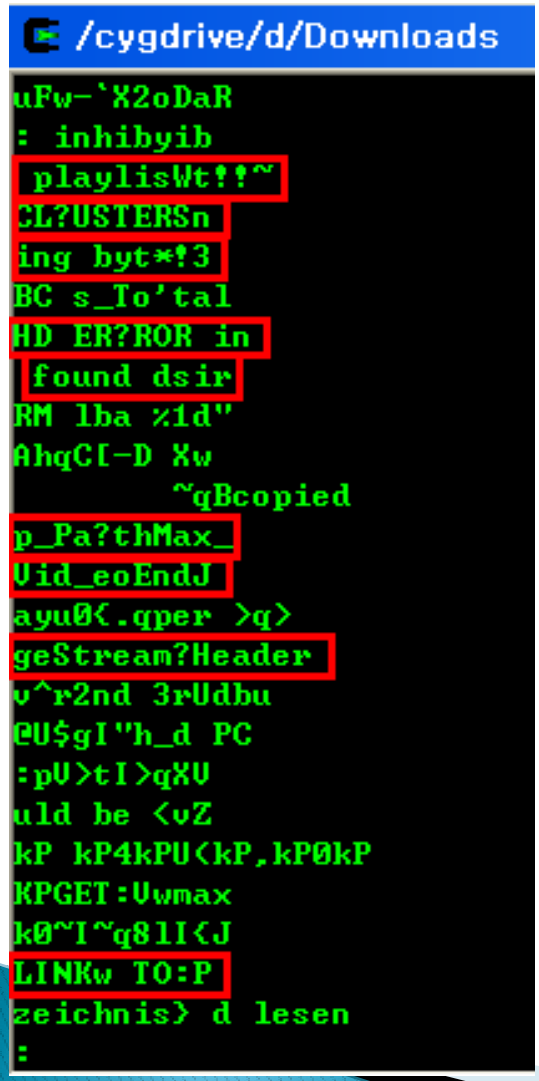
Key Found! **J**

RIMIDALV1AIDEMITLUMXOBEKUJSOHCRA

Decrypted File

```
00000000 FF00 000F E11F 00C0 E3FF D300 80E3 00F0 29E1 FF48 009F E548 209F E5FF 0020 52E0 ................)..H...H .... R.
00000020 0E00 000A FF04 10A0 E304 3090 E4FF 0430 81E4 0420 52E2 2FFB FFFF 1AEE F140 F3F0 .........0....0... R./......@..
00000040 FAF1 F501 0F00 1CFF F0B0 10C0 E1F5 1833 0414 3304 C417 00EA FF80 0000 0834 5D00 ..............3..3.........4].
00000060 08FF 0003 0300 8005 0300 79A0 5B00 6109 14F0 9FE5 6E0D EF09 0000 EA6E 01F8 9A04 ..........y.[.a.....n.....n....
00000080 BF08 109B 0408 908B 0074 FA8B 0098 8F00 8801 0000 D4FE 6108 04E0 4EE2 10E0 0FBF ..........t...........a...N.....
000000A0 E500 E04F E11C B302 0FBF E113 E08E E30E FBF0 08FF E02D E52C E01F E504 EFE0 8DE5 ...O.....................-,........
000000C0 38CF 000F 502D 7FE9 0FE0 A0E1 4CF0 D801 FFBD E80E F069 E100 C0FF FDE8 0000 A0E1 8...P-.....L.......i.........
000000E0 7847 BBC0 4622 091E FF2F F506 80E0 FF0F 1110 F8F3 0619 0611 5893 9FFF E501 8BA0 xG..F"../.............X.......
00000100 E30A 8048 FFE2 B880 19E1 010C 187F E30C 0000 1A44 A304 00FF 809A E501 8088 E200 ...H.............D.............
00000120 FF80 8AE5 88A0 A0E3 13FF AC8A E20A 0058 E107 FF00 00DA 0080 A0E3 46FF AEA0 E3BA ..............X.........F.....
00000140 8009 E1B0 FF80 C9E1 04F0 5EE2 00F6 6710 0C83 0400 A088 E508 BE8F 1298 E501 A08A ..........^...g.................
00000160 8910 88FE 9914 0AE2 B0A0 C9E1 F0F5 A290 119A 5910 98E5 1F80 FD08 6D12 FEAF A093 ..................Y.......m.....
00000180 FDAF E7A0 8301 7710 7E11 01A7 A0FF E3FD AC4A E2B0 80CA C7E1 12AD 7C13 8611 F601 ......w.~........J........|.....
000001A0 1040 FF2D E901 C0A0 E3AC 02BE 0400 C080 E5A8 4234 00C0 BFD4 E18C CFB0 E182 001A .@.-.............B4............
000001C0 EF44 00A0 E3DE 0194 F29F 95E5 DE01 901B 248C 1B20 0625 F5FF FFFF 0A06 CDA0 E3BC .D.............$.. .%.........
000001E0 FFC0 14E1 1000 1CE3 0B66 5310 16CE 3C27 7210 0A04 1324 5558 1B20 3013 2440 1B24 .........fS...<'r....$UX. 0.$@.$
00000200 481B 20A2 3A25 0443 2C82 2156 2102 1324 18EA 6328 001B 240C 1B20 02CC A01F E382 H. .:%.C,.!V!..$..c(..$.. ......
00000220 C04C E23E 250E 21BA 21D5 A2BF 2804 0B00 3813 24C4 F112 1C23 D4EB 20BA 2920 C72F .L.>%.!.!...(...8.$....#.. .) ./
00000240 0132 DE29 2588 EB24 9CF3 2C0A 2105 CB2C 0A21 BEDE 0011 74F1 9F15 BA29 0250 033F .2.)%..$..,.!..,.!....t....).P.?
00000260 6932 DE21 6625 20EB 243C F32C 9082 21CA 2D82 2182 39E4 6F00 DE01 0486 F32C ACC7 i2.!f% .$<.,..!.-.!.9.o.........,..
00000280 0C23 CE29 E231 DE21 39CA 1324 A8CB 34CC 6F00 5E3A 0A1C 5051 1272 3A1F 4056 2141 .#.).1.!9..$..4.o.^:..PQ.r:.@V!A
000002A0 1324 6C6F 0011 1D3B 4822 320F 4C80 A73F 6542 DE21 D517 1324 30CB 3450 6F00 1040 .$lo...;H"2.L..?eB.!...$0.4Po..@
000002C0 FBBD E806 1100 0A03 00F0 FF5E 0000 C05C 0000 ECFA A340 E8A3 4002 0703 0090 FD44 ...........^...\.....@..@.....D
000002E0 A000 4704 0858 4304 FB08 E0DE 2054 1000 0098 FE46 2080 7304 0800 7404 9B08 D442 ..G..XC..... T.....F .s...t..._B
```

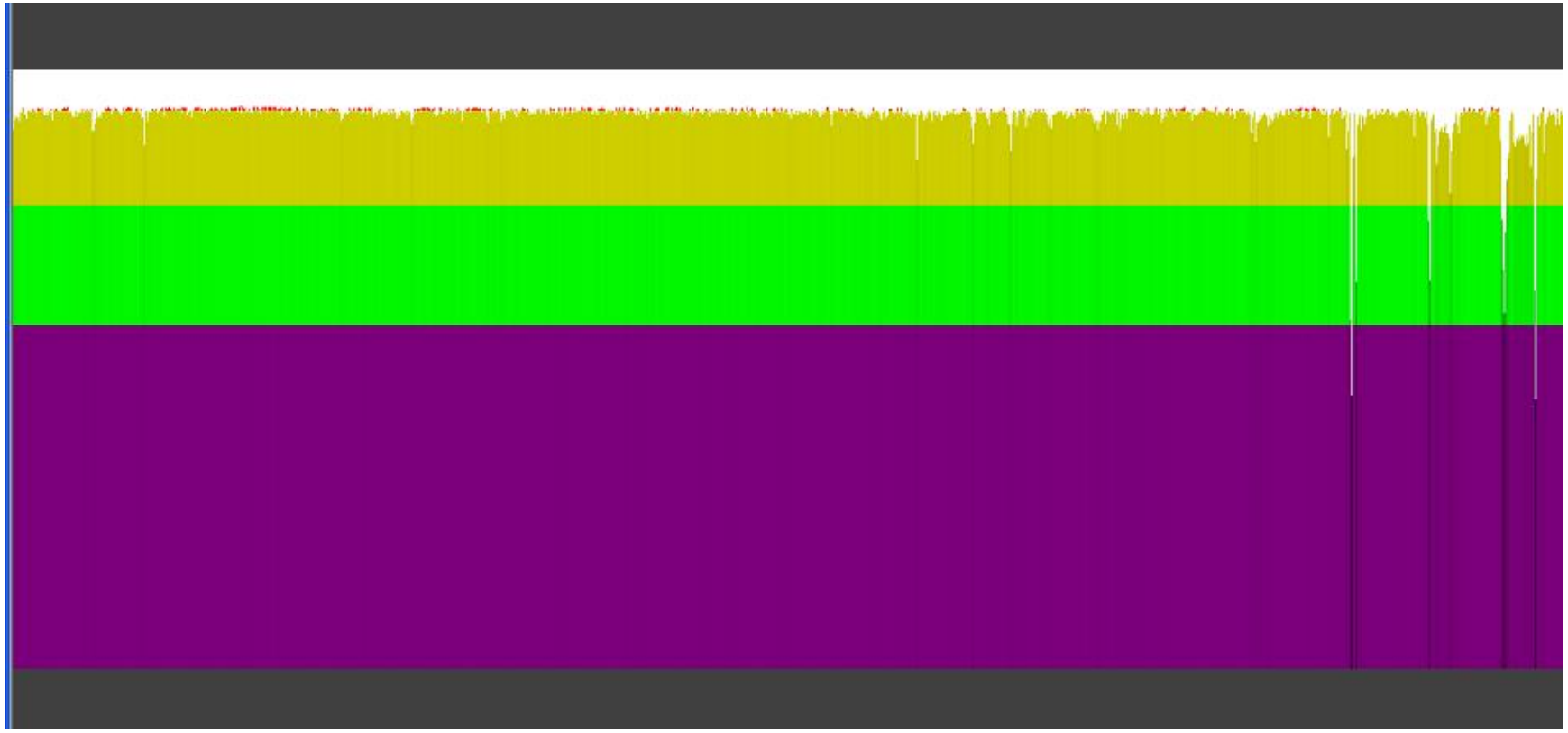
Searching for strings



- Strings are weirdly distorted
- 1 byte trash in the Middle of Strings
- What could this mean???
- What to do next?

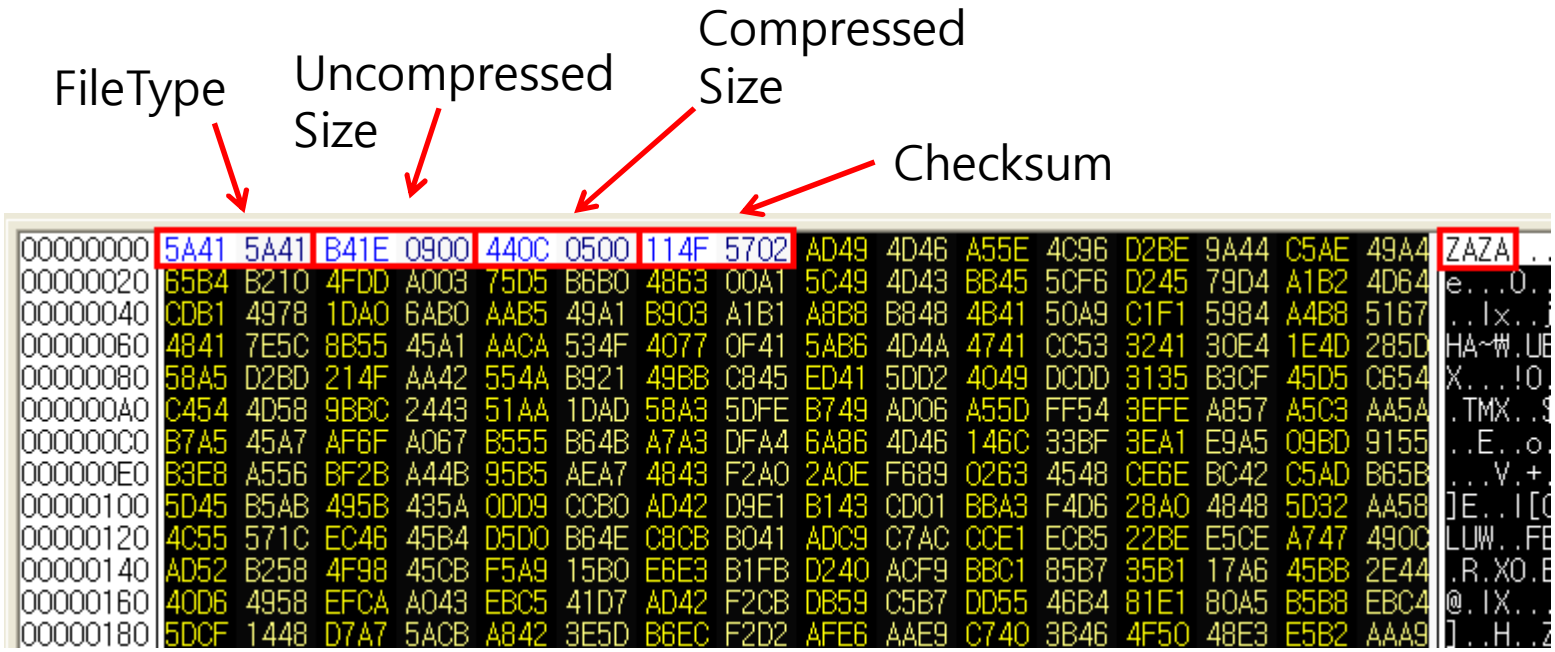
→ Calculate the Entropy

Entropy divided in 256 byte blocks



→ File is possibly Compressed

Refining the Header Structure



- Compression Ratio : 1.8

Choosing from the candidates

- Run Length Encoding
- Arithmetic coding
- Huffman
- LZ77
- LZ78

- ~~Run Length Encoding~~

(Repeating bytes exist)

- ~~Arithmetic coding~~

(No text remains)

- ~~Huffman~~

(No text remains)

- LZ77
- LZ78

A little intro on LZ77 Compression

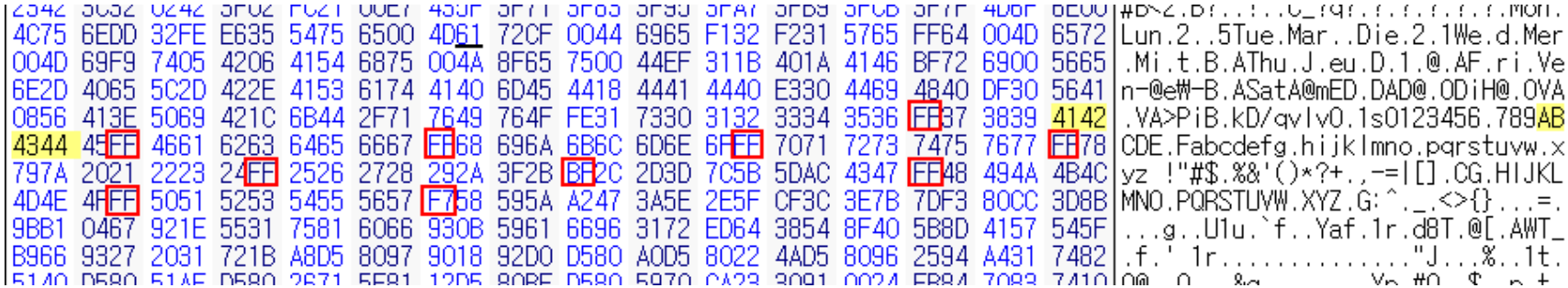
- the_rain_<3,3>Sp<9,4>falls
- <3,3> : go 3 bytes back, copy 3 bytes from there and paste at current location.

- The_rain_in_Sp<9,4>falls
- <9,4> : go 9 bytes back, copy 4 bytes from there and paste at current location.

- The_rain_in_Spain_falls

Compressed bytes info data(CBID) contains
<last occurred matching sequence position, length>

Recognizing Patterns



- Every n bytes there exists a Header byte
- HeaderByte probably tells the number of bytes until the next HeaderByte, and the position of the CBID(s)

- HeaderByte 0xFF means the next 8 bytes are uncompressed

Trying to guess the meaning of the Header Bytes

- We know that the Header byte contains the following information :
1. The number of bytes until the next Header
2. The position of the CBID(s)
- Start attacking no.1 theory by collecting samples.

FB : 9	EF : 9	BE : 10	DF : 9
FF : 8	EA : 11	AB : 11	AF : 10
CF : 10	EB : 10	0F : 12	F9 : 10
FE : 9	EE : 10	BF : 9	FD : 9

Guessing

- Compile Header Bytes with the same length(9)

FB : 9	EF : 9	DF : 9	FE : 9	BF : 9

- Use your brain and think. 'Why do all those different bytes end up in the same length?'
- Make assumptions.

(Length info is contained in High nibble | Low nibble | bitfields | etc…)

- Verify your theory by testing it on other samples.

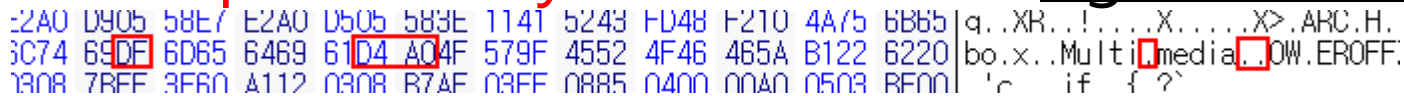
Answer : Bitfields

- Bit 0 means length = 2
- Bit 1 means length = 1

ex) 0xBF = 10111111 = 7*1 + 1*2 = 9

0xAC = 10101100 = 4*1 + 4*2 = 12

- Moreover, 0 is a CBID, while 1 is a normal uncompressed byte. Read from right to left.



- 0xDF = 11011111
- = 5 uncompressed , CBID, 2 uncompressed
- “media”,₩xD4₩xA0, “OW”

Confirm your theory by writing a parser/verifier

6769	0x11fb8	0xfd	0x9
6770	0x11fc2	0xff	0x8
6771	0x11fcb	0xdf	0x9
6772	0x11fd5	0x9f	0xa
6773	0x11fe0	0xfa	0xa

- Offsets found! **J**
- Verify for 5 or more header bytes just to make sure.
- The theory is correct.
- Something new learned : CBID is 2 bytes

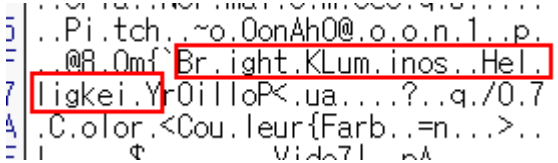
Trying to understand the meaning of the CBID

- What we already know about the CBID :
1. Contains the size of the compressed bytes
2. Contains the position of the compressed bytes
- Start attacking no.1 theory by collecting samples.

Figuring out which byte contains the length field

- "Firmw.are V..io." ≃ "Firmware Version"
- ∴ C8B0 → 3 bytes

- "H{.igkei" = "Helligkei"
- ∴ 7B80 → 3 bytes



- "Cre.a..Folder" = Create Folder
- ∴ B3A0 → 3 bytes

Guessing

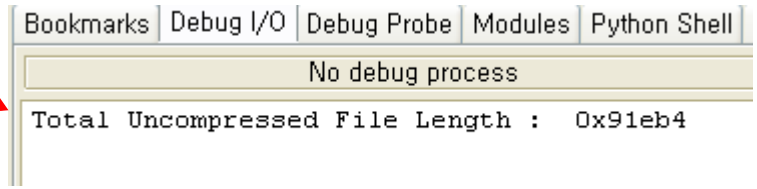
∴ C8B0 → 3 bytes

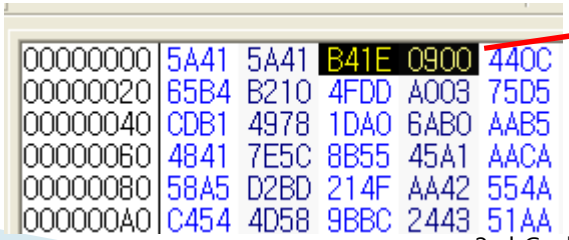
∴ 7B80 → 3 bytes

∴ B3A0 → 3 bytes

- The last nibble is all 0 for all CBIDs.
- (Compressed bytes length = Last Nibble + 3) ?
- Verify by parsing the file using that assumption.

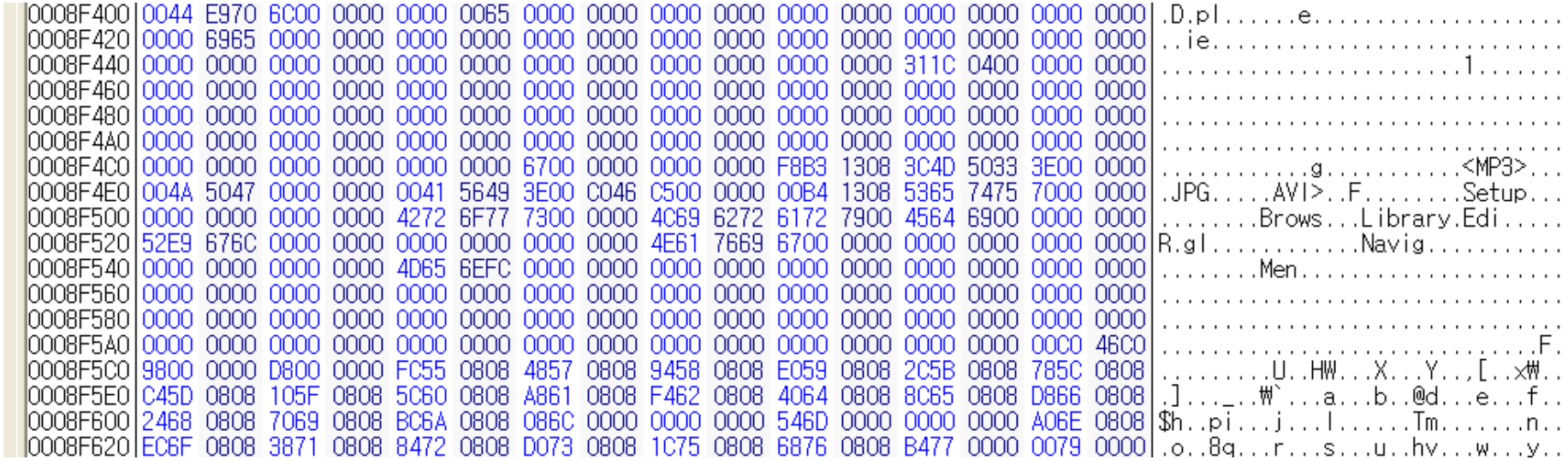
- Matches ! **J**
- Theory is Correct.





Uncompressed File Size

Uncompress the whole file using the newly found length field



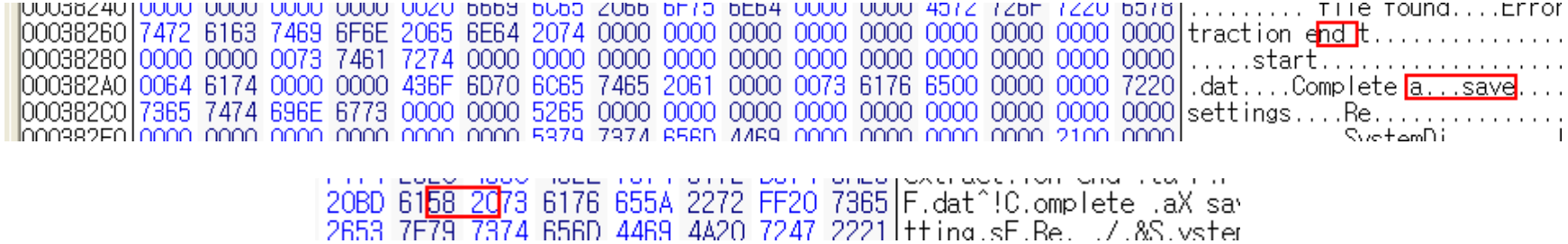
```
0008F400 0044 E970 6C00 0000 0000 0065 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 .D.pl.....e.....................
0008F420 0000 6965 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 ..ie.....................................
0008F440 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 311C 0400 0000 0000 ........................1.......
0008F460 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 ................................
0008F480 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 ................................
0008F4A0 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 ................................
0008F4C0 0000 0000 0000 0000 0000 0000 6700 0000 0000 0000 F8B3 1308 3C4D 5033 3E00 0000 ............g...........<MP3>...
0008F4E0 004A 5047 0000 0000 0041 5649 3E00 C046 C500 0000 00B4 1308 5365 7475 7000 0000 .JPG.....AVI>..F........Setup...
0008F500 0000 0000 0000 0000 4272 6F77 7300 0000 4C69 6272 6172 7900 4564 6900 0000 0000 ........Brows...Library.Edi.....
0008F520 52E9 676C 0000 0000 0000 0000 0000 0000 4E61 7669 6700 0000 0000 0000 0000 0000 R.gl............Navig...........
0008F540 0000 0000 0000 0000 4D65 6EFC 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 ........Men.....................
0008F560 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 ................................
0008F580 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 ................................
0008F5A0 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 00C0 46C0 ..............................F.
0008F5C0 9800 0000 D800 0000 FC55 0808 4857 0808 9458 0808 E059 0808 2C5B 0808 785C 0808 .........U..HW...X...Y..,[..x₩..
0008F5E0 C45D 0808 105F 0808 5C60 0808 A861 0808 F462 0808 4064 0808 8C65 0808 D866 0808 .]..._..₩`...a...b..@d...e...f..
0008F600 2468 0808 7069 0808 BC6A 0808 C86C 0000 0000 0000 546D 0000 0000 0000 A06E 0808 $h..pi...j...l.....Tm.......n..
0008F620 EC6F 0808 3871 0808 8472 0808 D073 0808 1C75 0808 6876 0808 B477 0000 0079 0000 .o..8q...r...s...u..hv...w...y..
```

- The compressed bytes are still unknown and are filled with zeroes

Guessing the remaining 3 nibble field meaning

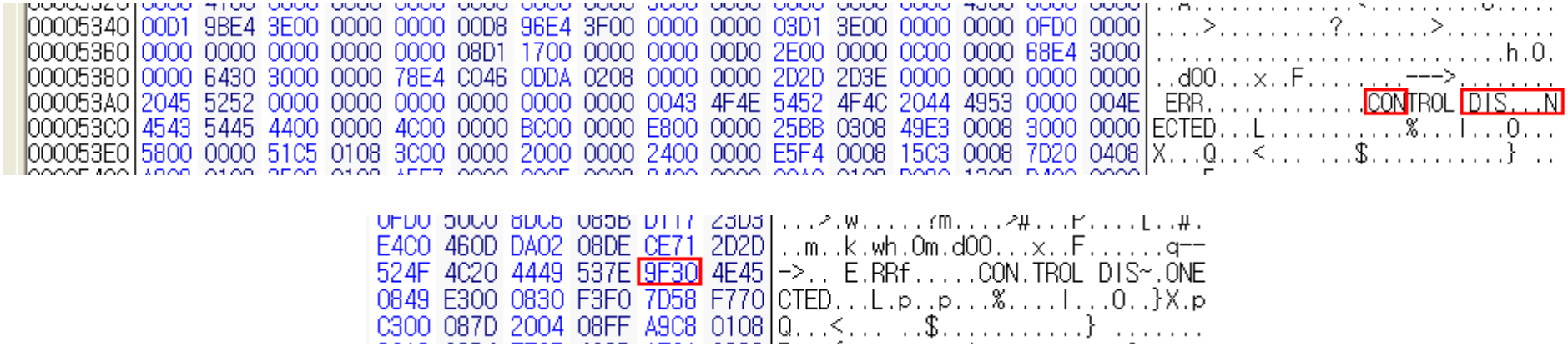
- We know that it represents the position of the last occurred matching bytes
- We know by the length field that the minimum length of the matching bytes is 3
- Collect samples where you can guess the compressed bytes, in which the bytes identical to the resulting bytes are located somewhere before the compressed bytes, where you can see them.

Collecting Samples (1)



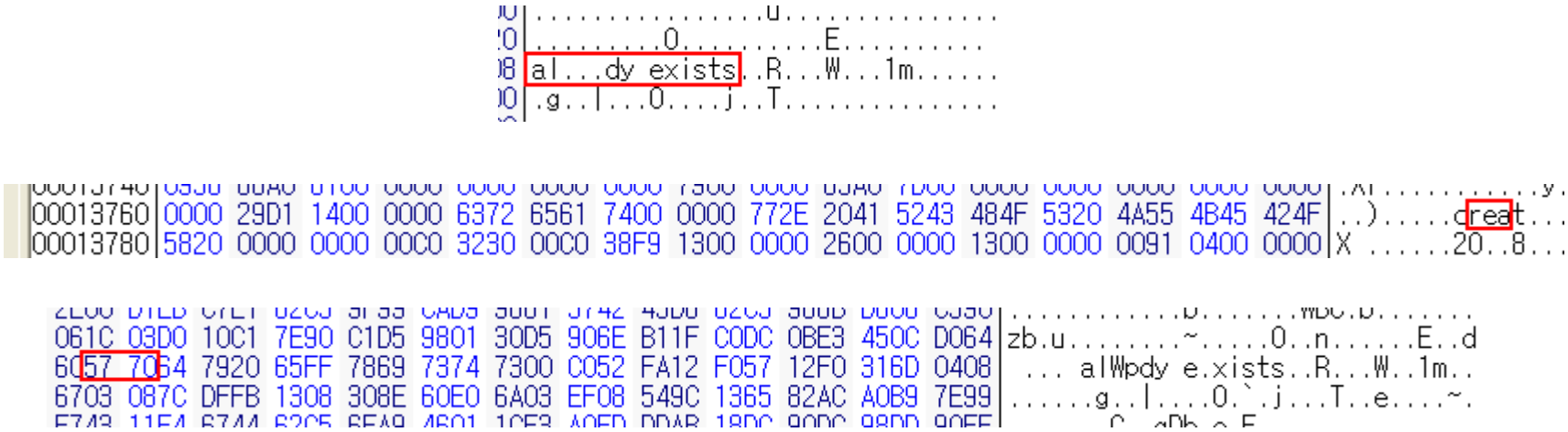
- "a...save" = "and save"
- Compressed bytes = "nd "
- Last occurred at 0x3826A and 0x48 bytes before the compressed bytes
- CBID position field = 0x258

Collecting Samples (2)



- "DIS…NECTED" = "DISCONNECTED"
- Compressed bytes = "CON"
- Last occurred at 0x53B1 and 0xB bytes before the compressed bytes
- CBID position field = 0x39F

Collecting Samples (3)



- "al...dy exists" = "already exists"
- Compressed bytes = "rea"
- Last occurred at 0x13769 and 0x6B9 bytes before the compressed bytes
- CBID position field = 0x757

Analyzing the samples

- 0x258 ≠ 0x3826A, 0x48
- But 0x258 is very close to 0x26A !
- In fact, 0x757 is also very close to 0x769, and so is 0x39F and 0x3B1
- Carefully observing the difference between the 2 numbers, we realize that the difference is constant !
- The difference between the two numbers is 0x12, therefore, the format of the CBID is
- <Offset of the last occurred identical byte sequence - 0x12(low 3 nibbles), Length – 3>

File Parsing Script

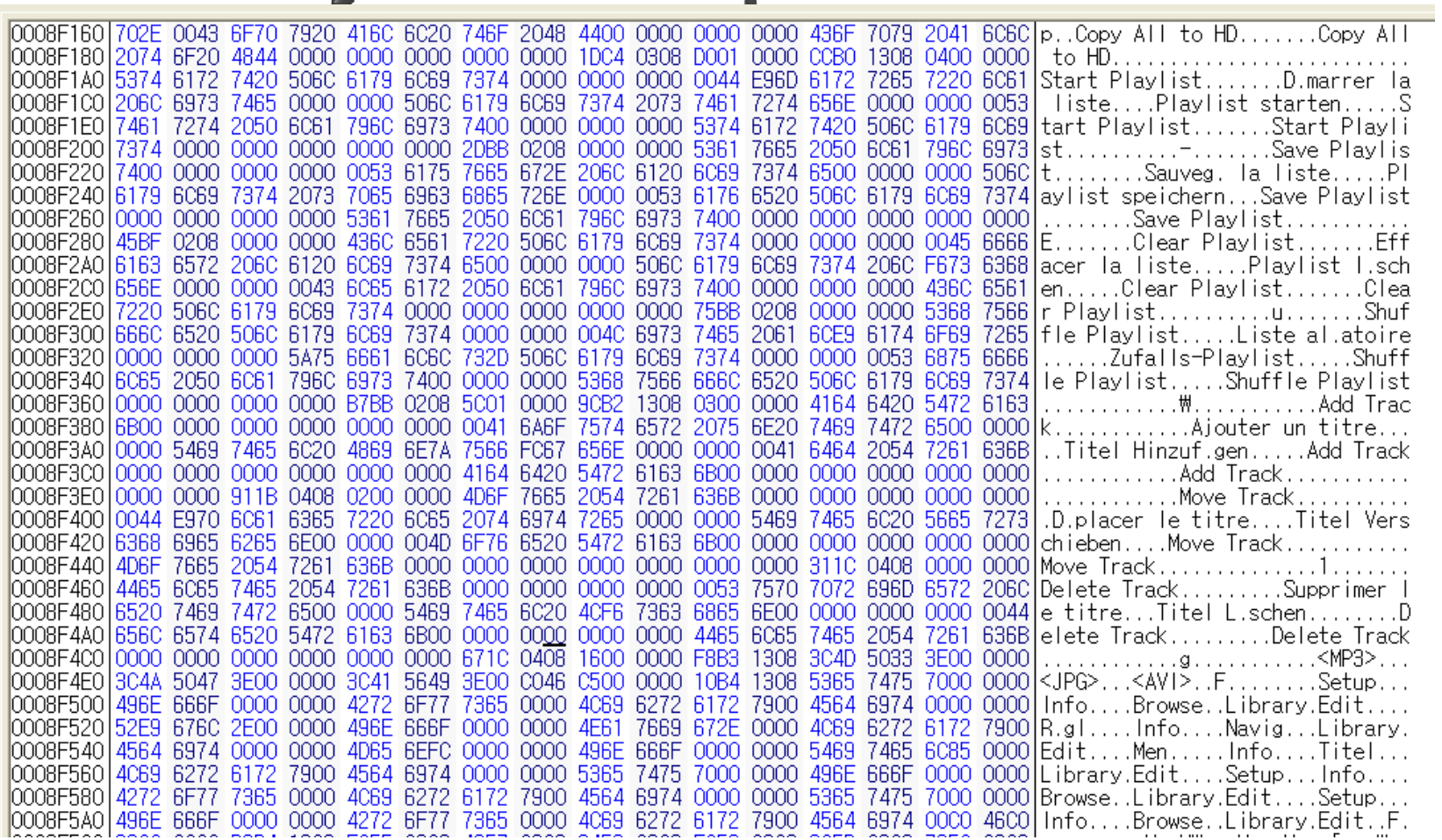
- Using the Above facts, uncompress the entire file and dump it into a new file.

```
uncompressed_filesize = 0
index = 0

while index < filesize:
    headerbyte = struct.unpack("B",data[index])[0]
    savedheaderbyte = headerbyte
    chunksize = 0
    for i in range(8):
        if (headerbyte & 1) == 0:
            compress_code = struct.unpack("H",data[index+chunksize+1:index+chunksize+3])[0]
            compressed_bytes_size = ((compress_code & 0xF00) >> 8) + 3
            compressed_bytes_offset = (compress_code & 0xFF) + ((compress_code & 0xF000) >> 4) + 0x12
            saved_offset = outfile.tell()
            compressed_bytes_offset += (saved_offset & 0xFFFFF000)
            if compressed_bytes_offset >= saved_offset:
                compressed_bytes_offset -= 0x1000
            outfile.seek(compressed_bytes_offset)
            uncompressed_bytes = outfile.read(compressed_bytes_size)
            outfile.seek(saved_offset)
            outfile.write(struct.pack("%ds"%compressed_bytes_size,uncompressed_bytes))
            chunksize += 2
        else:
            outfile.write(data[index+chunksize+1])
            chunksize += 1
        if index+chunksize+1 == filesize:
            break
        headerbyte = headerbyte >> 1
    index += chunksize + 1
print hex(uncompressed_filesize)

outfile.close()
```

Fully Uncompressed File



```
0008F160 702E 0043 6F70 7920 416C 6C20 746F 2048 4400 0000 0000 0000 436F 7079 2041 6C6C p..Copy All to HD.......Copy All
0008F180 2074 6F20 4844 0000 0000 0000 0000 0000 1DC4 0308 D001 0000 CCB0 1308 0400 0000  to HD..........................
0008F1A0 5374 6172 7420 506C 6179 6C69 7374 0000 0000 0000 0044 E96D 6172 7265 7220 6C61 Start Playlist.......D.marrer la
0008F1C0 206C 6973 7465 0000 0000 506C 6179 6C69 7374 2073 7461 7274 656E 0000 0000 0053  liste....Playlist starten.....S
0008F1E0 7461 7274 2050 6C61 796C 6973 7400 0000 0000 0000 5374 6172 7420 506C 6179 6C69 tart Playlist.......Start Playli
0008F200 7374 0000 0000 0000 0000 0000 2DBB 0208 0000 0000 5361 7665 2050 6C61 796C 6973 st..........-.......Save Playlis
0008F220 7400 0000 0000 0000 0053 6175 7665 672E 206C 6120 6C69 7374 6500 0000 0000 506C t........Sauveg. la liste.....Pl
0008F240 6179 6C69 7374 2073 7065 6963 6865 726E 0000 0053 6176 6520 506C 6179 6C69 7374 aylist speichern...Save Playlist
0008F260 0000 0000 0000 0000 5361 7665 2050 6C61 796C 6973 7400 0000 0000 0000 0000 0000 ........Save Playlist...........
0008F280 45BF 0208 0000 0000 436C 6561 7220 506C 6179 6C69 7374 0000 0000 0000 0045 6666 E.......Clear Playlist.......Eff
0008F2A0 6163 6572 206C 6120 6C69 7374 6500 0000 0000 506C 6179 6C69 7374 206C F673 6368 acer la liste.....Playlist l.sch
0008F2C0 656E 0000 0000 0043 6C65 6172 2050 6C61 796C 6973 7400 0000 0000 0000 436C 6561 en.....Clear Playlist.......Clea
0008F2E0 7220 506C 6179 6C69 7374 0000 0000 0000 0000 0000 75BB 0208 0000 0000 5368 7566 r Playlist.........u.......Shuf
0008F300 666C 6520 506C 6179 6C69 7374 0000 0000 004C 6973 7465 2061 6CE9 6174 6F69 7265 fle Playlist.....Liste al.atoire
0008F320 0000 0000 0000 5A75 6661 6C6C 732D 506C 6179 6C69 7374 0000 0000 0053 6875 6666 ......Zufalls-Playlist.....Shuff
0008F340 6C65 2050 6C61 796C 6973 7400 0000 0000 5368 7566 666C 6520 506C 6179 6C69 7374 le Playlist.....Shuffle Playlist
0008F360 0000 0000 0000 0000 B7BB 0208 5C01 0000 9CB2 1308 0300 0000 4164 6420 5472 6163 ............₩...........Add Trac
0008F380 6B00 0000 0000 0000 0000 0000 0041 6A6F 7574 6572 2075 6E20 7469 7472 6500 0000 k............Ajouter un titre...
0008F3A0 0000 5469 7465 6C20 4869 6E7A 7566 FC67 656E 0000 0000 0041 6464 2054 7261 636B ..Titel Hinzuf.gen.....Add Track
0008F3C0 0000 0000 0000 0000 0000 0000 4164 6420 5472 6163 6B00 0000 0000 0000 0000 0000 ............Add Track...........
0008F3E0 0000 0000 911B 0408 0200 0000 4D6F 7665 2054 7261 636B 0000 0000 0000 0000 0000 ............Move Track...........
0008F400 0044 E970 6C61 6365 7220 6C65 2074 6974 7265 0000 0000 5469 7465 6C20 5665 7273 .D.placer le titre....Titel Vers
0008F420 6368 6965 6265 6E00 0000 004D 6F76 6520 5472 6163 6B00 0000 0000 0000 0000 0000 chieben....Move Track...........
0008F440 4D6F 7665 2054 7261 636B 0000 0000 0000 0000 0000 0000 0000 311C 0408 0000 0000 Move Track.............1.......
0008F460 4465 6C65 7465 2054 7261 636B 0000 0000 0000 0000 0053 7570 7072 696D 6572 206C Delete Track.........Supprimer l
0008F480 6520 7469 7472 6500 0000 5469 7465 6C20 4CF6 7363 6865 6E00 0000 0000 0000 0044 e titre...Titel L.schen........D
0008F4A0 656C 6574 6520 5472 6163 6B00 0000 0000 0000 0000 4465 6C65 7465 2054 7261 636B elete Track.........Delete Track
0008F4C0 0000 0000 0000 0000 0000 0000 671C 0408 1600 0000 F8B3 1308 3C4D 5033 3E00 0000 ............g...........<MP3>...
0008F4E0 3C4A 5047 3E00 0000 3C41 5649 3E00 C046 C500 0000 10B4 1308 5365 7475 7000 0000 <JPG>...<AVI>..F........Setup...
0008F500 496E 666F 0000 0000 4272 6F77 7365 0000 4C69 6272 6172 7900 4564 6974 0000 0000 Info....Browse..Library.Edit....
0008F520 52E9 676C 2E00 0000 496E 666F 0000 0000 4E61 7669 672E 0000 4C69 6272 6172 7900 R.gl....Info....Navig...Library.
0008F540 4564 6974 0000 0000 4D65 6EFC 0000 0000 496E 666F 0000 0000 5469 7465 6C85 0000 Edit....Men.....Info....Titel...
0008F560 4C69 6272 6172 7900 4564 6974 0000 0000 5365 7475 7000 0000 496E 666F 0000 0000 Library.Edit....Setup...Info....
0008F580 4272 6F77 7365 0000 4C69 6272 6172 7900 4564 6974 0000 0000 5365 7475 7000 0000 Browse..Library.Edit....Setup...
0008F5A0 496E 666F 0000 0000 4272 6F77 7365 0000 4C69 6272 6172 7900 4564 6974 00C0 46C0 Info....Browse..Library.Edit..F.
```

- All strings are neatly displayed! Which means the whole file is uncompressed. **J**

Questions

ThankYou

안기찬 ; Ahn Ki-Chan

wringer@paran.com

www.Externalist.org