

JUST CTF 2020

ROXY Write-up

- The description says “Help me! Find little toy roxy.”, the attached files was “roxy.jpg”, download the file and let’s start working.
- First, we use “**file**” command to determine the file type as following:

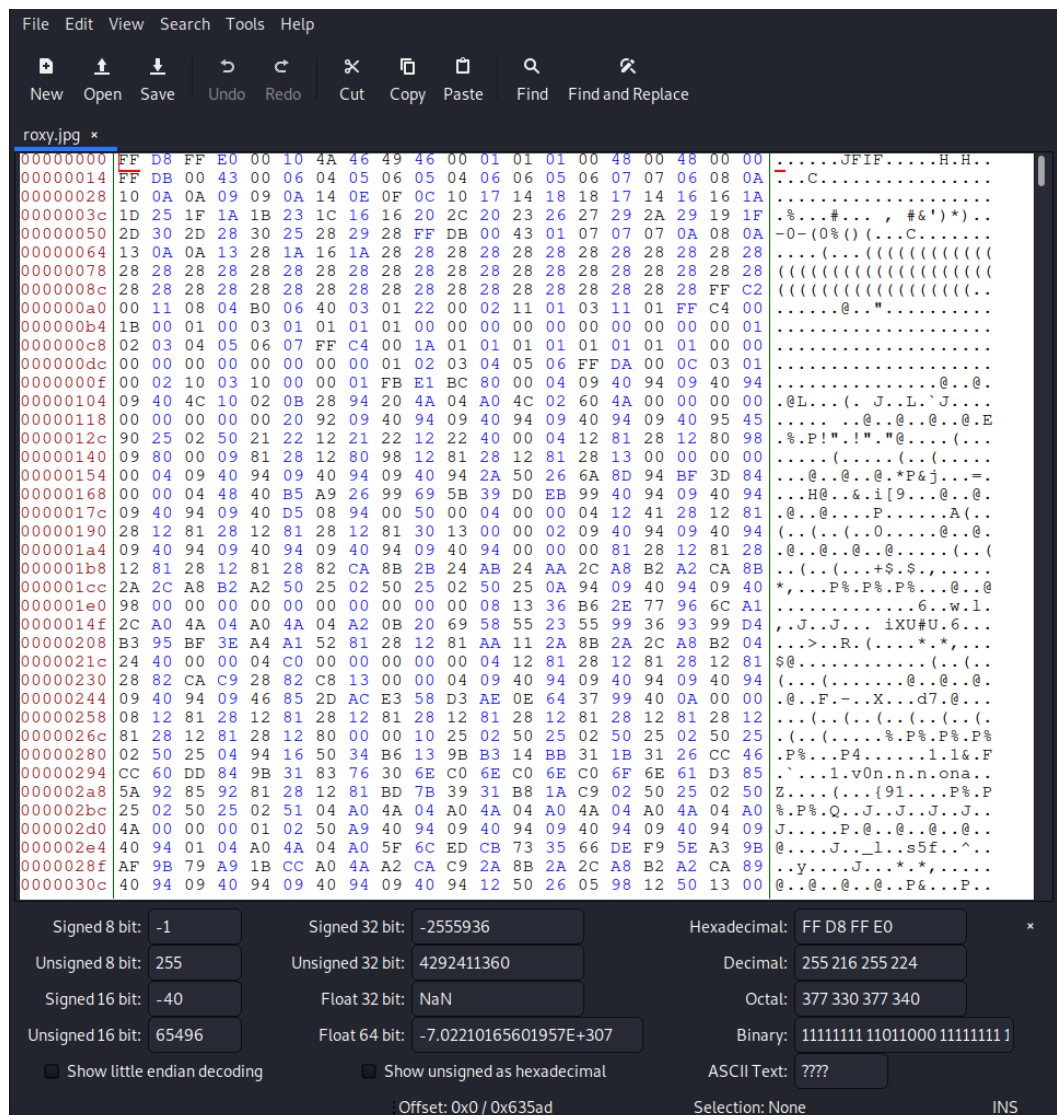
```
FILE(1) BSD General Commands Manual FILE(1)

NAME
    file - determine file type

SYNOPSIS
    file [-bcdEhiklLnprsvzZ] [--apple] [--extension] [--mime-encoding] [--mime-type] [-e testname]
    [-F separator] [-f namefile] [-m magicfiles] [-P name=value] file ...
    file -C [-m magicfiles]
    file [--help]

$ file roxy.jpg
roxy.jpg: JPEG image data, JFIF standard 1.01, resolution (DPI), density 72x72, segment length 16, progressive, pr
ecision 8, 1600x1200, components 3
[cryp70na@KALI]--[~/Downloads/justCTF]
```

The output indicates that “roxy.jpg” is a *JPEG image* file, let’s dump the file contents into hexadecimal format. For this to be done, you can use one of the hexadecimal editor like :“hexdump” , “xxd”, or the one I prefer “bless”,which has a graphical interface.



- Thing you have to know is the Magic numbers, which are the first bits of a file that uniquely identify the type of the file.
- The following table contains a samples of these magic number :

File type	Typical extension	Hex digits xx = variable	Ascii digits . = not an ascii char
Bitmap format	.bmp	42 4d	BM
FITS format	.fits	53 49 4d 50 4c 45	SIMPLE
GIF format	.gif	47 49 46 38	GIF8
Graphics Kernel System	.gks	47 4b 53 4d	GKSM
IRIS rgb format	.rgb	01 da	..
ITC (CMU WM) format	.itc	f1 00 40 bb
JPEG File Interchange Format	.jpg	ff d8 ff e0
NIFF (Navy TIFF)	.nif	49 49 4e 31	IIN1
PM format	.pm	56 49 45 57	VIEW
PNG format	.png	89 50 4e 47	.PNG
Postscript format	.[e]ps	25 21	%!
Sun Rasterfile	.ras	59 a6 6a 95	Y.j.
Targa format	.tga	xx xx xx	...
TIFF format (Motorola - big endian)	.tif	4d 4d 00 2a	MM.*
TIFF format (Intel - little endian)	.tif	49 49 2a 00	II*.
X11 Bitmap format	.xbm	xx xx	
XCF Gimp file structure	.xcf	67 69 6d 70 20 78 63 66 20 76	gimp xcf
Xfig format	.fig	23 46 49 47	#FIG

As the table shown, JPEG images start with “FF D8 FF E0”, and this can be obviously seen in *bless*.

```

roxy.jpg x
00000000 ff d8 ff e0 00 10 4a 46 49 46 00 01 01 01 00 48 00 48 00 00 .....JFIF.....H.H..
00000014 ff db 00 43 00 06 04 05 06 05 04 06 06 05 06 07 07 06 08 0a ...C.....

```

roxy.jpg *

00031808	CC 13 BD F1 80 ED 88 9F 71 A3 2B D1 28 E3 5A 1C 87 B7 B7 C4q+.(.Z.....
0003181c	09 CD F3 EB CC 6A F9 27 6B E3 A2 20 EC D2 38 E9 2A F2 BA BDj.'k'...8.*...
00031830	1E 88 A0 A5 BB 7A 82 00 2E B5 D4 1A 03 C9 5A F1 1A E0 7C A0z.....Z.... .
00031844	C7 0E 03 96 17 A0 96 5C 07 6C 39 56 B0 EF DB E2 74 3F 62 62\l9V....t?bb
00031858	2D 9F 75 88 3B 17 A3 88 43 4D FE 18 0E AA F7 75 0B D4 6E 07	-.u.;...CM....u..n.
0003186c	6E 5E 02 62 72 F4 7F 70 E6 EF 4F 25 E5 9D C3 93 76 AF 72 9E	n^.br..p.O%.v.r.
00031880	65 3B 57 1D 85 BD 9B A7 AF 50 3C 80 B5 13 4E D7 FE D4 46 28	e;W.....P<...N...F(
00031894	6B 8E A5 9A 32 71 10 C0 0E 8F 11 7B C2 36 CC 18 3C 95 DB 30	k...2q....{.6...<..0
000318a8	04 C8 F9 62 36 68 32 F0 EE 35 A1 E0 F5 D3 18 AC A0 34 47 C6	...b6h2..5.....4G.
000318bc	E5 F2 AB E8 8D AA 8B AC 70 4B 9A 6F AF C1 1C DC AB 20 DC 21pK.o.....!
000318d0	21 CE 87 88 B3 B8 CC 59 9A BA 5C 20 88 6C DA C3 B1 85 0B AA	!.....Y..\ .l.....
000318e4	31 2D 95 89 86 03 56 AC 81 00 4A 79 85 4F 3A F9 99 F2 DF DA	1-....V....Jy.O:....
000318f8	78 77 26 2A 9D DD 71 D4 5E 3B 00 0F 31 1E 5D 09 67 17 19 07	xw&*.q.^;..1.].g...
0003190c	0D 0A 83 65 51 B5 AD 43 AF 2E CF 6F 11 28 61 CC 19 54 D0 B4	...eQ..C...o.(a..T..
00031920	AF 10 55 65 43 7D C0 B5 31 94 44 34 B8 24 49 5A 5A BC 9A 8C	..UeC)..l.D4.\$IZZ...
00031934	23 4E 88 44 F2 AB 6E 29 4A C5 40 F2 30 D7 BB 5E 1C C4 11 75	#N.D..n)J..0..^...u
00031948	68 EC 8C D4 5A 51 1E AE 58 8B 03 CF B8 5D 12 A8 89 DC A9 70	h...ZQ.X...].p
0003195c	4C 1D 33 18 30 C0 9D CA F6 97 C3 9B 81 4C 0E 12 1D 84 06 7B	L.3.0.....L.....{
00031970	56 99 90 2B 2D EC E1 9A A5 96 05 E4 84 BE 12 A1 75 0B 70 09	V..+.....u.p.
00031984	12 1A 45 D8 1B EA 13 99 22 D6 92 2C 1A 16 C0 7C 15 99 51	..E....."..... .Q
00031998	AD 64 F1 EA 00 98 2C A6 BF 70 07 B2 B4 3E 63 F9 83 B3 22 CF	.d.....,p...>c...".
000319ac	10 D8 8A 70 F6 4B B0 B5 92 F9 26 93 C8 60 69 C3 CF B9 82 1D	...p.K.....&..i.....
000319bc	67 B9 D3 15 DF DA 22 50 18 69 02 9A AF 0B D9 28 9E D1 B2 BB	g....."P.i.....(...
000319d4	80 97 32 6E 5D 5C AA C8 EF 87 A4 12 F6 16 06 8A E6 4B 16 6F	..2n]\.....K.o
000319e8	71 47 19 16 5B 17 E4 E1 81 96 83 42 33 86 70 44 0B 16 6C 85	qG...[.....B3.pD..l.
000319fc	D9 B6 30 AB 3C 07 F3 2C A2 8E 4D 25 B2 A4 79 88 99 E5 BC E0	..0.<.....M%.y.....
00031a10	C2 4A 99 FC 4B 88 69 45 E9 E7 D4 5E AF E3 24 F9 76 3F 88 F4	.J..K.iE...^..\$.v?..
00031a24	7E 79 11 AF 53 7D C5 CB DF 87 EA 34 E7 DE 6A 54 DC 16 03 C9	~y..S}.....4..jT....
00031a38	6C 9A 23 C8 E6 62 AF A2 17 79 E2 E6 6A D1 B6 FE A7 17 1A 8D	l.#..b...y..j.....
00031a4c	EA A8 A5 61 DE 95 39 77 28 60 EE C4 AA 2B 61 4B AE 13 70 1E	...a..9w('...+aK..p.
00031a60	17 A1 08 5D 3C 89 8E 47 6C 2C 97 91 C3 0F F6 CE 19 57 2B EE	...].<.Gl.....W+.
00031a74	23 C1 5D 7D 33 6D 1E B1 A6 59 E5 6F 30 69 6D 78 87 59 E9 8C	#.]}3m...Y.o0imx.Y..
00031a88	55 FB 8C E5 D7 46 21 65 54 E5 47 33 F8 28 BC 87 4E A2 0A 74	U....F!eT.G3.(..N..t
00031a9c	A0 96 4F 61 22 2E E0 36 98 7E 65 2B A1 37 3F FF D9 50 4B 03	..Oa"...6..~e+.7?..PK.
00031ab0	04 14 00 00 00 08 00 43 7F 37 50 2D 21 77 E6 5B 1A 03 00 AEC.7P-!w.[....
00031ac4	1A 03 00 08 00 1C 00 64 61 74 61 2E 70 6E 67 55 54 09 00 03data.pngUT....
00031ad8	DD 08 2A 5E DE 08 2A 5E 75 78 0B 00 01 04 00 00 00 00 04 00	..*^..*^ux.....

Search for: **FF D9** as Hexadecimal Find Next Find Previous

Signed 8 bit:	80	Signed 32 bit:	1347093252	Hexadecimal:	50 4B 03 04
Unsigned 8 bit:	80	Unsigned 32 bit:	1347093252	Decimal:	080 075 003 004
Signed 16 bit:	20555	Float 32 bit:	1.362389E+10	Octal:	120 113 003 004
Unsigned 16 bit:	20555	Float 64 bit:	6.25550110675295E+78	Binary:	01010000 01001011 00000C

☐ Show little endian decoding ☐ Show unsigned as hexadecimal ASCII Text: PK[FF D9]

Offset: 203437 / 406957 Selection: 203435 to 203436 (2 bytes) INS

- To determine the end of the JPEG image, it also has EOI “End Of Image” bytes, which are “FF D9”.
- Using *bless*, search for “FF D9”, and get the offset of that pattern, which is “203437”.
- Now we know that the first “203437” bytes of “roxy.jpg” are the actual image, and the rest of the file is extra data. In order to extract the actual image from the full file, we use the following command.

```
[cryp70n@KALI]--[~/Downloads/justCTF]
$head -c +203437 roxy.jpg | tee roxyNoData.jpg
```

```
[cryp70n@KALI]--[~/Downloads/justCTF]
$ls -l roxyNoDat.jpg
-rw-r--r-- 1 cryp70n cryp70n 203437 Apr  5 03:05 roxyNoDat.jpg
```

- Now, we can search for embedded files that the image may hold using *binwalk* tool.

```
[cryp70n@KALI]--[~/Downloads/justCTF]
$binwalk roxy.jpg
```

DECIMAL	HEXADECIMAL	DESCRIPTION
0	0x0	JPEG image data, JFIF standard 1.01
203437	0x31AAD	Zip archive data, at least v2.0 to extract, compressed size: 203355, uncompressed size: 203438, name: data.png
406936	0x63598	End of Zip archive, footer length: 22

The image holds ZIP archive, which contains “data.png” file, to extract the embedded data, we add option *-e* binwalk.

- The hint for this task was “Reverse my name”, Roxy becomes *yXOR*, so let’s try to perform XORing between “roxyNoDat.jpg” and “data.png”, to do so , I’ve used the following python script which reads images as bytes and perform XOR operation byte wise.

```
def str_xor(data, key):
    for i in range(len(data)):
        data[i] ^= key[i % len(key)]
    return data

key = bytearray(open('data.png', 'rb').read())
data = bytearray(open('roxyNoDat.jpg', 'rb').read())
encoded = str_xor(data, key)
open("t.jpg", "wb").write(encoded)
```



That was the result of XORing the two files, I took this image to StegSolver , and I’ve got the flag.

```
[cryp70n@KALI]--[~/opt/stegsolver]
$java -jar stegsolve.jar
Picked up _JAVA_OPTIONS: -Dawt.useSystemAAFontSettings=on -Dswing.aatext=true
StegSolve 1.3 by Caesium
```

JUST{h3r3_y0u_4r3}