




0x003图片隐写之pngcheck+隐藏二维码

原创

任骏锋  于 2018-06-26 18:07:03 发布  14823  收藏 18

分类专栏: [CTF之图片隐写](#) 文章标签: [CTF Misc](#) [图片隐写](#)

版权声明: 本文为博主原创文章, 遵循 [CC 4.0 BY-SA](#) 版权协议, 转载请附上原文出处链接和本声明。

本文链接: <https://blog.csdn.net/u010391191/article/details/80818785>

版权



[CTF之图片隐写 专栏收录该内容](#)

7 篇文章 0 订阅

订阅专栏

图片隐写继续学习, 之前将学习记录都记在了有道云笔记上, 自己掉过的坑不想让别人再掉一遍。

先上题



哇塞美女!

图片链接地址: https://pan.baidu.com/s/1IXMU_xNJDmFoLVu-hs5XjA 密码: 6otw

1、上来使用通用方法。先用stegsolve查看一下, 看看其他的LSB之类的并没有发现什么问题, 然后使用checkpng 检查图片结构是否存在问题, 发现有一些异常的IDAT块。

```

E:\2018渗透测试工具\图片隐写工具集\pngcheck-2.3.0>E:\2018渗透测试工具\图片隐写工具集\pngcheck-2.3.0\pngcheck.exe -v D:\图片隐写题库\隐写本图片\png\sctf-nisc400\sctf.png
File: D:\图片隐写题库\隐写本图片\png\sctf-nisc400\sctf.png (1421461 bytes)
chunk IHDR at offset 0x0000c, length 13
  1000 x 562 image, 32-bit RGB+alpha, non-interlaced
chunk sRGB at offset 0x00025, length 1
  rendering intent = perceptual
chunk gAMA at offset 0x00032, length 4: 0.45455
chunk pHYs at offset 0x00042, length 9: 3780x3780 pixels/meter (96 dpi)
chunk IDAT at offset 0x00057, length 65524
  zlib: deflated, 32K window, fast compression
chunk IDAT at offset 0x10008, length 65524
chunk IDAT at offset 0x20008, length 65524
chunk IDAT at offset 0x30008, length 65524
chunk IDAT at offset 0x40008, length 65524
chunk IDAT at offset 0x50008, length 65524
chunk IDAT at offset 0x60008, length 65524
chunk IDAT at offset 0x70008, length 65524
chunk IDAT at offset 0x80008, length 65524
chunk IDAT at offset 0x90008, length 65524
chunk IDAT at offset 0xa0008, length 65524
chunk IDAT at offset 0xb0008, length 65524
chunk IDAT at offset 0xc0008, length 65524
chunk IDAT at offset 0xd0008, length 65524
chunk IDAT at offset 0xe0008, length 65524
chunk IDAT at offset 0xf0008, length 65524
chunk IDAT at offset 0x100008, length 65524
chunk IDAT at offset 0x110008, length 65524
chunk IDAT at offset 0x120008, length 65524
chunk IDAT at offset 0x130008, length 65524
chunk IDAT at offset 0x140008, length 65524
chunk IDAT at offset 0x150008, length 45027
chunk IDAT at offset 0x15aff7, length 138
chunk IEND at offset 0x15b08d, length 0
No errors detected in D:\图片隐写题库\隐写本图片\png\sctf-nisc400\sctf.png (28 chunks, 36.8% compression).

```

可疑，65524一个IDAT，最后未满载就提前分离

可以看到，正常的块的length是在65524的时候就满了，而倒数第二个IDAT块长度是45027，最后一个长度是138，很明显最后一个IDAT块是有问题的，因为他本来应该并入到倒数第二个未满载的块里。

2、查看异常数据块的情况，使用010editor/winhex打开，导出异常数据块

15:AF90h:	D9 73 B2 10 67 F8 09 E8 13 E5 D9 62 3E 73 9E BF	Ús°.gø.è.âÜb>sž¿
15:AFA0h:	9C 31 3E CD 73 9A 13 D0 1F 0B A0 F3 B4 68 12 A0	œ1>Ísš.Đ.. ó'h.
15:AFB0h:	4F 97 E6 D1 36 CF C6 74 7E 16 A6 B2 E8 F3 96 9A	O-æÑ6İEt~.!'èó-š
15:AFD0h:	20 A0 2E 05 FA 44 C3 FF 2E D0 69 5B 0A A0 97 FF	..úDÄy.Đi[. -ÿ
15:AFD0h:	17 40 AF C3 48 6B A5 00 3A 4F ED 05 FA 54 63	.@_ÄHk¥.:Ois,úT
15:AFE0h:	95 00 FA 54 0D 21 BD BA 02 FF 31 01 E7 98 5E 68	..úT.!'ç-ÿÿ'ç'^h
15:AFF0h:	95 8F CD 00 00 00 8A 49 44 41 54 78 9C 5D 91 01	..í...ŠIDAT[œ]'. [49 44 41 54]
15:B000h:	12 80 40 08 02 BF 04 FF FF 5C 75 29 4B 55 37 73	.€@..¿.ÿÿ(u)KU7s
15:B010h:	8A 21 A2 7D 1E 49 CF D1 7D B3 93 7A 92 E7 E6 03	Š!c}.IİÑ)'z'çæ.
15:B020h:	88 0A 6D 48 51 00 90 1F B0 41 01 53 35 0D E8 31	^.mHQ...°A.SS.è1
15:B030h:	12 EA 2D 51 C5 4C E2 E5 85 B1 5A 2F C7 8E 88 72	.è-QÀLáâ...iZ/Ç'Z'r
15:B040h:	F5 1C 6F C1 88 18 82 F9 3D 37 2D EF 78 E6 65 B0	ö.oÁ^.,ù=7-ixæ°
15:B050h:	C3 6C 52 96 22 A0 A4 55 88 13 88 33 A1 70 A2 07	ÄlR-" sU^.'3;pc.
15:B060h:	1D DC D1 82 19 DB 8C 0D 46 5D 8B 69 89 71 96 45	.ÜÑ,.Ü€Fj<i%q-E
15:B070h:	ED 9C 11 C3 (6A) E3 AB DA EF CF C0 AC F0 23 E7 7C	ie.Äjã«ÜiÿÄ-š#cl
15:B080h:	17 C7 89 76 67 D9 CF A5 A8 00 00 00 00 49 45 4E	.Ç%vgÜiÿ[....IEN [D9 CF A5 A8]
15:B090h:	44 AE 42 60 82	DøB', [44 AE 42 60 82]

CRC32校验

Find ASCII: IDAT 49 44 41 54

Address	Value
C0008h	IDAT
D0008h	IDAT
E0008h	IDAT
F0008h	IDAT
100008h	IDAT
110008h	IDAT

3、查找78 9C文件头标志，发现是zlib压缩。可以查找用python zlib解压方法

000000C0~000000F8 :

以上选中部分是IDAT数据块

- 00 00 00 27 数据长为39字节
- 49 44 41 54 IDAT标识
- 78 9C..... 压缩的数据, LZ77派生压缩方法
- DA 12 06 A5 CRC校验

IDAT中压缩数据部分在后面会有详细的介绍。

000000F9~00000104 :

IEND数据块, 这部分正如上所说, 通常都应该是

00 00 00 00 49 45 4E 44 AE 42 60 82

4、网上搜索zlib的python解压方法, 将异常的IDAT数据块斩头去尾之后使用脚本解压

```

2 import zlib
3 import binascii
4 IDAT =("789C5D91011280400802BF04FFFF5"+
5 "C75294B5537738A21A27D1E49CFD17DB39"+
6 "37A92E7E603880A6D485100901FB041015"+
7 "3350DE83112EA2D51C54CE2E585B15A2FC"+
8 "78E8872F51C6FC1881882F93D372DEF78E"+
9 "665B0C36C529622A0A45588138833A170A"+
10 "2071DDCD18219DB8C0D465D8B698971964"+
11 "5ED9C11C36AE3ABDAEFCFC0ACF023E77C1"+
12 "7C7897667").decode("hex")
13 #print IDAT
14 result = binascii.hexlify(zlib.decompress(IDAT))
15 print result

```

<https://blog.csdn.net/u010391191>

```

D:\图片隐写题库\0x003 图片隐写之pngcheck+隐藏二维码>python zlib.py
313131313131313130303031303030303131303131313131313131313131313131303030303031303131313030313031313031
3130313131303130313131303131303130303130313131303131303030303031303130313031313031313031313031
313130313030313130303030313031303031313130313130313130313131303130313031303130303130313030303131
3131303030303131313031313131303030313130303130313030303131303031313130303030313031303130313031
313130303131313030313030303031303131313131313031303030303030303131303130313030313030313030313030
3130313131303130303031313031303031313131313030303031303131313031303131303030313131303130313131303130
313131313131313031313031303131303131313031313031313130313130313131303131313031313130313131303131

```

<https://blog.csdn.net/u010391191>

发现是一堆的31303130, 此时赛棍立马就知道怎么回事, 初学者看了writeup复现的时候才知道, 如下:

00h:	78 9C 5D 91 01 12 80 40 08 02 BF 04 FF FF 5C 75	xœ] \. .€@. .¿. ýÿ\u
10h:	29 4B 55 37 73 8A 21 A2 7D 1E 49 CF D1 7D B3 93)KU7sŠ!◊}. IİÑ}:'"
20h:	7A 92 E7 E6 03 88 0A 6D 48 51 00 90 1F B0 41 01	z' çæ. ^. mHQ. . . °A.
30h:	53 35 0D E8 31 12 EA 2D 51 C5 4C E2 E5 85 B1 5A	S5. è1. ê-QÀLââ...±Z
40h:	2F C7 8E 88 72 F5 1C 6F C1 88 18 82 F9 3D 37 2D	/ÇŽ^ rð. oĂ^ ., ù=7-
50h:	EF 78 E6 65 B0 C3 6C 52 96 22 A0 A4 55 88 13 88	ixæe°ĂlR-" «U^ .^
60h:	33 A1 70 A2 07 1D DC D1 82 19 DB 8C 0D 46 5D 8B	3;pe. .ÜÑ, .Ûœ.F <
70h:	69 89 71 96 45 ED 9C 11 C3 6A E3 AB DA EF CF C0	i%q-Eiœ. Ăjă«ÚiİĂ
80h:	7C F0 23 E7 7C 17 C7 89 76 67 31 30 31 30 31 30	-ð#ç . Ç%vg101010
90h:	31 30 31 30 31 30 31 30 01 01 01 02 0b	10101010.039. 191

5、使用Hex to ASCII在线转换工具

<https://www.asciitohex.com/>

将所有Hex转成ASCII

6、转成ASCII后全选发现是625个binary，使用如下脚本画出二维码（第一次做此类题的时候，手动分割NXN的binary，然后粘贴到Excel中，0画黑1不画=，如果没出来，再1画黑...）

```
1 import PIL.Image
2 MAX = 25
3 pic = PIL.Image.new("RGB", (MAX, MAX))
4 str = "平方个数的binary"
5 i=0
6 for y in range (0,MAX):
7     for x in range (0,MAX):
8         if(str[i] == '1'):
9             pic.putpixel([x,y],(0, 0, 0))
10        else:
11            pic.putpixel([x,y],(255,255,255))
12        i = i+1
13
14 pic.show()
15 pic.save("flag.png") https://blog.csdn.net/u010391191
```

Tips: 某次比赛，屏蔽各种信号，画出二维码后没有本地二维码解析器，无法解析出flag。百度搜索准备一个本地二维码扫描器