

PicoCTF 2018 Writeup: Forensics

Oct 13, 2018 08:56 · 1346 words · 7 minute read

CTF CYBER-SECURITY WRITE-UP PICOCTF FORENSICS

Forensics Warmup 1

Problem

Solution

Forensics Warmup 2

Problem

Solution

Desrouleaux

Problem

Solution

Reading Between the Eyes

Problem

Solution

Recovering From the Snap

Problem

Solution

admin panel

Problem

Solution

hex editor

Problem

```
Solution
Truly an Artist
  Problem
  Solution
now you don't
  Problem
  Solution
Ext Super Magic
  Problem
  Solution
Lying Out
  Problem
  Solution
What's My Name?
  Problem
  Solution
core
  Problem
  Solution
Malware Shops
  Problem
  Solution
LoadSomeBits
  Problem
  Solution
```

Forensics Warmup 1

Problem

Can you unzip this [file](#) for me and retrieve the flag?

Solution

Just unzip the file.

flag: `picoCTF{welcome_to_forensics}`

Forensics Warmup 2

Problem

Hmm for some reason I can't open this [PNG](#)? Any ideas?

Solution

Using the `file` command, you can see that the image is, in fact, in `jpeg` format not `png`:

```
> file flag.png
flag.png: JPEG image data, JFIF standard 1.01
```

Open the image as a `jpeg` file to get the file.

flag: `picoCTF{extensions_are_a_lie}`

Desrouleaux

Problem

Our network administrator is having some trouble handling the tickets for all of our incidents. Can you help him out by answering all the questions? Connect with `nc`

`2018shell12.picocftf.com 10493`. [incidents.json](#)

Solution

Here is the solution script:

```
from sets import Set
from pwn import *
import json

sh = remote('2018shell12.picocftf.com', 10493)

with open('./incidents.json') as f:
    data = json.loads(f.read())

# question 1
src = {}

for each in data[u'tickets']:
    src_ip = each[u'src_ip']
    if src_ip in src:
        src[src_ip] += 1
    else:
        src[src_ip] = 1

print sh.recvuntil('ones.\n')
sh.sendline(max(src, key=src.get))

# question 2
```

```
target = sh.recvuntil('?\n').split(' ')[-1][:-2]
target_ls = {}
count = 0
for each in data[u'tickets']:
    if each[u'src_ip'] == target and each[u'dst_ip'] not in target_ls:
        target_ls[each[u'dst_ip']] = True
        count += 1

sh.sendline(str(count))

# question 3
hashes = {}
for each in data[u'tickets']:
    hash = each[u'file_hash']
    if hash not in hashes:
        hashes[hash] = Set()
    hashes[hash].add(each[u'dst_ip'])

avg = 0
for each in hashes:
    e = hashes[each]
    avg += len(e)
avg = (avg * 1.0) / len(hashes)

print sh.recvuntil('.\n')
sh.sendline(str(avg))

sh.interactive()
```

flag: `picoCTF{J4y_s0n_d3rUUUULo_a062e5f8}`

Reading Between the Eyes

Problem

Stego-Saurus hid a message for you in this [image](#), can you retrieve it?

Solution

This problem is about using the [Least Significant Bit algorithm for image steganography](#). It can be solved using an [online decoder](#).

Steganography Online

[Encode](#)

[Decode](#)

Decode image

To decode a hidden message from an image, just choose an image and hit the **Decode** button.

Neither the image nor the message that has been hidden will be at any moment transmitted over the web, all the magic happens within your browser.

[Choose File](#) husky.png

[Decode](#)

Hidden message

picoCTF{r34d1ng_b37w33n_7h3_by73s}

flag: `picoCTF{r34d1ng_b37w33n_7h3_by73s}`

Recovering From the Snap

Problem

There used to be a bunch of [animals](#) here, what did Dr. Xernon do to them?

Solution

This problem is about recovering files from a FAT filesystem. It can be done using [TestDisk](#), a powerful free data recovery software.

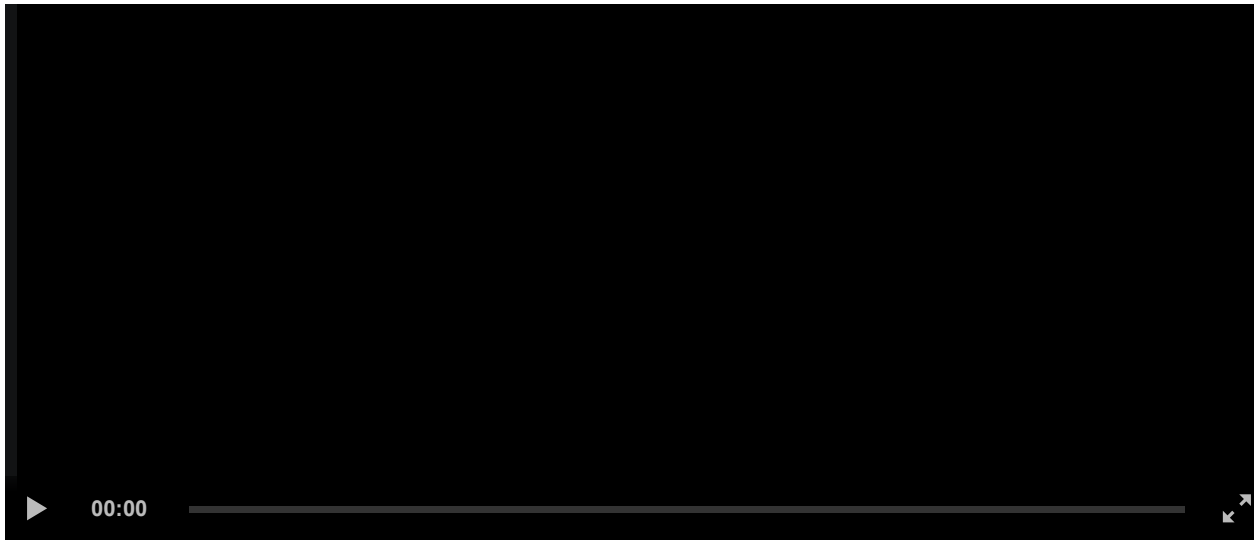
You can follow [this guide](#) to recover the `theflag.jpg` file.

```
TestDisk 7.0, Data Recovery Utility, April 2015
Christophe GRENIER <grenier@cgsecurity.org>
http://www.cgsecurity.org

Disk animals.dd - 10 MB / 10 MiB
  CHS 10 64 32 - sector size=512

[ Analyse ] Analyse current partition structure and search for lost partitions
>[ Advanced ] Filesystem Utils
[ Geometry ] Change disk geometry
[ Options ] Modify options
[ Quit ] Return to disk selection
```

Note: Correct disk geometry is required for a successful recovery. 'Analyse' process may give some warnings if it detects that the physical geometry is mismatched.



Recorded with [asciinema](#)

picoCTF{th3_5n4p_happ3n3d}

theflag.jpg

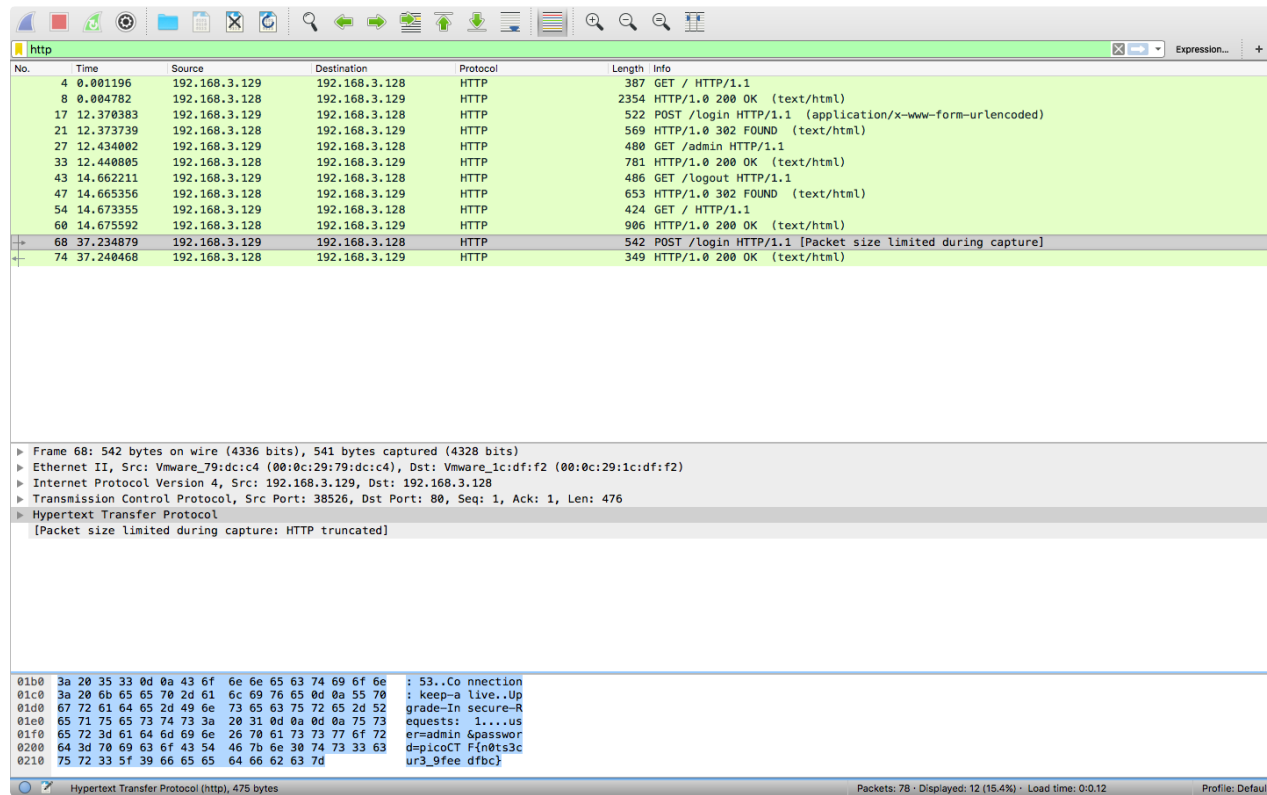
flag: `picoCTF{th3_5n4p_happ3n3d}`

admin panel

Problem

We captured some [traffic](#) logging into the admin panel, can you find the password?

Solution



No.	Time	Source	Destination	Protocol	Length	Info
4	0.001196	192.168.3.129	192.168.3.128	HTTP	387	GET / HTTP/1.1
8	0.004782	192.168.3.128	192.168.3.129	HTTP	2354	HTTP/1.0 200 OK (text/html)
17	12.370383	192.168.3.129	192.168.3.128	HTTP	522	POST /login HTTP/1.1 (application/x-www-form-urlencoded)
21	12.373739	192.168.3.128	192.168.3.129	HTTP	569	HTTP/1.0 302 FOUND (text/html)
27	12.434002	192.168.3.129	192.168.3.128	HTTP	480	GET /admin HTTP/1.1
33	12.440805	192.168.3.128	192.168.3.129	HTTP	781	HTTP/1.0 200 OK (text/html)
43	14.662211	192.168.3.129	192.168.3.128	HTTP	486	GET /logout HTTP/1.1
47	14.665356	192.168.3.128	192.168.3.129	HTTP	653	HTTP/1.0 302 FOUND (text/html)
54	14.673355	192.168.3.129	192.168.3.128	HTTP	424	GET / HTTP/1.1
60	14.675592	192.168.3.128	192.168.3.129	HTTP	906	HTTP/1.0 200 OK (text/html)
68	37.234879	192.168.3.129	192.168.3.128	HTTP	542	POST /login HTTP/1.1 [Packet size limited during capture]
74	37.240468	192.168.3.128	192.168.3.129	HTTP	349	HTTP/1.0 200 OK (text/html)

Frame 68: 542 bytes on wire (4336 bits), 541 bytes captured (4328 bits)

Ethernet II, Src: Vmware_79:dc:c4 (00:0c:29:79:dc:c4), Dst: Vmware_1c:df:f2 (00:0c:29:1c:df:f2)

Internet Protocol Version 4, Src: 192.168.3.129, Dst: 192.168.3.128

Transmission Control Protocol, Src Port: 38526, Dst Port: 80, Seq: 1, Ack: 476

Hypertext Transfer Protocol

[Packet size limited during capture: HTTP truncated]

01b0 3a 20 35 33 0d 0a 43 6f 6e 6e 65 63 74 69 6f 6e : 53..Co nnection
01c0 3a 20 6b 65 65 70 2d 61 6c 69 76 65 0d 0a 55 70 : keep-a live..Up
01d0 67 72 61 64 65 2d 49 6e 73 65 63 75 72 65 2d 52 : grade-In secure-R
01e0 65 71 75 65 73 74 73 3a 20 31 0d 0a 0d 0a 75 73 : equests: 1....us
01f0 65 72 3d 61 64 6d 69 6e 26 70 61 73 73 77 6f 72 : er=admin &passwor
0200 64 3d 70 69 63 6f 43 54 46 7b 6e 30 74 73 33 63 : dpicotCT Fln0ts3c
0210 75 72 33 5f 39 66 65 65 64 66 62 63 7d : ur3_9fee dfbc}

If you look for `http` requests, you will see two login attempts, and the second request contains the flag:

```
POST /login HTTP/1.1
Host: 192.168.3.128
User-Agent: Mozilla/5.0 (X11; Ubuntu; Linux x86_64; rv:59.0) Gecko/20100101 Firefox/59.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Referer: http://192.168.3.128/
Content-Type: application/x-www-form-urlencoded
Content-Length: 53
Connection: keep-alive
Upgrade-Insecure-Requests: 1

user=admin&password=picoCTF{n0ts3cur3_9feedfbc}
```

flag: `picoCTF{n0ts3cur3_9feedfbc}`

hex editor

Problem

This `cat` has a secret to teach you. You can also find the file in `/problems/hex-editor_4_0a7282b29fa47d68c3e2917a5a0d726b` on the shell server.

Solution

You can get the flag by looking at the hex hump of the image or just print out all the readable parts of the file:

```
> strings hex_editor.jpg | grep pico  
Your flag is: "picoCTF{and_thats_how_u_edit_hex_kittos_dF817ec5}"
```

flag: `picoCTF{and_thats_how_u_edit_hex_kittos_dF817ec5}`

Truly an Artist

Problem

Can you help us find the flag in this [Meta-Material](#)? You can also find the file in `/problems/truly-an-artist_3_066d6319e350c1d579e5cf32e326ba02`.

Solution

The flag is in the EXIF meta-data of the image:

```
> exiftool 2018.png  
ExifTool Version Number      : 11.01  
File Name                    : 2018.png  
Directory                    : .  
File Size                    : 13 kB  
File Modification Date/Time   : 2018:10:09 23:34:05+08:00  
File Access Date/Time        : 2018:10:10 09:15:07+08:00  
File Inode Change Date/Time   : 2018:10:09 23:34:06+08:00  
File Permissions              : rw-r--r--  
File Type                    : PNG  
File Type Extension          : png  
MIME Type                    : image/png
```

```
Image Width      : 1200
Image Height     : 630
Bit Depth        : 8
Color Type       : RGB
Compression      : Deflate/Inflate
Filter           : Adaptive
Interlace        : Noninterlaced
Artist           : picoCTF{look_in_image_eeee129e}
Image Size       : 1200x630
Megapixels       : 0.756
```

flag: `picoCTF{look_in_image_eeee129e}`

now you don't

Problem

We heard that there is something hidden in this [picture](#). Can you find it?

Solution

You can create another image with only one shade of red and diff that image with the one provided to get the flag:

```
> convert -size 857x703 canvas:"#912020" pure.png
> compare nowYouDont.png pure.png diff.png
```

picoCTF{n0w_y0u_533_m3}

diff.png

flag: `picoCTF{n0w_y0u_533_m3}`

Ext Super Magic

Problem

We salvaged a ruined Ext SuperMagic II-class mech recently and pulled the [filesystem](#) out of the black box. It looks a bit corrupted, but maybe there's something interesting in there. You can also find it in `/problems/ext-super-magic_4_f196e59a80c3fdac37cc2f331692ef13` on the shell server.

Solution

You are given a ext3 file image that is broken. To fix the image, you have to correct the magic number of the file. You can read more about the ext3 file format over [here](#).

Here is the script that writes the magic number `0xEF53` into the file:

```
# flag: picoCTF{a7DB29eCf7dB9960f0A19Fdde9d00Af0}nc 2018shell2.picoctf.com 2651

from pwn import *

with open('./ext-super-magic.img', 'rb') as f:
    data = f.read()

print enhex(data[1024:1024+82])
print enhex(data[1024+56:1024+56+2])

data = data[:1024+56] + p16(0xEF53) + data[1024+56+2:]

with open('fixed.img', 'wb') as f:
    f.write(data)
```

flag: `picoCTF{a7DB29eCf7dB9960f0A19Fdde9d00Af0}`

Lying Out

Problem

Some odd [traffic](#) has been detected on the network, can you identify it? More [info](#) here.

Connect with `nc 2018shell12.picoctf.com 27108` to help us answer some questions.

Solution

Just read the graph and do this problem by hand.

flag: `picoCTF{w4y_0ut_de051415}`

What's My Name?

Problem

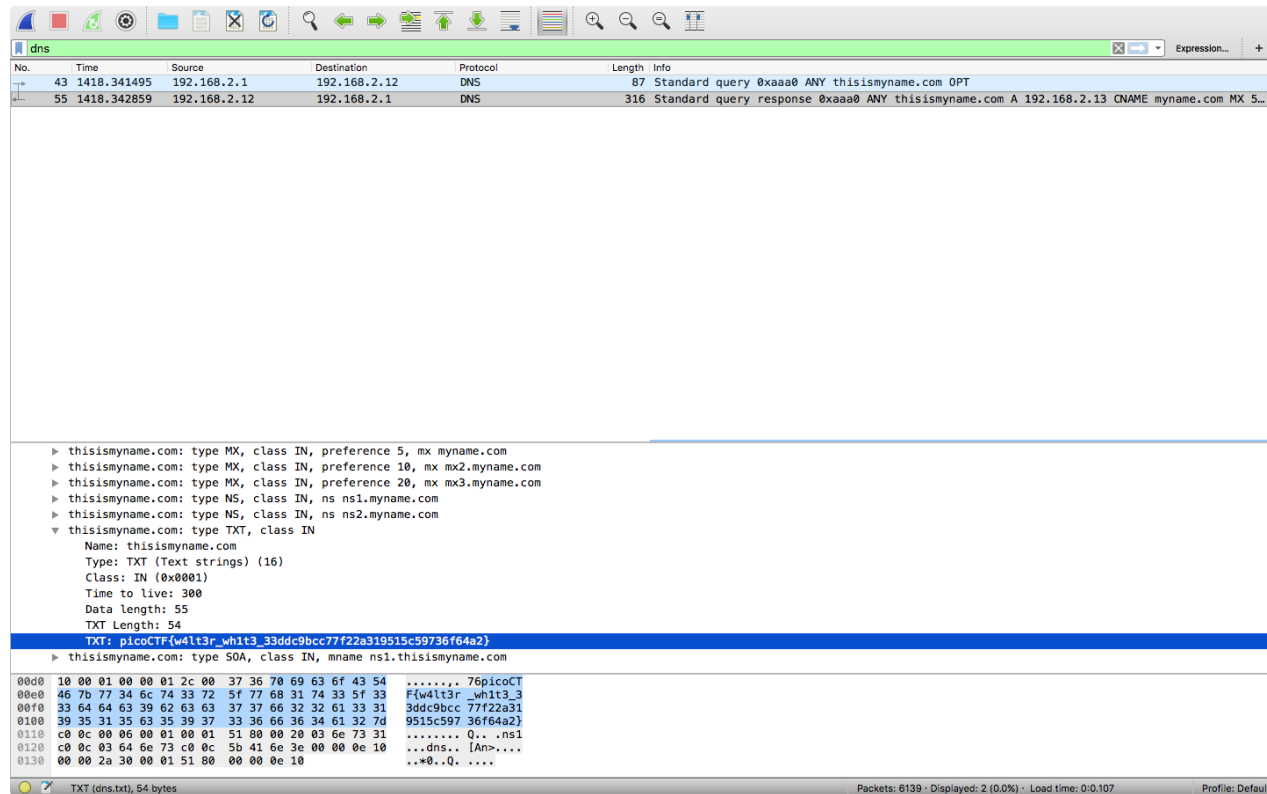
Say my name, say [my name](#).

Solution

The hint is very helpful. It asks `If you visited a website at an IP address, how does it know the name of the domain?`.

The answer to this question is that a domain is resolved through `DNS` packets.

If we only look for `DNS` packets in Wireshark, we will find the flag.



flag: `picoCTF{w4lt3r_wh1t3_33ddc9bcc77f22a319515c59736f64a2}`

core

Problem

This [program](#) was about to print the flag when it died. Maybe the flag is still in this [core](#) file that it dumped? Also available at [/problems/core_1_722685357ac5a814524ee76a3dcd1521](#) on the shell server.

Solution

Let's first take a look at the program using radare2:

```
[0x080484c0]> s sym.print_flag
[0x080487c1]> pdf
r (fcn) sym.print_flag 43
|   sym.print_flag ();
|       ; var int local_ch @ ebp-0xc
|       ; CALL XREF from sym.main (0x8048802)
|
|   0x080487c1      55          push ebp                ; ./print_flag.c:90
|   0x080487c2      89e5        ebp = esp
|   0x080487c4      83ec18      esp -= 0x18
|   0x080487c7      c745f4390500. dword [local_ch] = 0x539 ; ./print_flag.c:91 ;
|   0x080487ce      8b45f4      eax = dword [local_ch] ; ./print_flag.c:92
|   0x080487d1      8b048580a004. eax = dword [eax*4 + obj.strs] ; [0x804a080:4]=0
|   0x080487d8      83ec08      esp -= 8
|   0x080487db      50          push eax
|   0x080487dc      684c890408  push str.your_flag_is:picoCTF__s ; 0x804894c ; "y
|   0x080487e1      e82afcffff  sym.imp.printf () ; int printf(const cha
|   0x080487e6      83c410      esp += 0x10
|   0x080487e9      90          ; ./print_flag.c:93
```

	0x080487ea	c9	leave
L	0x080487eb	c3	return

As you can see, the flag pointer is located at `eax*4 + obj.strs` or `0x804a080+0x539*4` in memory:

```
> python
>>> hex(0x804a080+0x539*4)
'0x804b564'
```

Now, we can use gdb and the core file to restore the application state and extract the flag from that address:

```
$ gdb ./print_flag ./core
...
gef> x 0x804b564
0x804b564 <strs+5348>: 0x080610f0
gef> x 0x080610f0
0x080610f0: "e52f4714963eb207ae54fd424ce3c7d4"
```

flag: `picoCTF{e52f4714963eb207ae54fd424ce3c7d4}`

Malware Shops

Problem

There has been some [malware](#) detected, can you help with the analysis? More [info](#) here.

Connect with `nc 2018shell2.picoctf.com 46168`.

Solution

Just read the graph and do this problem by hand.

```
> nc 2018shell2.picoctf.com 46168
```

You'll need to consult the file `clusters.png` to answer the following questions.

How many attackers created the malware `in` this dataset?

5

Correct!

In the following sample of files from the larger dataset, which file was made by the same att

	hash	jmp_count	add_count
0	3ce8eb6f	33.0	28.0
1	55489271	40.0	2.0
2	33d91680	39.0	29.0
3	ebaf5ccd	9.0	17.0
4	e9c0ac07	17.0	61.0
5	628e79cf	9.0	18.0
6	b3ae7861	41.0	10.0
7	cc251d4b	16.0	41.0
8	0c91a83b	17.0	65.0
9	97a0fc46	10.0	38.0

33d91680

Correct!

Great job. You've earned the flag: picoCTF{w4y_0ut_dea1794b}

flag: `picoCTF{w4y_0ut_dea1794b}`

LoadSomeBits

Problem

Can you find the flag encoded inside this image? You can also find the file in `/problems/loadsomebits_2_c5bba4da53a839fcdda89e5203ac44d0` on the shell server.

Solution

Ryan Jung on our team solved this challenge. It is about looking at the least significant bit of each pixel value.

flag: `picoCTF{st0r3d_iN_th3_l345t_slgn1flc4nT_b1t5_2705826400}`

Feel free to leave a comment if any of the challenges is not well explained.

 [tweet](#)  [Share](#)

1 Comment [github-blog](#)

 Login ▾

 Recommend 1

 [Tweet](#)

 [Share](#)

Sort by Best ▾



Join the discussion...

LOG IN WITH

OR SIGN UP WITH DISQUS [?](#)



Name



[ctflower](#) • a year ago

hi,

i did not understand how you extracted lsb in LoadSomeBits challenge.
can you plz explain it.

^ | v • Reply • Share ›

ALSO ON GITHUB-BLOG

[PicoCTF 2018 Writeup: Reversing](#)

7 comments • a year ago



[Ye Thu](#) — thanks

[Avatar](#)

[35c3ctf 2018 Writeup · Alan's Blog](#)

1 comment • 10 months ago



[teamrocketist](#) — Looks like I "cheated" on the juggle challenge :O, the random was weird and allowed me guess the next

[Avatar](#)

[PicoCTF 2018 Writeup: Binary Exploitation](#)

8 comments • a year ago



[Roe Sefi](#) — U welcome

[Avatar](#)

[PicoCTF 2019 Writeup: Binary Exploitation](#)

5 comments • 8 days ago



[Nev River](#) — Hello Alan, thank you very much for your write-ups, they had helped me a lot. For the NewOverflow-1 problem,

[Avatar](#)

[Subscribe](#)

[Add Disqus to your site](#)

[Disqus' Privacy Policy](#)

DISQUS



© Copyright 2019 ♥ Alan Chang

Powered by [Hugo](#) Theme By [nodejh](#)