Files
master
Go to file t
_includes
cscit5 (2)
crypto
crypto.md
forensics
img
resources
flag.txt
morse.wav
panda.jpg
unseen.txt

CTF-writeups / cscit5 (2) / forensics / focused

noob-attach gh pages add
b97d9ef 4 years ago History

Preview Code Blame 351 lines (262 loc) · 7.7 KB
Raw

layout :
default :

FORENSIC

GRADIENT SKY

The above image was given following the basic commands I got this by binwalk

```
root@kali:~/CTF/cscit5/forensics# binwalk sky.jpg

DECIMAL       HEXADECIMAL     DESCRIPTION
--------------------------------------------------------------------------------
0             0x0             JPEG image data, JFIF standard 1.01
38            0x26            TIFF image data, little-endian offset of first image directory: 8
38            0x26            JPEG image data, JFIF standard 1.01
225858        0x3A202         ARJ archive data, version 5.4

root@kali:~/CTF/cscit5/forensics#
```

As results show it has some RAR content on unraring the content I got the flag

cscit{jOker_w4ont_happy}

ARCHED

As starting with the classical command to check the file formate and it was a .jpg file

```
root@kali:~/Desktop# file arched.jpg
arched.jpg: JPEG image data, JFIF standard 1.01, resolution (DPI), density 300x300, segment length 16, baseline, precision 8, 2
root@kali:~/Desktop#
```

After renaming it .jpg I ran some tools and steghide worked perfectly and I get a flag.zip file.

```
root@kali:~/Desktop# steghide extract -sf arched.jpg
Enter passphrase:
wrote extracted data to "flag.zip".
root@kali:~/Desktop#
```

Since it was password protected I use fcrack and everyone's fav rockyou.txt to crack it.

```
root@kali:~/Desktop# fcrackzip -u -D -p rockyou.txt flag.zip

PASSWORD FOUND!!!!: pw == kathmandu
root@kali:~/Desktop#
```

After unlocking we got a image which have the flag.

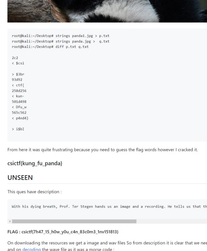
PANDA

In this question we were given a password protected zip file so by using fcrackzip tels crack it.

```
root@kali:~/Desktop#  fcrackzip -u -D -p rockyou.txt panda.zip

PASSWORD FOUND!!!!: pw == 2011
```

On extracting the zip file we get two panda images at first I tried a lot of tools but it much easier the flag was in the difference of the strings of the two images so



```
root@kali:~/Desktop# strings panda1.jpg > a.txt
root@kali:~/Desktop# strings panda.jpg > a.txt
root@kali:~/Desktop# diff a.txt a.txt

0a1
> $!w:
> ~Ldl
> ~rtf[
> jfb8Z5X
> kun:
> 5KLWkW
> 5Ldl
> 5Ch/bX2
> p4ndA]

> 1Ob1
```

From here it was quite frustrating because you need to guess the flag words however I cracked it.

cscit{kung_fu_panda}

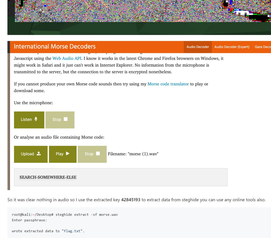
UNSEEN

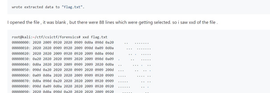
This ques have description :

```
With his dying breath, Prof. Tor Stegen hands us an image and a recording. He tells us that the image is least significant, but
```

FLAG : cscit{57b47_15_h0w_y0u_c4n_B3c0m3_1nv1S1B1l1}

On downloading the resources we get a image and wav files So from description it is clear that we need to do so using openpuzzl ! I got the key and we decrypted the wave file as it was a morse code :





So it was clear nothing in audio so I use the extracted key **42845193** to extract data from steghide you can use any online tools also.

```
root@kali:~/Desktop# steghide extract -sf morse.wav
Enter passphrase:

wrote extracted data to "flag.txt".
```

I opened the file , it was blank , but there were 88 lines which were getting selected, so I saw ctrl of the file .

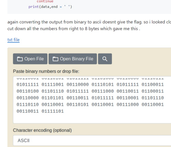
```
root@kali:~/CTF/cscit5/forensics# wc -l flag.txt
0000000: 2020 2020 2020 2020 0009 0004 0904 0a20  ........ .......
0000010: 2020 2020 0909 0920 2009 0904 0a04 0a0a  ... ......... ....
0000020: 0a20 2020 2020 2020 0009 0004 0a04 0a20  . ........ .......
0000030: 0a20 2020 2020 2020 2009 0904 0a04 0a0a  . ........ ....
0000040: 0a04 2020 2009 0909 0009 2009 2020 0a0a  .. ... .. .. ..
0000050: 0904 0a20 2020 2020 0009 2020 2020 2020  ... ..... ......
0000060: 0a04 0a0a 2020 2020 2009 0904 0920 0909  ..... ... ... ...
0000070: 0a04 0a20 2020 2020 2009 0904 0a04 0a20  ..... ......... .
0000080: 0904 0a04 0a0a 2020 2009 0920 0909 0a20  ..... ... ... ...
0000090: 2020 0a04 0a20 2020 2020 2020 0009 0920  ..... ... ... ...
00000a0: 0a04 0a0a 2020 2020 2009 0904 0a04 0a20  ..... ......... .
00000b0: 2009 0909 0a04 0a20 2020 2020 2020 2020  ..... ..... .....
00000c0: 0909 0909 0904 0a04 0a20 2020 2020 2020  ..... ..... .....
00000d0: 0909 2020 2009 0a04 0904 0a20 2020 2020  ..... ..... .....
00000e0: 2009 0920 0920 0904 0a04 0a20 2020 2020  ..... ..... .....
00000f0: 2009 0909 0a04 0a20 2020 2020 2020 2020  ..... ..... .....
0000100: 2020 2020 2009 0a04 0a20 2020 2020 2020  ..... ..... .....
0000110: 2020 2020 2020 2009 0904 0a04 0a20 2020  ..... ..... .....
0000120: 2020 2020 2020 2020 0909 0904 0a04 0a20  ..... ..... .....
0000130: 2020 2020 2009 0909 0909 0a04 0a20 2020  ..... ..... .....
0000140: 2020 2020 2020 2020 2009 0904 0a04 0a20  ..... ..... .....
0000150: 0904 0a20 2020 2020 2009 0920 2009 2020  ..... ... ... ...
0000160: 0a04 0a20 2020 2020 2009 0904 0a04 0a20  ..... ..... .....
0000170: 0904 0a04 0a20 2020 2020 2020 2020 2020  ..... ..... .....
0000180: 0909 0920 0a04 0a20 2020 2020 2020 2020  ..... ..... .....
0000190: 2009 0909 0904 0a04 0a20 2020 2020 2020  ..... ..... .....
00001a0: 2009 0909 0920 0a04 0a20 2020 2020 2020  ..... ..... .....
00001b0: 2009 0909 0920 2009 0904 0a04 0a20 2020  ..... ..... .....
00001c0: 2009 0909 0920 2009 0904 0a04 0a20 2020  ..... ..... .....
00001d0: 2009 0909 0920 2009 0904 0a04 0a20 2020  ..... ..... .....
00001e0: 2009 0909 0920 0a04 0a20 2020 2020 2020  ..... ..... .....
00001f0: 2009 0909 0904 0a04 0a20 2020 2020 2020  ..... ..... .....
0000200: 2009 0909 0904 0a04 0a20 2020 2020 2020  ..... ..... .....
0000210: 2009 0909 0904 0a04 0a20 2020 2020 2020  ..... ..... .....
0000220: 2009 0909 0904 0a04 0a20 2020 2020 2020  ..... ..... .....
0000230: 2009 0909 0904 0a04 0a20 2020 2020 2020  ..... ..... .....
0000240: 2009 0909 0904 0a04 0a20 2020 2020 2020  ..... ..... .....
0000250: 2009 0909 0904 0a04 0a20 2020 2020 2020  ..... ..... .....
0000260: 2009 0909 0904 0a04 0a20 2020 2020 2020  ..... ..... .....
0000270: 2009 0909 0904 0a04 0a20 2020 2020 2020  ..... ..... .....
0000280: 2009 0909 0904 0a04 0a20 2020 2020 2020  ..... ..... .....
0000290: 2009 0909 0904 0a04 0a20 20              ..... ..... ..
root@kali:~/CTF/cscit5/forensics#
```

with some research I found that it a type of data encoding and can be solved by replacing some two value with 1 and rest with 0 , which will give a binary and hence flag.txt wrote a python file which will convert 't' or 0x09 to "1" and " " or 0x20 to "0" and removed remaining others .

```
#!/usr/bin/env python3

from pwn import *

with open("flag.txt", "rb") as bin_file:
    for _ in range(88):
        data = bytearray(bin_file.readline())
        data = data.replace(b'\t', b'1')
        data = data.replace(b'\xd8', b'0')
        data = data.replace(b'\x0a', b'')
        data = data.replace(b'\xd0', b'')
        data = data.decode("ascii")
        if data[0:2] == b'\x00':
            continue
        print(data,end = '')
```

again converting the output from binary to ascii doesnt give the flag, so i looked closely and saw that so many numbers werent of 8 bytes . so i cut down all the numbers from right to 8 bytes which gave me this .

txt file