

WRITEUP FIND-IT! CTF 2022

Disusun oleh: pwnpeko, Universitas Diponegoro

1. Kategori: Crypto

Nama: EzRSA

Langkah pengeraan:

- a. Pada soal telah disediakan modulus, exponent, dan ciphertext dari sebuah RSA sebagai berikut:

c=48194219261855563203215132311565813649624212082168963448

568445899090

e=65537

n=63470050348776615811503850961942229596941767038091305819

0145906857689

- b. Pertama kali lihat adalah nilai n yang sangat kecil, sehingga mudah difaktorkan. Sehingga dengan menggunakan [RsaCtfTool](#), plaintext dapat didapatkan dengan mudah.

```
python3 RsaCtfTool.py -n  
6347005034877661581150385096194222959694176703809130581901  
45906857689 --uncipher  
4819421926185556320321513231156581364962421208216896344856  
8445899090 -e 65537
```

- c. Didapatkan plaintext sebagai berikut:

```
[hacking_env]-(gamer㉿kali)-[~/Hacking/RsaCtfTool]
$ python3 RsaCtfTool.py -n 63470050348776615811503850961942229596941767038091305819014590685768
9 --uncipher 48194219261855563203215132311565813649624212082168963448568445899090 -e 65537
private argument is not set, the private key will not be displayed, even if recovered.

[*] Testing key /tmp/tmp9_1jss86. sebagai berikut:
[*] Performing mersenne_primes attack on /tmp/tmp9_1jss86. 29321513231156581364962421208216896
24%|███████████| 12/51 [00:00<00:00, 449389.71it/s]
[*] Performing factordb attack on /tmp/tmp9_1jss86.
[*] Attack success with factordb method !5537

Results for /tmp/tmp9_1jss86:
n=634700503487766158115038509619422295969417670380913058190145906857689
45906857689

Unciphered data :
2. Pertama kali lihat adalah nilai n yang sangat kecil, sehingga mudah difak
HEX : 0x0000000000000046696e6449544354467b336135795f3132694768743f7d
INT (big endian) : 674411449301419664991736406792079648691256337408212861
INT (little endian) : 3376675894970921618247842278024399466162966805179680248753517102104576
utf-8 : FindITCTF{3a5y_12ighT?}
STR : b'\x00\x00\x00\x00\x00\x00FindITCTF{3a5y_12ighT?}'
```

Flag: FindITCTF{3a5y_12ight?}

2. Kategori: Crypto

Nama: Number

Langkah Penggerjaan:

- Pada soal terdapat suatu rangkaian text yang hanya berisi angka:

7020210520211020210020273202842026720284202702021232025220
2832026720233202732029520289202482028520295202712024820284
202952024920284202125

- Bila diteliti lebih lanjut, terdapat suatu susunan angka yang selalu berulang, yaitu angka 202. Bila semua susunan angka ini dihapus dan diganti dengan spasi, maka akan didapatkan representasi decimal dari range ascii printable:

70 105 110 100 73 84 67 84 70 123 52 83 67 33 73 95 89 48
85 95 71 48 84 95 49 84 125

- Menggunakan tool konversi [kode decimal ascii ke huruf ascii](#) akan didapatkan teks sebagai berikut:

Flag: FindITCTF{4SC! I_YOU_GOT_1T}

3. Kategori: Crypto

Nama: Vigenere

Langkah penggerjaan:

- Pada soal diberikan sebuah teks berisi:

Hint: vigorous and sincere. 45 5A 43 52 59 50 54 4F.

Code: XGQLYGM0ZHV1VLTKSID

Note: Masukkan jawaban yang ditemukan ke dalam format flag CTF Find IT untuk mendapatkan flag seutuhnya.

- Sesuai dengan namanya, soal ini pasti berhubungan dengan Vigenere Cipher.

Dengan adanya ciphertext, maka yang dibutuhkan adalah keynya.

- Pada hint terdapat rangkaian hexcode ascii range printable, setelah diubah menjadi huruf ascii, maka menjadi:

EZCRYPTO

- Dengan menggunakan tool online [Vigenere Cipher](#), key dan ciphertext dimasukkan sehingga ditemukan:

THOUARTAVIGENEREE

- Maka Flag: FindITCTF{THOUARTAVIGENEREE}

4. Kategori: Crypto

Nama: RevTrans

Langkah pengerajan:

- a. Pada challenge diberikan sebuah file python dan output. Yang pertama saya lakukan adalah memahami alur program dengan memberi komentar pada file. Dari kode tersebut sebahagian saya akan melakukan reverse dari plaintext, membagiannya menjadi sebanyak *order* sama rata, kemudian menyusun ciphertext perbagian mengikuti *order* dengan key *index*.

```
flag = FindItCTF{REDACTED}

# bagi into chunks
def split_len(seq, length):
    return [seq[i:i + length] for i in range(0, len(seq), length)]

def encode(key, plaintext):
    order = {
        int(val): num for num, val in enumerate(key)
    }

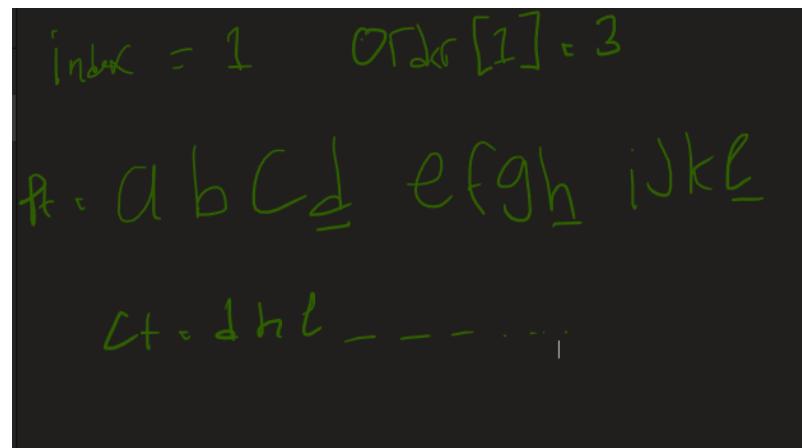
    reversetext = ''
    ciphertext = ''
    i=len(plaintext)-1
    while i >= 0:
        reversetext = reversetext + plaintext[i]
        i=i-1

    "untuk setiap order"
    for index in sorted(order.keys()):
        " bagi 4 4 bagian"
        for part in split_len(reversetext, len(key)):
            try:ciphertext += part[order[index]]
            except IndexError:
                continue

    return ciphertext

print(encode('4321', flag))
```

Berikut alur enkripsi secara manual yang saya lakukan



- b. Dari percobaan tersebut, saya merasa teknik enkripsi seperti ini dapat dilakukan secara brute force dengan melakukan mapping dari posisi awal ke posisi setelah dilakukan enkripsi. Untuk mengetes hipotesis saya, saya mengganti flag dengan flag buatan yang menyamai panjang dari output dan tetap mempertahankan posisi kurung kurawal.

```
flag = '123456789{qwertyuiopasdfghj}'
```

dari flag buatan tersebut dikeluarkan seperti ini

```
gaue951hsir{62jdotq73}fpyw84 # flag buatan  
eiluFIFatv_{Tiy_esyCn}y_o0Td # flag asli
```

hmmmm, sangat menarik kurung kurawal berada di tempat yang sama.

- c. Merasa yakin saya membuat script untuk melakukan mapping index dari posisi awal ke akhir

```
sample      = "123456789{qwertyuiopasdfghj}"  
sample_enc  = "gaue951hsir{62jdotq73}fpyw84"  
enc_flag    = "eiluFIFatv_{Tiy_esyCn}y_o0Td"  
  
mapping = {}  
  
for i in range(len(sample)):  
    mapping[i] = sample_enc.index(sample[i])  
  
for i in range(len(enc_flag)):  
    print(enc_flag[mapping[i]], end="")
```

dengan output

```
PS S:\Kerjaan\sybir sekus\Finditctf\Crypto\RevTrans> python .\asd.py  
FindITCTF{you_solve_it_yeay}  
PS S:\Kerjaan\sybir sekus\Finditctf\Crypto\RevTrans>
```

maka flag: FindITCTF{you_solve_it_yeay}

5. Kategori: Crypto

Nama: Happy Forever

Langkah pengerjaan:

Yang pertama saya lakukan pertama memahami chall.py dan melihat output.txt
chall.py

```
from secret import HAPPY_NUM, FLAG
import base64

"""reverse kemudian baca jadi hex """
FLAGenc = FLAG[::-1].encode().hex()

ciphertext = [(ord(c)^HAPPY_NUM[i]+i) for i,c in
enumerate(FLAGenc)]

"""setiap hex char di xor happy_num + i"""
"""assume happy_num list of num"""
for i, c in enumerate(FLAGenc):
    ciphertext += ord(c)^HAPPY_NUM[i]+i

print(f"ciphertext = {ciphertext}")
```

```
ciphertext = [54, 108, 63, 41, 36, 37, 17, 31, 27, 12, 8, 118, 97,
101, 85, 3, 88, 9, 64, 64, 79, 186, 164, 160, 174, 149, 246, 249,
237, 189, 232, 131, 223, 146, 206, 305, 317, 291, 294, 286, 284,
374, 379, 358, 362, 367, 337, 270, 344, 328, 333, 332, 328, 439,
425, 405, 403, 400, 415, 469, 388, 399, 500, 500, 509, 424, 480,
488, 458, 608, 547, 553, 533, 512, 512, 515, 599, 588, 692, 738,
693, 640, 652, 675, 759, 763, 740, 736, 738, 750, 726, 732, 735,
724, 705, 714, 823, 876, 827, 800, 788, 796]
```

dari sini saya mengubah dari list comprehension menjadi bentuk blok agar lebih mudah di pahami, dalam setiap iterasi, setiap index ascii dari char hex dalam FLAGenc dilakukan xor dengan HAPPY_NUM sebuah list yang saya asumsikan sebuah list angka yang mengikuti sebuah aturan. Dalam hint disebutkan *There is only one happiness in this life, to love and be loved. We should be 10 times happier from now on. :)*, pasti ada hubungan dengan 10.

Berjam jam stuck saya mencoba mencari “happy number” dan menemukan situs [wikipedia](#) di paling atas. WOW ternyata happy number merupakan sebuah angka spesial yang mengikuti sebuah aturan. Saya tidak membaca dengan detil dan langsung melakukan scroll ke bawah dan menemukan [ini](#).

[10-happy numbers](#) [edit]

For $b = 10$, the only positive perfect digital invariant for $F_{2,b}$ is the trivial perfect digital invariant 1, and the only cycle is the eight-number cycle

4 → 16 → 37 → 58 → 89 → 145 → 42 → 20 → 4 → ...

and because all numbers are preperiodic points for $F_{2,b}$, all numbers either lead to 1 and are happy, or lead to the cycle and are unhappy. Because base 10 has no other perfect digital invariants except for 1, no positive integer other than 1 is the sum of the squares of its own digits.

In base 10, the 143 10-happy numbers up to 1000 are:

1, 7, 10, 13, 19, 23, 28, 31, 32, 44, 49, 68, 70, 79, 82, 86, 91, 94, 97, 100, 103, 109, 129, 130, 133, 139, 167, 176, 188, 190, 192, 193, 203, 208, 219, 226, 230, 236, 239, 262, 263, 280, 291, 293, 301, 302, 310, 313, 319, 320, 326, 329, 331, 338, 356, 362, 365, 367, 368, 376, 379, 383, 386, 391, 392, 397, 404, 409, 440, 446, 464, 469, 478, 487, 490, 496, 536, 556, 563, 565, 566, 608, 617, 622, 623, 632, 635, 637, 638, 644, 649, 653, 655, 656, 665, 671, 673, 680, 683, 694, 700, 709, 716, 736, 739, 748, 761, 763, 784, 790, 793, 802, 806, 818, 820, 830, 836, 847, 860, 863, 874, 881, 888, 899, 901, 904, 907, 910, 912, 913, 921, 923, 931, 932, 937, 940, 946, 964, 970, 973, 989, 998, 1000 (sequence [A007770](#) in the OEIS).

The distinct combinations of digits that form 10-happy numbers below 1000 are (the rest are just rearrangements and/or insertions of zero digits):

1, 7, 13, 19, 23, 28, 44, 49, 68, 79, 129, 133, 139, 167, 188, 226, 236, 239, 338, 356, 367, 368, 379, 446, 469, 478, 556, 566, 888, 899. (sequence [A124095](#) in the OEIS).

The first pair of consecutive 10-happy numbers is 31 and 32.^[4] The first set of three consecutive is 1880, 1881, and 1882.^[5] It has been proven that there exist sequences of consecutive happy numbers of any natural number length.^[6] The beginning of the first run of at least n consecutive 10-happy numbers for $n = 1, 2, 3, \dots$ is^[7]

1, 31, 1880, 7839, 44488, 789999999999959999999996, 789999999999959999999996, ...

Saya langsung mengkopi sequence tersebut dan mencoba melakukan dekripsi

```
ciphertext = [54, 108, 63, 41, 36, 37, 17, 31, 27, 12, 8, 118, 97, 101, 85, 3, 88, 9, 64, 64, 79, 186, 164, 160, 174, 149, 246, 249, 237, 189, 232, 131, 223, 146, 206, 305, 317, 291, 294, 286, 284, 374, 379, 358, 362, 367, 337, 270, 344, 328, 333, 332, 328, 439, 425, 405, 403, 400, 415, 469, 388, 399, 500, 500, 500, 509, 424, 480, 488, 458, 608, 547, 553, 533, 512, 512, 515, 599, 588, 692, 738, 693, 640, 652, 675, 759, 763, 740, 736, 738, 750, 726, 732, 735, 724, 705, 714, 823, 876, 827, 800, 788, 796]

not_happy_number = [1, 7, 10, 13, 19, 23, 28, 31, 32, 44, 49, 68, 70, 79, 82, 86, 91, 94, 97, 100, 103, 109, 129, 130, 133, 139, 167, 176, 188, 190, 192, 193, 203, 208, 219, 226, 230, 236, 239, 262, 263, 280, 291, 293, 301, 302, 310, 313, 319, 320, 326, 329, 331, 338, 356, 362, 367, 368, 376, 379, 383, 386, 391, 392, 397, 404, 409, 440, 446, 464, 469, 478, 487, 490, 496, 536, 556, 563, 565, 608, 617, 622, 623, 632, 635, 637, 638, 644, 649, 653, 655, 656, 665, 671, 673, 680, 683, 694, 700, 709, 716, 736, 739, 748, 761, 763, 784, 790, 793, 802, 806, 818, 820, 833, 836, 847, 860, 863, 874, 881, 888, 899, 901, 904, 907, 910, 912, 913, 921, 923, 931, 932, 937, 940, 946, 964, 970, 973, 989, 998, 1000]

new_hex = ""
for i, ec in enumerate(ciphertext):
    new_hex += chr(ec^(not_happy_number[i] + i))

p = bytes.fromhex(new_hex).decode('utf-8')
print(p[::-1])
```

Dan output seperti ini

```
PS S:\Kerjaan\sybir sekus\Finditctf\Crypto> cd ..\happy forever\  
PS S:\Kerjaan\sybir sekus\Finditctf\Crypto\Happy Forever> python .\toottts.py  
FindITCTF{1_5H0Uld_B3_h4pPy_4f73r4Ll_r19H7?_999999}  
PS S:\Kerjaan\sybir sekus\Finditctf\Crypto\Happy Forever>
```

Maka flag: FindITCTF{1_5H0Uld_B3_h4pPy_4f73r4Ll_r19H7?_999999}

6. Kategori: Crypto

Nama: RandomSHA

Langkah penggeraan:

Dari output dan chall.py kita tahu program menyimpan hasil hashing setiap penambahan character, kita tinggal mengikuti jejak hashnya dan mendapatkan flagnya. Yang membuatnya tidak semudah itu adalah adanya campur tangan random, untungnya kita diberikan seednya dan dapat melakukan brute force dengan mengikuti alur program.

```
import random,string  
import hashlib  
ct =[  
16827491998982303935845912726250001246506259002934220576276361965668  
860811468,  
.,  
.,"  
32141083108499196137603954986721075935614843412879401775449594747381  
368240336  
]  
  
crip_flag = ""  
brute =  
"""abcdefghijklmnopqrstuvwxyz0123456789ABCDEFGHIJKLMNPQRSTUVWXYZ#$%  
&\\"'(),-.:/;<=>?@[\\]^_`|~\}{{\""""  
for i, hashc in enumerate(ct): # string main  
    for cc in brute: # guessing game  
        random.seed("FINDIT")  
        enc_fl_guess =""  
        for c in crip_flag + cc:  
            if c.islower():  
                enc_fl_guess +=  
chr((ord(c)-ord('a')+random.randrange(0,26)) % 26 + ord('a'))
```

```

        elif c.isupper():
            enc_fl_guess +=

chr((ord(c)-ord('A')+random.randrange(0,26))%26 + ord('A'))

        elif c.isdigit():
            enc_fl_guess +=

chr((ord(c)-ord('0')+random.randrange(0,10))%10 + ord('0'))

        else:
            enc_fl_guess += c

    if int(hashlib.sha512(enc_fl_guess.encode()).hexdigest(),
16)>>256 == ct[len(enc_fl_guess) - 1]:
        crip_flag += cc
        print(crip_flag)

```

Output :

```

PS S:\Kerjaan\sybir sekus\Finditctf\Crypto\RandomSHA> python
.\asd.py
F
Fi
Fin
Find
..
..
FindITCTF{W3ll_R4nd0m_4nd_SHA_r1ghtt
FindITCTF{W3ll_R4nd0m_4nd_SHA_r1ghtt?
FindITCTF{W3ll_R4nd0m_4nd_SHA_r1ghtt?}
PS S:\Kerjaan\sybir sekus\Finditctf\Crypto\RandomSHA>

```

Maka flag: FindITCTF{W3ll_R4nd0m_4nd_SHA_r1ghtt?}

7. Kategori: Rev

Nama: SimpleActivationKey

Langkah penggeraan:

- Pada soal diberikan sebuah teks hint.txt dan sebuah executable.
- Terdapat string berupa flag pada saat membedah data executable tersebut, dan flag ditemukan dengan melakukan strings pada executable tersebut:

```

__hacking_env__(gamer㉿kali)-[~/Hacking/CTFs/finditctf/rev_simpleactivation]
└─$ strings ActivationProgramR1.exe | grep FindIT
The flag is: FindITCTF{j4D1_Sk1Dr0w_aD4LAH_Imp!4NkU}

```

7. Kategori: Rev
Nama: SimpleActivationK
Langkah pengerjaan:

c. Maka Flag: FindITCTF{j4D1_Sk1Dr0w_aD4LAH_Imp!4NkU}

8. Kategori: Forensic

Nama: Citra

Langkah pengerjaan:

- Diberikan sebuah attachment txt oleh soal, sekilas tidak terlihat apapun melihat teks tersebut. Namun, saat diteliti lagi lebih dalam tiap line pada teks tersebut berisi 1920 angka, dan terdapat 1113 line. Kemungkinan besar teks ini mengarah kepada data gambar
- Terdapat juga pada salah satu clue yang dirilis oleh panitia:

Hint soal Citra

txt -> csv

- File txt tersebut juga memiliki struktur yang sama dengan file csv pada umumnya, maka yang akan dilakukan adalah dengan membuat script yang dapat mengubah data pixel pada teks tersebut, lalu mengubahnya menjadi sebuah gambar:

```

from numpy import genfromtxt
import matplotlib
from matplotlib import pyplot
from matplotlib.image import imread

my_data = genfromtxt('chall.txt', delimiter=',')
matplotlib.image.imsave('out.png', my_data)
image_1 = imread('out.png')
# plot raw pixel data
pyplot.imshow(image_1)
# show image
pyplot.show()

```

- Didapatkan gambar sebagai berikut:



- e. Dari gambar didapatkan teks hex sebagai berikut:

```
46696E6449544354467B59345F4E6  
4346B5F5461755F4B306B5F4E6434  
4B5F54346E79415F4D34737433725  
F393939393939393939397D
```

Hex tersebut kemudian di-decode menjadi ascii sehingga didapatkan:

```
FindITCTF{Y4_Nd4k_Tau_K0k_Nd4K_T4nyA_M4st3r_9999999999}
```

- f. MakaFlag:

```
FindITCTF{Y4_Nd4k_Tau_K0k_Nd4K_T4nyA_M4st3r_9999999999}
```

9. Kategori: Forensic

Nama: HierarchicalUGM

Langkah pengerjaan:

- Oleh soal diberikan sebuah attachment berupa gambar, yaitu chall.png. Sekilas tidak terlihat apapun, namun saat dioperasikan binwalk pada gambar tersebut, terdapat beberapa file yang tersembunyi, yaitu sebuah archive dan gambar:

```

└─(hacking_env)─(gamer㉿kali)─[~/Hacking/CTFs/finditctf/forensic_hierarchical]
└─$ binwalk --dd=".*" chall.png

DECIMAL      HEXADECIMAL      DESCRIPTION
-----      -----      -----
0            0x0          PNG image, 612 x 690, 8-bit/color RGBA, non-interlaced
14608        0x3910        Zip archive data, at least v2.0 to extract, compressed size: 1991786, uncompressed size:
1993069, name: pict1.png
2006524        0x1E9DFC        End of Zip archive, footer length: 22

```

- b. Gambar yang dihasilkan merupakan gambar yang sama seperti awal, namun ketika zip tersebut di-extract, muncul gambar yang berbeda, pict1.png:



- c. Dapat dicurigai bahwa challenge ini berupa archive bertumpuk karena memang nama challenge berhubungan dengan hierarchi/tree. Maka seperti pada gambar pertama, dilakukan juga binwalk pada gambar kedua, pict1.png:

```

└─(hacking_env)─(gamer㉿kali)─[~/.../CTFs/finditctf/forensic_hierarchical/_chall.png.extracted]
└─$ binwalk --dd=".*" pict1.png

DECIMAL      HEXADECIMAL      DESCRIPTION
-----      -----      -----
0            0x0          PNG image, 612 x 690, 8-bit/color RGBA, non-interlaced
14550        0x38D6        RAR archive data, version 5.x

```

ditemukan juga sebuah gambar yang masih sama dengan pict1.png, dan sebuah archive berupa rar. Ketika rar tersebut di-extract, didapatkan pict2.png:



- d. Sama halnya dengan file-file sebelumnya, binwalk juga dilakukan pada pict2.png:

```

└─(hacking_env)─(gamer㉿kali)─[~/.../finditctf/forensic_hierarchical/_chall.png.extracted/_pict1.png.extracted]
$ binwalk --dd=".*" pict2.png

DECIMAL      HEXADECIMAL      DESCRIPTION
-----      -----      -----
0            0x0          PNG image, 612 x 690, 8-bit/color RGBA, non-interlaced
14566        0x38E6        PNG image, 815 x 700, 8-bit/color RGBA, non-interlaced
14665        0x3949        Zlib compressed data, default compression

```

berbeda dengan sebelumnya, kali ini terdapat 2 gambar, gambar pict2.png yang sama, beserta 1 gambar lagi yang berbeda. Namun ketika gambar tersebut dibuka, terdapat error sebagai berikut:



Ketika dilakukan pngcheck pada gambar tersebut didapatkan output:

```

└─(hacking_env)─(gamer㉿kali)─[~/.../forensic_hierarchical/_chall.png.extracted]
└─$ pngcheck -v 38E6
File: 38E6 (1965409 bytes)
    invalid chunk name "|HDR" (7c 48 44 52)
ERRORS DETECTED in 38E6

```

- e. IHDR merupakan bagian dari header suatu gambar png yang berisi informasi detail dari gambar tersebut. Dalam hal ini, yang ditulis dalam file bukanlah string “IHDR”, melainkan “|HDR”. Maka hex editor, dalam hal ini ghex dapat digunakan untuk memperbaiki hal tersebut dengan mengubah |HDR menjadi IHDR:

The screenshot shows a hex editor window titled "38E6 - GHex". The menu bar includes File, Edit, View, Windows, and Help. The main pane displays the byte content of a file, starting with the IHDR chunk. A blue selection bar highlights the byte at offset 13, which is the value 49 (hex). The IHDR chunk bytes are: 89 50 4E 47 0D 0A 1A 0A 00 00 00 00 0D 49 48 44 52. The file extension ".PNG" is visible at the end of the file.

- f. Setelah dibuka lagi, didapatkan error yang berbeda lagi:



CRC (Cyclic Redundancy Check) pada file PNG bertujuan untuk mendeteksi error pada data gambar tersebut, apabila data apapun yang ada dalam gambar diubah, maka data CRC dalam file tersebut juga harus diubah.

Ketika dilakukan pngcheck pada gambar tersebut juga menandakan bahwa CRC harus diganti:

```
└─(hacking_env)─(gamer㉿kali)─[~/.../forensic_hierarchical/_chall.png  
ed]  
└$ pngcheck -v 38E6  
File: 38E6 (1965409 bytes)  
    chunk IHDR at offset 0x0000c, length 13  
        815 x 700 image, 32-bit RGB+alpha, non-interlaced  
        CRC error in chunk IHDR (computed e6a88069, expected 07d6edb4)  
ERRORS DETECTED in 38E6
```

Maka hex editor digunakan lagi untuk mengganti value 06d6edb4 menjadi e6a88069:

The screenshot shows the hex editor again with the file 38E6. The byte at offset 13, which was previously 49 (hex), has been changed to 69 (hex). The rest of the file content remains the same as shown in the first screenshot.

Didapatkan gambar sebagai berikut:



- g. Mulai dari sini berbeda dari yang sebelum-sebelumnya, bila binwalk dilakukan pada gambar tersebut, tidak ada lagi informasi berguna yang didapatkan. Namun, pada saat dilakukan pngcheck lagi pada gambar tersebut didapatkan informasi sebagai berikut:

```
chunk IDAT at offset 0x1deb93, length 4538
chunk IEND at offset 0x1dfd59, length 0
No errors detected in 38E6 (244 chunks, 13.9% compression).
```

Telah dilakukan kompresi yang cukup berat pada gambar tersebut. Hal ini sangat aneh karena pada saat melihat gambar tersebut, resolusi gambar sudah lumayan tinggi untuk ukuran 2mb. Setelah melakukan riset singkat dan clue dari panitia:

Clue HierarchicalUGM

Hex resize the PNG

Ternyata menyembunyikan pixel gambar dengan mengubah informasi IHDR pada suatu gambar PNG dapat dilakukan tanpa mengurangi data pada gambar tersebut.

- h. Maka yang dilakukan adalah menggunakan hex editor untuk mengubah informasi IHDR yang ada pada gambar tersebut, dan pada waktu yang sama memperbaiki juga CRC yang dihasilkan setiap kali mengubah informasi gambar. IHDR pada PNG secara garis besar terdiri dari 4 byte width dan 4 byte height. Karena mengubah width dapat mengubah keseluruhan dari gambar tersebut (skew), maka yang dapat dilakukan adalah mengubah informasi height dari gambar tersebut:

38E6.png - GHex

File Edit View Windows Help

00000000	89 50 4E 47 0D 0A 1A 0A 00 00 00 00 0D 49 48 44 52	.PNG.....IHDR
00000010	00 00 03 2F 00 00 05 BC 08 06 00 00 00 E6 A8 80	.../....
00000020	69 00 00 00 09 70 48 59 73 00 00 04 9D 00 00 04	i....pHs.....
00000030	9D 01 7C 34 6B A1 00 00 00 19 74 45 58 74 53 6F	.. 4k....tEXtSo
00000040	66 74 77 61 72 65 00 77 77 77 2E 69 6E 6B 73 63	ftware.www.inksc
00000050	61 70 65 2E 6F 72 67 9B EE 3C 1A 00 00 20 00 49	ape.org..<... .I
00000060	44 41 54 78 9C BD D9 CF 24 C9 76 1F F6 3B 91	DATx.....\$.v...;
00000070	99 55 F5 ED BD 2F B3 DE 7B 75 17 F2 82 14 37 1B	.U.../..{u....7.
00000080	F4 05 4D 09 22 0D 4B 36 21 3F 08 B0 F5 2C FF 3B	.M.".K6!?....,.
00000090	86 FF 02 43 B0 5E 0C FB C5 80 65 01 12 A0 07 19	...C.^....e....

Signed 8 bit: -68 Unsigned 8 bit: 188 Signed 16 bit: 2236 Unsigned 16 bit: 2236 Signed 32 bit: 395452 Unsigned 32 bit: 395452 Signed 64 bit: -6276329030693812036 Unsigned 64 bit: 12170415043015739580 Hexadecimal: BC Octal: 274 Binary: 10111100 Stream Length: 8 - +

Float 32 bit: 5.541463e-40 Float 64 bit: -1.143495e-111

Show little endian decoding Show unsigned and float as hexadecimal

Offset: 0x17

Height dari 0x000002bc (700 pixel) diubah menjadi 0x000005bc (1468 pixel).

CRC kemudian juga dicek dengan pngcheck dan diubah berdasarkan outputnya.

Sehingga didapatkan gambar sebagai berikut:



- i. Maka Flag: FindITCTF{y0u_g0t_m3}

10. Kategori: Forensic

Nama: RouterOS

Langkah pengerjaan:

- a. Oleh soal disediakan sebuah attachment berupa .ova. File ekstension ova merupakan suatu file yang menyimpan suatu data sebuah virtual machine, biasanya dipakai pada software virtual machine manager seperti virtualbox dan VMWare. Pada challenge ini saya akan menggunakan virtualbox.
- b. Pada virtualbox, attachment ova tersebut di-import menjadi sebuah virtual machine:

```
u
tdown
may/09/2022 08:48:42 system,error,critical router was rebooted without proper sh
u
tdown
may/09/2022 08:49:22 system,error,critical login failure for user admin via loca
l

may/09/2022 08:49:37 system,error,critical login failure for user admin via loca
l

may/09/2022 08:49:44 system,error,critical login failure for user admin via loca
l

may/09/2022 08:50:08 system,error,critical login failure for user admin via loca
l

[admin@Anna-Router] >
caps-man    interface  port      snmp      user-manager  ping
certificate ip        ppp       special-login beep      quit
console     ipv6       queue    system    export      redo
disk        log        radius   tool      import      undo
file        mpls      routing  user      password
line 5 of 5> _
```

- c. Berdasarkan namanya, virtual machine yang disediakan adalah Mikrotik RouterOS, ketika dilakukan command linux biasa seperti ls maupun cd tidak menghasilkan efek apapun, namun ketika dilakukan double tab dan mengetikkan nama folder, akan menuju ke folder tersebut, dan ketika mengetik nama program, akan mengeksekusi program tersebut.
- d. Setelah mengeksplor virtual machine tersebut, terdapat suatu program yang menampilkan flag, yaitu

export

```

>ip dhcp-client
add
>ip dhcp-server network
add address=10.0.0.0/24 dns-server=10.0.0.1 gateway=10.0.0.1
>ip dns
set allow-remote-requests=yes
>ip dns static
add name=google.com text="FindITCTF{You_Got_Me!_7789a}" type=TXT
add address=10.0.0.1 name=google.com
add cname=google.com regexp="^a-z0-9].google.com" type=CNAME
>ip service
set ftp disabled=yes
set www disabled=yes
set ssh disabled=yes
set api disabled=yes
set winbox disabled=yes
set api-ssl disabled=yes
>ip upnp
set enabled=yes
>system identity
set name=Anna-Router
>system note
set note="You can't find me!\r\n
          \nHAHAHAHA"
[admin@Anna-Router] > _

```

- e. Maka Flag: FindITCTF{You_Got_Me!_7789a}

11. Kategori: OSINT

Nama: 'Job'

Langkah penggeraan:

- Diberikan sebuah gambar, dengan pencarian google kata kunci tersebut, ditemukan hasil sebagai berikut:

hardware programmer gadjah mada robotic team jan 2022

About 80,800 results (0.49 seconds)

Tip: Search for English results only. You can specify your search language in Preferences

<https://id.linkedin.com/gigahidjrikaaa> · Translate this page

Giga Hidjrika Aura Adkhy - Gadjah Mada Robotic Team

My current job is being a **hardware programmer** for **Gadjah Mada Robotic Team**, sub-team "HEROES". I am fluent in Indonesian, Javanese, and English. Planning on ...

- Pada hasil pertama, terlihat posting yang berisi flag pada profil:

Activity

171 followers

Giga Hidjrika Aura Adkhy commented on a post • 5d

You found the flag! It's FindITCTF{Op3n_s0urc3_InGf0}

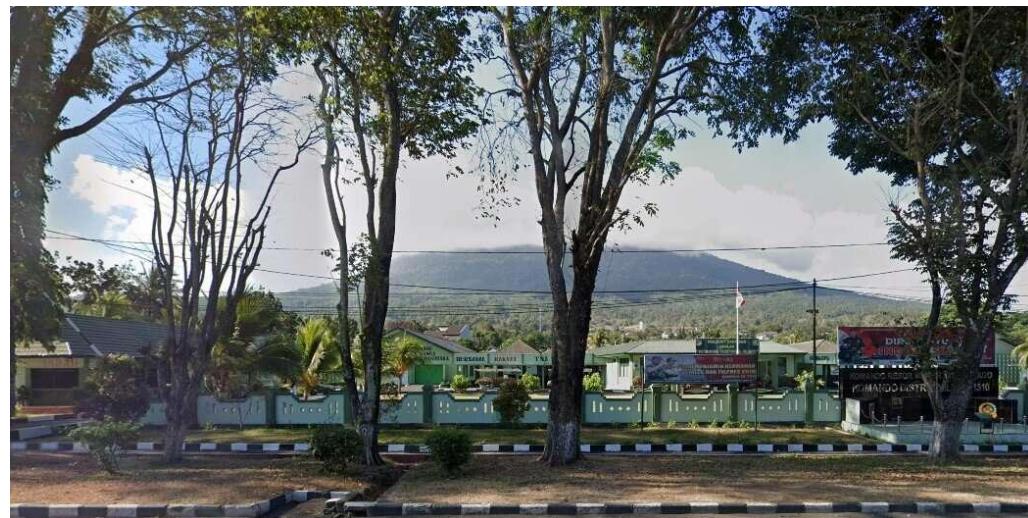
- c. Flag: FindITCTF{Op3n_s0urc3_InGf0}

12. Kategori: OSINT

Nama: Commander

Langkah pengerjaan:

- a. Oleh soal disediakan gambar suatu lokasi sebagai berikut:



Solusi soal ini adalah komandan yang beroperasi dari tempat tersebut.

- b. Bila diteliti lebih jelas lagi, terdapat beberapa clue:
 - i. Tempat tersebut merupakan Komando Distrik Militer (KODIM)
 - ii. Nomor Kodim tersebut memiliki akhiran 10
 - iii. Terdapat gunung yang besar di belakang tempat tersebut
 - iv. Tempat tersebut cukup terdokumentasi sehingga dapat diketahui siapa komandan yang sedang menjabat saat ini.
- c. Berdasarkan beberapa clue tersebut, diteliti Kodim yang nomornya berakhiran dengan nomor 10. Ditemukan Kodim 1310 yang mirip dengan gambar tersebut:



- d. Komandan yang sedang menjabat saat itu menurut wikipedia adalah Letkol Arm Yoki Efriandi Gumay, M.Han. Maka:
- e. Flag: FindITCTF{LETKOLARMYOKIEFRIANDIGUMAY,M.HAN.}

13. Kategori: OSINT

Nama: Dimana

Langkah penggeraan:

- a. Disediakan gambar suatu lokasi oleh soal, dan dicari letak jalannya:



- b. Dari clue gambar tersebut, terdapat banner TK Khalifah, hanya ada beberapa TK Khalifah yang ada pada sekitar Yogyakarta, sehingga setelah mencari satu per-satu, ditemukan:



- c. Lokasi terletak pada Jalan Tunjang 3, Yogyakarta, maka

Flag: FindITCTF{Tunjang}

14. Kategori: Misc

Nama: Wordle Moment

Langkah pengerjaan:

- Oleh soal disediakan attachment berupa executable windows (.exe). Hal pertama yang dilakukan adalah melakukan strings pada executable tersebut.
- strings WordleCTF.exe

Menghasilkan output yang bila dilihat sekilas, terdapat string berupa flag:

```
(hacking_env)–(gamer㉿kali)-[~/Hacking/CTFs/finditctf/misc_wordle]
└─$ strings WordleCTF.exe | grep FindIT
FindITCTF{Ez_Sc0Re_!yes_l1M1t}
```

- c. maka Flag: FindITCTF{Ez_Sc0Re_!yes_l1M1t}