

Writeup LycoReco

Gemastik 2022 - Penyisihan



Anggota tim:

Muhammad Garebaldhie Er Rahman (average kobo enjoyer)

Frederik Imanuel Louis (azuketto)

Rachel Gabriela Chen (chaerla)

Daftar Isi

Daftar Isi	2
Crypto	3
Doublesteg	3
Flag: Gemastik2022{uji_nyali_encrypt_message_pakai_weak_key}	6
Rev	7
CodeJugling	7
Flag: Gemastik2022{st45iUn_MLG_k07a_b4rU}	11
Dino	11
Flag: Gemastik2022{why_would_you_ever_beat_me}	14
Rubyte	14
Flag: Gemastik2022{i_still_remember_30_october}	16
Foren	17
Traffic Enjoyer	17
Flag : Gemastik2022{balapan_f1rst_b100d_is_real_f580c176}	19
Har	20
Flag : Gemastik2022{kinda_wish_this_werent_text}	22

Crypto

Doublesteg

892

Single STEG encryption is weak, how about double STEG encryption?

author - deomkicer#3362

```
def encrypt(msg: bytes, key: bytes):  
    key = SHA256.new(key).digest()  
    iv = STEG * 2  
    aes = AES.new(key, AES.MODE_CBC, iv)  
    enc = aes.encrypt(msg)  
    return enc  
  
def double(msg: bytes, keys: list[bytes]):  
    msg = pad(msg, AES.block_size)  
    for key in keys:  
        msg = encrypt(msg, key)  
    return msg
```

Pada chall, png pada dasarnya diencrypt menggunakan AES CBC mode sebanyak dua kali menggunakan key yang berbeda. Pada chall, iv yaitu STEG sudah diberikan, dan key diderive dari STEG yang hanya berukuran 8 byte yang dipermutasi. Perhatikan bahwa $8! = 4 * 10^5$. Karena enkripsi dilakukan dua kali, enkripsi rentan terhadap Meet in the Middle attack. Perhatikan pula bahwa file yang diencrypt adalah PNG,

sehingga kita memiliki sebagian plaintext, yaitu header dari PNG. Dengan itu kita dapat melakukan mapping dari enkripsi header PNG dengan semua kemungkinan key yang mungkin. Kemudian, kita mendekripsi ciphertext dengan semua key yang mungkin dan mengecek apakah header ada pada mapping yang kita buat. Search tersebut dapat dilakukan dengan kompleksitas $O(\log(n))$ dengan $n = 8!$, dan untuk semua key, kompleksitas total menjadi $O(n \log(n))$ yang masih feasible untuk bruteforce. Code bruteforce diimplementasikan sebagai berikut:

```
from Crypto.Cipher import AES
from Crypto.Hash import SHA256
from Crypto.Util.Padding import *
import random
import itertools

STEG = b"gemasteg"

import bisect

def find_in_sorted_list(elem, sorted_list):
    # https://docs.python.org/3/library/bisect.html
    'Locate the leftmost value exactly equal to x'
    i = bisect.bisect_left(sorted_list, elem)
    if i != len(sorted_list) and sorted_list[i] == elem:
        return i
    return -1

def fwrite(filename: str, data: bytes):
    f = open(filename, "wb")
    f.write(data)
    f.close()

def encrypt(msg: bytes, key: bytes):
    key = SHA256.new(key).digest()
    iv = STEG * 2
    aes = AES.new(key, AES.MODE_CBC, iv)
    enc = aes.encrypt(msg)
    return enc
```

```

def decrypt(msg: bytes, key: bytes):
    key = SHA256.new(key).digest()
    iv = STEG * 2
    aes = AES.new(key, AES.MODE_CBC, iv)
    enc = aes.decrypt(msg)
    return enc

def double(msg: bytes, keys: list[bytes]):
    msg = pad(msg, AES.block_size)
    for key in keys:
        msg = encrypt(msg, key)
    return msg

def double_decrypt(msg: bytes, keys: list[bytes]):
    msg = pad(msg, AES.block_size)
    for key in keys:
        msg = decrypt(msg, key)
    return msg

FLAG = open("flag.enc", "rb").read()

def getrandsteg():
    x = list(STEG)
    random.shuffle(x)
    return bytes(x)
print(getrandsteg())

sample = open("about_img.png", "rb").read()
a = sample[:16]

print(len(encrypt(a, STEG))) #16

head = []
mp = {}
for ls in list(itertools.permutations(STEG)):
    key = [chr(c).encode() for c in ls]
    key = b"".join(key)
    ct = encrypt(a, key)
    mp[ct]=key

```

```

head.append(ct)

head.sort()

for ls in list(itertools.permutations(STEG)):
    key = [chr(c).encode() for c in ls]
    key = b"".join(key)
    semi = decrypt(FLAG[:32], key)
    idx = find_in_sorted_list(semi[:16], head)
    if idx!=-1:
        mid = head[idx]
        newkey = mp[mid]
        buf = double_decrypt(FLAG, [key, newkey])
        fwrite("plain.png", buf)

```

Kemudian, pada plain.png, diperoleh png sebagai berikut:



Flag: Gemastik2022{uji_nyali_encrypt_message_pakai_weak_key}

Rev

CodeJugling

500

Find the flag!

author - vidner#6838

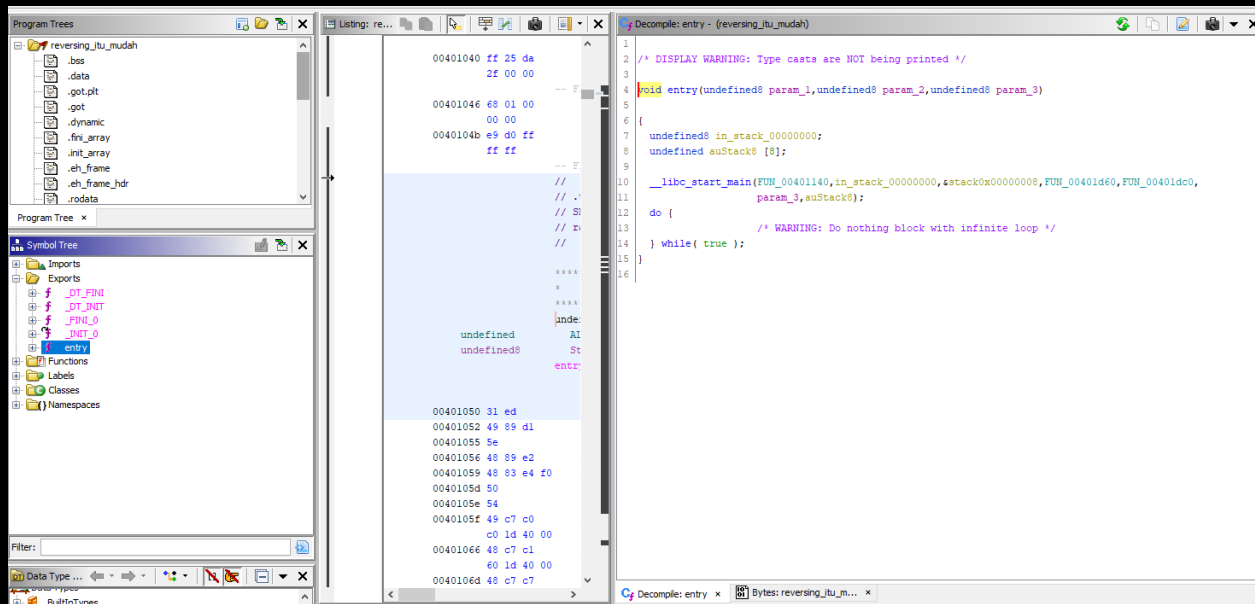
Kita diberikan sebuah binary file elf 64 bit stripped sebagai berikut

```
Permissions Size User Date Modified Name
.rwxrwxrwx 36 gawrgare 30 Oct 18:29 flag.txt
.rwxrwxrwx 14k gawrgare 29 Oct 09:44 reversing_itu_mudah
> file reversing_itu_mudah
reversing_itu_mudah: ELF 64-bit LSB executable, x86-64, version 1 (SYSV), dynamically linked, interpreter /lib64/ld-linux-x86-64.so.2, for GNU/Linux 3.2.0, BuildID[sha1]=fb53cf74b3f119b9d8ff345ae7fde4b43680a01c, stripped
```

Ketika dijalankan program meminta sebuah argumen yang nantinya akan digunakan sebagai pengecekan flag

```
> ./reversing_itu_mudah
Usage: ./reversing_itu_mudah flag
> ./reversing_itu_mudah test
Sorry, wrong flag
```

Langsung saja kami buka file tersebut dengan ghidra binary tersebut untuk dilihat dan dieksekusi lebih lanjut dan dengan mudah kami menemukan fungsi main dari binary



Dapat dilihat bahwa main memiliki beberapa fungsi yang dieksekusi ketika dijalankan


```

undefined4 main(int argc,undefined8 *argv)
{
    size_t sVar1;
    uint local_20;
    int local_1c;

    if (argc == 2) {
        FUN_004014a0(argv[1],0);
        FUN_004014e0(argv[1],1);
        FUN_00401520(argv[1],2);
        FUN_00401560(argv[1],3);
        FUN_004015a0(argv[1],4);
        FUN_004015e0(argv[1],5);
        FUN_00401620(argv[1],6);
        FUN_00401660(argv[1],7);
        FUN_004016a0(argv[1],8);
        FUN_004016e0(argv[1],9);
        FUN_00401720(argv[1],10);
        FUN_00401760(argv[1],0xb);
        FUN_004017a0(argv[1],0xc);
        FUN_004017e0(argv[1],0xd);
        FUN_00401820(argv[1],0xe);
        FUN_00401860(argv[1],0xf);
        FUN_004018a0(argv[1],0x10);
        FUN_004018e0(argv[1],0x11);
        FUN_00401920(argv[1],0x12);
        FUN_00401960(argv[1],0x13);
        FUN_004019a0(argv[1],0x14);
        FUN_004019e0(argv[1],0x15);
        FUN_00401a20(argv[1],0x16);
        FUN_00401a60(argv[1],0x17);
        FUN_00401aa0(argv[1],0x18);
        FUN_00401ae0(argv[1],0x19);
        FUN_00401b20(argv[1],0x1a);
        FUN_00401b60(argv[1],0x1b);
        FUN_00401ba0(argv[1],0x1c);
        FUN_00401be0(argv[1],0x1d);
        FUN_00401c20(argv[1],0x1e);
        FUN_00401c60(argv[1],0x1f);
        FUN_00401ca0(argv[1],0x20);
        FUN_00401ce0(argv[1],0x21);
        FUN_00401d20(argv[1],0x22);
        local_20 = 0;
    }
}

```

```

    checker = 0;
    for (i = 0; i < 0x23; i = i + 1) {
        checker = *(&DAT_00404050 + i * 4) | checker;
    }
    sVar1 = strlen(argv[1]);
    if (sVar1 != 0x23) {
        checker = 1;
    }
    if (checker == 0) {
        printf("Congratulations, the flag is: %s\n",argv[1]);
    }
    else {
        printf("Sorry, wrong flag\n");
    }
}
else {
    printf("Usage: %s flag\n",*argv);
}
return 0;
}

```

Dapat dilihat bahwa agar flagnya muncul maka nilai checker haruslah bernilai 1. Dan nilai checker didapat dengan melakukan bitwise or nilai 0 dengan nilai yang ada pada memory tersebut. Lalu mulailah kami melakukan pengecekan untuk setiap fungsi yang dijalankan

```

1
2 /* DISPLAY WARNING: Type casts are NOT being printed */
3
4 void FUN_004014a0(long param_1,int param_2)
5
6 {
7     *(&DAT_00404050 + param_2 * 4) = *(param_1 + param_2) != 'G';
8     return;
9 }
10

```

```

void FUN_004014e0(long param_1,int param_2)

{
    *(&DAT_00404050 + param_2 * 4) = *(param_1 + param_2) != 'e';
    return;
}

```

```
void FUN_00401520(long param_1,int param_2)
{
    *(&DAT_00404050 + param_2 * 4) = *(param_1 + param_2) != 'm';
    return;
}
```

Ketiga fungsi yang dijalankan pertama akan melakukan pengecekan apakah huruf dari index ke i sama dengan huruf yang akan di check dan curiga bahwa setiap fungsi membentuk sebuah kata yang merupakan flag maka kami catat setiap huruf yang di check pada fungsi tersebut dan didapat flag

```
> ./reversing_itu_mudah Gemastik2022{st45iUn_MLG_k07a_b4rU}
Congratulations, the flag is: Gemastik2022{st45iUn_MLG_k07a_b4rU}
```

Flag: Gemastik2022{st45iUn_MLG_k07a_b4rU}

Dino

500






Beat my highscore!

author - vidner#6838

Diberikan sebuah jar file dan highscore.txt yang berisi

```
2147482310 21cb61a
```

Lalu tanpa berpikir panjang kami pun melakukan decompilasi file Dino.jar tersebut dengan online decompiler dan mendapat empat buah file .java

	ByteArrayToImage.java	U
	Cactus.java	U
	Cloud.java	U
	dino.jar	U
	DinoGame.java	U

Pada dino game.java terdapat fungsi ls Yang berfungsi untuk membaca highscore.txt dan mencocokkan dengan hasil crc yang terpisah oleh spasi. Probset meminta untuk kita mematikan program tersebut

```
private int ls() {
    this.gf();
    try {
        final BufferedReader bufferedReader = new BufferedReader(new FileReader(fileName: "highscore.txt"));
        final String line = bufferedReader.readLine();
        bufferedReader.close();
        final String[] split = line.split(regex: " ");
        final int int1 = Integer.parseInt(split[0]);
        this.csss = split[1];
        final int rcr = this.rcr(int1);
        if (!Integer.toHexString(rcr).equals(this.csss)) {
            throw new Error(message: "Invalid checksum");
        }
        this.ssss = this.rcr(this.rcr(rcr) ^ int1);
        return int1;
    }
    catch (Exception ex) {
        System.out.println(x: "Error loading highscore");
        System.exit(status: 0);
        return 0;
    }
}
```

```

private int rcr(final int n) {
    int n2 = -1;
    for (int i = 0; i < 4; ++i) {
        n2 ^= n >> i * 8;
        for (int j = 0; j < 8; ++j) {
            if ((n2 & 0x1) == 0x1) {
                n2 = (n2 >> 1 ^ 0xEDB88320);
            }
            else {
                n2 >>= 1;
            }
        }
    }
    return n2;
}

```

Lalu kami mencoba untuk merubah highscore.txt sekaligus mencocokkan crc nya. Kami mengeset highscore dengan 23 sekaligus crc nya

```

57546378
penyisihan/rev/dino
Python 3.8.2 (tags/
Type "help", "copyr
>>> hex(57546378)
'0x36e168a'
>>>

```

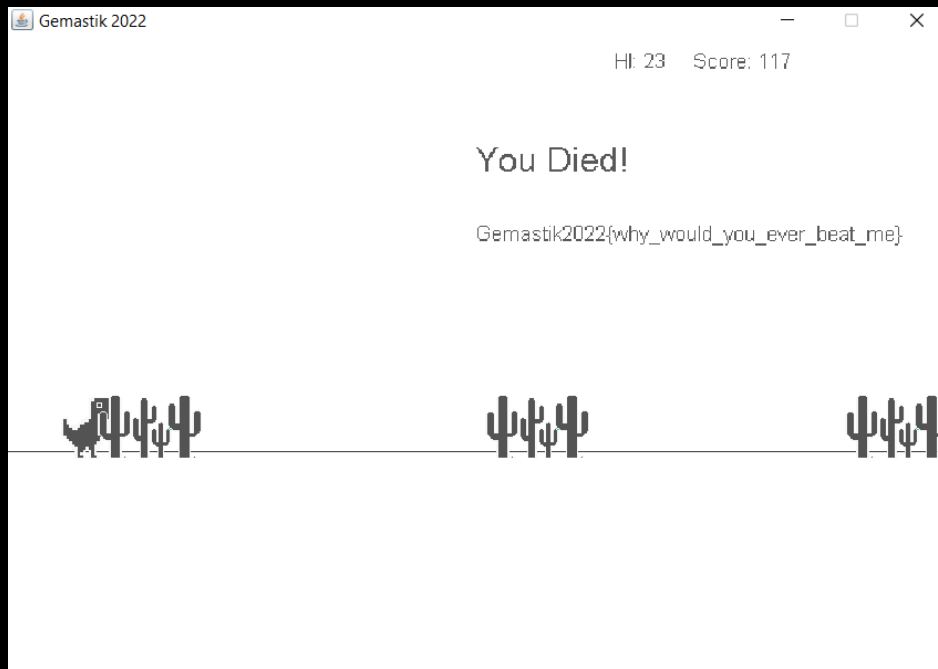
highscore.txt - Notepad

File Edit View

|23 36e168a

Lalu kami pun menjalankan game nya dan ketika melebihi 23 dan flag pun

didapatkan



Flag: Gemastik2022{why_would_you_ever_beat_me}

Rubyte

820

Hope you find the hidden gem!

author - vidner#6838

```

== disasm: #<ISeq:<compiled>@<compiled>:1 (1,0)-(1,99)> (catch: FALSE)
0000 putself ( 1)[Li]
0001 opt_getinlinecache 8, <is:0>
0004 getconstant :File
0006 opt_setinlinecache <is:0>
0008 putstring "flag"
0010 opt_send_without_block <callinfo!mid:read, argc:1, ARGS_SIMPLE>, <callcache>
0013 putstring "H*"
0015 opt_send_without_block <callinfo!mid:unpack, argc:1, ARGS_SIMPLE>, <callcache>
0018 putobject_INT2FIX_0_
0019 opt_aref <callinfo!mid:[], argc:1, ARGS_SIMPLE>, <callcache>
0022 putobject 16
0024 opt_send_without_block <callinfo!mid:to_i, argc:1, ARGS_SIMPLE>, <callcache>
0027 opt_getinlinecache 34, <is:1>
0030 getconstant :File
0032 opt_setinlinecache <is:1>
0034 putstring "flag"
0036 opt_send_without_block <callinfo!mid:read, argc:1, ARGS_SIMPLE>, <callcache>
0039 putstring "H*"
0041 opt_send_without_block <callinfo!mid:unpack, argc:1, ARGS_SIMPLE>, <callcache>
0044 putobject_INT2FIX_0_
0045 opt_aref <callinfo!mid:[], argc:1, ARGS_SIMPLE>, <callcache>
0048 putobject 16
0050 opt_send_without_block <callinfo!mid:to_i, argc:1, ARGS_SIMPLE>, <callcache>
0053 putobject_INT2FIX_1_
0054 opt_send_without_block <callinfo!mid:>, argc:1, ARGS_SIMPLE>, <callcache>
0057 opt_send_without_block <callinfo!mid:^, argc:1, ARGS_SIMPLE>, <callcache>
0060 opt_send_without_block <callinfo!mid:puts, argc:1, FCALL|ARGS_SIMPLE>, <callcache>
0063 leave

```

Pada byte.txt, kita lihat bahwa dilakukan load flag sebanyak dua kali (line 4-24 dan line 30-50), dimana flag yang dibaca sudah dikonversi menjadi integer (pertama menjadi hex, kemudian menjadi integer). Setelah itu, flag kedua di-right shift sebanyak satu (line 53-54), kemudian di-xor dengan flag pertama.

Maka, kita cukup me-reverse hasil right-shift dan xor. Hal tersebut dapat dilakukan sebagai berikut:

Misalnya flag awal (per bit) adalah $f[0], f[1], \dots, f[n]$. Setelah right shift dan xor, ciphertext akan menjadi $f[0], f[0]^f[1], f[1]^f[2], \dots, f[n-1]^f[n]$. Maka, untuk mendecrypt ciphertext ($c[0], \dots, c[n]$), kita lakukan $\text{flag} = (c[0], c[0]^c[1], c[1]^c[2], \dots, c[n-1]^c[n])$. Implementasi algoritma tersebut dilakukan sebagai berikut:

```

from Crypto.Util.number import *
ct =
215399763437993922857257938507183571899033473988099831289577921701237839559370177
126393638370659139 # x ^ (x>>1)
arr = []
temp = ct
while temp>0:
    arr.append(temp%2)
    temp//=2
arr.reverse()

```

```
for i in range(1, len(arr)):
    arr[i]^=arr[i-1]

arr.reverse()
ans = 0
for i in range(len(arr)):
    ans += pow(2, i)*arr[i]

print(long_to_bytes(ans))
```

Flag: Gemastik2022{i_still_remember_30_october}

Foren

Traffic Enjoyer

500

P balap first blood

author - deomkicer#3362

Kita diberikan file traffic.pcap. Dengan mengurutkan packet bytes berdasarkan length, kita mendapatkan sejumlah HTTP RESPONSE. Setelah mengecek packet bytesnya, kita mendapatkan file .txt yang diencode dengan base64:

```
File Data: 219 bytes
▼ Line-based text data: text/html (4 lines)
iVBORw0KGgoAAAANSUUEUgAAAGQAAABKCAIAAAD/gAIDAAAZ01EQVR4n03QAQ3A\n
IAxFQTYPdVIvaEQbIjYLIyFNRu4E/KavNQAAAAAAAAAAAAAAAAAAAAAAAA\n
gCNdu4Yiove+a+2jMcacs/joBpn51MvMyh/vymN/J9YCsRaItUAsAADgUC9meXjS\n
CVyYRAAAAABJRUSerkJggg==
```

Kita kemudian mencoba mendecode isi dari salah satu .txt tersebut:

```
Input
length: 1482
lines: 23

tra22GnRaFRo34vUxcUF8rQ5L+Z0OpHnqdVq1bRq1SSTSfbc5ubmxKVhGPb19cUO
HBoakrZsPsqkjTObzdxV1ezxy8tLcYHZbPbx8ZE9Xl9fLy7wJ8q1jautrUWOPz8/
i870eDw4jv826PP5RACwi460DuQths+7VfGT+DLU6/XI8Y+PD2k/SBUSN6u0QbME
gGYJAM0SAJolADRLAGiWANAsAaBZAKCzBIBmCSBxsZ4/P5HjBdbE/iASf0Xz/v6O
HMdx/OnpSVzmwMBATU3Nb4N+vz8cDosLLBZmsxn5FU1XV5fozIeHB3agy+W5sGye
JL4MX19fU6kUe9zhcIgLNBgMFouFPR6NRSUF/oT0N/hAImAeFP2V+eDgIHJlWEY
cYHFBbkUls/n29vbRaTt7++z02KxwIkshXETsp6enpaVCTuRu7u7kftwdnZ2ZCpe
aQWw71dXV/nn1NXVhcNhZM7w8LB89SutwMaQ9fV1Pu9cLS0tNzc3yASGYUpnY4hG
o9HpdMFgKktf9/f3brfbYDAgj7VYLCsrK+l0muvwiYkjhafzi1y3yc70Tq/Xy2cz
Wzwe/7WZraenp62trcCt7fDwcGxsTJ6SVTU5OSntNkmapPHL3SVidnZWqg24DMMg
305Lyujo6MvLyw87dXJywv73sDQ1Njbu7e2JuyTj8fj09HSJvILyR5Lk5uYm/5+j
0DS9sLBQWVmpduH/UuGHTv39/Q6H4/sPncrLy7PZbDKZjEQiDMP4fL6zs7NQQKRw
bQAAAAAAAAAAAAAgP+rVwHI5LpMo21vyGAAAABJRu5ErKJggg==

Output
time: 1ms
length: 1093
lines: 6

.PNG
.
...
IHDR...d...d....ÿ.....IDATx.i0ËK2k..pí1.0´.jrã&(èâDa..
.0
"Z[(.0&è..çè/hs.eñ""hñe0"0$çd3.1.-P´.2´òs..b.ï<ã..æâëù}..îxßócnù<j4.....b´
```

Sehingga kita mengetahui bahwa setiap file .txt merupakan sebuah file .png yang diencode dengan base64. Kita kemudian menggunakan online tool <https://base64.guru/converter/decode/image/png> untuk mendecode semua file .txt. Kita mendapatkan bahwa setiap file .txt mewakili sebuah huruf pada flag dengan index yang bersesuaian dengan index yang di-GET.

Base64*

```
iVBORw0KGgoAAAANSUhEUgAAAGQAAABkCAIAAAD/gAIDAAAEDE1EQVR4n03by0sy  
axwHcM1sBKm0iWpy4yYo60JEYZCLCoIwCiJaWygE1SboHwii6C9oU6260SIiaNF1  
0yLbJKJkMx05CKxQtBQytNJzFgfeE+8845mZ5uLr+X2WD83X3/Njbvk8ajQAAAAA  
AAAAAAAAAAAAAAAAAAAAAAAAAAdetAp/HoZhFX19vb29JE1arVaCIIXGo16v  
z2QyqVQqEonQNH19fX1+fk5R1MK1FRGbzbxsfH29vYXP7e3t/Pz80ajUe3C1UUQ  
x07uhi6f59mm72KxmNvtVnsGShk7GIIkkFila9N3x8bH17E17Kikhm7n15XT/7N0/
```

Decode Base64 to PNG

Preview PNG Image | [Toggle Background Color](#)



Kita kemudian mengurutkan semua karakter yang didapatkan dan mendapatkan:

Flag : Gemastik2022{balapan_f1rst_blood_is_real_f580c176}

Har
500

Har Har Har!

author - vidner#6838

Diberikan sebuah har file yang di zip. Kami langsung membuka file tersebut di chrome dev tools untuk melakukan analysis terhadap request yang dijalankan.

Kami mendownload file tersebut dan membukanya di figma

Untitled?node-id=0%3A1	200	document	Other
Inter-Regular.woff?v=3.10	200	font	www.figma.com/file/N...
Inter-Italic.woff?v=3.10	200	font	www.figma.com/file/N...
Inter-Medium.woff?v=3.10	200	font	www.figma.com/file/N...
Inter-MediumItalic.woff?v=3.10	200	font	www.figma.com/file/N...
Inter-SemiBold.woff?v=3.10	200	font	www.figma.com/file/N...
Inter-SemiBoldItalic.woff?v=3.10	200	font	www.figma.com/file/N...
figma_app.min.js.br	200	script	www.figma.com/file/N...
DSEG7Classic-Italic-Custom2.woff2	304	font	www.figma.com/file/N...
state?file_key=N2D2B1mcWjiqgKZciiPSJ5	200	xhr	www.figma.com/file/N...
state	200	xhr	www.figma.com/file/N...
compiled_wasm.wasm.br	200	wasm	www.figma.com/file/N...
compiled_wasm.js.br	200	script	www.figma.com/file/N...
Whyte-Regular.woff	200	font	www.figma.com/esbui...
Whyte-Bold.woff	200	font	www.figma.com/esbui...
N2D2B1mcWjiqgKZciiPSJ5?role=editor&trac...	101	websocket	www.figma.com/file/N...
page	200	xhr	www.figma.com/esbui...
page	200	xhr	www.figma.com/esbui...
?sentry_key=d1b12a8fbc424e4b956eb33ca...	200	fetch	www.figma.com/esbui...
page_load	200	xhr	www.figma.com/esbui...
font-files?freetype_minimum_api_version=20	Finished	xhr	www.figma.com/esbui...
font-files?freetype_minimum_api_version=20	Finished	xhr	www.figma.com/esbui...
index_b424c639f9493264c6b715bcd20020b...	200	xhr	www.figma.com/esbui...
N2D2B1mcWjiqgKZciiPSJ5	200	xhr	www.figma.com/esbui...
monitor	200	xhr	www.figma.com/esbui...
import.shim.js.br	200	script	www.figma.com/esbui...
track	200	xhr	www.figma.com/esbui...
monitor	200	xhr	www.figma.com/esbui...
N2D2B1mcWjiqgKZciiPSJ5	200	xhr	www.figma.com/esbui...
track	200	xhr	www.figma.com/esbui...
monitor	200	xhr	www.figma.com/esbui...
N2D2B1mcWjiqgKZciiPSJ5	200	xhr	www.figma.com/esbui...
realtime_token	200	xhr	www.figma.com/esbui...
monitor	200	xhr	www.figma.com/esbui...
track	200	xhr	www.figma.com/esbui...

clearbit.js	302
published_and_moved_components?fv=20	200
monitor	200
track	200
monitor	200
track	200
monitor	200
track	200
monitor	200
track	200
monitor	200
track	200
performance.fullscreen.join_time	200
livegraph-next	101
track	200
monitor	200
monitor	200
track	200
insight.old.min.js	200
• adsct?bci=3&eci=2&event_id=729b965c-64...	200
• adsct?bci=3&eci=2&event_id=729b965c-64...	200
?random=1667116666676&cv=9&fst=1667...	200
?random=1667116666677&cv=9&fst=1667...	200
871885529854177?v=2.9.89&r=stable	200
monitor	200
track	200
• collect?v=2&fmt=js&pid=1644642&time=1...	302
?random=767018113&cv=9&fst=16671166...	302
?id=871885529854177&ev=PageView&dl=...	200
tags.js?reveal=false&reveal_async=false&tra...	200
• ?random=1667116666676&cv=9&fst=1667...	200
• ?random=1667116666676&cv=9&fst=1667...	200
destinations.min.js	200
tracking.min.js	200
p	200
?random=767018113&cv=9&fst=16671166...	302
identify.js	200

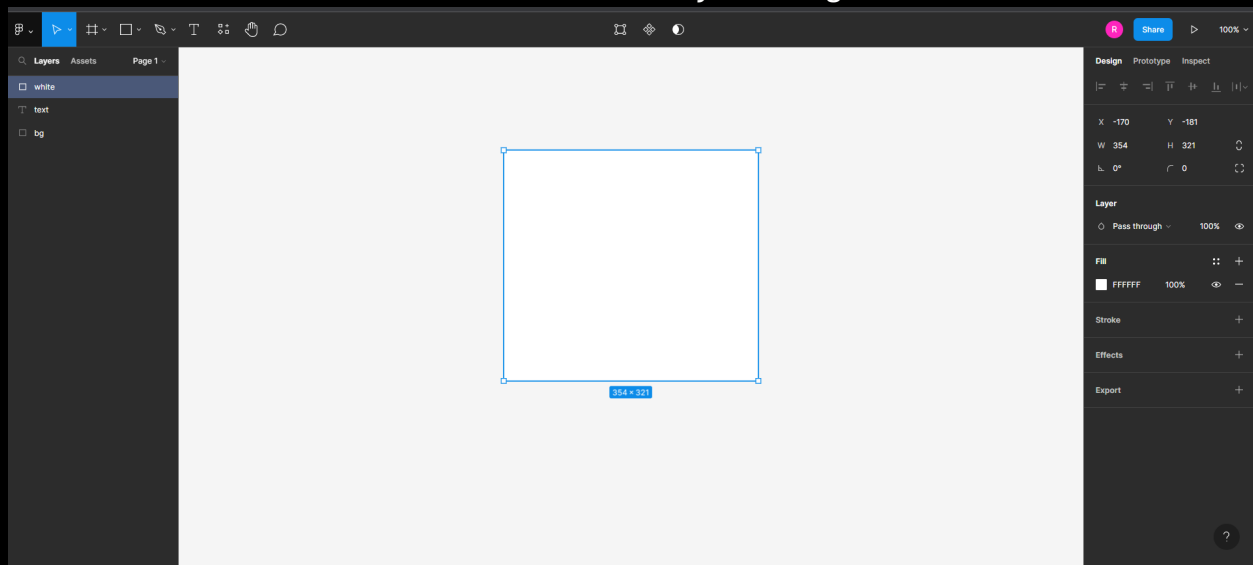
Dapat dilihat bahwa terdapat banyak request yang terjadi dan kami melakukan analysis untuk setiap request. Dan menemukan beberapa hal yang menarik.

```
i18n: null
▼ meta: {is_default: null, script: "cef6a458c5668f8c7570d55230ad629630428847", name: "Untitled",-}
  canvas_url: "https://s3-alpha-sig.figma.com/checkpoints/pUE/y5i/cFaNjWYebNG1JmUi/3qnr9v3zrywpThYIfXxA4.fiq?Expires=1667779200&Signature=URzyL8YhplY9wKiGPZBEIrhomda~LAAoem5a9SVtu7LNvem2iswOXca9gZr69rSQ
  current_team_user: null
  feature_flags: {disable_registerframecallback: true, survey_pro_cart_abandon: true, widgets_paste_as_widget: true,-}
  activerecord_cache: true
  activerecord_cache_livelaunch: true
```

Kami mendapatkan sebuah canvas_url =

https://s3-alpha-sig.figma.com/checkpoints/pUE/y5i/cFaNjWYebNG1JmUi/3qnr9v3zrywpThYIfXxA4.fiq?Expires=1667779200&Signature=URzyL8YhplY9wKiGPZBEIrhomda~LAAoem5a9SVtu7LNvem2iswOXca9gZr69rSQu4LjLpPSPCrDT-kdYRBqY5p4jnKxjad5nwiG6KbbLjZrzTmStHIMbg0wJLf0cd3w77UwL2fnNuUvL6MKYE02I3qGH50M0YDrj0IrtYfhi471o26v~qDLB7pdrZn9ycRTKZbLGXtJyJ1Iq90nq0i5yx4i6NxkrTq4K5kxjZGL8VUzUXKBdXbQZANGLpsuEA04aALfqS5vCpko870yTJUxR5b0LAP1Y0jTWMMoHoLM3eUyPNy2y~e77Wduk2o~vcMJzSKxoR0k7xX1sWKLhagspg_&Key-Pair-Id=APKAINTVSGUEWH5XD5UA

Kami mendownload file tersebut dan membukanya di figma:



Terdapat sebuah kotak putih. Dengan menggeser kotak putih tersebut, kita mendapatkan flag.

Flag : Gemastik2022{kinda_wish_this_werent_text}