WRITE UP ARA-ITS 2023

~ by Heker 1MISSU ~

TEAMS :

IndianaJones

BayzLightyear

UkiyoAgusta

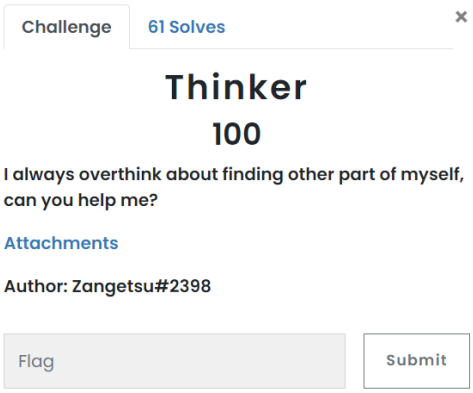
CTF COMPETITION: *EXIST

ASENG:

TABLE OF CONTENTS

FORENSIC

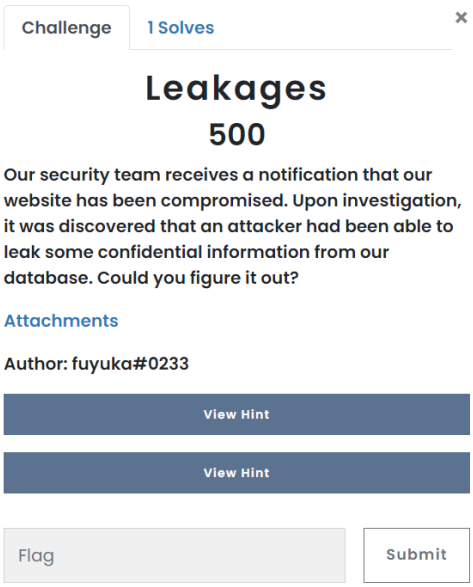
Thinker



Jadi pada chall ini, diberikan sebuah gambar. Seperti biasa kita cek dengan command file, exiftool, strings, binwalk dll dlu sebagai langkah awal. Setelah di binwalk ternyata di dalam gambar tersebut terdapat banyak file zip. Lalu saya mencoba mengekstrak filenya, didalamnya terdapat file zip lagi, unzip lagi, ada pecahan flag yang di encode ke base64 dan file zip lagi, unzip lagi dan didapat pecahan flag dalam bentuk hex dan lagi lagi ada file zip, unzip lagi dan ada pecahan flag dalam bentuk biner dan file zip lagii, di unzip lagi sampai AKHIRNYA ada file png yang headernya rusak, jadi kami mencoba memperbaiki 16 bytes pertama di headernya dan berhasil. Ternyata gambarnya berisi pecahan flag terakhir dalam bentuk ascii decimal

Flag: ARA2023{5!mpl3_C0rrupt3d_1m4ge5}

Leakages



Jadi pada chall ini diberikan logs sebuah website yang habis dieksploitasi menggunakan sqli, jadi berdasarkan hint di soal, aplikasi web ini memiliki bug pada file upload form yang coba diexploitasi oleh attacker. Si attacker berusaha untuk me-leak informasi dengan menggunakan query yang membrute karakter satu persatu, dan sebagai indikator benar atau salahnya menggunakan nama file yang akan di reflect ke user.

```
PK<0x03><0x04><0x14><0x00><0x00><0x00><0x00><0x00><0x19>]8V<0x00><0x00><0x00><0x00><0x00><0x00><0x00><0x00><0x00><0x00><0x00>Đ<0x00>
<0x00><0x00>'||(selEct caSE wHEn hex(SUbStr((SELECt FIleNAMe fRoM USER_stoRAgES LIMIT 0,1), 8, 1)) likE '67' tHEn x'6e764a6867645846' ElSE
x'6c564f6357494766' END),(seLEct pAThnAME FrOM UseR_sToRagEs LImIT 0,1),322) -- -PK<0x03><0x04><0x14><0x00><0x00><0x00><0x00><0x00><0x19>]8V
<0x00><0x00><0x00><0x00><0x00><0x00><0x00><0x00><0x00><0x00><0x00><0x00>Đ<0x00><0x00><0x00>'||(sElecT casE WHeN heX(SUBStR((SeLEcT FILenAmE
FrOM UsEr_StORAGEs lIMit 0,1), 8, 1)) LIKe '43' tHeN x'476e6C4b595a6E70' ElsE X'70505043614A6e51' end),(selEcT pAThNaME FROm uSer_STOrAges
limIT 0,1),322) -- -PK<0x03><0x04><0x14><0x00><0x00><0x00><0x00><0x00><0x19>]8V<0x00><0x00><0x00><0x00><0x00><0x00><0x00><0x00><0x00>
<0x00><0x00>Đ<0x00><0x00><0x00>'||(sELEcT cAsE WHEN hEx(suBstR((SELEcT FiLEnAME frOM usER_sTorages limit 0,1), 8, 1)) like '4D' ThEn
X'6772676e48416A4D' elSE x'6E585A6177595957' EnD),(SElEcT pAtHNAMe FrOM USER_stORAGes LIMit 0,1),322) -- -PK<0x03><0x04><0x14><0x00><0x00>
<0x00><0x00><0x00><0x19>]8V<0x00><0x00><0x00><0x00><0x00><0x00><0x00><0x00><0x00><0x00><0x00><0x00>Đ<0x00><0x00><0x00>'||(SELect cASE WHeN
HEx(sUBStr((sElECt fiLENAme from USEr_sTOrAgeS LiMIt 0,1), 8, 1)) lIKe '33' THen x'765178786F4f4E46' eLsE X'4B56597346454458' enD),(SeLect
PATHNamE FROM useR_stoRages liMIt 0,1),322) -- -PK<0x03><0x04><0x14><0x00><0x00><0x00><0x00><0x00><0x19>]8V<0x00><0x00><0x00><0x00><0x00>
<0x00><0x00><0x00><0x00><0x00><0x00><0x00><0x00>Đ<0x00><0x00><0x00>'||(SelecT CASe WhEN HEx(SUbstR((SElect fileNaMe frOM user_STORaGeS LImIt 0,1),
8, 1)) LIke '61' theN X'72465A6a4A45504A' Else X'5464666656B616176' End),(sELECT pATHNamE fRom USEr_StOraGEs lIMIt 0,1),322) -- -PK<0x01>
<0x02><0x14><0x03><0x14><0x00><0x00><0x00><0x00><0x00><0x19>]8V<0x00><0x00><0x00><0x00><0x00><0x00><0x00><0x00><0x00><0x00><0x00>Đ
<0x00><0x00><0x00><0x00><0x00><0x00><0x00><0x00><0x00><0x00><0x00><0x00>¤<0x81><0x00><0x00><0x00><0x00>'||(selEct caSE wHEn hex(SUbStr((SELECt
FIleNAMe fRoM USER_stoRAgES LIMIT 0,1), 8, 1)) likE '67' tHEn x'6e764a6867645846' ElSE x'6c564f6357494766' END),(seLEct pAThnAME FrOM
UseR_sToRagEs LImIT 0,1),322) -- -PK<0x01><0x02><0x14><0x03><0x14><0x00><0x00><0x00><0x00><0x00><0x19>]8V<0x00><0x00><0x00><0x00><0x00><0x00>
<0x00><0x00><0x00><0x00><0x00><0x00><0x00>Đ<0x00><0x00><0x00><0x00><0x00><0x00><0x00><0x00><0x00><0x00><0x00>¤<0x81>î<0x00><0x00><0x00>'||(sElecT
casE WHeN heX(SUBStR((SeLEcT FILenAmE FrOM UsEr_StORAGEs lIMit 0,1), 8, 1)) LIKe '43' tHeN x'476e6C4b595a6E70' ElsE X'70505043614A6e51'
end),(selEcT pAThNaME FROm uSer_STOrAges limIT 0,1),322) -- -PK<0x01><0x02><0x14><0x03><0x14><0x00><0x00><0x00><0x00><0x00><0x19>]8V<0x00>
<0x00><0x00><0x00><0x00><0x00><0x00><0x00><0x00><0x00><0x00>Đ<0x00><0x00><0x00><0x00><0x00><0x00><0x00><0x00><0x00><0x00><0x00>¤<0x81>Ü
<0x01><0x00><0x00>'||(sELEcT cAsE WHEN hEx(suBstR((SELEcT FiLEnAME frOM usER_sTorages limit 0,1), 8, 1)) like '4D' ThEn X'6772676e48416A4D'
elSE x'6E585A6177595957' EnD),(SElEcT pAtHNAMe FrOM USER_stORAGes LIMit 0,1),322) -- -PK<0x01><0x02><0x14><0x03><0x14><0x00><0x00><0x00>
<0x00><0x00><0x00><0x19>]8V<0x00><0x00><0x00><0x00><0x00><0x00><0x00><0x00><0x00><0x00><0x00>Đ<0x00><0x00><0x00><0x00><0x00><0x00><0x00>
<0x00><0x00><0x00><0x00><0x00>¤<0x81>É<0x02><0x00><0x00>'||(SELect cASE WHeN HEx(sUBStr((sElECt fiLENAme from USEr_sTOrAgeS LiMIt 0,1), 8, 1))
lIKe '33' THen x'765178786F4f4E46' eLsE X'4B56597346454458' enD),(SeLect PATHNamE FROM useR_stoRages liMIt 0,1),322) -- -PK<0x01><0x02><0x14>
<0x03><0x14><0x00><0x00><0x00><0x00><0x00><0x00><0x19>]8V<0x00><0x00><0x00><0x00><0x00><0x00><0x00><0x00><0x00><0x00><0x00>Đ<0x00><0x00>
<0x00><0x00><0x00><0x00><0x00><0x00><0x00><0x00><0x00><0x00>¤<0x81>_<0x03><0x00><0x00>'||(SelecT CASe WhEN HEx(SUbstR((SElect fileNaMe frOM
user_STORaGeS LImIt 0,1), 8, 1)) LIke '61' theN X'72465A6a4A45504A' Else X'5464666656B616176' End),(sELECT pATHNamE fRom USEr_StOraGEs lIMIt
0,1),322) -- -PK<0x05><0x06><0x00><0x00><0x00><0x00><0x00><0x05><0x00><0x05><0x00>ö<0x04><0x00><0x00>¦<0x04><0x00><0x00><0x00><0x00>
--514891fca025e809ed09482e38dbc623--
,7:headers;464:28:4:Host,17:192.168.1.95:5000,]40:10:User-Agent,22:python-requests/2.20.0,]36:15:Accept-Encoding,13:gzip, deflate,]15:6:Accep
```

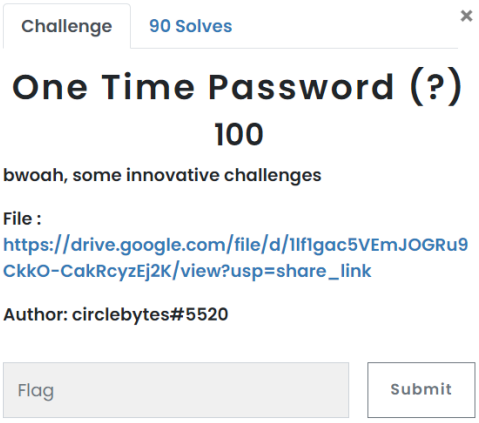
Jadi pada intinya query sql ini yang akan membrute karakter ke-n dari flag dengan membandingkan hex value dari karakter ke-n tersebut dengan sebuah hex value, dimana hasil true/false nya nanti dapat dibedakan dari nama file yang muncul pada list yang ada pada laman html dibawahnya

nah jadi nama file yang ada di list tersebut ditentukan dari hasil query yang ada pada gambar sebelumnya. Dan nama file ini akan di reflect ke user karena merupakan bagian dari laman html, dari sinilah si attacker mengetahui apakah karakter yang dicari sesuai apa tidak. Jadi Hal selanjutnya yang kita lakukan tinggal memetakan saja karakter yang dicari dengan hasil querynya yang berada pada list file. Disini kami memetakan setiap karakter yang didapat melalui querynya. Karena di dalam querynya terdapat char ke berapa yang coba untuk di retrieve. jadi tinggal disusun berdasarkan query tersebut dan didapatlah flagnya

Flag: ARA2023{r3visit1ng_4wkward_sqlite_1njection_4210f9e471}

CYRPTOGRAPHY

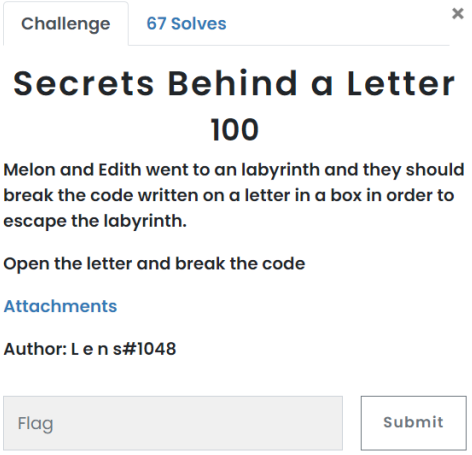
One Time Password (?)



Di chall ini diberikan file .txt yang berisi hex yang diidentifikasikan dengan A, B, dan XOR. Awalnya kami pikir untuk mendapatkan flag tersebut kami perlu melakukan kalkulasi XOR blablabla, ternyata cukup dengan mengonversi hex yang diidentifikasi dengan XOR dan dan kemudian akan mendapatkan flag yang dicari-cari.

Flag : ARA2023{th3_p_5t4nd5_f0r_p4dzz}

Secrets Behind a Letter




```
p:
125753336941212676905219718556916381441368103311882482367708803389058118
```

```
348506410486564983492781972561769555447210034136189616202231165330153281
0101344273
q:
124974834261750724658521679369605262322848918767879810806711627835614115
216758091122045736173583897427325462935027095851292058857260784924171098
67512398747
c:
360629344957317929086395350628331806510228135895355928518025722643282990
274064139273468524542176277933151448929420268869808236222401574057174997
879599430405407341221428388984827675412726778370913038246699129635727146
561394220118530281335561114050725265098398467015701334377461027276449823
4471257184433228021 8

e = 65537
```

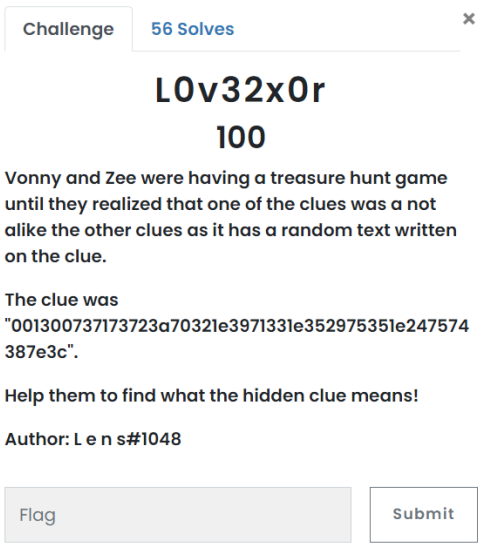
Kalau dilihat-lihat, ini, mah, namanya RSA (Rivest-Shamir-Adleman), Masbro. Bisa pakai rumus dan kode python berikut:

```
import gmpy2
p = 125753336941212676905219718556916381441368103311882482367708803389058118
8348506410486564983492781972561769555447210034136189616202231165330153281
0101344273
q = 124974834261750724658521679369605262322848918767879810806711627835614115
216758091122045736173583897427325462935027095851292058857260784924171098
67512398747
c = 360629344957317929086395350628331806510228135895355928518025722643282990
274064139273468524542176277933151448929420268869808236222401574057174997
879599430405407341221428388984827675412726778370913038246699129635727146
561394220118530281335561114050725265098398467015701334377461027276449823
4471257184433228021 8
e = 65537
n = p*q
phi_n = (p-1)*(q-1)
d = gmpy2.invert(e, phi_n)
m = pow(c, d, n)
print("Hasile, Masbro: ", m)
```

Setelah dapat hasilnya, tinggal convert dari long to bytes saja, Masbro.

Flag: ARA2023{1t_turn5_0ut_to_b3_an_rsa}

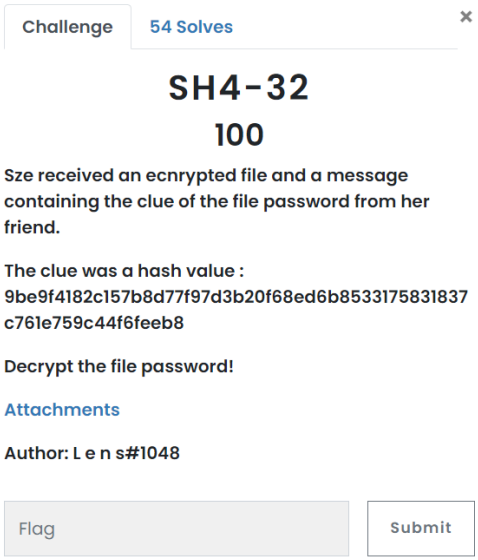
L0v32x0r



Jadi karena ini soal xor dan hanya diberi satu data, saya asumsikan chall ini brute xor, jadi karena format flagnya ARA2023, saya coba men xor 3 karakter pertama ke 3 hex value pertama pada data, ternyata kuncinya 0x41, jadi tinggal di xor aja dapet deh flagnya

Flag: ARA2023{1s_x0r_th4t_e45y?}

SH4-32

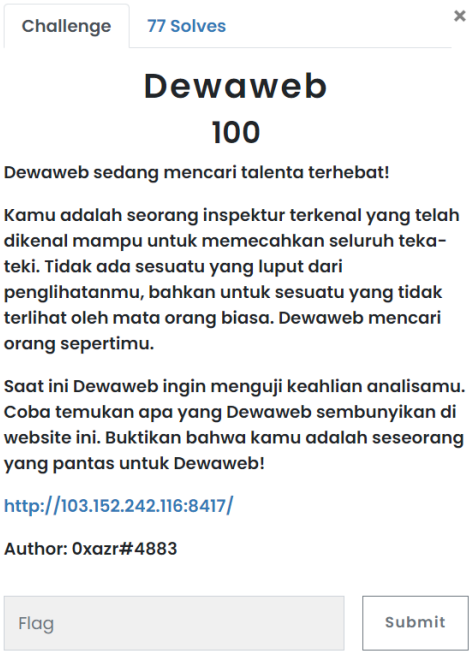


Mirip dengan chall ‘One Time Password (?)’, kami berpikir terlalu rumit padahal flagnya terdapat pada Dicitionary.txt yang merupakan attachment pada chall ini. Pada Dictionary.txt terdapat satu string yang berbeda dengan string lainnya karena berupa hex. Jadi cukup dengan mengonversi hex string tersebut kami dapat menemukan flag yang dicari-cari.

Flag : ARA2023{h4sh3d_0R_nOT_h4sh3d}

WEB EXPLOITATION

Dewaweb



Chall ini sebenarnya persis dengan chall saat Warm Up, namun bedanya ini di web dewaweb. Jadi part pertama flag ada pada comment source html.

```
part-1 : ARA2023{s4nt4I_
```

part kedua berada pada soure js di file custom.js

```
part-2 : dUlu_
```

part ketiga berada pada source css di file style.css

```
part-3 : g4k_
```

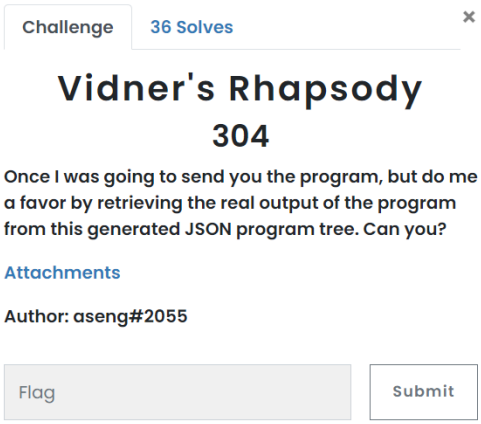
part terakhir berada pada header response http

```
X-4th-Flag: s1h?XD}
```

Flag : ARA2023{s4nt4I_dUlu_g4k_s1h?XD}

REVERSE ENGINEERING

Vidner's Rhapsody



Di chall ini, terdapat attachment berupa file .json yang berisi hasil parsing dari suatu kode program. Setelah melakukan analisa, kami mencoba menyusun program ulang berdasarkan file .json tersebut. Dan akhirnya tersusun kode program sebagai berikut :

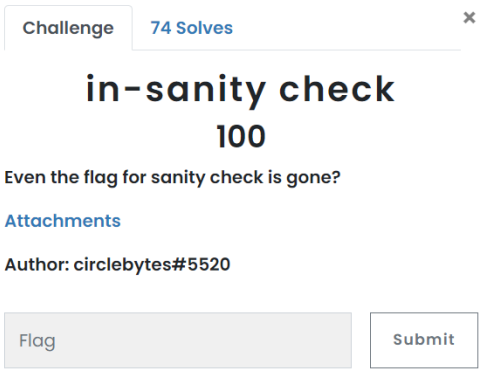
```
function mystenc(berserk, guts) {
    var s = [],
        j = 0,
        x,
        res = "";
    for (var i = 0; i < 256; i++) {
        s[i] = i;
    }
    for (i = 0; i < 256; i++) {
        j = (j + s[i] + berserk.charCodeAt(i % berserk.length)) % 256;
        x = s[i];
        s[i] = s[j];
        s[j] = x;
    }
    i = 0;
    j = 0;
    for (var y = 0; y < guts.length; y++) {
        i = (i + 1) % 256;
        j = (j + s[i]) % 256;
```

```
        x = s[i];
        s[i] = s[j];
        s[j] = x;
        res += String.fromCharCode(guts[y] ^ s[(s[i] + s[j]) % 256]);
    }
    console.log(res);
}
var berserk = "achenk";
var strenk = [244, 56, 117, 247, 61, 16, 3, 64, 107, 57, 131, 13, 137, 113,
214, 238, 178, 199, 4, 115, 235, 139, 201, 22, 164, 132, 175];
mystenc(berserk, strenk);
```

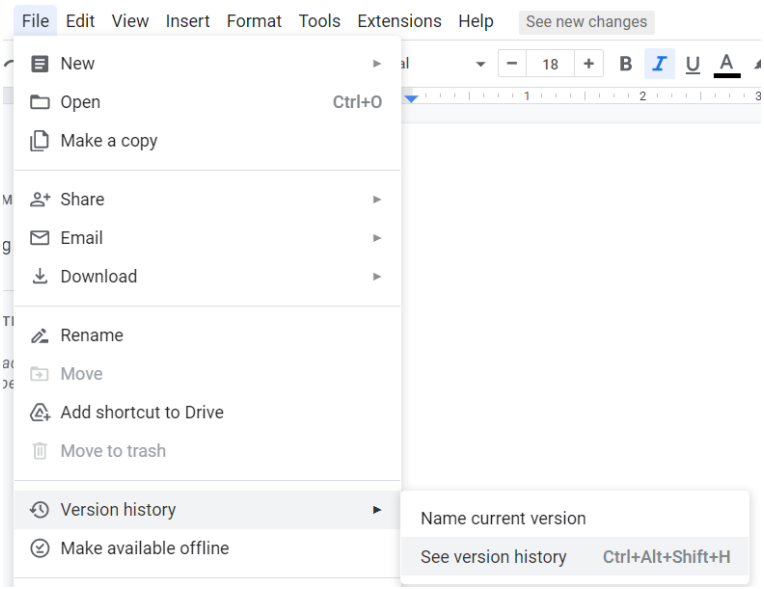
Flag : ARA2023{ j4vAST_l!ke_84831_t0wer_lol}

MISCELLANEOUS

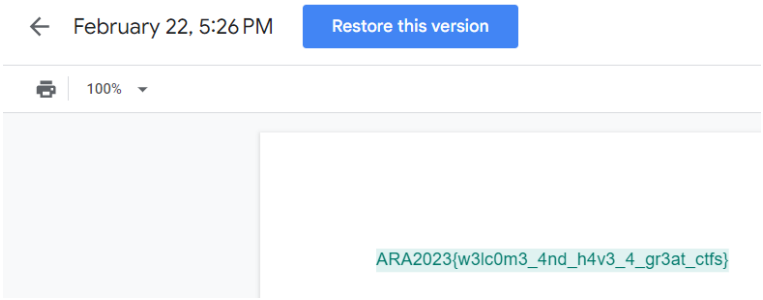
in-sanity check



Kami dapat file Google Docs yang bisa diakses dan diedit oleh siapa pun, jadi kami berinisiatif mencoba untuk cek history dari file tersebut.

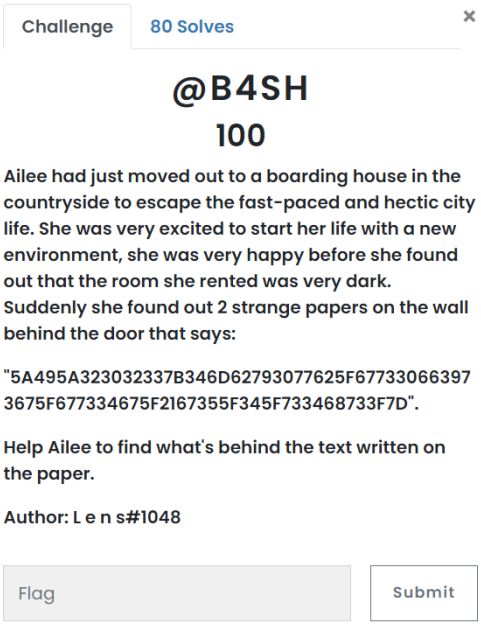


Kami coba untuk mengecek version history paling pertama dan akhirnya menemukan flag yang dicari-cari.



Flag : ARA2023{w3lc0m3_4nd_h4v3_4_gr3at_ctfs}

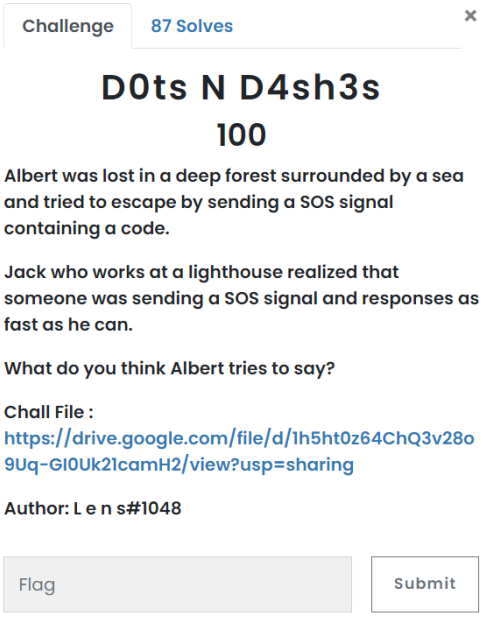
@B4SH



Pada deskripsi chall tersebut terdapat hex code yang jika didecode akan menghasilkan string ZIZ2023{4mby0wb_gs0f9sg_gs4g_!g5_4_s4hs?}. Sesuai title dari chall ini yaitu @B4SH atau atbash, jadi kami mencoba untuk melakukan decode dengan menggunakan algoritma atbash.

Flag : ARA2023{4nyb0dy_th0u9ht_th4t_!t5_4_h4sh?}

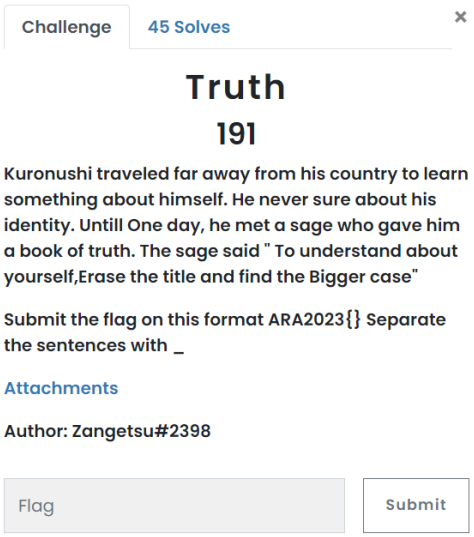
D0ts N D4sh3s



Diberikan file .txt yang berisi rangkaian sandi morse. Kami mencoba menerjemahkan sandi tersebut menggunakan morse code translator online yaitu https://morsecode.world/international/translator.html dan menghasilkan susunan bilangan biner. Susunan bilangan biner tersebut dapat dikonversi dan akan menghasilkan flag yang dicari-cari.

Flag : ARA2023{!ts_ju5t_4_m0rs3_aft312_a1!}

Truth



Pada Chall kali ini diberikan file pdf yang memiliki password, dan karena tidak diberi password kami mencoba brute dengan john the ripper menggunakan password list. Pertama kami membuat hash khusus untuk john dengan menggunakan pdf2john lalu disimpan pada file dengan nama hash

```
indianajones@bukan-heker:~/Documents/ARA/truth$ ./JohnTheRipper/run/pdf2john.pl  Truth.pdf
Truth.pdf:$pdf$4*4*128*-1060*1*16*077e10eba516a741a6285385b42f5b27*32*df507156115f50098c3d8c
6fdb1d662200000000000000000000000000000000*32*7a46addd4179a8ab90812ae8876369522d5facc72245be
4f28b3559473767d57
```

setelah itu tinggal dibrute hash tersebut dengan john the ripper menggunakan wordlist. Didapat passnya subarukun

```
Proceeding with wordlist:./JohnTheRipper/run/password.lst
Enabling duplicate candidate password suppressor
subarukun        (Truth.pdf)
1g 0:00:00:33 DONE 2/3 (2023-02-26 22:01) 0.03023g/s 53267p/s 53267c/s 53267C/s babyj90..vin
```

Unlock pdfnya, sesuai clue kita cari huruf kapitalnya saja dan buang judulnya. kami mencopy textnya ke file.txt agar mudah dibaca oleh script untuk mengambil huruf besarnya saja, berikut script singkat yang kami gunakan

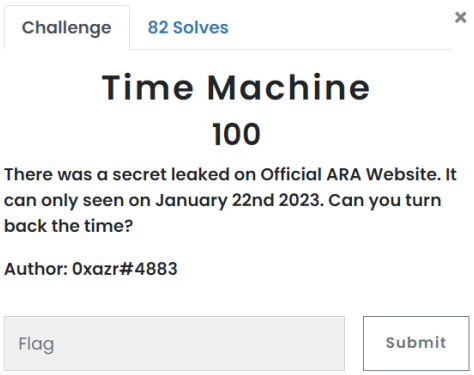
```python
#!/usr/bin/env python3
ct = open("tes.txt", "r").read()
flag = ""
for char in ct:
    if (char.isupper()):
        flag += char
print(flag)
```

Jalankan scriptnya, dan didapat huruf besarnya, pisahkan tiap kata, dan tambah format flag

Flag: ARA2023{SOUNDS_LIKE_FANDAGO}

OSINT

Time Machine

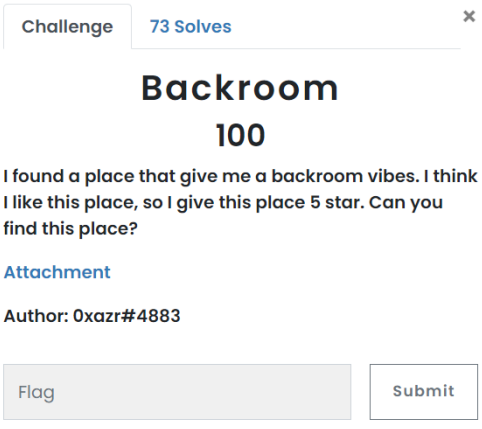


Pada chall ini kita hanya tinggal mengunjungi website official ara pada tanggal yang sudah ditentukan menggunakan wayback machine lalu view page source untuk mendapatkan flag

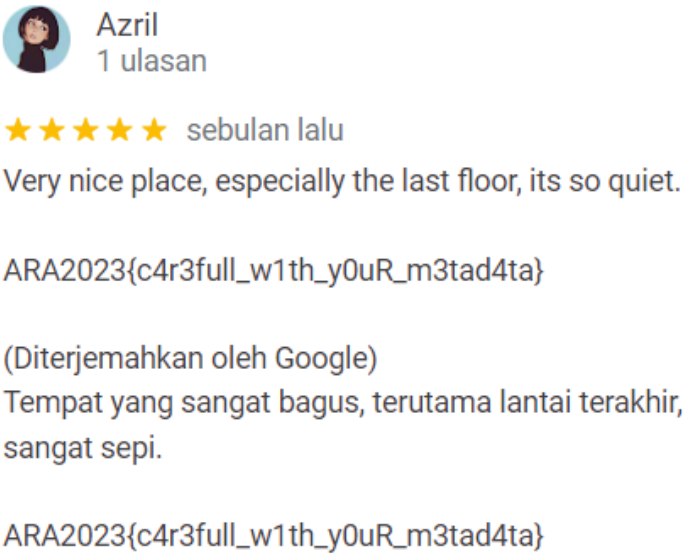
```
  </div>
</section>
<!-- ARA2023{d1gIt4l_f00tpr1nt_1s_sC4ry} -->
  </main>
```

Flag: ARA2023{d1gIt4l_f00tpr1nt_1s_sC4ry}

Backroom

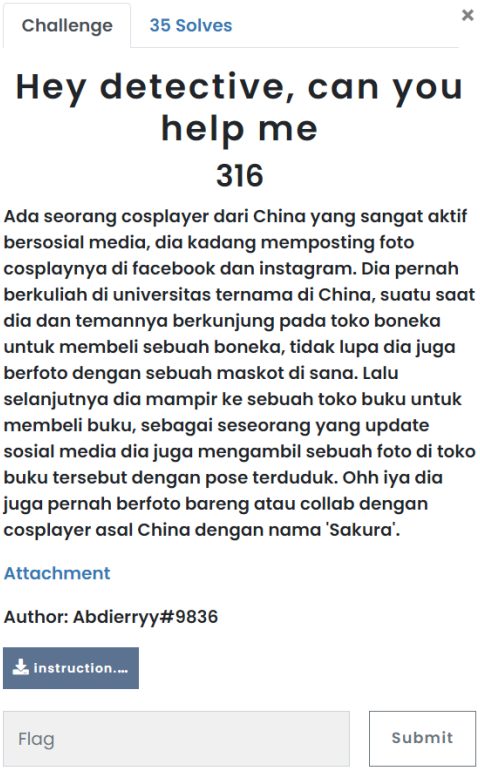


Kami menggunakan exiftool terhadap gambar tersebut, lalu menemukan sebuah koordinat -7.252769, 112.750573. Langsung saja mencari di maps dan mendapati bahwa tempat tersebut adalah Hi-Tech Mall Surabaya. Lalu menemukan review ini ....



Flag: ARA2023{c4r3full_w1th_y0uR_m3tad4ta}

Hey detective, can you help me

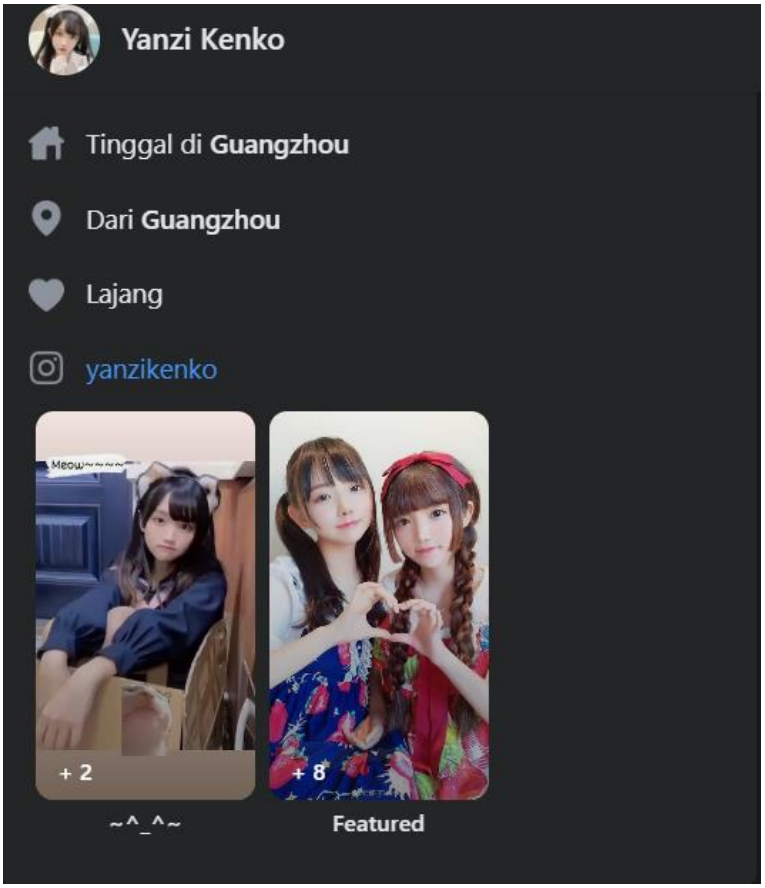


Pertama, diberikan sebuah video yang awalnya kami kira ini adalah video Sakura karena dia adalah klu satu-satunya yang kami miliki.

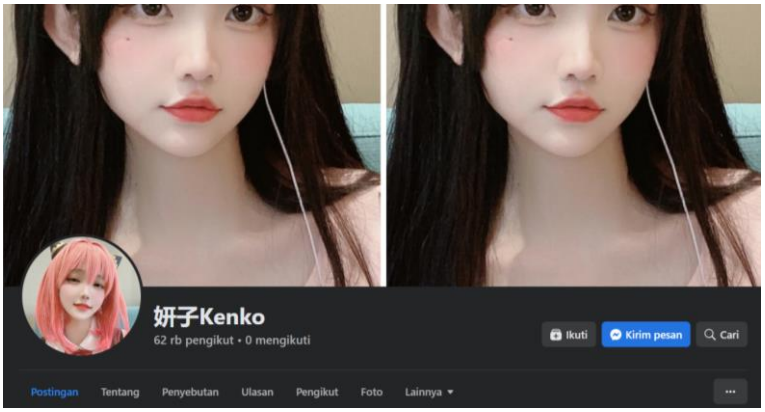
Lalu, kami mencoba mencari akun instagram seorang cosplayer asal Negeri Tirai Bambu yang bernama Sakura ini. Ujungnya, kami menemukan pengguna akun instagram @sakura.gun karena kemiripannya dengan gadis yang terdapat di video tersebut.

Kami pencari beberapa postingan yang mungkin bisa menjadi identitas si cosplayer yang dikatakan. Sampai akhirnya kami menemukan postingan ini di bagian tag.

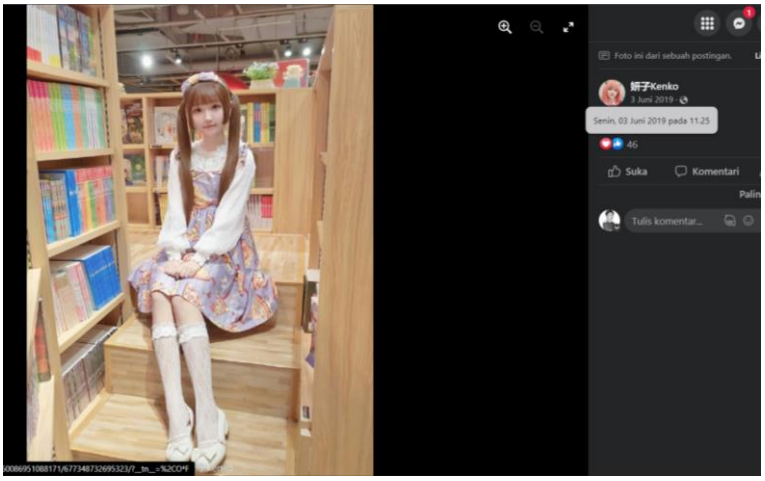
Anggota kami menganggap orang tersebut adalah pengguna dengan akun @yanzikenko karena memiliki kemiripan dengan @sakura.gun dan gadis di video. Lalu kami “menyelam” sedalam mungkin di beberapa sosial media miliknya. Sampai kami menemukan sebuah klu yang menarik pada akun yang satu ini.



Yap ..., benar sekali, video yang sama. Pernah belajar di Beijing Normal University (BNU), begitulah yang terisi di profilenya. Dan juga, lihat, akun facebook itu langsung terhubung dengan instagramnya. Tapi, akun tersebut hanya memiliki sedikit foto, jadi kami mencari lagi berbagai akun lain miliknya di facebook. Hingga akhirnya kami menemukan akun dengan lebih banyak foto.



Dan tebak, kami menemukan apa? Lebih banyak informasi.



Karena ARA CTF berlokasi di ITS, jadi kamu gunakan GMT+7, yaitun 10:25.

Yo …. Lalu kami menggeser lebih ke atas lagi ….

Kami menemukan bahwa maskot tersebut bernama Molly. Lalu, kami mencari ID akunnya, awalnya kami menggunakan ID facebook, tapi sepertinya lebih relevan menggunakan ID instagram dengan menggunakan https://followersgratis.web.id/cek-user-id-instagram/. Dan didapat id @yanzikenko adalah 44793134117. Akhirnya flag sudah terkumpul.

Flag: ARA2023{44793134117_BNU_Molly_3Juni2019-10:25_Y0u4r3ThE0s1nTm45t3R}