

REZKA NORHAFIZAH

WRITE UP LKS SMK 2020

Contents

REVERSE 1	3
REVERSE 2	4
CRACKME	5
PASS.PCAPNG	6
BONGKARZZZ	8
FILE.EXE.....	11
FLAG.JPEG.EXE.DOC.RAR	14


```

(kali@kali)-[~/Documents]
$ ./reverse1

.CTF:~ .LKS-SMK28~

CTF LKS SMK28
password:> k0opi_hitam_pht

LKSSMK28{01c9fsd3gt34zxxcb0eb8a42d3c534rf3c570703e3t}

```

Flag = LKSSMK28{01c9fsd3gt34zxxcb0eb8a42d3c534rf3c570703e3t}

REVERSE 2

Diberikan sebuah soal, kemudian saya lakukan hal yang sama seperti yang saya lakukan untuk reverse1, ternyata diminta memasukkan password. Lalu saya menggunakan tool ltrace dan didapat password :

```

(kali@kali)-[~/Documents]
$ ltrace ./reverse2
puts("||=====| CT" ... ||=====| CTF |=====|
)
= 73
puts("|| ///////////////////////////////////////////////////////////////////" ... || ///////////////////////////////////////////////////////////////////"
///////////////////////////////////////////////////////////////////"
)
= 73
puts("|| ()=====| LKS SM" ... || ()=====| LKS SMK 28
=====O||
)
= 66
puts("|| ()=====| LKS SM" ... || ()=====| LKS SMK 28
=====O||
)
= 66
puts("|| ///////////////////////////////////////////////////////////////////" ... || ///////////////////////////////////////////////////////////////////"
///////////////////////////////////////////////////////////////////"
)
= 73
puts("||=====| CT" ... ||=====| CTF |=====|
)
= 73
puts("Password:Password:
)
= 10
__isoc99_scanf(0x55d3dbf2313c, 0x7ffdd395ad90, 0, 0x7f76e0577f33.
) = 1
strcmp("0x00007fff", ".")
puts("You FailedYou Failed
)
= 11
+++ exited (status 0) +++

```

Kemudian saya coba jalankan lagi file tersebut dan saya masukkan password dan didapatkan flagnya :

```

(kali@kali)-[~/Documents]
$ ./reverse2
REVERSE 1
////////////////////////////////////////////////////////////////////
//()===== CTF =====()//
//()===== LKS SMK 28 =====()//
////////////////////////////////////////////////////////////////////
Password:
0x00007fff
You Win
LKSSMK28{LKSSMK28_486619254b9c9f6e6800cfaf77}

```

Flag = LKSSMK28{LKSSMK28_486619254b9c9f6e6800cfaf77}

CRACKME

Diberikan soal, lalu saya coba jalankan tetapi menggunakan password. Kemudian saya coba menggunakan perintah strings dan didapat clue :

```

(kali@kali)-[~/Documents]
$ strings crackme
/lib64/ld-linux-x86-64.so.2
NK'K
Jw_s:\
mgUa
libc.so.6
puts
stdin
printf
fgets
memset
malloc
__cxa_finalize
strcmp
__libc_start_main
free
GLIBC_2.2.5
_ITM_deregisterTMCloneTable
__gmon_start__
_ITM_registerTMCloneTable
u/UH
[]A\A]A^A_
Input Your Password
Please try later.
MANTUL, flag is LKSSMK28{%s}
Password Salah!
;*3$"
sGCC: (Debian 9.2.1-19) 9.2.1 20191109
crtstuff.c
deregister_tm_clones

```

Kemudian saya menggunakan tool ltrace dan didapat :

```
(kali@kali)-[~/Documents]
$ ltrace ./crackme
puts("Hi!\nInput Your Password"Hi!
Input Your Password
)
= 24
malloc(18)
= 0x55e1a7fd36b0
memset(0x55e1a7fd36b0, '\0', 18)
= 0x55e1a7fd36b0
fgets(0x55e1a7fd36b0
"0x55e1a7fd36b0\n", 18, 0x7ff66b4bb980)
= 0x55e1a7fd36b0
strcmp("0x55e1a7fd36b0\n", "JJJJJJJJJJJJJBxs")
= -26
puts("Password Salah!"Password Salah!
)
= 16
free(0x55e1a7fd36b0)
= <void>
++ exited (status 0) ++

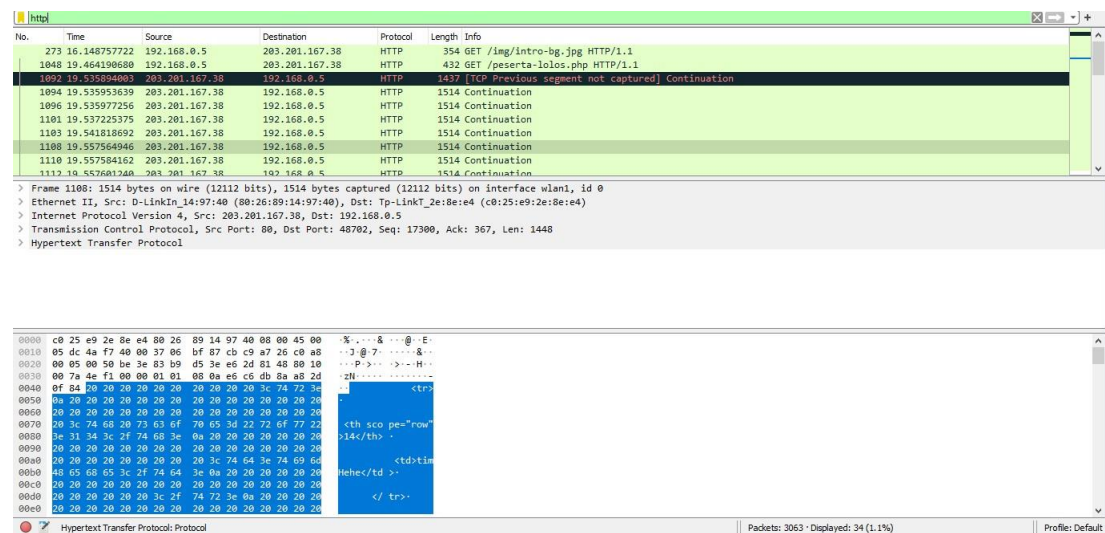
(kali@kali)-[~/Documents]
$ ./crackme
Hi!
Input Your Password
JJJJJJJJJJJJJBxs
MANTUL, flag is LKSSMK28{JJJJJJJJJJJJJBxs}
```

Selanjutnya, saya jalankan lagi filenya dan didapat :

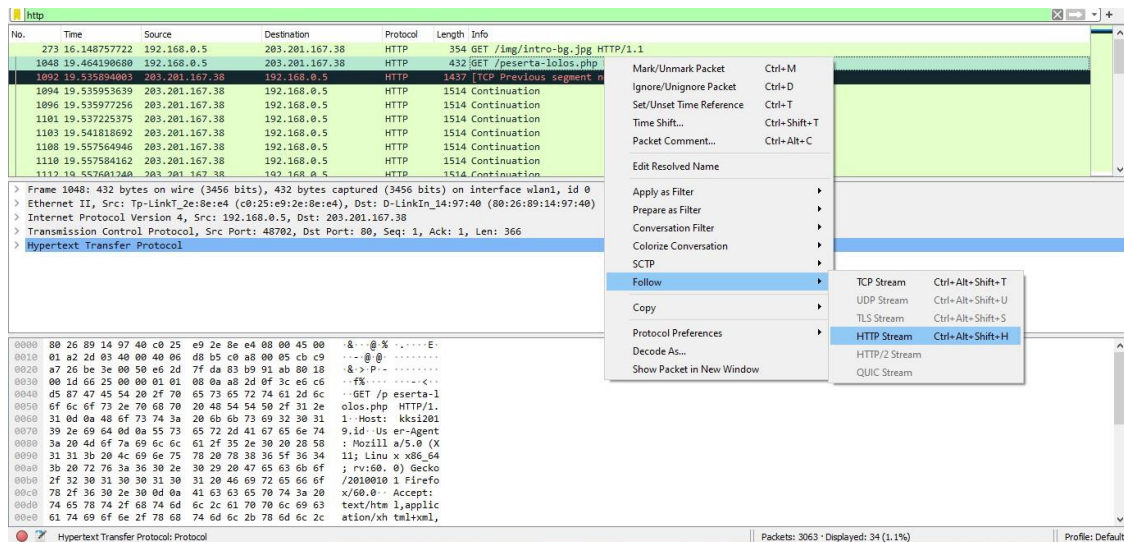
Flag = LKSSMK28{JJJJJJJJJJJJJBxs}

PASS.PCAPNG

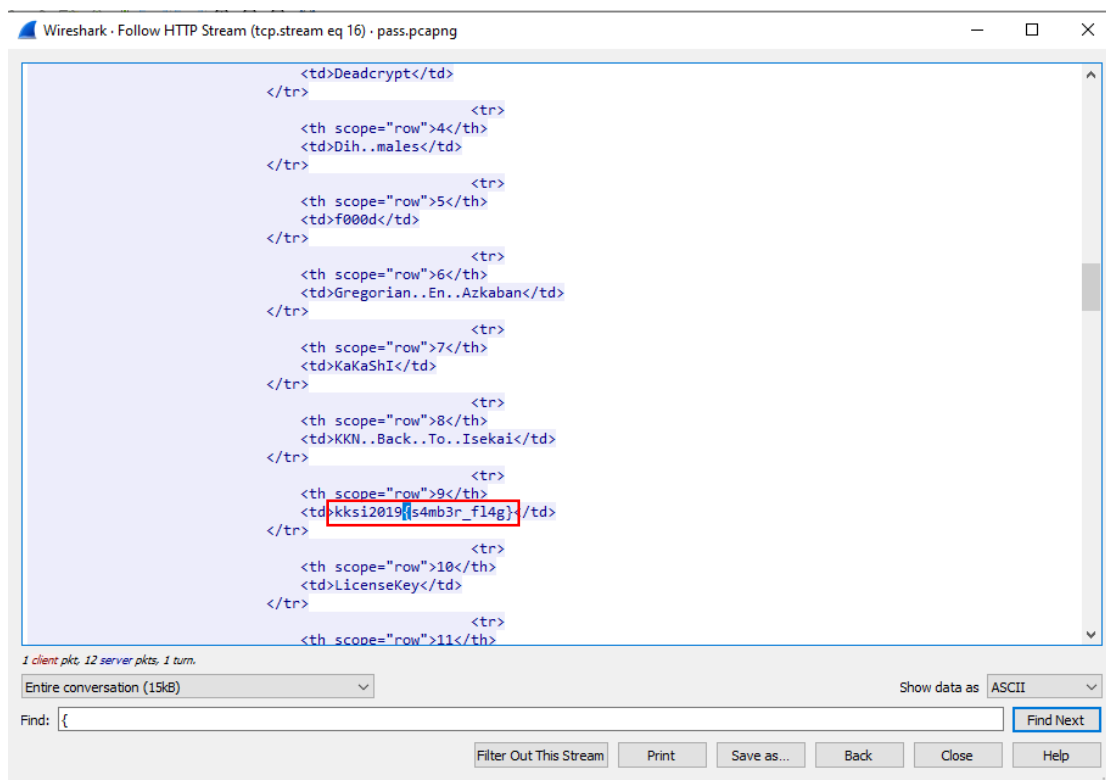
Diberikan soal, lalu saya buka menggunakan aplikasi wireshark kemudian saya lakukan filter packet HTTP seperti berikut :



Setelah itu saya lakukan follow http stream pada packet :



Dan didapat flag :



Flag = kks2019{s4mb3r_fl4g}

BONGKARZZZ

Diberikan file, lalu saya coba buka ternyata diperlukan username dan password. Kemudian saya mencari username dan password untuk file tersebut. Saya coba menggunakan perintah strings tetapi tidak kunjung ditemukan juga. Selanjutnya saya coba menggunakan radare2 dan saya lakukan analyze terhadap file.

```
(kali@kali)-[~/Documents]
$ r2 -d ./bongkarzzz
Process with PID 1455 started ...
= attach 1455 1455
bin.baddr 0x08048000
Using 0x08048000
asm.bits 32
glibc.fc_offset = 0x00148
[0xf7f960b0]> aaa
[x] Analyze all flags starting with sym. and entry0 (aa)
[x] Analyze function calls (aac)
[x] Analyze len bytes of instructions for references (aar)
[x] Check for vtables
[TOFIX: aaft can't run in debugger mode.ions (aaft)]
[x] Type matching analysis for all functions (aaft)
[x] Propagate noreturn information
[x] Use -AA or aaaa to perform additional experimental analysis.
```

Lalu, untuk melihat list function saya ketikkan afl dan disana terlihat ada main function.

```
[0xf7f960b0]> afl
0x08048380 1 33 entry0
0x08048360 1 6 sym.imp.__libc_start_main
0x080483c0 4 43 sym.deregister_tm_clones
0x080483f0 4 53 sym.register_tm_clones
0x08048430 3 30 sym.__do_global_dtors_aux
0x08048450 4 43 → 40 entry.init0
0x08048560 1 2 sym.__libc_csu_fini
0x080483b0 1 4 sym.__x86.get_pc_thunk.bx
0x08048564 1 20 sym._fini
0x080484f0 4 97 sym.__libc_csu_init
0x0804847b 4 115 main
0x080482f4 3 35 sym._init
0x08048350 1 6 loc.imp.__gmon_start__
0x08048330 1 6 sym.imp.printf
0x08048340 1 6 sym.imp.puts
0x08048370 1 6 sym.imp.__isoc99_scanf
```


Lalu saya menemukan nilai pembandingnya :

```
[0xf7f960b0]> pdf @main
; DATA XREF from entry0 @ 0x8048397
115: int main(int argc, char **argv, char **envp);
; var int32_t var_ch @ ebp-0xc
; var int32_t var_sh @ ebp-0x4
; arg char **argv @ esp+0x34
0x0804847b 8d4c2404 lea ecx, [argv]
0x0804847f 83e4f0 and esp, 0xffffffff
0x08048482 ff71fc push dword [ecx - 4]
0x08048485 55 push ebp
0x08048486 89e5 mov ebp, esp
0x08048488 51 push ecx
0x08048489 83ec14 sub esp, 0x14
0x0804848c 83ec0c sub esp, 0xc
0x0804848f 6880850408 push str.Cari_Username_:_ ; 0x8048580 ; "Cari Userna
me : "
0x08048494 e897feffff call sym.imp.printf ; int printf(const char *f
ormat)
0x08048499 83c410 add esp, 0x10
0x0804849c 83ec08 sub esp, 8
0x0804849f 8d45f4 lea eax, [var_ch]
0x080484a2 50 push eax
0x080484a3 6891850408 push 0x8048591
0x080484a8 e8c3feffff call sym.imp.__isoc99_scanf ; int scanf(const char *fo
rmat)
0x08048499 83c410 add esp, 0x10
0x0804849c 83ec08 sub esp, 8
0x0804849f 8d45f4 lea eax, [var_ch]
0x080484a2 50 push eax
0x080484a3 6891850408 push 0x8048591
0x080484a8 e8c3feffff call sym.imp.__isoc99_scanf ; int scanf(const char *fo
rmat)
0x080484ad 83c410 add esp, 0x10
0x080484b0 8b45f4 mov eax, dword [var_ch]
0x080484b3 3de01e1100 cmp eax, 0x111ee0
0x080484b8 7412 je 0x80484cc
0x080484ba 83ec0c sub esp, 0xc
0x080484bd 6894850408 push str.Cari_Password_:__ ; 0x8048594 ; "Cari Passwo
rd :("
0x080484c2 e879feffff call sym.imp.puts ; int puts(const char *s)
0x080484c7 83c410 add esp, 0x10
0x080484ca eb15 jmp 0x80484e1
0x080484cc 83ec08 sub esp, 8
0x080484cf 68d9a46500 push 0x65a4d9
0x080484d4 68a6850408 push 0x80485a6
0x080484d9 e852feffff call sym.imp.printf ; int printf(const char *f
ormat)
0x080484de 83c410 add esp, 0x10
; CODE XREF from main @ 0x80484ca
0x080484e1 b800000000 mov eax, 0
0x080484e6 8b4dfc mov ecx, dword [var_sh]
0x080484e9 c9 leave
0x080484ea 8d61fc lea esp, [ecx - 4]
0x080484ed c3 ret
```

Dan saya konversikan nilai pembanding tersebut ke desimal :

Hexadecimal to Decimal converter

From

To

Hexadecimal

Decimal

Enter hex number

111ee0

16

= Convert

✕ Reset

↕ Swap

Decimal number

1122016

10

Saya coba angka tersebut untuk username dan ternyata didapat password nya juga :

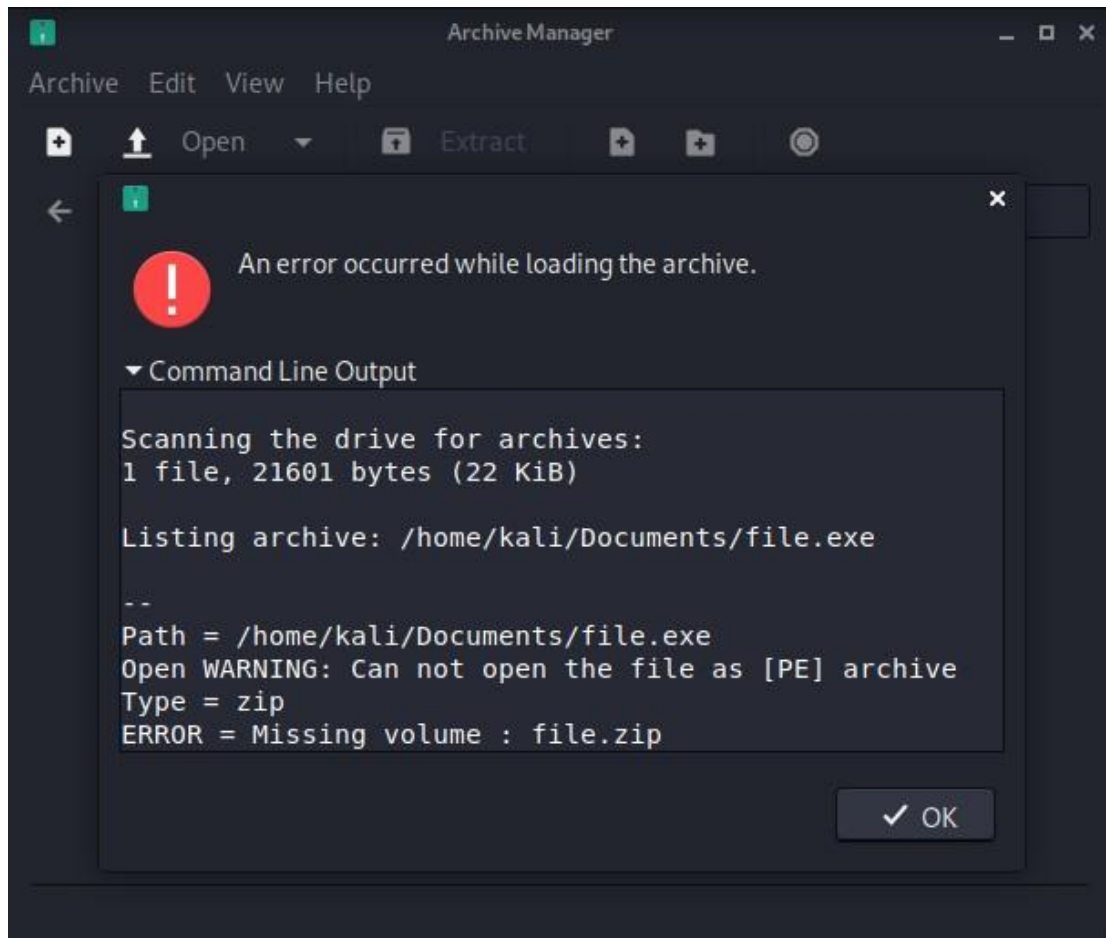
```
(kali@kali)-[~/Documents]
$ ./bongkarzzz
Cari Username : 1122016
6661337
```

Username : 1122016

Password : 6661337

FILE.EXE

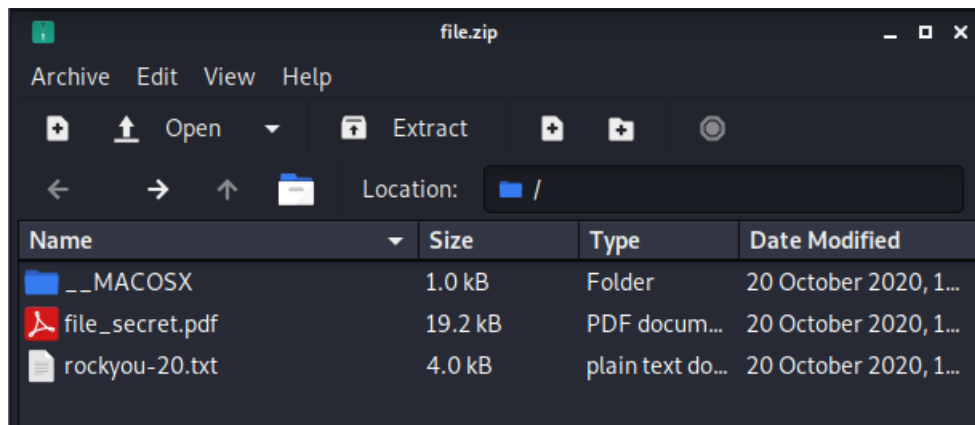
Setelah diberikan file, saya coba buka tetapi ternyata didapat error seperti berikut :



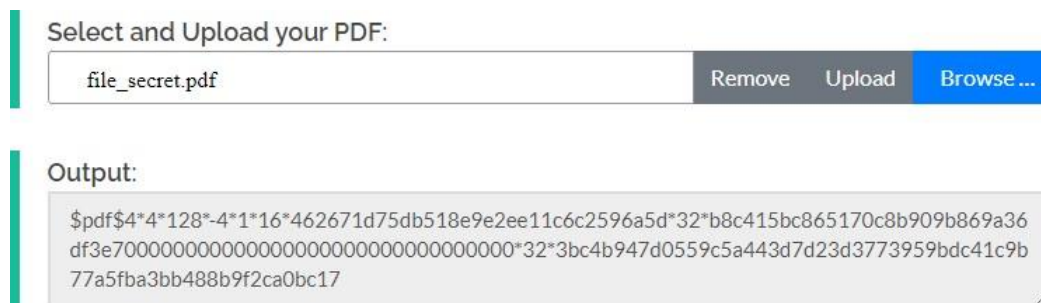
Dari gambar di atas dapat dilihat terdapat hint yang menunjukkan bahwa file tersebut adalah file dengan ekstensi zip, maka saya ubah nama file tersebut ke file.zip :

```
(kali@kali)-[~/Documents]
$ mv file.exe file.zip
```

Lalu saya coba buka dan ternyata ada dua file yang tersimpan di sana yaitu sebagai berikut :



Diberikan sebuah file pdf yang terkunci dengan password dan sebuah file wordlist. Kemudian langsung saja saya brute force dengan John The Ripper :



```
(root@kali) - [/home/kali/mnt/LKSN 2021]
# john --wordlist=rockyou-20.txt pdf.txt
Using default input encoding: UTF-8
Loaded 1 password hash (PDF [MD5 SHA2 RC4/AES 32/64])
Cost 1 (revision) is 4 for all loaded hashes
Will run 8 OpenMP threads
Press 'q' or Ctrl-C to abort, almost any other key for status
hellokitty (?)
1g 0:00:00:00 DONE (2021-10-23 22:18) 20.00g/s 5120p/s 5120c/s 5120C/s 123456..angelo
Use the "--show --format=PDF" options to display all of the cracked passwords reliably
Session completed
```

Didapat password nya yaitu "hellokitty" lalu saya buka dokumennya dan muncul hint lagi berupa base64 :

TetTU01LMjh7Y3JhY2sxbjlfZG9jdW0zbIR9==

Base64 tersebut saya konversi ke ascii dan didapat flag :

Text (ASCII / ANSI)

LKSSMK28{crack1n9_docum3nT}

Convert

Highlight Text

Hexadecimal

4c 4b 53 53 4d 4b 32 38 7b 63 72 61 63 6b 31 6e 39
5f 64 6f 63 75 6d 33 6e 54 7d

BASE64

TEtTU01LMjh7Y3JhY2sxbjlfZG9jdW0zbIR9

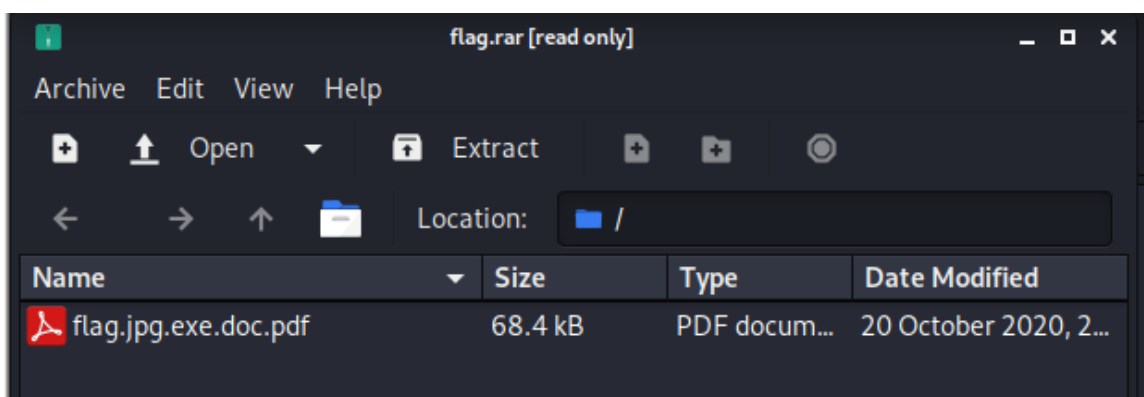
Flag = LKSSMK28{crack1n9_docum3nT}

FLAG.JPEG.EXE.DOC.RAR

Diberikan sebuah file dengan ekstensi jpeg, exe, doc, dan rar saya langsung melakukan identifikasi menggunakan command "file" di terminal :

```
(kali㉿kali)-[~/Documents]
$ file flag.exe
flag.exe: RAR archive data, v5
Devices
(kali㉿kali)-[~/Documents]
$ mv flag.exe flag.rar
```

Setelah diketahui file tersebut merupakan file rar, maka saya rename lalu saya buka dan ternyata berisi file dengan ekstensi pdf :



Kemudian saya ekstrak file tersebut dan setelah diidentifikasi file tersebut merupakan file corrupt yang tidak dapat dibaca :

```
(kali㉿kali)-[~/Documents]
$ file flag.jpg.exe.doc.pdf
flag.jpg.exe.doc.pdf: data
```

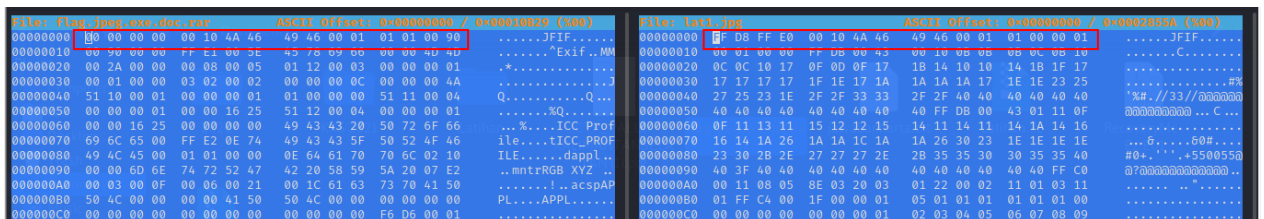
Selanjutnya saya melakukan strings terhadap file dan di bagian header terdapat kata JFIF yang mengindikasikan file tersebut adalah file JPG :

```
(kali㉿kali)-[~/Documents]
$ strings flag.jpg.exe.doc.pdf
JFIF
^Exif
ICC Profile
tICC_PROFILE
dapl
mnrRGB XYZ
acspAPPL
APPL
-appl
desc
bdscm
Jcprt
#wtpt
```

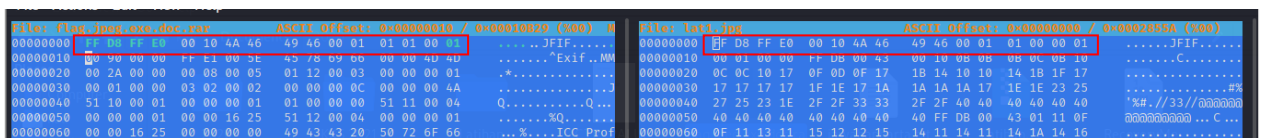

Setelah itu saya rename menjadi file berekstensi jpg dan saya coba buka namun tidak berhasil. Di sini saya mengasumsikan kita harus memperbaiki bagian header file tersebut agar file dapat dibuka.

```
(root@kali)-[/home/kali/Documents]
# mv flag.jpg.exe.doc.pdf flag.jpg
```

Saya menggunakan tool hexeditor di kali dan header nya saya bandingkan dengan file jpg yang sebenarnya, dan benar saja terdapat kesalahan dalam nilai hexa seperti pada gambar :



Lalu langsung saja saya perbaiki menjadi seperti berikut :



Kemudian saya coba buka kembali dan ternyata berhasil :



Flag = LKSSMK28{Yuu-kita-ke-jakarta}