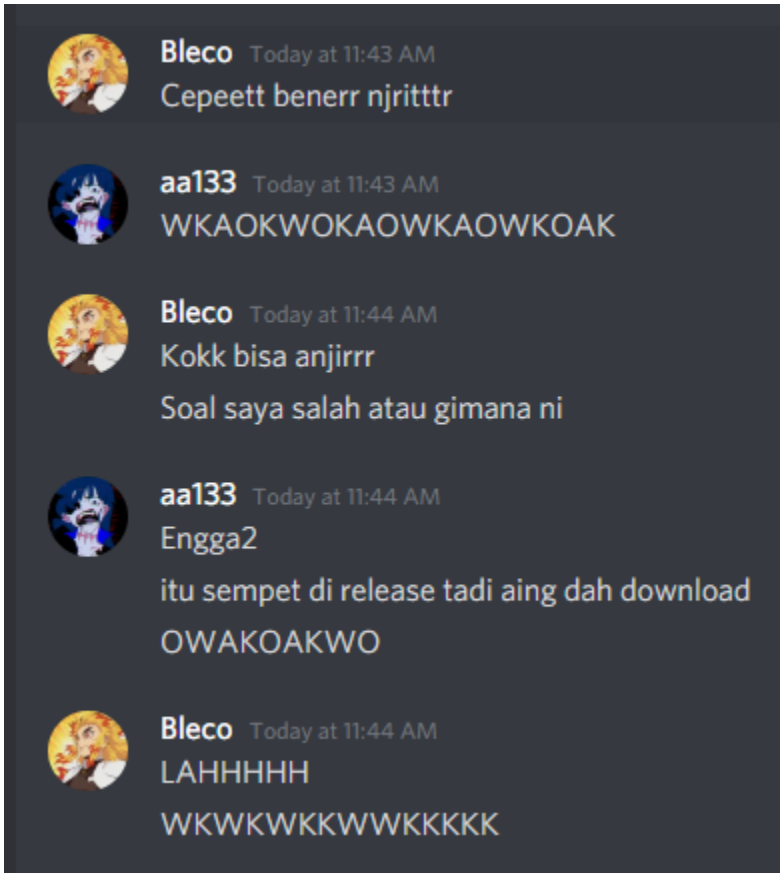
Terlantarkan

Daftar Isi

Forensics

noodles

Langkah Penyelesaian:



Diberikan sebuah zip file berisi gambar noodle dan .git jadi penulis melihat history commit menggunakan git log dan git reflog

```
commit 8ddd9f7f1a2a9923d94e9dca60fa47c9393acd2e
Author: blecow <blacowhait@gmail.com>
Date:   Sat Sep 4 17:47:59 2021 +0100

    noodles

commit 4bfe165a7fca5f4d82cc1bf82ec0e99f5096e5fb
Author: blecow <blacowhait@gmail.com>
Date:   Sat Sep 4 17:47:58 2021 +0100

    noodles

commit d74e906aae1ce4fef15e18829ec473f2ec88cf70
Author: blecow <blacowhait@gmail.com>
Date:   Sat Sep 4 17:47:56 2021 +0100

    noodles

commit 63e7cbdbcfd98032f72a3c0e8a14de039f4fa56f
Author: blecow <blacowhait@gmail.com>
Date:   Sat Sep 4 17:47:54 2021 +0100
```

Penulis mengextract semua noodles.png di commit dan membandingkannya, ternyata last byte selain null bytes berbeda di setiap commit, tinggal di extract saja menggunakan script yang akan dicantumkan dibawah.

```
root@kali:~/Documents/hacktoday/final_hacktoday/noodles/all_commit# python script.py
}Hai, hai, hai, JS(HY)*F@G)*h9081f2y308r i8302wbhf 08h80y)*GH* 8y2308ghioBdfb97a t3
97rgt2193gtr ogou hai, hai, hai, never gona give you up, Never gona let you down, Nev
er gona run around and desert you, Never gona make you cry, Never gona say godbye, Ne
ver gona tel a lie and hurt you, ok this is your flag : Hacktoday{r1ckr0l OPHDuogfbja
sbdasldhu8asduhku21983y2108g210uvfu2 just kiding, this is real flag : Hacktoday{r1ckr
0l_in_fl4g}
root@kali:~/Documents/hacktoday/final_hacktoday/noodles/all_commit#
```

Code:

[code(jika ada)]

script.py

```
#!/usr/bin/bash
import os
from binascii import unhexlify


dirs        =        ["8ddd9f7f1a2a9923d94e9dca60fa47c9393acd2e",
"4bfe165a<--snipped→>
```

```
flag = ""

for dire in dirs[::-1]:
     os.system("xxd -p "+dire+"/noodles.jpg > temp")
     with open("temp", "r") as f:
          output = f.read()

flag+=unhexlify(output.replace("\n","").replace("00","")[-2:])
print(flag)
```

Flag:
hacktoday{r1ckr0l_in_fl4g}

Heicyuuu!

Langkah Penyelesaian:

Diberikan 2 file .heic yang bisa di convert ke .jpg

Tapi salah satu file .heic corrupt headernya perlu diselamatkan dulu hehe.

Bisa di convert ke jpg menggunakan tool online

https://heictojpg.com/

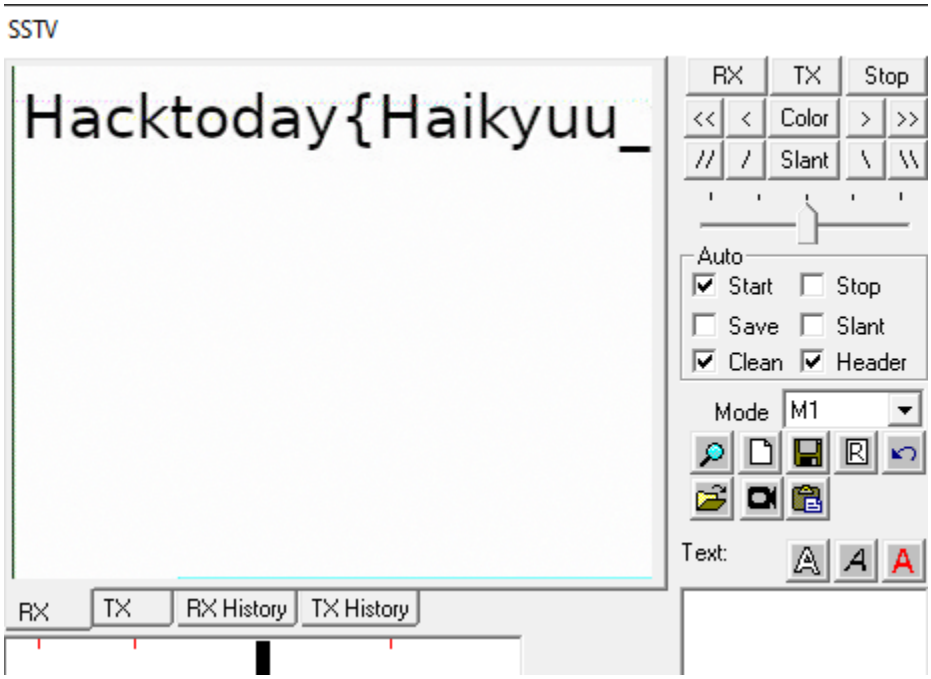
https://mega.nz/file/z7pQIBxa#LjL4jXpzQedaiVIN-VCHSnL4MFY2bb0ayq-fCwATdlg

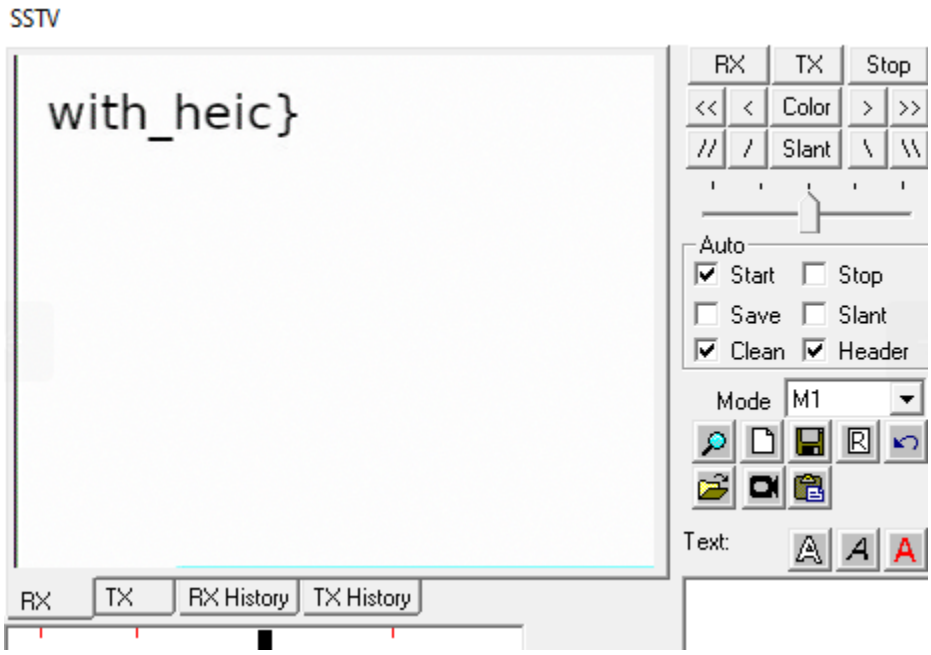
https://mega.nz/file/3nwwHbLT#Vf11VQ4yDTR10aFbbAHxE7jJ3_PWWlyNwtximXxBb2M

shimizu : https://mega.nz/file/z7pQIBxa#LjL4jXpzQedaiVIN-VCHSnL4MFY2bb0ayq-fCwATdlg

kiyoko : https://mega.nz/file/3nwwHbLT#Vf11VQ4yDTR10aFbbAHxE7jJ3_PWWlyNwtximXxBb2M

Keduanya berisi audio SSTV





Flag:

hacktoday{Haikyuu_with_heic}

Cryptography

impozzibl-ez

Langkah Penyelesaian:

```python
def generatez_kamusz():
    digitz = list(hexdigitz[:-6])
    digitzz = list(hexdigitz[:-6])
    shuffle(digitzz)
    kamusz = {}
    for iz in range(len(digitz)):
        kamusz.update({digitz[iz] : digitzz[iz]})
    return kamusz

def encryptz(msgz):
    kamusz = generatez_kamusz()
    scrambledz_hexz = "".join([kamusz[charz] for charz in msgz])
    return scrambledz_hexz

flagz = "# CENSORED"
flagz = [md5(iz.encode()).hexdigest() for iz in flagz]

scrambledz_flagz = [encryptz(haz) for haz in flagz]

open("out.py","w").write(f"{scrambledz_flagz = }")
```

Untuk soal ini, kita bisa lihat masing-masing karakter di hash dengan md5 lalu huruf nya di acak, tapi hanya diacak per karakter. Jadinya bila huruf 'a' diganti dengan huruf 'b', semua huruf 'a' di hash tersebut berubah dengan huruf 'b'. Untuk cara malas kita bisa menghitung count masing masing karakter dari hash yang valid (hash dari printable) lalu bandingkan ada kah counter yang sama hanya berbeda huruf-hurufnya.

```
PS C:\Users\EternalBeats\Documents\CTF\hacktoday\final\impozzibl-ez> python .\out.py
 ja0g.!!!  sambit n!h.  falgznya:  th3_h3x__i5_scr4mb3ld_butt_t3h_p4ttr3n__i5_th3_s4m3
```

Code:

```
out.py
```

```
from hashlib import md5

from string import printable

from collections import Counter


scrambledz_flagz         =         ['3214997b3d7db227d2721afc98779b46',
'57536ee4bedfb8af81c88737313cbb6b',
'cff42a9dfc749be604f0dd83bd223bb4',
'35379abf18786825662d755151bd6f7e',
'e93733454122 56e2b711808c12640478',
'131fab5afdff2dda23c7586e15e187c8',
'26ddf6fd6c014a767d7e6567ae4e7e9d',
'0a22bab2acd74e3a3238afa3e8483862',
'43ff636f320d57e3efe83b3e7858e8cf',
'75fbcc467e4e6554e545fd92c044c6ba',
'caf96638c2328aa32a3af0bd6e336894',
'da070d30ca65f8d18efbd73ce0ad9da2',
'49923e6094d26afb729700c8a0338aa2',
'd1a1366f3beb89cd4c734eaa8e9f3bf5',
'8e06f99002c0e9074c5317333f41f3d9',
'3f6b2b275972d2f4d6b994431ba9315a',
'3ac832ed842c85902f5117af9b9c9d43',
'bc13ff58b7578cc57c5c1069fe55f83a',
'6717c409fb7398db59705fd672524295',
'e13361631bc7f9010304181094f404d3',
'413fb27f33b1c6af93d4942628071673',
'9094b18b4344633b60758dfe98e9d57d',
'61b599a063a3011a31a1b2c794aa905d',
'85a6dd07840475504505ae1bdf00d763',
'196db044a8b97d00388b07ead72700ea',
'7442fa3847123ced92498860c8ff0cc2',
'5714f989eb45c212bee88ffc156ed1dd',
'4bc5cc898f339043e5ffada6f308d85a',
'91e8aca5fe595f858fbf40132367088b',
'17e78209df7a95bf69706db174642496',
'28b309f60b0226913124b033092ce2b7',
'8441a53e48213cbd9149ee67ceaa7cc1',
'f8a0345d5a81ace297a3061836ef7817',
```

```
'0138dd460c4c6114c1413ea5d744d689',
'067911380434866346 3675ae1b33189c',
'80e68daf69de61b2d3177c03b4bebf98',
'63d824a8dd23b098fde6f640475a30ad',
'b77573fba567b9eb93203ed1e98f58e2',
'8014b8539262a399291a2edb1acd337e',
'e238cd0833c26b98a3feaedbd1702b03',
'9cc6ceb9f6dc98498e05e413482b6240',
'a88e3ec7dc9b965e04b94ce3744bcf2c',
'fd6c5fa273b312773761340561802294',
'1b9e317d6c8c0d66c690cf439054ddaf',
'f813a3a9629aca84c132244fd3e2fd6e',
'ca98357255fcb3ad9777b5607a834e9d',
'a2ce1a867f5f9677f7c9f301c9d066b3',
'7befe73e0b82917a16957f306eb747bc',
'bfef1e01d6a454a4d87bd1aeb1e580aa',
'ede2d8a3327b4ecf07d098960d2fc429',
'e1a663afed670a59f515e3a4ac8d5ddb',
'76865ee351f1b247c4d5cf88bf235130',
'81b9622bbfdb12bc4d0aecaaa64e6a52',
'acc4c56a248ca17a15ef57097136437e',
'f7d89131b68fc2d26bb3399cdf4b5d55',
'71228b4cb62abc4a9ee72420c08b4cf6',
'5a980562fdbd12ffdf91d7c0914c22e7',
'c26a8ff660462f61d47b51bbb8d58b9f',
'a4aa53003562591f88bd685050d78251',
'a03ca5d8c153c4e75b499f0be2e3e81a',
'47b04d1302db0956dc988a7c5e5b5324',
'ca90bc146e5ed466e69def3b9d23447f',
'2c03271f3b7035487e566dce49404fb2',
'7dd2d0b7f29d715710c305e8514b245c',
'597bac4b77a9138b072505c3ced49347',
'8514d83c0676ac00601a692d1ab2cce9',
'464347d1131664d2892c92c5d3e47a3e',
'd3677461da7b56981939d46f620a9aac',
'c4adc815df8ad76b82700e42696a65fc',
'2c9e2b4fe7b9ed16bad005ca18191f72',
'8d8adceffa02186350d54c495da361a4',
'75535f47c3857a97afb6f9d19a04309b',
```

```
'95c58f2ba15eb7d10b520ad95303f3b0',
'210eb28963f3c966360c354b0cd49975',
'71b8c7d5e62695ee6eb96a0cb9f0554a',
'795eaea3103aba98b5e008874ef0741f',
'a40f817511dab84e0777b12974f8c60e',
'b8f16bec34d4ac3343fa4956fa75cc09',
'ae2caf31c8f2c579f65ddbe67472718a',
'42d6ce96ddc278f63d1434e8eb59289d',
'b00209fb3280bc5bc9d195675cef2e5d',
'2ba4625c7f9f0c77f7a0f136a0e3cc81',
'a8c4cabc58d0e2a926e3a4b56c8afa87',
'a5200328ad0912768757a32c24ed7ddf',
'7404cdd3cb9b1ea7fa2cf90019e3cb36',
'abb6bcda76eba9fa9c28cf03f95d65f2']


validmd5 = {}
charset = printable[:-5]
for c in charset:
    validmd5[md5(c.encode()).hexdigest()] = c


listCountValid = {}
for i in validmd5:
    listCountValid[i] = Counter(i)


for s in scrambledz_flagz:
    counter = Counter(s)
    flag = False
    for l in listCountValid:
        flag = True
        tmp = list(counter)
        tmp2 = list(listCountValid[l])
        if len(tmp) != len(tmp2):
```

```
            continue
        for i in range(len(tmp)):
            if counter[tmp[i]] != listCountValid[l][tmp2[i]]:
                flag = False
                break
        if flag:
            print(validmd5[l], end='')
```

Flag:
hacktoday{th3_h3x__i5_scr4mb3ld_butt_t3h_p4ttr3n__i5_th3_s4m3}

Reverse Engineering

ludwig

Langkah Penyelesaian:

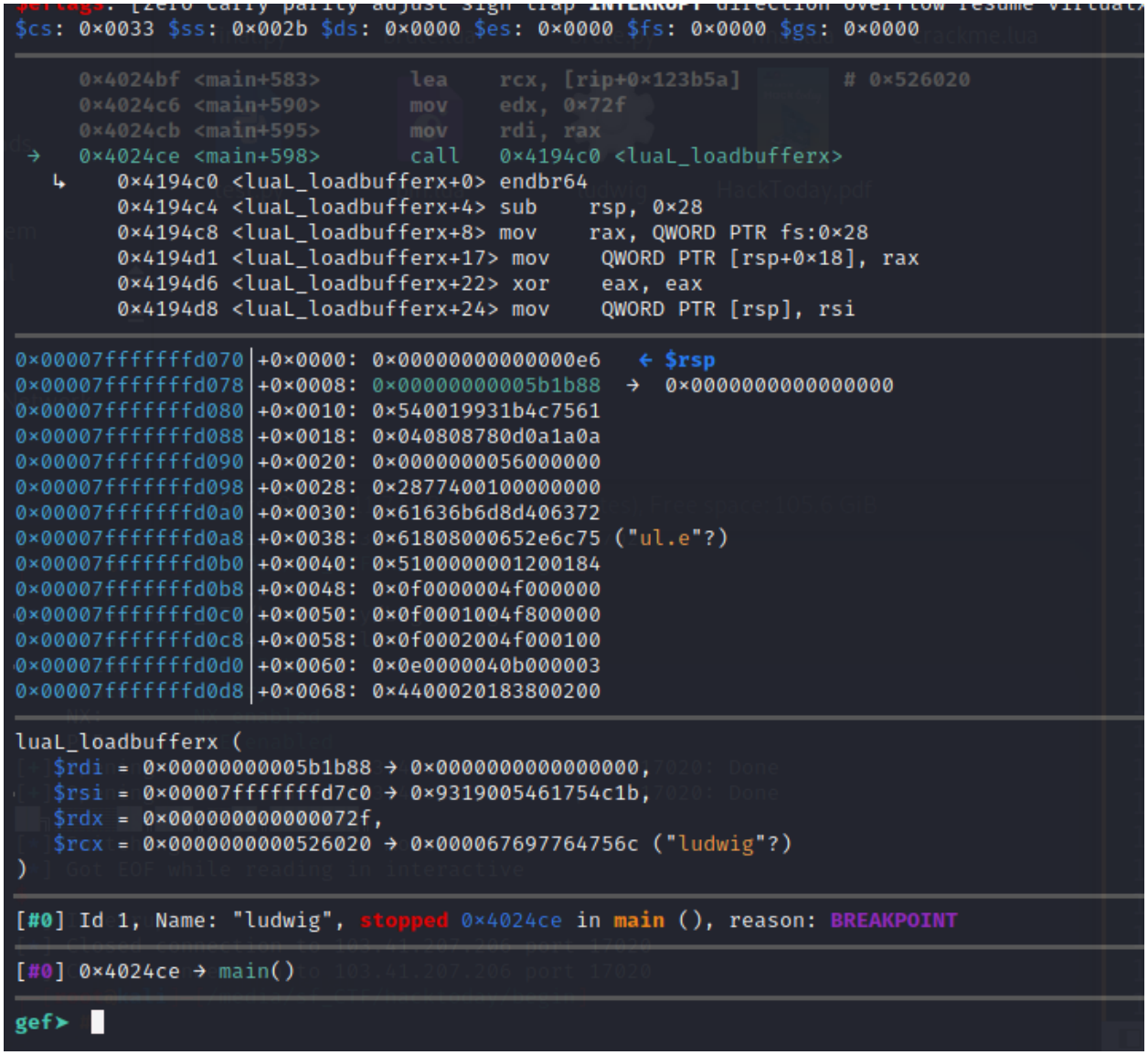
Pertama diberikan file ELF, penulis langsung decompiler menggunakan ida, dan langsung melihat function main.

```
__asm { endbr64 }
v10 = v3;
v9 = __readfsqword(0x28u);
v6 = luaL_newstate(argc, argv, envp);
luaL_openlibs(v6);
qmemcpy(&v7, &unk_526040, 0x738uLL);
for ( i = 0; i <= 229; ++i )
{
  load(&v7 + i);
  *((_BYTE *)&v8 + 8 * i) = *((_DWORD *)&v10 + 2 * i - 928) >> 24;
  *((_BYTE *)&v8 + 8 * i + 1) = *((_DWORD *)&v10 + 2 * i - 928) >> 16;
  *((_BYTE *)&v8 + 8 * i + 2) = *((_WORD *)&v10 + 4 * i - 1856) >> 8;
  *((_BYTE *)&v8 + 8 * i + 3) = *((_DWORD *)&v10 + 2 * i - 928);
  *((_BYTE *)&v8 + 8 * i + 4) = *((_DWORD *)&v10 + 2 * i - 927) >> 24;
  *((_BYTE *)&v8 + 8 * i + 5) = *((_DWORD *)&v10 + 2 * i - 927) >> 16;
  *((_BYTE *)&v8 + 8 * i + 6) = *((_WORD *)&v10 + 4 * i - 1854) >> 8;
  *((_BYTE *)&v8 + 8 * i + 7) = *((_DWORD *)&v10 + 2 * i - 927);
}
if ( !(unsigned int)luaL_loadbufferx(v6, (__int64)&v8, 1839LL, "ludwig", 0LL) )
  lua_pcallk(v6, 0, -1, 0, 0LL, 0LL);
lua_close(v6);
return 0;
}
```

Didalam function main ada kata kunci lua, penulis mengetahui lua adalah programming language.

Setelah mencari digoogle yang membahas cara decompiler lua file didalam elf file, penulis menemukan https://tripoloski1337.github.io/ctf/2019/09/09/reverse-engineering-lua-bytecode.html

Didalam website tersebut code luanya ada berada sebelum lua_load, didalam file challenge ada luaL_loadbufferx, jadi kemungkinan codenya bisa ada di variable v8. Karena diatas function luaL_loadbufferx ada beberapa dekripsi, jadi penulis langsung menggunakan gdb dan break sebelum menjalankan luaL_loadbufferx.



Code lua nya ada berada didalam address 0x00007fffffffd7c0, langsung ajah diprint sepanjang 1839. Penulis akan mengunakan command x/229gx 0x00007fffffffd7c0.

```
gef➤  x/229gx 0x00007fffffffd7c0
0x7fffffffd7c0: 0x9319005461754c1b      0x780808040a1a0a0d
0x7fffffffd7d0: 0x0000000000000056      0x0140772800000000
0x7fffffffd7e0: 0x6d6b63617263408d      0x00808061756c2e65
0x7fffffffd7f0: 0x0000005184012001      0x0000000f0000004f
0x7fffffffd800: 0x0001000f0000804f      0x0002000f0001004f
0x7fffffffd810: 0x0400000e0300000b      0x0102004400028083
0x7fffffffd820: 0x0600000e0300000b      0x0000008b02010044
0x7fffffffd830: 0x0003818300000100      0x0082013d030300c4
0x7fffffffd840: 0x0300019380003538      0x0004020300000052
0x7fffffffd850: 0x0005030300048283      0x7fff8201000301ce
0x7fffffffd860: 0x000103000b00028b      0x000402cb050202c4
0x7fffffffd870: 0x000a06000200058b      0x0903060c020205c4
0x7fffffffd880: 0x800000b8000b0639      0x0680022f80040215
0x7fffffffd890: 0x000502cd020002cc      0x0082023d000002b6
0x7fffffffd8a0: 0x0c00028b800026b8      0x000003000d05028e
0x7fffffffd8b0: 0x009902bd020202c4      0x00070283800023b8
0x7fffffffd8c0: 0x1006030e0f00030b      0x0202034400010380
0x7fffffffd8d0: 0x0000005218000393      0x8012848180198401
0x7fffffffd8e0: 0x8010058180180501      0x800f868180170601
0x7fffffffd8f0: 0x80170781801f0701      0x80278881802c8801
0x7fffffffd900: 0x80100981801f0901      0x80198a81801a0a01
0x7fffffffd910: 0x801d8b8180120b01      0x80130c81802e0c01
0x7fffffffd920: 0x80100d81801d8d01      0x801c0e81800f8e01
0x7fffffffd930: 0x802d8f8180110f01      0x80000401001803ce
0x7fffffffd940: 0x80000501800b8481      0x00050600000b044a
0x7fffffffd950: 0x110d068e0c00068b      0x0c00078b0b07070c
0x7fffffffd960: 0x00060800120f078e      0x020307c4000b0880
0x7fffffffd970: 0x0f0f072e0f0e072b      0x00020635020206c4
0x7fffffffd980: 0x130b0619000c0280      0x007f063d091305b0
0x7fffffffd990: 0x00050600800001b8      0x00020635000a0683
0x7fffffffd9a0: 0x000b8449000c0280      0x8000050115058414
0x7fffffffd9b0: 0x020404447ffe8581      0x0300040b00080280
```

Penulis akan melakukan secara manual decode hex, setelah itu untuk mendapatkan original source code, penulis akan menggunakan unluac yang ada didalam url website yang saya berikan.

Dibawah ini adalah snippet haisl decompiler.

Isi file Crackme.lua adalah hasil decode by hex sebelumnya.

Saya akan memasukan ke dalam file crackme.lua

```
┌──[root@kali]─[/media/sf_CTF/hacktoday]
└─# ./unluac_2021_08_29b.jar crackme.lua
Picked up _JAVA_OPTIONS: -Dawt.useSystemAAFontSettings=on -Dswing.aatext=true
Picked up _JAVA_OPTIONS: -Dawt.useSystemAAFontSettings=on -Dswing.aatext=true
Picked up _JAVA_OPTIONS: -Dawt.useSystemAAFontSettings=on -Dswing.aatext=true
function split_serial(inputstr, sep)
  local count = 0
  if nil == sep then
    sep = "%s"
  end
  local serial = {}
  for str in string.gmatch(inputstr, "([^" .. sep .. "]+)") do
    table.insert(serial, str)
    count = count + 1
  end
  return serial, count
end
function hex(str)
  return (str:gsub(".", function(c)
    return string.format("%02X", string.byte(c))
  end))
end
function convert(str)
  local value = tonumber(hex(str), 16)
  for i = 1, 1337 do
    value = value ~ value >> 1
  end
  return value
end
io.write([[
INPUT YOUR SERIAL NUMBER !!!
ex: VERY1234-1SECURE1-6SERIAL9
]])
```

```lua
io.write([[
INPUT YOUR SERIAL NUMBER !!!
ex: VERY1234-1SECURE1-6SERIAL9
]])
local serial = io.read()
local splitted, length = split_serial(serial, "-")
if 3 == length then
  local secret = {
    9013365925341683735,
    3208797737010034330,
    2619883148120664450
  }
  local count = 0
  for k, v in pairs(splitted) do
    local value = convert(v)
    if secret[k] == value then
      count = count + 1
    end
  end
```

```lua
function convert(str)
  local value = tonumber(hex(str), 16)
  for i = 1, 1337 do
    value = value ~ value >> 1
  end
  return value
end
```

Dari sini kita bisa lihat variable 'secret' merupakan target kita, kita butuh hasil yang di convert menjadi angka di 'secret' itu. Kita bisa mencoba dari huruf pertama yang di split untuk di test huruf apa yang paling dekat dengan target yang kita inginkan. Dari situ kita coba huruf kedua, dan seterusnya.

```python
target = [9013365925341683735,3208797737010034330,2619883148120664450]
serialKey = b""
for t in target:
    template = b""
    for i in range(8):
        minimal = t
        hasil = {}
        for c in digits+ascii_uppercase:
            tmp = template
            tmp += c.encode()
            hasil[c] = convert(bytes_to_long(tmp.ljust(8, b'0')))

        for c in hasil:
            if minimal >= abs(t-hasil[c]):
                minimal = abs(t-hasil[c])
                curr = c
        template += curr.encode()
```

```
PS C:\Users\EternalBeats\Documents\CTF\hacktoday\final\ludwig> python .\brute.py
b'VOVFJRQZ-72ZSWQQH-8ULDRJL2'
```

Setelah dapat serialKey nya kita hanya perlu masukan ke programnya dan kita dapat flag.

```
┌──(kali㉿kali)-[~/Desktop/CTFStuff/hacktoday/final]
└─$ ./ludwig
INPUT YOUR SERIAL NUMBER !!!
ex: VERY1234-1SECURE1-6SERIAL9
VOVFJRQZ-72ZSWQQH-8ULDRJL2
hacktoday{bigger_number_better_person}
```

Code:

brute.py

brute.py

```lua
function split_serial(inputstr, sep)
  local count = 0
```

```
  if nil == sep then
    sep = "%s"
  end
  local serial = {}
  for str in string.gmatch(inputstr, "([^" .. sep .. "]+)") do
    table.insert(serial, str)
    count = count + 1
  end
  return serial, count
end
function hex(str)
  return (str:gsub(".", function(c)
    return string.format("%02X", string.byte(c))
  end))
end
function convert(str)
  local value = tonumber(hex(str), 16)
  for i = 1, 1337 do
    value = value ~ value >> 1
  end
  return value
end
io.write([[
INPUT YOUR SERIAL NUMBER !!!
ex: VERY1234-1SECURE1-6SERIAL9
]])
local serial = io.read()
local splitted, length = split_serial(serial, "-")
if 3 == length then
  local secret = {
    9013365925341683735,
    3208797737010034330,
    2619883148120664450
  }
  local count = 0
  for k, v in pairs(splitted) do
    local value = convert(v)
    if secret[k] == value then
      count = count + 1
    end
  end
  if 3 == count and 26 == string.len(serial) then
    local flag = ""
    local concatted = table.concat(splitted)
    local more_secret = {
      52,
```

```
      38,
      49,
      33,
      47,
      32,
      63,
      47,
      90,
      80,
      63,
      33,
      53,
      52,
      37,
      60,
      93,
      39,
      60,
      33,
      32,
      57,
      35,
      92
    }
    for i = 1, 24 do
      flag = flag .. string.char(more_secret[i] ~
string.byte(concatted, i))
      if 0 == i % 6 then
        flag = flag .. "_"
      end
    end
    flag = flag:sub(1, -2)
    io.write("hacktoday{" .. flag .. "}\n")
  else
    io.write("INVALID SERIAL NUMBER !!!")
  end
else
  io.write("INVALID SERIAL NUMBER !!!")
end
```

brute.py

```
from Crypto.Util.number import bytes_to_long
```

```
from string import ascii_uppercase, digits

def convert(v):
    for _ in range(1337):
        v = ~(~v >> 1 ^ v)
    return v


v = bytes_to_long(b"VERY1234")
assert convert(v) == 9014765630940167186


target                                                                  =
[9013365925341683735,3208797737010034330,2619883148120664450]
serialKey = b""
for t in target:
    template = b""
    for i in range(8):
        minimal = t
        hasil = {}
        for c in digits+ascii_uppercase:
            tmp = template
            tmp += c.encode()
            hasil[c] = convert(bytes_to_long(tmp.ljust(8, b'0')))


        for c in hasil:
            if minimal >= abs(t-hasil[c]):
                minimal = abs(t-hasil[c])
                curr = c
        template += curr.encode()
```

```
    if convert(bytes_to_long(template)) == t:
        serialKey = serialKey + template + b"-"


serialKey = serialKey[:-1]
print(serialKey)
```

Flag:
hacktoday{bigger_number_better_person}