Writeup ITobaFest2017

$(Cyber Security IPB - Pwning The World)

- | Recon | -

Temukan Jalannya

Diberikan sebuah string base64

```
QmFuayBJZGVudGlmaWNhdGlvbiBOdW1iZXINCjQ1NzA5OElTTzMxNjYtMQ==
```

Kita decode

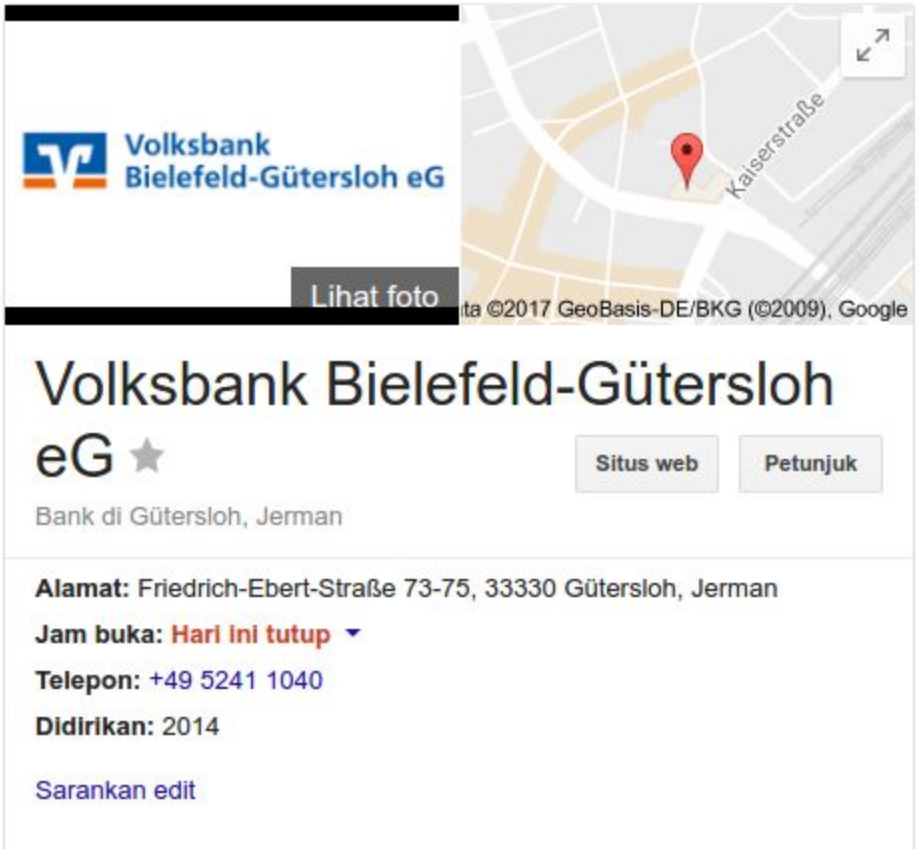
```
$ echo 'QmFuayBJZGVudGlmaWNhdGlvbiBOdW1iZXINCjQ1NzA5OElTTzMxNjYtMQ==' | base64 -d

Bank Identification Number
457098ISO3166-1
```

Dengan skill Google-Fu, kita mencoba searching dengan keyword Bank Identification Number kita dapatkan https://binlist.net/ , nah pada website tersebut di jelaskan cara lookup Bank Identification Number

```
$ curl "https://lookup.binlist.net/457098"
{"scheme":"VISA","number":{"length":null,"prefix":"457098"},"type":"CREDIT","brand":"","prepaid":false,"bank":{"name":"VOLKSBANK GUETERSLOH","logo":"","url":"","city":"","phone":"49"},"country":{"alpha2":"DE","name":"Germany","numeric":"276","latitude":51,"longitude":9}}
```

Didapatkan nama sebuah bank yaitu :
VOLKSBANK GUETERSLOH

Cari di google: "VOLKSBANK GUETERSLOH"



Flagnya adalah Alamat: Friedrich-Ebert-Straße 73-75

Flagnya tanpa spasi
Flag: **ITF{Friedrich-Ebert-Straße73-75}**

- | Reversing | -

File binary bisa di-download di:

https://drive.google.com/open?id=0B2bPphSraI9FQ0pGZVRkX3A1R1U

Rev1

Diberikan binary

```
ev1: ELF 64-bit LSB executable, x86-64, version 1 (SYSV),
dynamically linked, interpreter /lib64/ld-linux-x86-64.so.2, for
GNU/Linux 2.6.24,
BuildID[sha1]=e6bc58d8305e173e600769315fd849e85d12c1d0, not
stripped
```

Yang jika dijalankan akan meminta flag dari inputan scanf kita. Coba buka dengan IDA.

```
int __cdecl main(int argc, const char **argv, const char **envp)
{
  int result; // eax@9
  __int64 v4; // rcx@9
  signed int i; // [sp+8h] [bp-38h]@1
  int v6; // [sp+Ch] [bp-34h]@1
  char v7[40]; // [sp+10h] [bp-30h]@1
  __int64 v8; // [sp+38h] [bp-8h]@1

  v8 = *MK_FP(__FS__, 40LL);
  printf("Flag: ", argv, envp);
  __isoc99_scanf(4196251LL, v7);
  v6 = 0;
  for ( i = 0; i <= 30; ++i )
  {
     if ( v7[i] == s[30 - i] )
     ++v6;
  }
  if ( v6 == 31 )
     puts("Correct!");
  else
     puts("Wrong!");
  result = 0;
  v4 = *MK_FP(__FS__, 40LL) ^ v8;
  return result;
}
```

Terlihat bahwa inputan kita, v7 akan dicek dengan array s. Namun urutannya terbalik. Dan dengan panjang 31, karena v6 harus bernilai 31. Kita coba lihat isi array s tersebut.

```
.data:0000000000601080 ; int s[]
.data:0000000000601080 s               dd 7Dh                  ; DATA
XREF: main+6Ar
.data:0000000000601084                 db  79h ; y
.data:0000000000601085                 db    0
.data:0000000000601086                 db    0
.data:0000000000601087                 db    0
.data:0000000000601088                 db  73h ; s
.data:0000000000601089                 db    0
.data:000000000060108A                 db    0
.data:000000000060108B                 db    0
.data:000000000060108C                 db  61h ; a
.data:000000000060108D                 db    0
```

```
.data:000000000060108E                db    0
.data:000000000060108F                db    0
.data:0000000000601090                db  65h ; e
.data:0000000000601091                db    0
.data:0000000000601092                db    0
.data:0000000000601093                db    0
.data:0000000000601094                db  5Fh ; _
.data:0000000000601095                db    0
.data:0000000000601096                db    0
.data:0000000000601097                db    0
.data:0000000000601098                db  73h ; s
.data:0000000000601099                db    0
.data:000000000060109A                db    0
.data:000000000060109B                db    0
.data:000000000060109C                db  69h ; i
.data:000000000060109D                db    0
.data:000000000060109E                db    0
.data:000000000060109F                db    0
.data:00000000006010A0                db  5Fh ; _
.data:00000000006010A1                db    0
.data:00000000006010A2                db    0
.data:00000000006010A3                db    0
.data:00000000006010A4                db  65h ; e
.data:00000000006010A5                db    0
.data:00000000006010A6                db    0
.data:00000000006010A7                db    0
.data:00000000006010A8                db  73h ; s
.data:00000000006010A9                db    0
.data:00000000006010AA                db    0
.data:00000000006010AB                db    0
.data:00000000006010AC                db  72h ; r
.data:00000000006010AD                db    0
.data:00000000006010AE                db    0
.data:00000000006010AF                db    0
.data:00000000006010B0                db  65h ; e
.data:00000000006010B1                db    0
.data:00000000006010B2                db    0
.data:00000000006010B3                db    0
.data:00000000006010B4                db  76h ; v
.data:00000000006010B5                db    0
.data:00000000006010B6                db    0
.data:00000000006010B7                db    0
.data:00000000006010B8                db  65h ; e
.data:00000000006010B9                db    0
.data:00000000006010BA                db    0
.data:00000000006010BB                db    0
.data:00000000006010BC                db  72h ; r
```

```
.data:00000000006010BD                db    0
.data:00000000006010BE                db    0
.data:00000000006010BF                db    0
.data:00000000006010C0                db  5Fh ; _
.data:00000000006010C1                db    0
.data:00000000006010C2                db    0
.data:00000000006010C3                db    0
.data:00000000006010C4                db  79h ; y
.data:00000000006010C5                db    0
.data:00000000006010C6                db    0
.data:00000000006010C7                db    0
.data:00000000006010C8                db  73h ; s
.data:00000000006010C9                db    0
.data:00000000006010CA                db    0
.data:00000000006010CB                db    0
.data:00000000006010CC                db  61h ; a
.data:00000000006010CD                db    0
.data:00000000006010CE                db    0
.data:00000000006010CF                db    0
.data:00000000006010D0                db  65h ; e
.data:00000000006010D1                db    0
.data:00000000006010D2                db    0
.data:00000000006010D3                db    0
.data:00000000006010D4                db  7Bh ; {
.data:00000000006010D5                db    0
.data:00000000006010D6                db    0
.data:00000000006010D7                db    0
.data:00000000006010D8                db  74h ; t
.data:00000000006010D9                db    0
.data:00000000006010DA                db    0
.data:00000000006010DB                db    0
.data:00000000006010DC                db  73h ; s
.data:00000000006010DD                db    0
.data:00000000006010DE                db    0
.data:00000000006010DF                db    0
.data:00000000006010E0                db  65h ; e
.data:00000000006010E1                db    0
.data:00000000006010E2                db    0
.data:00000000006010E3                db    0
.data:00000000006010E4                db  46h ; F
.data:00000000006010E5                db    0
.data:00000000006010E6                db    0
.data:00000000006010E7                db    0
.data:00000000006010E8                db  61h ; a
.data:00000000006010E9                db    0
.data:00000000006010EA                db    0
.data:00000000006010EB                db    0
```

```
.data:00000000006010EC                db  62h ; b
.data:00000000006010ED                db    0
.data:00000000006010EE                db    0
.data:00000000006010EF                db    0
.data:00000000006010F0                db  6Fh ; o
.data:00000000006010F1                db    0
.data:00000000006010F2                db    0
.data:00000000006010F3                db    0
.data:00000000006010F4                db  54h ; T
.data:00000000006010F5                db    0
.data:00000000006010F6                db    0
.data:00000000006010F7                db    0
.data:00000000006010F8                db  49h ; I
.data:00000000006010F9                db    0
.data:00000000006010FA                db    0
.data:00000000006010FB                db    0
```

Ok, terlihat ya dari bawah ke atas (terbalik) bahwa itu adalah flagnya.
Flag : **ITobaFest{easy_reverse_is_easy}**

Rev2

Diberikan sebuah file berformat .pyc yang merupakan file format python yang sudah ter-compile. Dengan menggunakan uncompyle6 kami mendapatkan hasil decompilenya.

```
$ uncompyle6 rev2.pyc
# uncompyle6 version 2.9.11
# Python bytecode 2.7 (62211)
# Decompiled from: Python 2.7.12 (default, Nov 19 2016, 06:48:10)
# [GCC 5.4.0 20160609]
# Embedded file name: medium.py
# Compiled at: 2017-05-08 00:00:46
exec 'import re;import base64'
exec (lambda p, y: (lambda o, b, f: re.sub(o, b, f))('([0-9a-f]+)',
lambda m: p(m, y),
base64.b64decode('NSAzNwoKMWQgPSBbMjgsIDJjLCAyYiwgMWIsIDMwLCAxNCwgZ
SwgZSwgMjksIDJhLCBjLCAzNiwgMTMsIDI1LCAyZiwgMzEsIDcsIDM1LCAyMSwgMTYs
IDgsIDMyLCAyMSwgMTUsIGQsIGQsIDEyLCA2LCBjLCAyMywgMzQsIDgsIDE3LCA2LCA
yNywgMjQsIDMzLCAxMiwgMmUsIDJkLCA3LCAxYywgMjYsIDIyXQoKMTAgPSAiIgoxOC
AxZSAxZiBhKDAsIDE5KDFkKSk6CgkxMCA9IDEwICsgMWEoMWUgXiAxZFsxZV0pCgpmI
D0gMygiMTE6ICIpCjIwICgzNy4yKDEwKSA9PSBmKToKICAxICI0ISIKYjoKICAxICI5
ISI=')))(lambda a, b: b[int('0x' + a.group(1), 16)],
'0|print|b64decode|raw_input|Correct|import|119|114|112|Wrong|range
|else|68|65|64|f|s|Flag|72|105|104|125|126|122|for|len|chr|117|113|
t|i|in|if|88|22|46|89|63|26|27|83|82|81|80|87|85|96|92|93|99|77|70|
76|35|59|base64'.split('|'))
```

```
# okay decompiling rev2.pyc
```

Ternyata hasil decompile-nya di-obfuscate, mudahnya kami hanya tinggal mengganti ‘exec’ kedua dengan fungsi ‘print’ agar string source codenya terprint

```
exec 'import re;import base64'
print (lambda p, y: (lambda o, b, f: re.sub(o, b,
f))('([0-9a-f]+)', lambda m: p(m, y),
base64.b64decode('NSAzNwoKMWQgPSBbMjgsIDJjLCAyYiwgMWIsIDMwLCAxNCwgZ
SwgZSwgMjksIDJhLCBjLCAzNiwgMTMsIDI1LCAyZiwgMzEsIDcsIDM1LCAyMSwgMTYs
IDgsIDMyLCAyMSwgMTUsIGQsIGQsIDEyLCA2LCBjLCAyMywgMzQsIDgsIDE3LCA2LCA
yNywgMjQsIDMzLCAxMiwgMmUsIDJkLCA3LCAxYywgMjYsIDIyXQoKMTAgPSAiIgoxOC
AxZSAxZiBhKDAsIDE5KDFkKSk6CgkxMCA9IDEwICsgMWEoMWUgXiAxZFsxZV0pCgpmI
D0gMygiMTE6ICIpCjIwICgzNy4yKDEwKSA9PSBmKToKICAxICI0ISIKYjoKICAxICI5
ISI=')))(lambda a, b: b[int('0x' + a.group(1), 16)],
'0|print|b64decode|raw_input|Correct|import|119|114|112|Wrong|range
|else|68|65|64|f|s|Flag|72|105|104|125|126|122|for|len|chr|117|113|
t|i|in|if|88|22|46|89|63|26|27|83|82|81|80|87|85|96|92|93|99|77|70|
76|35|59|base64'.split('|'))
```

Sehingga hasilnya:

```
import base64

t = [83, 87, 80, 117, 93, 104, 64, 64, 82, 81, 68, 59, 105, 63, 92,
99, 114, 35, 88, 126, 112, 77, 88, 125, 65, 65, 72, 119, 68, 46,
76, 112, 122, 119, 27, 89, 70, 72, 96, 85, 114, 113, 26, 22]

s = ""
for i in range(0, len(t)):
     s = s + chr(i ^ t[i])

f = raw_input("Flag: ")
if (base64.b64decode(s) == f):
  print "Correct!"
else:
  print "Wrong!"
```

Flagnya tinggal di print

```
import base64
```

```
t = [83, 87, 80, 117, 93, 104, 64, 64, 82, 81, 68, 59, 105, 63, 92,
99, 114, 35, 88, 126, 112, 77, 88, 125, 65, 65, 72, 119, 68, 46,
76, 112, 122, 119, 27, 89, 70, 72, 96, 85, 114, 113, 26, 22]

s = ""
for i in range(0, len(t)):
     s = s + chr(i ^ t[i])

print base64.b64decode(s)
```

Flag: **ITobaFest{deobfuscate_the_snake}**

Rev3

Diberikan sebuah binary

```
rev3: ELF 64-bit LSB executable, x86-64, version 1 (SYSV),
dynamically linked, interpreter /lib64/ld-linux-x86-64.so.2, for
GNU/Linux 2.6.24,
BuildID[sha1]=e34b346ae24465d98bfce358395a1ddfd5d4ec71, stripped
```

Sama seperti sebelumnya, diminta flag dari scanf juga. Oke langsung saja dibuka dengan IDA.

```
int sub_400677()
{
  int v0; // eax@2
  int result; // eax@3
  int v2; // eax@7
  int v3; // eax@12
  int v4; // eax@17
  int v5; // eax@22
  __int64 v6; // rbx@27
  int i; // [sp+8h] [bp-48h]@1
  int j; // [sp+8h] [bp-48h]@6
  int k; // [sp+8h] [bp-48h]@11
  int l; // [sp+8h] [bp-48h]@16
  int m; // [sp+8h] [bp-48h]@21
  int v12; // [sp+Ch] [bp-44h]@1
  char v13[40]; // [sp+10h] [bp-40h]@1
  __int64 v14; // [sp+38h] [bp-18h]@1

  v14 = *MK_FP(__FS__, 40LL);
  printf("Flag: ");
  __isoc99_scanf(4196651LL, v13);
```

```
v12 = 0;
for ( i = 0; i < dword_601114; ++i )
{
    v0 = v12++;
    if ( dword_601060[i] != v13[v0] )
    {
    result = sub_40064D();
    goto LABEL_27;
    }
}
for ( j = 0; j < dword_601118; ++j )
{
    v2 = v12++;
    if ( dword_601080[j] != (v13[v2] ^ dword_601060[j]) )
    {
    result = sub_40064D();
    goto LABEL_27;
    }
}
for ( k = 0; k < dword_60111C; ++k )
{
    v3 = v12++;
    if ( dword_6010A0[k] != (v13[v3] ^ dword_601080[k]) )
    {
    result = sub_40064D();
    goto LABEL_27;
    }
}
for ( l = 0; l < dword_601120; ++l )
{
    v4 = v12++;
    if ( dword_6010C0[l] != (v13[v4] ^ dword_601080[l]) )
    {
    result = sub_40064D();
    goto LABEL_27;
    }
}
for ( m = 0; m < dword_601124; ++m )
{
    v5 = v12++;
    if ( dword_6010E0[m] != (v13[v5] ^ dword_601100[m]) )
    {
    result = sub_40064D();
    goto LABEL_27;
    }
}
result = sub_400662();
```

```
LABEL_27:
  v6 = *MK_FP(__FS__, 40LL) ^ v14;
  return result;
}
```

Jika dilihat, terdapat beberapa looping yang mengolah inputan kita dengan operasi xor. Fungsi xor adalah reversible, artinya jika a ^ b = c, maka b ^ c = a. Ok, dengan menelusuri setiap looping tersebut, kita dapat mengembalikan kembali inputan apa yang seharusnya dimasukkan agar dapat menjadi benar inputan kita tersebut. Dan juga, sudah jelas apa isi variable array tujuannya agar menjadi benar. Dengan begitu, didapatlah flagnya.

Flag : **ITobaFest{ok_this_is_not_hard}**

Rev4

Diberikan sebuah binary

```
easy: ELF 32-bit LSB executable, Intel 80386, version 1 (SYSV), for
GNU/Linux 2.6.32, dynamically linked, interpreter \004, stripped
```

Binary tersebut jika dijalankan akan langung akan bernilai salah. Coba dibuka dengan IDA untuk didapatkan hasil decompile kode C nya.

Berikut kodenya yang penting. Terdapat 2 fungsi.

```
int __cdecl sub_804872B(int a1, int a2)
{
  FILE *stream; // [sp+8h] [bp-10h]@1
  char v4; // [sp+Fh] [bp-9h]@1

  v4 = 0;
  snprintf(ptr, 0x100u, "%s.txt", *(_DWORD *)a2);
  stream = fopen(ptr, "r");
  if ( stream )
  {
     memset(ptr, 0, 0x100u);
     fread(ptr, 1u, 0x14u, stream);
     fclose(stream);
     v4 = sub_804854B(ptr);
  }
  if ( v4 != 1 )
     puts("Masih salah, silakan coba lagi...");
  return 0;
```

```
}
```

Pada fungsi tersebut akan dibuka sebuah file .txt dan dibaca isinya. Dilakukan pengecekan dengan fungsi berikut.

```
int __cdecl sub_804854B(int a1)
{
  char v2; // [sp+Bh] [bp-Dh]@1
  signed int i; // [sp+Ch] [bp-Ch]@21

  v2 = 0;
  if ( *(_BYTE *)(a1 + 19) == 33
      && *(_BYTE *)(a1 + 5) == 118
      && *(_BYTE *)(a1 + 9) == 48
      && *(_BYTE *)(a1 + 15) == 117
      && (*(_BYTE *)(a1 + 1) == 109 || *(_BYTE *)(a1 + 18) == 109)
      && (*(_BYTE *)(a1 + 3) == 55 || *(_BYTE *)(a1 + 8) == 55)
      && (*(_BYTE *)(a1 + 4) == 95 || *(_BYTE *)(a1 + 13) == 95)
      && (*(_BYTE *)(a1 + 6) == 49 || *(_BYTE *)(a1 + 11) == 49)
      && (*(_BYTE *)(a1 + 7) == 99 || *(_BYTE *)(a1 + 14) == 99)
      && (*(_BYTE *)(a1 + 10) == 114 || *(_BYTE *)(a1 + 16) == 114)
      && (*(_BYTE *)a1 == 52 || *(_BYTE *)(a1 + 2) == 52 || *(_BYTE
*)(a1 + 12) == 52 || *(_BYTE *)(a1 + 17) == 52) )
  {
      for ( i = 0; i <= 7; ++i )
      {
      dword_804A060[i] ^= 0xCAFEBABE;
      printf(
      "%c%c%c%c",
      (unsigned int)dword_804A060[i] >> 24,
      (unsigned int)dword_804A060[i] >> 16,
      (unsigned int)dword_804A060[i] >> 8,
      (unsigned __int8)dword_804A060[i]);
      }
      v2 = 1;
  }
  return (unsigned __int8)v2;
}
```

Pada fungsi tersebut, terlihat bahwa ada nilai ASCII untuk setiap karakternya yang kemudian bernilai 4m47_v1c70r14_cur4m!. Namun jika dimasukkan sebagai flag, tetap salah. Lalu dilihat lebih jauh, ternyata jika benar isi dari file tersebut, akan menjalankan printah printf. Akhirnya, langsung dicoba saja mereplika fungsi tersebut agar jawabannya dapat pasti. Kemungkinan itu adalah flagnya. Berikut scriptnya dalam bahasa C.

```
#include<stdio.h>

int main()
{
    int i = 0;
    int a[] = {0x83AAD5DC,0x0ABB8DFCD,0xBE85D7DF,0xB88DD3CE,
0xAB8ACFCC,0xAFA1D2CB,0xBE9FD4DF,0x959CDFC3};

    for(i = 0; i < 8; ++i)
    {
      a[i] ^= 0xCAFEBABE;
      printf("%c%c%c%c", a[i] >> 24, a[i] >> 16, a[i] >> 8, a[i]);
    }
    return 0;
}
```

Flag : **ITobaFest{marsipature_hutana_be}**

Rev5

Diberikan sebuah binary

```
medium: ELF 32-bit LSB executable, Intel 80386, version 1 (SYSV),
for GNU/Linux 2.6.32, dynamically linked, interpreter \004,
stripped
```

Jika dijalankan, akan keluar komen salah. Langsung decompile dengan IDA. Didapatkan fungsi mainnya seperti berikut.

```
int __cdecl main(int argc, const char **argv, const char **envp)
{
  int result; // eax@7
  void *thread_return; // [sp+0h] [bp-1Ch]@3
  int v5; // [sp+4h] [bp-18h]@3
  int v6; // [sp+8h] [bp-14h]@3
  int v7; // [sp+Ch] [bp-10h]@3
  int i; // [sp+10h] [bp-Ch]@3
  int *v9; // [sp+18h] [bp-4h]@1

  v9 = &argc;
  if ( argc != 2 || strlen(argv[1]) != 34 )
  {
```

```
LABEL_8:
      puts("Silakan coba lagi...");
      result = 1;
  }
  else
  {
      memset(dest, 0, 0x24u);
      strncpy(dest, argv[1], 0x22u);
      pthread_create(&th, 0, start_routine, 0);
      pthread_create(&dword_804A074, 0, sub_80486BF, 0);
      pthread_create(&dword_804A078, 0, sub_8048765, 0);
      pthread_create(&dword_804A07C, 0, sub_804880B, 0);
      pthread_join(th, &thread_return);
      pthread_join(dword_804A074, (void **)&v5);
      pthread_join(dword_804A078, (void **)&v6);
      pthread_join(dword_804A07C, (void **)&v7);
      for ( i = 0; i <= 3; ++i )
      {
      if ( !*(_DWORD *)*(&thread_return + i) )
      goto LABEL_8;
      }
      puts("Selamat, Anda benar...");
      result = 0;
  }
  return result;
}
```

Jika dilihat program meminta input melalui argv dengan panjang 34 karakter dan akan menghasilkan hasil yang benar setelah melalui pthread yang ada. Terlihat bahwa akan dibuat 4 pthread yang dijalankan bersamaan dan digabung lagi. Coba buka masing - masing fungsi tersebut.

Start_routine

```
void __noreturn start_routine()
{
  char v0; // [sp+Bh] [bp-1Dh]@1
  char v1; // [sp+Ch] [bp-1Ch]@1
  char v2; // [sp+Dh] [bp-1Bh]@1
  char v3; // [sp+Eh] [bp-1Ah]@1
  char v4; // [sp+Fh] [bp-19h]@1
  char v5; // [sp+10h] [bp-18h]@1
  char v6; // [sp+11h] [bp-17h]@1
  char v7; // [sp+12h] [bp-16h]@1
  char v8; // [sp+13h] [bp-15h]@1
  char v9; // [sp+14h] [bp-14h]@1
```

```
  char v10; // [sp+15h] [bp-13h]@1
  char v11; // [sp+16h] [bp-12h]@1
  char v12; // [sp+17h] [bp-11h]@1
  char v13; // [sp+18h] [bp-10h]@1
  char v14; // [sp+19h] [bp-Fh]@1
  char v15; // [sp+1Ah] [bp-Eh]@1
  char v16; // [sp+1Bh] [bp-Dh]@1
  int i; // [sp+1Ch] [bp-Ch]@1

  v0 = 4;
  v1 = 5;
  v2 = 6;
  v3 = 6;
  v4 = 6;
  v5 = 4;
  v6 = 6;
  v7 = 7;
  v8 = 7;
  v9 = 7;
  v10 = 7;
  v11 = 7;
  v12 = 6;
  v13 = 7;
  v14 = 6;
  v15 = 6;
  v16 = 5;
  dword_804A060 = 0;
  for ( i = 0; i <= 16; ++i )
  {
     if ( (char)(*(_BYTE *)(i + 134520960) >> 4) != (unsigned
__int8)*(&v0 + i) )
     goto LABEL_6;
  }
  dword_804A060 = 1;
LABEL_6:
  pthread_exit(&dword_804A060);
}
```

sub_80486BF

```
void __noreturn sub_80486BF()
{
  char v0; // [sp+Bh] [bp-1Dh]@1
  char v1; // [sp+Ch] [bp-1Ch]@1
  char v2; // [sp+Dh] [bp-1Bh]@1
  char v3; // [sp+Eh] [bp-1Ah]@1
```

```
  char v4; // [sp+Fh] [bp-19h]@1
  char v5; // [sp+10h] [bp-18h]@1
  char v6; // [sp+11h] [bp-17h]@1
  char v7; // [sp+12h] [bp-16h]@1
  char v8; // [sp+13h] [bp-15h]@1
  char v9; // [sp+14h] [bp-14h]@1
  char v10; // [sp+15h] [bp-13h]@1
  char v11; // [sp+16h] [bp-12h]@1
  char v12; // [sp+17h] [bp-11h]@1
  char v13; // [sp+18h] [bp-10h]@1
  char v14; // [sp+19h] [bp-Fh]@1
  char v15; // [sp+1Ah] [bp-Eh]@1
  char v16; // [sp+1Bh] [bp-Dh]@1
  int i; // [sp+1Ch] [bp-Ch]@1

  v0 = 6;
  v1 = 6;
  v2 = 7;
  v3 = 6;
  v4 = 6;
  v5 = 6;
  v6 = 7;
  v7 = 7;
  v8 = 5;
  v9 = 7;
  v10 = 6;
  v11 = 7;
  v12 = 6;
  v13 = 6;
  v14 = 6;
  v15 = 6;
  v16 = 7;
  dword_804A064 = 0;
  for ( i = 0; i <= 16; ++i )
  {
     if ( (char)(dest[i + 17] >> 4) != (unsigned __int8)*(&v0 + i)
)
     goto LABEL_6;
  }
  dword_804A064 = 1;
LABEL_6:
  pthread_exit(&dword_804A064);
}
```

sub_8048765

```
void __noreturn sub_8048765()
```

```
{
  char v0; // [sp+Bh] [bp-1Dh]@1
  char v1; // [sp+Ch] [bp-1Ch]@1
  char v2; // [sp+Dh] [bp-1Bh]@1
  char v3; // [sp+Eh] [bp-1Ah]@1
  char v4; // [sp+Fh] [bp-19h]@1
  char v5; // [sp+10h] [bp-18h]@1
  char v6; // [sp+11h] [bp-17h]@1
  char v7; // [sp+12h] [bp-16h]@1
  char v8; // [sp+13h] [bp-15h]@1
  char v9; // [sp+14h] [bp-14h]@1
  char v10; // [sp+15h] [bp-13h]@1
  char v11; // [sp+16h] [bp-12h]@1
  char v12; // [sp+17h] [bp-11h]@1
  char v13; // [sp+18h] [bp-10h]@1
  char v14; // [sp+19h] [bp-Fh]@1
  char v15; // [sp+1Ah] [bp-Eh]@1
  char v16; // [sp+1Bh] [bp-Dh]@1
  int i; // [sp+1Ch] [bp-Ch]@1

  v0 = 9;
  v1 = 4;
  v2 = 15;
  v3 = 2;
  v4 = 1;
  v5 = 6;
  v6 = 5;
  v7 = 3;
  v8 = 4;
  v9 = 11;
  v10 = 4;
  v11 = 5;
  v12 = 13;
  v13 = 0;
  v14 = 1;
  v15 = 12;
  v16 = 15;
  dword_804A068 = 0;
  for ( i = 0; i <= 16; ++i )
  {
     if ( (*(_BYTE *)(i + 134520960) & 0xF) != *(&v0 + i) )
     goto LABEL_6;
  }
  dword_804A068 = 1;
LABEL_6:
  pthread_exit(&dword_804A068);
}
```

sub_804880B

```
void __noreturn sub_804880B()
{
  char v0; // [sp+Bh] [bp-1Dh]@1
  char v1; // [sp+Ch] [bp-1Ch]@1
  char v2; // [sp+Dh] [bp-1Bh]@1
  char v3; // [sp+Eh] [bp-1Ah]@1
  char v4; // [sp+Fh] [bp-19h]@1
  char v5; // [sp+10h] [bp-18h]@1
  char v6; // [sp+11h] [bp-17h]@1
  char v7; // [sp+12h] [bp-16h]@1
  char v8; // [sp+13h] [bp-15h]@1
  char v9; // [sp+14h] [bp-14h]@1
  char v10; // [sp+15h] [bp-13h]@1
  char v11; // [sp+16h] [bp-12h]@1
  char v12; // [sp+17h] [bp-11h]@1
  char v13; // [sp+18h] [bp-10h]@1
  char v14; // [sp+19h] [bp-Fh]@1
  char v15; // [sp+1Ah] [bp-Eh]@1
  char v16; // [sp+1Bh] [bp-Dh]@1
  int i; // [sp+1Ch] [bp-Ch]@1

  v0 = 4;
  v1 = 15;
  v2 = 2;
  v3 = 9;
  v4 = 1;
  v5 = 14;
  v6 = 5;
  v7 = 3;
  v8 = 15;
  v9 = 0;
  v10 = 1;
  v11 = 2;
  v12 = 4;
  v13 = 5;
  v14 = 4;
  v15 = 5;
  v16 = 13;
  dword_804A06C = 0;
  for ( i = 0; i <= 16; ++i )
  {
     if ( (dest[i + 17] & 0xF) != *(&v0 + i) )
     goto LABEL_6;
  }
```

```
  dword_804A06C = 1;
LABEL_6:
  pthread_exit(&dword_804A06C);
}
```

Jika dilihat, fungsi start_routine akan berpadanan dengan sub_8048765, sementara sub_80486BF akan berpadanan dengan sub_804880B. Hal ini dikarenakan variable yang digunakan sama dan thread tersebut jalan secara bersamaan. Sebagai contoh, untuk sub_routine dan sub_8048765, `if ( (*(_BYTE *)(i + 134520960) & 0xF) != *(&v0 + i) )` dan `if ( (char)(*(_BYTE *)(i + 134520960) >> 4) != (unsigned __int8)*(&v0 + i) ).` Artinya bahwa kondisi tersebut harus terpenuhi. Begitu pula untuk padanan sub_80486BF dan sub_804880B. Fungsi - fungsi tersebut menjalankan hal yang sama namun untuk 17 karakter pertama, dan juga 17 karakter berikutnya.

Cara termudah adalah dengan melakukan bruteforce. Oleh karena itu, dengan memperhatikan kondisi tersebut, dibuat script yang paling sederhana dengan python.

```
#!/usr/bin/python

a = [4,5,6,6,6,4,6,7,7,7,7,7,6,7,6,6,5]
b = [9,4,15,2,1,6,5,3,4,11,4,5,13,0,1,12,15]
for j in range(17):
    for i in range(255):
      if (i >> 4 == a[j]) and (i & 0xF == b[j]):
           print(chr(i), end="")

a = [6, 6, 7, 6, 6, 6, 7, 7, 5, 7, 6, 7, 6, 6, 6, 6, 7]
b = [4, 15, 2, 9, 1, 14, 5, 3, 15, 0, 1, 2, 4, 5, 4, 5, 13]

for j in range(17):
    for i in range(255):
      if (i >> 4 == a[j]) and (i & 0xF == b[j]):
           print(chr(i), end="")
```

Didapatkan
Flag : **ITobaFest{tumpal_dorianus_pardede}**

-|Stegano|-

Morning Glory Pool

Diberikan sebuah gambar dengan hint : Perhatikan Gambar,LETAK tanpa koma dan simbol.

Kita mencoba membuka file gambar dengan

```
$ strings morning-glory-pool.jpg | grep AA
BBAAAAABAAABABBABABBABBBABABBABAABABAABBABBBAABBABAABAABAABA
AAAABABAAAAABBA AAAABBAAABABBBABAABBAABBBAABAABAAAB
```

Didapatkan sebuah string Bacon Chiper, decrypt di web
http://rumkin.com/tools/cipher/baconian.php

Decrypt

Distinct codes

Your message: (Swap A and B)

BBAAAAABAAABABBABABBABBBABABBABAABABAABBABBBAABBABAABAABAABA AAAABABAAAAABBA
AAAABBAAABABBBABAABBAABBBAABAABAAAB

This is your encoded or decoded text:

YELLOWSTONES BIG BROTHER

Didapatkan string YELLOWSTONES BIG BROTHER

Dengan skill Google-Fu saya mendapatkan beberapa informasi yang berkaitan dengan YELLOWSTONES BIG BROTHER , Ternyata YELLOWSTONES BIG BROTHER = Danau Toba

Hint dari soal adalah Perhatikan Gambar,**LETAK tanpa koma** dan **simbol**.

Maka kita coba cari yang berkaitan dgn **LETAK** dari danau Toba

Koordinat : 3,58°LU 98,67°BT

Flag **tanpa koma** dan **simbol**
Flag: **ITF{358LU9867BT}**

-|Web|-

Web1

Diberikan sebuah target (http://ctf.itobafest.del.ac.id/source/1/), kami lihat source html-nya. Terdapat string

```
admin:&lt;~1bpas1M/UU0P34T3+4g%2)d&lt;N1,h%%@:_H.@q@PQA25u$3FX^O~&gt;
```

Diduga string tersebut merupakan credential untuk mendapatkan flag. Dengan format [username]:[password].

```
username: admin
password: &lt;~1bpas1M/UU0P34T3+4g%2)d&lt;N1,h%%@:_H.@q@PQA25u$3FX^O~&gt;
```

Namun ternyata passwordnya salah, disinilah skill TEBAK-TEBAKAN kami diuji.

```
&lt;~1bpas1M/UU0P34T3+4g%2)d&lt;N1,h%%@:_H.@q@PQA25u$3FX^O~&gt;
```

Terlihat ada karakter yang ter-encode HTML, setelah di-decode

```
<~1bpas1M/UU0P34T3+4g%2)d<N1,h%%@:_H.@q@PQA25u$3FX^O~>
```

Namun tetap salah, string apakah itu? Hasrat **tebak-tebakan** kami muncul yang sudah lama terpendam dalam lubuk hati, perasaan kesal pun membara… didapatkan bahwa itu merupakan ASCII85!!

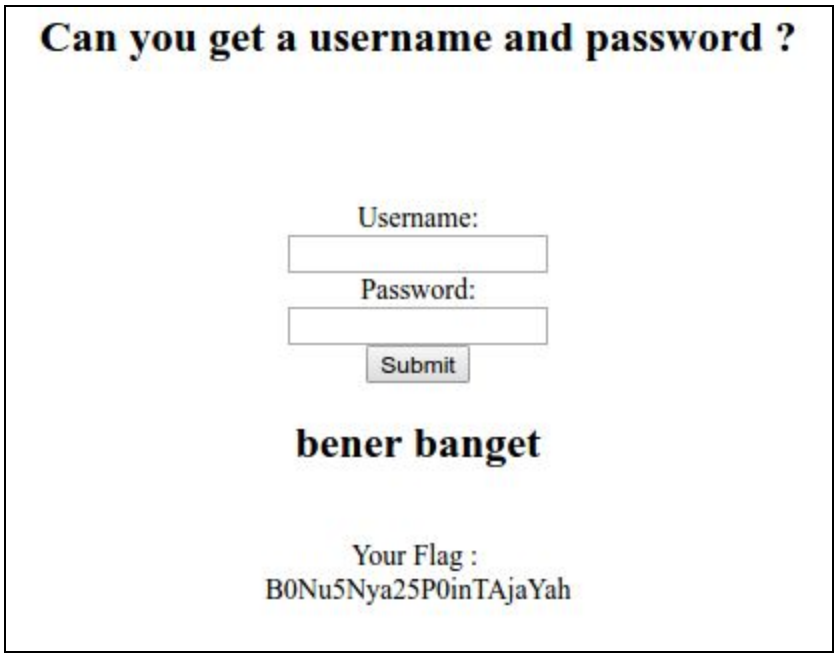
Setelah di-decode didapatkan sebuah string hexadecimal

```
420f3f8b0f6f8a915738274fae9bce61d2489b1a
```

Setelah ditebak string itu tipe hash SHA1 karena 40 karakter. Kami decrypt menggunakan hashkiller.co.uk .

Lalu masukkan credential

username: admin
password: kambing



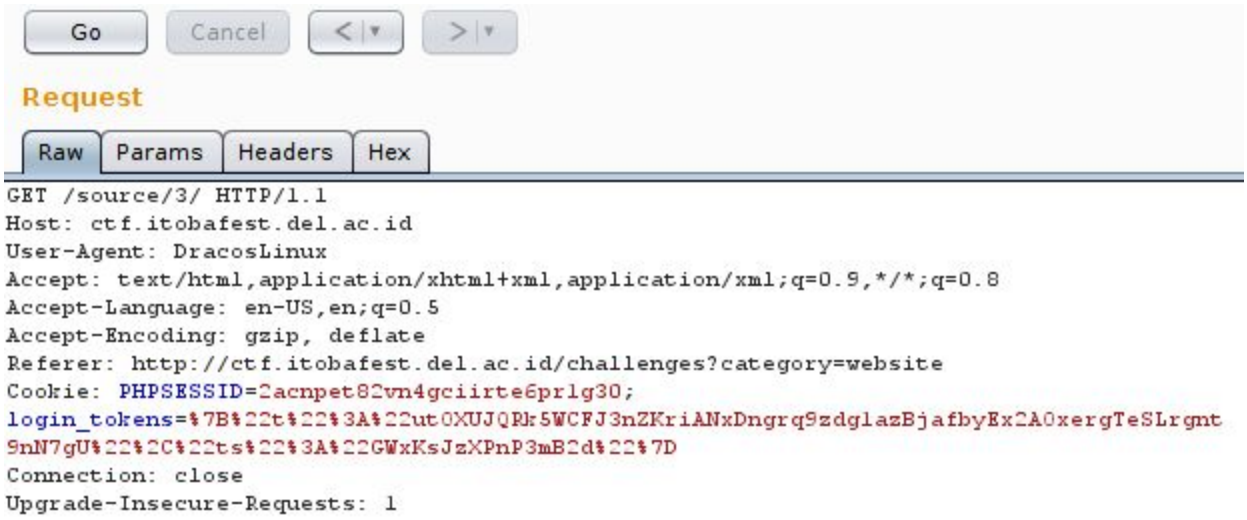
Flag: **ITF{B0Nu5Nya25P0inTAjaYah}**

Web3

Diberikan sebuah target (http://ctf.itobafest.del.ac.id/source/3/), kami lihat ada string user agent harus DracosLinux untuk mendapatkan flag.

Pakai tools Burpsuite untuk mengganti UserAgent

```
$(Cyber Security IPB - Pwning The World)
```



Flag : **ITF{U5eRA6entMakeY0uCo0l}**