

Міністерство освіти і науки України
Національний технічний університет України
"Київський політехнічний інститут імені Ігоря Сікорського"
Фізико-технічний інститут

ЗВОРОТНА РОЗРОБКА ТА АНАЛІЗ ШКІДЛИВОГО ЗАБЕЗПЕЧЕННЯ

Лабораторна робота №2
Засоби автоматизації аналізу
Варіант 19

Виконала:
студентка 3 курсу
гр. ФБ-92
Шатковська Діана

Перевірив:
Якобчук Д.І.

Засоби автоматизації аналізу

Мета роботи:

Отримати навички автоматизації методів аналізу програмного коду.

Хід роботи

- Реалізуйте статичний деобфускатор для Вашого варіанту(19 - x86/alpha_upper), розділ 2.3.2.

Створимо текстовий файл і запишемо туди будь-який рядок тексту.

```
└─(kali㉿kali)-[~/RE_labs/lab2]
└─$ cat sc
Bibr pun: oh no our server is being flooded! dam it.
```

Деобфускуємо цей файл за допомогою даного обфускатора:

```
└─(kali㉿kali)-[~/RE_labs/lab2]
└─$ msfvenom -p generic/custom payloadfile=sc -f raw -o payload.bin -e x86/alpha_upper
[-] No platform was selected, choosing Msf::Module::Platform from the payload
[-] No arch selected, selecting arch: x86 from the payload
Found 1 compatible encoders
Attempting to encode payload with 1 iterations of x86/alpha_upper
x86/alpha_upper succeeded with size 1432 (iteration=0)
x86/alpha_upper chosen with final size 1432
Payload size: 1432 bytes
Saved as: payload3.bin
```

Спробуємо переглянути вміст отриманого файлу, а також його хексдамп:

```
└─(kali㉿kali)-[~/RE_labs/lab2]
└─$ cat payload.bin
◆◆◆◆◆w◆^VYIIIICCCCCQZVTX30VX4AP0A3HH0A00ABAABTAAQ2AB2BB0BBXP8ACJJIPBU92B2R7PT0D5BN7JWPBOU8
WP2NRO10ROT53BQ03CSUSBVSUBR102I3C10BBE559RNE7GPCV2L2OBOD2EBD11Q02DE1BM10RI2TVNDJAA
```

```
└─(kali㉿kali)-[~/RE_labs/lab2]
└─$ hexdump -C payload.bin
1 ↗
00000000  89 e7 db cc d9 77 f4 5e  56 59 49 49 49 49 49 43 43 |.....w.^VYIIIIICC|
00000010  43 43 43 43 51 5a 56 54  58 33 30 56 58 34 41 50 |CCCCQZVTX30VX4AP|
00000020  30 41 33 48 48 30 41 30  30 41 42 41 41 42 54 41 |0A3HH0A00ABAABTA|
00000030  41 51 32 41 42 32 42 42  30 42 42 58 50 38 41 43 |AQ2AB2BB0BBXP8ACI|
00000040  4a 4a 49 50 42 55 39 32  42 32 52 37 50 54 30 44 |JJIPBU92B2R7PT0D|
00000050  35 42 4e 37 4a 57 50 42  4f 55 38 57 50 32 4e 52 |5BN7JWPBOU8WP2NR|
00000060  4f 31 30 52 4f 54 35 33  42 51 30 33 43 53 55 53 |O10ROT53BQ03CSUS|
00000070  42 42 56 53 55 42 52 31  30 32 49 33 43 31 30 42 |BBVSUBR102I3C10B|
00000080  42 45 35 35 39 52 4e 45  37 47 50 43 56 32 4c 32 |BE559RNE7GPCV2L2|
00000090  4f 42 4f 42 44 32 45 42  44 31 31 51 30 32 44 45 |OBOBD2EBD11Q02DE|
000000a0  31 42 4d 31 30 52 49 32  54 56 4e 44 4a 41 41 |1BM10RI2TVNDJAA|
000000af
```

Тепер розглянемо дизасемблізований payload.bin :

```
└─(kali㉿kali)-[~/RE_labs/lab2]
└─$ cat disasm_payload1
1 ↗
00000000  89E7          mov edi,esp
00000002  DBCC          fcmove st4
00000004  D977F4        fnstenv [edi-0xc]
00000007  5E             pop esi
00000008  56             push esi
00000009  59             pop ecx
```

```

0000000A 49          dec ecx
0000000B 49          dec ecx
0000000C 49          dec ecx
0000000D 49          dec ecx
0000000E 43          inc ebx
0000000F 43          inc ebx
00000010 43          inc ebx
00000011 43          inc ebx
00000012 43          inc ebx
00000013 43          inc ebx
00000014 51          push ecx
00000015 5A          pop edx
00000016 56          push esi
00000017 54          push esp
00000018 58          pop eax
00000019 3330        xor esi,[eax]
0000001B 56          push esi
0000001C 58          pop eax
0000001D 3441        xor al,0x41
0000001F 50          push eax
00000020 304133      xor [ecx+0x33],al
00000023 48          dec eax
00000024 48          dec eax
00000025 304130      xor [ecx+0x30],al
00000028 304142      xor [ecx+0x42],al
0000002B 41          inc ecx
0000002C 41          inc ecx
0000002D 42          inc edx
0000002E 54          push esp
0000002F 41          inc ecx
00000030 41          inc ecx
00000031 51          push ecx
00000032 324142      xor al,[ecx+0x42]
00000035 324242      xor al,[edx+0x42]
00000038 304242      xor [edx+0x42],al
0000003B 58          pop eax
0000003C 50          push eax
0000003D 384143      cmp [ecx+0x43],al
00000040 4A          dec edx
00000041 4A          dec edx
00000042 49          dec ecx
00000043 50          push eax
00000044 ...

```

Проаналізувавши даний код, на~~котимо~~ статичний дизасемблер на Python(файл додається):

```

from sys import byteorder
from pwn import *

context.arch = 'i386'

for file in sys.argv[1:]:
    print("Analysing", file)
    filedata = bytearray(read(file))
    result = filedata
    # print(filedata)
    # print(hexdump(result))

    for i in range(0x0, len(filedata), 0x1):
        try:
            result[0x41+i] = result[0x42+2*i]^(result[0x41+2*i]*0x10)%0x100
        except:
            print('Yay =^.^=')
            break

```

```

# print(hexdump(result))

end = result.find(b'\n')
print(result[0x42:end].decode())

```

Результат із покроковим виведенням хексдампу:

```

└─(kali㉿kali)-[~/RE_labs/lab2]
└─$ python3 static_alpha_upper.py payload.bin
Analysing payload.bin
00000000  89 e7 db cc d9 77 f4 5e  56 59 49 49  49 49 43 43 | . . . . w . ^ | VYII | IICC
00000010  43 43 43 43 51 5a 56 54  58 33 30 56  58 34 41 50 | CCCC | QZVT | X30V | X4AP
00000020  30 41 33 48 48 30 41 30  30 41 42 41  41 42 54 41 | 0A3H | H0A0 | 0ABA | ABTA
00000030  41 51 32 41 42 32 42 42  30 42 42 58  50 38 41 43 | AQ2A | B2BB | 0BBX | P8AC
00000040  4a e9 49 50 42 55 39 32  42 32 52 37  50 54 30 44 | J · IP | BU92 | B2R7 | PT0D
00000050  35 42 4e 37 4a 57 50 42  4f 55 38 57  50 32 4e 52 | 5BN7 | JWPB | OU8W | P2NR
00000060  4f 31 30 52 4f 54 35 33  42 51 30 33  43 53 55 53 | O10R | OT53 | BQ03 | CSUS
00000070  42 42 56 53 55 42 52 31  30 32 49 33  43 31 30 42 | BBVS | UBR1 | 02I3 | C10B
00000080  42 45 35 35 39 52 4e 45  37 47 50 43  56 32 4c 32 | BE55 | 9RNE | 7GPC | V2L2
00000090  4f 42 4f 42 44 32 45 42  44 31 31 51  30 32 44 45 | OBOB | D2EB | D11Q | 02DE
000000a0  31 42 4d 31 30 52 49 32  54 56 4e 44  4a 41 41 | 1BM1 | ORI2 | TVND | JAA |
000000af
00000000  89 e7 db cc d9 77 f4 5e  56 59 49 49  49 49 43 43 | . . . . w . ^ | VYII | IICC
00000010  43 43 43 43 51 5a 56 54  58 33 30 56  58 34 41 50 | CCCC | QZVT | X30V | X4AP
00000020  30 41 33 48 48 30 41 30  30 41 42 41  41 42 54 41 | 0A3H | H0A0 | 0ABA | ABTA
00000030  41 51 32 41 42 32 42 42  30 42 42 58  50 38 41 43 | AQ2A | B2BB | 0BBX | P8AC
00000040  4a e9 42 50 42 55 39 32  42 32 52 37  50 54 30 44 | J · EP | BU92 | B2R7 | PT0D
00000050  35 42 4e 37 4a 57 50 42  4f 55 38 57  50 32 4e 52 | 5BN7 | JWPB | OU8W | P2NR
00000060  4f 31 30 52 4f 54 35 33  42 51 30 33  43 53 55 53 | O10R | OT53 | BQ03 | CSUS
00000070  42 42 56 53 55 42 52 31  30 32 49 33  43 31 30 42 | BBVS | UBR1 | 02I3 | C10B
00000080  42 45 35 35 39 52 4e 45  37 47 50 43  56 32 4c 32 | BE55 | 9RNE | 7GPC | V2L2
00000090  4f 42 4f 42 44 32 45 42  44 31 31 51  30 32 44 45 | OBOB | D2EB | D11Q | 02DE
000000a0  31 42 4d 31 30 52 49 32  54 56 4e 44  4a 41 41 | 1BM1 | ORI2 | TVND | JAA |
000000af
...
00000000  89 e7 db cc d9 77 f4 5e  56 59 49 49  49 49 43 43 | . . . . w . ^ | VYII | IICC
00000010  43 43 43 43 51 5a 56 54  58 33 30 56  58 34 41 50 | CCCC | QZVT | X30V | X4AP
00000020  30 41 33 48 48 30 41 30  30 41 42 41  41 42 54 41 | 0A3H | H0A0 | 0ABA | ABTA
00000030  41 51 32 41 42 32 42 42  30 42 42 58  50 38 41 43 | AQ2A | B2BB | 0BBX | P8AC
00000040  4a e9 42 69 62 72 20 70  75 6e 3a 20  6f 68 20 6e | J · Bi | br p un: | oh n |
00000050  6f 20 6f 75 72 20 73 65  72 76 65 72  20 69 73 20 | o ou | r se | rver | is |
00000060  62 65 69 6e 67 20 66 6c  6f 6f 64 65  64 21 20 64 | bein | g fl | oode | d! d |
00000070  61 6d 20 69 74 2e 0a 51  30 32 49 33  43 31 30 42 | am i | t. | Q | 02I3 | C10B
00000080  42 45 35 35 39 52 4e 45  37 47 50 43  56 32 4c 32 | BE55 | 9RNE | 7GPC | V2L2
00000090  4f 42 4f 42 44 32 45 42  44 31 31 51  30 32 44 45 | OBOB | D2EB | D11Q | 02DE
000000a0  31 42 4d 31 30 52 49 32  54 56 4e 44  4a 41 41 | 1BM1 | ORI2 | TVND | JAA |
000000af

```

```

Yay =^.^=
Bibr pun: oh no our server is being flooded! dam it.

```

2. Реалізуйте динамічний деобфускатор для Вашого варіанту (19 - x86/alpha_upper), розділ 2.3.

Використаємо приклад емулятора, наданий у методичних вказівках _emu.py. А також трошки змодифікуємо його, щоб отримати hexdump даних після кожного кроку.

Код _emu.py (файл додається):

```
#!/usr/bin/env python3
from unicorn import *
from unicorn.x86_const import *
from capstone import *
from pwn import *
import sys

cs = Cs(CS_ARCH_X86, CS_MODE_32)
code = open(sys.argv[1], "rb").read()
address = 0

def hook_code(uc, address, size, user_data):
    global cs
    ins = uc.mem_read(address, size)
    print("hook called at 0x{:x}, instruction {}".format(address, ins.hex()))

    for i in cs.disasm(ins, 0):
        print("hook 0x{:03x} size {:2d}: {:03x}: {:20s} {} {}".format(address
            , size, address + i.address, i.bytes.hex(), i.mnemonic, i.op_str))
    print(hexdump(uc.mem_read(0, len(code))))
    print(uc.mem_read(0, len(code)))

def hook_syscall(mu, user_data):
    rax = mu.reg_read(UC_X86_REG_RAX)
    rdi = mu.reg_read(UC_X86_REG_RDI)
    if rax == 59:
        fn = mu.mem_read(rdi, 0x1000)
        fn = fn.split(b"\0") [0]
        fn = bytes(fn)
        print("SYS_execve {}".format(fn))
    else:
        print("syscall rax=0x{:x}, rdi=0x{:x}".format(rax, rdi))

mu = UC(UC_ARCH_X86, UC_MODE_32)
mu.mem_map(address, address + 0x2000)
mu.mem_write(address, code)
mu.reg_write(UC_X86_REG_ESP, address + 0x1000)
mu.hook_add(UC_HOOK_CODE, hook_code)
mu.hook_add(UC_HOOK_INSN, hook_syscall, None, 1, 0, UC_X86_INS_SYSCALL)

try:
    mu.emu_start(address, address + len(code))
except Exception:
    print("done.")
else:
    print("done.")
```

Результат виконання динамічного деобфускатора:

```
[kali㉿kali)-[~]
└─$ python3 _emy.py ~/RE_labs/lab2/payload.bin
```

```

...
hook called at 0x4d, instruction 68206e6f20
hook 0x04d size 5: 04d: 68206e6f20          push 0x206f6e20
00000000  89 e7 db cc d9 77 f4 5e 56 59 49 49 49 49 49 43 43 | . . . | .w.^|VYII|IICC|
00000010  43 43 43 43 51 5a 56 54 58 33 30 56 58 34 41 50 |CCCC|QZVT|X30V|X4AP|
00000020  30 41 33 48 48 30 41 30 30 41 42 41 41 42 6b 41 |0A3H|H0A0|0ABA|ABkA|
00000030  41 10 32 41 42 32 42 42 30 42 42 58 50 38 41 43 |A.2A|B2BB|0BBX|P8AC|
00000040  75 e9 42 69 62 72 20 70 75 6e 3a 20 6f 68 20 6e |u.Bi|br pun:|oh n|
00000050  6f 20 6f 75 72 20 73 65 72 76 65 72 20 69 73 20 |o ou|r se|rver| is |
00000060  62 65 69 6e 67 20 66 6c 6f 6f 64 65 64 21 20 64 |being f1|oode|d! d|
00000070  61 6d 20 69 74 2e 0a 31 30 32 49 33 43 31 30 42 |am i|t.·1|02I3|C10B|
00000080  42 45 35 35 39 52 4e 45 37 47 50 43 56 32 4c 32 |BE55|9RNE|7GPC|V2L2|
00000090  4f 42 4f 42 44 32 45 42 44 31 31 51 30 32 44 45 |OBOB|D2EB|D11Q|02DE|
000000a0  31 42 4d 31 30 52 49 32 54 56 4e 44 4a 41 41 |1BM1|0RI2|TVND|JAA|
000000af

bytearray(b'\x89\xe7\xdb\xcc\xd9w\xf4^VYIIICCCCCQZVTX30VX4AP0A3HH0A00ABAABkAA\x10
2AB2BB0BBXP8ACu\xe9Bi|br pun: oh no our server is being flooded! dam
it.\n102I3C10BBE559RNE7GPCV2L2OB0BD2EBD11Q02DE1BM10RI2TVNDJAA')
done.

```