

Trochę o mnie

- Eternal@freenode
- CTFy w drużynie Just Hit the Core
- github.com/Eterna1

Wstęp do RE

- Co to jest?
- Do czego się może przydać?
- Analiza statyczna a dynamiczna

Narzędzia

Do analizy statycznej:

Disassemblery: Ida Pro, binary
ninja

hex edytory

Do analizy dynamicznej:

Debuggery: gdb i pwndbg

Timeless debuggery: qira

ltrace, strace, wireshark

Qira - emulator

- Natywne wykonywanie kodu a emulator
- program pod debuggerem wykonuje się na procesorze
- plusy: pozwala na większą kontrolę nad instrukcjami (przykłady), odporny na sztuczki anty-dbg
- minusy: wolniejsze, malware to mogą wykorzystać

Kod programu

```
int main(){  
    printf("main start\n");  
    free(malloc(100));  
    printf("main end\n");  
    return 0;  
}
```

strace

```
directory)
stat("/usr/local/cuda/lib64/x86_64", 0x7ffc83bf05e0) = -1 ENOENT (No such file or directory)
open("/usr/local/cuda/lib64/libc.so.6", O_RDONLY|O_CLOEXEC) = -1 ENOENT (No such file or directory)
stat("/usr/local/cuda/lib64", {st_mode=S_IFDIR|0755, st_size=4096, ...}) = 0
open("/etc/ld.so.cache", O_RDONLY|O_CLOEXEC) = 3
fstat(3, {st_mode=S_IFREG|0644, st_size=225158, ...}) = 0
mmap(NULL, 225158, PROT_READ, MAP_PRIVATE, 3, 0) = 0x7fe1e5536000
close(3) = 0
access("/etc/ld.so.nohwcap", F_OK) = -1 ENOENT (No such file or directory)
open("/lib/x86_64-linux-gnu/libc.so.6", O_RDONLY|O_CLOEXEC) = 3
read(3, "\177ELF\2\1\1\0\0\0\0\0\0\0\0\0\3\0>\0\1\0\0\0P \2\0\0\0\0\0"... , 832) = 832
fstat(3, {st_mode=S_IFREG|0755, st_size=1840928, ...}) = 0
mmap(NULL, 3949248, PROT_READ|PROT_EXEC, MAP_PRIVATE|MAP_DENYWRITE, 3, 0) = 0x7fe1e4f88000
mprotect(0x7fe1e5142000, 2097152, PROT_NONE) = 0
mmap(0x7fe1e5342000, 24576, PROT_READ|PROT_WRITE, MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE, 3, 0x1ba000) = 0x7fe1e5342000
mmap(0x7fe1e5348000, 17088, PROT_READ|PROT_WRITE, MAP_PRIVATE|MAP_FIXED|MAP_ANONYMOUS, -1, 0) = 0x7fe1e5348000
close(3) = 0
mmap(NULL, 4096, PROT_READ|PROT_WRITE, MAP_PRIVATE|MAP_ANONYMOUS, -1, 0) = 0x7fe1e5535000
mmap(NULL, 8192, PROT_READ|PROT_WRITE, MAP_PRIVATE|MAP_ANONYMOUS, -1, 0) = 0x7fe1e5533000
arch_prctl(ARCH_SET_FS, 0x7fe1e5533740) = 0
mprotect(0x7fe1e5342000, 16384, PROT_READ) = 0
mprotect(0x600000, 4096, PROT_READ) = 0
mprotect(0x7fe1e556f000, 4096, PROT_READ) = 0
munmap(0x7fe1e5536000, 225158) = 0
fstat(1, {st_mode=S_IFCHR|0600, st_rdev=makedev(136, 8), ...}) = 0
mmap(NULL, 4096, PROT_READ|PROT_WRITE, MAP_PRIVATE|MAP_ANONYMOUS, -1, 0) = 0x7fe1e556c000
write(1, "main start\n", 11main start
) = 11
brk(0) = 0x1af4000
brk(0x1b15000) = 0x1b15000
write(1, "main end\n", 9main end
) = 9
exit_group(0) = ?
+++ exited with 0 +++
```


RE a assembler



Metody na uniknięcie analizy assemblera



1. dekompilator Ida Pro



Originalny kod

```
int fibonacci(int arg)
{
    if (arg<=1)
        return 1;
    else
        return fibonacci(arg-1)+fibonacci(arg-2);
}
```

Zdisassemblerowny

```
; Attributes: bp-based frame

public fibonacci
fibonacci proc near

var_14= dword ptr -14h

push    rbp
mov     rbp, rsp
push    rbx
sub     rsp, 18h
mov     [rbp+var_14], edi
cmp     [rbp+var_14], 1
jg      short loc_400546
```

```
mov     eax, 1
jmp     short loc_400564
```

```
loc_400546:
mov     eax, [rbp+var_14]
sub     eax, 1
mov     edi, eax
call    fibonacci
mov     ebx, eax
mov     eax, [rbp+var_14]
sub     eax, 2
mov     edi, eax
call    fibonacci
add     eax, ebx
```

```
loc_400564:
add     rsp, 18h
pop     rbx
pop     rbp
retn
fibonacci endp
```

zdekompilowany

```
1 signed __int64 __fastcall fibonacci(signed int arg1)
2 {
3     signed __int64 result; // rax@2
4     int v2; // ebx@3
5
6     if ( arg1 > 1 )
7     {
8         v2 = fibonacci((unsigned int)(arg1 - 1));
9         result = (unsigned int)(v2 + fibonacci((unsigned int)(arg1 - 2)));
10    }
11    else
12    {
13        result = 1LL;
14    }
15    return result;
16 }
```

2. monitorowanie API/syscall

- ltrace, strace
- ltrace demo

Kod crackme

```
void mangle(char *a, int len)
{
    for(int i=0;i<len;i++){
        a[i]=a[i]-10+i;
    }
}

int main()
{
    char passwd[20];
    printf("podaj haslo:");
    scanf("%s",passwd);

    mangle(passwd,strlen(passwd));

    if (!strcmp(passwd,"ffkrjZgliru"))
    {
        printf("udalo Ci sie!!!!!!\n");
    }
    else{
        printf("zle haslo :( \n");
    }
    return 0;
}
```

3. Lekkie zmodyfikowanie “działania programu”

- zmiana jakiegoś rejestru
- zmiana jakiejś flagi (to też jest rejestr)
- zmiana rejestru RIP (co to jest RIP? :))
- modyfikowanie pamięci programu: dane lub instrukcje
 - gdb i pwndbg demo

4. zmiany w pamięci programu

Obserwowanie zmian w pamięci wykonywanego programu

- qira demo

Zadanie move / Co to jest
obfuskacja?

mov eax, 5

Zadanie move / Co to jest obfuskacja?

mov eax, 2

add eax, 1

add eax, 4

sub eax, 2

Zadanie move / Co to
jest obfuskacja?

mov eax, 2 ->

lea edx, [eax+1]

xchg edx, ecx

lea eax, [ecx+1]

Zadanie move / Co to jest obfuskacja?

```
mov     eax, sel_data[edx*4]
mov     edx, off_840B140
mov     edx, [edx-200068h]
mov     [eax], edx
mov     eax, off_840B140
mov     edx, on
mov     data_p, eax
mov     eax, sel_data[edx*4]
mov     edx, stack_temp
mov     [eax], edx
mov     eax, 8804857Eh
mov     eax, eax
mov     stack_temp, eax
mov     eax, offset off_840B140
mov     edx, on
mov     data_p, eax
mov     eax, sel_data[edx*4]
mov     edx, off_840B140
mov     edx, [edx-200068h]
mov     [eax], edx
mov     eax, off_840B140
mov     edx, on
mov     data_p, eax
mov     eax, sel_data[edx*4]
mov     edx, stack_temp
mov     [eax], edx
mov     eax, 880487C4h
mov     branch_temp, eax
mov     eax, on
mov     eax, sel_target[eax*4]
mov     edx, branch_temp
mov     [eax], edx
mov     ecx, on
mov     data_p, offset jmp_r0
mov     eax, sel_data[ecx*4]
mov     edx, R0
mov     [eax], edx
mov     edx, R1
mov     [eax+4], edx
mov     edx, R2
mov     [eax+8], edx
mov     edx, R3
mov     [eax+0Ch], edx
```

5. Zliczanie wykonanych instrukcji

- Działa w przypadku gdy jest sprawdzane hasło znak po znaku
- Qira demo



Pytania

