SAPIENZA
UNIVERSITÀ DI ROMA

Anti-Disassembler techniques

Emilio Coppa

CyberChallenge.IT 2018

March 2, 2018

Anti-Disassembler techniques

Anti-Disassembler technique: issues with linear disassembly

```
E8 06 00 00 00                call    near ptr loc_4011CA+1
68 65 6C 6C 6F                push    6F6C6C5h
                  loc_4011CA:
00 58 C3                      add     [eax-3Dh], bl
```

Something is strange. What?

Anti-Disassembler technique: issues with linear disassembly

```
E8 06 00 00 00                call    near ptr loc_4011CA+1
68 65 6C 6C 6F                push    6F6C6C5h
                loc_4011CA:
00 58 C3                      add     [eax-3Dh], bl
```

Something is strange. What?

```
E8 06 00 00 00                call    near ptr loc_4011CB
68 65 6C 6C 6F 00             db      'hello', 0
                loc_4011CB:
58                            pop eax
C3                            retn
```

Different ways of disassembling depending on the execution flow.

Anti-Disassembler technique: (un)conditional jmp

```
            jz      short near ptr loc_4011C4+1
            jnz     short near ptr loc_4011C4+1
loc_4011C4:
            call near ptr 90D0D521h
```

Something is strange. What?

Anti-Disassembler technique: (un)conditional jmp

```
            jz      short near ptr loc_4011C4+1
            jnz     short near ptr loc_4011C4+1
loc_4011C4:
            call near ptr 90D0D521h
```

Something is strange. What?

```
            jz      short near ptr loc_4011C5
            jnz     short near ptr loc_4011C5
            db      0E8h
loc_4011C5:
            pop     eax
            retn
```

Alternative: jz to loc_4011C4, jnz to loc_4011C5

Anti-Disassembler technique: (un)conditional jmp (2)

```
            xor     eax, eax
            jz      short near ptr loc_4011C4+1
loc_4011C4:
            jmp     near ptr 94A8D521h
```

Something is strange. What?

Anti-Disassembler technique: (un)conditional jmp (2)

```
            xor     eax, eax
            jz      short near ptr loc_4011C4+1
loc_4011C4:
            jmp     near ptr 94A8D521h
```

Something is strange. What?

```
            xor     eax, eax
            jz      short near ptr loc_4011C5
            db      0E9h
loc_4011C5:
            pop     eax
            retn
```

Anti-Disassembler technique: impossible disassembly

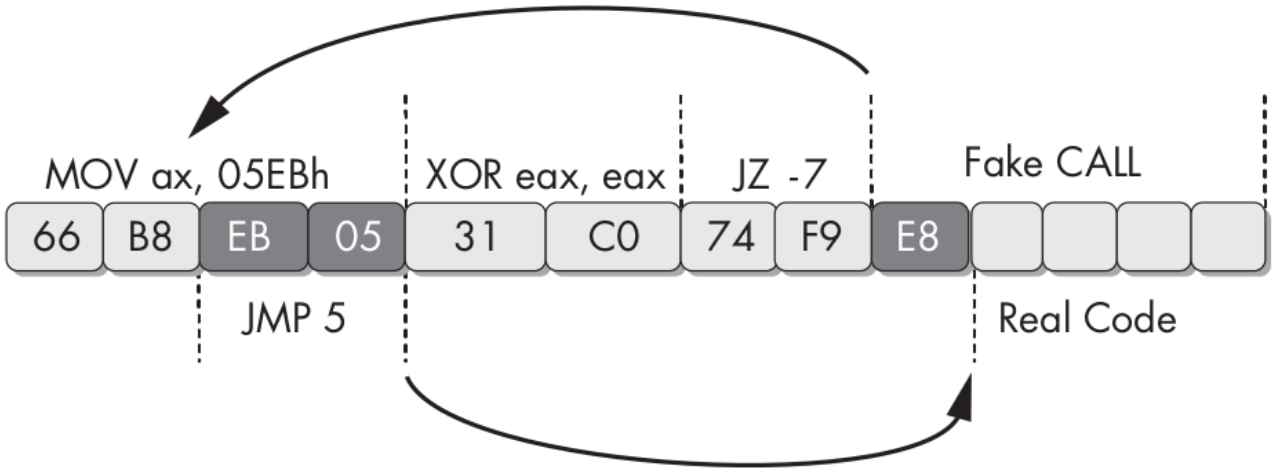
```
loc_4011C0:
            mov     ax, 5EBh
            xor     eax, eax
            jz      short near ptr loc_4011C0+2
loc_4011C8:
            call    near ptr 98A8D525h
```

Something is strange. What?

Anti-Disassembler technique: impossible disassembly

```
loc_4011C0:
            mov     ax, 5EBh
            xor     eax, eax
            jz      short near ptr loc_4011C0+2
loc_4011C8:
            call    near ptr 98A8D525h
```

Something is strange. What?



Anti-Disassembler technique: impossible disassembly (2)

```
loc_4011C0:
            mov     ax, 5EBh
            xor     eax, eax
            jz      short near ptr loc_4011C0+2
loc_4011C8:
            call    near ptr 98A8D525h
```

Compared to:

```
loc_4011C0:
            db      66h
            db      0B8h
            jmp     5               ; =>loc_4011C8+1
            xor     eax, eax
            jz      short near ptr loc_4011C0+2
loc_4011C8:
            db      0E8h
            pop     eax
            retn
```

Anti-Disassembler technique: abuse of retn

```
004011C0 sub_4011C0      proc near               ; CODE XREF: _main+19p
004011C0                                         ; sub_401040+8Bp
004011C0
004011C0 var_4           = byte ptr -4
004011C0
004011C0                 call    $+5
004011C5                 add     [esp+4+var_4], 5
004011C9                 retn
004011C9 sub_4011C0      endp ; sp-analysis failed
004011C9
```

Something is strange. What?

Anti-Disassembler technique: abuse of retn

```
004011C0 sub_4011C0      proc near               ; CODE XREF: _main+19p
004011C0                                         ; sub_401040+8Bp
004011C0
004011C0 var_4           = byte ptr -4
004011C0
004011C0                 call    $+5
004011C5                 add     [esp+4+var_4], 5
004011C9                 retn
004011C9 sub_4011C0      endp ; sp-analysis failed
004011C9
```

Something is strange. What?

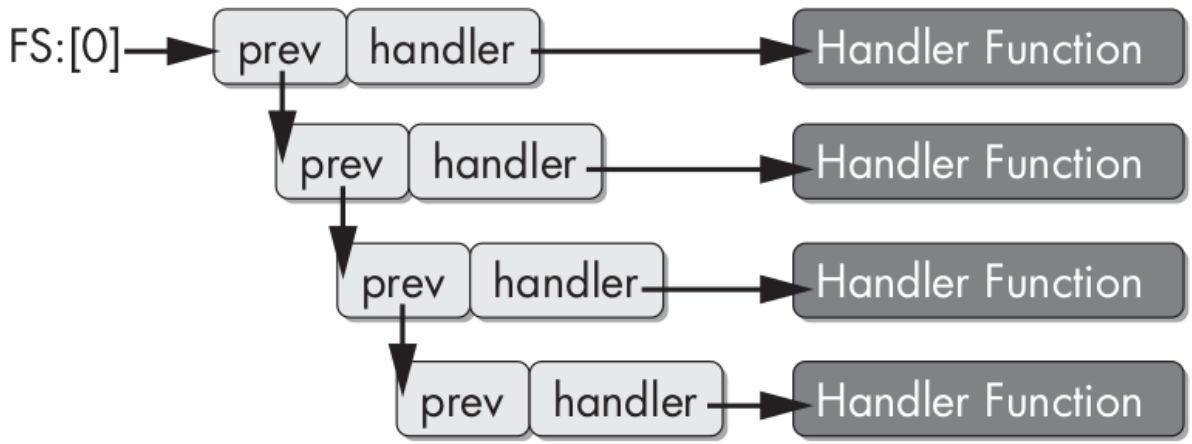
This is a jmp to the code following retn.

Anti-Disassembler technique: exception handler

In Windows, a Structural Exception Handler (SEH):

```
struct _EXCEPTION_REGISTRATION {
   DWORD prev;
   DWORD handler;
};
```

This struct is allocated on the stack. There is a chain of handlers:



FS[0] is referenced by the Thread Information Block (TIB)

Anti-Disassembler technique: exception handler (5)

Abuse of SEH to fool disassembler:

```
00401050                ❷mov     eax, (offset loc_40106B+1)
00401055                  add     eax, 14h
00401058                  push    eax
00401059                  push    large dword ptr fs:0 ; dwMilliseconds
00401060                  mov     large fs:0, esp
00401067                  xor     ecx, ecx
00401069                ❸div     ecx
0040106B
0040106B loc_40106B:                                ; DATA XREF: sub_401050
0040106B                  call    near ptr Sleep
00401070                  retn
00401070 sub_401050       endp ; sp-analysis failed
00401070
00401070 ; ---------------------------------------------------------------------
00401071                  align 10h
00401080                ❶dd 824648Bh, 0A164h, 8B0000h, 0A364008Bh, 0
00401094                  dd 6808C483h
00401098                  dd offset aMysteryCode  ; "Mystery Code"
0040109C                  dd 2DE8h, 4C48300h, 3 dup(0CCCCCCCCh)
```

Anti-Disassembler technique: thwarting stack-frame analysis

```
00401543     sub_401543      proc near           ; CODE XREF: sub_4012D0+3Cp
00401543                                         ; sub_401328+9Bp
00401543
00401543     arg_F4          = dword ptr  0F8h
00401543     arg_F8          = dword ptr  0FCh
00401543
00401543 000                 sub     esp, 8
00401546 008                 sub     esp, 4
00401549 00C                 cmp     esp, 1000h
0040154F 00C                 jl      short loc_401556
00401551 00C                 add     esp, 4
00401554 008                 jmp     short loc_40155C
00401556     ; ---------------------------------------------------------------
00401556
00401556     loc_401556:                         ; CODE XREF: sub_401543+Cj
00401556 00C                 add     esp, 104h
0040155C
0040155C     loc_40155C:                         ; CODE XREF: sub_401543+11j
0040155C -F8❶                mov     [esp-0F8h+arg_F8], 1E61h
00401564 -F8                 lea     eax, [esp-0F8h+arg_F8]
00401568 -F8                 mov     [esp-0F8h+arg_F4], eax
0040156B -F8                 mov     edx, [esp-0F8h+arg_F4]
0040156E -F8                 mov     eax, [esp-0F8h+arg_F8]
00401572 -F8                 inc     eax
00401573 -F8                 mov     [edx], eax
00401575 -F8                 mov     eax, [esp-0F8h+arg_F4]
00401578 -F8                 mov     eax, [eax]
0040157A -F8                 add     esp, 8
0040157D -100                retn
```