Defeating Anti-Reversing Tricks

AFK.conf

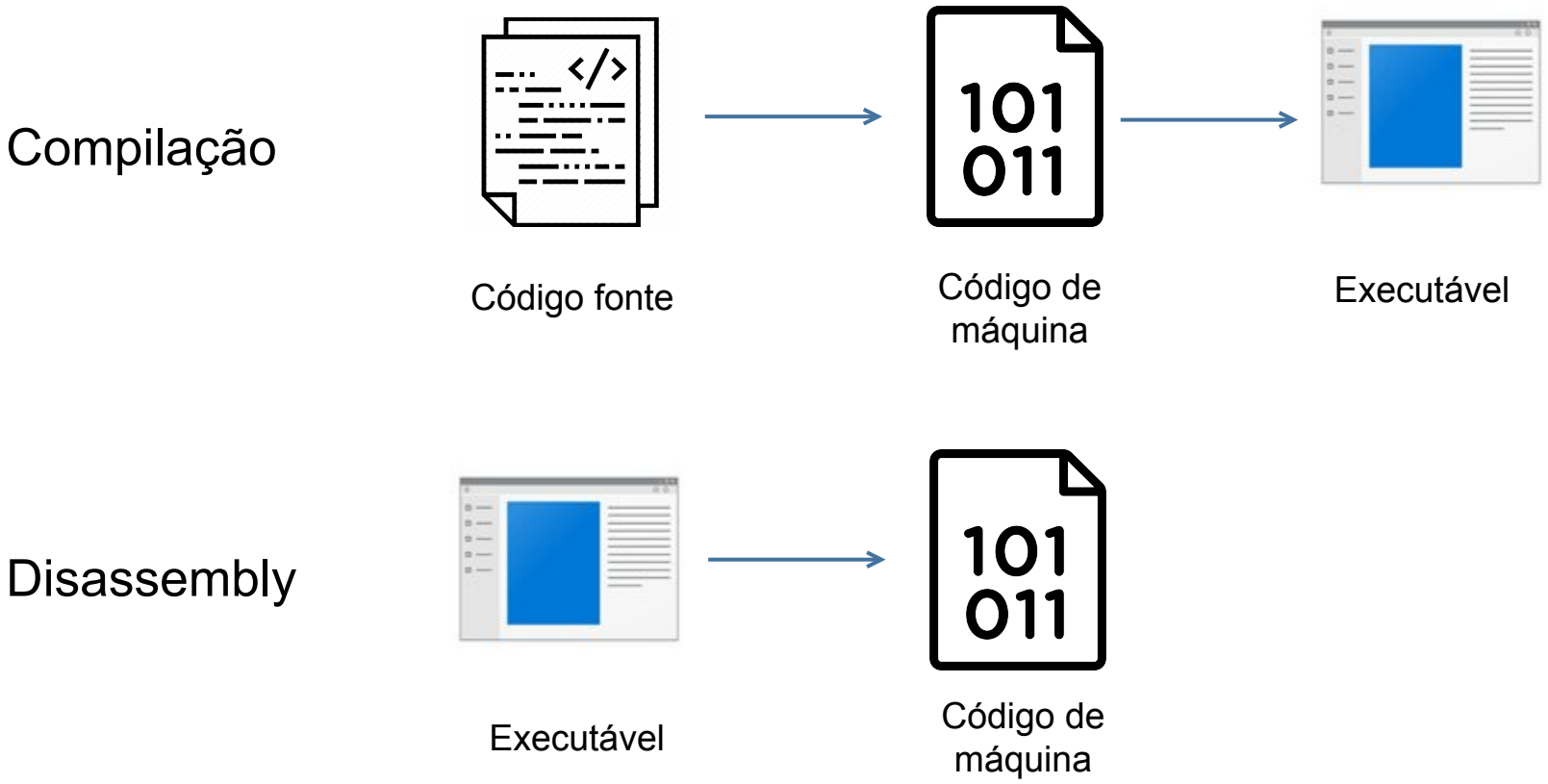
Euler Neto

Agenda

- Introdução
- Ofuscação
- Debugging
- Anti-Debugging / Anti-VM
- Exemplo prático

Introdução

- Engenharia Reversa



Introdução

- Assembly

Registradores

EAX (Acumulador)	EBX (Base)
ECX (Counter)	EDX (Data)
ESI (Source Index)	EDI (Destination Index)
ESP (Stack Pointer)	EBP (Base Pointer)

Operações

MOV (Copiar valores)	ADD / INC (Aritimética)
XOR (Bit-a-bit)	CMP (Comparação)

Saltos

JMP (Incondicional)	JE (Equals)
JG / JGE (Greater / G. Or Equals)	JL / JLE (Less / L. Or Equals)

Introdução

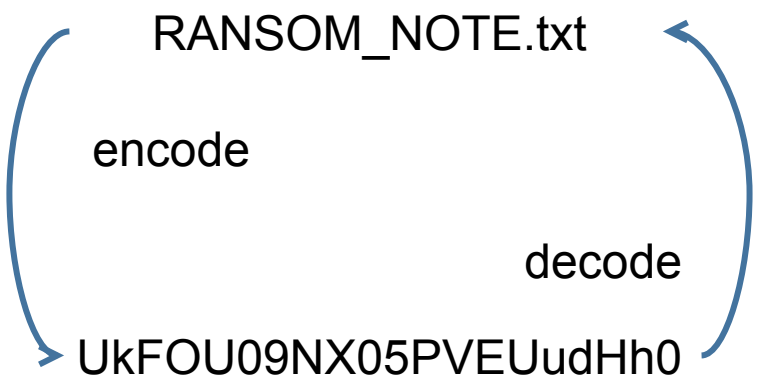
- Assembly

```c
int soma_ate_dez() {
    int a = 6;
    int b = 2;
    int c = a + b;
    if (c > 10){
        c = 0;
    }
    return c;
}
```

```asm
soma_ate_dez():
        push    rbp
        mov     rbp, rsp
        mov     DWORD PTR [rbp-8], 6
        mov     DWORD PTR [rbp-12], 2
        mov     edx, DWORD PTR [rbp-8]
        mov     eax, DWORD PTR [rbp-12]
        add     eax, edx
        mov     DWORD PTR [rbp-4], eax
        cmp     DWORD PTR [rbp-4], 10
        jle     .L2
        mov     DWORD PTR [rbp-4], 0
.L2:
        mov     eax, DWORD PTR [rbp-4]
        pop     rbp
        ret
```

Ofuscação

- Base64



- XOR

Input

```
This program cannot
```

Output

```
Key = 01: Uihr!qsnfs`l!b`oonu
Key = 02: Vjkq"rpmepco"acllmv
Key = 03: Wkjp#sqldqbn#`bmmlw
Key = 04: Plmw$tvkcvei$gejjkp
Key = 05: Qmlv%uwjbwdh%fdkkjq
Key = 06: Rnou&vtiatgk&eghhir
Key = 07: Sont'wuh`ufj'dfiihs
Key = 08: \`a{(xzgozie(kiffg|
Key = 09: ]a`z)y{fn{hd)jhggf}
```

Ofuscação

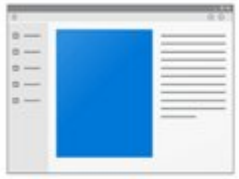
- Avoiding static strings

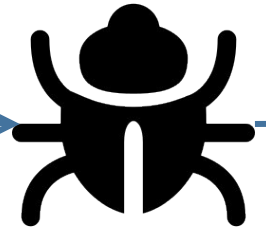
```
char s01='C';
char s02='R';
char s03='Y';
char s04='P';
char s05='T';
char s06='3';
char s07='2';
char s08='.';
char s90='D';
char s10='L';
char s11='L';

char* s12 = s01 + s02 + s03 + s04 + s05 + s06 +
            s07 + s08 + s09 + s10 + s11;

HINSTANCE hGetProcIDDLL = LoadLibrary(s12);
```

Debugging

- Processo de Debugging

```
CE523DA    CC                int3
CE523DB    CC                int3
CE523DC    48:83EC 38        sub rsp,38
CE523E0    48:8D0D 51060000  lea rcx,qword ptr ds:[<_RTC_Terminate>]
CE523E7    E8 88FEFFFF       call <x64dbg.atexit>
CE523EC    8B05 2A330000     mov eax,dword ptr ds:[<_newmode>]
CE523F2    44:8B0D 1F330000  mov r9d,dword ptr ds:[<_dowildcard>]
CE523F9    8905 952D0000     mov dword ptr ds:[<startinfo>],eax
CE523FF    48:8D05 8E2D0000  lea rax,qword ptr ds:[<startinfo>]
CE52406    4C:8D05 7B2D0000  lea r8,qword ptr ds:[<envp>]
CE5240D    48:8D15 6C2D0000  lea rdx,qword ptr ds:[<argv>]
CE52414    48:8D0D 612D0000  lea rcx,qword ptr ds:[<argc>]
CE5241B    48:894424 20      mov qword ptr ss:[rsp+20],rax
CE52420    FF15 7A0D0000     call qword ptr ds:[<&__getmainargs>]
CE52426    8905 642D0000     mov dword ptr ds:[<argret>],eax
CE5242C    85C0              test eax,eax
CE5242E    79 0A             jns x64dbg.7FF7CCE5243A
```

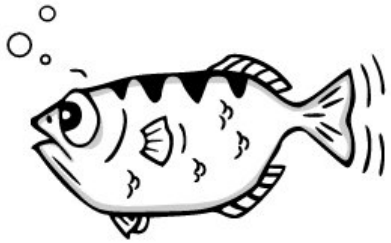




WINDOWS
x64dbg

Debugging

- Processo de Debugging



```
EAX: 0xbffff7f4 --> 0xbffff916 ("/root/a.out")
EBX: 0xb7fcbff4 --> 0x155d7c
ECX: 0xd5eeaa03
EDX: 0x1
ESI: 0x0
EDI: 0x0
EBP: 0xbffff748 --> 0xbffff7c8 --> 0x0
ESP: 0xbffff748 --> 0xbffff7c8 --> 0x0
EIP: 0x80483e7 (<main+3>:       and    esp,0xfffffff0)
EFLAGS: 0x200246 (carry PARITY adjust ZERO sign trap INT
[-----------------------------------code-----------
   0x80483e3 <frame_dummy+35>:  nop
   0x80483e4 <main>:    push   ebp
   0x80483e5 <main+1>:  mov    ebp,esp
=> 0x80483e7 <main+3>:  and    esp,0xfffffff0
   0x80483ea <main+6>:  sub    esp,0x110
   0x80483f0 <main+12>: mov    eax,DWORD PTR [ebp+0xc]
   0x80483f3 <main+15>: add    eax,0x4
   0x80483f6 <main+18>: mov    eax,DWORD PTR [eax]
```

LINUX
gdb (PEDA
plugin)

Debugging

- Processo de Debugging
  - Single-Stepping

```
mov     edi, DWORD_00406904
mov     ecx, 0x0d
LOC_040106B2
xor     [edi], 0x9C
inc     edi
loopw   LOC_040106B2
...
DWORD:00406904:   F8FDF3D0❶
```

Debugging

- Processo de Debugging
  - Stepping Over

```
mov     edi, DWORD_00406904
mov     ecx, 0x0d
LOC_040106B2
xor     [edi], 0x9C
inc     edi
loopw   LOC_040106B2
...
DWORD:00406904:   F8FDF3D0❶
```

  - Stepping Into

```
call    GetSystemDefaultLCID
mov     [ebp+var_4], eax
cmp     [ebp+var_4], 409h
jnz     short loc_411360
call    sub_411037
cmp     [ebp+var_4], 411h
```

```
xor ecx, ecx
add ecx, eax
push eax ❷
ret
```

Debugging

- Processo de Debugging
  - Breakpoints

```
mov     edi, DWORD_00406904
mov     ecx, 0x0d
LOC_040106B2
xor     [edi], 0x9C
inc     edi
loopw   LOC_040106B2
...
DWORD:00406904:   F8FDF3D0❶
```

b

  - Modificar Execução

```
call    GetSystemDefaultLCID
mov     [ebp+var_4], eax
cmp     [ebp+var_4], 409h
jnz     short loc_411360
call    sub_411037
cmp     [ebp+var_4], 411h
jz      short loc_411372
cmp     [ebp+var_4], 421h
jnz     short loc_411377
call    sub_41100F
cmp     [ebp+var_4], 0C04h
jnz     short loc_411385
call    sub_41100A
```

```
EAX 00000000
ECX 0007FFB0
EDX 7C90EB94
EBX 7FFDD000
ESP 0007FFC4
EBP 0007FFF0
ESI FFFFFFFF
EDI 7C910738
```

Anti-Debugging / Anti-VM

- IsDebuggerPresent

IsDebuggerPresent function (debugapi.h)

Article • Feedback

Determines whether the calling process is being debugged by a user-mode debugger.

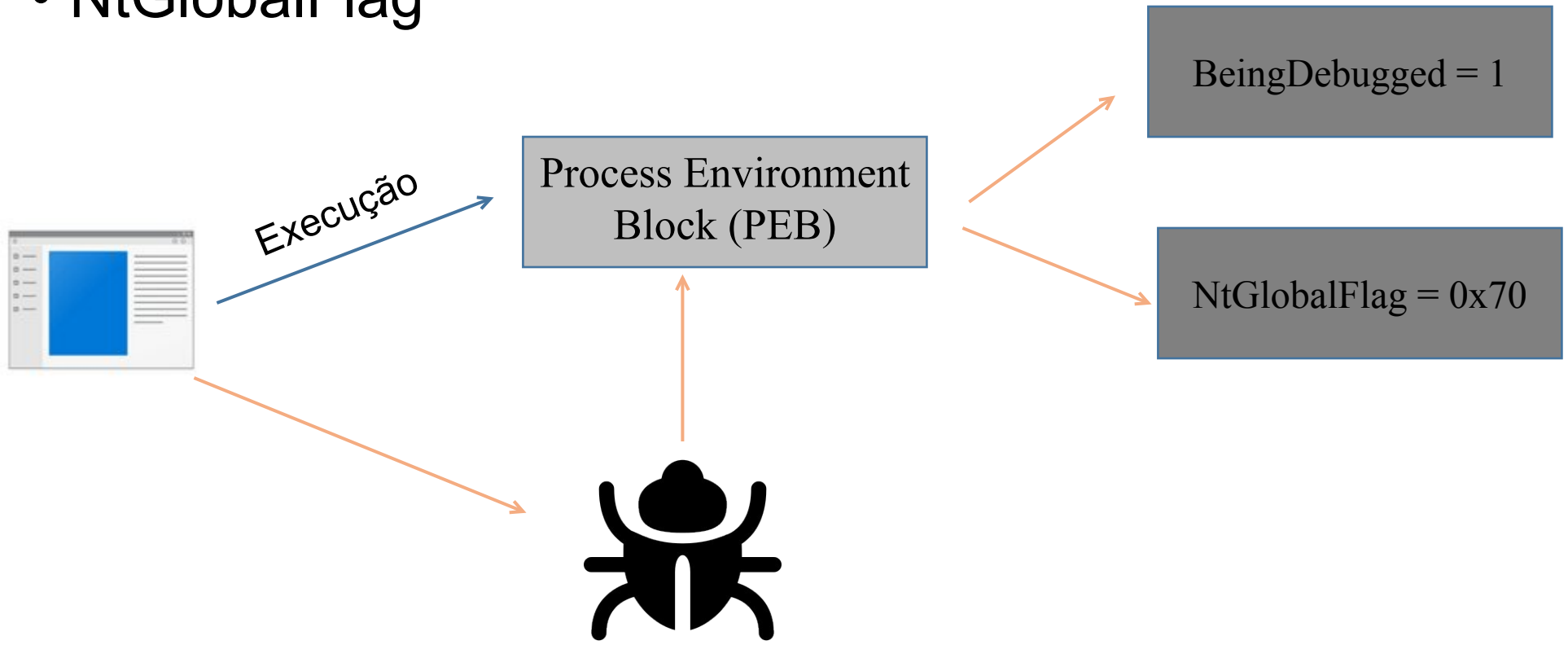
Return value

If the current process is running in the context of a debugger, the return value is nonzero.

If the current process is not running in the context of a debugger, the return value is zero.

Anti-Debugging / Anti-VM

- NtGlobalFlag



Anti-Debugging / Anti-VM

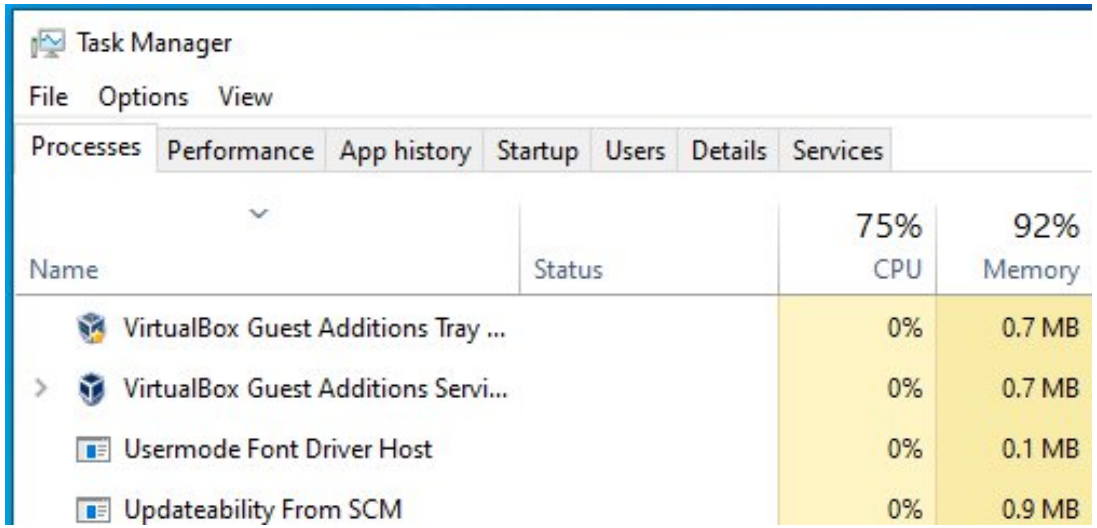
- Sleep / GetTickCount

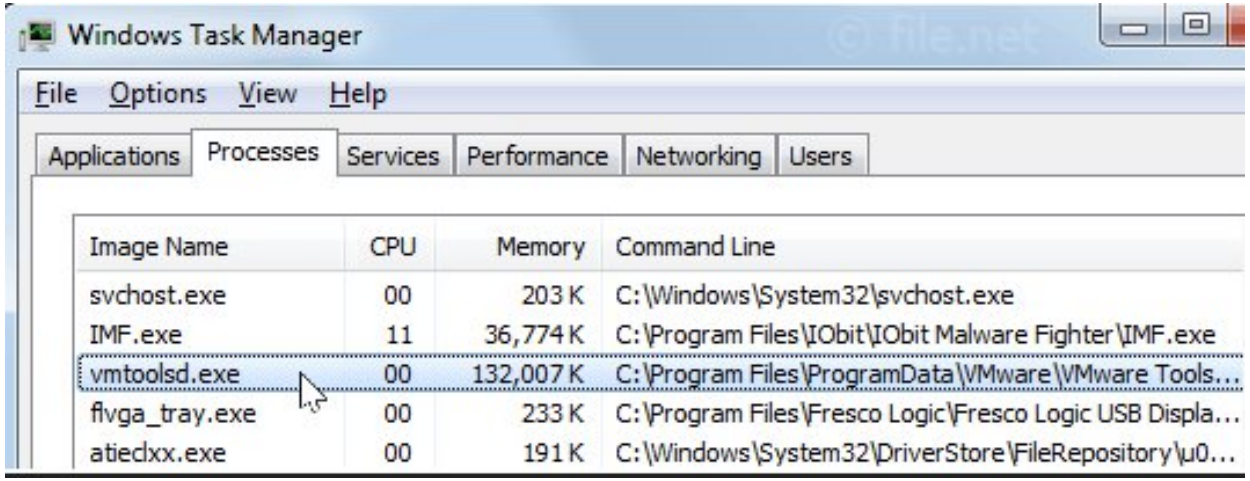
```
// Do something
Sleep(5000);
// Do something
Sleep(5000);
// Do something
Sleep(5000);
// Do something
Sleep(5000);
// Do something
Sleep(5000);
```

```
DWORD currentTime = GetTickCount();

// Do something

// Detect time difference
If ( GetTickCount() - currentTime > 1000 )
{
    printf("Debug detected!!!");
    exit(1);
}
```

Anti-Debugging / Anti-VM

- Detect VM





- Vmtoolsd.exe
- Vmwaretrat.exe
- Vmwareuser.exe
- Vmacthlp.exe

Exemplo prático

- Code

Referências

- https://x64dbg.com/
- https://sourceware.org/gdb/
- https://github.com/whichbuffer/Antidebug
- https://www.apriorit.com/dev-blog/367-anti-reverse-engineering-protection-techniques-to-use-before-releasing-software#p2
- https://www.mentebinaria.com.br/artigos/tudo/categorias-de-anti-debugging-tls-callback-r85/
- https://www.mentebinaria.com.br/artigos/engenharia-reversa/flags-de-depura%C3%A7%C3%A3o-ntglobalflag-r101/