Introduction to Windows exploitation

7 juillet 2011

Why this talk ?

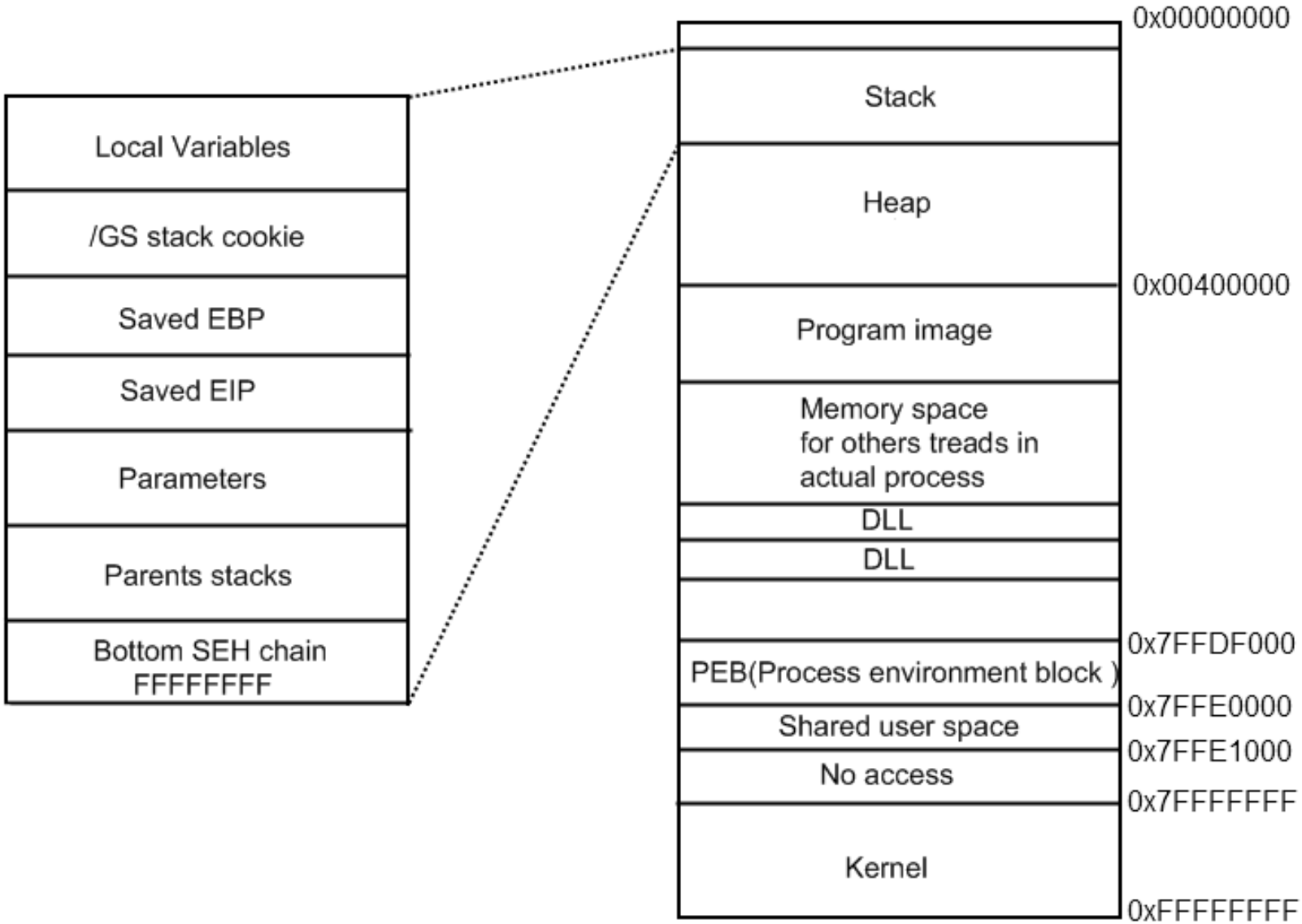
- System most used
- No courses about windows internal
- Fun

Why this talk ?

- System most used
- No courses about windows internal
- Fun

For who ?

- Curious people
- Need some base on application exploitation
- Knowledge assembly language


Local Variables
/GS stack cookie
Saved EBP
Saved EIP
Parameters
Parents stacks
Bottom SEH chain
FFFFFFFF
0x00000000
Stack
Heap
0x00400000
Program image
Memory space
for others treads in
actual process
DLL
DLL
0x7FFDF000
PEB(Process environment block)
0x7FFE0000
Shared user space
0x7FFE1000
No access
0x7FFFFFFF
Kernel
0xFFFFFFFF

Process Environment Block

- Image Base Address
- BeingDebugged (IsDebuggerPresent())
- Start Address of the heap
- Information about loaded modules

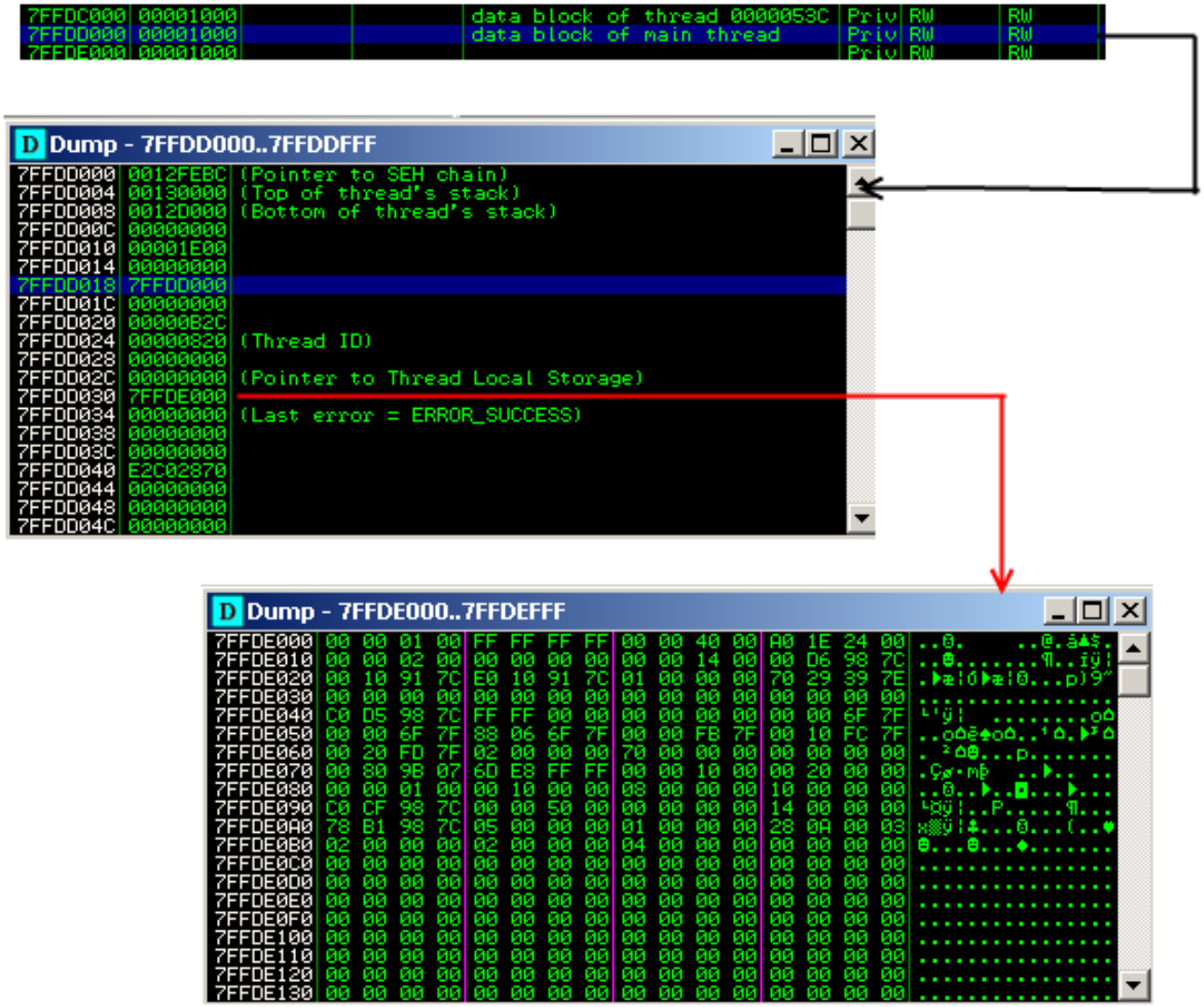
- EPROCESS
- ntdll
- NtCreateUserProcess()
- IMAGE_OPTIONAL_HEADER
- OsMajorVersion, ...

Thread Environment Block

- Location of PEB
- Location of the stack (start & end)
- First entry in the SEH chain
- NT_TIB 0x018 (0x1C)
- Last error
- Process ID, Thread ID
- Can be accessed by segment FS.

```
void *getTEB()
{
        void *teb = NULL;

        __asm__("movl %%fs:0x0, %0" : "=r" (teb) : : );
        return (teb);
}
```


7FFDC000 00001000 data block of thread 0000053C Priv RW RW
7FFDD000 00001000 data block of main thread Priv RW RW
7FFDE000 00001000 Priv RW RW
Dump - 7FFDD000..7FFDDFFF
7FFDD000 0012FEBC (Pointer to SEH chain)
7FFDD004 00130000 (Top of thread's stack)
7FFDD008 0012D000 (Bottom of thread's stack)
7FFDD00C 00000000
7FFDD010 00001E00
7FFDD014 00000000
7FFDD018 7FFDD000
7FFDD01C 00000000
7FFDD020 00000B2C
7FFDD024 00000820 (Thread ID)
7FFDD028 00000000
7FFDD02C 00000000 (Pointer to Thread Local Storage)
7FFDD030 7FFDE000
7FFDD034 00000000 (Last error = ERROR_SUCCESS)
7FFDD038 00000000
7FFDD03C 00000000
7FFDD040 E2C02870
7FFDD044 00000000
7FFDD048 00000000
7FFDD04C 00000000
Dump - 7FFDE000..7FFDEFFF
7FFDE000 00 00 01 00 FF FF FF FF 00 00 40 00 A0 1E 24 00
7FFDE010 00 00 02 00 00 00 00 00 00 00 14 00 00 D6 98 7C
7FFDE020 00 10 91 7C E0 10 91 7C 01 00 00 00 70 29 39 7E
7FFDE030 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
7FFDE040 C0 D5 98 7C FF FF 00 00 00 00 00 00 00 00 6F 7F
7FFDE050 00 00 6F 7F 88 06 6F 7F 00 00 FB 7F 00 10 FC 7F
7FFDE060 00 20 FD 7F 02 00 00 00 70 00 00 00 00 00 00 00
7FFDE070 00 80 9B 07 6D E8 FF FF 00 00 10 00 00 20 00 00
7FFDE080 00 00 01 00 00 10 00 00 08 00 00 00 10 00 00 00
7FFDE090 C0 CF 98 7C 00 00 50 00 00 00 00 00 14 00 00 00
7FFDE0A0 78 B1 98 7C 05 00 00 00 01 00 00 00 28 0A 00 03
7FFDE0B0 02 00 00 00 02 00 00 00 04 00 00 00 00 00 00 00
7FFDE0C0 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
7FFDE0D0 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
7FFDE0E0 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
7FFDE0F0 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
7FFDE100 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
7FFDE110 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
7FFDE120 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
7FFDE130 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00


0x00000000
...
AAAA
AAAA
AAAA
~~Save EBP~~ AAAA
~~Save EIP~~ AAAA
~~ptr argv[1]~~ AAAA
...
0xFFFFFFFF

jump / call reg

- Register that points to shellcode
- Find opcode into loaded dll

jump / call reg

- Register that points to shellcode
- Find opcode into loaded dll

pop return

- No register point to shellcode
- Magic stack (first, second, . . . address of the stack)

push return

- No opcode jmp / call
- Find Push reg ; ret

push return

- No opcode jmp / call
- Find Push reg ; ret

Safedisc

- Macrovision
- SimCity 3000, Need for speed 2 ...
- File .exe & .icd
- Buffer overflow
- Push XXX ret
- Obfuscation ?
- bp WriteProcessMemory
- jmp 0x0

jmp [reg + offset]

- Register doesn't point at the begining of shellcode

jmp [reg + offset]

- Register doesn't point at the begining of shellcode

blind return

- Overwrite EIP with a ret instruction
- Hardcode Adress of the shellcode into esp

Software Exception Handler

- Gestionnaire d'exceptions
- __try / __except / __finally
- ExitProcess()

Software Exception Handler

- Gestionnaire d'exceptions
- __try / __except / __finally
- ExitProcess()

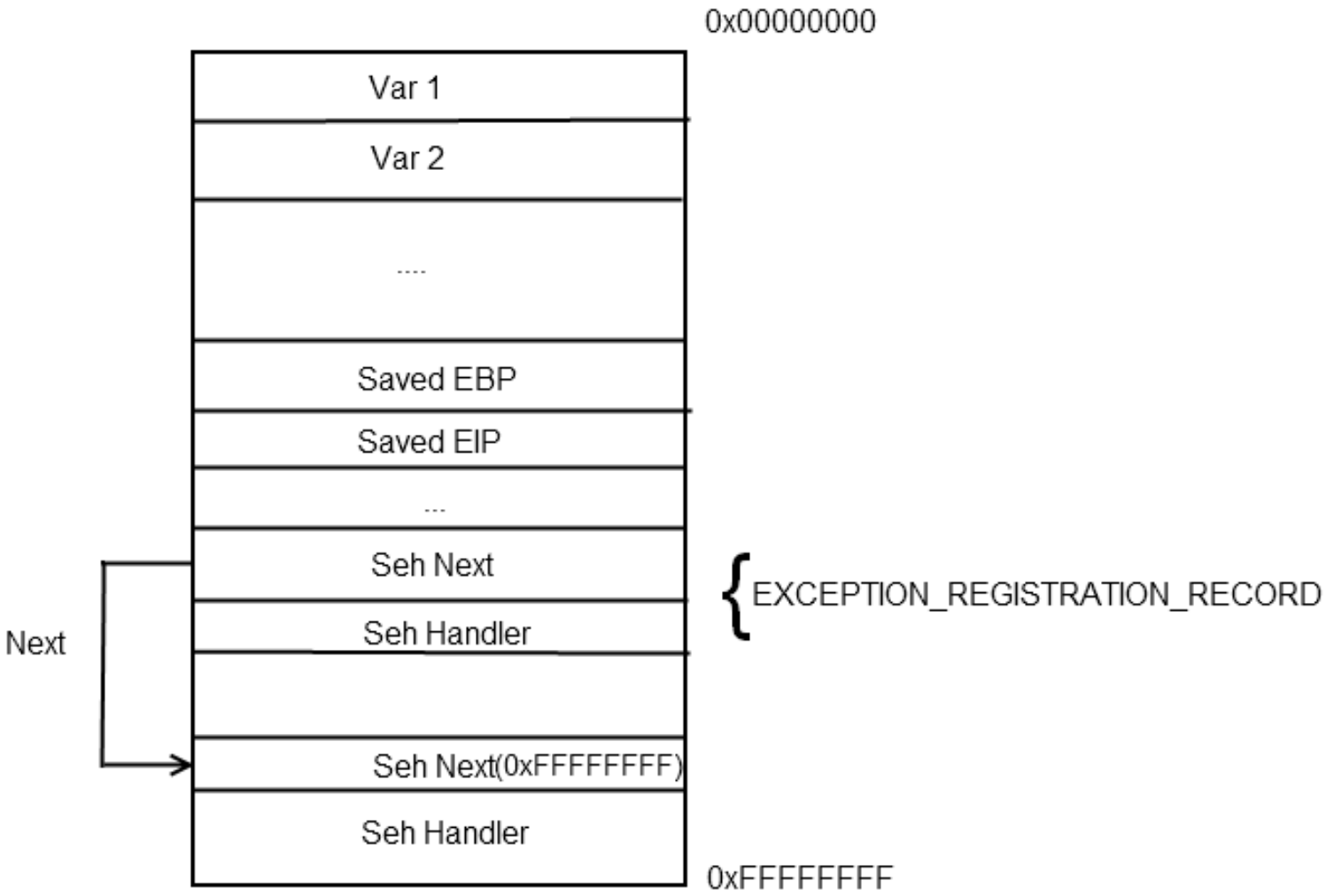
Exception's type

Hardware Exception :

- ACCES_VIOLATION
- DIVISION_BY_ZERO

Software Exception :

- RaiseException()
- NOT_ENOUGH_MEMORY
- BAD_FILE_FORMAT
- ...


0x00000000
Var 1
Var 2
....
Saved EBP
Saved EIP
...
Seh Next
Seh Handler
Next
EXCEPTION_REGISTRATION_RECORD
Seh Next(0xFFFFFFFF)
Seh Handler
0xFFFFFFFF

Défaut

Default seh exception (UnhandledExceptionFilter()).
BaseProcessStart.
WinMain().
JustInTimeDebugging.

SEH Struct

```
typedef void* PVOID;

typedef struct _EXCEPTION_REGISTRATION_RECORD
{
  struct _EXCEPTION_REGISTRATION_RECORD *next;
  struct PVOID Handler;
}EXCEPTION_REGISTRATION_RECORD;
```

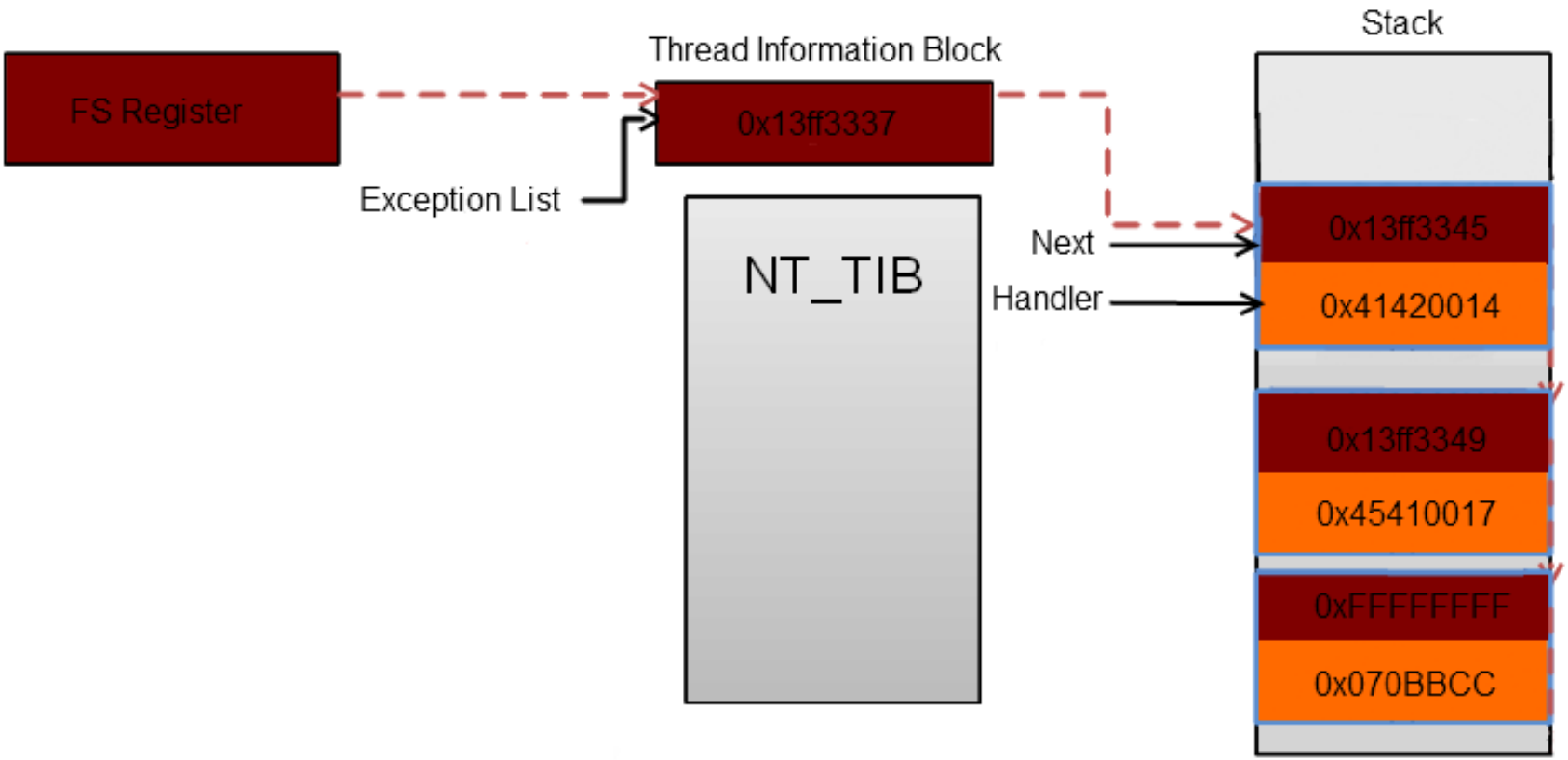
SEH Struct

```
typedef void* PVOID;

typedef struct _EXCEPTION_REGISTRATION_RECORD
{
  struct _EXCEPTION_REGISTRATION_RECORD *next;
  struct PVOID Handler;
}EXCEPTION_REGISTRATION_RECORD;
```

How it works ?

- ntdll !KiUserDispatchException
- Top of linked list store in fs :[0] (TEB)
- Update of this top address after each call
- Bottom linked chain is FFFFFFFF. (Os take the hand)


FS Register
Thread Information Block
0x13ff3337
Exception List
NT_TIB
Stack
0x13ff3345
Next
Handler
0x41420014
0x13ff3349
0x45410017
0xFFFFFFFF
0x070BBCC

Demo Time

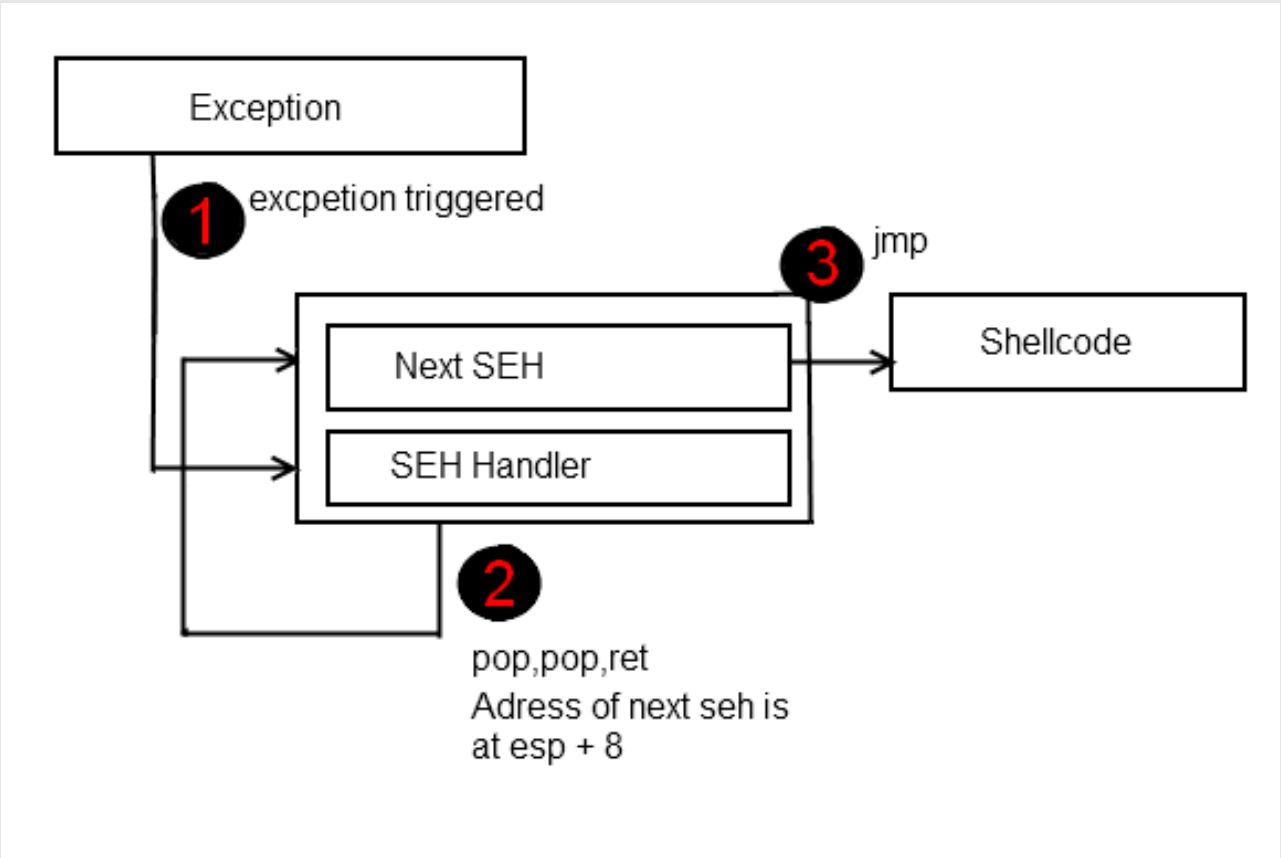
A simply example in ASM

```
push handler
push fs:[0]
mov fs:[0], esp
; ...
; Code protected by SEH ?
; ...
pop fs:[0]
add esp, 4
ret
```

Stack when exception handler called

```
EXCEPTION_DISPOSITION __cdecl _except_handler(
struct _EXCEPTION_RECORD *ExceptionRecord,
void *EstablishFrame,
struct CONTEXT *ContextThread,
void *DispatcherContext
);
```

HOW TO


Exception
1
excpetion triggered
3
jmp
Next SEH
SEH Handler
Shellcode
2
pop,pop,ret
Adress of next seh is
at esp + 8

Demo Time

tElock

- Old Packer but fun
- Lots of division by zero / int3
- IAT Redirection
- Get Context thread
- Clear DebugRegister

Presentation

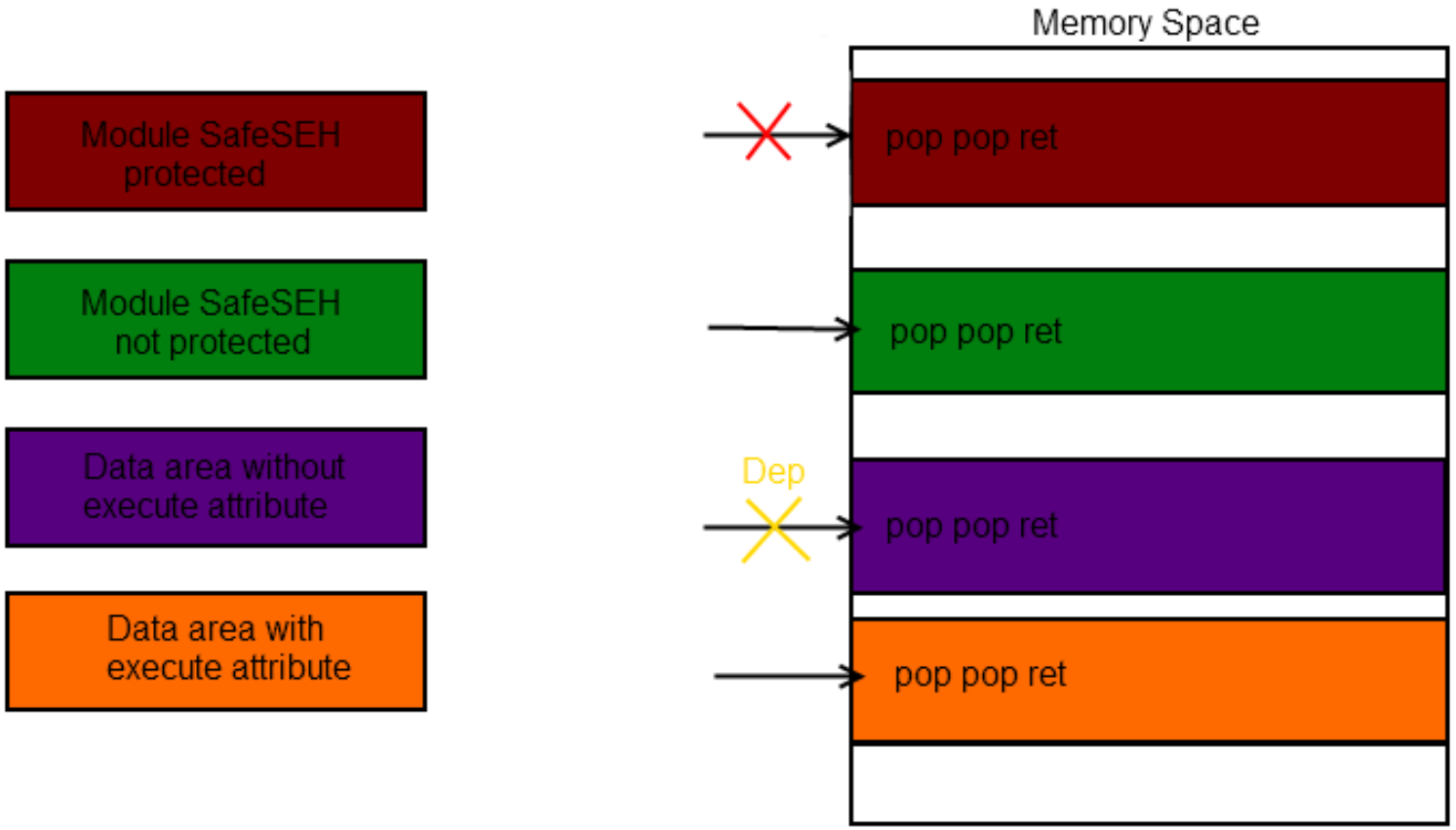
- Sup XP SP2
- /SAFESEH
- IMAGE_LOAD_CONFIG_DIRECTORY32
- SEHandlerTable
- SEHandlerCount
- ntdll !RtlIsValidHandler

```
BOOL RtlIsValidHandler(Handler)
   if (handler is in image)
        if (image has no IMAGE_DLLCHARACTERISTICS_NO_SEH flag)
                return FALSE;
        if (image has a SEH Table)
                if (handler found in seh table)
                        return TRUE;
                else
                        return FALSE;
   if (handler is on a non-exec page)
        if (ExecuteDispatchEnable bit set in KPROCESS)
           return TRUE;
        else
           raise ACCESS_VIOLATION;
   if (handler is not in an image)
        if (ImageDispatcherEnable bit set in KPROCESS)
           return TRUE;
        else
           return FALSE;
```

Bypass it

- Adress From a module without /SafeSEH, IMAGE_DLLCHARACTERISTICS_NO_SEH
- Instruction from a predictable spot in memory
- Adress from the Heap

Demo Time


Module SafeSEH protected
Module SafeSEH not protected
Data area without execute attribute
Data area with execute attribute
Memory Space
pop pop ret
pop pop ret
Dep
pop pop ret
pop pop ret

SEHOP

- Windows Server 2008
- HKLM\SYSTEM\CurrentControlSet\Control\SessionManager
- \kernel\DisableExceptionChainValidation
- RtlDispatchException()
- 0xFFFFFFFF

Presentation

- Since XP SP2
- STATUS_ACCESS_VIOLATION
- Page Table Entry
- /NXCOMPACT
- SetProcessDEPPolicy

DEP options

- OptIn : System process, choosen process by user
- OptOut : All system process, user can discard some process
- AlwaysOn : All process
- AlwaysOff : Nothing

Bypass it

- WinExec (Ret-into-libc)
- Mark page as executable
- Alloc, copy, jump
- Change DEP Settings

Function

- VirtualAlloc()
- HeapCreate()
- SetProcessDEPPolicy()
- NtSetInformationProcess()
- VirtualProtect()
- WriteProcessMemory()

Function/OS	XP SP2	XP SP3	Vista SP0	Vista SP1	Win 7	2008
VirtualAlloc()	yes	yes	yes	yes	yes	yes
HeapCreate()	yes	yes	yes	yes	yes	yes
SetProcessDEPPolicy()	no	yes	no	yes	no	yes
NTsetInformationProcess()	yes	yes	yes	no	no	no
VirtualProtect()	yes	yes	yes	yes	yes	yes
WriteProcessMemory()	yes	yes	yes	yes	yes	yes

VirtualAlloc()

```
LPVOID WINAPI VirtualAlloc(
__in_opt        LPVOID lpAddress,
__in            SIZE_T dwSize,
__in            DWORD flAllocationType,
__in            DWORD flProtect
);
```

Need

- Return Address
- Setup correctly stack
- Copy
- Jump

HeapCreate()

```
HANDLE WINAPI HeapCreate(
__in        DWORD flOptions,
__in        SIZE_T dwInitialSize,
__in        SIZE_T dwMaximumSize,
);
```

Need

- HeapAlloc()
- Return Address
- Setup correctly stack
- Copy
- Jump

SetProcessDEPPolicy()

```
BOOL WINAPI SetProcessDEPPolicy(
__in    DWORD dwFlags
);
```

Need

- Work only on XP SP3, Vista SP1, 2008
- Return Adress
- Stack (only 0)

NtSetInformationProcess()

```
NtSetInformationProcess(
__in HANDLE     ProcessHandle,
__in PROCESS_INFORMATION_CLASS ProcessInformationClass,
__in PVOID      ProcessInfo,
__in ULONG      ProcessInformationLength
);
```

Need

- Windows XP, Vista SP0, 2003
- Proprer stack
- Return adress

VirtualProtect()

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

Need

- Work Everywhere
- Proper Stack
- Return Address

Return Oriented Programming

- Use code from the stack won't work
- Gadget
- Existing Instructions
- Build a chain of instructions
- Return from one insttruction to an other one

SEH

- Pop pop ret Not Possible
- Code Execution on stack failed
- Rop Chain
- Bypass Execution Prevention
- Way to return to our payload

First Step : Stack pivot

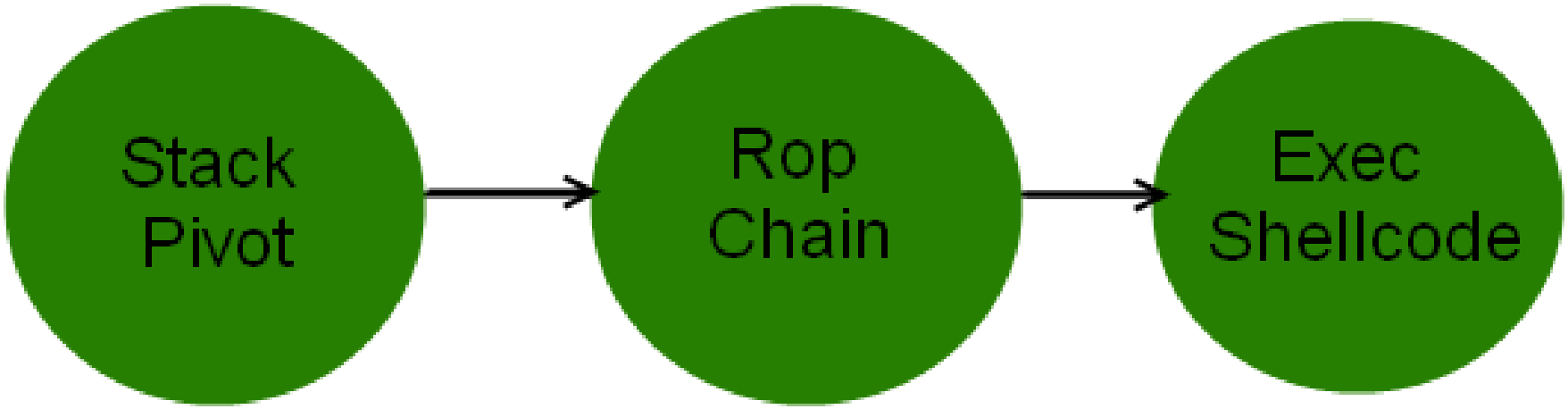
- add esp, XXX ; ret
- mov esp, [reg] ; ret
- xchg [reg], esp ; ret
- call [reg]
- push [reg] + pop esp ; ret

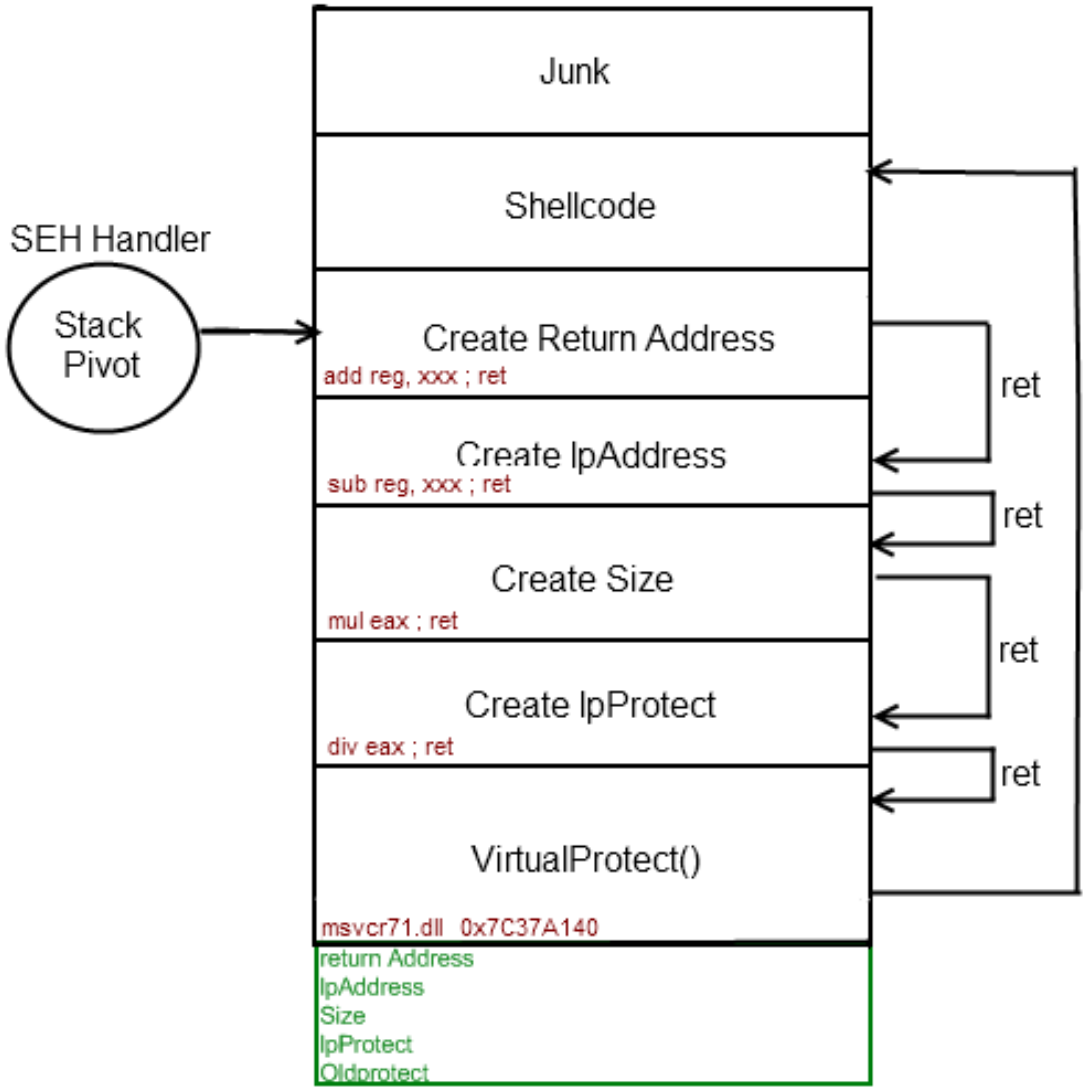
Second Step : Setup stack / registers

- Instruction ; ret = Rop gadget
- VirtualProtect
- 5 parameters

Final Stage

Execute Shellcode


Stack
Pivot
Rop
Chain
Exec
Shellcode


Junk
Shellcode
SEH Handler
Stack Pivot
Create Return Address
add reg, xxx ; ret
Create lpAddress
sub reg, xxx ; ret
Create Size
mul eax ; ret
Create lpProtect
div eax ; ret
VirtualProtect()
msvcr71.dll 0x7C37A140
return Address
lpAddress
Size
lpProtect
Oldprotect
ret
ret
ret
ret

Demo Time

Presentation

- > Vista
- Image, stack, heap, TEB, PEB
- Seven, each execution

System parameter

HKLM\SYSTEM\CurrentControlSet\Control\SessionManager\Memory Management\MoveImages

- 0 : Disable
- 1 : All processus
- IMAGE_DLLCHARACTERISTICS_DYNAMIC_BASE (PE Header)
- /dynamicbase

Bypass it

- Partial overwrite
- Load module without aslr
- Second vulnerability (read memory)
- ntdll!_KUSER_SHARED_DATA (0x7ffe0300) -> ntdll!KiFastSystemCall

Bypass it

- Partial overwrite
- Load module without aslr
- Second vulnerability (read memory)
- ntdll!_KUSER_SHARED_DATA (0x7ffe0300) -> ntdll!KiFastSystemCall

Heap Spraying

```
var x = new Array();

for (var i = 0; i < 200; i++) {
        x[i] = nop + shellcode;
}
```

Presentation

- Visual Studio 2002
- Like SSP
- /GS
- 4 octets
- 2 * 0x00
- GetCurrentProcessId, GetTickCount, QueryPerformanceCounter

CVE-2007-0038

```
void fail(char *src, int len)
{
        struct {
                int a;
                int b;
        } buf;
        memcpy(&buf, src, len);
}
```

Bug

- No string buffer detected
- Size controlled by the user

Another Failed

```
void fail(int count, int data)
{
        int     array[10];
        int     i;

        for (i = 0; i < count; i++)
                array[i] = data;
}
```

Fix

- VS 2005
- #pragma strict_gs_check(on)

Vérification de la sécurité de la mémoire tampon **Non (/GS-)**

```
00411400  55              PUSH EBP
00411401  8BEC            MOV EBP,ESP
00411403  81EC E4000000   SUB ESP,0E4
00411409  53              PUSH EBX
0041140A  56              PUSH ESI
0041140B  57              PUSH EDI
0041140C  8DBD 1CFFFFFF   LEA EDI,DWORD PTR SS:[EBP-E4]
00411412  B9 39000000     MOV ECX,39
00411417  B8 CCCCCCCC     MOV EAX,CCCCCCCC
0041141C  F3:AB           REP STOS DWORD PTR ES:[EDI]
0041141E  8B45 08         MOV EAX,DWORD PTR SS:[EBP+8]
00411421  50              PUSH EAX
00411422  8D4D E0         LEA ECX,DWORD PTR SS:[EBP-20]
00411425  51              PUSH ECX
00411426  E8 70FCFFFF     CALL bypass_g.0041109B
0041142B  83C4 08         ADD ESP,8
0041142E  52              PUSH EDX
0041142F  8BCD            MOV ECX,EBP
00411431  50              PUSH EAX
00411432  8D15 54144100   LEA EDX,DWORD PTR DS:[411454]
00411438  E8 40FCFFFF     CALL bypass_g.0041107D
0041143D  58              POP EAX
0041143E  5A              POP EDX
0041143F  5F              POP EDI
00411440  5E              POP ESI
00411441  5B              POP EBX
00411442  81C4 E4000000   ADD ESP,0E4
00411448  3BEC            CMP EBP,ESP
0041144A  E8 E7FCFFFF     CALL bypass_g.00411136
0041144F  8BE5            MOV ESP,EBP
00411451  5D              POP EBP
00411452  C3              RETN
```

Vérification de la sécurité de la mémoire tampon **Oui (/GS)**

```
00411400  55              PUSH EBP
00411401  8BEC            MOV EBP,ESP
00411403  81EC E8000000   SUB ESP,0E8          1
00411409  53              PUSH EBX
0041140A  56              PUSH ESI
0041140B  57              PUSH EDI
0041140C  8DBD 18FFFFFF   LEA EDI,DWORD PTR SS:[EBP-E8]
00411412  B9 3A000000     MOV ECX,3A
00411417  B8 CCCCCCCC     MOV EAX,CCCCCCCC
0041141C  F3:AB           REP STOS DWORD PTR ES:[EDI]
0041141E  A1 00704100     MOV EAX,DWORD PTR DS:[417000]   2
00411423  33C5            XOR EAX,EBP
00411425  8945 FC         MOV DWORD PTR SS:[EBP-4],EAX
00411428  8B45 08         MOV EAX,DWORD PTR SS:[EBP+8]
0041142B  50              PUSH EAX
0041142C  8D4D DC         LEA ECX,DWORD PTR SS:[EBP-24]
0041142F  51              PUSH ECX
00411430  E8 66FCFFFF     CALL bypass_g.0041109B
00411435  83C4 08         ADD ESP,8
00411438  52              PUSH EDX
00411439  8BCD            MOV ECX,EBP
0041143B  50              PUSH EAX
0041143C  8D15 68144100   LEA EDX,DWORD PTR DS:[411468]
00411442  E8 36FCFFFF     CALL bypass_g.0041107D
00411447  58              POP EAX
00411448  5A              POP EDX
00411449  5F              POP EDI
0041144A  5E              POP ESI
0041144B  5B              POP EBX
0041144C  8B4D FC         MOV ECX,DWORD PTR SS:[EBP-4]    3
0041144F  33CD            XOR ECX,EBP
00411451  E8 BEFBFFFF     CALL bypass_g.00411014
00411456  81C4 E8000000   ADD ESP,0E8
0041145C  3BEC            CMP EBP,ESP
0041145E  E8 D3FCFFFF     CALL bypass_g.00411136
00411463  8BE5            MOV ESP,EBP
00411465  5D              POP EBP
00411466  C3              RETN
```

How TO

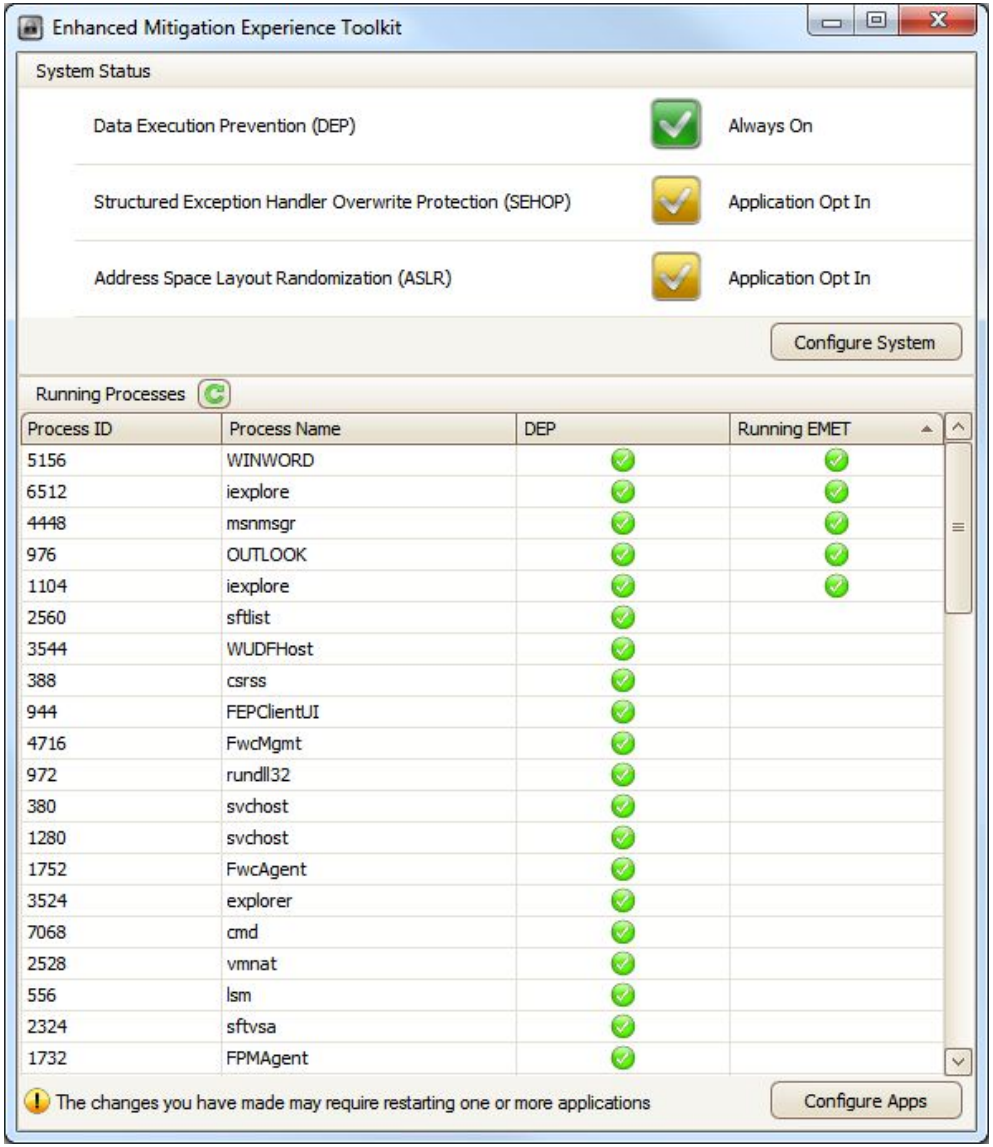
- Bypass using Exception Handling
- Bypass by replacing cookie on stack and in .data section
- Bypass because not all buffers are protected
- Bypass by overwriting stack data in functions up the stack
- Bypass because the cookie is static ?

Example

```
void foo(char *src)
{
  char buffer[100];
  try
  {
    strcpy(buffer, src);
  }
  catch (char *str)
  {
    printf("Bhuitre\n");
  }
}
int main(int argc, char **argv)
{
  foo(argv[1]);
  return (0);
}
```

Demo Time

- Enhanced Mitigation Experience Toolkit
- Control the activation (DEP, SEHOP, ASLR)
- Specific process


Enhanced Mitigation Experience Toolkit
System Status
Data Execution Prevention (DEP)
Always On
Structured Exception Handler Overwrite Protection (SEHOP)
Application Opt In
Address Space Layout Randomization (ASLR)
Application Opt In
Configure System
Running Processes
Process ID
Process Name
DEP
Running EMET
5156 WINWORD
6512 iexplore
4448 msnmsgr
976 OUTLOOK
1104 iexplore
2560 sftlist
3544 WUDFHost
388 csrss
944 FEPClientUI
4716 FwcMgmt
972 rundll32
380 svchost
1280 svchost
1752 FwcAgent
3524 explorer
7068 cmd
2528 vmnat
556 lsm
2324 sftvsa
1732 FPMAgent
The changes you have made may require restarting one or more applications
Configure Apps

6 protections

- SEHOP
- DEP (program without /nxcompat)
- Anti Heap Spraying
- Null dereferencing pointer
- ASLR (program without /DYNAMICBASE)
- Export Address Table Filtering

I

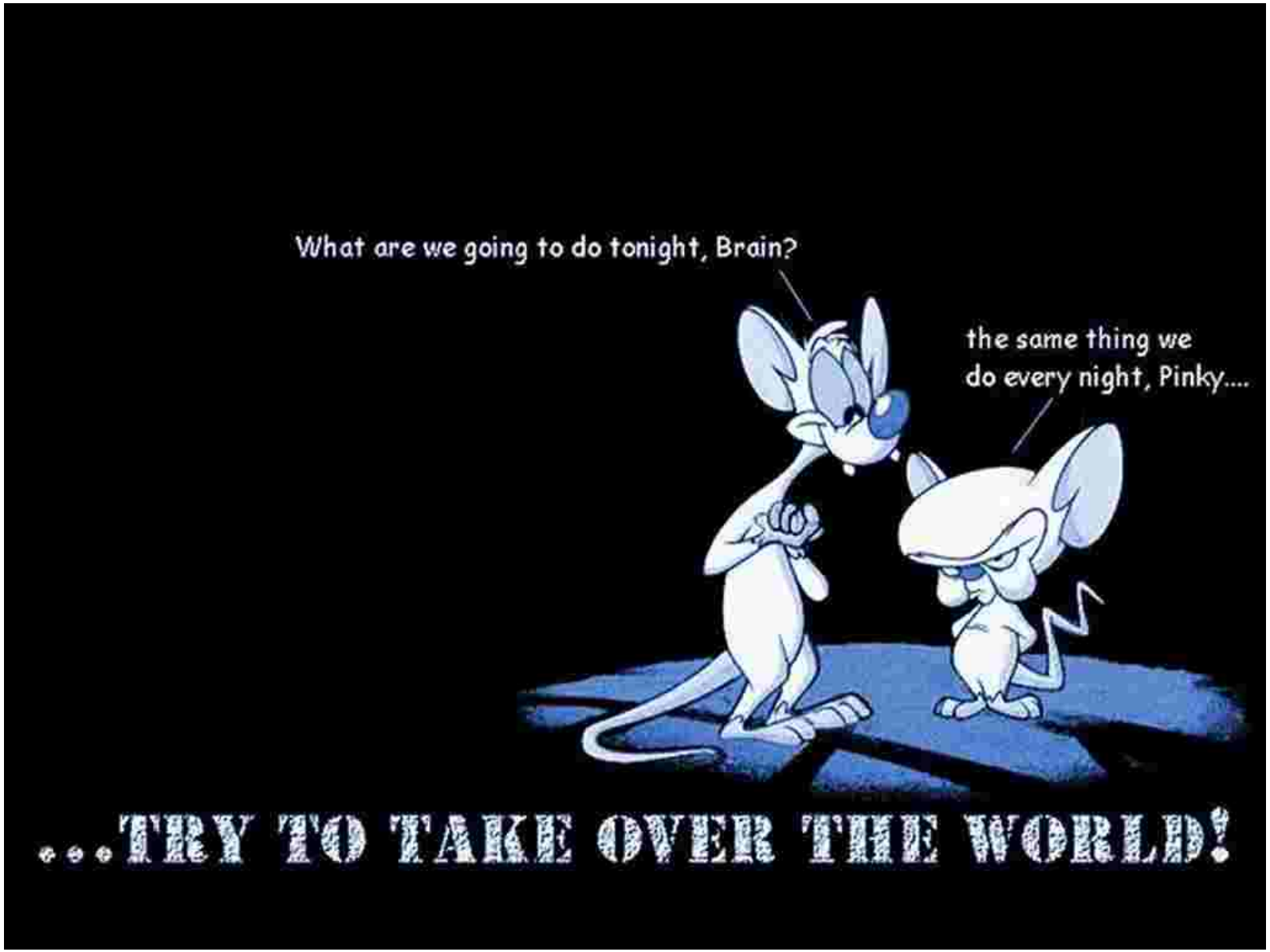
nject Dll (EMET.dll)

Protections/OS	XP SP3	Vista SP1	Win7
GS + SafeSEH	Use data area	Use data area	Use data area
GS + SafeSEH + DEP	Ret-into-libc / Rop	Ret-into-libc / Rop	Ret-into-libc / Rop
GS + SafeSEH + SEHOP		Recreating proper chain, use data area	Recreating proper chain, use data area
GS + SafeSEH + SEHOP + DEP		Recreating proper chain, Rop	Recreating proper chain, Rop
GS + SEHOP + ASLR		Difficult	Difficult
GS + SEHOP + ASLR + DEP		Difficult	Difficult

Thanks

- LSE
- Epita / Epitech
- thrashboul / delroth

Questions ?


What are we going to do tonight, Brain?
the same thing we do every night, Pinky....
...TRY TO TAKE OVER THE WORLD!