



Introduction to stack protections in Windows Vista

Julien Bachmann
Pierre-Olivier Martel

[bachma_j@epita.fr]
[martel_p@epita.fr]





Introduction

- Vista is « dramatically more secure than any other operating system released » (B. Gates)
 - A target loved by hacker
 - 93% of the market
- 3 (good) reasons to study vista ;)





Stack protections

- safeCRT
- Canary
- safeSEH
- ASLR





Buffer overflow (example)

- Bad usage of *strcpy()*
- Shellcode injection
- The return address is modified





```
#include "stdafx.h"  
#include <iostream>
```

```
using namespace std;
```

```
void    bug(_TCHAR* str)  
{  
    _TCHAR    buf[512];  
  
    _tcscpy(buf, str);  
    wcout << buf;  
}
```

```
int _tmain(int argc, _TCHAR* argv[])  
{  
    if (argc <= 1)  
        return 1;  
    bug(argv[1]);  
    return 0;  
}
```





safeCRT

- « Critical » functions are replaced by safer ones
- Those functions check the destination buffer size before copying data into it





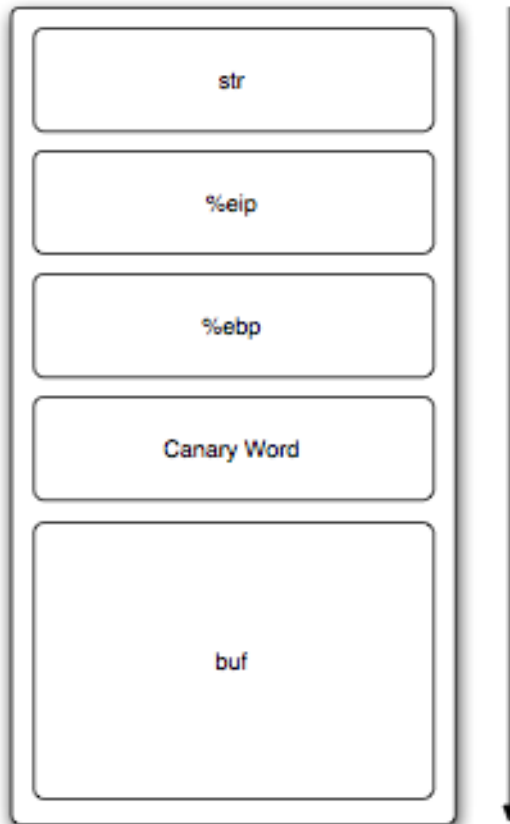
```
void Function(char *s1, char *s2) {  
    char temp[32];  
    strcpy(temp,s1);  
    strcat(temp,s2);  
}
```

```
HRESULT Function(char *s1, char *s2) {  
    char temp[32];  
    HRESULT hr =  
    StringCchCopy(temp,sizeof(temp),s1);  
    if (FAILED(hr)) return hr;  
    return  
    StringCchCat(temp,sizeof(temp),s2);  
}
```





Cookies



- Monitors buffer overflows
- Placed between a buffer and control data
- Only present when a function contains a local static buffer

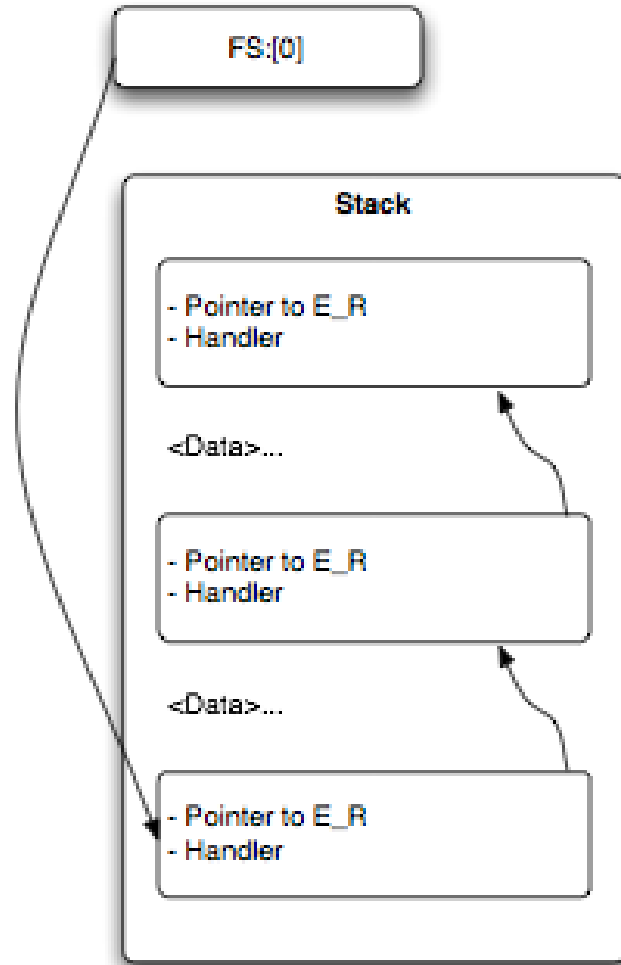




SEH

- Implemented since Windows NT 3.1
- Handle exceptions
- EXCEPTION_REGISTRATION
- The linked list is stored in the stack
- The head is located in *FS:[0]*







safeSEH

- The handler will be called only if:
 - The address does not belong to the stack
 - It is a valid handler from a loaded dll

but...





... guess what 😊

- A handler is also called if:
 - The address is part of a file loaded in memory but not by the target
 - If the address is in the heap (appears less often)





SEH to defeat cookies

- During the epilogue, the cookie is checked
- But exception handling come first !





Concept

- Rewrite the nearest EXCEPTION_REGISTRATION
- Fill all the stack with garbage in order to generate an exception

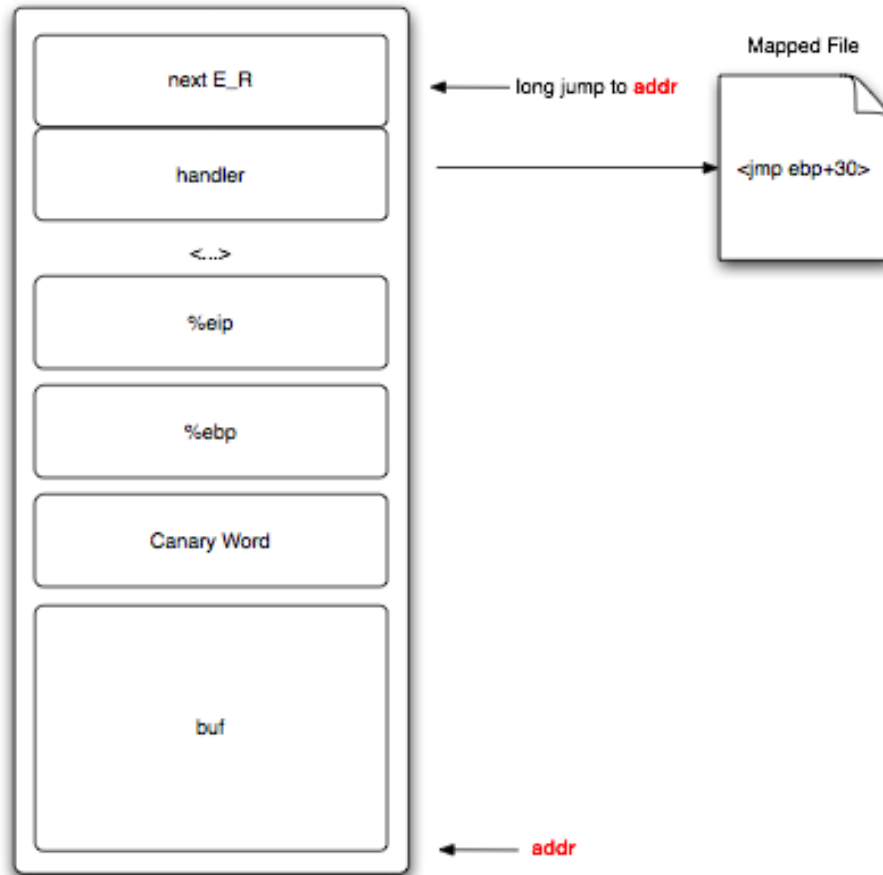




jmp/call

ESP	+8	+14	+1C	+2C	+44	+50
EBP	+0C	+24	+30	-04	-0C	-18



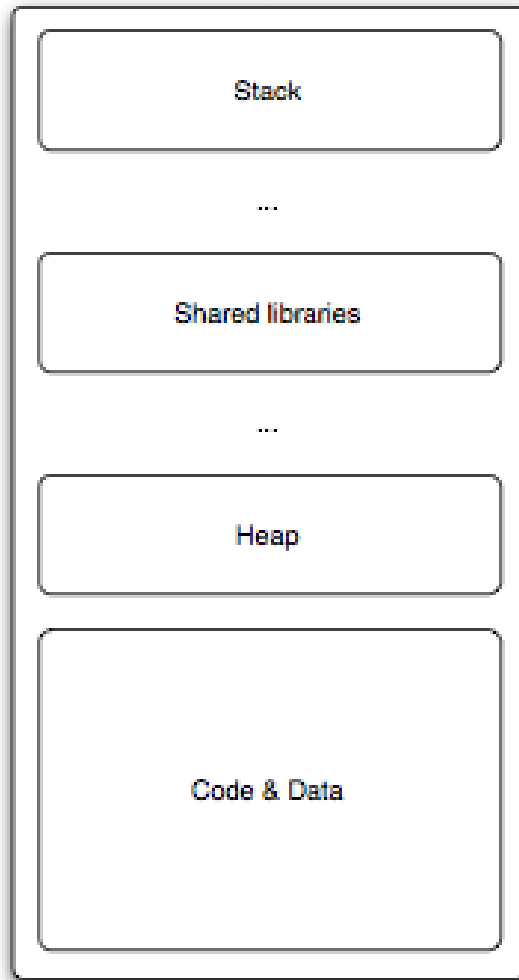


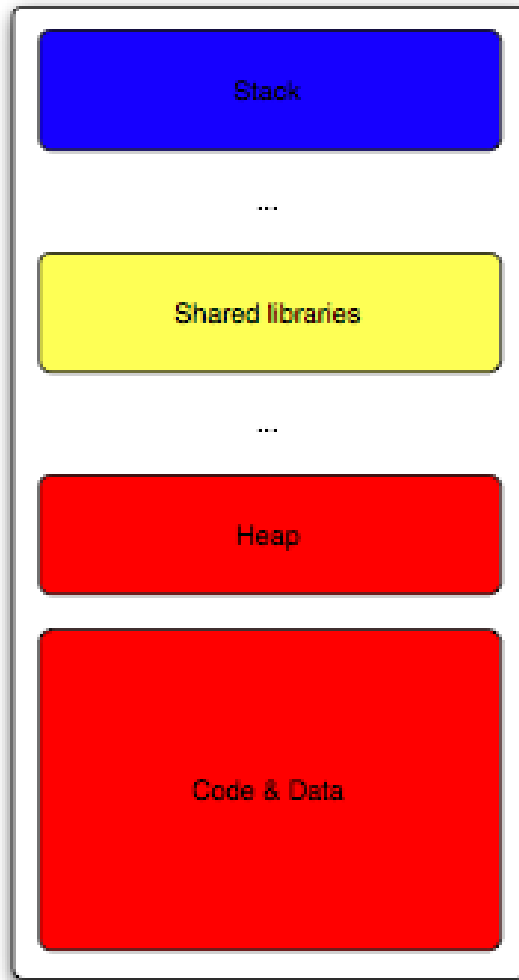


ASLR

- Introduces randomness into addresses used by a given task
- Force an attacker to guess to position of the memory area
- Implemented in OpenBSD, GNU/Linux (PaX & ExecShield, Gentoo Hardened
- Available for program compiled with `/dynamicbase`









Mechanisms

- Base address of each group randomized independently
- Only allow guessing or brute-forcing
- Guessing: For “non-forked” applications since the randomization information are re-newed at launch time
- Brute-forcing: For “forked” applications (network daemons) since each attempt will only make one child crash





Probabilities Quantification

Let's introduce several variables:

- R_k : Number of bits randomized in the 'k' area
- L_k : Least significant bit position in the 'k' area
- A_k : Number of bits of 'k' randomness attacked during an attempt

K can take the following values:

- Executable (x)
- Mappings (m)
- Stack (s)





Now, some maths

- The number of bits of randomness to guess

$$N = (Rs - As) + (Rm - Am) + (Rx - Ax)$$

- The success probabilities using guess method

$$Pg(x) = 1 - (1 - 2^{-N})^x, 0 \leq x$$

- The success probabilities using brute-force method

$$Pb(x) = \frac{x}{2^N}, 0 \leq x \leq 2^N$$





Possible Attacks

- String format bug to determine the return address or the frame pointer of a function, so that we obtain the address of a library
- Reduce the entropy for the stack and the heap:
 - Stack typically aligned to 16 bytes
 - Heap aligned to a memory page (typically 4096 bytes)





Final words





Questions ?

