Offensive Software Exploitation

SEC-300-01/CSI-301-02

Ali Hadi

@binaryz0ne

Exploit Mitigation

Preventing memory corruption techniques!!!

Slides are modified from Memory Corruption 101, NYU Poly, by Dino Dai Zovi

Exploit Mitigation

Cited [1]

- Finding and fixing every vulnerability is impossible
- It is possible to make exploitation more difficult through:
  - Memory page protection
  - Run-time validation
  - Obfuscation and Randomization
- Making every vulnerability non-exploitable is impossible

Timeline of Mitigation

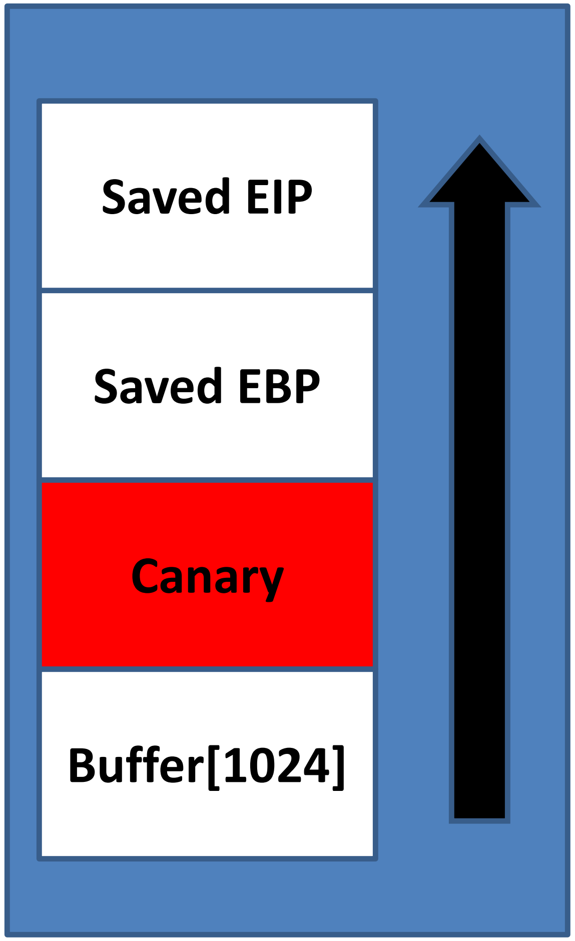
Cited [1]

- Windows 1.0 - Windows XP SP1
  - Corruption of stack and heap metadata is possible
- Windows Server 2003 RTM
  - Operating System is compiled with stack cookies
- Windows XP SP 2
  - Stack/heap cookies, SafeSEH, Software/Hardware DEP
- Windows Vista
  - Address Space Layout Randomization

Visual Studio /GS Flag

Cited [1]

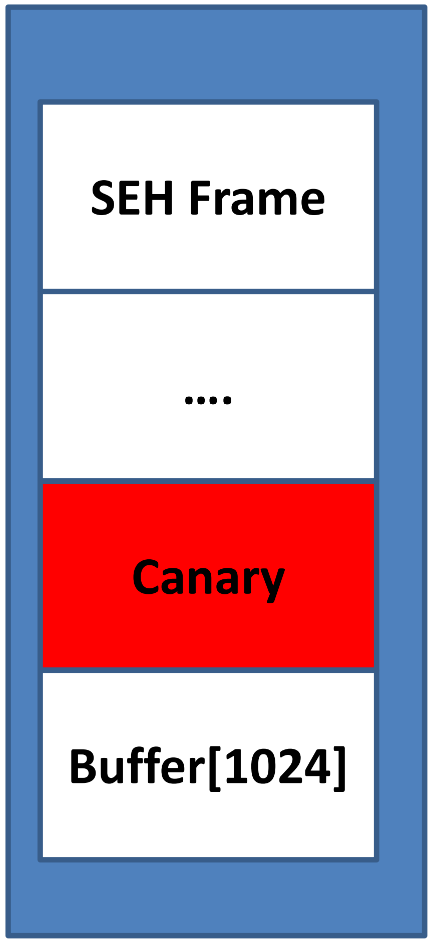
- Place a random “cookie” in the stack frame before frame pointer and return address
- Check cookie before using saved frame pointer and return address



Structured Exception Handling

Cited [1]

- Supports try, except blocks in C and C++ exceptions
- Nested SEH frames are stored on the stack
- Contain pointer to next frame and exception filter function pointer



Visual Studio /SafeSEH

Cited [1]

- Pre-registers all exception handlers in the DLL or EXE
- When an exception occurs, Windows will examine the pre-registered table and only call the handler if it exists in the table
- What if one DLL wasn't compiled w/ SafeSEH?
  - Windows will allow any address in that module as an SEH handler
  - This allows an attacker to still gain full control

References

- Memory Corruption 101, NYU Poly, Dino Dai Zovi
- SEHOP, http://www.sysdream.com/articles/sehop_en.pdf
- Shellcode Storm, http://shell-storm.org/shellcode/
- Stack /GS, https://msdn.microsoft.com/en-us/library/8dbf701c%28VS.80%29.aspx?f=255&MSPPError=-2147217396