IBM Security

Exploit Development 101

ATTACK & DEFENSE HISTORY OF WINDOWS BUFFER OVERFLOW

Peter Chi
chiwp@tw.ibm.com

2020/08/11

IBM

About Me

- IBM CDL Software Engineer
- Columbia Univ. Master of Science
  - Computer Security Track
- OSCP / OSCE / eWPT / eWPTX
- Security Enthusiast

- Contact email - chiwp@tw.ibm.com

Disclaimer of Liability

- The information contained in this presentation and the information presented by the presenter in the live session is for education purpose only, and should not be used in any way against government laws & regulations and IBM's best interests.

- The responsibility of the misuse of the techniques and methods taught in this session should be taken solely by the perpetrator. IBM Taiwan and the presenter do not hold any liability if the participants misuse the information against the law and inflicts damages.

- Tools, techniques, exploitation methods, and any other potentially harmful maneuver should **NOT** be conducted without agreement from the service/application owner. If you are not sure, consult with a subject matter expert. The responsibilities of violating government law & regulations or any other applicable laws and rules should be taken solely by the violator.

Agenda (1/2)

- What is Exploit Development?
  - Concepts
  - Stack-based BufferOverflow
  - Quick Demo
- Attack & Defense Techniques
  - Defense – Security Cookie
  - Attack – SEH based exploit
  - Defense – SafeSEH
  - Defense – DEP (Data Execution Prevention)
  - Attack – ROP (Return Oriented Programming)
  - Defense – ASLR (Address Space Layout Randomization)

Agenda (2/2)

- Summary
- Q&A
- Reference

What is Exploit Development?

Concepts of Exploit Development

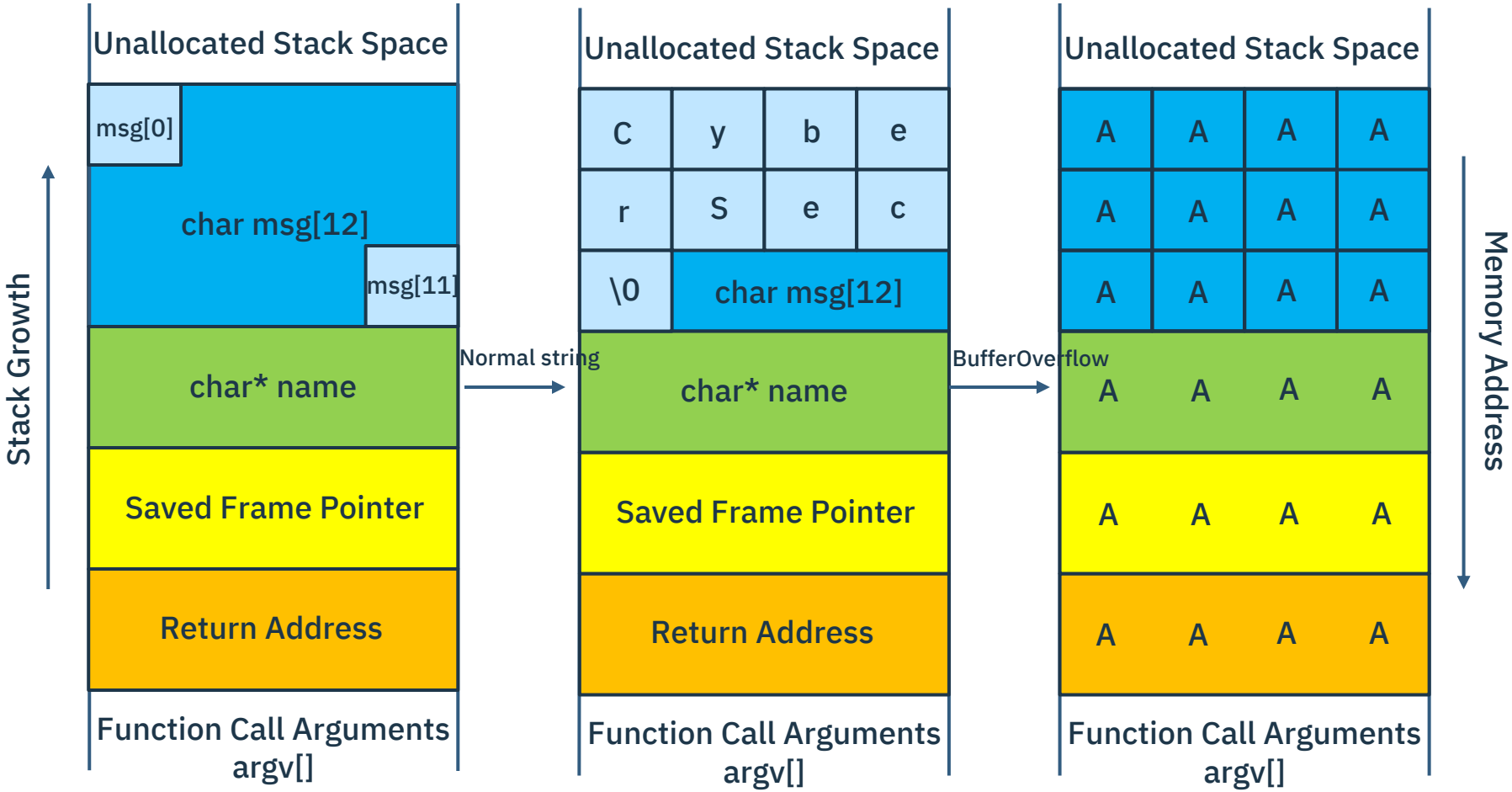
- An exploit is a piece of software, a chunk of data, or a sequence of commands that takes advantage of a bug or vulnerability to cause unintended or unanticipated behavior to occur on computer software, hardware, or something electronic (usually computerized).

  Such behavior frequently includes things like gaining control of a computer system, allowing privilege escalation, or a denial-of-service (DoS or related DDoS) attack.
  (From Wikipedia - https://en.wikipedia.org/wiki/Exploit_%28computer_security%29)

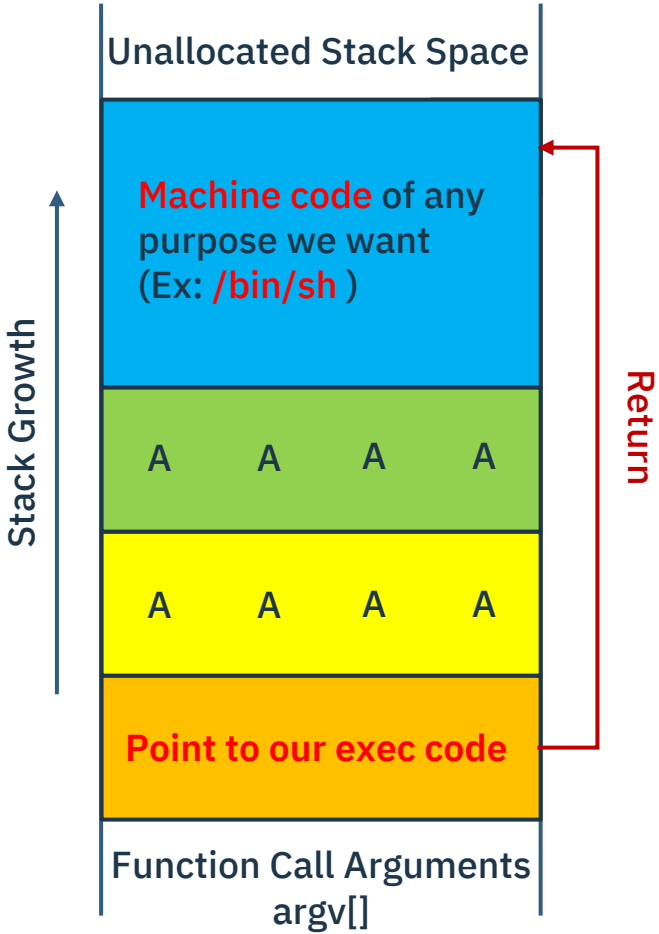
- As a result, an Exploit Development is the process of developing an exploit against certain vulnerability to gain advantages like taking control of a server : P

- The most basic exploit development is to against stack-based buffer overflow vulnerability on a system without any protection mechanism implemented.

Stack-based BufferOverflow (1/2)



Stack-based BufferOverflow (2/2)



Steps to develop an exploit:

- Identify the buffer overflow
  - Fuzzing the input fields to identify a buffer overflow
- Locate the target address (EIP register)
  - Find the relative position of target address
- Remove bad characters
  - Remove the characters which have special meanings to the program
- Input pointer (JMP to certain addr.)
  - Input a pointer that jump to our machine code space
- Generate Machine code of purpose we want
  - Generate the machine code for our purpose
- Send the exploit

Quick Demo

Prereq:

- Freefloat FTP Server 1.0
  - https://www.exploit-db.com/exploits/40681
- Olly Debugger (Or any debugger you like : P)
- Metasploit Framework

Steps:

- Identify the buffer overflow
- Locate the target address (EIP register)
- Remove bad characters
- Input pointer (JMP to certain addr.)
- Generate Machine code of purpose we want
- Send the exploit

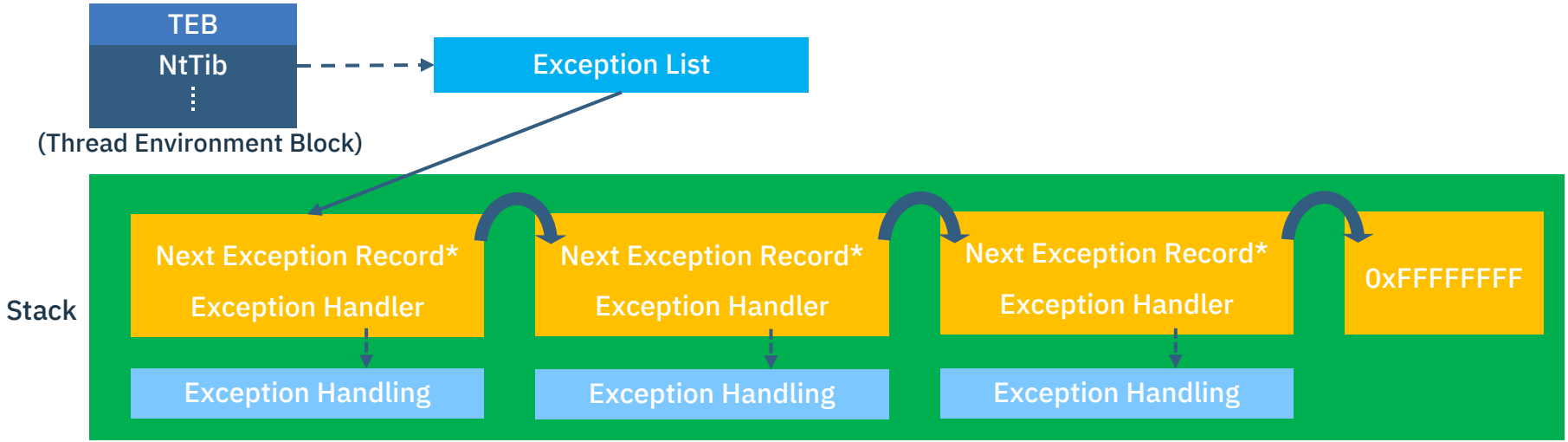
Attack & Defense Techniques

[Defense] Security Cookie (Canary)

- Security Cookie is also called Canary, which is a reference to the historic practice of using canaries in coal mines.
- From 2003, Visual Studio C/C++ will default enable this mechanism by adding /GS into compile parameters.

- The idea is to put a random value in position of the first local parameter(EBP - 4).
- That means if an attacker want to leverage any local parameter buffer overflow to overwrite the Return Address(EBP+4), he/she must also overwrite the Security Cookie.
- As a result, our system could detect if the Return Address had been overwritten by checking the value of Security Cookie(EBP - 4)!

[Attack] SEH-based exploit (Structed Exception Handler)



- Windows uses SEH to handle the exceptions & Windows has a default SEH which will catch exceptions
- The idea of SEH-based exploit is to overwrite the Structed Exception Handler & intentionally cause exception
- As a result, the machine code is executed via Structed Exception Handler (POP POP RETN<Handler> -> ESP+8 <Next Exception Record> -> Short JMP)

[Defense] SafeSEH

- SafeSEH could be enabled by linker's parameter /safeseh, which is not default enabled.
- From Windows XP SP2, the SafeSEH mechanism is introduced.

- The idea is to create a table for recording all the addresses of exception handler
- If any exception handler's address is not pre-recorded in the table, then the program will be terminated

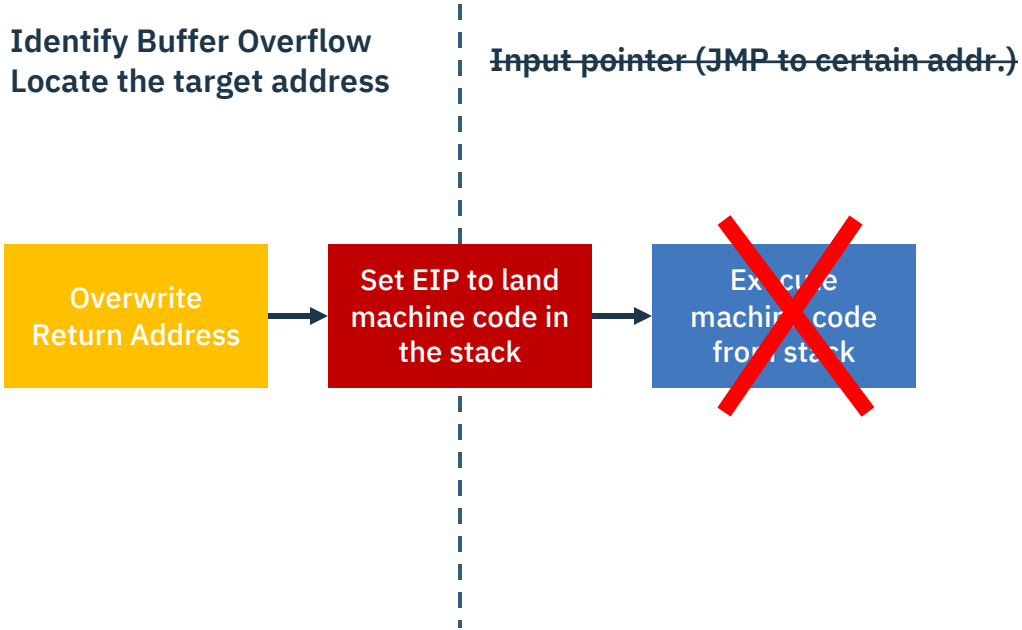
SafeSEH Table

```
 Address
------------
00401A2C   _except_handlerA
00403C3B   _except_handlerB
```

*To make the SafeSEH works, every modules loaded should have the SafeSEH enabled.
It is hard to achieve, especially when a program is developed by multiple parties.

*SEHOP is another mechanism introduced from Windows Server 2008. It is an OS feature, that check the end of SEH is correct.

[Defense] Data Execution Prevention

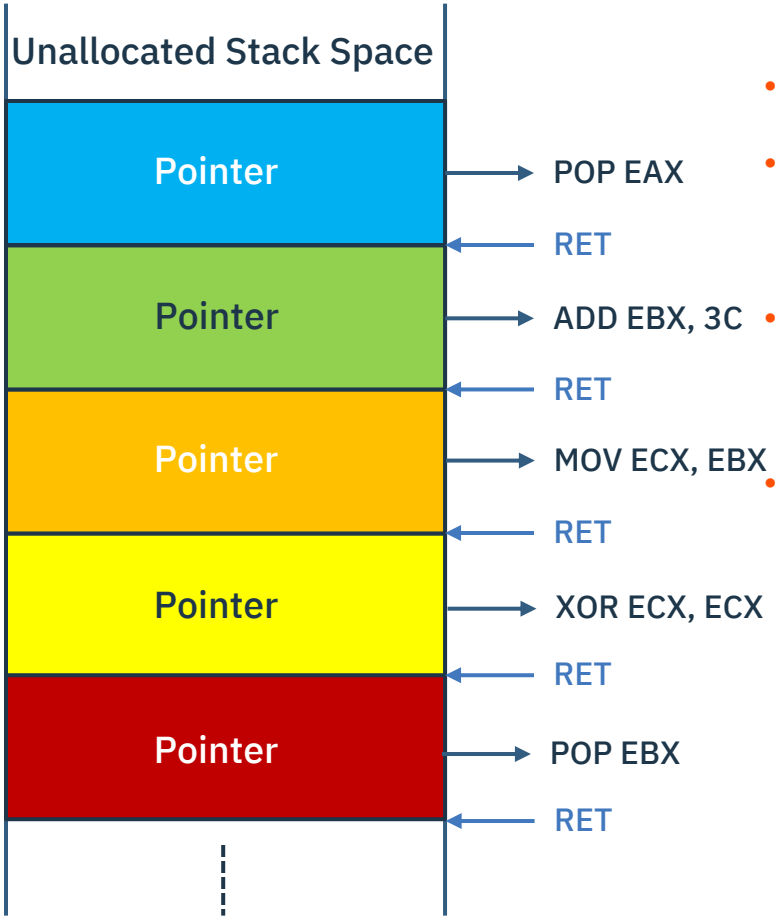


- Entire Stack space is marked as "Non-Executable"
- EIP could still redirect code execution flow to stack, but CPU will reject to execute any code in the stack

- DEP needs the support from CPU (NX – No eXecute) & Operating System (Control NX bit)
- DEP could be enabled by linker's parameter /NXCOMPAT, which is default enabled after Windows Vista & Visual Studio 2005
- From Windows XP SP2 & Server 2003 SP1, DEP is implemented

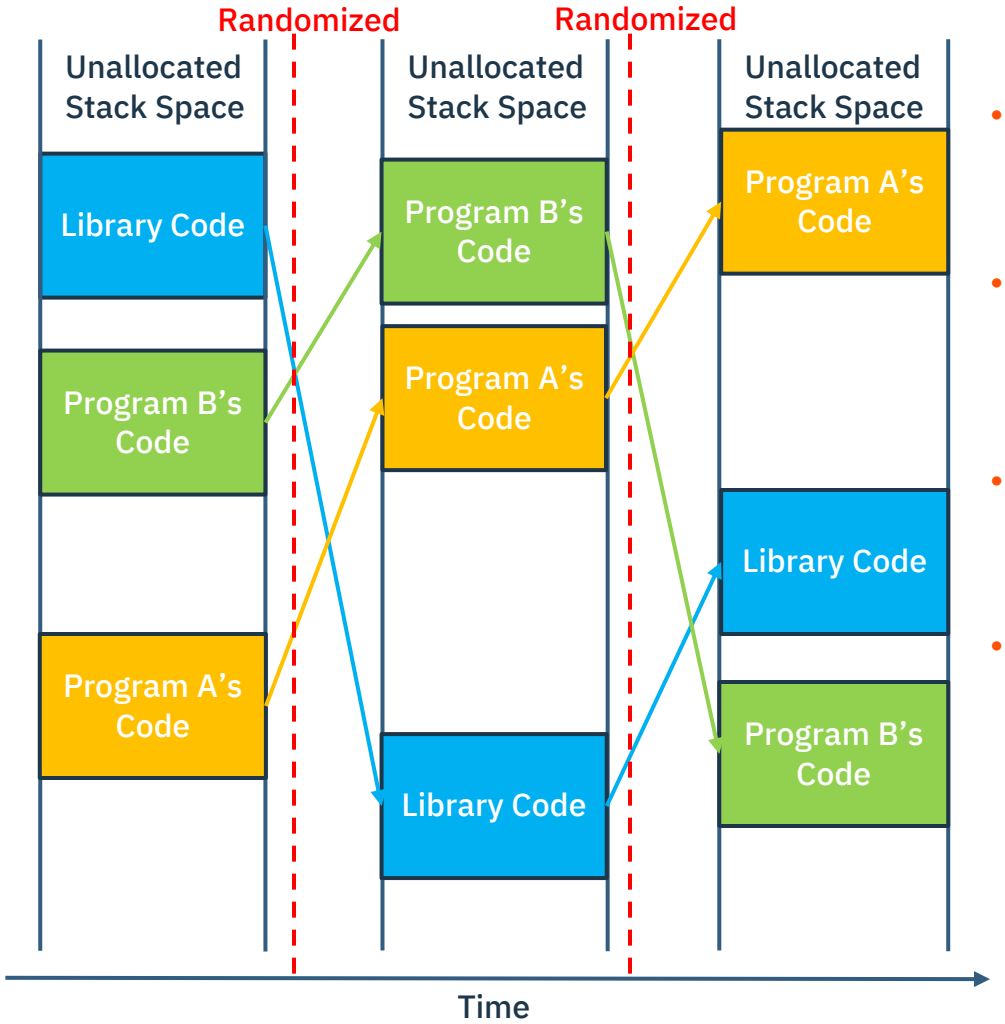
- The idea is to disable the execution permission of stack space
- As a result, no machine code in the stack could be executed

[Attack] Return Oriented Programming



- Based on Return to Libc technique
- Used to mark stack as "Executable" (Bypass DEP)
- In ROP, an attacker needs combine small pieces of code with a few machine language instructions followed by a RET to form a specific machine code
- ROP is usually used to disable DEP via making system call like VirtualProtect(), SetProcessDEPPolicy(), NtSetInformationProcess(), WriteProtectMemory(), etc.

[Defense] Address Space Layout Randomization



- ASLR could be enabled by linker's parameter /DYNAMICBASE, which is default enabled after Visual Studio 2008
- From Windows Vista & Server 2008, ASLR is implemented
- The idea is to randomized the base address of program & library loaded, whenever the system is reboot.
- As a result, attackers can't locate the JMP code or perform ROP easily (Because the address will change every time, the address used when developing won't be always the same.)

Summary

OllyDbg - FTPServer.exe - [/SafeSEH Module Scanner]

P File View Debug Plugins Options Window Help

SEH mode	Base	Limit	Module version	ASLR enable	NX enabled	Module Name
/SafeSEH ON	0x77a10000	0x77a45000	6.1.7600.16385 (win7_rtm.090713	On	On	C:\Windows\system32\WS2_32.dll
/SafeSEH ON	0x75860000	0x758ab000	6.1.7601.17965 (win7sp1_gdr.121	On	On	C:\Windows\system32\KERNELBASE.dll
No SEH	0x75b10000	0x75b1a000	6.1.7600.16385 (win7_rtm.090713	On	On	C:\Windows\system32\LPK.dll
/SafeSEH ON	0x75b20000	0x75b77000	6.1.7600.16385 (win7_rtm.090713	On	On	C:\Windows\system32\SHLWAPI.dll
/SafeSEH ON	0x75c00000	0x75cac000	7.0.7601.17744 (win7sp1_gdr.111	On	On	C:\Windows\system32\msvcrt.dll
/SafeSEH ON	0x75ea0000	0x75f42000	6.1.7600.16385 (win7_rtm.090713	On	On	C:\Windows\system32\RPCRT4.dll
No SEH	0x75f50000	0x75f56000	6.1.7600.16385 (win7_rtm.090713	On	On	C:\Windows\system32\NSI.dll
/SafeSEH ON	0x75f70000	0x7603c000	6.1.7600.16385 (win7_rtm.090713	On	On	C:\Windows\system32\MSCTF.dll
/SafeSEH ON	0x76040000	0x76115000	6.1.7601.17965 (win7sp1_gdr.121	On	On	C:\Windows\system32\kernel32.dll
/SafeSEH ON	0x76120000	0x761bd000	1.0626.7601.17514 (win7sp1_rtm.	On	On	C:\Windows\system32\USP10.dll
/SafeSEH ON	0x76400000	0x7704a000	6.1.7601.17514 (win7sp1_rtm.101	On	On	C:\Windows\system32\SHELL32.dll
/SafeSEH ON	0x77380000	0x7739f000	6.1.7601.17514 (win7sp1_rtm.101	On	On	C:\Windows\system32\IMM32.DLL
/SafeSEH ON	0x77590000	0x77659000	6.1.7601.17514 (win7sp1_rtm.101	On	On	C:\Windows\system32\USER32.dll
/SafeSEH ON	0x77660000	0x776ae000	6.1.7601.17514 (win7sp1_rtm.101	On	On	C:\Windows\system32\GDI32.dll
/SafeSEH ON	0x77800000	0x77942000	6.1.7600.16385 (win7_rtm.090713	On	On	C:\Windows\SYSTEM32\ntdll.dll
/SafeSEH OFF	0x400000	0x40f000		Off	Off	C:\Users\Server\Desktop\FreeFloatFTPServer\Win32\FTPServer.exe

Always remember to check if the protection mechanisms are enabled : P

Q&A

Thank you for your participation :)

Feel free to contact me via chiwp@tw.ibm.com !

Reference

- Wikipedia – Stack Buffer Overflow
  - https://en.wikipedia.org/wiki/Stack_buffer_overflow
- Windows 軟體安全實務 - 緩衝區溢位攻擊
  - http://securityalley.blogspot.com/2014/06/buffer-overflow-windows.html
- made0x78 Security - Binary Exploitation Series (6): Defeating Stack Cookies
  - https://made0x78.com/bseries-defeat-stack-cookies/
- Sam’s Class - CNIT127 Proj 11: Defeating DEP with ROP
  - https://samsclass.info/127/proj/p11-rop.htm
- RAPID7 - Return Oriented Programming (ROP) Exploits Explained
  - https://www.rapid7.com/resources/rop-exploit-explained/

Who is X-Force Red?

X-Force Red is an autonomous team of veteran hackers, within IBM Security, hired to break into organizations and uncover risky vulnerabilities that criminal attackers may use for personal gain.

X-Force Red offers offensive security services which includes penetration testing, vulnerability management programs, red teaming, code review, static analysis and vulnerability assessments.

Their goal is to help security leaders identify and remediate security flaws, covering their entire digital and physical ecosystem.

170 people globally & counting

Industry renown hackers such as:

- Space Rogue
- Evilmog
- Snow
- Videoman
- Q0phi
- retBandit
- keybored
- Q

and more…

X-Force Red Penetration Testing Services

- "Criminal-minded" testing to uncover vulnerabilities
- Virtual and on-site manual testing
- Automated testing using IBM's scanning tool (AppScan) or any third-party scanning tool
- Ad-hoc testing or subscription services
- "Red Labs" IoT, IIoT, OT testing during design and beyond
- ATM-specific testing
- Testing covers various possibilities for intrusion
  - External Threat
  - Insider
  - Malicious User or Customer
  - Hacktivist
- Report findings of what we tried and what we found

Application
- Web
- Mobile
- Terminal
- Thick-client
- Mainframe
- Middleware

Network
- Internal
- External
- Wireless
- Other radio frequencies
- SCADA

Human
- Physical
- Social engineering
- Phishing

Hardware and embedded devices
- Internet of Things
- Wearable technologies
- Point of Sale
- ATMs
- Self-checkout kiosks