Advanced Windows Exploitation

Matteo Memelli

Alexandru Uifalvi

Morten Schenk

All rights reserved to Offensive Security, 2018. No part of this publication, in whole or in part, may be reproduced, copied, transferred or any other right reserved to its copyright owner, including photocopying and all other copying, any transfer or transmission using any network or other means of communication, any broadcast for distant learning, in any form or by any means such as any information storage, transmission or retrieval system, without prior written permission from the author.

Table of Contents

1 Module 0x00 Introduction 8
2 Module 0x01 DEP/ASLR Bypass and Sandbox Escape via Flash Heap Overflow 9
2.1 Lab Objectives 9
2.1.1 Overview 9
2.2 DEP and ASLR 10
2.2.1 Ret2Lib Attacks and Their Evolution 11
2.2.2 Return Oriented Programming Exploitation 12
2.2.3 Address Space Layout Randomization 15
2.2.4 Debugger automation: Pykd and findrop.py 16
2.2.5 Exercises 22
2.3 Flash Player Heap Internals Key Points 23
2.4 Heap Spray: The Technique 27
2.4.1 Exercises 32
2.5 Heap Overflow Case Study: CVE-2015-3104 Proof of Concept 33
2.5.1 Exercises 35
2.6 Heap Overflow Case Study: A Deeper Look at the Bug 36
2.6.1 Exercises 40
2.7 Heap Overflow Case Study: Allocation Control 41
2.7.1 Exercises 48
2.8 Heap Overflow Case Study: Gaining Read/Write Access to the Memory Space 48
2.8.1 Exercises 53
2.9 Heap Overflow Case Study: Defeating ASLR 53
2.9.1 Finding our object in memory 55
2.9.2 Exercises 57
2.9.3 Obtaining NPSWF32 base address 57
2.9.4 Exercises 58
2.10 Heap Overflow Case Study: Gaining code execution 59
2.10.1 Exercises 64
2.11 Heap Overflow Case Study: Stack Pivoting 64
2.11.1 Exercises 65
2.12 Heap Overflow Case Study: Defeating DEP 66

2.12.1 GetModuleHandle ROP Chain ... 66
2.12.2 Exercises ... 68
2.12.3 GetProcAddress ROP Chain ... 69
2.12.4 Exercises ... 70
2.12.5 WriteProcessMemory ROP Chain ... 71
2.12.6 Exercises ... 74
2.13 Executing Shellcode and Restoring the execution flow ... 74
2.13.1 Exercises ... 76
2.14 Sandbox Escape ... 76
2.14.1 Exercises ... 90
2.15 Windows Defender Exploit Guard ... 90
2.16 Testing WDEG Protections on CVE-2015-3104 ... 92
2.17 Disarm vs Bypass ... 95
2.17.1 Disarming WDEG: Theory ... 95
2.17.2 Disarming WDEG: Practice (CVE-2015-3104) ... 98
2.17.3 Exercises ... 99
2.17.4 Defeating EAF ... 99
2.17.5 Exercises ... 100
2.18 Wrapping Up ... 100
3 Module 0x02 CFG/ACG Bypass and Sandbox Escape via Microsoft Edge Type Confusion 101
3.1 Lab Objectives ... 101
3.1.1 Overview ... 101
3.2 64-bit Windows ... 102
3.2.1 Main 64-bit Enhancements ... 104
3.2.2 JavaScript on 64-bit ... 107
3.2.3 Microsoft Edge and WinDbg ... 108
3.3 Type Confusion Case Study: CVE-2017-8601 POC ... 109
3.3.1 Exercises ... 114
3.4 Type Confusion Case Study: Read and Write Primitive ... 114
3.4.1 Exercises ... 126
3.5 Control Flow Guard Theory ... 126
3.5.1 CFG Implementation ... 127

3.5.2 CFG History and Limitations 129
3.5.3 Exercises 129
3.5.4 CFG Bypass Techniques 130
3.6 Type Confusion Case Study: Leaking the stack 131
3.6.1 Exercises 134
3.7 Type Confusion Case Study: RIP Control and Stack Pivot 134
3.7.1 Stack pivoting 138
3.7.2 Confuse me again 139
3.7.3 Exercises 141
3.7.4 ROP preps 141
3.7.5 Exercises 143
3.8 Arbitary Code Guard Theory 143
3.9 ACG Bypass Case Study: CVE-2017-8637 147
3.9.1 ACG Bypass Mechanics 147
3.9.2 Locating the JIT Process Handle 148
3.9.3 Duplicating the JIT Process handle 150
3.9.4 ACG ROP Chain 152
3.9.5 Step 1: DuplicateHandle Call 153
3.9.6 Exercise 156
3.9.7 Step 2: VirtualAllocEx Call 156
3.9.8 Exercise 160
3.9.9 Step 3: WriteProcessMemory Call 160
3.9.10 Exercise 163
3.9.11 Step 4: CreateRemoteThread Call 163
3.9.12 Exercise 168
3.9.13 Step 5: Thread leak 169
3.9.14 Exercise 173
3.9.15 Step 6: Stack manipulation 174
3.9.16 Exercise 177
3.9.17 Step 7: Remote RIP hijacking 177
3.9.18 Exercises 182
3.9.19 Extramile 182

Copyright © Offensive Security Ltd. All rights reserved.

3.10 Type Confusion Case Study: Process Continuation 182
3.10.1 Exercises 187
3.11 AppContainer Sandbox and Code Integrity Guard 188
3.11.1 AppContainer Protections Overview 188
3.11.2 Appcontainer in the Creators Update 190
3.12 Sandbox Escape Case Study: CVE-2016-0165 190
3.13 Sandbox Escape Case Study: Shellcode 191
3.13.1 Exercises 193
3.14 Wrapping up 194
4 Module 0x03 64-bit Kernel Driver Exploitation 195
4.1 Lab Objectives 195
4.2 Overview 195
4.3 Windows I/O System and Device Drivers 195
4.4 Communicating with Drivers 196
4.5 I/O Control Codes 197
4.6 Privilege Levels and Ring0 Payloads 197
4.7 Token Stealing Payload 199
4.8 CVE-2015-5736 : Vulnerability Overview 204
4.9 CVE-2015-5736 : Overwriting the callback function 205
4.9.1 Exercises 211
4.10 CVE-2015-5736 : Triggering the Vulnerable Code 212
4.10.1 Exercises 214
4.11 CVE-2015-5736 : SMEP Says Hello 214
4.11.1 Exercise 216
4.12 CVE-2015-5736 : Introduction to Memory Paging and Structures 217
4.12.1 Exercise 224
4.13 CVE-2015-5736 : The PML4 Self-reference Entry 224
4.14 CVE-2015-5736 : ROP Based Attack 228
4.14.1 Restricted Callers 228
4.14.2 PML4 self-reference entry randomization 232
4.15 CVE-2015-4077 : Memory Leak Vulnerability 233
4.15.1 Bypassing Driver Checks 233

Copyright © Offensive Security Ltd. All rights reserved.

4.15.2 Exercises 245
4.15.3 Leak an Arbitrary Address 246
4.15.4 Exercises 253
4.15.5 Leaking The Kernel Base Address 253
4.15.6 Exercises 260
4.15.7 Leaking Additional DLLs Mapped In Kernel Space 261
4.15.8 Exercises 263
4.16 CVE-2015-5736 : Stack Pivoting 263
4.16.1 Exercises 270
4.17 CVE-2015-5736 : Bypassing SMEP 270
4.17.1 Exercises 274
4.18 CVE-2015-5736 : Restoring the Execution Flow 275
4.18.1 Exercises 283
4.19 Wrapping Up 283