Didier Stevens

Saturday 20 April 2019

Extracting “Stack Strings” from Shellcode

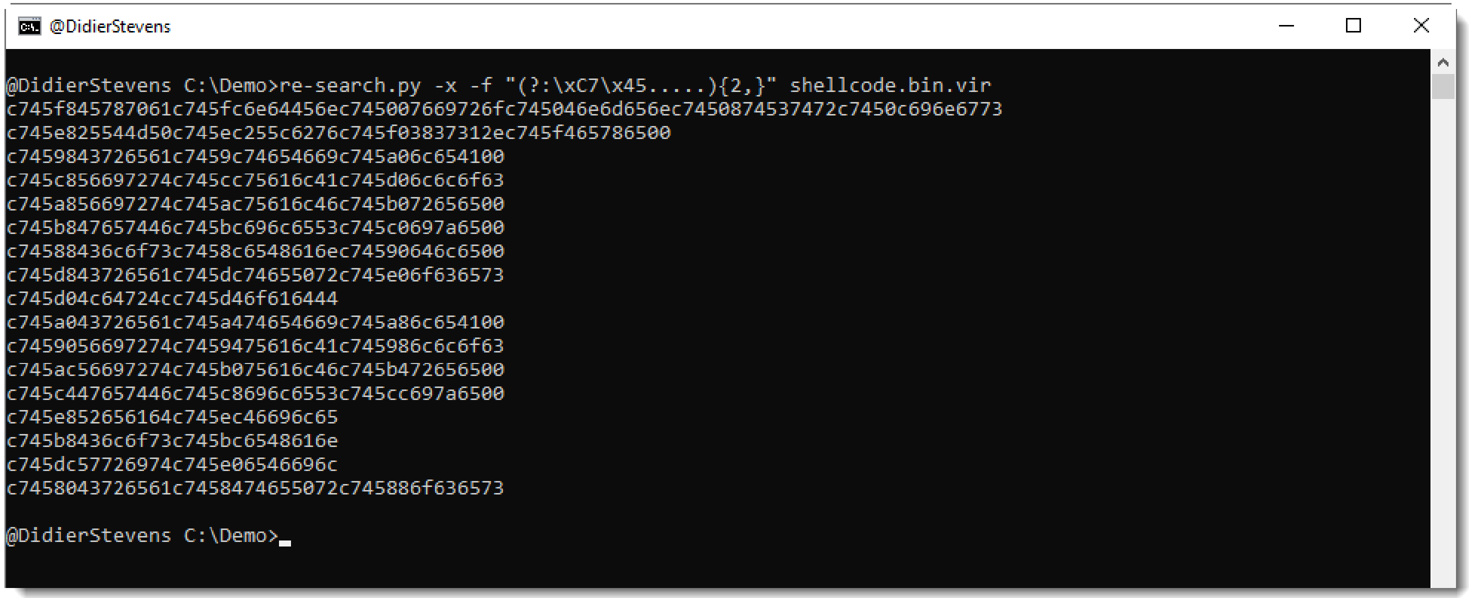
Filed under: Malware,My Software,Reverse Engineering — Didier Stevens @ 0:00

A couple of years ago, I wrote a Python script to enhance Radare2 listings: the script extract strings from stack frame instructions.

Recently, I combined my tools to achieve the same without a 32-bit disassembler: I extract the strings directly from the binary shellcode.

What I’m looking for is sequences of instructions like this: mov dword [ebp – 0x10], 0x61626364. In 32-bit code, that’s C7 45 followed by one byte (offset operand) and 4 bytes (value operand).

Or: C7 45 10 64 63 62 61. I can write a regular expression for this instruction, and use my tool re-search.py to extract it from the binary shellcode. I want at least 2 consecutive mov … instructions: {2,}.



```
@DidierStevens C:\Demo>re-search.py -x -f "(?:\xC7\x45.....){2,}" shellcode.bin.vir
c745f845787061c745fc6e64456ec745007669726fc745046e6d656ec7450874537472c7450c696e6773
c745e825544d50c745ec255c6276c745f03837312ec745f465786500
c7459843726561c7459c74654669c745a06c654100
c745c856697274c745cc75616c41c745d06c6c6f63
c745a856697274c745ac75616c46c745b072656500
c745b847657446c745bc696c6553c745c0697a6500
c74588436c6f73c7458c6548616ec74590646c6500
c745d843726561c745dc74655072c745e06f636573
c745d04c64724cc745d46f616444
c745a043726561c745a474654669c745a86c654100
c7459056697274c7459475616c41c745986c6c6f63
c745ac56697274c745b075616c46c745b472656500
c745c447657446c745c8696c6553c745cc697a6500
c745e852656164c745ec46696c65
c745b8436c6f73c745bc6548616e
c745dc57726974c745e06546696c
c7458043726561c7458474655072c745886f636573

@DidierStevens C:\Demo>
```

I’m using option -f because I want to process a binary file (re-search.py expects text files by default).

And I’m using option -x to produce hexadecimal output (to simplify further processing).

Didier Stevens Labs

Visit my company, Didier Stevens Labs

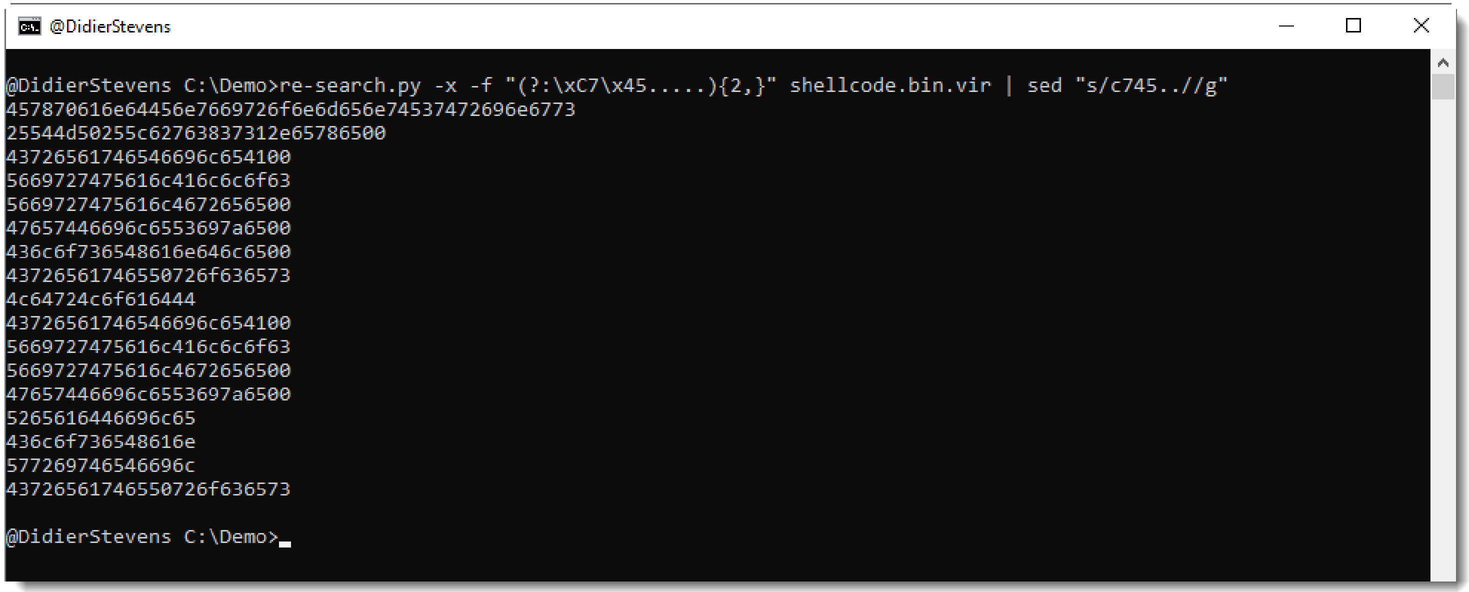
Register for my BruCON training in October:

Malicious Documents For Red Teams

Pages

- About
- Didier Stevens Suite
- Links
- My Software
- Professional
- Programs
  - Ariad
  - Authenticode Tools
  - Binary Tools
  - CASToggle
  - Disitool
  - EICARgen
  - ExtractScripts
  - FileGen
  - HeapLocker
  - Network Appliance Forensic Toolkit
  - Nokia Time Lapse Photography
  - oledump.py

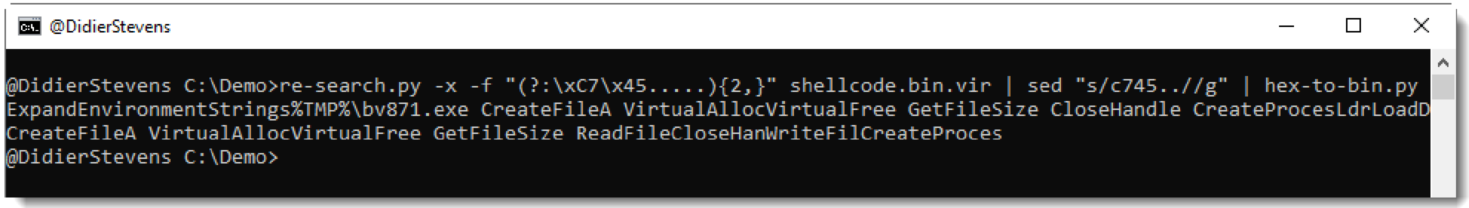
I want to get rid of the bytes for the instruction and the offset operand. I do this with sed:



```
@DidierStevens C:\Demo>re-search.py -x -f "(?:\xC7\x45.....){2,}" shellcode.bin.vir | sed "s/c745..//g"
457870616e64456e7669726f6e6d656e74537472696e6773
25544d50255c62763837312e65786500
43726561746546696c654100
5669727475616c416c6c6f63
5669727475616c4672656500
47657446696c6553697a6500
436c6f736548616e646c6500
43726561746550726f636573
4c64724c6f616444
43726561746546696c654100
5669727475616c416c6c6f63
5669727475616c4672656500
47657446696c6553697a6500
5265616446696c65
436c6f736548616e
577269746546696c
43726561746550726f636573

@DidierStevens C:\Demo>
```

I could convert this back to text with my tool hex-to-bin.py:



```
@DidierStevens C:\Demo>re-search.py -x -f "(?:\xC7\x45.....){2,}" shellcode.bin.vir | sed "s/c745..//g" | hex-to-bin.py
ExpandEnvironmentStrings%TMP%\bv871.exe CreateFileA VirtualAllocVirtualFree GetFileSize CloseHandle CreateProcesLdrLoadD
CreateFileA VirtualAllocVirtualFree GetFileSize ReadFileCloseHanWriteFilCreateProces
@DidierStevens C:\Demo>
```

But that’s not ideal, because now all characters are merged into a single line.

My tool python-per-line.py gives a better result by processing this hexadecimal input line per line:

- OllyStepNSearch
- PDF Tools
- Shellcode
- SpiderMonkey
- Translate
- USBVirusScan
- UserAssist
- VirusTotal Tools
- XORSearch & XORStrings
- YARA Rules
- ZIPEncryptFTP
- Public Drafts
- Cisco Tricks
- Reverse Engineering Mentoring
- Screencasts & Videos

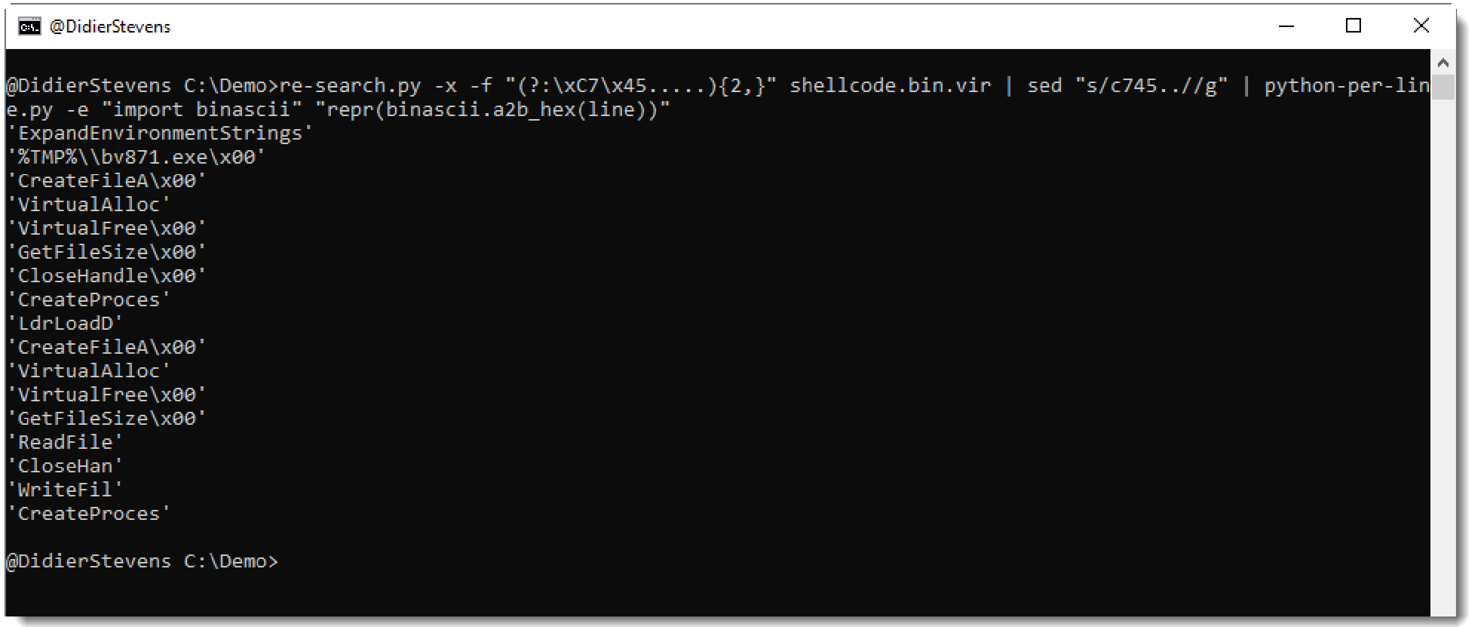
Search

Top Posts

- PDF Tools
- Howto: Make Your Own Cert With OpenSSL on Windows
- oledump.py
- VirusTotal Tools
- Using Metasploit On Windows

Categories

- .NET
- 010 Editor
- Announcement
- Arduino
- Bash Bunny
- Beta
- bpmtk
- Certification
- Didier Stevens Labs
- Eee PC
- Encryption
- Entertainment
- Fellow Bloggers
- Forensics
- Hacking
- Hardware
- maldoc



```
@DidierStevens C:\Demo>re-search.py -x -f "(?:\xC7\x45.....){2,}" shellcode.bin.vir | sed "s/c745..//g" | python-per-line.py -e "import binascii" "repr(binascii.a2b_hex(line))"
'ExpandEnvironmentStrings'
'%TMP%\\bv871.exe\x00'
'CreateFileA\x00'
'VirtualAlloc'
'VirtualFree\x00'
'GetFileSize\x00'
'CloseHandle\x00'
'CreateProces'
'LdrLoadD'
'CreateFileA\x00'
'VirtualAlloc'
'VirtualFree\x00'
'GetFileSize\x00'
'ReadFile'
'CloseHan'
'WriteFil'
'CreateProces'

@DidierStevens C:\Demo>
```

Remark that I also use function repr to escape unprintable characters like 00.

This output provides a good overview of all API functions called by this shellcode.

If you take a close look, you'll notice that the last strings are incomplete: that's because they are missing one or two characters, and these are put on the stack with another mov instruction for single or double bytes. I can accommodate my regular expression to take these instructions into account:

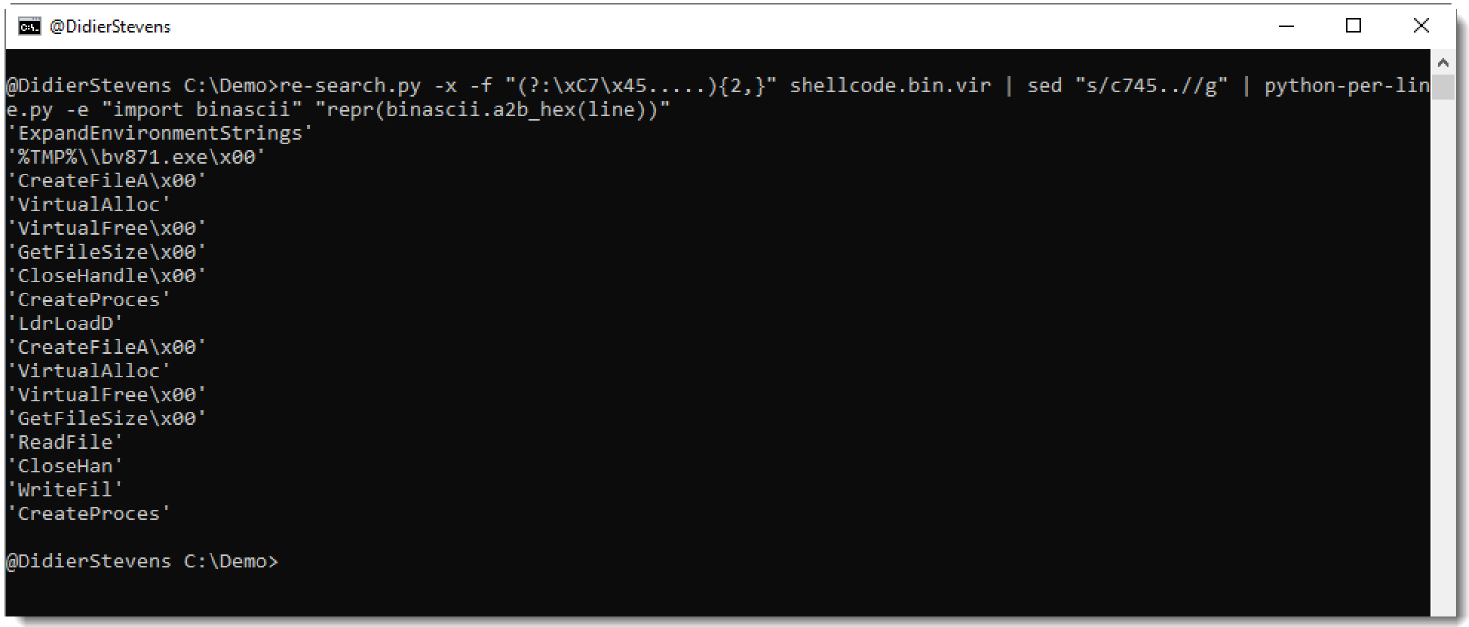
- Malware
- My Software
- N800
- Networking
- Nonsense
- nslu2
- OSX
- PDF
- Personal
- Physical Security
- Poll
- Puzzle
- Quickpost
- Release
- Reverse Engineering
- RFID
- Shellcode
- smart card
- Spam
- technology
- UltraEdit
- Uncategorized
- Update
- Vulnerabilities
- WiFi
- Windows 7
- Windows 8
- Windows Vista
- Wireshark

Blog Stats

- 5,855,853 hits

Twitter @DidierStevens

- Flying back from #barcelona #r2con2019 and passing the #MontBlanc :-) https://t.co/LRHjRNCfMO 2 days ago
- RT @markrussinovich: New Sysinternals updates, including major Sysmon filtering enhancements, including nested rules and "contains any" and… 4 days ago



```
@DidierStevens C:\Demo>re-search.py -x -f "(?:\xC7\x45.....){2,}" shellcode.bin.vir | sed "s/c745..//g" | python-per-line.py -e "import binascii" "repr(binascii.a2b_hex(line))"
'ExpandEnvironmentStrings'
'%TMP%\\bv871.exe\x00'
'CreateFileA\x00'
'VirtualAlloc'
'VirtualFree\x00'
'GetFileSize\x00'
'CloseHandle\x00'
'CreateProces'
'LdrLoadD'
'CreateFileA\x00'
'VirtualAlloc'
'VirtualFree\x00'
'GetFileSize\x00'
'ReadFile'
'CloseHan'
'WriteFil'
'CreateProces'

@DidierStevens C:\Demo>
```

This is the complete command:

```
re-search.py -x -f "(?:\xC7\x45.....){2,}(?:(?:\xC6\x45..)|(?:\x66\xC7\x45...))?" shellcode.bin.vir | sed "s/66c745..//g" | sed "s/c[67]45..//g" | python-per-line.py -e "import binascii" "repr(binascii.a2b_hex(line))"
```

Share this:

Twitter Facebook

Related

Maldoc With Process Hollowing Shellcode
In "maldoc"

Update: XORSearch With Shellcode Detector
In "My Software"

Update: XORSearch Version 1.9.2
In "Forensics"

Comments (1)

1 Comment »

1. […] Extracting “Stack Strings” from Shellcode […]

Pingback by Overview of Content Published in April | Didier Stevens — Sunday 19 May 2019 @ 7:55

- RT @xme: [/dev/random] #BruCON Challenge: The Solution blog.rootshell.be/2019/09/06/bru… 5 days ago
- RT @r2gui: We just released Cutter v1.9 on #r2con2019 with a HUGE surprise - a full integration of Ghidra decompiler in Cutter! 🥳 Download… 5 days ago
- #r2con #r2con2019 R2Ghidra decompiler github github.com/radareorg/r2gh… 5 days ago

Archives

- August 2019
- July 2019
- June 2019
- May 2019
- April 2019
- March 2019
- February 2019
- January 2019
- December 2018
- November 2018
- October 2018
- September 2018
- August 2018
- July 2018
- June 2018
- May 2018
- April 2018
- March 2018
- February 2018
- January 2018
- December 2017
- November 2017
- October 2017
- September 2017
- August 2017
- July 2017
- June 2017
- May 2017
- April 2017

RSS feed for comments on this post. TrackBack URI

Leave a Reply (comments are moderated)

Enter your comment here...

This site uses Akismet to reduce spam. Learn how your comment data is processed.

- March 2017
- February 2017
- January 2017
- December 2016
- November 2016
- October 2016
- September 2016
- August 2016
- July 2016
- June 2016
- May 2016
- April 2016
- March 2016
- February 2016
- January 2016
- December 2015
- November 2015
- October 2015
- September 2015
- August 2015
- July 2015
- June 2015
- May 2015
- April 2015
- March 2015
- February 2015
- January 2015
- December 2014
- November 2014
- October 2014
- September 2014
- August 2014
- July 2014
- June 2014
- May 2014
- April 2014
- March 2014
- February 2014
- January 2014
- December 2013
- November 2013
- October 2013
- September 2013
- August 2013
- July 2013
- June 2013
- May 2013
- April 2013

- March 2013
- February 2013
- January 2013
- December 2012
- November 2012
- October 2012
- September 2012
- August 2012
- July 2012
- June 2012
- May 2012
- April 2012
- March 2012
- February 2012
- January 2012
- December 2011
- November 2011
- October 2011
- September 2011
- August 2011
- July 2011
- June 2011
- May 2011
- April 2011
- March 2011
- February 2011
- January 2011
- December 2010
- November 2010
- October 2010
- September 2010
- August 2010
- July 2010
- June 2010
- May 2010
- April 2010
- March 2010
- February 2010
- January 2010
- December 2009
- November 2009
- October 2009
- September 2009
- August 2009
- July 2009
- June 2009
- May 2009
- April 2009

- March 2009
- February 2009
- January 2009
- December 2008
- November 2008
- October 2008
- September 2008
- August 2008
- July 2008
- June 2008
- May 2008
- April 2008
- March 2008
- February 2008
- January 2008
- December 2007
- November 2007
- October 2007
- September 2007
- August 2007
- July 2007
- June 2007
- May 2007
- April 2007
- March 2007
- February 2007
- January 2007
- December 2006
- November 2006
- October 2006
- September 2006
- August 2006
- July 2006
- June 2006

April 2019

M	T	W	T	F	S	S
1	2	3	4	5	6	7
8	9	10	11	12	13	14
15	16	17	18	19	20	21
22	23	24	25	26	27	28
29	30					

« Mar **May »**