Lesson

```
[*] Question number 0x1:

Is this a '32-bit' or '64-bit' ELF? (e.g. 1337-bit)

>> |
```

```
┌──(vigneswar㉿VigneswarPC)-[~/Pwn/Lesson/challenge]
└─$ file main
main: ELF 64-bit LSB executable, x86-64, version 1 (SYSV), dynamically linked, interpreter ./glibc/ld-linux-x86-64.so.2,
 BuildID[sha1]=da663acb70f9fa157a543a6c4affd05e53fbcb07, for GNU/Linux 3.2.0, not stripped
```

```
vigneswar@VigneswarPC: ~

⊙⊙⊙⊙⊙⊙⊙⊙⊙⊙⊙⊙⊙⊙⊙⊙⊙⊙⊙⊙⊙⊙⊙⊙⊙⊙⊙⊙⊙⊙⊙⊙⊙⊙⊙⊙⊙⊙⊙⊙⊙⊙⊙⊙⊙⊙⊙⊙⊙⊙⊙⊙⊙⊙⊙⊙⊙⊙⊙⊙⊙⊙⊙⊙⊙⊙⊙⊙⊙⊙
⊙                                                                    ⊙
⊙  HINT: Run 'gdb ./main' to open the binary in the debugger, then   ⊙
⊙        run 'checksec' to see the protections.                       ⊙
⊙                                                                    ⊙
⊙⊙⊙⊙⊙⊙⊙⊙⊙⊙⊙⊙⊙⊙⊙⊙⊙⊙⊙⊙⊙⊙⊙⊙⊙⊙⊙⊙⊙⊙⊙⊙⊙⊙⊙⊙⊙⊙⊙⊙⊙⊙⊙⊙⊙⊙⊙⊙⊙⊙⊙⊙⊙⊙⊙⊙⊙⊙⊙⊙⊙⊙⊙⊙⊙⊙⊙⊙⊙⊙

[*] Question number 0x2:

Which of these 3 protections are enabled (Canary, NX, PIE)?

>> |
```

```
┌──(vigneswar㉿VigneswarPC)-[~/Pwn/Lesson/challenge]
└─$ checksec main
[*] '/home/vigneswar/Pwn/Lesson/challenge/main'
    Arch:       amd64-64-little
    RELRO:      Full RELRO
    Stack:      No canary found
    NX:         NX enabled
    PIE:        No PIE (0x400000)
    RUNPATH:    b'./glibc/'
```

```
⊙⊙⊙⊙⊙⊙⊙⊙⊙⊙⊙⊙⊙⊙⊙⊙⊙⊙⊙⊙⊙⊙⊙⊙⊙⊙⊙⊙⊙⊙⊙⊙⊙⊙⊙⊙⊙⊙⊙⊙⊙⊙⊙⊙⊙⊙⊙⊙⊙⊙⊙⊙⊙⊙⊙⊙⊙⊙⊙⊙⊙
⊙                                                           ⊙
⊙  HINT: Pay attention to the 'void print_msg(char *user)'  ⊙
⊙        and the 'strncmp(arg1, arg2, n_bytes)'.            ⊙
⊙                                                           ⊙
⊙⊙⊙⊙⊙⊙⊙⊙⊙⊙⊙⊙⊙⊙⊙⊙⊙⊙⊙⊙⊙⊙⊙⊙⊙⊙⊙⊙⊙⊙⊙⊙⊙⊙⊙⊙⊙⊙⊙⊙⊙⊙⊙⊙⊙⊙⊙⊙⊙⊙⊙⊙⊙⊙⊙⊙⊙⊙⊙⊙⊙

[*] Question number 0x3:

What do you need to enter so the message 'Welcome admin!' is printed?

>> |
```

```
void print_msg(char *user){
  char formatter[0x20];
  strncpy(formatter, user, 5);
  for (size_t i = 0; i < 5; i++) formatter[i] = tolower(formatter[i]);
  printf(strncmp(formatter, "admin", 5) == 0 ? "\nWelcome admin!\n\n" : "\nWelcome user!\n\n");
}
```

```
⊙⊙⊙⊙⊙⊙⊙⊙⊙⊙⊙⊙⊙⊙⊙⊙⊙⊙⊙⊙⊙⊙⊙⊙⊙⊙⊙⊙⊙⊙⊙⊙⊙⊙⊙⊙⊙⊙⊙⊙⊙⊙⊙⊙⊙⊙⊙⊙⊙⊙⊙⊙⊙⊙⊙⊙⊙
⊙                                                       ⊙
⊙  HINT: This is the buffer --> char name[0x20] = {0};  ⊙
⊙                                                       ⊙
⊙⊙⊙⊙⊙⊙⊙⊙⊙⊙⊙⊙⊙⊙⊙⊙⊙⊙⊙⊙⊙⊙⊙⊙⊙⊙⊙⊙⊙⊙⊙⊙⊙⊙⊙⊙⊙⊙⊙⊙⊙⊙⊙⊙⊙⊙⊙⊙⊙⊙⊙⊙⊙⊙⊙⊙⊙

[*] Question number 0x4:

What is the size of the 'name' buffer (in hex or decimal)?

>> |
```

```
int main(int argc, char **argv){
   char name[0x20] = {0};
   unsigned long x, y;
   printf("Enter your name: ");
   scanf("%s", name);
   print_msg(name);
   return 0;
}
```

```
oooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooo
o                                                               o
o  HINT: Only functions inside 'main()' are called.            o
o        Also, the functions these functions call.              o
o                                                               o
oooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooo

[*] Question number 0x5:

Which custom function is never called? (e.g. vuln())

>>
```

```
void under_construction(){
   printf("This is under development\n");
}
```

```
oooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooo
o                                                            o
o  HINT: Which function reads the string from the stdin?     o
o                                                            o
oooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooo

[*] Question number 0x6:

What is the name of the standard function that could trigger a Buffer Overflow? (e.g. fprintf())

>>
```

```
int main(int argc, char **argv){
   char name[0x20] = {0};
   unsigned long x, y;
   printf("Enter your name: ");
   scanf("%s", name);
   print_msg(name);
   return 0;
}
```

scanf

```
oooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooo
o                                                                  o
o  HINT: A Segmentation Fault occurs when the return              o
o        address is overwritten with an invalid address.           o
o                                                                  o
oooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooo

[*] Question number 0x7:

Insert 30, then 39, then 40 'A's in the program and see the output.

After how many bytes a Segmentation Fault occurs (in hex or decimal)?

>>
```

32 + 8 (rbp) + newline -> 40 A

```
oooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooo
o                                                                                   o
o  HINT: Run 'gdb ./main' to open the binary in the debugger, then                  o
o        run 'p <function_name>' to see the address of a function.                  o
o                                                                                   o
o        e.g. pwndbg> p main                                                        o
o             $2 = {<text variable, no debug info>} 0x401294 <main>                 o
o                                                                                   o
oooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooooo

[*] Question number 0x8:

What is the address of 'under_construction()' in hex? (e.g. 0x401337)

>>
```

```
┌──(vigneswar㉿VigneswarPC)-[~/Pwn/Lesson/challenge]
└─$ objdump -d main -M intel | grep under
00000000004011d6 <under_construction>:
```