DESCRIPTION

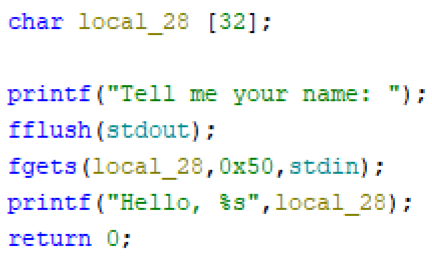
It's very easy. Connect to 141.85.224.106:31345 and get the flag.

RESOURCES

As part of the challenge I received an executable file called **piece_of_cake** as an attachment for analysis.

APPROACHES

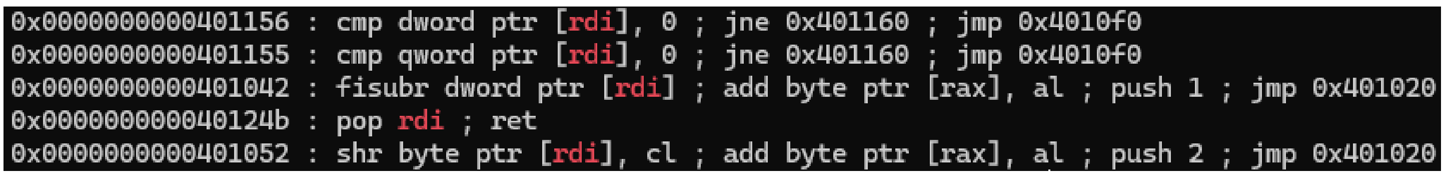
1. The first approach here was to run the program and after running it I could see that it is requiring a input string which after it is provided it is returned back;
2. Thus, I had no other idea than to decompile the program using Ghidra. The decompilation can be seen below:



```c
char local_28 [32];

printf("Tell me your name: ");
fflush(stdout);
fgets(local_28,0x50,stdin);
printf("Hello, %s",local_28);
return 0;
```

3. Here we can directly see that we have a buffer overflow as we have a buffer of only 0x20 characters and we are reading 0x50 from the keyboard.
4. Additionally there was present in the binary another call to **system** at address **0x40117f**, which if we call with **rdi = „sh"** we would obtain a shell and could get the flag. Thus, I tried to get a gadget that modifies **rdi** and then returns to a specified address:



```
0x0000000000401156 : cmp dword ptr [rdi], 0 ; jne 0x401160 ; jmp 0x4010f0
0x0000000000401155 : cmp qword ptr [rdi], 0 ; jne 0x401160 ; jmp 0x4010f0
0x0000000000401042 : fisubr dword ptr [rdi] ; add byte ptr [rax], al ; push 1 ; jmp 0x401020
0x000000000040124b : pop rdi ; ret
0x0000000000401052 : shr byte ptr [rdi], cl ; add byte ptr [rax], al ; push 2 ; jmp 0x401020
```

5. Thus, we can see that at address **0x40124b**. Then using the pwntools features, we search for the presence of a „sh" string inside the binary and we find one.
6. Then, the overflow from keyboard can be described as follows:
   a. We jump to address 0x40124b where the gadget is present;
   b. Then, the next instruction executed will be „pop rdi" so we make sure that on the stack the next present entry is the address returned by the instruction **next(e.search(b'sh'))**
   c. Then, we execute a **ret** and we make sure that on the stack the next instruction will be the address of the **system** call, **0x40117f**.
7. If we execute the created script now (**python3 script.py**) we will obtain a shell and if we go to path **/home/ctf/flag** we will obtain the flag which is:

CNS_CTF{db70613f2f45109dfbb301ca1bdcdcd5}