Opening the binary and studying it, we can see that there is the *decrypt_flag* function. Disassemble it and break just before it prints the break-line.

After it decrypts, print the flag as string. We can use printf and cast the *flag_buf* variable into *char* * by doing `printf "%s", (char *) flag_buf`.

```
(gdb) disas decrypt_flag
...
...
0x0000000000400878 <+242>:
0x0000000000400896 <+272>:    mov    rdx,QWORD PTR [rip+0x200b4b]        #
0x6013e8 <flag_buf>
0x000000000040089d <+279>:    mov    eax,DWORD PTR [rbp-0x20]
0x00000000004008a0 <+282>:    cdqe
0x00000000004008a2 <+284>:    add    rax,rdx
0x00000000004008a5 <+287>:    mov    BYTE PTR [rax],0x0
0x00000000004008a8 <+290>:    mov    edi,0xa
0x00000000004008ad <+295>:    call   0x4005f0 <putchar@plt> ; Prints break-
line
...
...
(gdb) b *0x00000000004008a8
Breakpoint 1 at 0x4008a8
(gdb) r
Starting program: run
Decrypting the Flag into global variable 'flag_buf'
.....................................
(gdb) printf "%s", (char*) flag_buf
picoCTF{gDb_iS_sUp3r_u53fuL_a6c61d82}
```

Alternatively, you can use the command x/s flag_buf to see the content of the variable.

So the flag is: `picoCTF{gDb_iS_sUp3r_u53fuL_a6c61d82}`