Nº a93280

Aluno: Miguel Ângelo Machado Martins

PL: 05

---

(1)

```
int main()
{
        char* linha;
        linha = getline();
        return 0;
}
```

(2)

```
[00ha93280@setpe8] $ gcc -o tpe8 -g tpe8.c
[00ha.93280@setpe8] $ gdb
(gdb) file tpe8
(gdb) b main
(gdb) b getline
(gdb) run
(gdb) disas getline
```

Dump of assembler code for function getline: | fp → frame pointer

```
0x08048400 <getline+0>:  push %ebp  // Inicialização da função/fp antigo
0x08048401 <getline+1>:  mov  %esp,%ebp  // Ligação do novo fp
0x08048403 <getline+3>:  sub  $0x18,%esp  // Guarda espaço para o array (buf[8])
0x08048406 <getline+6>:  sub  $0x4,%esp  // guarda espaço para o endereço de retorno
0x08048409 <getline+9>:  lea  0xfffffff8(%ebp),%eax  // Inicialização do array
0x0804840e <getline+12>:
                         push %eax  // Salvaguarda regist %eax
0x0804840d <getline+13>:
                         call 0x8048304  // Invocação da gets
```

(3) Depois de fazer o ex 2:

(gdb) run tp8

(gdb) c
(gdb) c } Para continuar, visto que, para fazer o exercício 2 tinha posto breakpoints

→ Mau input:

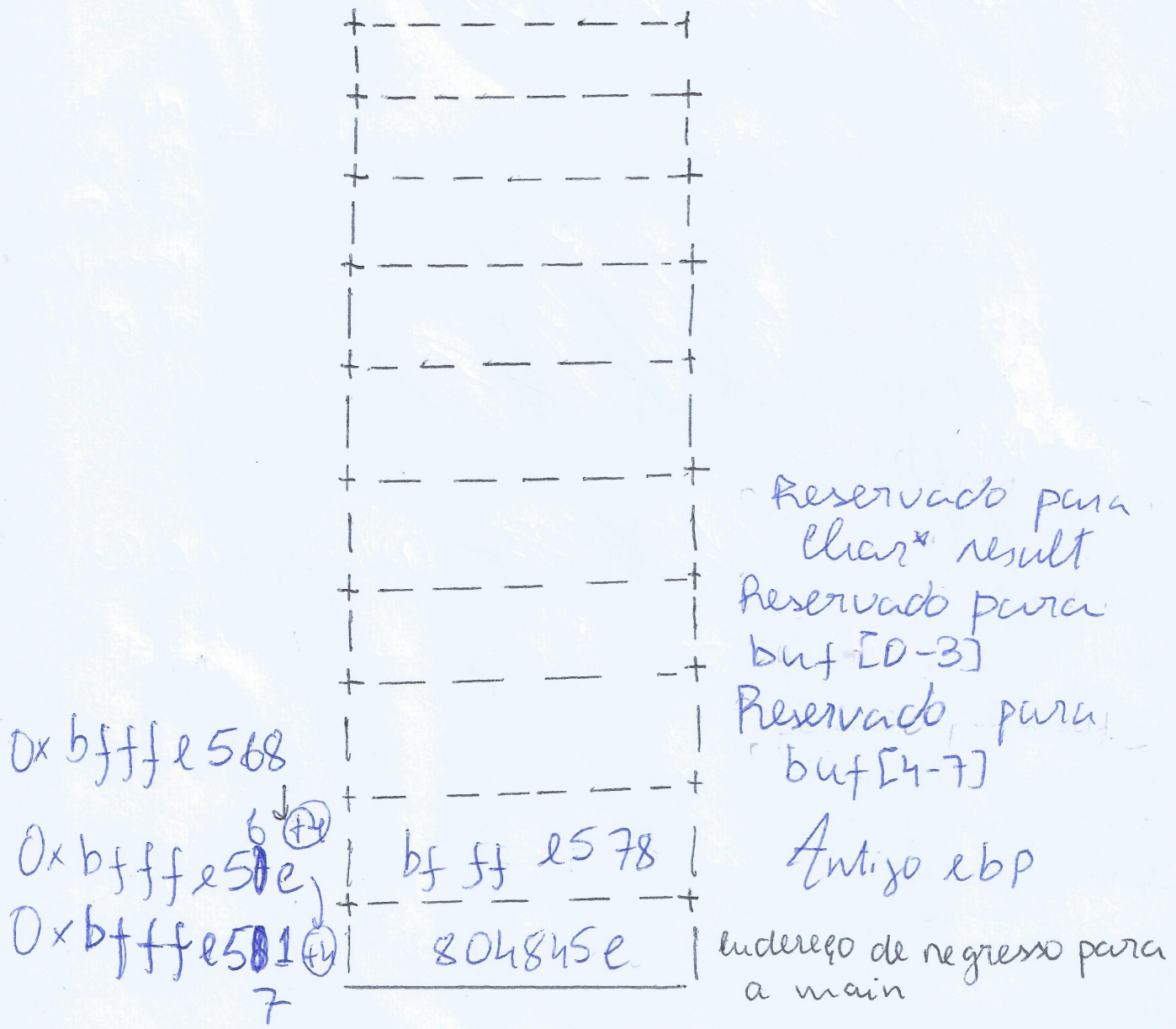
123456789012
(igual ao enunciado)

→ Output:

Program received signal SIGSIGV,
Segmentation fault.

0x08040032 in ??()

(4)



Comandos gdb que fui usando

```
(gdb) b main
   --- at 0x8048457
(gdb) b getline
   --- at 0x8048406
```

```
(gdb) print $ebp
$2: (void*) 0xbfffe568
(gdb) x $ebp
0xbfffe568: 0xbfffe578
```

(3)

(5) Indo no gdb: info registers → depois de um só com breakpoint no getline

%ebp → 0xbffffbu38

↳ 0xbffffd288

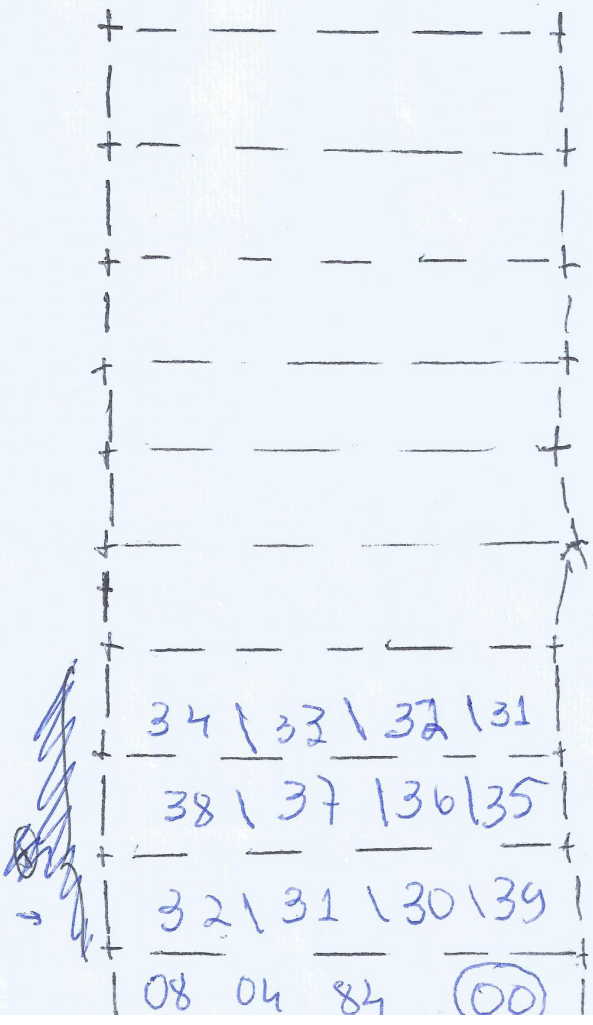
%esp → 0xbffffbu30

↳ 0xbffffd270

%eip → 0x8d0eee83

(gdb) x $eip

(6)



4

(7)

~~0~~, Registos corrompidos no regresso da função gets: %ebp e %eip

→ São alterados pela função gets, que guarda em ASCII os chars inseridos. Porém, não tendo endereço de regresso ou sendo este incorreto, a utilização de gets não corre bem.

Gets não tem limites a guardar a informação. Mas na stack temos definido um array de tamanho limitado