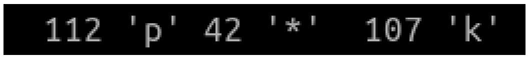
Burak Yesil
I pledge my honor that I have abided by the Stevens Honor System.

PART 1:

MY Stevens ID:
10468913

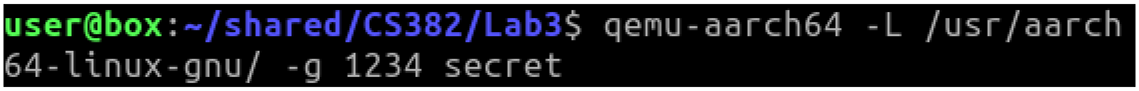
String I got (3 character bytes):
“p*k”

Comes from:



```
 112 'p' 42 '*'  107 'k'
```

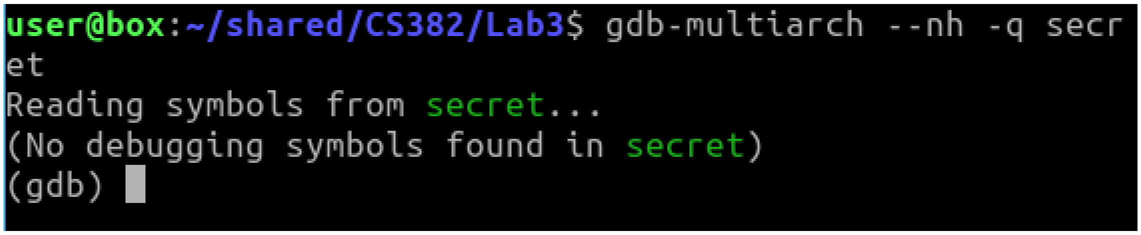
PART 2:



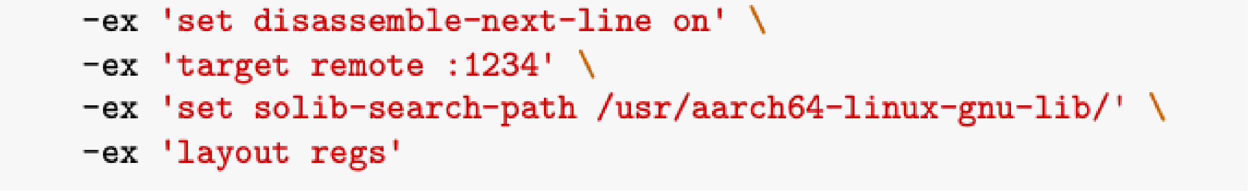
```
user@box:~/shared/CS382/Lab3$ qemu-aarch64 -L /usr/aarch
64-linux-gnu/ -g 1234 secret
```

I first setup my QEMU emulator using the command above

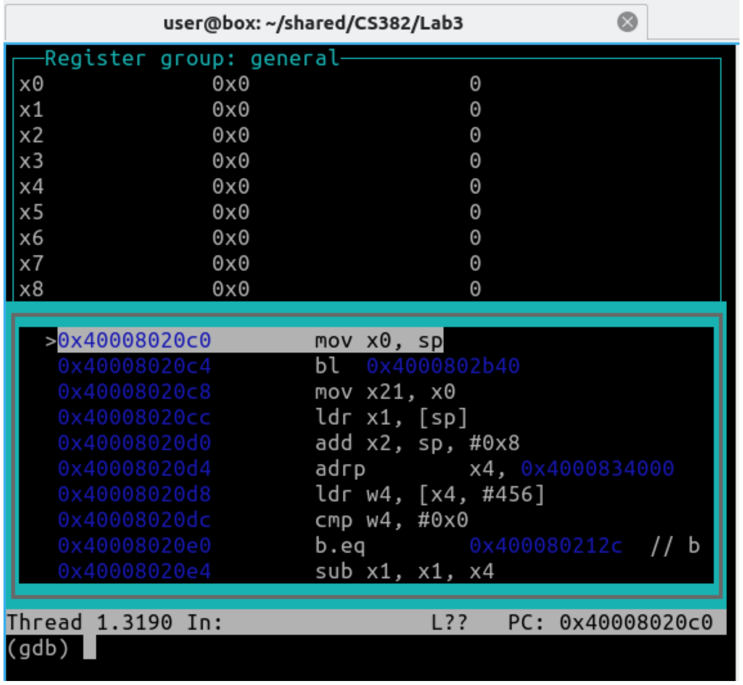
I then setup my actual debugger by using the following command and then inputted the values in the red quotes the get the layout mode shown in the next section.



```
user@box:~/shared/CS382/Lab3$ gdb-multiarch --nh -q secr
et
Reading symbols from secret...
(No debugging symbols found in secret)
(gdb) 
```

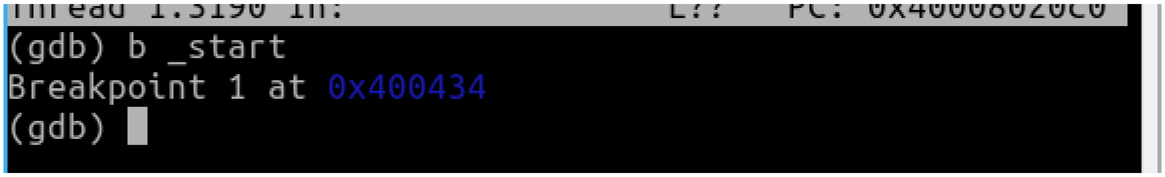


```
    -ex 'set disassemble-next-line on' \
    -ex 'target remote :1234' \
    -ex 'set solib-search-path /usr/aarch64-linux-gnu-lib/' \
    -ex 'layout regs'
```

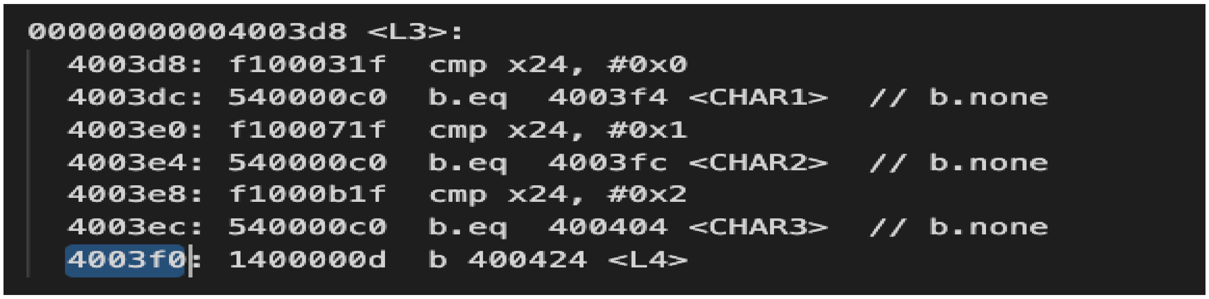


```
Register group: general
x0             0x0                 0
x1             0x0                 0
x2             0x0                 0
x3             0x0                 0
x4             0x0                 0
x5             0x0                 0
x6             0x0                 0
x7             0x0                 0
x8             0x0                 0

  >0x40008020c0          mov x0, sp
   0x40008020c4          bl  0x4000802b40
   0x40008020c8          mov x21, x0
   0x40008020cc          ldr x1, [sp]
   0x40008020d0          add x2, sp, #0x8
   0x40008020d4          adrp        x4, 0x4000834000
   0x40008020d8          ldr w4, [x4, #456]
   0x40008020dc          cmp w4, #0x0
   0x40008020e0          b.eq        0x400080212c  // b
   0x40008020e4          sub x1, x1, x4

Thread 1.3190 In:                       L??   PC: 0x40008020c0
(gdb) 
```

Once I got the layout mode to show up, I started by setting my breakpoints. I set my first break point at the _start label



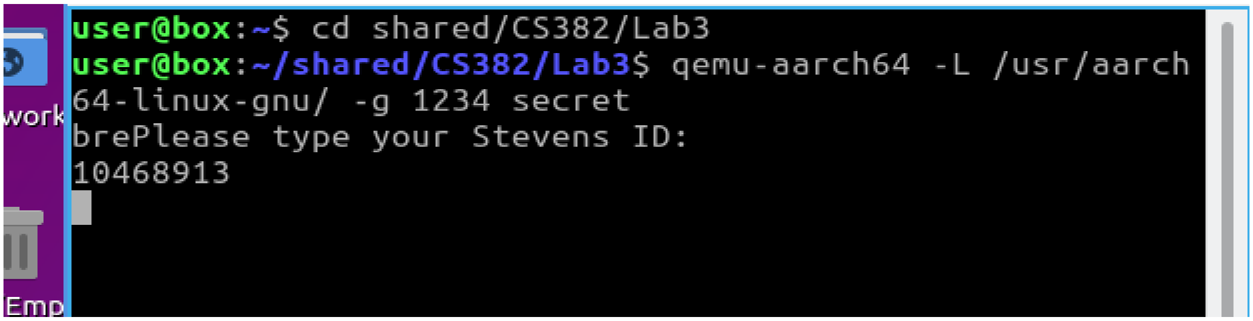
```
Thread 1.3190 In:                  L??   PC: 0x40008020c0
(gdb) b _start
Breakpoint 1 at 0x400434
(gdb) 
```



```
00000000004003d8 <L3>:
  4003d8: f100031f   cmp x24, #0x0
  4003dc: 540000c0   b.eq   4003f4 <CHAR1>   // b.none
  4003e0: f100071f   cmp x24, #0x1
  4003e4: 540000c0   b.eq   4003fc <CHAR2>   // b.none
  4003e8: f1000b1f   cmp x24, #0x2
  4003ec: 540000c0   b.eq   400404 <CHAR3>   // b.none
  4003f0: 1400000d   b 400424 <L4>
```

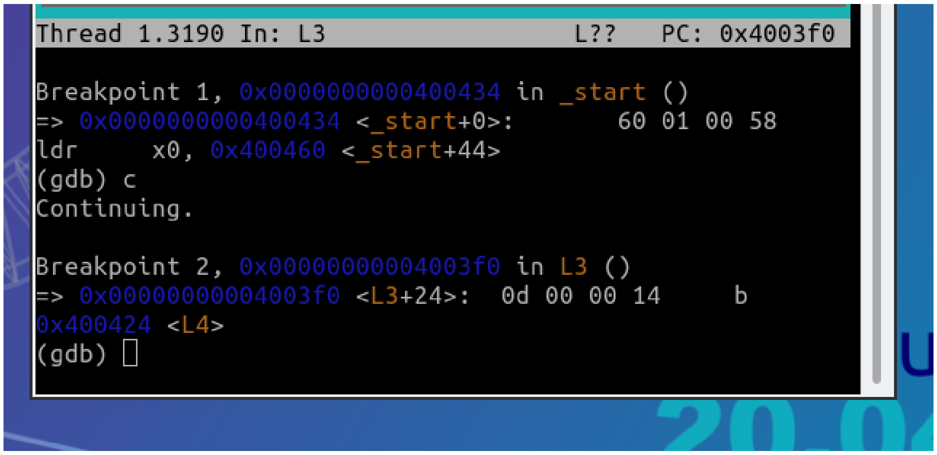
However, when I tried running with only one breakpoint my debugger didn’t work, so I put a second breakpoint at line 4003fo after all of the char procedures were called. I figured this out after writing the command in the instructions to load the .lst file and actual analyzing the file.

I then went and clicked continue, which gave me the following output in my first terminal window.



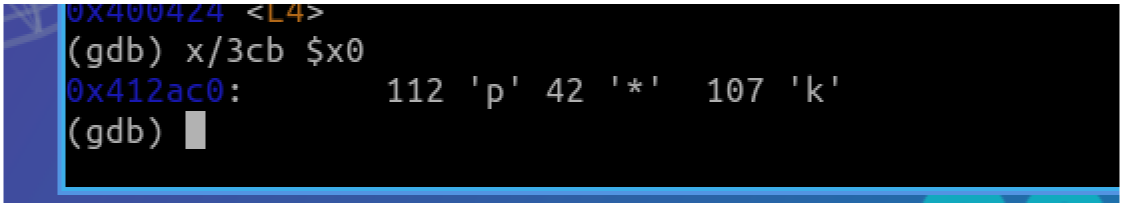
```
user@box:~$ cd shared/CS382/Lab3
user@box:~/shared/CS382/Lab3$ qemu-aarch64 -L /usr/aarch
64-linux-gnu/ -g 1234 secret
brePlease type your Stevens ID:
10468913
```

After successfully writing my Stevens ID, I got the Debugger to finish continuing.



```
Thread 1.3190 In: L3                        L??   PC: 0x4003f0

Breakpoint 1, 0x0000000000400434 in _start ()
=> 0x0000000000400434 <_start+0>:       60 01 00 58
ldr     x0, 0x400460 <_start+44>
(gdb) c
Continuing.

Breakpoint 2, 0x00000000004003f0 in L3 ()
=> 0x00000000004003f0 <L3+24>:  0d 00 00 14     b
0x400424 <L4>
(gdb) 
```

All that was left was to access the first three character bytes of the x0 register and by writing the command below, I got the following output. I picked x0 because when malloc is called to allocate space in the heap, it takes in a size input and returns the pointer to the memory location in the heap and stores the pointer to the address in the head in register x0.



```
0x400424 <L4>
(gdb) x/3cb $x0
0x412ac0:       112 'p' 42 '*'  107 'k'
(gdb) 
```