NAME: HARRIS SPAHIC

PLEDGE: I PLEDGE MY HONOR I HAVE ABIDED BY THE STEVENS HONOR SYSTEM.

SOLUTION: “H$/”

ID: 10460436

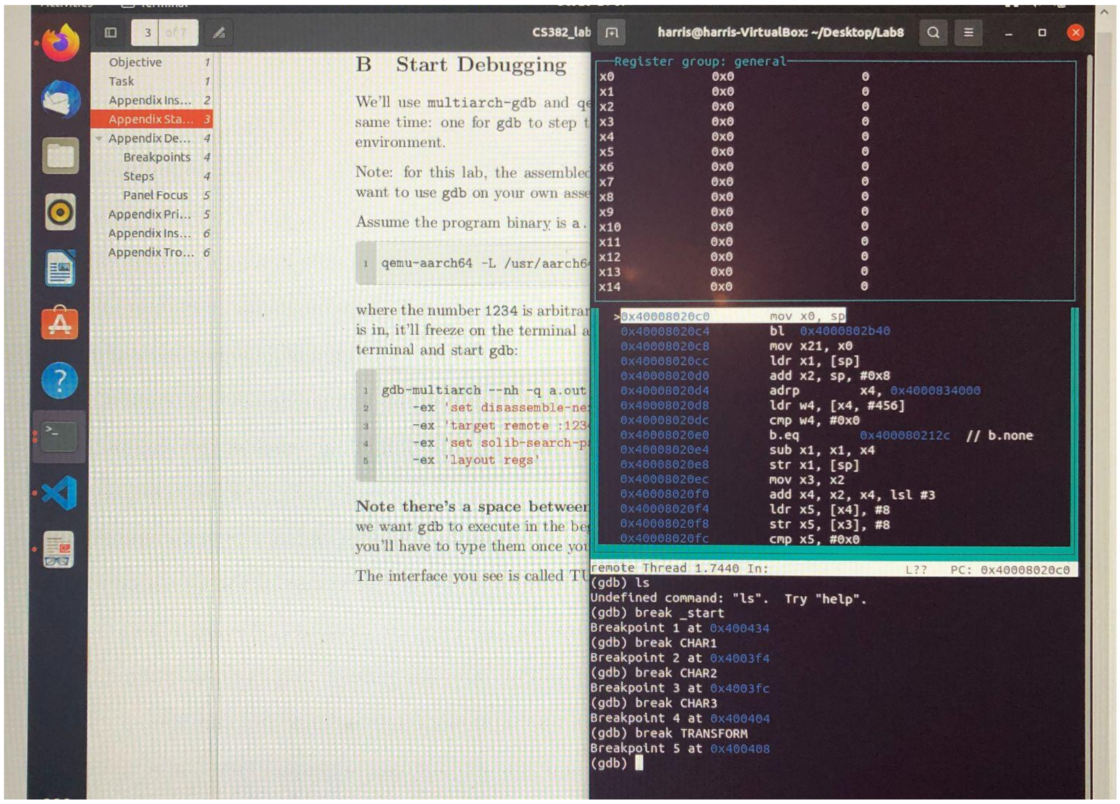
Explanation

The first thing we do is take a look at the code. We want to see if there are any labels of interest we might want to stop at using our GDB.

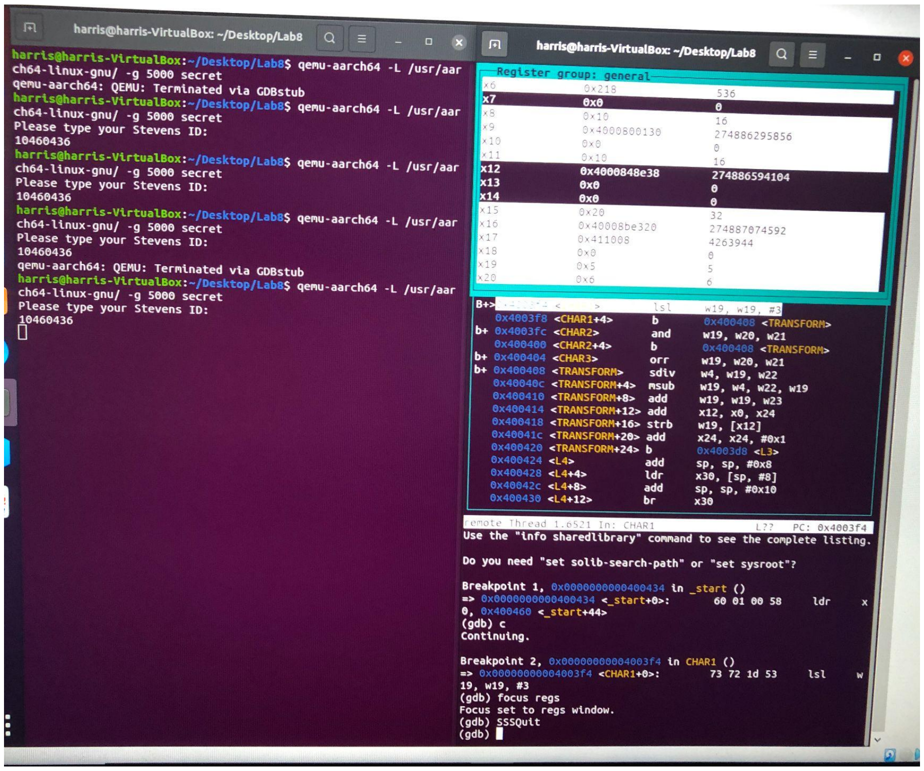
We take a look at the secret.lst that comes out of our cmd:
`aarch64-linux-gnu-objdump secret -D > secret.lst`

And find a few promising candidates.
Namely ~ CHAR1, CHAR2, CHAR3 & the branch they call TRANSFORM

We set breakpoints at each one of these branches. Also the _start branch, that starts our code.



```
┌─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             │
│x9             0x0                 0             │
│x10            0x0                 0             │
│x11            0x0                 0             │
│x12            0x0                 0             │
│x13            0x0                 0             │
│x14            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.none
   0x40008020e4         sub x1, x1, x4
   0x40008020e8         str x1, [sp]
   0x40008020ec         mov x3, x2
   0x40008020f0         add x4, x2, x4, lsl #3
   0x40008020f4         ldr x5, [x4], #8
   0x40008020f8         str x5, [x3], #8
   0x40008020fc         cmp x5, #0x0
remote Thread 1.7440 In:                         L??   PC: 0x40008020c0
(gdb) ls
Undefined command: "ls".  Try "help".
(gdb) break _start
Breakpoint 1 at 0x400434
(gdb) break CHAR1
Breakpoint 2 at 0x4003f4
(gdb) break CHAR2
Breakpoint 3 at 0x4003fc
(gdb) break CHAR3
Breakpoint 4 at 0x400404
(gdb) break TRANSFORM
Breakpoint 5 at 0x400408
(gdb) █
```



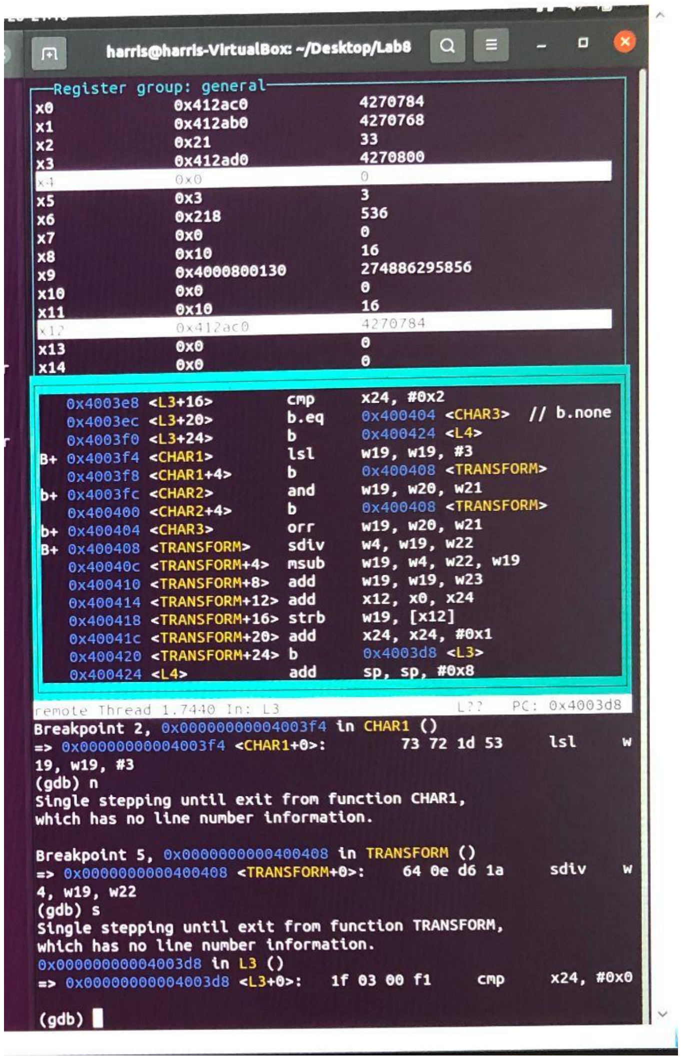
```
harris@harris-VirtualBox:~/Desktop/Lab8$ qemu-aarch64 -L /usr/aar
ch64-linux-gnu/ -g 5000 secret
qemu-aarch64: QEMU: Terminated via GDBstub
harris@harris-VirtualBox:~/Desktop/Lab8$ qemu-aarch64 -L /usr/aar
ch64-linux-gnu/ -g 5000 secret
Please type your Stevens ID:
10460436
harris@harris-VirtualBox:~/Desktop/Lab8$ qemu-aarch64 -L /usr/aar
ch64-linux-gnu/ -g 5000 secret
Please type your Stevens ID:
10460436
harris@harris-VirtualBox:~/Desktop/Lab8$ qemu-aarch64 -L /usr/aar
ch64-linux-gnu/ -g 5000 secret
Please type your Stevens ID:
10460436
qemu-aarch64: QEMU: Terminated via GDBstub
harris@harris-VirtualBox:~/Desktop/Lab8$ qemu-aarch64 -L /usr/aar
ch64-linux-gnu/ -g 5000 secret
Please type your Stevens ID:
10460436
```

```
Register group: general
x6             0x218               536
x7             0x0                 0
x8             0x10                16
x9             0x4000800130        274886295856
x10            0x0                 0
x11            0x10                16
x12            0x4000848e38        274886594104
x13            0x0                 0
x14            0x0                 0
x15            0x20                32
x16            0x40008be320        274887074592
x17            0x411008            4263944
x18            0x0                 0
x19            0x5                 5
x20            0x6                 6

B+> 0x4003f4 <CHAR1>          lsl    w19, w19, #3
    0x4003f8 <CHAR1+4>        b      0x400408 <TRANSFORM>
b+  0x4003fc <CHAR2>          and    w19, w20, w21
    0x400400 <CHAR2+4>        b      0x400408 <TRANSFORM>
b+  0x400404 <CHAR3>          orr    w19, w20, w21
b+  0x400408 <TRANSFORM>      sdiv   w4, w19, w22
    0x40040c <TRANSFORM+4>    msub   w19, w4, w22, w19
    0x400410 <TRANSFORM+8>    add    w19, w19, w23
    0x400414 <TRANSFORM+12>   add    x12, x0, x24
    0x400418 <TRANSFORM+16>   strb   w19, [x12]
    0x40041c <TRANSFORM+20>   add    x24, x24, #0x1
    0x400420 <TRANSFORM+24>   b      0x4003d8 <L3>
    0x400424 <L4>             add    sp, sp, #0x8
    0x400428 <L4+4>           ldr    x30, [sp, #8]
    0x40042c <L4+8>           add    sp, sp, #0x10
    0x400430 <L4+12>          br     x30
remote Thread 1.6521 In: CHAR1                         L??   PC: 0x4003f4
Use the "info sharedlibrary" command to see the complete listing.

Do you need "set solib-search-path" or "set sysroot"?

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

Breakpoint 2, 0x00000000004003f4 in CHAR1 ()
=> 0x00000000004003f4 <CHAR1+0>:        73 72 1d 53     lsl     w
19, w19, #3
(gdb) focus regs
Focus set to regs window.
(gdb) SSSQuit
(gdb) 
```

At this point we continue past the start state & input our ID number in the hosting terminal. Now the meat of the code has started and we get into our first call to CHAR1.

We can see in our instructions terminal, that <CHAR 1> first lsl W19 by #3. Then branches to <TRANSFORM>. We'll keep an eye on X19.

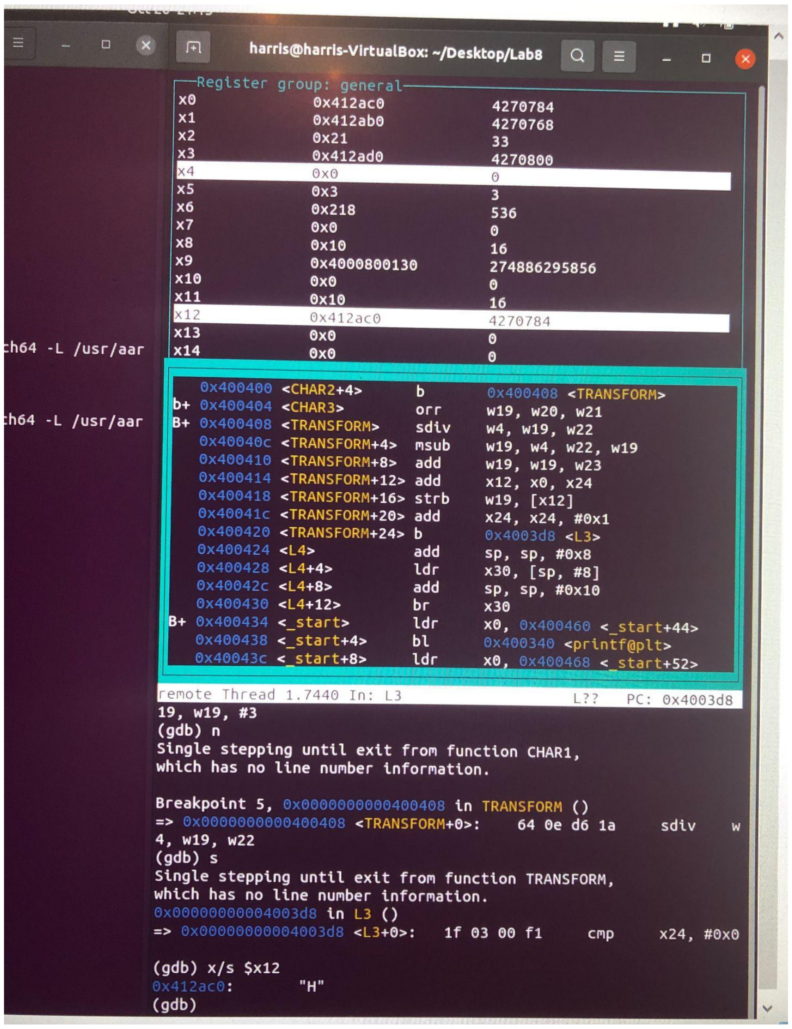
We then continue into the <TRANSFORM> branch.



```
harris@harris-VirtualBox: ~/Desktop/Lab8
┌─Register group: general─────────────────────────────────────┐
│x0             0x412ac0              4270784                 │
│x1             0x412ab0              4270768                 │
│x2             0x21                  33                      │
│x3             0x412ad0              4270800                 │
│x4             0x0                   0                       │
│x5             0x3                   3                       │
│x6             0x218                 536                     │
│x7             0x0                   0                       │
│x8             0x10                  16                      │
│x9             0x4000800130          274886295856            │
│x10            0x0                   0                       │
│x11            0x10                  16                      │
│x12            0x412ac0              4270784                 │
│x13            0x0                   0                       │
│x14            0x0                   0                       │

    0x4003e8 <L3+16>          cmp    x24, #0x2
    0x4003ec <L3+20>          b.eq   0x400404 <CHAR3>  // b.none
    0x4003f0 <L3+24>          b      0x400424 <L4>
B+  0x4003f4 <CHAR1>          lsl    w19, w19, #3
    0x4003f8 <CHAR1+4>        b      0x400408 <TRANSFORM>
b+  0x4003fc <CHAR2>          and    w19, w20, w21
    0x400400 <CHAR2+4>        b      0x400408 <TRANSFORM>
b+  0x400404 <CHAR3>          orr    w19, w20, w21
B+  0x400408 <TRANSFORM>      sdiv   w4, w19, w22
    0x40040c <TRANSFORM+4>    msub   w19, w4, w22, w19
    0x400410 <TRANSFORM+8>    add    w19, w19, w23
    0x400414 <TRANSFORM+12>   add    x12, x0, x24
    0x400418 <TRANSFORM+16>   strb   w19, [x12]
    0x40041c <TRANSFORM+20>   add    x24, x24, #0x1
    0x400420 <TRANSFORM+24>   b      0x4003d8 <L3>
    0x400424 <L4>             add    sp, sp, #0x8

remote Thread 1.7440 In: L3                          L??   PC: 0x4003d8
Breakpoint 2, 0x00000000004003f4 in CHAR1 ()
=> 0x00000000004003f4 <CHAR1+0>:        73 72 1d 53     lsl     w
19, w19, #3
(gdb) n
Single stepping until exit from function CHAR1,
which has no line number information.

Breakpoint 5, 0x0000000000400408 in TRANSFORM ()
=> 0x0000000000400408 <TRANSFORM+0>:    64 0e d6 1a     sdiv    w
4, w19, w22
(gdb) s
Single stepping until exit from function TRANSFORM,
which has no line number information.
0x00000000004003d8 in L3 ()
=> 0x00000000004003d8 <L3+0>:   1f 03 00 f1     cmp     x24, #0x0

(gdb) 
```

We see that the <TRANSFORM> function takes some W19 and does something with it, eventually storing it in the address of X12. What is X12? The line before has X12 equal to X0 & X24. X0 is usually the return address of any function, but we use **"focus regs"** (I didn't take a picture of the shift, please forgive me. I really don't want to redo the whole process just for the registers.) to shift the registers to show x0 & x12. **Notice they are the same random value.** Hence X24 is 0.

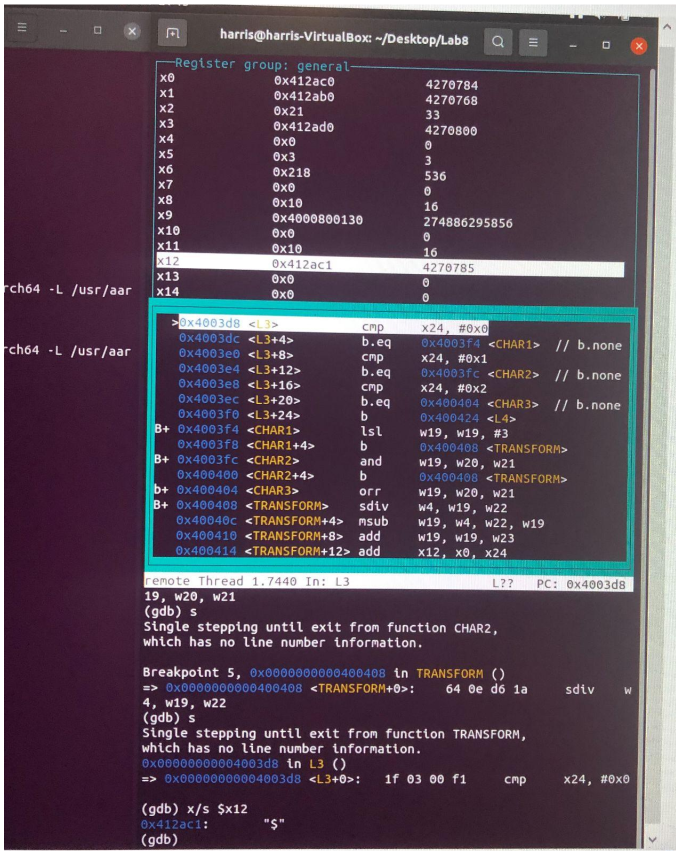
We continue stepping in until after the transformed value of X19 is stored into X12. Then check the contents of X12 to see what's inside the memory address.



```
harris@harris-VirtualBox: ~/Desktop/Lab8

┌─Register group: general─────────────────────────────────────────┐
│x0             0x412ac0                4270784                   │
│x1             0x412ab0                4270768                   │
│x2             0x21                    33                        │
│x3             0x412ad0                4270800                   │
│x4             0x0                     0                         │
│x5             0x3                     3                         │
│x6             0x218                   536                       │
│x7             0x0                     0                         │
│x8             0x10                    16                        │
│x9             0x4000800130            274886295856              │
│x10            0x0                     0                         │
│x11            0x10                    16                        │
│x12            0x412ac0                4270784                   │
│x13            0x0                     0                         │
│x14            0x0                     0                         │

    0x400400 <CHAR2+4>       b      0x400408 <TRANSFORM>
 b+ 0x400404 <CHAR3>         orr    w19, w20, w21
 B+ 0x400408 <TRANSFORM>     sdiv   w4, w19, w22
    0x40040c <TRANSFORM+4>   msub   w19, w4, w22, w19
    0x400410 <TRANSFORM+8>   add    w19, w19, w23
    0x400414 <TRANSFORM+12>  add    x12, x0, x24
    0x400418 <TRANSFORM+16>  strb   w19, [x12]
    0x40041c <TRANSFORM+20>  add    x24, x24, #0x1
    0x400420 <TRANSFORM+24>  b      0x4003d8 <L3>
    0x400424 <L4>            add    sp, sp, #0x8
    0x400428 <L4+4>          ldr    x30, [sp, #8]
    0x40042c <L4+8>          add    sp, sp, #0x10
    0x400430 <L4+12>         br     x30
 B+ 0x400434 <_start>        ldr    x0, 0x400460 <_start+44>
    0x400438 <_start+4>      bl     0x400340 <printf@plt>
    0x40043c <_start+8>      ldr    x0, 0x400468 <_start+52>

remote Thread 1.7440 In: L3                          L??   PC: 0x4003d8
19, w19, #3
(gdb) n
Single stepping until exit from function CHAR1,
which has no line number information.

Breakpoint 5, 0x0000000000400408 in TRANSFORM ()
=> 0x0000000000400408 <TRANSFORM+0>:    64 0e d6 1a     sdiv    w
4, w19, w22
(gdb) s
Single stepping until exit from function TRANSFORM,
which has no line number information.
0x00000000004003d8 in L3 ()
=> 0x00000000004003d8 <L3+0>:   1f 03 00 f1     cmp     x24, #0x0

(gdb) x/s $x12
0x412ac0:       "H"
(gdb)
```

Using the x/s @X12 command we find that the address of X12 holds the character "H". Hey that's what we're looking for!

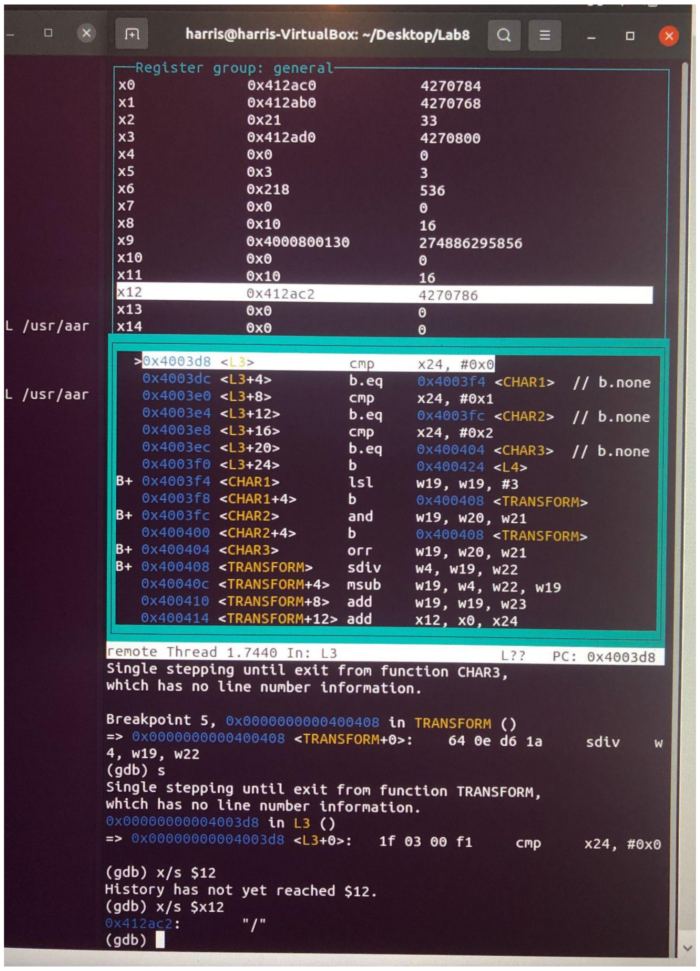
We continue the same process, checking the newly stored contents of X12 each time we finish our transform call for each character. Since we added breaks this is easy to do. We just go to the "**next**" break call. And then **"step"** to where we want to be.



```
rch64 -L /usr/aar
rch64 -L /usr/aar
```

```
-Register group: general-
x0             0x412ac0            4270784
x1             0x412ab0            4270768
x2             0x21                33
x3             0x412ad0            4270800
x4             0x0                 0
x5             0x3                 3
x6             0x218               536
x7             0x0                 0
x8             0x10                16
x9             0x4000800130        274886295856
x10            0x0                 0
x11            0x10                16
x12            0x412ac1            4270785
x13            0x0                 0
x14            0x0                 0

  >0x4003d8 <L3>             cmp     x24, #0x0
   0x4003dc <L3+4>           b.eq    0x4003f4 <CHAR1>  // b.none
   0x4003e0 <L3+8>           cmp     x24, #0x1
   0x4003e4 <L3+12>          b.eq    0x4003fc <CHAR2>  // b.none
   0x4003e8 <L3+16>          cmp     x24, #0x2
   0x4003ec <L3+20>          b.eq    0x400404 <CHAR3>  // b.none
   0x4003f0 <L3+24>          b       0x400424 <L4>
B+ 0x4003f4 <CHAR1>          lsl     w19, w19, #3
   0x4003f8 <CHAR1+4>        b       0x400408 <TRANSFORM>
B+ 0x4003fc <CHAR2>          and     w19, w20, w21
   0x400400 <CHAR2+4>        b       0x400408 <TRANSFORM>
b+ 0x400404 <CHAR3>          orr     w19, w20, w21
B+ 0x400408 <TRANSFORM>      sdiv    w4, w19, w22
   0x40040c <TRANSFORM+4>    msub    w19, w4, w22, w19
   0x400410 <TRANSFORM+8>    add     w19, w19, w23
   0x400414 <TRANSFORM+12>   add     x12, x0, x24
remote Thread 1.7440 In: L3                          L??   PC: 0x4003d8
19, w20, w21
(gdb) s
Single stepping until exit from function CHAR2,
which has no line number information.

Breakpoint 5, 0x0000000000400408 in TRANSFORM ()
=> 0x0000000000400408 <TRANSFORM+0>:    64 0e d6 1a     sdiv    w
4, w19, w22
(gdb) s
Single stepping until exit from function TRANSFORM,
which has no line number information.
0x00000000004003d8 in L3 ()
=> 0x00000000004003d8 <L3+0>:   1f 03 00 f1     cmp     x24, #0x0

(gdb) x/s $x12
0x412ac1:       "$"
(gdb)
```

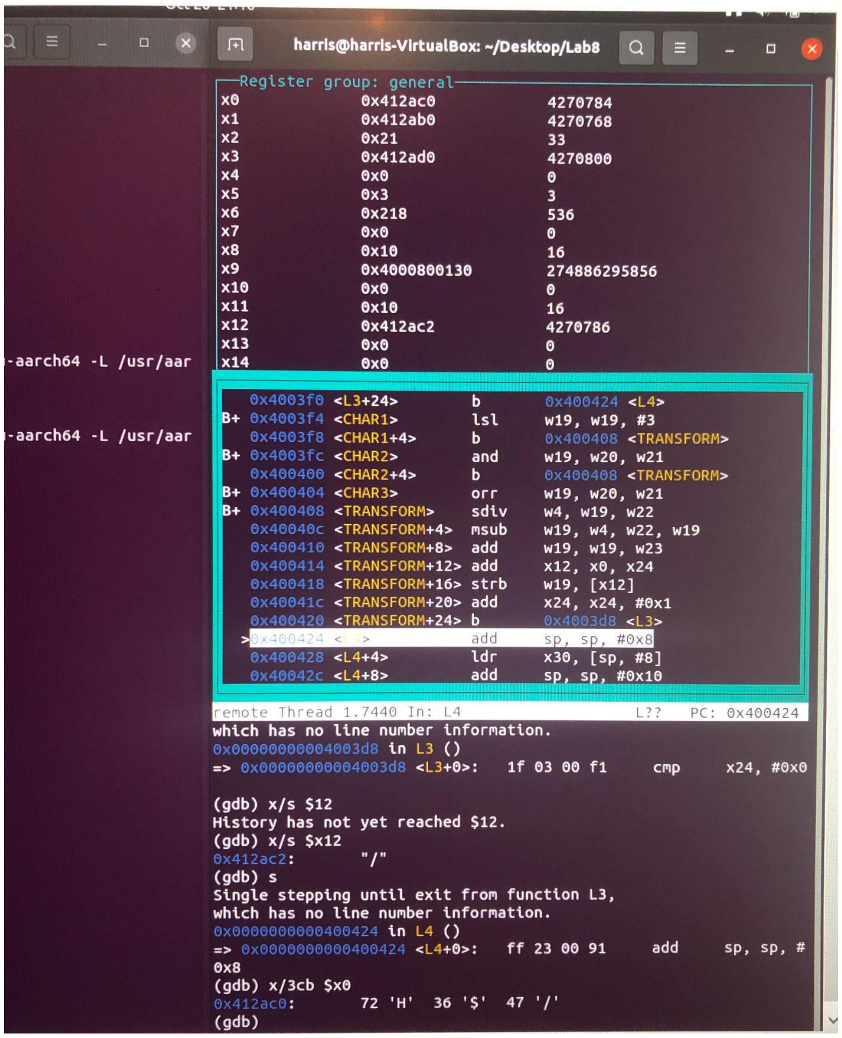
This is what we get checking X12 at the end of the second TRANSFORM call.


```
harris@harris-VirtualBox: ~/Desktop/Lab8

┌─Register group: general─────────────────────────────────────
│x0             0x412ac0             4270784
│x1             0x412ab0             4270768
│x2             0x21                 33
│x3             0x412ad0             4270800
│x4             0x0                  0
│x5             0x3                  3
│x6             0x218                536
│x7             0x0                  0
│x8             0x10                 16
│x9             0x4000800130         274886295856
│x10            0x0                  0
│x11            0x10                 16
│x12            0x412ac2             4270786
│x13            0x0                  0
│x14            0x0                  0

  >0x4003d8 <L3>             cmp    x24, #0x0
   0x4003dc <L3+4>           b.eq   0x4003f4 <CHAR1>  // b.none
   0x4003e0 <L3+8>           cmp    x24, #0x1
   0x4003e4 <L3+12>          b.eq   0x4003fc <CHAR2>  // b.none
   0x4003e8 <L3+16>          cmp    x24, #0x2
   0x4003ec <L3+20>          b.eq   0x400404 <CHAR3>  // b.none
   0x4003f0 <L3+24>          b      0x400424 <L4>
B+ 0x4003f4 <CHAR1>          lsl    w19, w19, #3
   0x4003f8 <CHAR1+4>        b      0x400408 <TRANSFORM>
B+ 0x4003fc <CHAR2>          and    w19, w20, w21
   0x400400 <CHAR2+4>        b      0x400408 <TRANSFORM>
B+ 0x400404 <CHAR3>          orr    w19, w20, w21
B+ 0x400408 <TRANSFORM>      sdiv   w4, w19, w22
   0x40040c <TRANSFORM+4>    msub   w19, w4, w22, w19
   0x400410 <TRANSFORM+8>    add    w19, w19, w23
   0x400414 <TRANSFORM+12>   add    x12, x0, x24

remote Thread 1.7440 In: L3                          L??   PC: 0x4003d8
Single stepping until exit from function CHAR3,
which has no line number information.

Breakpoint 5, 0x0000000000400408 in TRANSFORM ()
=> 0x0000000000400408 <TRANSFORM+0>:     64 0e d6 1a      sdiv    w
4, w19, w22
(gdb) s
Single stepping until exit from function TRANSFORM,
which has no line number information.
0x00000000004003d8 in L3 ()
=> 0x00000000004003d8 <L3+0>:   1f 03 00 f1      cmp      x24, #0x0

(gdb) x/s $12
History has not yet reached $12.
(gdb) x/s $x12
0x412ac2:       "/"
(gdb)
```

And this is the third TRANSFORM X12 value.

So our solution is H$/.

Just to make sure we check X0 our “assumed” return address, and see if we get the same solution. We check the 3 character bytes at the address of X0 using **x/3cb $X0**.



```
harris@harris-VirtualBox: ~/Desktop/Lab8

-aarch64 -L /usr/aar
-aarch64 -L /usr/aar

┌─Register group: general──────────────────────────────────
│x0             0x412ac0            4270784
│x1             0x412ab0            4270768
│x2             0x21                33
│x3             0x412ad0            4270800
│x4             0x0                 0
│x5             0x3                 3
│x6             0x218               536
│x7             0x0                 0
│x8             0x10                16
│x9             0x4000800130        274886295856
│x10            0x0                 0
│x11            0x10                16
│x12            0x412ac2            4270786
│x13            0x0                 0
│x14            0x0                 0

   0x4003f0 <L3+24>         b       0x400424 <L4>
B+ 0x4003f4 <CHAR1>         lsl     w19, w19, #3
   0x4003f8 <CHAR1+4>       b       0x400408 <TRANSFORM>
B+ 0x4003fc <CHAR2>         and     w19, w20, w21
   0x400400 <CHAR2+4>       b       0x400408 <TRANSFORM>
B+ 0x400404 <CHAR3>         orr     w19, w20, w21
B+ 0x400408 <TRANSFORM>     sdiv    w4, w19, w22
   0x40040c <TRANSFORM+4>   msub    w19, w4, w22, w19
   0x400410 <TRANSFORM+8>   add     w19, w19, w23
   0x400414 <TRANSFORM+12>  add     x12, x0, x24
   0x400418 <TRANSFORM+16>  strb    w19, [x12]
   0x40041c <TRANSFORM+20>  add     x24, x24, #0x1
   0x400420 <TRANSFORM+24>  b       0x4003d8 <L3>
  >0x400424 <L4>            add     sp, sp, #0x8
   0x400428 <L4+4>          ldr     x30, [sp, #8]
   0x40042c <L4+8>          add     sp, sp, #0x10

remote Thread 1.7440 In: L4                          L??   PC: 0x400424
which has no line number information.
0x00000000004003d8 in L3 ()
=> 0x00000000004003d8 <L3+0>:   1f 03 00 f1     cmp     x24, #0x0

(gdb) x/s $12
History has not yet reached $12.
(gdb) x/s $x12
0x412ac2:       "/"
(gdb) s
Single stepping until exit from function L3,
which has no line number information.
0x0000000000400424 in L4 ()
=> 0x0000000000400424 <L4+0>:   ff 23 00 91     add     sp, sp, #
0x8
(gdb) x/3cb $x0
0x412ac0:       72 'H'  36 '$'  47 '/'
(gdb)
```

We get the same result! Hence our solution is correct (unless this was all a front and the real characters are hidden somewhere else).