15-213 Summer 2018 Lecture 20*: Malloc Lab GDB review

- Login to a shark machine
- wget http://www.cs.cmu.edu/~213/activities/recML.tar
- tar xf recML.tar
- cd recML
- make

**1. Activity 1 --- debugging with gdb**

$ gdb --args ./mdriver -c traces/syn-mix-short.rep

(gdb) run

(gdb) backtrace

(gdb) list

// to inspect block content

(gdb) x /10gx block

// how do we print out the block content? What do we cast it to?

(gdb) print (block_t)*[address]

// to go up a frame for the inspection of call stack

(gdb) frame 1

**2. Activity 2 --- debugging with gdb continued**

$ gdb --args ./mdriver-2 -c traces/syn-array-short.rep

(gdb) run

//How can we know when this inconsistency first occurred, and therefore figure out why?

(gdb) watch *[header address] // where is header address?

(gdb) watch *[footer address]

**Download the (second) handout.**

**$ wget http://www.cs.cmu.edu/~213/activities/recMLb.tar**
**$ tar xf recMLb.tar**
**$ cd recMLb**
**$ make**

**3. Run mdriver using GDB.**

**$ gdb --args ./mdriver -c ./traces/syn-array-short.rep -D**

**…**

**(gdb) run**

**You should see "garbled bytes" errors:**

**…**

```
Throughput targets: min=6528, max=11750, benchmark=13056

Malloc size 9904 on address 0x800000010.
Malloc size 50084 on address 0x8000026d0.
ERROR [trace ././traces/syn-array-short.rep, line 7]: block 0 has 8
garbled bytes, starting at byte 0
```

**…**

```
Terminated with 14 errors
[Inferior 1 (process 30988) exited normally]
```

**4. Set a watchpoint on the first garbled address.**

**(gdb) watch *0x800000010**
**(gdb) run**

**… a few continues …**

```
Hardware watchpoint 1: *0x800000010

Old value = -7350814
New value = 9928
mm_malloc (size=50084) at mm.c:276
276 dbg_printf("Malloc size %zd on address %p.\n", size, bp);
```

**(gdb) c**

```
Continuing.
Malloc size 50084 on address 0x8000026d0.
ERROR [trace ././traces/syn-array-short.rep, line 7]: block 0 has 8
garbled bytes, starting at byte 0
```

**5. What happened?**

**Run mdriver-2 using GDB.**

**$ gdb --args ./mdriver-2 -c traces/syn-array-short.rep**

**...**

**(gdb) run**

**You should see this error:**

```
Malloc size 9904 on address 0x8000036d0
ERROR [trace ./traces/syn-array-short.rep, line 5]: Payload
(0x8000036d0:0x800005d7f) lies outside heap (0x800000000:0x8000036cf)
```

**2. Set a watchpoint on the header of the payload.**

**(gdb) watch *0x8000036c8**

**(gdb) run**

**...**

```
Hardware watchpoint 1: *0x8000036c8
Old value = 1
New value = 9921
write_header(block=0x8000036c8, size=9920, alloc=true) at mm-2.c:573
```

**3. Backtrace to see what function called write_header.**

**(gdb) bt**

```
#0 write_header (block=0x8000036c8, size=9920, alloc=true) at mm-
2.c:573
#1 0x0000000000407d93 in place (block=0x8000036c8, asize=9920) at mm-
2.c:458
```

**...**

**4. The writes occurred in place. Is place implemented incorrectly, or was it given a bad argument?**

## 5. GDB Appendix

- backtrace: Shows the call stack

  // “bt” for short

- list: Shows source code
- print <expression>

  // “p” for short, you can practically print anything, whatever you can print in your c file
  // p[rint] *<name>: print what is pointed to by <name>
  // p/x <name>: print value of <name> in hex format

- watch <expression>

  // typically an address where a var that we care about is stored

- break <function / line>

  // “b” for short

- break <function / line> if <expression>

  // only stops execution when the expression evaluates to true
  // dis[able] 1: disable breakpoint 1
  // en[able] 1: enable breakpoint 1
  // d[elete] 1: delete breakpoint 1
  // cond 1: make breakpoint 1 unconditional