

Introduction to Computer System*

Bug Report L^AT_EX

* Teacher: Feng Su, Liang Wang. TA: Tianheng Wu

1st 张逸凯 171840708

Department of Computer Science and Technology

Nanjing University

zykhelloha@gmail.com

I. 问题(BUG)

`__ref_lgdt`, 没有装载 `cpu.gdtr.base`, 也就是 `libs/nemu-ref` 下的 `lib-nemu-ref.a` 里之前生成的 `lgdt.o` 有问题.

问题的发现过程: 不加 `--autorun` 用 `si` 看每条指令执行后 `cpu.gdtr.base` 情况, 如下图.

```
uint32_t eval(int s, int e, bool *success) {  
    return cpu.gdtr.base;  
}
```

图 1. 让 2-3 中的 `p sth` 指令输出的都是 `cpu.gdtr.base`

主要问题:

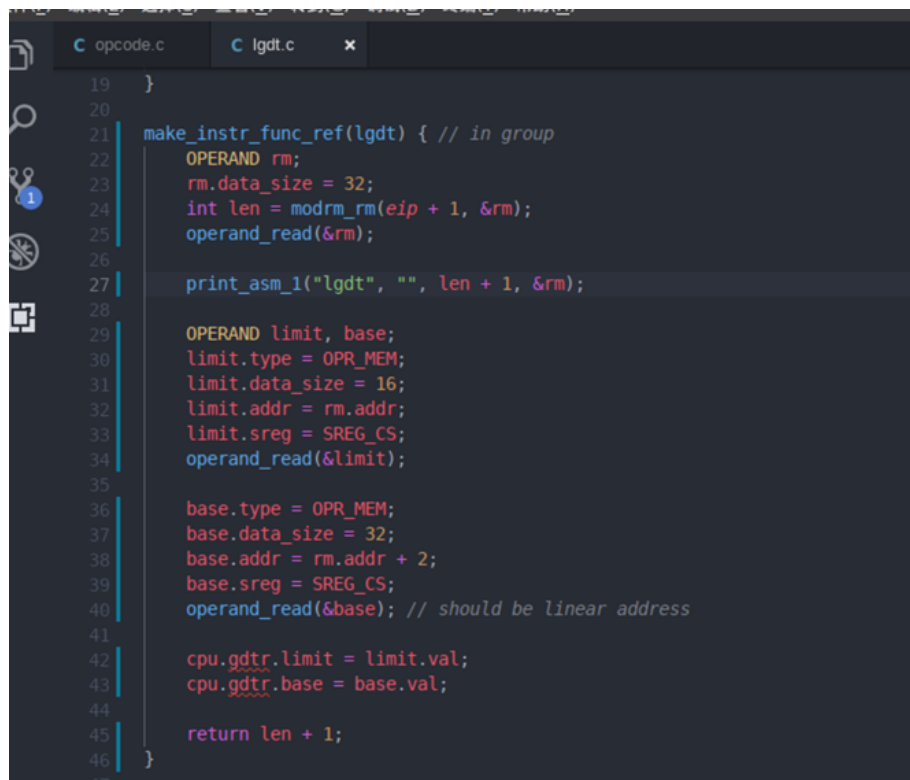
- `nemu/src/cpu/instr/lgdt.c` 为空, `opcode.c` 里使用 `__ref_lgdt` 时结果:

```
zhangyikai@debian:~/4_1_pa2018_fall$ ./nemu/nemu --testcase mov-c --kernel  
NEMU load and execute img: ./kernel/kernel.img elf: ./testcase/bin/mov-c  
(nemu) si  
00030000: fa cli  
(nemu) p 0  
0  
(nemu) si  
00030001: 0f 01 15 4c 00 03 lgdt 0x3004c  
(nemu) p 0  
0  
(nemu) si  
00030008: 0f 20 mov %cr0, %eax  
(nemu) p 0  
0  
(nemu)
```

图 2. 可以看到在 `lgdt` 执行之后 `cpu.gdtr.base` 未被装载

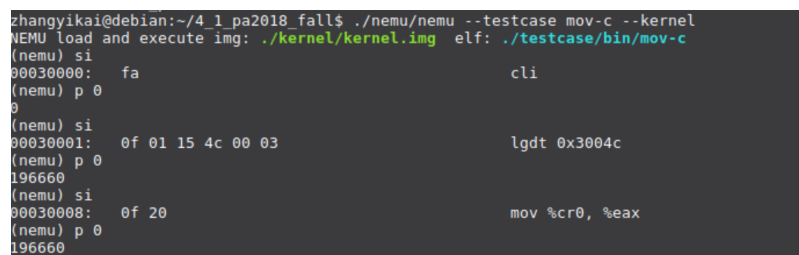
*为更好的 `nemu` 而努力.

- 当nemu/src/cpu/instr/lgdt.c为汪老师发给我的源码, 如下图:



```
19 }
20
21 make_instr_func_ref(lgdt) { // in group
22     OPERAND rm;
23     rm.data_size = 32;
24     int len = modrm_rm(eip + 1, &rm);
25     operand_read(&rm);
26
27     print_asm_1("lgdt", "", len + 1, &rm);
28
29     OPERAND limit, base;
30     limit.type = OPR_MEM;
31     limit.data_size = 16;
32     limit.addr = rm.addr;
33     limit.sreg = SREG_CS;
34     operand_read(&limit);
35
36     base.type = OPR_MEM;
37     base.data_size = 32;
38     base.addr = rm.addr + 2;
39     base.sreg = SREG_CS;
40     operand_read(&base); // should be linear address
41
42     cpu.gdtr.limit = limit.val;
43     cpu.gdtr.base = base.val;
44
45     return len + 1;
46 }
```

在opcode.c里使用__ref_lgdt时结果:



```
zhangyikai@debian:~/4_1_pa2018_fall$ ./nemu/nemu --testcase mov-c --kernel
NEMU load and execute img: ./kernel/kernel.img elf: ./testcase/bin/mov-c
(nemu) si
00030000: fa cli
(nemu) p 0
0
(nemu) si
00030001: 0f 01 15 4c 00 03 lgdt 0x3004c
(nemu) p 0
196660
(nemu) si
00030008: 0f 20 mov %cr0, %eax
(nemu) p 0
196660
```

图 3. 可以看到在lgdt执行之后cpu.gdtr.base正常装载

而且在当前情况下程序经过lgdt.c里面的make_instr_func_ref(lgdt)(已assert测试)

这很明显是__ref_lgdt的内置文件(.o静态库.a文件)里面的lgdt出现了问题, 接下来我将更精确地证明它的错误

II. 精确证明BUG存在

首先我们objdump -d对比libs/nemu-ref/lib-nemu-ref.a里的lgdt.o和make test_pa-4-1之后在nemu/src/cpu/instr下的lgdt.o文件。

这是可对比的, 虽然我们不知道__ref_.o文件如何生成, 但是nemu/src/cpu/instr下的lgdt.o文件指令为:

```
gcc -ggdb3 -MMD -MP -Wall -Werror -O2 -I./include -I../include -I../libs -I../libs/nemu-ref/include -c -o src/cpu/instr/lgdt.o src/cpu/instr/lgdt.c
```

对比结果如下:

```
1 00000000 <__ref_lgdt>:
2 0: 57          push    %edi
3 1: 56          push    %esi
4 2: 53          push    %ebx
5 3: e8 fc ff ff call    4 <__ref_lgdt+0x4>
6 8: 81 c3 02 00 add     $0x2,%ebx
7 e: 83 ec 70    sub     $0x70,%esp
8 11: e8 fc ff ff call    12 <__ref_lgdt+0x12>
9 16: c7 44 24 14 movl    $0x20,0x14(%esp)
10 1d: 00
11 1e: 83 ec 08    sub     $0x8,%esp
12 21: 8d 7c 24 0c lea     0xc(%esp),%edi
13 25: 57          push    %edi
14 26: 8b 84 24 8c mov     0x8c(%esp),%eax
15 2d: 83 c0 01    add     $0x1,%eax
```

(a) __ref_.lgdt.o

```
1 00000000 <__ref_lgdt>:
2 0: 57          push    %edi
3 1: 56          push    %esi
4 2: 53          push    %ebx
5 3: e8 fc ff ff call    4 <__ref_lgdt+0x4>
6 8: 81 c3 02 00 add     $0x2,%ebx
7 e: 83 ec 78    sub     $0x78,%esp
8 11: 8d 7c 24 0c lea     0xc(%esp),%edi
9 15: c7 44 24 1c movl    $0x20,0x1c(%esp)
10 1c: 00
11 1d: 57          push    %edi
12 1e: 8b 84 24 8c mov     0x8c(%esp),%eax
13 25: 83 c0 01    add     $0x1,%eax
14 28: 50          push    %eax
15 29: e8 fc ff ff call    2a <__ref_lgdt+0x2a>
```

(b) 老师代码的lgdt.o

图 4. lgdt.o开头部分的不同

```
55 b0: 8b 83 00 00 mov     0x0(%ebx),%eax
56 b6: 8b 54 24 44 mov     0x44(%esp),%edx
57 ba: 66 89 50 28 mov     %dx,0x28(%eax)
58 be: 8b 54 24 68 mov     0x68(%esp),%edx
59 c2: 89 50 2c     mov     %edx,0x2c(%eax)
60 c5: 83 ec 80    sub     $0xfffffffff80,%esp
61 c8: 8d 46 01    lea     0x1(%esi),%eax
62 cb: 5b          pop     %ebx
63 cc: 5e          pop     %esi
64 cd: 5f          pop     %edi
65 ce: c3          ret
```

(a) __ref_.lgdt.o

```
52 a8: 8b 83 00 00 mov     0x0(%ebx),%eax
53 ae: 8b 54 24 44 mov     0x44(%esp),%edx
54 b2: 66 89 50 34 mov     %dx,0x34(%eax)
55 b6: 8b 54 24 68 mov     0x68(%esp),%edx
56 ba: 89 50 38    mov     %edx,0x38(%eax)
57 bd: 83 ec 80    sub     $0xfffffffff80,%esp
58 c0: 8d 46 01    lea     0x1(%esi),%eax
59 c3: 5b          pop     %ebx
60 c4: 5e          pop     %esi
61 c5: 5f          pop     %edi
62 c6: c3          ret
```

(b) 老师代码的lgdt.o

图 5. lgdt.o结尾部分的不同

从老师代码链接生成的lgdt.o文件和libs/nemu-ref/lib-nemu-ref.a里的lgdt.o有且仅有开头和结尾部分不同, 但是从汇编层面很难窥探出内存分配的东西, 因为源码里面函数调用顺序可能不同. 但是libs/nemu-ref/lib-nemu-ref.a的源代码肯定是和老师给的代码不一样的.

精确证明:

我把libs/nemu-ref/lib-nemu-ref.a里面通过ar -x解包出来的 lgdt.o换成老师代码(在lgdt.c中), 并make test_pa-4-1之后在nemu/src/cpu/instr下的lgdt.o文件并使用

ar -r lib-nemu-ref.a adc.o and.o cltd.o dec.o fpu_ref.o imul.o int_.o lea.o lidt.o neg.o out.o rep_repe.o scoring.o stos.o add.o bt.o cmov.o div.o fpu_test.o iret.o leave.o mov.o nop.o pop.o ret.o setcc.o sub.o xor.o alu_ref.o call.o cmp.o hlt.o inc.o jcc.o lgdt.o movs.o not.o push.o sar.o shl.o test.o alu_test.o cbw.o cmps.o flags.o idiv.o in.o jmp.o mul.o or.o reg.o sbb.o shr.o 命令

生成一个新的, 包含老师代码的lgdt.o的, lib-nemu-ref.a;

并把lgdt.c清空, opcode.c里面改成__ref_lgdt, 结果喜闻乐见:

```
make: *** [test_pa-3-3] Aborted
zhangyikai@debian:~/4_1_pa2018_fall$ ./nemu/nemu --testcase mov-c --kernel
NEMU load and execute img: ./kernel/kernel.img elf: ./testcase/bin/mov-c
(nemu) si
00030000: fa cli
(nemu) p 0
0
(nemu) si
00030001: 0f 01 15 4c 00 03 lgdt 0x3004c
(nemu) p 0
196660
(nemu) si
00030008: 0f 20 mov %cr0, %eax
(nemu)
```

图 6. 全新的lib-nemu-ref.a通过测试

这样就证明了原先的lib-nemu-ref.a中lgdt.o是有问题的!

找bug的时候一直麻烦了汪老师, 谢谢老师这些日子的教导!!!