假如我们在内核没有配置回溯信息，那么如何确定函数的调用关系？

根据栈信息找出函数的调用关系？

假如我们制造在insmod的错误。

首先找到pc值. 0xbf000018,

在kallsyms.txt 找到相关的函数名，基本跟我们上一节的学习分析类似。

//反汇编的

00000000 <first\_drv\_open>:

0: e1a0c00d mov ip, sp

4: e92dd800 stmdb sp!, {fp, ip, lr, pc}//会存在栈中

8: e24cb004 sub fp, ip, #4 ; 0x4

c: e59f1024 ldr r1, [pc, #36] ; 38 <\_\_mod\_vermagic5>

10: e3a00000 mov r0, #0 ; 0x0

14: e5912000 ldr r2, [r1]

18: e5923000 ldr r3, [r2]

1c: e3c33c3f bic r3, r3, #16128 ; 0x3f00

20: e5823000 str r3, [r2]

24: e5912000 ldr r2, [r1]

28: e5923000 ldr r3, [r2]

2c: e3833c15 orr r3, r3, #5376 ; 0x1500

30: e5823000 str r3, [r2]

34: e89da800 ldmia sp, {fp, sp, pc}

38: 00000000 andeq r0, r0, r0

//错误的堆栈错误：

Stack: (0xc3ec5e88 to 0xc3ec6000)

5e80: c3ec5ebc c3ec5e98 c008d888 bf000010 00000000 c3ec63e0

First\_drv\_open函数的栈c3ec5ebc到bf000010 lr是c008d888这个肯定是内核的

在内核的反汇编中查找c008d888 它属于 chrdev\_open 它有九个寄存器

5ea0: c3e9b0c0 c008d73c c0474e20 c3e9d24c c3ec5ee4 c3ec5ec0 c0089e48 c008d74c

chrdev\_open的lr c0089e48 因为sp-4所以

5ec0: c3ec63e0 c3ec5f04 00000003 ffffff9c c002c044 c04de000 c3ec5efc c3ec5ee8

caller’s sp c3ec63e0

5ee0: c0089f64 c0089d58 00000000 00000002 c3ec5f68 c3ec5f00 c0089fb8 c0089f40

5f00: c3ec5f04 c3e9d24c c0474e20 00000000 00000000 c3d35000 00000101 00000001

5f20: 00000000 c3ec4000 c046d8c8 c046d8c0 ffffffe8 c04de000 c3ec5f68 c3ec5f48

5f40: c008a16c c009fc70 00000003 00000000 c3ec63e0 00000002 bee4aee0 c3ec5f94

5f60: c3ec5f6c c008a2f4 c0089f88 00008520 bee4aed4 0000860c 00008670 00000005

5f80: c002c044 4013365c c3ec5fa4 c3ec5f98 c008a3a8 c008a2b0 00000000 c3ec5fa8

5fa0: c002bea0 c008a394 bee4aed4 0000860c 00008720 00000002 bee4aee0 00000001

5fc0: bee4aed4 0000860c 00008670 00000002 00008520 00000000 4013365c bee4aea8

5fe0: 00000000 bee4ae84 0000266c 400c98e0 60000010 00008720 00766564 00000000

根据韦东山开发手册ATPCS:

fp, r11

ip, R12

lr, r14

pc r15寄存器

高地址存高标号，在栈中由高到低：

pc

Lr //程序的返回地址，根据栈信息

Ip

Fp

栈是向下生长的。

