作业5

3.32、3.33、3.35、3.38、3.41、3.45、3.48、3.69、3.70、

3.32

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 指令 | | | 状态值（指令执行前） | | | | | |
| 标号 | PC | 指令 | %rdi | %rsi | %rax | %rsp | \* %rsp | 描述 |
| M1 | 0x400560 | callq | 10 | - | - | 0x7fffffffe820 | - | 调用first(10) |
| F1 | 0x400548 | lea | 10 | - | - | 0x7fffffffe818 | - | first的入口 |
| F2 | 0x40054c | sub | 10 | 11 | - | 0x7fffffffe818 | 0x400565 |  |
| F3 | 0x400550 | callq | 9 | 11 | - | 0x7fffffffe818 | 0x400565 | 调用last(9, 11) |
| L1 | 0x400540 | mov | 9 | 11 | - | 0x7fffffffe810 | 0x400555 | last的入口 |
| L2 | 0x400543 | imul | 9 | 11 | 9 | 0x7fffffffe810 | 0x400555 |  |
| L3 | 0x400547 | retq | 9 | 11 | 99 | 0x7fffffffe810 | 0x400555 | 从last返回99 |
| F4 | 0x400555 | repz repq | 9 | 11 | 99 | 0x7fffffffe818 | 0x400565 | 从first返回99 |
| M2 | 0x400565 | mov | 9 | 11 | 99 | 0x7fffffffe820 | - | 继续执行main |

3.33

1. 假设第一次加法对应的是\*u += a，第二次加法对应的是\*v += b，由汇编代码可见，a先被从4个字节转成了8个字节，再加到u指向的8个字节上，即a为int型，u为long \*型，而b的低位字节被加到了v指向的字节，则v是char \*型，又因为返回值为size(a) + size(b) = 6，且已知size(a) = 4，则size(b) = 2，b为short型。

2. 假设第一次加法对应的是\*v += b，第二次加法对应的是\*u += a，同理由汇编代码可得a为short型，b为int型，u为char \*型，v为long \*型。

综上，4个参数的合法顺序和类型分别为：

int a, short b, long \*u, char \*v或int b, short a, long \*v, char \*u。

3.35

A. rfun存储在寄存器%rbx中的是参数x的值。

B. long rfun(unsigned long x) {

if (x == 0)

return 0;

unsigned long nx = x >> 2;

long rv = rfun(nx);

return x + rv;

}

3. 38

P[i][j]的地址为P + (7 \* i + j) \* 8，Q[j][i]的地址为Q + (5 \* j + i) \* 8，则M = 5，N = 7。

3.41

A. p：0

s.x：8

s.y：12

next：16

B. 这个结构总共需要24个字节

C. void sp\_init(struct prob \*sp) {

sp->s.x = sp->s.y;

sp->p = &(sp->s.x);

sp->next = sp;

}

3.45

A. a：0 b：2 c：8 d：16 e：24 f：32 g：40 f：48。

B. 这个结构总的大小是56字节。

C. 将结构的元素按大小降序排列得到：

struct {

char \*a;

double c;

long g;

float e;

int h;

short b;

char d;

char f;

}

此时偏移量为：a：0，c：8:，g：16，e：24；h：32；b：36；d：38；f：39，浪费的空间最小化了，重排过的结构的字节偏移量和总的大小为40字节。

3.48

A. 没有保护的代码：v存放在24(%rsp)，buf存放在(%rsp)

有保护的代码：金丝雀存放在40(%rsp)，v存放在8(%rsp)，buf存放在16(%rsp)

B. 在有保护的代码中，v比buf更靠近栈顶，buf溢出不会破坏v的值。

3.69

A. 由汇编代码可知a\_struct结构的大小是40字节，0x120 = 288 = 8 + 7 \* 40，CNT = 7

B. typedef struct {

long idx,

long x[4]

} a\_struct

3.70

A. e1.p 0

e1.y 8

e2.x 0

e2.next 8

B. 这个结构总共需要16字节

C. void proc(union ele \*up) {

up->e2.x = \*( \*(up->e2.next).e1.p ) - \*(up->e2.next).e1.y

}