4

1. 立即数0xFF前要加$.
2. 源操作数是16位和指令长度后缀b不一致.
3. 目的操作数不能是立即数.
4. 源操作数长度超过16位和指令长度后缀w不一致.
5. 8位寄存器不能作为目的操作数地址所在寄存器.
6. 源操作数和目的操作数长度不一致.
7. 不存在ESX寄存器
8. 缺少变址寄存器.

5

|  |  |  |
| --- | --- | --- |
| src\_type | dst\_type | 机器级表示 |
| char | int | movsbl %al, (%edx) |
| int | char | movb %al, (%edx) |
| int | unsigned | movl %eax, (%edx) |
| short | int | movswl %ax, (%edx) |
| unsigned char | unsigned | movzbl %al, (%edx) |
| char | unsigned | movzbl %al, (%edx) |
| int | int | movl %eax, (%edx) |

6

(1)

xptr, yptr, zptr分别在R[ebp] + 8, R[ebp] + 12, R[ebp] + 16.

(2)

void func(int \*xptr, int \*yptr, int \*zptr)

{

int x = \*xptr; int y = \*yptr; int z = \*zptr;

\*yptr = x; \*zptr = y; \*xptr = z;

}

15

int f1(unsigned x)

{

int y = 0;

while (x != 0)

{

y ^= x;

x >>= 1;

}

return y & 1;

}

判断x的二进制表示中1的个数的奇偶性, 若有奇数个1, 则返回1.

17

unsigned test(char a, unsigned short b, unsigned short c, short \*p);

22

根据汇编指令, 最终返回值为M[a + 4 \* (7i + j)] + M[b + 4 \* (5j + i)],

于是M = 5, N = 7.

25

(1)

16字节. 偏移地址分别为0, 4, 8, 12.

(2)

void np\_init(struct node \*np)

{

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

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

np->next = np;

}

28

偏移量分别为0, 8, 16, 20, 24, 28, 32, 40.

因为d和g都有8字节长, 所以结构总大小必须是8的倍数, 所以为44 + 4 = 48字节.

调整为:

struct {

double d;

long long g;

int i;

char \*p;

long l;

void \*v;

short s;

char c;

} test;

此时结构总大小为40字节.

31

(1)

movl 8(%ebp), %edx // edx = x

movl 12(%ebp), %ecx // ecx = k

movl $255, %esi // esi = val, 初值为255

movl $-2147483648, %edi // edi = i, 初值为-2147483648

.L3:

movl %edi, %eax // eax = edi = i

andl %edx, %eax // eax = eax & edx = i & x, 连上前一句即i &= x;

xorl %eax, %esi // esi = esi ^ eax = val ^ (i & x). 即 val ^= (i & x)

movl %ecx, %ebx // ebx = ecx = k

shrl %bl, %edi // edi = edi >> bl = i >> (k & 0xFF), 即i >>= (k & 0xFF)

testl %edi, %edi // (edi == 0) ?, 即(i == 0)?

jne .L3 // edi != 0, jump to .L3

movl %esi, %eax // return value = eax = esi, 即return val

(2)

x和k分别存放在edx和ecx中, val和i分别存放在esi和edi中.

(3)

val的初始值为255, i的初始值为-2147483648

(4)

循环中止条件为 i == 0, 每次循环i = i >> (k & 0xFF), 这里的右移是逻辑右移, 同时由于i为32位的有符号数, 所以这里可以简化为i = (unsigned) i >> k.

(5)

int lproc(int x, int k)

{

int val = 255;

int i;

for (i =0x80000000; i != 0; i = (unsigned) i >> k)

val ^= (i & x);

return val;

}

33

(1)

分别为0, 4, 0, 4.

(2)

占8字节.

(3)

void chain\_proc(union node \*uptr)

{

uptr->n2.next->n1.data1 = \*(uptr->n2.next->n1.ptr) – uptr->n2.data2;

}