南京邮电大学 CTF 逆向部分 Writeup

转载

メイ 于 2017-01-07 23:04:00 发布 113 收藏
原文链接：http://www.cnblogs.com/Chesky/p/nuptzj_re_writeup.html

Hello,RE!

提示 IDA 中按 R 。
Google 到 IDA 中 R 快捷键是 Character ， 转为字符串。
丢进 IDA（虽然我并不会使用 IDA

有个 strcmp 函数，比较 v4 和 v5 地址的。选中 v5 ，发现倒序的 flag 字样。
得到 Flag 为 flag{Welcome_To_RE_World!}

ReadAsm2

main函数部分：

```
int main(int argc, char const *argv[])
{
  char input[] = {0x0,  0x67, 0x6e, 0x62, 0x63, 0x7e, 0x74, 0x62, 0x69, 0x6d,
                  0x55, 0x6a, 0x7f, 0x60, 0x51, 0x66, 0x63, 0x4e, 0x66, 0x7b,
                  0x71, 0x4a, 0x74, 0x76, 0x6b, 0x70, 0x79, 0x66 , 0x1c};
  func(input, 28);
  printf("%s\n",input+1);
  return 0;
}
```

ASM中文件内容（func函数）

```
00000000004004e6 <func>:
  4004e6: 55                    push   rbp
  4004e7: 48 89 e5              mov    rbp,rsp
  4004ea: 48 89 7d e8           mov    QWORD PTR [rbp-0x18],rdi
  4004ee: 89 75 e4              mov    DWORD PTR [rbp-0x1c],esi
  4004f1: c7 45 fc 01 00 00 00  mov    DWORD PTR [rbp-0x4],0x1
  4004f8: eb 28                 jmp    400522 <func+0x3c>
  4004fa: 8b 45 fc              mov    eax,DWORD PTR [rbp-0x4]
  4004fd: 48 63 d0              movsxd rdx,eax
  400500: 48 8b 45 e8           mov    rax,QWORD PTR [rbp-0x18]
  400504: 48 01 d0              add    rax,rdx
  400507: 8b 55 fc              mov    edx,DWORD PTR [rbp-0x4]
  40050a: 48 63 ca              movsxd rcx,edx
  40050d: 48 8b 55 e8           mov    rdx,QWORD PTR [rbp-0x18]
  400511: 48 01 ca              add    rdx,rcx
  400514: 0f b6 0a              movzx  ecx,BYTE PTR [rdx]
  400517: 8b 55 fc              mov    edx,DWORD PTR [rbp-0x4]
  40051a: 31 ca                 xor    edx,ecx
  40051c: 88 10                 mov    BYTE PTR [rax],dl
  40051e: 83 45 fc 01           add    DWORD PTR [rbp-0x4],0x1
  400522: 8b 45 fc              mov    eax,DWORD PTR [rbp-0x4]
  400525: 3b 45 e4              cmp    eax,DWORD PTR [rbp-0x1c]
  400528: 7e d0                 jle    4004fa <func+0x14>
  40052a: 90                    nop
  40052b: 5d                    pop    rbp
  40052c: c3                    ret
```

先 Mark 一记可能会关闭的网站：http://www.aogosoft.com/

逐行注释：

分析过程记录：

1.main() 部分

0x 表示十六进制。
flag 是对此 input 数组的操作组合而成。

http://www.bluesock.org/~willg/dev/ascii.html

2.函数调用部分

4004e6：表示该指令对应的虚拟内存地址
55：该指令对应的计算机指令

函数调用过程：

> 入栈，将寄存器的值压入调用 bp 栈中
> 建立新栈帧，别掉函数栈帧栈底地址放入寄存器

实现

```
push   rbp
mov    rbp,rsp
```

寄存器类型：

```
ax(accumulator): 可用于存放函数返回值
bp(base pointer): 用于存放执行中的函数对应的栈帧的栈底地址
sp(stack poinger): 用于存放执行中的函数对应的栈帧的栈顶地址
ip(instruction pointer): 指向当前执行指令的下一条指令
```

前缀加上 r 表示 64 位， e 表示 32 位，使用时表示该寄存器存储 xx 位的数据。

3.执行过程

```
00000000004004e6 <func>:
  4004e6: 55                    push   rbp                          /*函数调用
  4004e7: 48 89 e5              mov    rbp,rsp                             */
  4004ea: 48 89 7d e8           mov    QWORD PTR [rbp-0x18],rdi       //rdi 存第一个参数
  4004ee: 89 75 e4              mov    DWORD PTR [rbp-0x1c],esi       //esi 存第二个参数
  4004f1: c7 45 fc 01 00 00 00  mov    DWORD PTR [rbp-0x4],0x1        //在[rbp-0x4]写入 0x1
  4004f8: eb 28                 jmp    400522 <func+0x3c>
  4004fa: 8b 45 fc              mov    eax,DWORD PTR [rbp-0x4]         //把[rbp-0x4]的值送入 eax ,即 eax = 1
  4004fd: 48 63 d0              movsxd rdx,eax                         //扩展,传送 rdx=1
  400500: 48 8b 45 e8           mov    rax,QWORD PTR [rbp-0x18]        //第一个参数 [rbp-0x18]，rax=input[0]
  400504: 48 01 d0              add    rax,rdx                         //rax = input[1]
  400507: 8b 55 fc              mov    edx,DWORD PTR [rbp-0x4]         //第 6 行中存储的 0x1 ,传入 edx ,即 ed
  40050a: 48 63 ca              movsxd rcx,edx                          //rcx=1
  40050d: 48 8b 55 e8           mov    rdx,QWORD PTR [rbp-0x18]         // rdx = input[0]
  400511: 48 01 ca              add    rdx,rcx                          //rdx += rcx ,rdx = input[1]
  400514: 0f b6 0a              movzx  ecx,BYTE PTR [rdx]               //ecx = input[1]
  400517: 8b 55 fc              mov    edx,DWORD PTR [rbp-0x4]          //edx = 0x1
  40051a: 31 ca                 xor    edx,ecx                          //edx ^= ecx ,原先 ecx 为 1100111，e
  40051c: 88 10                 mov    BYTE PTR [rax],dl                //rax = dl
  40051e: 83 45 fc 01           add    DWORD PTR [rbp-0x4],0x1          //[rbp-0x4]处为 0x1
  400522: 8b 45 fc              mov    eax,DWORD PTR [rbp-0x4]          //把[rbp-0x4]的值送入 eax
  400525: 3b 45 e4              cmp    eax,DWORD PTR [rbp-0x1c]         // 比较操作，将[rbp-0x1c] 处的值和eax
  400528: 7e d0                 jle    4004fa <func+0x14>               //eax < 28 时跳转至 4004fa   func(in
  40052a: 90                    nop
  40052b: 5d                    pop    rbp
  40052c: c3                    ret
```

- -word 表示字
  q 四字 d 双字
  dword qword

```
dword 2*16 =32 位
qword 4*16 = 64 位
```

PTR 指针（pointer）
没有寄存器名时， X ptr 指明内存单元的长度，X 在汇编指令中可以为word 或 byte 。

- 内存地址

```
[rbp-0x18]
```

- 涉及指令

```
1.movsxd 指令为扩展至零
将32位的寄存器和内存操作数符号扩展到64位的寄存器
2.逻辑异或运算指令 XOR
XOR OPRD1,OPRD2
实现两个操作数按位‘异或’(异为真,相同为假)运算,结果送至目的操作数中.
OPRD1<--OPRD1 XOR OPRD2
3.JLE
 小于等于时转移
```

操作行为链：

```
rdx—rax
edx—rcx
rcx—rdx 作为累加
```

总而言之，func函数部分进行的操作是
遍历数组，将一个值和该值的前一个值进行异或运算，得到这个具体字母。

```
input = [0x67,0x6e,0x62,0x63,0x7e,0x74, 0x62, 0x69, 0x6d, 0x55, 0x6a, 0x7f, 0x60, 0x51, 0x66, 0x63, 0x4e,
flag = ""
for x in range(1,28):
    flag = flag + chr(input[x-1]^ x)
print flag
```

文章MD文件备份：
链接: http://pan.baidu.com/s/1nvPkNOH 密码: 84dp

参考资料：
http://www.cnblogs.com/clover-toeic/p/3755401.html
http://www.cnblogs.com/bangerlee/archive/2012/05/22/2508772.html
http://book.51cto.com/art/201210/359678.htm

转载于:https://www.cnblogs.com/Chesky/p/nuptzj_re_writeup.html