2021/03/25_x86逆向_第16课_结构体

笔记本： x86逆向-C
创建时间： 2021/3/25 星期四 10:07
作者： ileemi

结构体

当程序中使用结构体且满足下面的条件时（通过汇编代码观察）就可以将其识别为结构体，反之就将其解释为变量赋值：

- 结构体变量当作函数参数
- 结构体指令变量当作函数参数
- 结构体变量当作返回值
- 结构体指针变量当作返回值
- 有明显的对其行为

结构体的识别

> 满足一段内存块中的数据连续且不一致（作用不一致）的条件就可以将其还原为结构体。

- 成员访问：存在常量折叠
- 指针访问：直接从结构体指针变量的首地址+成员在结构中所表示的offset
- 结构体变量进行赋值：Debug下会产生类似 "memcpy" 的汇编代码（memcpy(&tagTest2, &tagTest1, sizeof(tagTest)); -- 生成对应的无分支循环拷贝），Release版下会优化。
- 结构体变量传参
  - 结构体较小时，VC++6.0会引发一次拷贝，拷贝结构体数据到栈中，有 "mov edi, esp" 一句标志性汇编代码（VC++ 6.0 Release也有）Debug，示例：

```
sub esp, 20h // 向上抬栈
mov ecx, 8  // 拷贝的次数
// 源操作数：结构体变量的首地址
lea esi, [ebp-20h]
//  目标操作数：栈顶拷贝，该行汇编代码为结构体变量传参的标志
mov edi, esp
```

```
rep movs dword ptr [edi], dword ptr [esi]
call xxx
```

- 结构体较小时，高版本VS编译的程序会使用 "xmm0" 向栈上传递数据。
- 结构体较大时，高版本VS编译的程序会使用类似 "memcpy" 的无分支向栈拷贝方法，同样会生成带有 "mov edi, esp" 的标志性汇编代码。
- 小微型结构体（结构体有两个成员时）会使用edx、eax进行传参

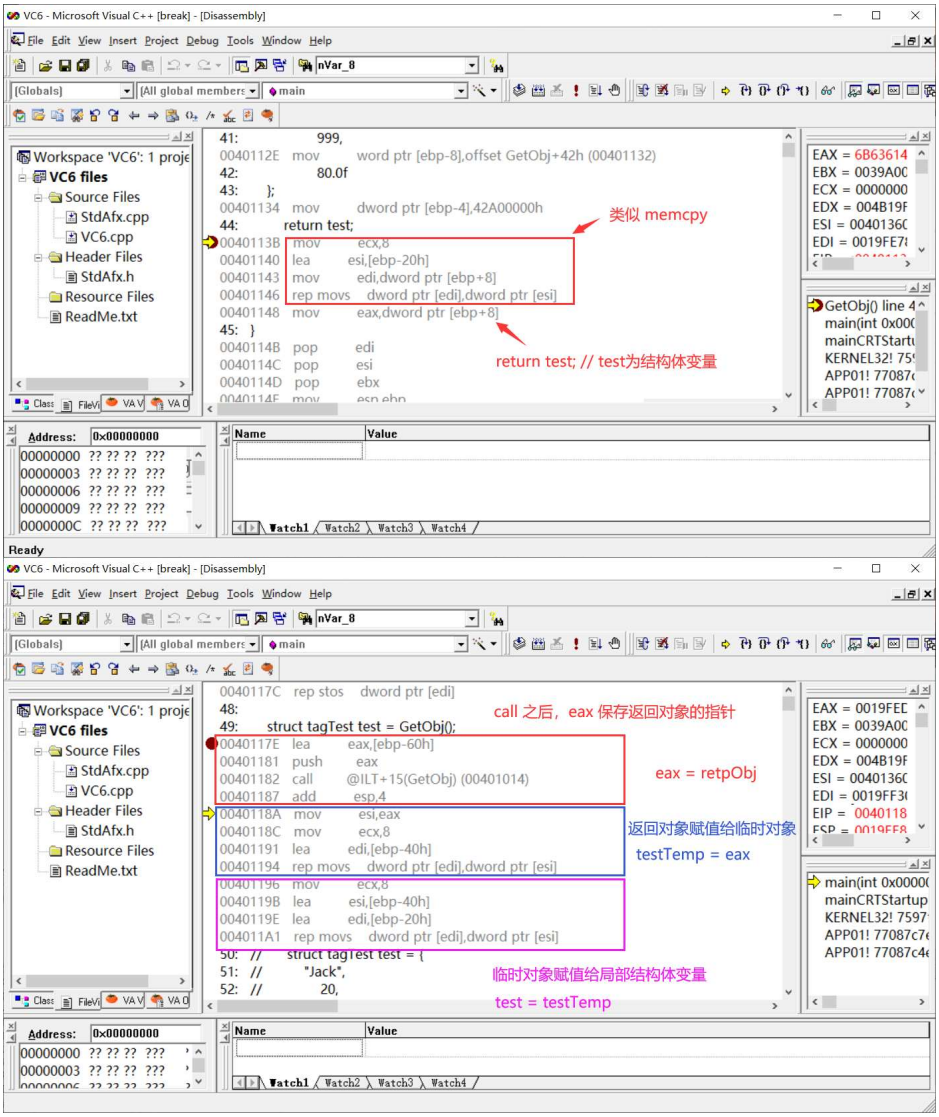
- 返回值

  结构体对象：

  - 对于小微型结构体（结构体有两个成员时），函数内部会通过 edx.eax 往栈上的临时变量拷贝，即返回临时对象，再进行变量之间的拷贝。
  - 结构体比较大时，编译器生成的汇编代码会额外加一个隐藏参数，该参数就是返回值，eax 存放这个对象的指针，依然会产生临时对象。

```
lea eax, [ebp-xxh] // 隐含参数，结构体指针
push eax // eax 保存返回对象的指针
call xxx
...
// 返回对象赋值给临时对象
...
// 临时对象赋值给局部结构体变量
...

// 对此分析出来的函数会有以下特征
tagTest *GetObj(tagTest* pObj) {
    // 功能操作
    ...
    // 将结果拷贝给参数并返回
    memcpy(pObj, xxx, sizeof(tagTest));
    return pObj;
}
...
```



> [reg + imm]（寄存器相对寻址） 如果不是lea的运算应用，就有以下两种可能：

1. 数组常量下标访问
2. 结构体成员访问

识结构体的要点：看成员访问，看传参，看返回值。

传递参数：

- 结构体指针：比产生 "[reg + imm]" 的访问方式
- 对象：mov esi, esp

返回值：

- 结构体变量：观察该函数的第一个参数
- 结构体指针：调用后，比产生 "[reg + imm]" 的访问方式

以上结构体的识别，在高版本VS编译器编译程序生成的汇编代码会进行优化。