2021/05/19_x86逆向C++_第6课_虚继承

笔记本: x86逆向-C++
创建时间: 2021/5/19 星期三 15:11
作者: ileemi

课前会议

多重继承的继承顺序，需要在构造函数中查看，this指针+0表示第一个基类，this指针+4表示第二个基类。基类的虚表会被子类虚表覆盖。
多重继承子类第一个继承的类共用this指针（共享），第二个继承的类this指针需要+4，依次类推。

多重继承子类中交换重写的基类虚函数位置，不会影响基类的虚表顺序（由基类决定）。

强转基类指针，编译器会自动调整this指针位置。
Sofabed* pObject1 = new Sofabed();
Sofa* pSofa = pObject1; // pSofa = (char*)pObject1 + 4;
pSofa->sit();
Bed* pBed = pObject1; // pSofa = (char*)pObject1;
pBed->lie();

继承，子类在覆盖虚表前会填充基类的虚表（在同一个地址进行覆盖）。如果填虚表操作在覆盖虚表之后，那么就可以将其解释为类成员（VS高版本，VC++6.0 会把该操作反过来）。

菱形继承

有虚函数的情况下，其内存结构如下：

```
//////////////
// A对象
A::vatable  =  {A::~A,  A::fun1}
A::member
//////////////
```

```
// B对象
B::vatable = {B::~B, A::fun1, B::fun2}
A::member
B::member
//////////////
// C对象
C::vatable = {C::~C, A::fun1, C::fun3}
A::member
C::member
//////////////
// BC对象
B::vatable = {BC::~BC, BC::fun1, B::fun2, BC::fun4}
A::member
B::member
C::vatable = {BC::~BC, BC::fun1, BC::fun3}
A::member
C::member
BC::member
```

存在访问对象不明确，需要指定类的作用域。

虚继承

> 虚继承：解决菱形继承的问题

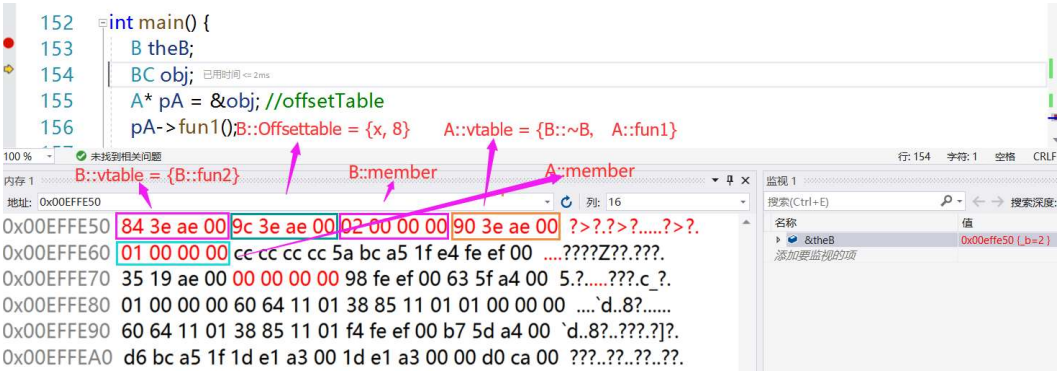
VC编译器有提供虚基类偏移表（和虚表在同一个位置）：根据构造情况会实时记录偏移位置。获取虚表地址可通过对象首地址加偏移。偏移表由编译器扫描类代码定位。基类偏移表要早于虚表。每个派生类对象都有一个虚基类偏移表。

虚基类对象内存结构：

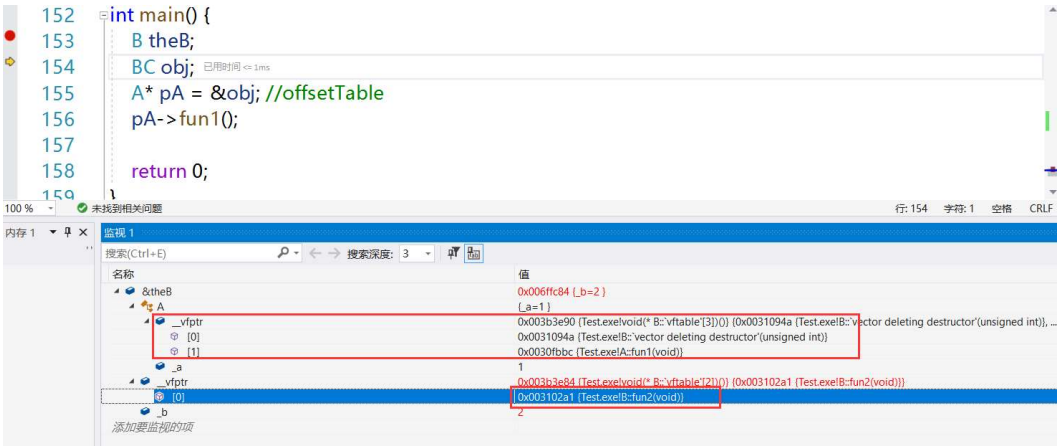
```
// A对象
A::vtable = {B::~B, A::fun1}
A::member
```

派生类（B）虚继承 A 的内存结构：

```
// B对象
B::vtable = {B::fun2}
B::Offsettable = {x, 8}
B::member
A::vtable = {B::~B, A::fun1}
A::member
```

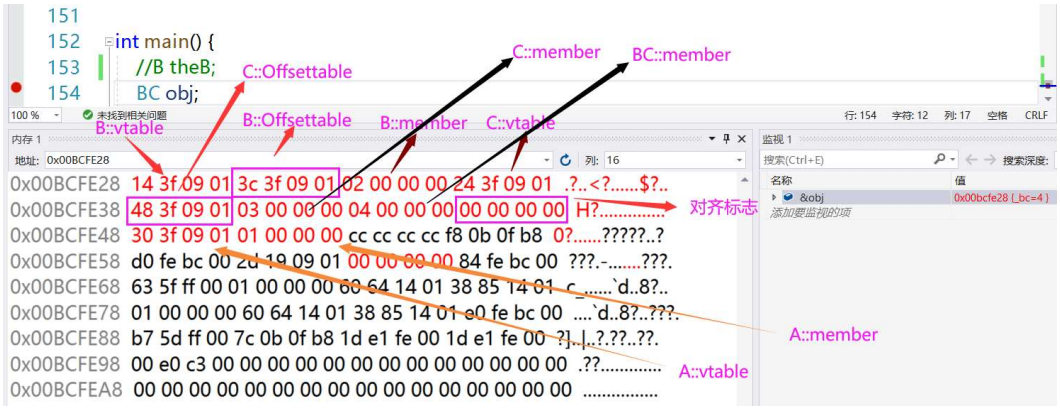


此时虚表结构：



派生类（BC）的内存结构如下：

```
// BC对象的内存结构(BC类继承 B、C， B、C继承A)
B::vtable = {B::fun2, BC::fun4}
B::Offsettable = {x, 28} // 虚基类执行构造代码之前
B::member
C::vtable = {BC::fun3}
C::Offsettable = {x, 16} // 虚基类执行构造代码之前
C::member
BC::member
00000000 对齐标志
A::vtable = {BC::~BC, BC::fun1} // 虚基类
A::member
```



虚继承的特征

1. 通过构造函数的参数区分要不要调用基类构造（Release版，虚继承派生类对象单独使用不会优化掉）

```
push    1
lea     ecx, [ebp+var_30]
call    BC_clasee_Construct
```

2. 会出现偏移表，在覆盖虚表之前填充

```
cmp     [ebp+arg_0], 0
jz      short loc_454901
mov     eax, [ebp+var_8]
mov     dword ptr [eax+4], offset unk_4F3E9C
mov     ecx, [ebp+var_8]
add     ecx, 0Ch
call    B_clasee_Construct
mov     eax, [ebp+var_D4]
or      eax, 1
mov     [ebp+var_D4], eax

loc_454901:                             ; CODE XREF: sub_4548A0+3B↑j
mov     eax, [ebp+var_8]
mov     dword ptr [eax], offset ??_7B@@6B@ ; const B::`vftable'
mov     eax, [ebp+var_8]
mov     ecx, [eax+4]
mov     edx, [ecx+4]
mov     eax, [ebp+var_8]
mov     dword ptr [eax+edx+4], offset ??_7B@@6B@_0 ; const B::`vftable'
mov     eax, [ebp+var_8]
mov     dword ptr [eax+8], 2
```

偏移表

覆盖虚表

3. 覆盖虚基类的虚表，需要通过偏移表定位位置

```
mov     [ebp+var_D4], eax

loc_454901:                             ; CODE XREF: sub_4548A0+3B↑j
mov     eax, [ebp+var_8]
mov     dword ptr [eax], offset ??_7B@@6B@ ; const B::`vftable'
mov     eax, [ebp+var_8]
mov     ecx, [eax+4]    ; 获取偏移表地址
mov     edx, [ecx+4]    ; 获取偏移值
mov     eax, [ebp+var_8]
mov     dword ptr [eax+edx+4], offset ??_7B@@6B@_0 ; const B::`vftable'
mov     eax, [ebp+var_8]
mov     dword ptr [eax+8], 2
push    offset aBB      ; "B::B()"
call    sub_44EA5F
loc_454A11:                             ; CODE XREF: sub_454980+5A↑j
push    0
mov     ecx, [ebp+var_14]
call    sub_44F969
mov     [ebp+var_4], 1
push    0
mov     ecx, [ebp+var_14]
add     ecx, 0Ch
call    sub_44FF40
mov     eax, [ebp+var_14]
mov     dword ptr [eax], offset ??_7BC@@6B@ ; const BC::`vftable'
mov     eax, [ebp+var_14]
mov     dword ptr [eax+0Ch], offset ??_7BC@@6B@_0 ; const BC::`vftable'
mov     eax, [ebp+var_14]
mov     ecx, [eax+4]    ; 获取偏移表地址
mov     edx, [ecx+4]    ; 获取偏移值
mov     eax, [ebp+var_14]
mov     dword ptr [eax+edx+4], offset ??_7BC@@6B@_1 ; const BC::`vftable'
mov     eax, [ebp+var_14]
mov     ecx, [eax+4]
mov     edx, [ecx+4]
sub     edx, 1Ch
```

通过间接访问，获取虚基类虚表地址，并进行覆盖

安全公司

安芯网盾