之前提到IDA可以将一长串的数组数据声明变成一行数组声明，简化反汇编代码，对于结构体，IDA也同样支持通过各种设置工具来改善结构体代码的可读性。

这篇文章的目标是将[edx+10h]之类的结构体元素访问 变成 [edx+struct_filed5]之类。

只要发现一个程序正操纵某种数据结构，你就需要确定：你是否希望将结构体的字段名称合并到反汇编代码清单中，或者你是否理解分散在代码清单中的所有数字偏移量。

如果是C标准库或者IDA能够确定的已知结构体，IDA会自动将数字偏移量转换成更加符号化的字段名称，但如果是自定义的结构体，就需要我们手动帮助IDA进行识别。

IDA之所以在分析阶段无法识别结构体，可能源于两个原因。首先，虽然IDA了解某个结构体的布局，但它并没有足够的信息，能够判断程序确实使用了结构体。其次，程序中的结构体可能是一种IDA对其一无所知的非标准结构体。在这两种情况下，问题都可以得到解决，且首先从Structures窗口下手。

创建一个新的结构体（或Union）

在IDA Structures窗口（默认展示，如果没有可以在View -> Open SubViews里打开）中，列举了当前反汇编文件的所有已知结构体，并且在注释中也提示了我们如何在该窗口中添加/删除/编辑一个结构体:

```
00000000 ; Ins/Del : create/delete structure 或者 Edit -> Add struct
type
00000000 ; D/A/*   : create structure member (data/ascii/array)
00000000 ; N       : rename structure or structure member
00000000 ; U       : delete structure member
```

注意：**只有当一个字段是结构体中的最后一个字段时，使用U键才能删除该字段。对于所有其他字段，按下U键将取消该字段的定义，这样做仅仅删除了该字段的名称，并没有删除分配给该字段的字节。**

为了创建一个新的结构体，你必须首先在Structure name（结构体名称）字段中指定结构体的名称。前两个复选框用于决定新结构体在Structures窗口中的显示位置，或者是否在窗口中显示新结构体。第三个复选框Creat union（创建联合），指定你定义的是否为C风格联合结构体(Union和结构体的区别在于union中字段相互重叠，因此总大小等于其中最大字段的大小)。

此外还有一个按钮「add standard structure」，用于添加标准结构体。IDA内置了大量的各种库和API函数有关的数据结构，IDA会尝试确定与二进制文件有关的编译器和平台，并加载适当的结构体模板。单击这个按钮，IDA将显示与当前编译器（在分析阶段检测出来）和文件格式有关的结构体主列表。这个结构体主列表中还包含通过解析C头文件添加到数据库中的结构体。

编辑结构体成员

为了给新结构体添加字段，你必须利用字段创建命令D、A和数字键盘上的星号键（*），需要注意的是D快捷键还可用于为当前光标所所在的字段调整数据大小。所以推荐的创建结构体的步骤为：

1. 在结构体定义的最后一行(包含ends的那一行)按下D键，这时候，IDA会在结构体末尾添加一个新字段，默认的字段名称为filed_N，N为结构体开头到字段开头的数字偏移量。
2. 如果需要修改字段大小，可以重复按D，循环数据转盘上的数据类型，或者通过Options->Setup Data Types来指定一个在数据转盘上不存在的大小，如果是数组，右键名称然后选择Array。
3. 如果要更改字段名称，使用N或者右键然后Rename

如果知道结构体的大小，而不了解它的布局，你需要创建两个字段。第一个字段为一个数组，它的大小为结构体的大小减去1个字节（size-1）；第二个字段应为1个字节。创建第二个字段后，取消第一个（数组）字段的定义。这样，结构体的大小被保留下来，随后，当你进一步了解该结构体的布局后，你可以回过头来定义它的字段及其大小。

此外，在你定义和编辑结构体时，IDA会有一些提示：

- 一个字段的字节偏移量以一个8位十六进制值在Structures窗口的左侧显示。
- 每次你添加或删除一个结构体字段，或更改一个现有字段的大小时，结构体的新大小都会在结构体定义的第一行反映出来。
- 你必须对一个结构体定义中的所有字段进行适当的对齐。如果你需要填补字节，那么你必须负责添加这些字节。填补字节最好作为适当大小的哑字段添加。
- 分配到结构体中间的字节只有在取消关联字段的定义后才能删除，使用Edit▶Shrink Struct Type（缩小结构体类型）即可删除被取消定义的字节。
- 你也可以在结构体的中间添加新的字节：选择新字节后面的一个字段，然后使用Edit▶Expand Struct Type（扩大结构体类型）在选中的字段前插入一定数量的字节。

通过重复应用这些步骤（添加字段，设置字段大小，添加填补字节等），就可以完成IDA结构体定义的创建。接下来就可以利用这些定义好的结构体改善反汇编代码。

使用结构体模版

前面提到这篇文章的目标是将[edx+10h]之类的结构体元素访问 变成[edx+struct_filed5]之类。

此时有了结构体定义之后，就可以右键10h，选择Structure Offset（结构体偏移量）选项，然后就可以进行格式化了，如果有多个结构体中字段匹配上了偏移量，则会有多个选择。

如果是栈变量或者全局变量，则可以直接将栈和全局变量格式化为整个结构体，首先打开栈帧的详细视图（双击函数栈帧中的变量），然后使用ALT+Q（Edit->Struct Var）命令显示一组已知的结构体并选择对应想要格式化的结构体即可。如果是全局变量，则选择要格式化的全局变量或者表示结构体开头部分的地址，再使用ALT+Q（Edit->Struct Var）即可。

重新格式化之后，IDA认识到，任何对分配给var_18的24个字节块的内存引用，都必须引用该结构体中的一个字段。如果IDA发现这样一个引用，它会尽一切努力，将这个内存引用与结构体变量中的一个已定义的字段关联起来。

导入结构体

IDA能够解析C（而非C++）数据声明，以及整个C头文件，并自动为在这些声明或头文件中定义的结构体创建对应的IDA结构体。如果你碰巧拥有你正进行逆向工程的二进制文件的源代码，或者至少是头文件，那么，你就可以让IDA直接从源代码中提取出相关结构体，从而节省大量时间。

解析C结构体声明

如果我们已经有某个结构体的声明代码，那可以直接在View▶Open Subviews▶Local Types（查看▶打开子窗口▶本地类型）窗口中，输入我们的结构体声明代码，例如:

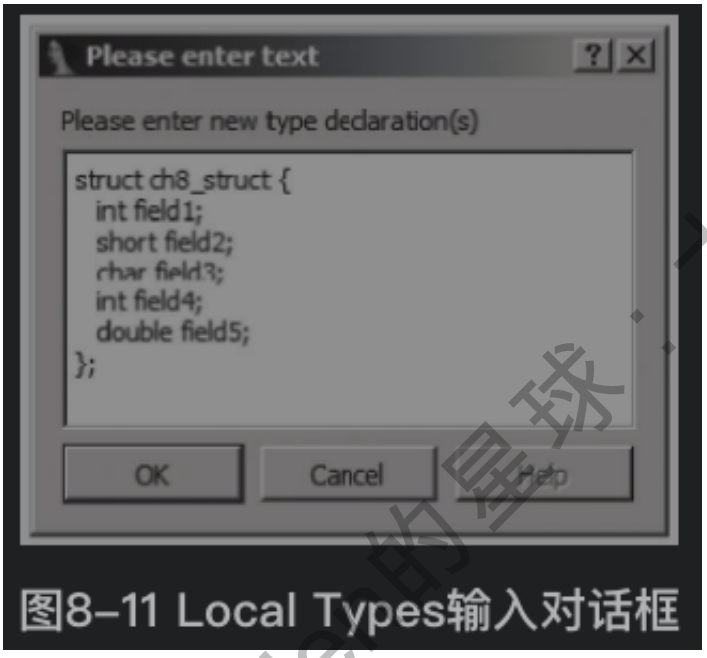

图8–11 Local Types输入对话框

注意，IDA解析器使用4字节的默认结构体成员对齐方式。如果你的结构体需要其他对齐方式，你可以包括该对齐方式，IDA认可使用pragma pack指令来指定所需的结构体成员对齐方式。

添加到Local Types（本地类型）窗口中的数据类型不会立即在Structures（结构体）窗口中出现。有两种方法可以将本地类型声明添加到Structures窗口中。最简单的方法是在相关本地类型上单击鼠标右键，并选择Synchronize to idb。或者，由于每个新类型均被添加到标准结构体列表中，因而也可将新类型导入到Structures窗口中即add standard structure。

解析C头文件

要解析头文件，可以使用File►Load File►Parse C Header File（文件►加载文件►解析C头文件）选择你想要解析的头文件。如果一切正常，IDA会通知你Compilation successful（编译完成）。如果解析器遇到任何问题，IDA将会在输出窗口中显示错误消息。

IDA会将所有被成功解析的结构体添加到当前数据库的标准结构体列表末尾，此时同样需要手动从标准结构体列表中添加到Structures窗口中。如果新结构体的名称与现有结构体的名称相同，IDA会用新结构体布局覆盖原有结构体定义。

一般而言，要最大限度地提高成功解析一个头文件的几率，你需要使用标准C数据类型，并尽可能地减少使用include文件，从而最大程度地简化结构体定义。记住，在IDA中创建结构体时，正确布局最为重要。正确的布局更多地取决于每个字段的正确大小和结构体的正确对齐，而不只是对每个字段都使用正确的类型。

IDA TIL文件（Type Library）

IDA中的所有数据类型和函数原型信息都存储在TIL文件中。IDA拥有存储在<IDADIR>/til目录中的许多主要编译器和API的类型库信息。Types窗口（View►Open subview►Type Libraries）列出了当前加载的.til文件，并可用于加载你想要使用的其他.til文件。IDA将根据在分析阶段发现的二进制文件属性，自动加载类型库。**正常情况下，一般不需要直接处理.til文件。**

IDA还利用.til文件存储你在Structures窗口中手动创建的或者通过解析C头文件获得的任何自定义结构体定义。这些结构体存储在一个与创建它们的数据库有关的专用.til文件中。该文件的名称与其相关数据库的名称相同，扩展名为.til。

例如，如果数据库名为some_file.idb，则相应的类型库文件则为some_file.til。

.idb文件实际上是一个归档文件（类似于.tar文件），用于保存不使用的数据库组件。打开一个数据库时，其组件文件（.til文件为其中之一）将被提取出来，成为IDA中的运行文件。

在其他数据库中共享til文件

有两种共享方法。第一种方法有些不太正规，即将．til文件由打开的数据库复制到另一个目录中，然后再通过Types窗口，在任何其他数据库中打开这个．til文件。第二种是一种正式的方法，即从一个数据库中提取出自定义类型信息，生成一段IDC脚本，用于在任何其他数据库中重建自定义结构体。使用File▶Produce File▶Dump Type. into to IDC File（文件▶生成文件▶转储类型信息到IDC文件）命令可生成该脚本。但是，与第一种方法不同的是，这种方法只能转储Structures窗口中列出的结构体，但并不转储通过解析C头文件得到的结构体（而复制．til文件却可以转储这类结构体）。