**0 前言**

作为Windows程序员，对PDB文件肯定不陌生，不管是平时的本机调试还是现网Crash的Dump文件分析，如有PDB文件的辅助，可以大大提高调试效率。

本文主要从PDB的文件格式，PDB的生成与存储以及PDB的使用几个方面介绍PDB的相关知识。

**1 PDB简介**

PDB(Program DataBase)是Vsiual Studio构建工程时生成的用于调试的符号文件。主要包含了如下信息：

* 全局变量
* 局部变量
* 函数名和它们的入口点地址
* 帧指针省略记录[FPO(Frame Pointer Omission)](https://link.jianshu.com?t=https:/blogs.msdn.microsoft.com/larryosterman/2007/03/12/fpo/)
* 源码行号

下面是与PDB相关的几个VS工程配置选项：

* 指定调试信息格式，设置为[/Zi或/ZI](https://link.jianshu.com?t=https:/msdn.microsoft.com/en-us/library/958x11bc.aspx)时会生成PDB文件：  
  Project -> Properites -> Configuration Properties -> C/C++ -> General -> Debug Information Format

调试信息格式设置

* 指定编译器(Compiler)生成的PDB文件名，默认为VCx0.pdb，其中x代表当前VS的版本，例如VS2010默认PDB名为VC100.pdb：  
  Project -> Properites -> Configuration Properties -> C/C++ -> Output Files -> Program Database File Name

编译器生成PDB文件名设置

* 指定连接器(Linker)是否生成调试信息，YES为生成，NO为不生成。  
  Project -> Properites -> Configuration Properties -> Linker -> Debugging -> Generate Debug Info
* 指定连接器(Linker)生成的PDB文件名，默认为TargetName.pdb，其中TargetName为连接后生成的文件名。  
  Project -> Properites -> Configuration Properties -> Linker -> Debugging -> Generate Program Database File

链接器生成PDB文件名设置

**1.1 PE文件格式**

连接器在连接时将PDB文件名及路径信息存放到可执行文件的"CODEVIEW"调试目录中。该目录包含如下PDB相关信息：

| **偏移地址** | **数据类型** | **注释** |
| --- | --- | --- |
| +0h | dword | "RSDS"签名 |
| +4h | GUID | 16字节的[GUID](https://link.jianshu.com?t=https:/msdn.microsoft.com/en-us/library/system.guid(v=vs.110).aspx) |
| +14h | dword | "age" |
| +18h | byte string | 以0结尾的UTF-8格式文件路径和文件名 |

下面我们就通过[PEView](https://link.jianshu.com?t=https:/www.aldeid.com/wiki/PEView)来查看一个PE文件。

PE CODEVIEW信息

上图中对应的数据信息如下所示：

| **偏移地址** | **数据类型** | **注释** |
| --- | --- | --- |
| +0h | dword | RSDS |
| +4h | GUID | E6AC12193E049A47AD6BD95644534850 |
| +14h | dword | 1 |
| +18h | byte string | d:\Test\MyTest\Output\MyTest\MyTest.pdb |

**1.2 PDB文件格式**

PDB文件格式并未公开，但是Microsoft提供了API来读取PDB中的数据，可以参考[CCI](https://link.jianshu.com?t=http:/ccimetadata.codeplex.com/)开源项目。

虽然官方未公开格式，但是[The RSDS pdb format](https://link.jianshu.com?t=http:/www.godevtool.com/Other/pdb.htm)对PDB的文件格式做了较详细的介绍。

PDB的文件格式类似于磁盘的文件系统，每个磁盘会被划分成很多个大小一样的扇区，文件中的数据就存放在不同的扇区中，而且无需保证这些扇区在磁盘上是连续的。PDB文件用page进行划分，类似于扇区，stream就类似于文件，stream directory类似于文件目录。

下面我们就用winhex来看下PDB中到底存放了那些信息吧。

**1.2.1 PDB头部结构**

PDB头部结构信息

上图中PDB头部信息解析如下：

| **偏移地址** | **数据类型** | **数据** | **备注** |
| --- | --- | --- | --- |
| +0h | byte string | Microsoft C/C++ MSF 7.00 | PDB版本信息,不同版本字符串长度不一样 |
| +1Ah | byte | EOF | 标志PDB版本信息字符串结束 |
| +1Bh | byte string | DS | 签名 |
| +1Eh | byte | null-terminator | 终结符 |
| +20h | dword | 00000400h(1024) | page的大小(单位:字节) |
| +24h | dword | 00000002h(2) | 未知 |
| +28h | dword | 00000293h(659) | 整个文件有占用多少个page |
| +2ch | dword | 00000AE0h(2784) | stream directory占用大小(单位:字节) |
| +30h | dword | 00000000h(0) | 未知 |
| +34h | dword | 00000291h(657) | stream directory指针的页地址(单位:页) |

**1.2.2 PDB Stream Directory Pointers信息结构**

根据上述信息可以计算出stream directory指针地址为A4400h，对应信息如下：

PDB Stream Directory Pointers信息

从图中可以看出存放了三个stream的页地址指针，分别为028Eh, 028Fh, 0290h，之所以是三个是因为stream directory占用了0AE0(2784)字节，需要三个page才能存放。

**注意:这里的指针并不需要是连续的，只是这里刚好是连续的而已。**

**1.2.3 PDB Stream Directory信息结构**

Stream Directory的数据结构如下所示：

| **偏移地址** | **数据类型** | **注释** |
| --- | --- | --- |
| +0h | dword | stream的个数 |
| +4h | dword | stream大小(单位:字节)，忽略0和-1(FFFFFFFF) |
| +?h | dword | stream的指针数组 |

以028Eh为例，乘以400h之后得到A3800h，该地址对应信息如下：

PDB\_STREAM\_DIRECTORY.png

从上图可以看出该stream directory中存在002Fh(47)个stream，而接下来的47个dword分别表示每个stream的大小。因此page指针将从A38C0h处开始。

前3个stream信息解析如下：

| **Streams** | **size** | **page pointers** |
| --- | --- | --- |
| Stream1 | 0000001Ch | 00000004h |
| Stream2 | 00000072h | 0000028Ch |
| Stream3 | 00050FD0h | 0000028Dh,00000134h,... |

**1.2.4 PDB Stream信息结构**

不同的stream包含不同信息，这里主要介绍下存放PDB文件真实性相关的信息。一般Stream2包含了这些信息。通过计算可以得到Stream2在文件中的地址为A3000h,对应地址的信息如下：

PDB文件真实性信息

红框中记录的就是前面通过PEView工具查看CODEVIEW中的GUID，在它前面的dword就是对应的age字段。

**2 PDB的生成与存储**

一般在本机编译工程的时候，PDB文件会在本机指定的目录生成，如果要到其它计算机上调试，需要同时拷贝二进制文件和PDB文件。

而对于发布版本，我们一般都会在编译机上编译，并且将生成的PDB文件保存到符号服务器，这样不管在哪台计算机上都可以通过配置符号服务器路径来拉取到对应的PDB文件。

那么PDB文件在符号服务器是如何存储的呢？调试工具又是如何定位到对应的PDB文件的呢？答案就是[SysStroe](https://link.jianshu.com?t=https:/msdn.microsoft.com/zh-cn/library/windows/desktop/ms681417(v=vs.85).aspx)。

下面我们来详细介绍下整个过程。

**2.1 SysStore**

SymStore(symstore.exe)是用于创建符号存储的工具。包含在[Windows调试工具包](https://link.jianshu.com?t=https:/developer.microsoft.com/en-us/windows/hardware/windows-driver-kit)中。

SymStore按照某种格式存储符号，使调试器可以通过时间戳、映像大小(对于.dbg或可执行文件)、签名和寿命(.pdb文件)来查找符号。 使用符号存储而不是常规的符号存储格式的好处是，所有符号都可以在同一个服务器上进行存储或引用，而调试器不需要知道具有哪些产品对应的符号。

**注意，不同版本的.pdb符号文件(例如公有和私有符号)不能保存在相同的符号服务器中，因为他们具有相同的签名和寿命。**

**2.1.1 SymStore事务**

* SymStore的每次调用都被记录为事务。事务分两种：添加和删除。
* 当创建符号存储时，会在服务器根目录下创建一个"000admin"目录。它为每个事务都包含一个文件以及作为日志记录的server.txt和history.txt。
  + server.txt包含当前服务器上的所有事务的列表。
  + history.txt包含所有历史事务，按时间排序。
* 每次添加或删除符号文件都会创建一个新的事务号，然后在000admin中创建一个以事务号为名字的文件。文件中包含所有在该事务中添加到符号存储服务器的文件或文件指针的列表。如果某个事务被删除，SymStore会读取它的事务文件来决定哪些文件和指针需要删除。
* add和del选项用来指定是否进行事务添加或删除。在添加操作中包含/p选项表示添加一个指针，忽略/p指定添加实际的符号文件。
* 也可以分两步来创建符号存储。
  + 使用SymStore和/x选项来创建一个索引文件。
  + 使用SymStore 和/y选项通过索引文件中的信息来创建真正的存储文件或指针。
* 这是一种很有用的技术。比如：
  + 方便符号存储由于某些原因丢失后的重建，只要索引文件还存在。
  + 当包含符号文件的计算机和创建符号存储的计算机之间网络连接速度很慢时。可以在符号文件所在机器上创建索引文件，并将该文件传输到第二台机器，在第二台机器上创建符号存储。
* SymStore参数的完整列表，请参阅[SymStore命令行选项](https://link.jianshu.com?t=https:/msdn.microsoft.com/zh-cn/library/windows/desktop/ms681378(v=vs.85).aspx)。

**注意: SymStore不支持多个用户同时发起事务。建议指派一个用户作为符号存储的"管理员"并处理所有的add和del事务。**

**2.1.2 事务示例**

1. 在\sampledir\symsrv中添加Windows Server 2003 build 3790符号指针。

作者：小猪啊呜  
链接：https://www.jianshu.com/p/7ad20a047bb4  
来源：简书  
著作权归作者所有。商业转载请联系作者获得授权，非商业转载请注明出处。