

[PE文件结构学习]1.相对虚拟地址(RVA)与物理地址的转换

原创

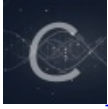
碎炎 于 2012-09-06 15:40:41 发布 6690 收藏 4

分类专栏: [win32/mfc](#) 文章标签: [dos exe table header file 存储](#)

版权声明: 本文为博主原创文章, 遵循 [CC 4.0 BY-SA](#) 版权协议, 转载请附上原文出处链接和本声明。

本文链接: <https://blog.csdn.net/sryan/article/details/7950950>

版权



[win32/mfc](#) 专栏收录该内容

6 篇文章 0 订阅

订阅专栏

正在学习PE文件结构, 本人小菜鸟一个, 如有不正确, 欢迎指正。

PE结构即可执行文件的硬盘存储结构, Windows系统下面有exe dll等格式, PE结构先不介绍了, 网上资料很多, 看雪的文库里面的资料很好。

首先以我个人的理解来解释一下各种概念。

1.VA VA即virtual address。解释为虚拟地址, 它是经过PE载入器重定位后的在该进程地址空间中能访问到的地址, 在调试时候访问一些全局变量, 变量的地址即虚拟地址。

2.RVA RVA即Relative virtual address。它是一个虚拟地址, 凡是牵扯到虚拟地址的, 都是在PE文件被PE载入器映射入内存后的地址, 即与内存有关, 与在磁盘文件中的存储无关。RVA主要是减轻PE载入器的工作量和方便计算VA而存在的, PE物理文件中的地址基本全都是RVA, RVA是一个偏移地址, 是相对于ImageBase的偏移, 假如代码段内访问一个RVA为0x00000111的内存单元, 那该内存单元的VA即为(实际的)ImageBase + 0x00000111。

3.ImageBase ImageBase为映像的基址, PE文件在映射入内存的时候, 会载入一个特定的基址, PE文件头中的ImageBase是推荐映射入内存的地址, 一般为0x00400000, 所有的RVA再根据ImageBase来转换成VA进行正确的内存访问行为。

4.物理地址 Physical Address PA 物理地址即该字节在文件中的位置, 当然是相对于文件头的偏移位置。

在实际情况下, 我们常常会遇到将代码段中访问的RVA转换成PA, 在这种情况下需要读取文件头来做相应的转换。

转换的一般步骤为:

1.将exe文件映射入内存中，读取Dos MZ Header，在这个结构中，我们能够通过e_lfanew来获取NT文件头相对于Dos文件头的偏移。

2.获取到了NT文件头的地址，NT文件头中包含了两个文件头，一个是FILE文件头，一个是Optional可选文件头，在FILE文件头中我们可以读取到段的数量，在转换RVA地址的时候，我们只需要得到这个数量。

3.将NT文件头之后紧跟着的就是SECTION TABLE，这是段描述头，在这个段描述头中，我们几乎可以获取到全部的段的资料。

VirtualAddress: 这个是一个RVA地址，代表的意义是告诉PE加载器该段存在于RVA地址为VirtualAddress处

PointerToRawData: 这是一个物理偏移地址，告诉PE加载器将物理文件偏移PointerToRawData处的数据映射入VirtualAddress处

VirtualSize: 该段的大小

SizeOfRawData: 该段的物理大小。由于存在对齐问题，所以该大小为不小于VirtualSize大小的对齐大小的整数倍。

4.通过遍历SECTION TABLE，判断要转换的RVA地址是否处于所有段的RVA地址范围内。该段的RVA范围为: VirtualAddress + SizeOfRawData，当然更精确点是VirtualAddress + VirtualSize。

5.假如存在于地址范围内，那么我们可以通过要转换的RVA地址减去该段的VirtualAddress算出相对于该段的偏移量

6.将偏移量加上PointerToRawData，即可算出物理地址