2021/02/04_PE_第4课_可执行文件内存的Dump

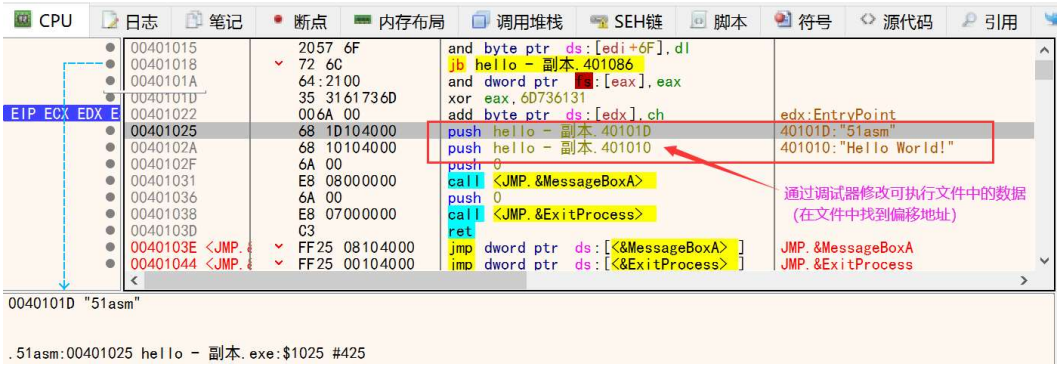
笔记本： PE
创建时间： 2021/2/4 星期四 10:08
作者： ileemi

- 内存 dump
- 防止内存dump

添加节时需要注意镜像值的大小（所有节的大小+PE头的大小，需要注意对齐值）。

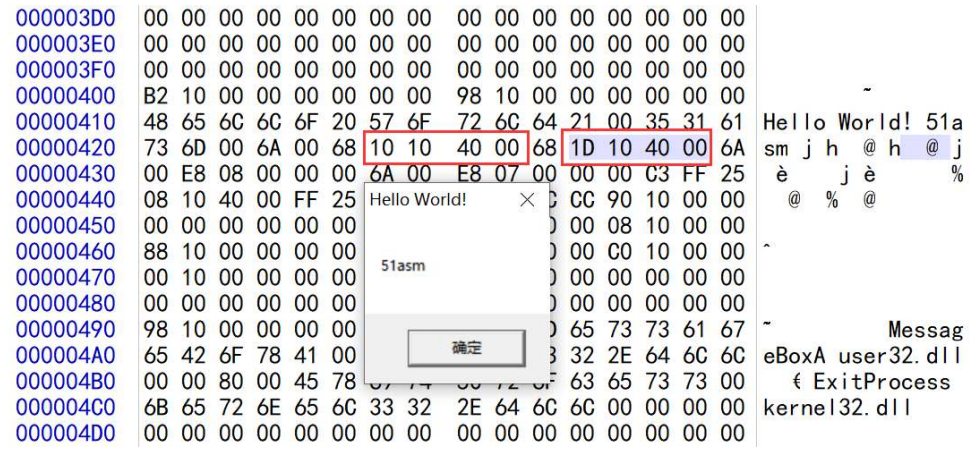
> FA != RVA，主要原因就是文件对齐值和内存对齐值不一样。

VA --> RVA --> FA
FA --> RVA --> VA



- VA == 0x0040102A
- ImageBase == 0x00401000（不是随机基址）
- RVA = 0x0040102A - 0x00401000 = 102A（命中下面的第一个节）
- 遍历节表：
  - PE头：0000~1000（内存地址），0000~0400（文件偏移）
  - 第一个节：1000~2000（内存地址），0400~0600（文件偏移）
  - 第二个节：2000~3000（内存地址），0600~0800（文件偏移）
  - 第三个节：3000~4000（内存地址），0800~1000（文件偏移）
  - ...
- 102A（命中下面的第一个节），102A-1000 = 2A，0400+2A = 042A，在可执行文件中找到偏移 "042A" 的位置，修改目标数据保存即可，类似调试器 "打补丁" 功能（当通过调试器修改内存数据后，使用打补丁功能，但是在文件中没

有找到修改的目标偏移位置，就会导致打补丁失败），操作如下：



> CFF.exe 地址转换是通过访问文件数据进行转换的，不是通过内存访问数据（涉及到随机基址）。

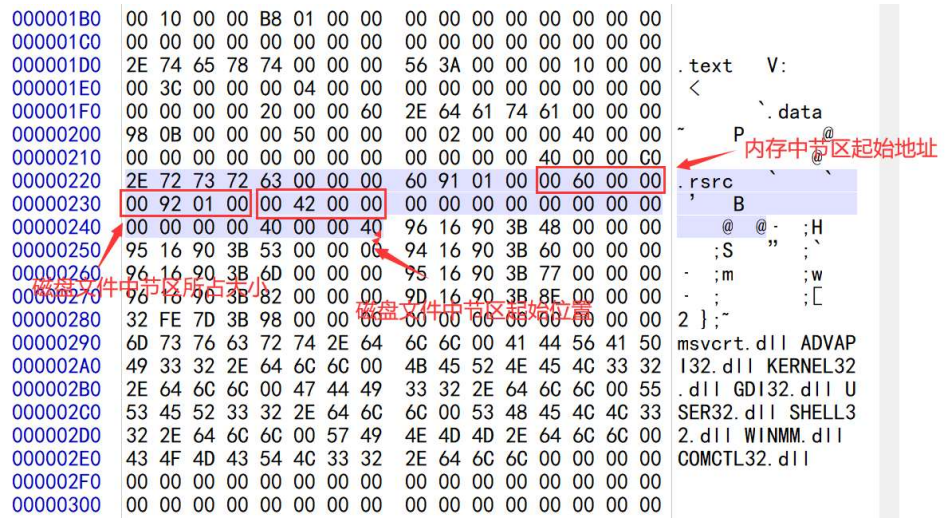
可执行文件的代码进行加密了加密（此时对该可执行文件进行静态反汇编没有用），当软件运行起来的时候对加密代码进行解密，使用调试器等待程序运行起来后，通过从内存中的节表重建PE格式，生成新的可执行文件（此时源程序的加密已经不存在），这种实现方法就叫：内存 dump

静态分析反汇编比动态分析反汇编要快，从文件中静态反汇编还原代码效率更高，调试程序适合分析程序的运行流程（定位程序关键的代码），不适合还原代码。

内存 dump

> 手动实现dump

- 运行目标程序
- 通过WinHex加载目标进程内存数据
- dump PE头
- dump 节区（根据磁盘文件中节区所占大小从源程序中拷贝对应的字节数到dump程序中区）



> 需要注意：从内存中dump目标进程的数据时，需要注意全局变量的问题，如果dump下来的全局变量是目标程序修改后的数据，就会出现一些问题。

> 目标程序代码区的加密问题，dump下来的目标程序节区数据是包含目标程序的加密代码的，需要修复OEP（跳过目标程序的解密代码）

防止内存dump

> 当程序运行起来的时候，对内存中的PE数据进行破坏（修改内存中节区所占的大小数据）。