aspack 壳：

一、特征：

ASPack 壳的典型特征包括：**入口点的转移**、**额外节的创建**、**内存解压**、**常用的系统 API 调用**以及 **PUSHAD/POPAD 的使用**。此外，ASPack 壳常常使用**反调试技术**（检测调试器的存在），并且通过压缩 .text 节，动态解压并跳转到 OEP。

1、PUSHAD/POPAD 指令：ESP 定律
2、ASPack 壳在加壳时，会对导入表（Import Address Table, IAT）进行重定向和隐藏。
3、额外节的创建：.adata 或 .aspack

二、工具：win10 虚拟机、Ollydbg+ollydump 插件

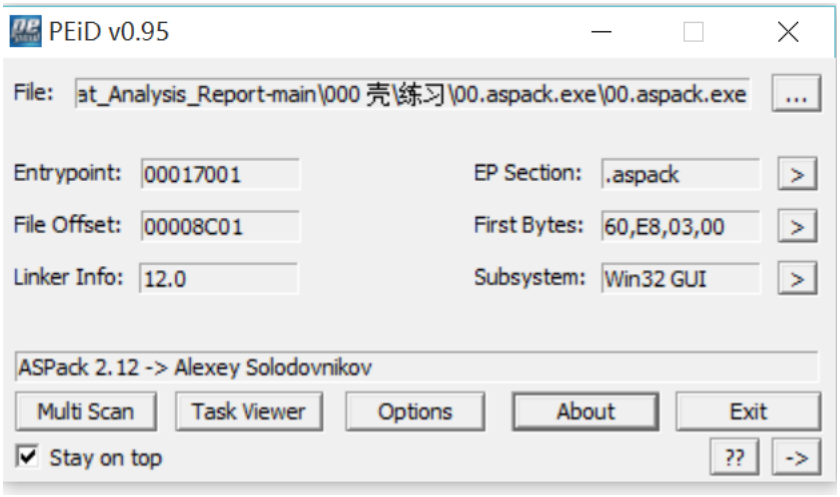
二、总结：

1、程序允许重定位，导致 dump 的出错更无法重建导入表。

2、ESP 定律：可以通过在 PUSHAD/POPAD 时对栈指针（ESP）的变化来追踪 OEP。
加壳程序运行，解压缩或解密前会执行 PUSHAD 将 EAX~EDI 寄存器值存于栈中，待解压完成后又会执行 POPAD 将前面所存的值全部弹回。可以根据所保存的 ESP 值（PUSHAD 执行后，对 ESP 值设置硬件访问断点，当 POPAD 执行时程序会访问先前保存的 ESP 值，从而触发中断，这通常发生在接近程序的 OEP 处。）来追踪寻找 OEP。---ESP 值恢复到脱壳前定律

四、脱壳示例：

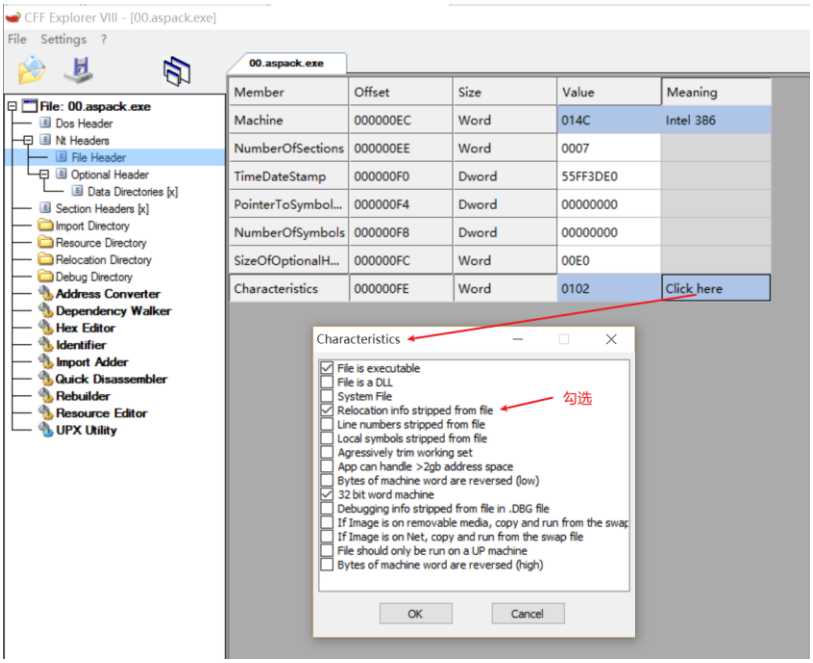
检查带脱壳文件信息：ASPack2.12 壳。



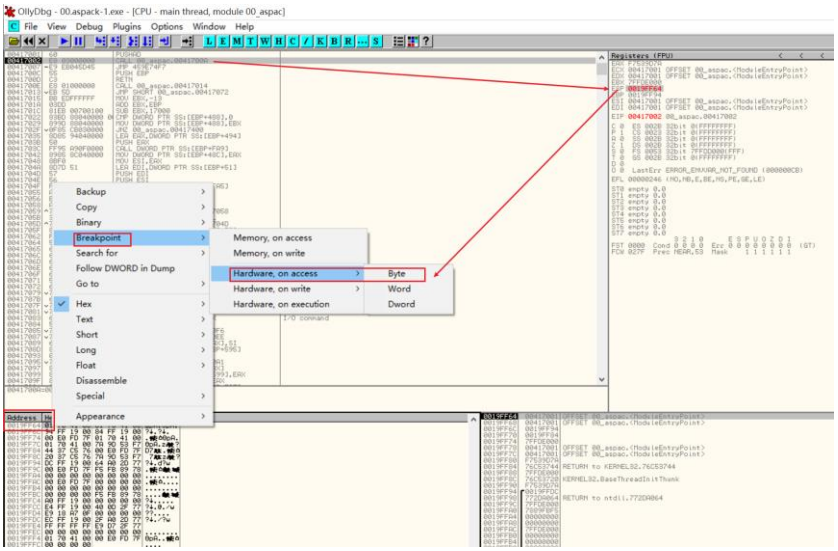
OllyDbg 调试：

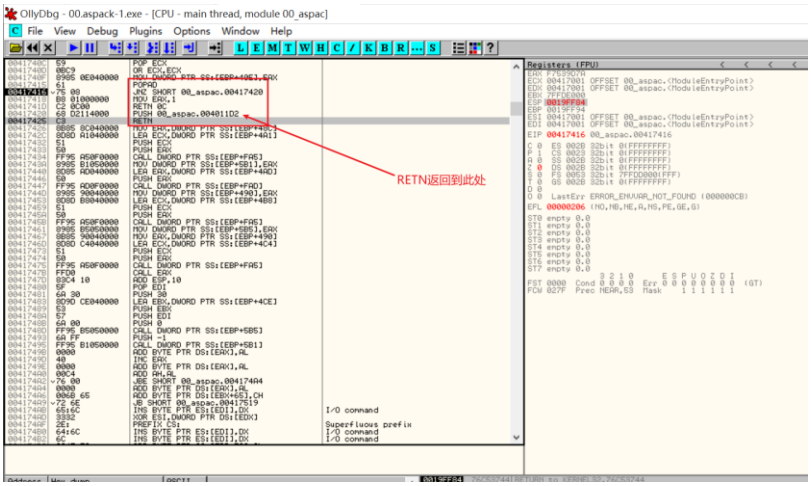
不允许重定位：

勾选（Relocation info stripped from file）表示重定位信息已被移除。也就是说，PE 文件在加载时不能进行重定位，必须加载到一个固定的内存地址。

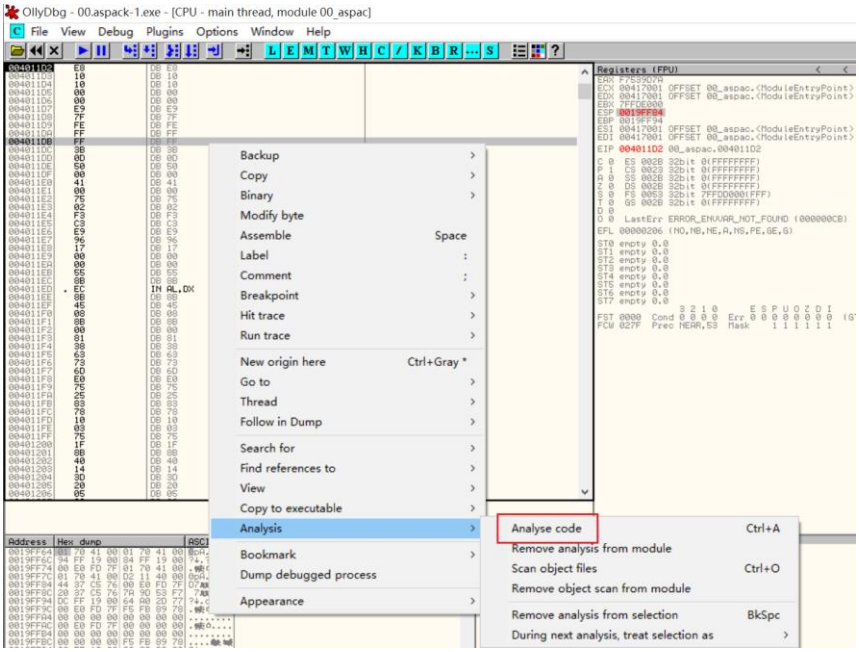


运行 PUSHAD 后根据 ESP 在内存窗口设置硬件访问断点：ESP 定律


OllyDbg - 00.aspack-1.exe - [CPU - main thread, module 00_aspac]
File View Debug Plugins Options Window Help
Backup
Copy
Binary
Breakpoint
Search for
Follow DWORD in Dump
Go to
Hex
Text
Short
Long
Float
Disassemble
Special
Appearance
Memory, on access
Memory, on write
Hardware, on access
Hardware, on write
Hardware, on execution
Byte
Word
Dword

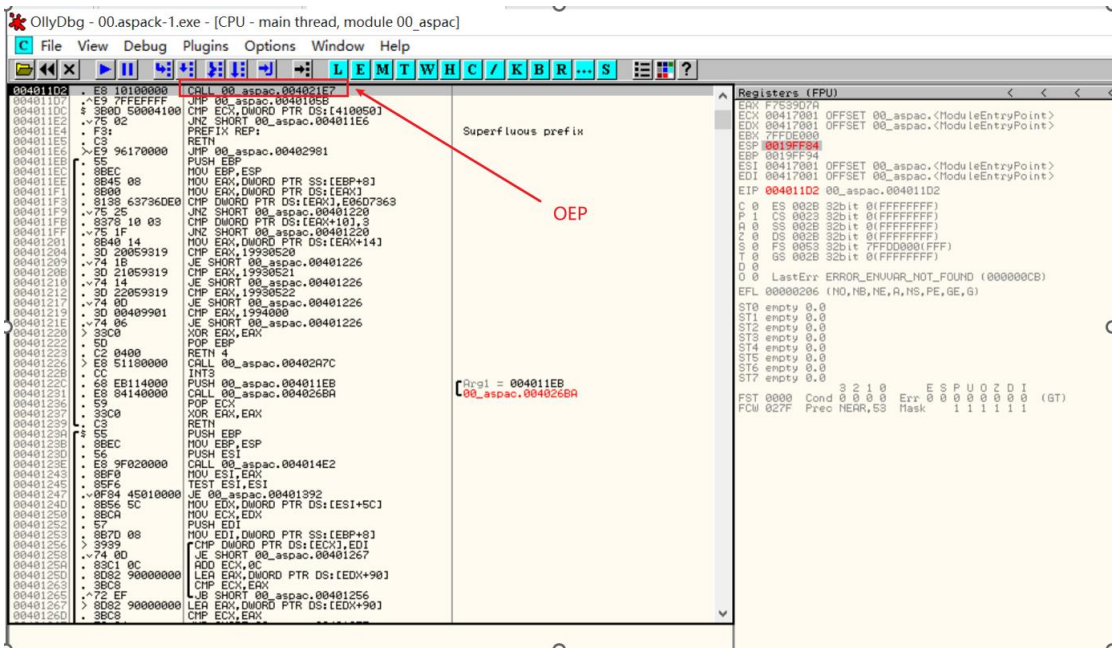

OllyDbg - 00.aspack-1.exe - [CPU - main thread, module 00_aspac]
File View Debug Plugins Options Window Help
RETN返回到此处
I/O command
Superfluous prefix
I/O command
I/O command

手动分析:


OllyDbg - 00.aspack-1.exe - [CPU - main thread, module 00_aspac]
File View Debug Plugins Options Window Help
Backup
Copy
Binary
Modify byte
Assemble Space
Label :
Comment ;
Breakpoint
Hit trace
Run trace
New origin here Ctrl+Gray *
Go to
Thread
Follow in Dump
Search for
Find references to
View
Copy to executable
Analysis
Bookmark
Dump debugged process
Appearance
Analyse code Ctrl+A
Remove analysis from module
Scan object files Ctrl+O
Remove object scan from module
Remove analysis from selection BkSpc
During next analysis, treat selection as

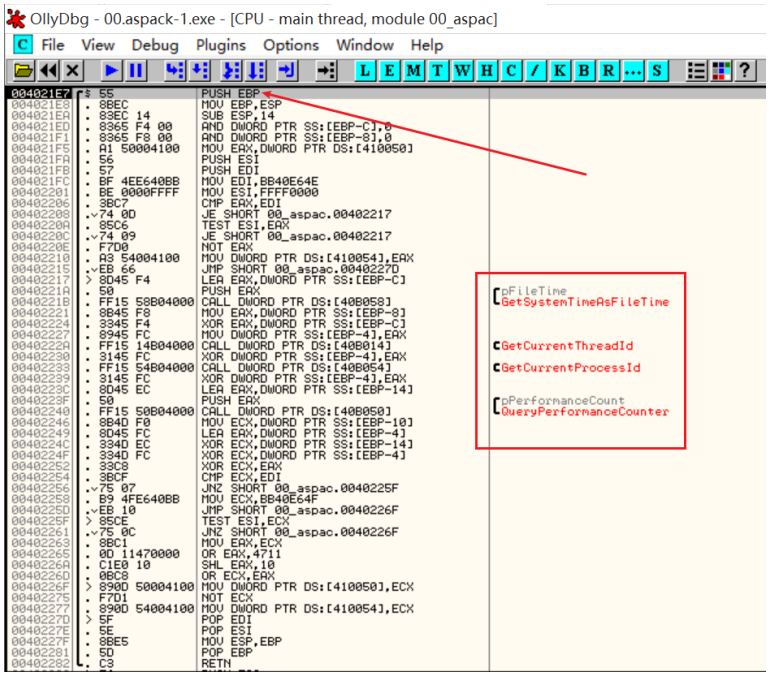
找 OEP：

F9 运行到 POPAD 附近：RETN 返回一个很远地址，相当于 jmp 大跳。设想如下地址为 OEP，原因下面解释。

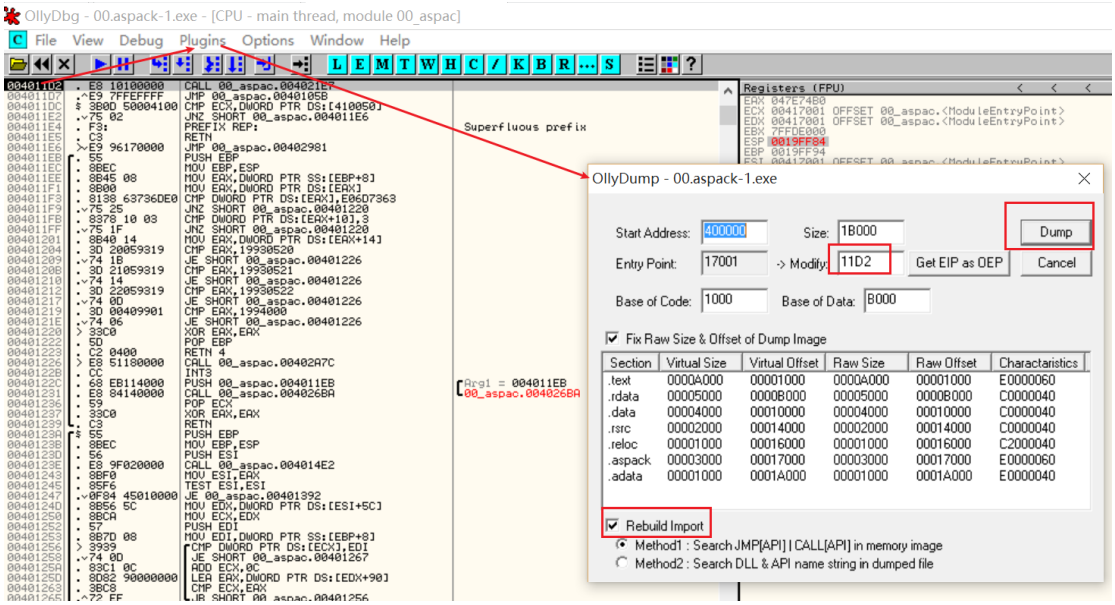


进入 call 函数分析，确实是程序开始部分：

OEP 附近通常会看到初始化相关的 API 调用，尤其是在**加壳程序**中，通过调用一些系统 API（如时间、进程、线程相关的 API）进行环境检查，然后进入程序的实际业务逻辑。由此确定 OEP。



Dump 文件：勾选重建导入表



脱壳成功：PEID 检测无壳。