# aspack壳：

## 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

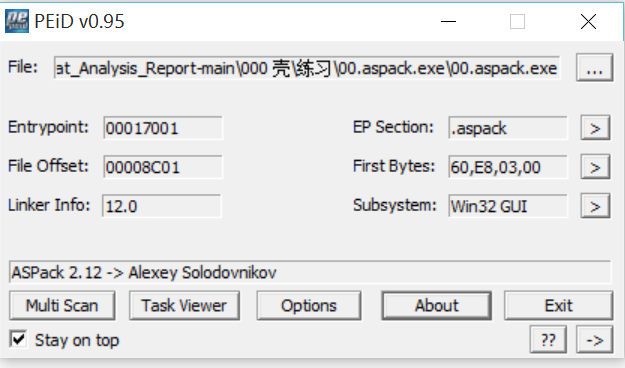
## 总结：

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：

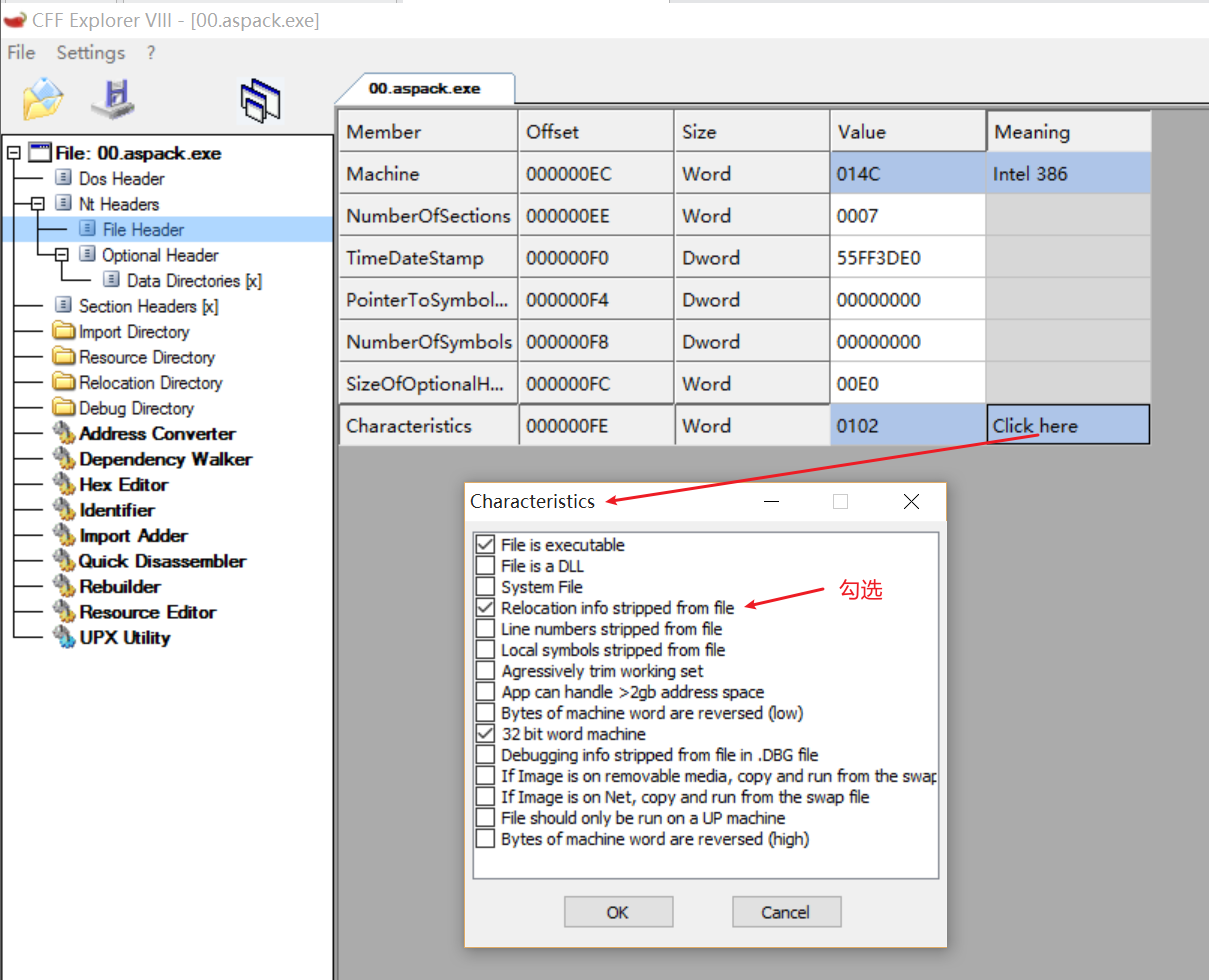
### 检查带脱壳文件信息：ASPack2.12壳。



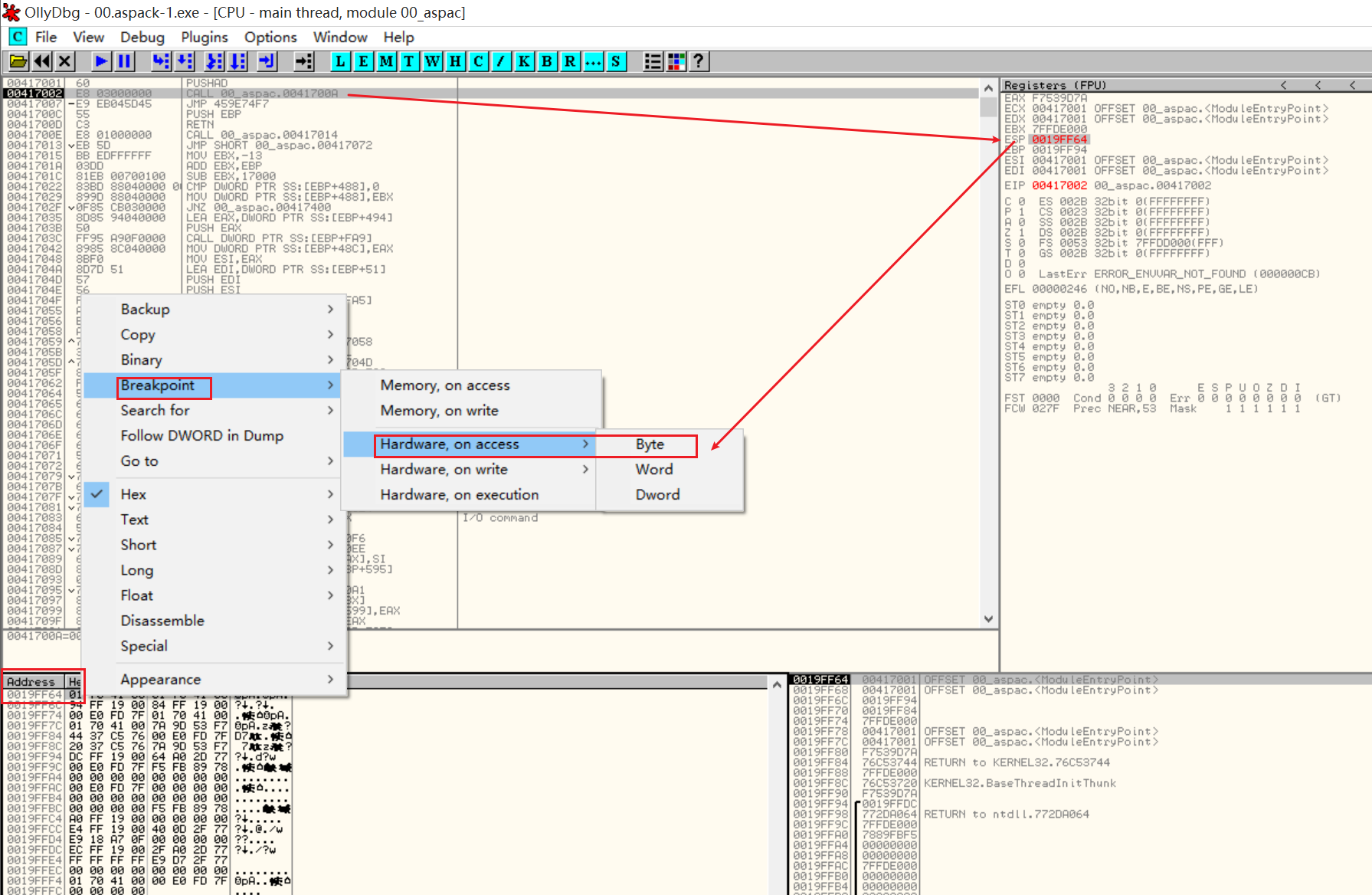
### OllyDbg调试：

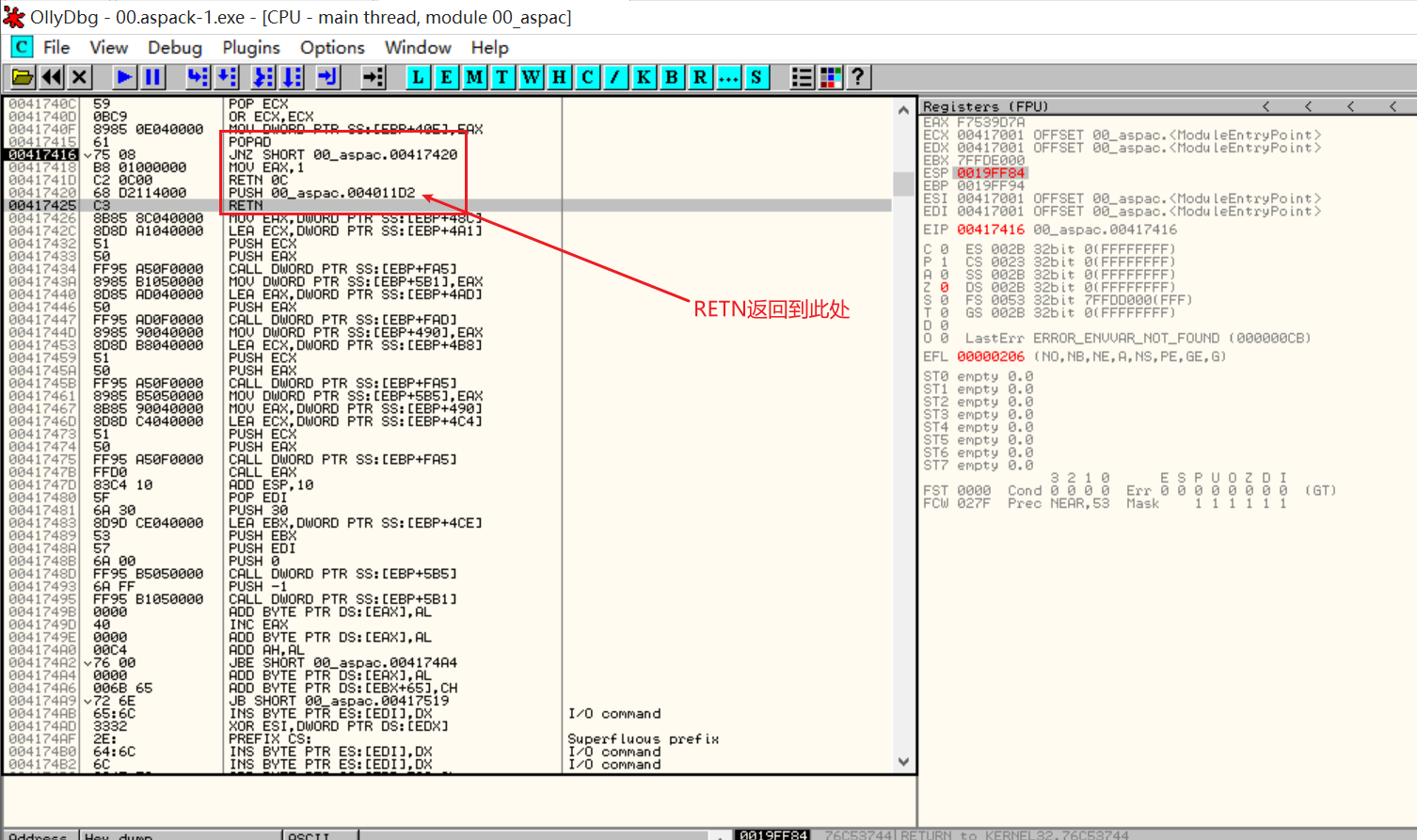
不允许重定位：

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。

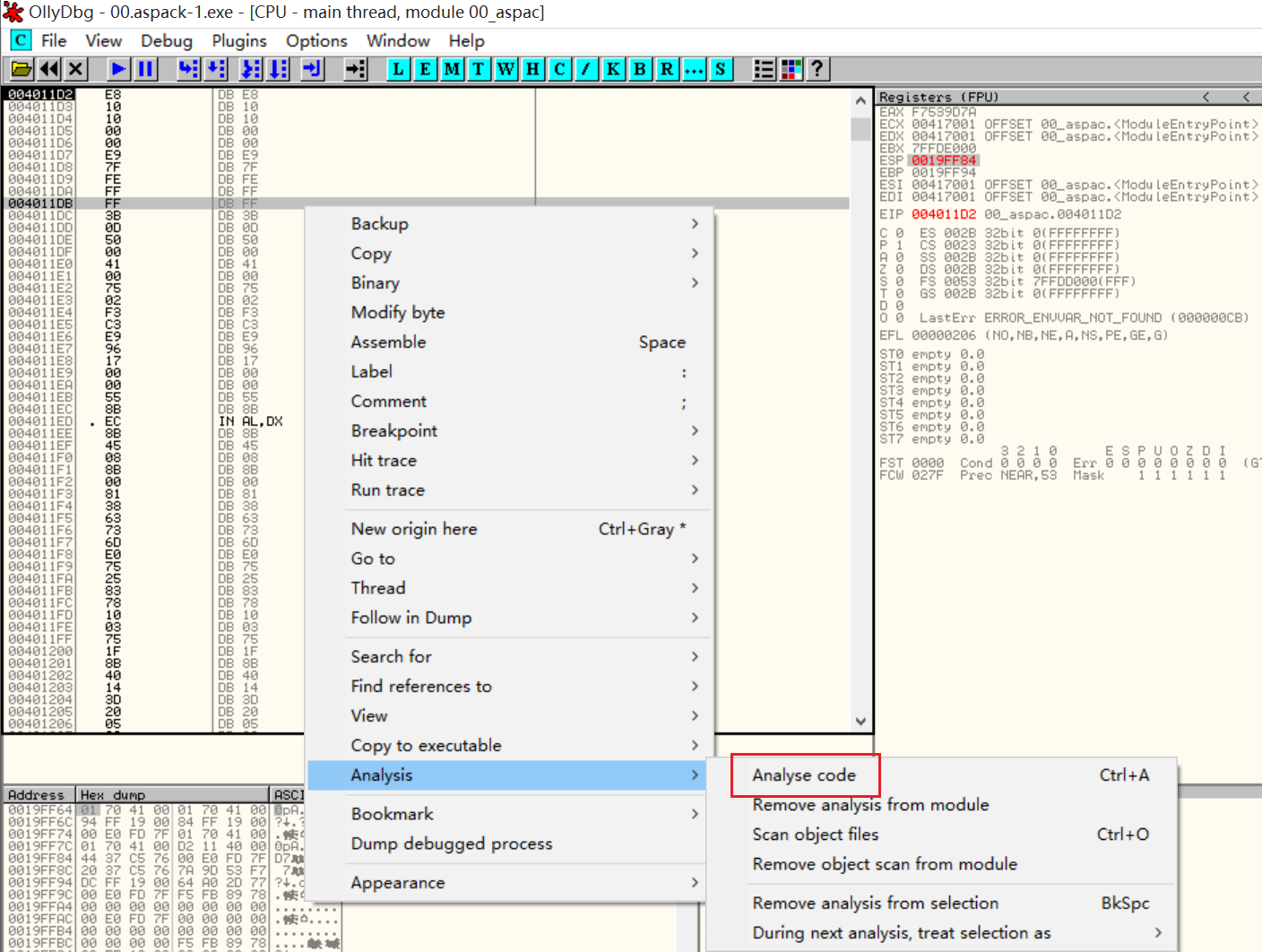


运行PUSHAD后根据ESP在内存窗口设置硬件访问断点：ESP定律



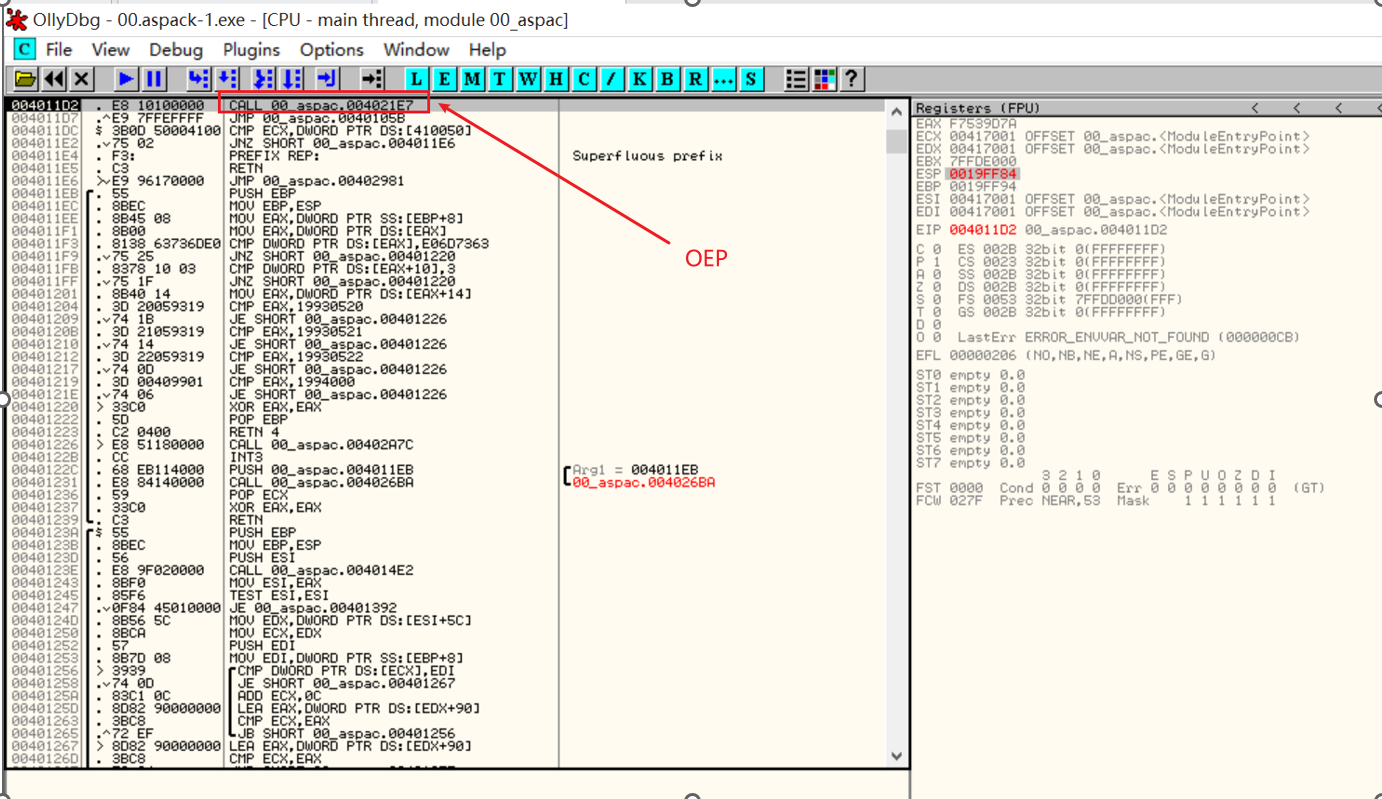


手动分析：



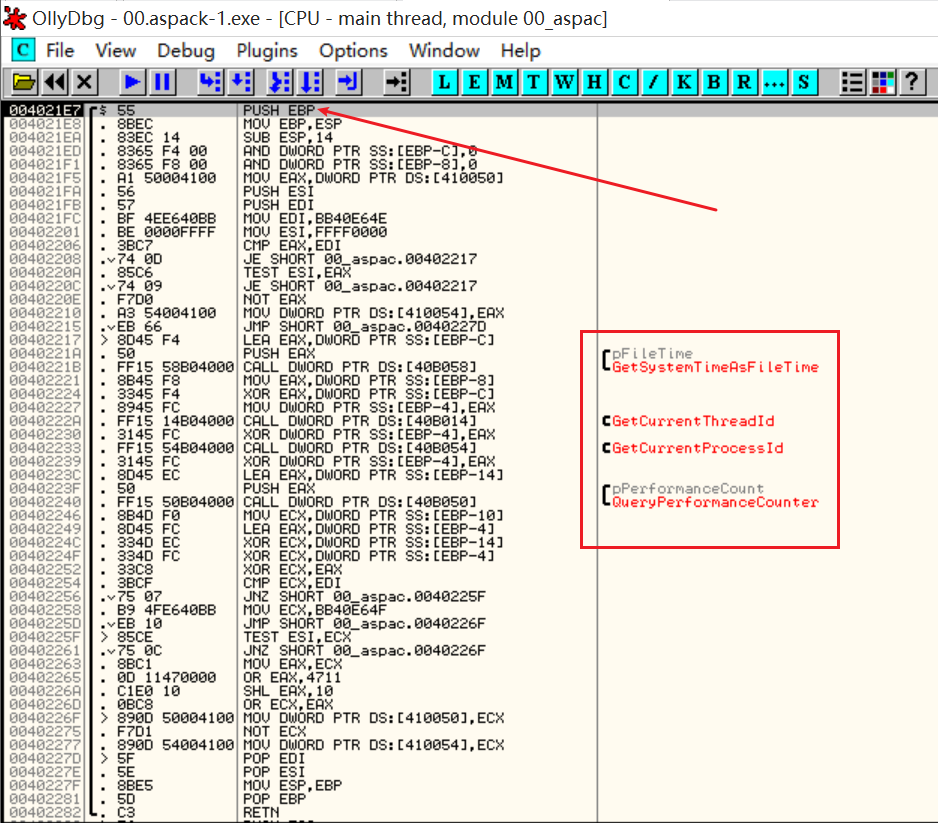
### 找OEP：

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。

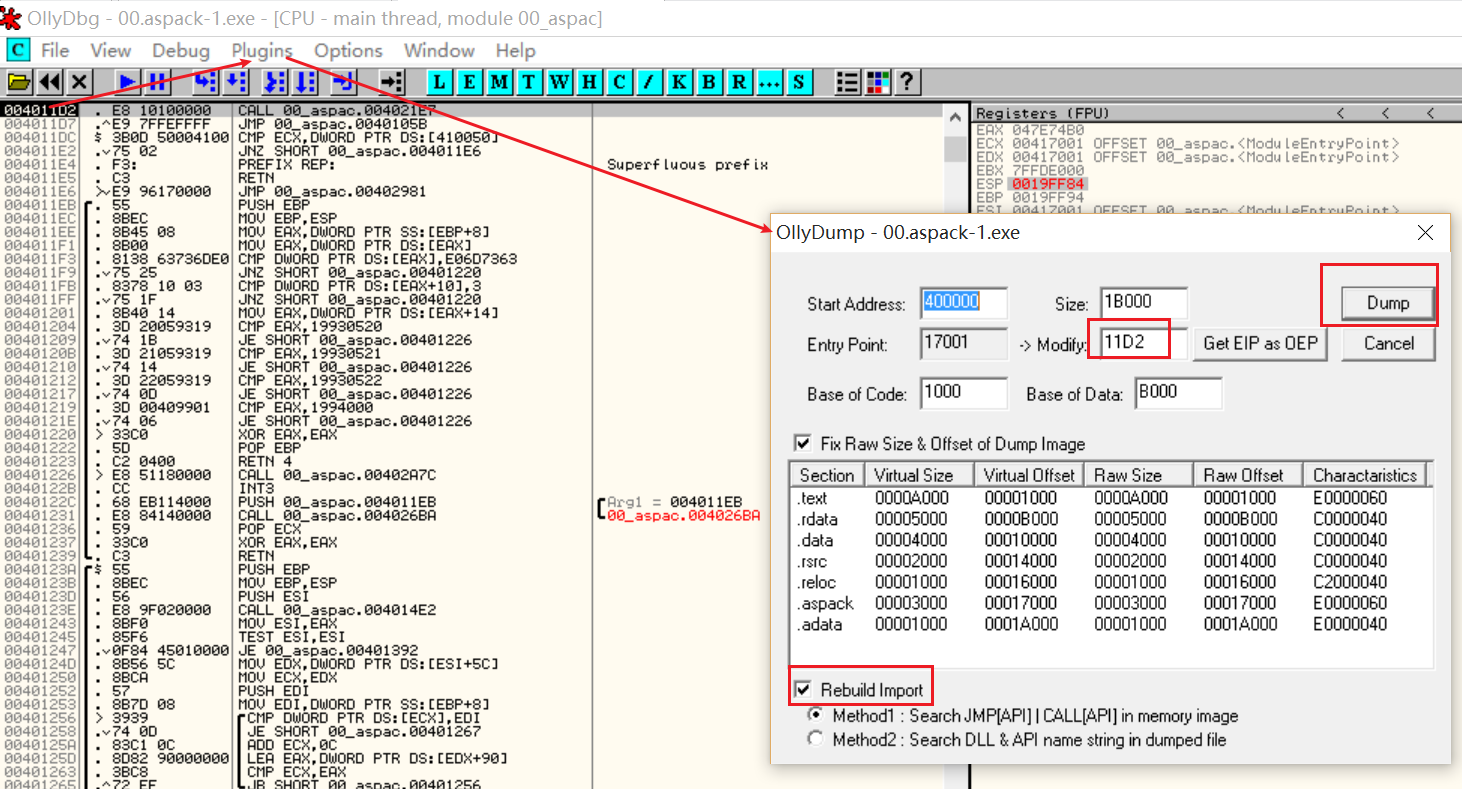


进入call函数分析，确实是程序开始部分：

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。

