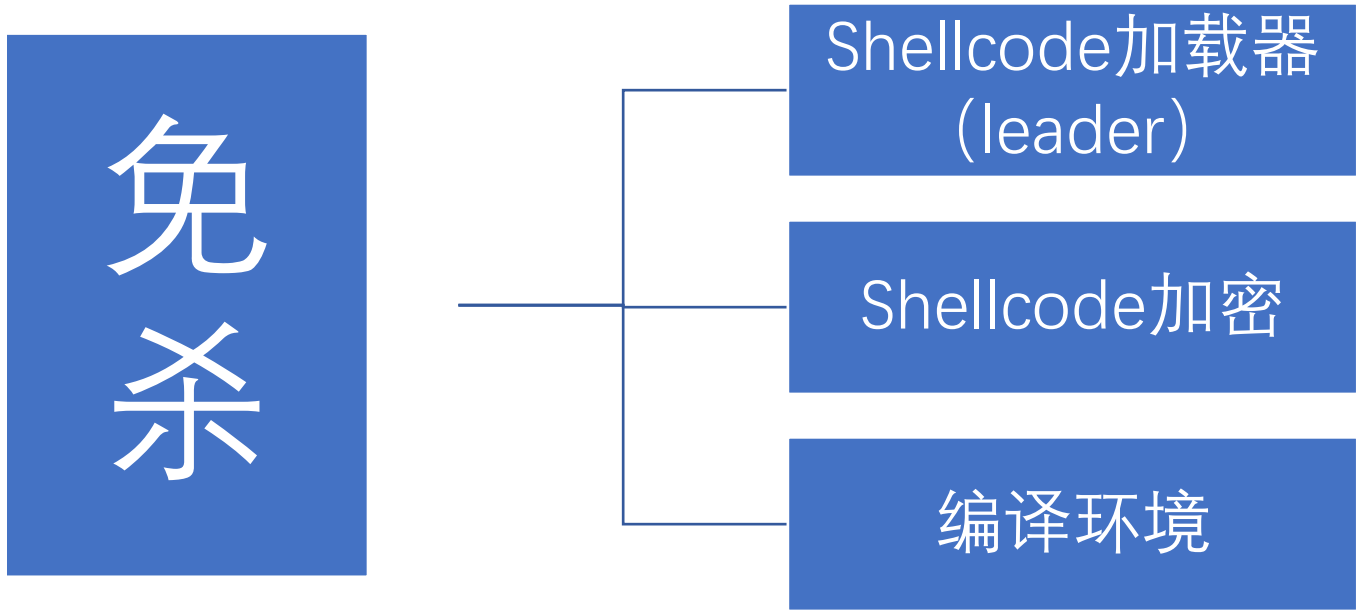
突破主动防御

什么是主动防御

- 主动防御简单来讲，就是一种针对防火墙、注册表、服务、进程、流量等进行监控以及对异常进行拦截的行为。

突破主动防御（免杀）



Shellcode加载器（leader）

普通加载器：申请动态内存，汇编加载，汇编花指令等。

高级加载器：管道连接（进程间通讯）等。

Shellcode加密

普通加密：base64，xor，AES等。

自定义加密：计算内存偏移，注意32位和64位的指针。

编译环境

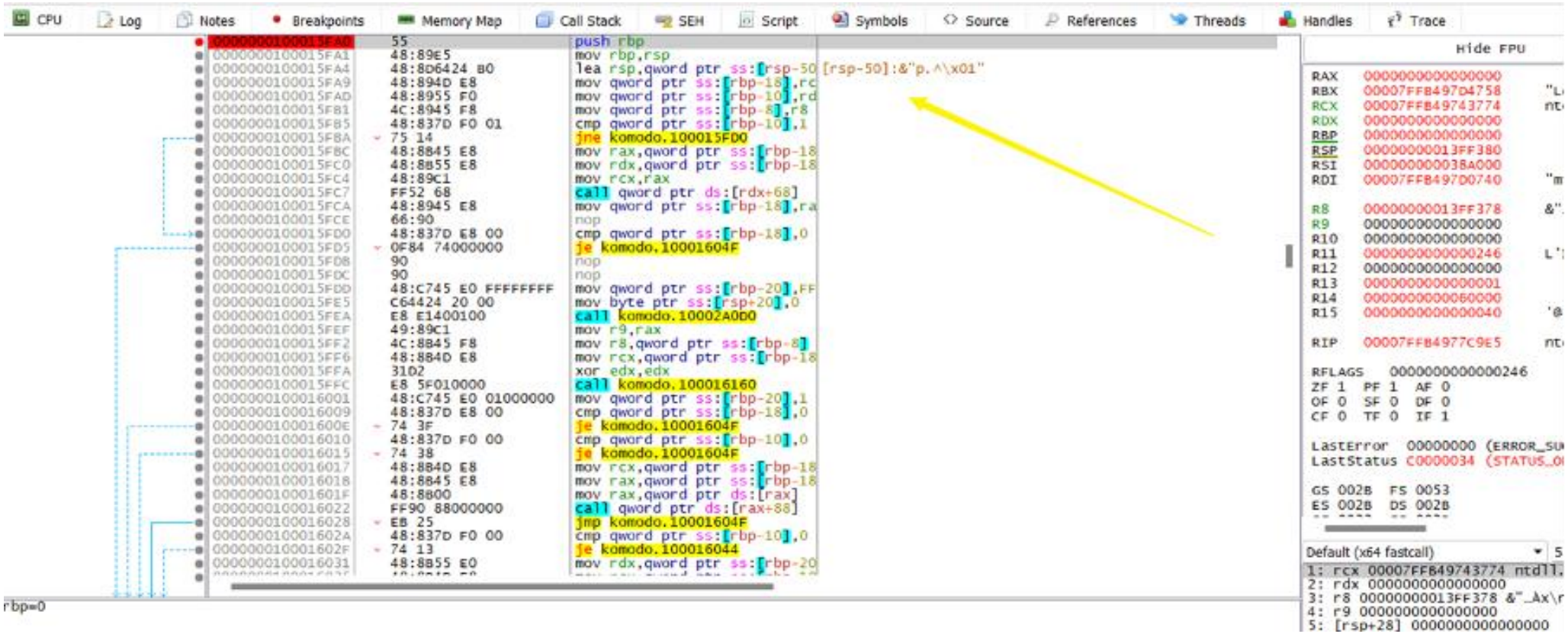
- c，c#，c++，python，ruby，go等

突破主动防御的难易程度

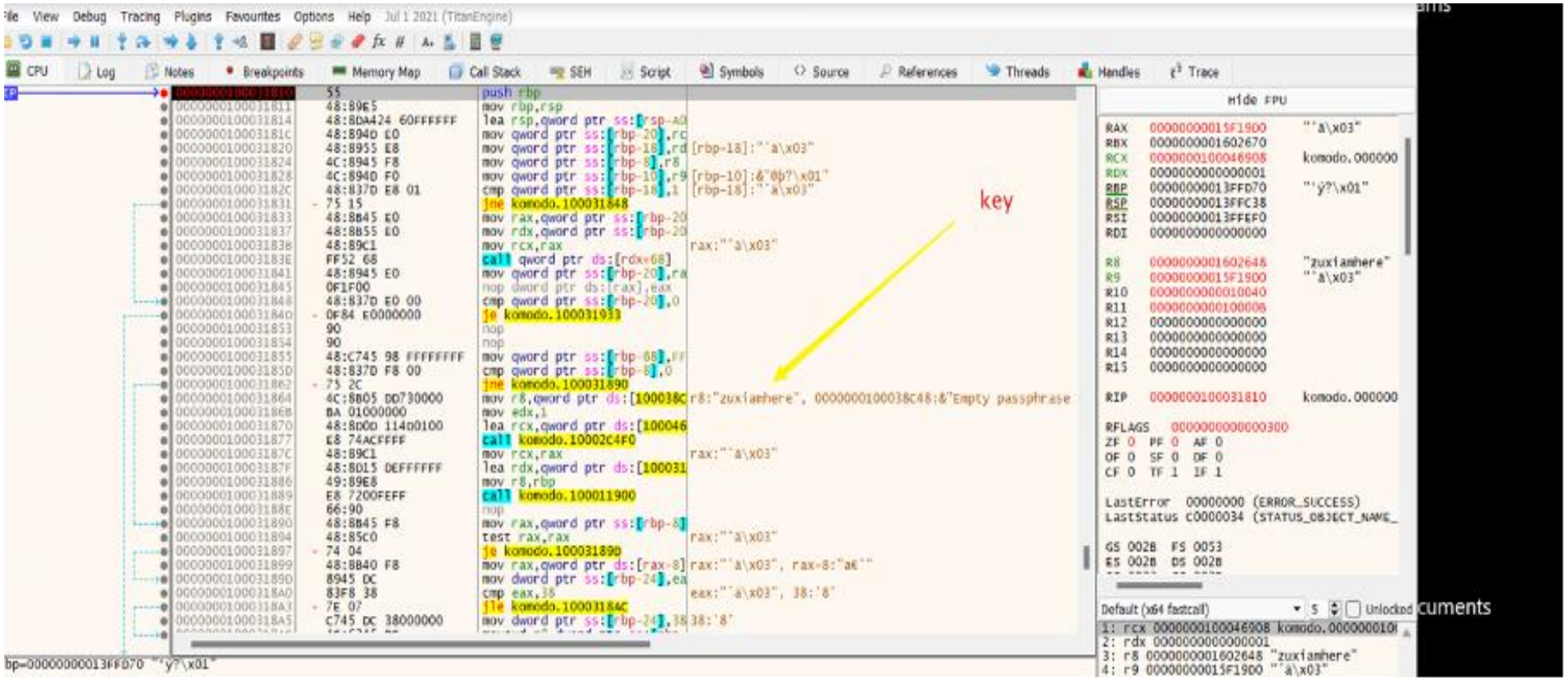
简单	• 360 • 腾讯电脑管家
复杂	• 小红伞 • malwarebytes
困难	• Defender（2022年11月15号，单层自定义加密和管道连接） • 赛门铁克，卡巴斯基（2022年11月15号，双层自定义加密和管道连接）

进阶免杀（内存）

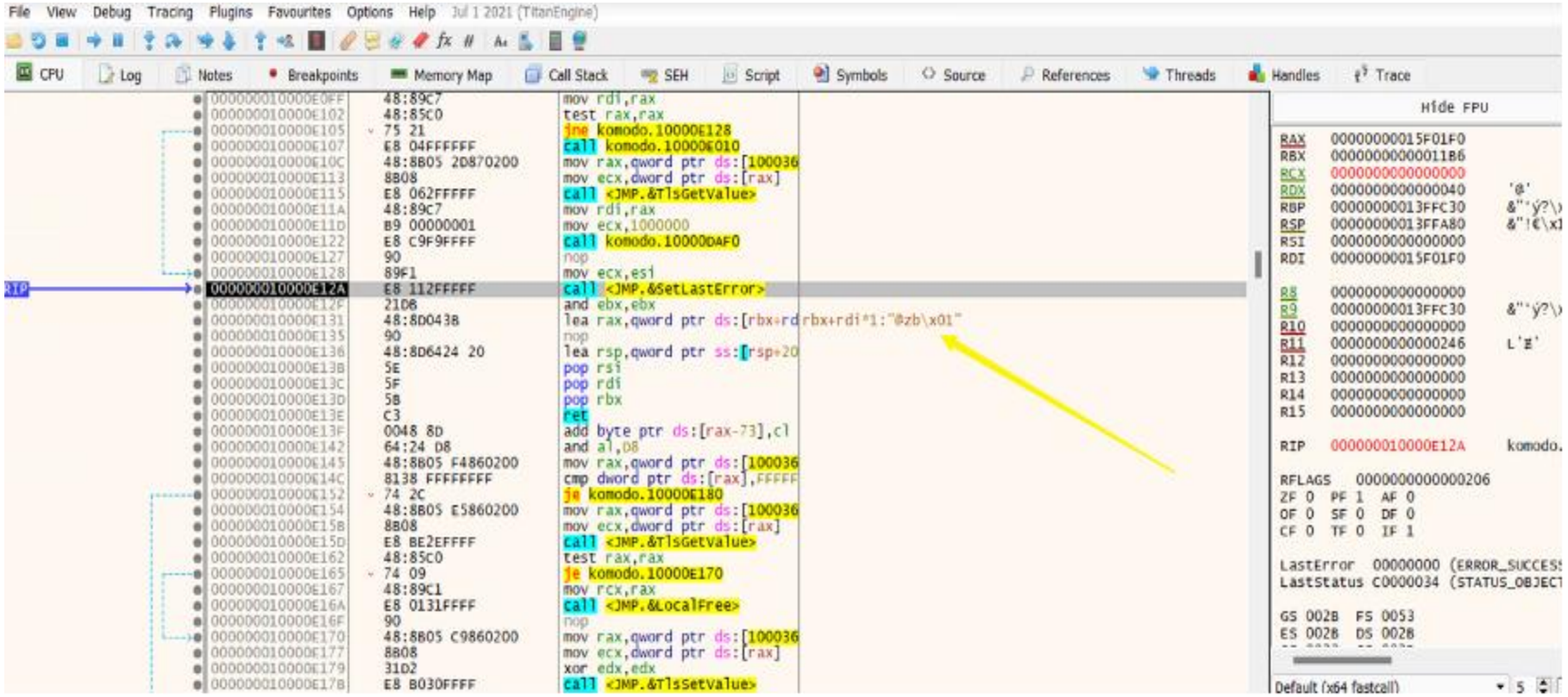
- 1.将加密文件加载到字符串存储器时将断点切换到第一个过程后，使用 lea 指令将加密输入数据的有效地址加载到 ESP/RSP 的分配地址中。



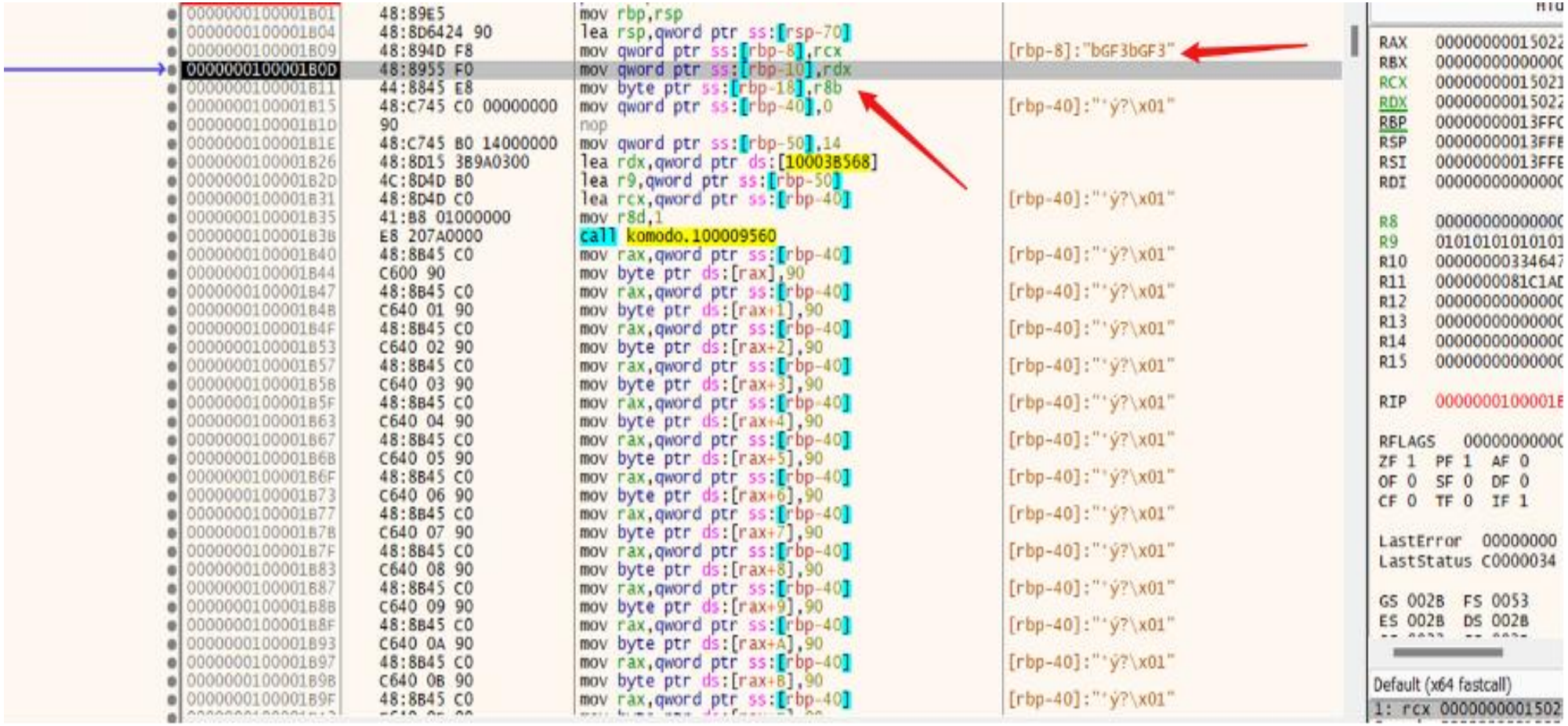
2.让加载程序使用有效的解密密钥解密内存流内的加密数据。



3.由于该输出格式是base64，AV检测是不会查杀的。



4.将恶意有效负载复制到另一个分配的内存地址中。



监管绕过：windows虚拟化与驱动

- windows虚拟化：系统间通讯、流量抓包、锁屏掉线。
- 驱动：windowsPG保护。

针对人工分析处理问题

- 在我们开发过程中，难免遇见免杀程序被人工分析，一套代码使用一辈子不太现实，毕竟如果人工分析都永远分析不出来，那所有搞查杀的也没必要存在，我们总得给别人一条活路。哪怕曾经的经典熊猫烧香，放在现在，也已经不值一提。我们唯一能做的就是怎么让自己的程序更难被分析。给大家举一个列子吧！这是在国外论坛看到的思路，也使用了近两年，python大家都知道，但是python编译好程序之后，我们可以逆向修改里面的变量以及部分内容，使它不符合python语法的同时，却可以在windows下正常运行，当有程序进行逆向时，程序会直接崩溃，导致程序不可逆。网络安全本就是不断学习的过程，我们与杀软相辅相成，共同进步。学习，借鉴，修改，最后走出自己的路，才是长久之道。

总结

- 免杀无高低，适合的才是最好的，在免杀这条道路上，我们与杀软相辅相成，只有走出自己的路，自己的特点，才可以走得更远。