360 版本更新之后，免杀没有原来好做了，因为很多开源的项目库都加了特征，需要自己去改源码。

免杀分两种，单体的和分离的。

不管是单体还是分离，其实原理都很简单。

就是一个 loader，一个 shellcode。

单体就是把 shellcode 写死在 loader 里面。

分离就是 loader 和 shellcode 分开写。

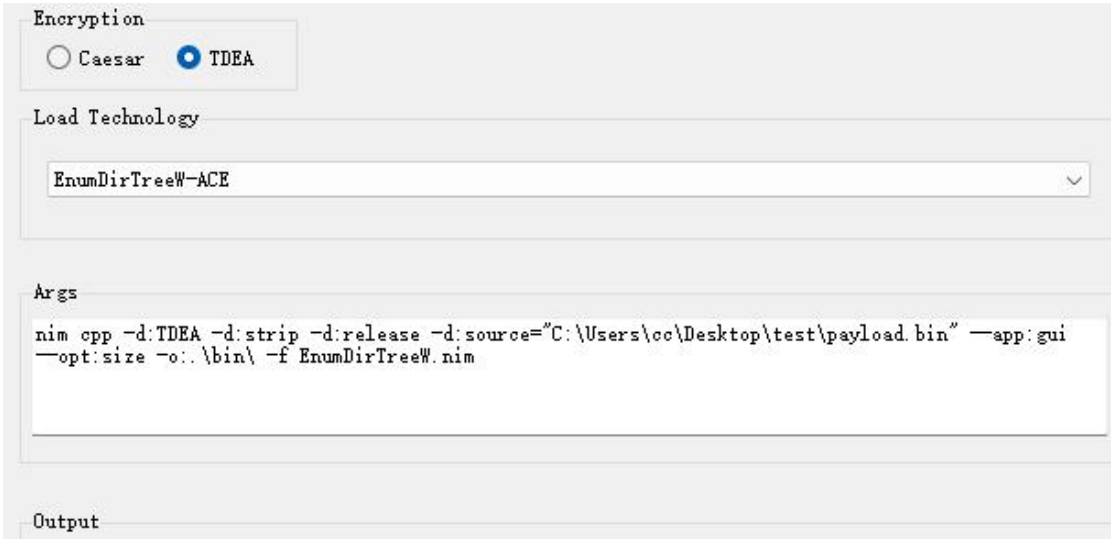
本质没有什么区别，分离是因为 shellcode 写死在 loader 里面，被杀的太厉害了，因此才需要分离。

单体免杀，可以用开源项目

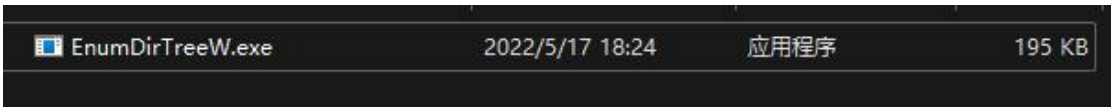
https://github.com/aeverj/NimShellCodeLoader

效果还可以，能上线，但是不稳定，上线之后建议立即迁移进程

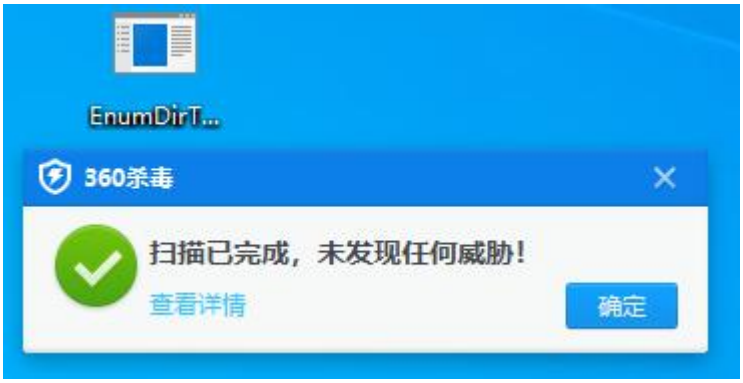
先用 cs 生成 raw 格式的 shellcode，然后加载进 loader



然后得到一个 exe



直接丢到有最新版 360 的环境



这边也是成功上线了

迁移进程，防止本体被杀

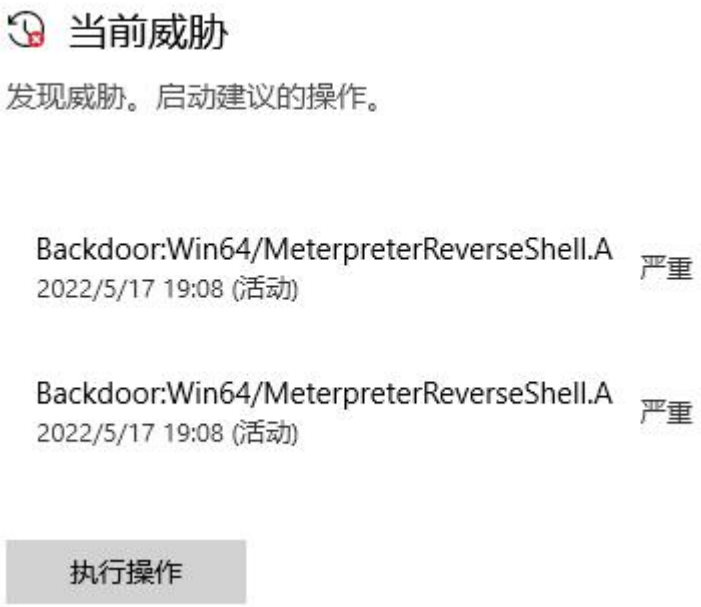
Kill 本体进程，删除本体 exe

然后用这个进程继续操作

这个方法的缺点就是没有持久化，一旦关机，机器就下线了，但是可以适用于一些不需要持久化的项目，上线之后迅速打完内网，然后收工。

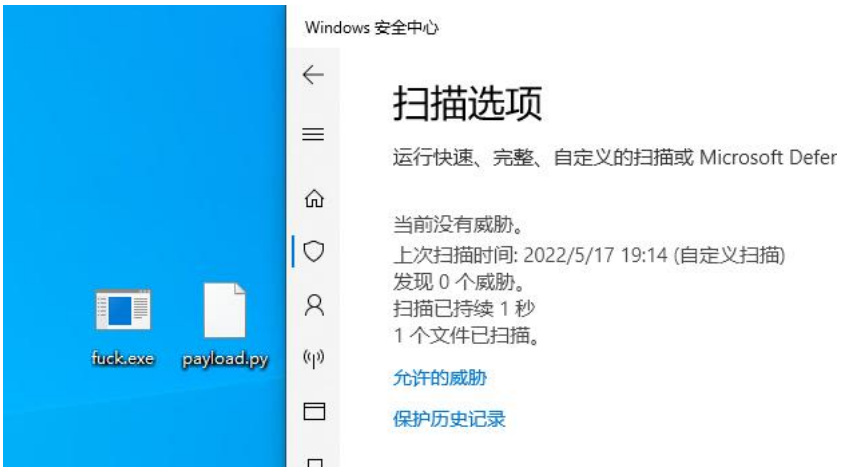
火绒也可以过。

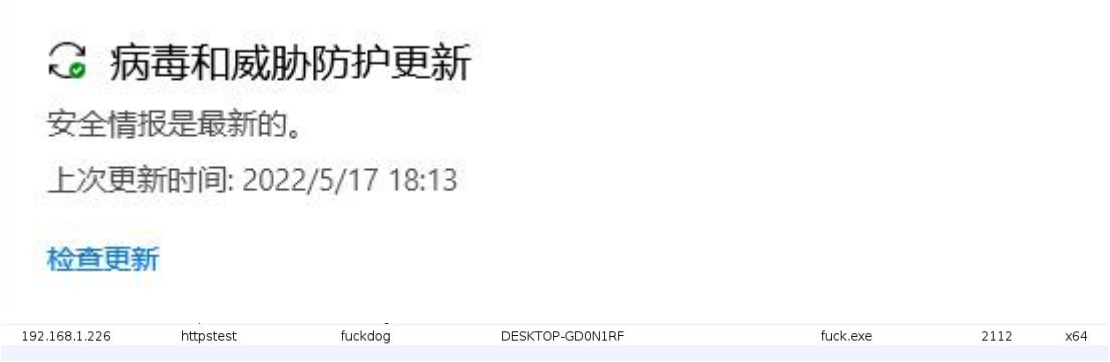
但是过不了 defender。



分离的话，loader 在 github 上有很多，很多都能过 360，如果不能过，稍微改改就能过了。

Defender 这里我是用的分离的过的，因为项目还在用，这里先不放源码出来。





但是思路可以讲，本质国内过的这些杀软，比如 360，火绒，df，主要还是在查特征。

这里为了控制变量，我的 cobaltstrike 用的是纯净版的，就不涉及流量层面的免杀。

如果是说更高级一点的杀软，比如 nod32，卡巴斯基，以及国外很强的 edr crowdstrike 等，这些都会对流量做检测，有些还会 hook 到 windows 底层的函数，就不只是特征这么简单了。

因此过国内这些，其实就是不断的去 fuzz 特征，然后规避特征去做绕过，和绕 waf 是一个道理。

相对来讲，国内主流的三大项，360，火绒，df，过起来还是很简单的。

上难度，卡巴斯基，nod32，cs，过起来难一些，但是也并非无解，无非就是多花一些时间罢了。