“0”代码高强度加密专家－ET 智能虚拟化加密

在古代，人和人之间的信任并不是唯一要关心的问题，身份验证也很重要。如果只有少数人能读会写，那么签名或许足以证明一个人的身份。但随着掌握读写知识的人越来越多，印章逐渐成为“签署人”一种独特的记号。利用这种记号，便可证明信件、文档和法令签署人的身份确实无误。但随着技术的发展，人们可轻松仿制出各式各样的印章，所以它也失却了原先的“独特”性。事实上，伪造印章是件再容易不过的事情。

加密技术的出现最早在军队的通讯上。相传当年凯撒大帝行军打仗时为了保证自己的命令不被敌军知道，使用了一种特殊的方法进行通信，以确保信息传递的安全。这种密码便是著名的“凯撒密码（The Caesar Cipher）。他的原理很简单，说到底就是字母于字母之间的替换。自凯撒大帝的年代开始，一直到当代计算机网络，通信技术在稳步地发展着，同时，保障这些通信的安全也逐渐成为一项重要课题。

10-20 年前，那时懂计算机语言的人极少，懂得破解程序的人就更少了。后来 IT 行业经过几十年的迅猛发展，从 DOS 时代走向 WINDOWS 时代，程序理论走向软件工程理论，软件加密解密技术也逐渐趋于成熟，这几年加密解密技术无论是软件还是硬件都一直像矛与盾一样对抗着。

早期的加密压缩壳，那时我完全不明白它是如何完成这件事，只觉得很神奇，或者说它们不是为了加密，而单纯是为了压缩，这种壳很好脱，简简单单的单步几下就能看到原来的内容了。随着技术的逐渐发展，战场逐渐转向了对 PE 文件的修改，如 IAT 加密，不过还原起来还是很简单。

随着时间的推移，一系列的新技术新思想都逐渐出炉，软件保护由 PE 文件压缩走向对 PE 文件的保护，一系列的 anti 出现，不过道高一尺，魔高一丈，这些都被 Cracker 找出来并饶过了它的 anti 又一次看到了软件的真面目。以前的壳都是对 PE 文件加密,而不会对代码做任何改动，后来又出现了一项新技术：stolen code,修复这种壳开始有点费时了。

传统的保护软件都有一个共同的弱点，即他们都不修改源代码。保护方式仅仅是通过“信封”原理将软件主体封装起来，然后通过一个装载器解压缩保护的软件主体，解压后的软件在内存中很容易被转储并被非法修改。破解者拥有一系列反编译工具可以破坏这种保护，网络上也有许多文章阐述如何脱壳常规的保护软件。

ET 智能虚拟化高强度加密软件是新一代的软件保护系统，不像市场上其它常见的保护软件，ET 智能虚拟化高强度加密软件可以修改软件产品的源代码，转换部分代码为在虚拟机上运行的字节码（bytecode）。

ET 智能虚拟化高强度加密是软件保护技术一个新的进步-VM 技术与硬件相结合高强度保护，具有自设计实现的 VM 引擎。通过对代码的变形来达到保护软件的目的，想修复被 ET 智能虚拟化高强度加密保护的软件，用现有的解密理论是极其痛苦的。VM 的保护技术基本原理。

VM 其实就是 Virtual Machine（虚拟机）的缩写，这里说的 VM 并不是像 VMWare 那样的虚拟机，而是将一系列的指令解释成 bytecode(字节码)放在一个解释引擎中执行。VM 的原理由一个虚拟机引擎由编译器、解释器和 VPU Context（虚拟 CPU 环境）组成，再配上一个或多个指令系统。

编译器：将一条条 X86 指令解释成自己的指令系统。

解释器：解释器附加在被加壳的软件中，用来解释这些自定义的指令。

指令系统：一套或多套自己定义的指令系统，用来虚拟执行这些 bytecode。

1：设计一个好的、简洁的指令系统是很有必要的，写出越少却能执行最全的指令，就说明这些指令复用性越高。

2：在分析 X86 指令时最好为它们分类，注意流程指令、不可模仿指令的处理。

3：解释器的代码设计得越少越好，换来的速度可以变形一下。

首先加壳程序先把已知的 X86 指令解释成了字节码，放在 PE 文件中，然后将原处代码删掉，

改成类似的代码进入虚拟机执行循环。

push bytecode

jmp VstartVM

VstartVM 是进入虚拟机的函数，它的代码大概类似这样的

代码:

```
// 进入虚拟机函数，我的最初版本是写死在 VC 环境下的，不过这样处理起来很麻烦。

void _declspec(naked) VStartVM()

{

    _asm

    {

        // 将寄存器压入堆栈,由伪指令取出存放到 VMReg 中

        //可以考虑为压栈时加入一些随机性

        push eax
```

```
        push ebx
        push ecx
        push edx
        push esi
        push edi
        push ebp
        pushfd
        mov     esi,[esp+0x20]
        mov     ebp,esp
        sub     esp,0x200
        mov     edi,esp
        sub     esp,0x40
Next:
        movzx   eax,byte ptr [esi]
        lea     esi,[esi+1]
        jmp  dword ptr [eax*4+JUMPADDR]  ;跳到 Handler 执行处，由加壳引擎填充
        VM_END                           ;VM 结束标记
    }
 }
```

然后每读一个 byte 跳到目标处模拟执行代码。 JUMPADDR 就是一张函数表，每次从内存里读出一个 command code（其实就是偏移），然后转向那个过程 现在再来看看它们的执行过程：

1. 首先进入虚拟机，压入寄存器，然后设计一些伪指令来将寄存器保存到虚拟机环境中去
2. 一切工作就绪，真实寄存器存放的值是什么已经不重要了，因为我们将它存放到了虚拟环境中去了，自己可以随便使用而不用怕影响到原来的代码。
3. 从[esi]得到一个偏移，它指向加密函数的地址
4. 从内存中得到 opcode，模拟出这句代码，并存放到堆栈。
5. 注意有时要修正虚拟堆栈寄存器的值，否则会挂得很惨。
6. Jmp 到 next，这时转向第 2 步。

一个简单的 VM 引擎就是这个样子，不过一个专业化、的引擎可不仅是这个样子，还要有很多东西，比如多线程、流程化指令，虚拟-现实之间的转换，支持 SEH 异常（包括 VB、VC 的异常等等）。看到这里，你可以发现，虚拟机就是一层复杂的壳子，把原来的代码藏得毫无踪影，如果你想还原出原来的代码基本是不可能的

ET 智能虚拟化高强度加密软件正是基于最新虚拟机加密技术设计开发。

众所周知虚拟机加密 - 目前最强的软件加密保护方式

虚拟机加密保护是目前最强的软件加密保护方式。由于虚拟机是带有不同于 Intel 8086 处理器指令系统的虚拟处理器，可以将部分代码转换为在虚拟机上运行的字节码，并且它的指令集是不为人知的非大众化的汇编指令；您可以将虚拟机想象成为带有不同于 Intel 8086 处理器系统指令的虚拟处理器。例如，虚拟机没有比较两个操作数的指令，也没有条件跳转和无条件跳转指令等等。这样一来，破解者就需要开发一整套的解析引擎来分析和反编译字节码。

因此，自主研发的虚拟机解析引擎具有很高的强度，破解者就需要开发一整套的解析引擎来分析和反编译字节码。以现有的解密理论，破解者想要还原出源代码几乎是不可能的。 虚拟机加密是目前最有效的保护软件不被破解的方式，所以大家在选择软件加密保护系统时一定要看该系统是否有虚拟机解析引擎，也就是通常所说的 VM 加密。目前综合国内外各虚拟机加密软件采用最新技术 ET 智能虚拟化高强度加密软件是很好的选择，无论是兼容性或是强度均处于行业领先地位。

ET 智能虚拟化高强度加密基于 ET199/ET 金刚锁两种加密锁，采用先进的智能虚拟机技术，是目前操作最简单，加密强度最高的加密方法。ET 智能虚拟机高强度加密支持各种常见的 PE 文件。

ET 智能虚拟化加密特点如下：

- **智能化：自动智能选取加密点，可根据实际情况定义加密点数量**
- **透明化：加密人员不需要有专业的加密经验就可以完成汇编级别的高强度加密**
- **灵活化：具备“模拟运行”功能，可根据软件运行情况调整加密点**
- **安全化：采用先进的智能虚拟技术，与传统的外壳加密有本质的区别**
- **标准化：兼容任何木马病毒查杀软件**
- **广泛化：适用于所有 PE 结构的程序软件**

ET 智能化虚拟机高强度加密可**有效免杀**—ET 智能化虚拟机高强度加密软件的虚拟机加密代码应用并非传统的修改特征码，也不是修改入口点+花指令，更不是加壳压缩！更不会破坏标准 PE 结构可有效免杀！借于这种技术你可以千变万化，可有效免杀。

ET199 智能虚拟化高强度加密工具是一款先进高效的软件保护产品。它可以保护您的程序不被非法复制，非授权访问和使用。

ET199 智能虚拟化高强度加密工具的设计在硬件上基于 ET199、ET 金刚锁这 2 种类型的加密锁，软件上基于虚拟机技术，这两方面技术的结合使其具有操作简单，保护强度高，使用透明等特点，该工具支持各种常见的 PE 可执行文件。

ET199 智能虚拟化高强度加密工具是一款透明的软件保护产品，使用 ET199 智能虚拟化高强度加密工具加密软件，用户无需具有专业的软件加密知识就可以非常便捷的将自己各

种类型的 PE 文件进行加密保护，受保护的程序只有通过指定的加密锁硬件存在的条件下才可以被运行。加密后的程序即使被别人非法获取，由于没有所依赖的硬件，程序依然无法运行

ET199 智能虚拟化高强度加密工具能够对 Windows 平台下的各种常见类型的 32 位 PE 文件进行保护。这些文件包括.EXE，.dll，.ocx 等文件格式，及各种语言编写的 PE 文件。

用户只需简单的用鼠标选择要保护的 PE 文件，即可完成对 PE 文件的保护。所有的安全检测、PE 文件的加解密等操作都是由 ET199 智能虚拟化高强度加密工具在后台自动处理完成，从用户角度看对文件的保护是完全透明的。

ET199 智能虚拟化高强度加密工具在硬件上基于 ET199 和 ET 金刚锁这 2 种加密锁实现。该锁使用虚拟机技术进行以函数为单位的加密，从而提高加密锁的加密强度。基于此硬件实现的 ET199 智能虚拟化高强度加密工具使用户数据的安全性提高到一个新的高度。

文件保护中心可以运行于多种版本的 Windows 操作系统之上。加密的 PE 文件可以支持的操作系统有：Windows 2000/XP/Sever 2003/Vista/ Server 2008/7，为用户提供更多的平台选择。

工作原理：

要让 ET199 智能虚拟化高强度加密工具正确保护你的文件，首先要执行将 PE 文件（Windows 平台下的.exe、.dll、.ocx 等文件）进行加密操作。例如我们要加密一个 “记事本”程序 Notepad.exe，加密后的 Notepad.exe 文件只有使用加密时设置的对应加密锁才能打开。若未插入对应的加密锁，那么加密后的 PE 文件是不能被打开的，默认会提示“找不到指定的加密锁”，从而保证了 PE 文件的安全性。

ET 智能虚拟机高强度加密基本原理如下：

加密方式采用以函数为单位，既可以由用户选择加密哪些函数，且加密后所有被加密的函数都被虚拟化了（且一条指另被虚拟化成 N 条未知虚拟机指令），虚拟化的原理就是，将 win32 平台汇编代码转换成自定义的虚拟机指令，自定义虚拟机指令只有虚拟机的设计和开发者知道其它人不可能知道，因此这是安全性的有效保障，比如我们可以将 call jump 指令自定义转换成自实现的虚拟机指令从而达到外部无法还原成汇编代码目的，从而使破解者无法跟踪程序的逻辑及更不可能推导出关键算法的实现。并且每个被虚拟化的函数都与硬件有关（即没有硬件的支持不可运行）又在虚拟化的基础上增加了加密的强度。

当然 ET 智能虚拟机高强度加密软件系统虚拟机引擎内部设计是相当复杂的，要想破解被 ET 智能虚拟机高强度加密软件保护的程序几乎是不可能的。

ET 智能虚拟机高强度加密，通过 4 种方式定位要保护的函数，分别为标志定位，输入表定位，自动搜索定位和手工输入定位。标志定位方式和输入表定位方式适用于待加密可执行文件在有源代码的情况下进行加密，在源代码里可以添加自己想要保护的函数标志进行保护，如果待加密的可执行文件已经带有标志定位或输入表定位标志，在加载此文件时就可自动获取要保护的函数，需要注意的是：被保护函数需在函数返回前加以保护，否则会找不到

标志。自动搜索定位方式适用于没有源代码的情况下模糊查找函数，当然标志定位或输入表定位可以和自动搜索定位同时使用，此时自动搜索方式不会重复找出标志定位或输入表定位已找的函数。手工输入定位适用于在知道具体要保护哪个函数的情况下进行保护。

此外在虚拟化的基础上也进行了反调试处理，屏蔽大部分调试工具对通过 ET 智能虚拟化工具加密后程序的调试。

正是采用了最新虚拟加密技术（加自设计复杂的虚拟机引擎）+硬件依赖双重保护，ET 智能虚拟机高强度加密软件可有效保护软件被非法破解。