

# 软件加密良方

## ——软件加密之决定因素与错误观念

**引言：** 软件加密对于软件开发商来说至关重要，出于对软件安全方面的考虑，很少有人能够把软件加密方面的“秘笈”拿出来共享，大部分是孤军奋战，了解和认识都存在一定的片面性。

究竟我们应该如何看待软件加密？如何系统地规划软件加密方案？如何总结和弥补加密方案中的薄弱环节？有没有人能够把上述信息写成文章，实现“资源的共享”呢？目前，介绍上述内容的文章还是非常“罕见”的，在杂志和网上也是很难见到，相反介绍软件破解方面的信息却是很多。

本文虽然没有提到具体的程序代码和具体的加密产品，但从软件加密的决定因素和软件加密常见的错误观念入手，对软件加密方案的规划方法进行了全面的介绍。可以说本文是飞天诚信多年以来从事软件加密行业的经验之谈，是软件加密的一剂良方。希望本文对软件开发商及从事软件加密的相关人员能够有所帮助。

### 一、软件加密之决定因素

很多开发商认为软件加密就是保护软件不被拷贝就行了，在做加密的时候，最终加密开发者很少或从来没有对如何加密进行规划，从而导致这样做出来的加密方案存在一定的弊端或不足。在这里，我们根据多年来的软件加密经验，总结了以下与软件加密相关的一些因素，供大家参考。

#### 1、开发环境



开发环境在很大程度上会影响你对加密方式的选择。开发环境一方面包括你用来开发的语言环境，另一方面也包括了软件运行的系统环境。

如果是在 Windows 平台上面运行的软件，能够选择的加密方式会比较多，但如果是在 Linux 或其它操作系统下，可供选择的加密方式就会少得多。对于那些需要编织跨平台应用产品的开发商来说，如果希望能够保持加密上的兼容性，就必须谨慎的选择他所需要的加密产品。

#### 2、软件类型

软件类型的分类方法有很多种，对加密而言比较看重的是通用软件和专用软件分类。因维护代价的不同，所能采取的加密方式也不同。对于通用类型的软件来说，因为用户群是海量的，即使在加密上出现一点点小问题都可能带来海量的维护性工作，对于这种类型软件的加密，可靠性是第一位的。

### 3、加密强度

加密强度是一个相对性的概念，软件保护并非越强越好。软件加密强度是以开发时间和运行效率为代价的，要选择能够满足要求的加密方式。

对于那些受盗版侵害比较大的成熟软件来说，希望有更好的软件保护技术来保护自己的权益。但对于那些未经市场检验的新软件来说，投入太多精力在提高加密强度上会让用户有本末倒置的感觉。

### 4、生命周期

软件保护是否合适的检查标准应当以软件的生命周期来衡量。一味的追求高强度、不可破解的保护是得不偿失的。我们的标准是，如果一个软件能够在其所期望的生命周期内不被破解或不被完全破解，那么这个加密就是成功的。

### 5、用户群

软件产品的用户群也是影响加密效果的一个重要因素。一般来说越常用的软件受解密者攻击的可能性也就越大，但解密者的水平以业余者居多。

对于那些售价高昂的行业软件来说，受到攻击的机会比较少，但这些软件也往往是那些专业解密者所青睐的目标，这些解密者的水平也比较高。

### 6、产品售价

软件的价格往往也决定了软件加密上投入的成本，因为在销售渠道上还要让利给销售商，软件加密的成本一般控制在软件价格的10%以内。

### 7、销售模式

软件的销售模式与加密方式紧密相关。销售模式上的要求会直接影响到软件编制和加密。譬如软件的多模块管理、软件租用...都是通过加密控制来管理



的，开发商应该在软件开发过程中就把相关因素考虑进去。

### 8、升级模式

软件在销售以后，不可避免的要遇到升级和维护的问题，而升级和维护也并非都是无偿的。很多开发商也希望能够通过软件保护来更好地管理这些问题，以便降低软件维护的成本。

## 二、软件加密之错误观念

就我们多年从事软件加密的经验来看，大多数的开发商在设计 and 实现自己的软件保护过程中对软件加密理解上存在某些偏差所导致，我们在这里分析一些比较典型的问题，以便大家将来在实现加密的时候不要犯同样的错误。

### 1、加密是专业公司的事，只要选择好加密产品就行了

这个观点可以认为是一种对软件加密比较片面的理解。所有从事加密行业的公司所提供的产品基本都属于需要二次开发的类型，加密公司只能够保证它所完成部分的安全性，而开发商如何有效的使用加密公司所提供的功能来完成加密要求，很大一部分取决于

开发商自身。

虽然有些加密行业的公司可能会提供一些比较典型的解决方案，开发商基本可以照此实施，但这些典型方案会产生相对固定的加密模式，解密者分析起来因而会相对容易很多。如果固定的加密方案能够解决盗版问题的话，加密公司直接提供方案就好了，而不必要求开发商进行二次开发。

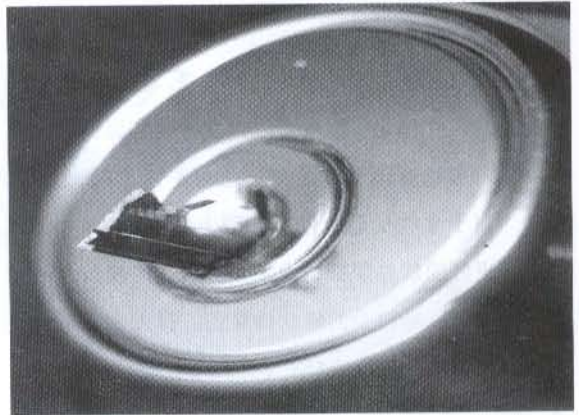
### 2、加密等程序都编好以后，让一个人搞两天就行了

这种思想在我们碰到过的很多软件开发商那里都感受到，在项目测试基本完成后，只抽出1个人专门给软件搞两天加密工作。但编程人员原来就没有太多的加密经验，而且时间安排又非常紧，往往凑合一下就发出去了，这样的加密效果可想而知。现在的解密者，大多都是“身经百战”的高手，即使是殚精竭虑设计出来的加密也不见得能挡得住，何况这种匆匆忙忙设计出来的加密，效果就更差了。

软件加密如果想要做得好，不一定非要在项目初期就考虑，但起码要有个相关的加密设计方案才行。从上文的软件保护的应用模式来说，要根据自己开发的软件类型来设计相应的保护方式是最好的。

### 3、负责加密的人越少越好

很多软件公司认为软件的加密应该属于公司的核心机密领域，整个加密编制和处理了解的人越少越安全。事实并非如此。我们都知道加密属于百密一疏的问题，也许加密者很多地方都考虑得比较周全，但若有一个地方没有考虑周到，就有可能被解密者作为突破点，把整个软件破解掉。如果多个人参加得话，即使每个人



都会有一定疏漏的地方，但却有取长补短的效果。

对于一个软件公司来说，完全有能力让多个人来搞加密处理。如果担心安全性出问题的话，可以考虑让每个人只搞一个加密点，再由专门的人来汇总这些加密点。这样完整的加密还是掌握在少数人的手中。

### 4、软件采用外壳加密就好了，强度高，加密简单

这也是很多软件开发商所采用的方法，目前在网络上有很多免费的外壳程序可以下载，而且很多加密锁厂家也都支持外壳型加密，直接对可执行文件进行保护。这种方式的保护使用起来非常容易，加上这些外壳都具有专门的反跟踪、反调试的功能，看起来非常完美。但外壳加密有其利就有其弊，外壳加密正因为是通用的加密方法，因而也就有通用的解密方法。留心一下就可以发现，网络上很多名气很大的外壳加密程序，譬如：ASProtect, PELock... 都有通用的解密工具。

也许你会说，那就找一些名气不那么大的外壳就好了，但往往那些不很流行的外壳工具通常都会有这样或那样的兼容性问题，譬如程序在 Windows9x 下面可能没有问题，但到了 Windows 2000 上就会出错。

另外，随着各种外壳程序的出现，通用的解外壳工具也出现了。这些工具并不针对哪种特别的外壳程序，但能够采用某种特别的方法脱掉大部分外壳加密程序的壳，譬如：ProcDump...。外壳加密不再是软件保护的捷径了。

以上我们列举的问题并不是说外壳加密没有用了，我们想说的是最好不要把外壳加密作为软件保护的唯一手段。作为一种辅助手段来说，如果配合开发者精心设计的加密方案，就可能会发挥外壳加密的最大长处。

## 5、采用国际标准加密算法加密的程序一定很难破解

对于软件加密的概念理解不很清楚的人，往往会有这样的想法，他们认为好的加密算法就会带来好的软件保护效果。实际上这完全是两个不同的概念。象 DES、RSA 等国际标准加密算法的对象是数据，对于纯数据保护来说，好的加密算法代表一切，但对于软件来说则不然。

软件是运行在客户计算机上面的程序，这个程序对于客户来说是完全公开的。如果能让用户在他的计算机上正确无误的运行软件，所有的解密、判别过程都必须在程序中出现。即使破解者看不懂开发商所使用的加密算法，但若把解密后的比较过程改写为强制成功的话，一样可以破解开发商的软件。

所以，对于软件保护来说，防止破解的有效方法是“隐藏算法”而不是“使用高难度算法”，当然在算法能够很好隐藏的前提下使用国际标准算法会比自己编的算法效果更好些。既然算法能够被隐藏，那么最

好也同时把算法进行某些变形处理，让算法和标准算法的处理结果有一些差异性，这样才能实现更好的隐藏效果。

## 6、编译出来的程序一定比解释运行的程序难破解

很多开发商会有这样的认知，完全是因为很多解释运行的程序有反编译工具的存在。因为被解释的内容就是实际程序的二进制压缩编码，反编译工具能够一点不差的把它回复成原始的代码。正因为这个原因，大多数开发商都认为解释运行的程序在安全性能上都很差，如果要搞加密的话一定要使用 C/C++ 这样的编译类型的语言来开发。其实这个观点是不正确的。在当前市场上开发语言种类很多，并非所有的解释型开发语言都有反编译器的存在。没有反编译器存在的解释运行类的开发语言所开发出来的程序在跟踪和破解上比编译类型的开发语言要困难得多，在加密上具备了先天上的优势。

这也是我们所提倡的一个观点，真正好的加密应该是无为而治，通过技术复杂度来实现安全。开发者不需要特别考虑软件加密的问题，但加密出来的软件却非常难破解。

在软件技术极大丰富的今天，即使是最好的破解者也不可能掌握所有的技术。一个最蹩脚的 Lisp 编写程序的加密效果可能比一个最好的 C 编写程序的加密效果要好得多。因为 Lisp 这种比较冷僻的语言懂得的人很少，99%的解密者根本不知道如何分析一个编译 Lisp 语言编写出来的执行程序。这就是我想表达的一种思路，加深软件上的技术复杂度，往往比采用任何特别的反跟踪、反调试手段更为有效。