# 针对黑客行为采取反击措施：保护软件产品免受 8 大盗版威胁

白皮书

当今，越来越多的独立软件供应商（ISV）选择实施某种类型的保护以确保他们能够从付费用户身上获得完整收益。

## 目录

根据 BSA/IDC 2010 隐私报告，对应每 100 美元的合法软件销售额，市场上另外会有价值 75 美元的非许可软件，而这部分软件是不会给原始 ISV 带来任何收益的。

## 软件攻击类型和如何针对攻击进行防护

当今，独立软件供应商（ISV）必须实施某种类型的保护以确保他们能够从付费用户身上获得完整收益。

任何软件保护解决方案背后的基本原理都是通过某种许可授权方式来限制未许可软件的使用。可以通过硬件锁、软许可证或自己开发的解决方案来实现这一目标。总的来说，基于硬件锁形式的保护可以提供最强效的保护，而基于软许可证的形式则可以为试用版软件和电子软件分发（ESD）方式提供更好的支持。

基于外部硬件的解决方案可以提供当今可用的最高水平的保护。但是黑客行为是软件业内的一个顽疾，导致全球范围内软件供应商有数十亿美元的收益损失。因此，选择合适的方案来保护好每一个有可能被黑客攻击的入口点是很重要的。

**本白皮书研究了多种最常见的黑客攻击技术以及保护应用程序免受盗版危害的最佳反黑客攻击方法。**

## 黑客行为 1：修补可执行文件或动态链接库

在这种情境下，软件黑客会尝试破解和/或调试可执行文件或动态连接库以获取受保护的代码。可执行文件随后会被修补从而更改运行逻辑或者在代码内移除查询请求。通常情况下，软件黑客会发送一个小的独立补丁可执行文件（被称为“破解程序”），让终端用户运行，从而修补被许可的软件。

### 应对措施

目标是让修补可执行文件的过程比购买软件更复杂而且代价更高。受保护文件数量和软件黑客破解保护所需时间是呈正比的。保护大量可执行文件和动态连接库的最便利的方式是使用自动保护工具，被称为“加壳工具（packers）”。将保护代码插入到编辑后的可执行文件中会缩短对单个文件实施繁复写保护代码所需的时间。最有效的工具也会提供特定的反盗版措施，如反调试机制，可执行文件保护和数据文件保护机制允许受保护的应用程序实时的加密和解密数据。

## 黑客行为 2：暴力攻击

暴力攻击情境包括（无论一般的还是特定的）包括黑客费尽心力地尝试冲破每一道保护机密数据的安全防线。通用攻击是指硬件锁本身被破解，任何实施改进措施都不能阻止这通用攻击，被破解的硬件锁所保护的所有应用程序都会受到威胁。。

特定攻击只会对一种特定软件应用程序破坏一个特定的硬件锁保护，这不会对使用相同硬件锁的其他软件供应商带来风险。

### 应对措施

如果受保护的数据足够大的话可以克服暴力攻击问题，例如破解所有保护几乎是不可能的。为了保护核心的算法，硬件锁应该有一个受保护的内核和通信通道，用来存储算法密钥。硬件锁的保护内核包含 128 位 AES 加解密密钥，该密钥能够对加密狗的 运算进行保护。

在多个经济体内强制推动减少盗版的方法包括供应商合法项目、政府教育、强制行动和主要技术革新——如数字版权管理（DRM）部署等。

## 黑客行为 3：仿制安全密钥

在仿制（记录/回放）攻击环境下，在应用程序和硬件锁之间交换的所有信息都会被记录下来，尝试替代设备驱动器。

中间件的使用（常常是一个软件应用程序）会提供一个平台，该平台会尝试模拟请求和主机发出的响应。如果实际密钥返回查询/响应的话，硬件锁会回放先前记录的信息。

有限功能仿制程序只会记录并回放先前记录的查询/响应，而完整功能仿制程序也会仿制密钥，包括其受保护设置。

为了创建一个完整功能的模拟器，软件破解者要求访问加解密密钥。成功地模仿双通道通信意味着该应用程序已被破解。模仿加解密密钥并不是通用的攻击，它只会影响密钥其所存在的应用程序，可以被轻松的修改。

### 应对措施

安全密钥依赖于应用程序和硬件密钥之间的一个受保护的通道。在受保护应用程序和密钥之间交换的数据都会被保护。芯片内置的 128 位随机 AES 确保通信通道受到可靠保护。通道会对每个对话使用不同的安全密钥。这意味着尝试记录通道中所交换数据的黑客不能重放数据，因为用于对数据进行保护的密钥与用于对数据进行解密的密钥是不同的。此外，通过使用硬件锁保护引擎来解密硬件编码数据，受保护的应用程序可以检测到有限功能仿制程序。仿制程序本身不能正确解密此类数据，因为它并不能访问受安全密钥。

## 黑客行为 4：修改硬件锁内的内存

许可数据一般会被保存在软件保护锁内的特定内存中。在这种情境下，软件黑客会尝试访问硬件锁内存以修改其许可条款。例如，黑客会尝试将一个到期的基于执行次数的许可更改为一个永久许可，或者激活一个并未付费购买的功能。这会显著降低软件发行商的当前和未来的软件许可收益。

### 应对措施

为了防止硬件锁内存被修改，许可实施必须包括使用内置只读内存（ROM）。ROM 是一个内存片段，该片段可以包含只有保护应用程序才能访问的数据，但是不会重写该数据，与常规的读/写内存不同。只能使用远程更新（C2V 和 V2C 文件类型）才能对这种内存片段进行更新，只有特定供应商才能创建这类更新。

## 黑客行为 5：设备共享

设备共享是指在未经许可的情况下多台 PC 共享一份许可，例如在运行带有多个并行终端的同一网络内的多台机器上使用一个硬件锁。这种情况不会影响应用程序本身的保护强度，但是它会显著降低软件供应商的软件许可收益，因为多名用户共享一份应用程序许可。使用静态密钥进行保护很容易导致这种攻击。

软件发行商自有的保护实施必须足够彻底全面并且有一定的创新性。只有最大程度提高基础架构以及对它的保护，才能确保软件得到真正可靠的保护。这听起来好像是一个庞大的任务，绝非易事，实际上这是很容易实现的。实施有效的保护不仅可以延长您产品的生命期，同时可以显著提高您公司的收入，让您的努力获得丰厚回报。

有两种可以实现设备共享的潜在情境：

- 多个 VMware 会话：在此情境下，多个 VMware 会话并行运行在一个硬件平台上。例如，每个 VMware 会话都会使用其自有的驱动程序实例来与硬件锁通信，允许多个虚拟操作系统访问一个应用程序。
- USB 共享集线器：在此情境下，共享一个 USB 集线器的多台电脑会并行使用许可硬件锁（如果单个与其进行通信的话）。即使只有一个硬件许可的情况下也能让多台电脑运行一个应用程序。

#### 应对措施

软件供应商需要确保访问硬件锁的电脑或设备驱动器的数量不会超过客户的采购数量。硬件锁保护通道可以防止多个设备或机器同时访问一个硬件锁。

应用程序的每次使用都会打开一个驱动程序示例；两个或多个驱动程序示例就不同通过受保护通道与硬件锁进行通信。硬件锁驱动程序会针对这些情境进行保护，强制只能运行允许数量的许可。

### 黑客行为 6：克隆硬件密钥

在这种不寻常的情境下，黑客会尝试对一个硬件锁进行逆向工程处理并制造精确的复制密钥。从逆向工程角度和专业技能角度来说，这种做法都是非常有难度而且成本高昂的，更不用说仿制这种硬件锁的生产成本。

#### 应对措施

针对这类攻击的应对措施可以是基于硬件的也可以是基于软件的，最好是两者兼备。降低这类攻击可能性的一种办法是采用不能购买到的并且包含特定反篡改措施的定制硬件组件。而软件方面，可以使用固件保护应用程序避免硬件密钥被提取/克隆/复制。

### 黑客行为 7：时间篡改

这种情况相对比较简单：终端用户尝试对运行受保护软件的电脑上的机器系统时钟进行更改，将其改为一个更高的日期，这样可以重新运行一个已经到期的时间限制许可。

#### 应对措施

确保试用版软件可以正常发行的一种方法是：只提供受限或不完整的软件版本。但是如果您想创建基于时间的许可以提供软件订阅或完整的时间限制试用规定的话，那么时间篡改风险就是一个必须进行评估并应对的一个问题。

确保基于时间许可授权的另外一种方法是使用一个内置到硬件密钥中的电池供电 RTC（真实时间时钟）。这样硬件锁就能够检查并保存当前系统时间，让软件开发商可以检测到系统时间变化，使用此信息阻止对系统时间进行篡改。因此，可以对应用程序进行预编程以禁用该程序或在受限模式下运行。

## 常规保护策略

- 同时使用基于 API 的实施和自动保护
- 在您的代码中插入多个查询调用
- 使用硬件锁的芯片安全引擎加解密数据

## 黑客行为 8：篡改的许可申请和更新

这种情境是指实际实施许可请求和更新。如果在软件层面上验证这类请求的话，例如使用硬件锁设备驱动程序，黑客会尝试攻击此代码并强制自动生成并应用所有许可。这可以适用于已经到期的许可或已经被应用的许可更新，从而会授权对作废/过期硬件锁的全面访问。

### 应对措施

应该由硬件锁本身执行许可授权和许可更新应用，不要让调试程序参与。不能获得有效的许可会导致一些功能被锁定。例如，加解密引擎应该被禁用，直至成功获得一份许可授权为止。类似地，只有在硬件密钥自身验证签名后才能应用许可更新。

## 常规保护策略

推荐使用硬件保护锁的软件发行商采用以下常规软件保护策略：

### 同时使用基于 API 的实施和自动保护

通过使用 API 来实施与软件保护锁的查询/响应来最大程度提高保护强度，使用具有反调试和反逆向工程功能的自动保护工具。使用一种保护方法不会妨碍使用其他方法。

### 将多个查询调用插入到您的代码中

将多个查询调用插入到保护密钥代码并使用来自硬件锁的数据可以挫败对您软件进行破解的尝试行为。多个查询/响应会提高对一个保护方案进行跟踪的难度。

### 使用硬件锁的芯片安全引擎　加解密数据

在软件发行之前实施的保护程序和在受保护应用程序的运行过程中实施的解密程序均应该在硬件锁内执行，不能让任何调试程序参与。安全密钥应该采用一种高强度保护的公共保护算法，如 AES 等。选择用采用此类算法的安全密钥对数据进行保护可以显著提高软件保护强度。黑客想要破解软件首先要对数据进行解密。安全密钥长度需要在保护强度和性能方面进行平衡。一个短安全密钥可以提供更高的性能，保护强度。128 位 AES 是一个很好的选择，它比 256 位 AES 处理速度更快，而且很难被破解。

牢记安全水平永远是由整个保护系统中最脆弱的那一点决定的。有效的实施必须建立在一套能够为所有构建模块提供安全实施的软件保护方案基础上。

- 采用经验证的公共安全算法（如 AES）的保护引擎
- 具有反调试保护和其他反逆向工程功能的自动保护功能
- 受保护的通信通道
- ROM-只读内存

## 结论

软件发行商自有的保护实施必须足够彻底全面并且有一定的创新性。只有最大程度提高基础架构及其实施的保护强度，才能确保软件得到真正可靠的保护。这听起来好像是一个庞大的任务，绝非易事，实际上这是很容易实现的。实施有效的保护不仅可以延长您产品的生命期，同时可以显著提高您公司的收入，让您的努力获得丰厚回报。

## 赛孚耐圣天诺软件货币化解决方案

赛孚耐在为全球的软件和技术供应商提供创新且稳定的软件许可和版权管理解决方案方面拥有 25 年之多的从业经验。公司的圣天诺®软件货币化解决方案易于整合和使用，具有创新性并且是专注于功能的，该方案设计用于满足任何公司（机构）的独特许可实施、强化和管理需求，而无论公司（机构）有怎样的规模、技术要求或组织结构，均能满足其需求。只有选择赛孚耐产品，客户才能解决其所有的反盗版、IP 保护、许可实施和许可管理难题，同时提高整体盈利性、改善内部运行、保持具有竞争力的市场位置，同时改进其与客户和终端用户的关系。赛孚耐一直以来都不断适应新要求并不断推出新技术以满足不断发展的市场需求。赛孚耐遍及全球的 25,000 多名客户都深知选择了圣天诺就等于选择了在当今和未来不断实现业务成长的自由。

**北京明幻畅想科技有限公司**

咨询电话：010-51292955　或 400-605-8577