灵锐 I　　开发指南

2012年9月

Version 1.0.0.0

SENSELOCK
Reliable Professional Partner

许可协议

同意本许可协议的所有条款及此处包含的任何补充或特殊的许可条款是获得本产品许可的必要条件。如果您不同意此协议的所有条款，请在三天内将产品退还北京深思洛克软件技术股份有限公司。您对本软件的使用将表明您同意接受本协议中条款的约束。

1. 授予您使用许可权。您可以为了备份的目的而复制磁盘中的软件，可以为了将本产品集成到您的软件的目的，根据本产品的文档说明将我们提供的软件合并进您的程序中。

2. 除已按上述第一条被授权外，不可以复制、修改、逆向工程、分解或重组该产品的全部或部分，不可向他人销售、租借、许可、转让、分发全部或部分本产品或本协议授予的权利。

3. 保证在自产品交给您之日起的 12 个月内，在正常使用情况下，产品不会出现实质性的材料上和生产制造上的缺陷。北京深思洛克软件技术股份有限公司的全部责任和您能获得的全部补救措施为：可选择尝试更换或修理或其他补救措施。

4. 除了上述对本产品的原始购买者所提供的有限保证之外，不向任何人作任何其他的保证。对北京深思洛克软件技术股份有限公司的产品、性能或服务亦没有明示的或暗示的或其他任何形式的保证，包括但不仅仅限于商品的适销性和对特定用途的适用性。

5. 任何情况下，无论如何引起及依据何种责任理论，均不负担任何因使用或不能使用本产品造成的损失责任，包括：丢失数据、损失利润及其他特别的、偶然的、附随的、继发的或间接的损失。

6. 所有的产品，包括灵锐 I 设备、软件、文档、与本产品一并附送的其他材料及您制作的备份的所有权与版权均属于北京深思洛克软件技术股份有限公司。

7. 违反上述条款时，本协议的授权将自动终止。

联系深思洛克

公司名称：北京深思洛克软件技术股份有限公司

办公地点：北京市海淀区中关村南大街甲 6 号铸诚大厦 B 座 1201 室

邮　　编：100086

电　　话：+86-10-51581366

传　　真：+86-10-51581365

电子邮件：sense@sense.com.cn

网　　址：http://www.sense.com.cn

阅读指南

〖手册目标〗

本手册主要对北京深思洛克软件技术股份有限公司开发的灵锐 I 加密锁的使用进行说明，由于实际情况千变万化，本手册很难一次做到面面俱到，需要逐渐完善。

〖手册约定〗

手册中出现该标志的地方表示需要您引起高度重视，否则可能会引发严重的后果。

手册中出现该标志的地方表示需要您特别引起注意的内容。

目 录

第 1 章　产品简介

1.1 了解灵锐 I

灵锐 I 是北京深思洛克软件技术股份有限公司开发的一款加密锁，属于我公司的低端产品。灵锐 I 是基于硬件（加密锁）的软件防盗版系统，采用了国际标准的 AES 加密算法技术，您可以使用它来保护自己的软件，从而远离盗版的困扰。此外，灵锐 I 还提供了标准的 HMAC-SHA1 算法功能，可用于认证网络客户端的身份，实现身份认证功能。

1.2 灵锐 I 的特点

- **AES 高级加密算法**

  灵锐 I 支持 128 位 AES 高级加密算法，受保护的软件可以使用 AES 算法来验证连接到计算机上的灵锐 I 是否合法，这样就使软件与灵锐 I 设备紧密相连、不可分割。

  使用 AES 算法对软件进行保护的优势还在于保护软件的关键在于所使用的密钥，而不是算法本身的私密性，所以您只需严格控制密钥就可以实现可靠的验证。

◆ 我们为您分配的 AES 密钥是唯一的，并预存在设备的不可读写区域之中，即您的所有灵锐 I 设备的 AES 密钥都是相同的，各软件开发商的 AES 密钥不同。

- **大容量数据存储空间**

  灵锐 I 为您提供了 1920 字节的非易失性数据存储区，可以满足您同时保护多个软件的数据存储需要。

- **安全隧道技术**

  灵锐 I 利用 AES 算法建立了与设备通信的“安全隧道”，并使用了随机加扰措施，让破解者无法通过数据侦听来获得有效的信息，从而提高了软件保护的安全强度。

- **无需安装驱动**

  灵锐 I 设备遵循 USB 2.0 设备规范，并符合 HID 协议要求，在大多数常用操作系统上无需另外安装设备驱动程序，降低了部署和维护的难度。

- **全球唯一序列号**

  全球唯一序列号是灵锐 I 出厂时设定的硬件序列号，您可以利用它来实现软件与设备的绑定或者实现产品的跟踪与追溯。

- **高度集成技术**

  灵锐 I 的 CPU 内核、ROM、RAM、非易失性存储器这些关键部件全部在单一的硅片上集成制造，使其稳定性达到了前所未有的高度，从而最大程度的降低了由于坏锁和丢数问题所带来的风险。

- **HMAC-SHA1 算法**

  灵锐 I 支持 HMAC-SHA1 算法，可以实现“挑战-响应”方式的认证，从而保证用户的合法性。

1.3 灵锐 I 的工作原理

灵锐 I 提供了大容量的数据区，可以将重要的数据存放到加密锁的数据区中，当软件中用到这些数据时，使用我们提供的 API 函数来读取加密锁中的数据。这样，就可以实现将软件与加密锁绑定在一起，达到保护软件的效果。

灵锐 I 支持 AES 高级加密算法，利用该算法可以实现更加丰富的功能。除此之外，灵锐 I 还提供了标准的 HMAC-SHA1 算法功能，可用于实现身份认证。

1.4 名词解释

- **全球唯一序列号**

全球唯一序列号是灵锐 I 出厂时设定的硬件序列号，您可以利用它来实现软件与设备的绑定或者实现产品的跟踪与追溯。

- **开发商编号**

当您初次订购灵锐 I 产品时，我们会为您分配一个唯一的开发商编号，在以后的订购中，所有提供给您的灵锐 I 设备都会设置成和初次订购相同的编号。

- 所有试用版的灵锐 I，其开发商编号都是相同的。试用版的开发商编号是：十六进制：0x44454D4F（“DEMO”），十进制：1145392463。
- 您可以使用灵锐开发设置工具查看设备的开发商编号，工具中显示的开发商编号为十六进制表示。
- 您可以通过在 API 中指定开发商编号来识别您自己的灵锐 I 设备。

- **密码**

灵锐 I 使用密码来管理不同的权限，密码验证通过后即可获得相应的权限，共设置了三种密码，即管理密码、普通用户密码和认证密码。

➢ 管理密码

管理密码是您管理灵锐 I 设备时所使用的密码，管理密码拥有操作灵锐 I 设备的最高权限，验证通过后可以完成设置数据区、设置密钥以及重置密码等管理级别的操作。

➢ 普通用户密码

普通用户密码验证通过后，可以用于调用 AES 算法和读写数据区，拥有此密码的权限为对区块 0 可读可写，对区块 1～3 只读，关于数据区的详细内容请参考第 3.1 章节。

➢ 认证密码

认证密码验证通过后，可以用于调用 HMAC 算法进行身份认证操作和读写数据区，拥有此级密码的权限为对区块 0 可读可写，对区块 1～3 只读。

- 对于新版（V1.1.0）灵锐 I 设备，可以为管理密码、普通用户密码和认证密码设置 1～15 次不等的重试次数，对于旧版（V1.1.0 以前的版本）灵锐 I 设备，只可以为认证密码设置 1～15 次不等的重试次数。通过设置密码的重试次数，可以有效的防止穷举攻击。此外，还可以通过将重试次数设置为“-1”来禁用密码重试机制。一般情况下，我们建议为密码设置合适的重试次数，以提高安全性。
- 在软件发版前请务必将管理密码修改为一个秘密值，且一定不能忘记，并避免非授权人员接触。为了最大程度保障您软件的安全，我们没有保留任何可以对灵锐 I 进行解锁的接口，因此管理密码一旦被忘记，我们也无法帮您解锁。

- 所有密码的初始值都是字符串“12345678”。
- 所谓数据区“只读”是指对于普通用户密码的权限而言，对于管理密码的权限，所有数据区均可读可写。

- **密钥**

灵锐 I 的密钥有两种，即远程升级密钥和身份认证密钥。

➢ 远程升级密钥

远程升级密钥是验证远程升级包时使用的密钥，长度为 20 字节。

➢ 身份认证密钥

身份认证密钥用来鉴别终端用户的身份，长度为 20 字节，您需要为每把灵锐 I 设备设置不同的身份认证密钥，通过此密钥来鉴别终端用户的身份。

- 建议您为灵锐 I 设备设置远程升级密钥，以备将来需要远程升级功能时使用。

对于上述几个名词的其他属性，请参见表 1-1 中的内容。

表 1-1 名词的属性

名词	长度	初始值	唯一性	软件开发商是否可更改
全球唯一序列号	8 字节	各设备均不相同	全球唯一	不可更改
开发商编号	4 字节	出厂时设定	各软件开发商不同	不可更改
密码	8 字节	12345678	软件开发商自己设置	可更改
密钥	20 字节	软件开发商自己设置	软件开发商自己设置	可更改

第 2 章　灵锐 I 的工具

为了便于测试灵锐 I 硬件和开发应用程序，我们提供了灵锐开发设置工具和灵锐批量设锁工具，这几个工具存放在开发包的 Tools 目录中，其作用如表 2-1 中的内容所示。

表 2-1 灵锐 I 的工具

文件名	工具名称	作用
DevelTool.exe	灵锐开发设置工具	初始化灵锐 I 设备，如向数据区中写入数据、修改各种密码以及密钥等
LivBatchTool.exe	灵锐批量设锁工具	批量初始化灵锐 I 设备

2.1 灵锐开发设置工具

灵锐开发设置工具是最常用的工具软件，该工具主要用来初始化加密锁，基本上包括了对灵锐 I 设备所有功能的操作。

插入一把灵锐 I 设备，双击开发包 Tools 目录中的“DevelTool.exe”启动灵锐开发设置工具，其界面如图 2-1 所示。

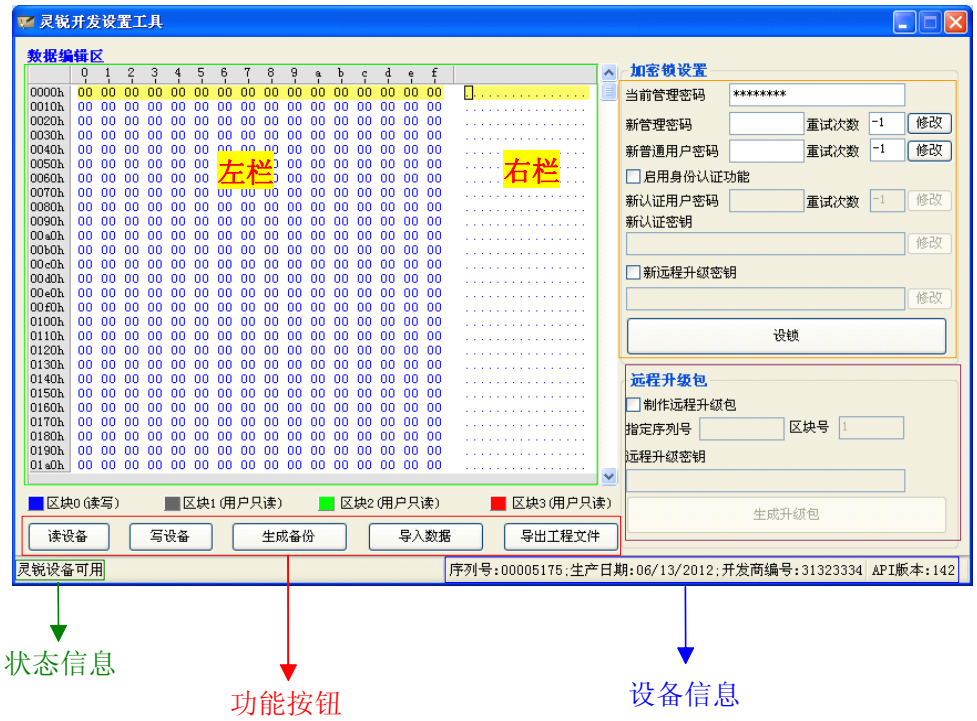

图 2-1 灵锐开发设置工具

从图中可以看出，工具启动后在状态信息栏显示“灵锐设备可用”，这表明工具连接设备成功。在设备信息栏显示的内容包括当前设备的序列号（全球唯一序列号）、生产日期、开发商编号以及 API 版本几项。工具的界面主要分为数据编辑区、功能按钮、加密锁设置和远程升级包几项，下面分别对这几项进行介绍。

一、数据编辑区

数据编辑区主要是对灵锐 I 设备的数据存储区进行编辑，在编辑区的下面显示了对各个数据区的说明，蓝色表示区块 0，灰色表示区块 1，绿色表示区块 2，红色表示区块 3。数据编辑区分为左右两栏，左栏是十六进制显示区，右栏是字符显示区，十六进制显示区中的横纵地址坐标对应设备内部唯一地址字节的数据。关于灵锐 I 数据存储区的详细内容请参考第 3.1 章节。数据区的编辑方法主要有以下两种：

（一）直接输入

数据编辑区分为左右两栏，左栏中可以直接输入 0～9 和 A～F 的十六进制数字，右栏中可以输入文本字符。比如直接在右栏中输入“senselock”，如图 2-2 所示，在右栏输入的同时左栏会自动填充对应的十六进制数，相应的在左栏中直接输入十六进制数时，右栏也会自动填充相应的文本字符。

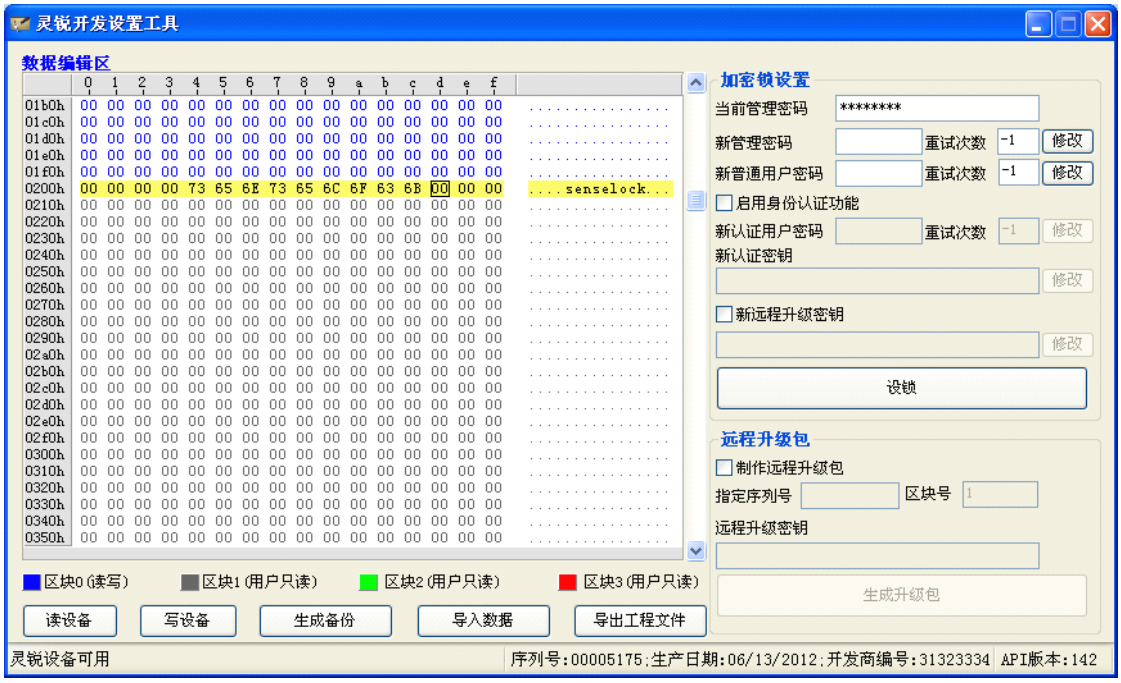

图 2-2 数据编辑区直接输入

（二）从剪切板粘贴

在要插入数据的位置上点击鼠标右键，在弹出的菜单中选择“粘贴”，就可将剪切板中的内容填充到当前数据区中，如图 2-3 所示。如果剪切板里的数据长度超过数据区边界，将弹出图 2-4 所示的对话框，此时数据无法保存，数据区的内容不会改变。另外，我们可以看到在弹出的菜单中还有一项“将全部数据区清零”，如果选择此项，会清除掉数据编辑区的所有内容。

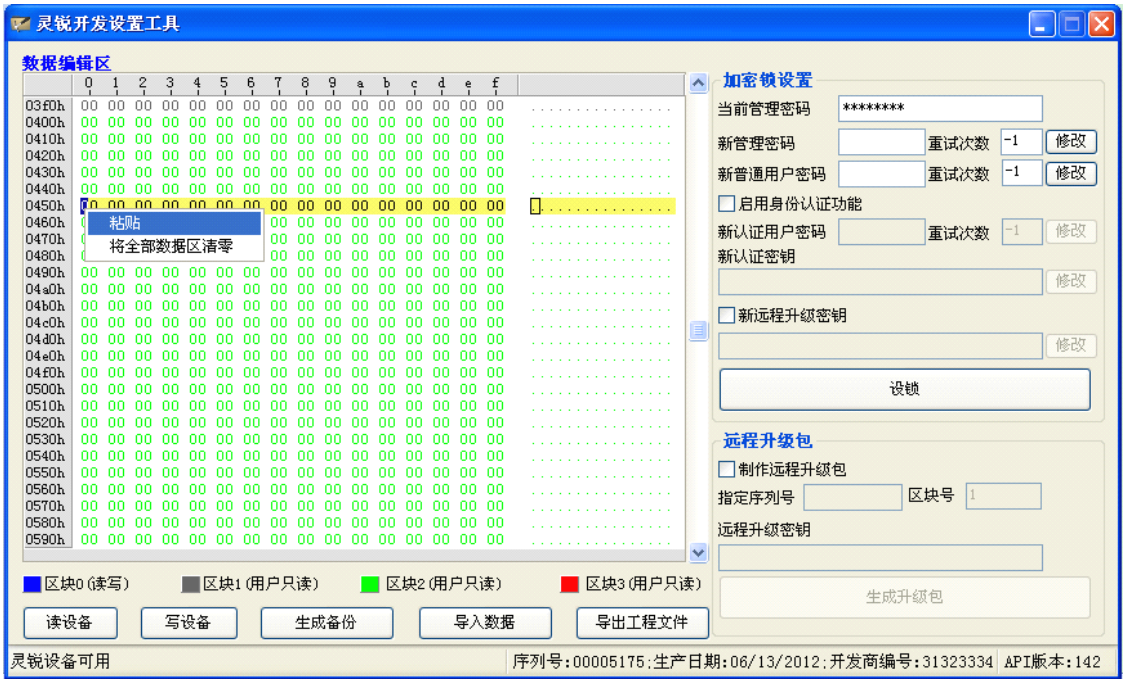

图 2-3 数据编辑区从剪贴板粘贴

图 2-4 错误提示

二、功能按钮

功能按钮主要包括读设备、写设备、生成备份、导入数据和导出工程文件几项。

- 读设备

点击“读设备”按钮，会将当前设备内的数据区内容读出，如果读取成功，除了会在数据编辑区显示读取的数据内容之外，还会在状态信息栏显示“读设备成功”，否则将提示相应的错误信息。

- 写设备

如果要将数据编辑区中的内容写入到当前设备中，则需要在编辑好数据之后点击“写设备”按钮。如果写入成功，会在状态信息栏显示“写设备成功”，否则将提示相应的错误信息。

- 生成备份

点击“生成备份”按钮可以将当前编辑的数据区内容保存到方案文件中，默认的文件名称为“data.bak”。如果导出成功，会在状态信息栏显示“生成备份成功!”，否则将提示相应的错误信息。

- 导入数据

点击“导入数据”按钮可以将之前保存的方案文件导入到工具中，如果导入成功，会在状态信息栏显示“导入备份成功!”，否则将提示相应的错误信息。

- 导出工程文件

此项主要用于灵锐 I 的批量生产功能中，此处不做介绍，详细内容请参考第 5 章。

三、加密锁设置

加密锁设置主要用于设置灵锐 I 设备的各种密码和密钥。

- 当前管理密码

输入当前设备的管理密码。

- 新管理密码

输入新的管理密码。

- 重试次数

输入管理密码的重试次数，输入范围是 1～15（十进制），输入“-1”表示禁用密码重试机制，输入完毕后点击“修改”按钮，如果修改成功，会在状态信息栏显示“修改管

理密码成功”，否则将提示相应的错误信息。

- 新普通用户密码

  输入新的普通用户密码。

- 重试次数

  输入普通用户密码的重试次数，输入范围是 1～15（十进制），输入“-1”表示禁用密码重试机制，输入完毕后点击“修改”按钮，如果修改成功，会在状态信息栏显示“修改普通用户密码成功”，否则将提示相应的错误信息。

- 启用身份认证功能

  如果要使用灵锐 I 的身份认证功能，则需要勾选此项，默认为不勾选，在不勾选的状态下，新认证用户密码、重试次数以及新认证密钥几项为不可用状态。

- 新认证用户密码

  输入新的认证用户密码。

- 重试次数

  输入认证用户密码的重试次数，输入范围是 1～15（十进制），输入“-1”表示禁用密码重试机制，输入完毕后点击“修改”按钮，如果修改成功，会在状态信息栏显示“修改认证用户密码成功”，否则将提示相应的错误信息。

- 新认证密钥

  输入新的认证密钥（16 进制），然后点击“修改”按钮，如果修改成功，会在状态信息栏显示“设置认证密钥成功”，否则将提示相应的错误信息。

- 新远程升级密钥

  如果要使用灵锐 I 的远程升级功能，则需要勾选此项，并在下面的文本框中输入远程升级密钥，然后点击“修改”按钮，如果修改成功，会在状态信息栏显示“设置升级密钥成功”，否则将提示相应的错误信息。

- 设锁

  您也可以在设置好各项内容之后不点击每项的“修改”按钮，而是在全部输入好之后，点击“设锁”按钮，此时会同时设置各项内容。如果设置成功，会在状态信息栏显示“设锁成功”，否则将提示相应的错误信息。

- 由于只有新版（V1.1.0）灵锐 I 设备支持为管理密码和普通用户密码设置重试次数，所以当插入旧版（V1.1.0 以前的版本）灵锐 I 设备时，新管理密码和新普通用户密码后面的重试次数将为不可用状态。

四、远程升级包

本章节对于此部分内容不做介绍，详细内容请参考第 4 章。

2.2 灵锐批量设锁工具

本章节对于此部分内容不做介绍，详细内容请参考第 5 章。

第 3 章　灵锐 I 编程指南

在 1.3 章节我们简单的介绍了灵锐 I 的工作原理，本章将详细的介绍灵锐 I 提供的各种功能及应用方法。

3.1 灵锐 I 的数据区

灵锐 I 共提供了 1920 字节的非易失性存储区，可以在断电之后长期保存数据。该存储区被划分为 4 个区块，分别是区块 0、区块 1、区块 2 和区块 3，这四个区块的划分情况请参考表 3-1。

表 3-1 灵锐 I 的数据区划分

数据区	起始地址	结束地址	容量（字节）
区块 0	0	511	512
区块 1	512	1023	512
区块 2	1024	1535	512
区块 3	1536	1919	384

- 在对数据区进行读写操作时，所有的区块都必须按照区块的大小整体读取或者写入，不能够跨区块读写。
- 在普通用户权限和认证权限下，区块 0 可读可写，而区块 1、区块 2 和区块 3 为只读。在管理权限下，所有的区块均可读写。
- 灵锐 I 数据存储区的写入次数限制为 1 万次，读取次数没有限制，所以请您务必合理安排写入操作。

3.2 使用灵锐 I 保护软件的基本方案

一、一般方式

对于灵锐 I 来说，一般的使用方式就是将其作为数据存储设备，您可以将软件运行过程中需要使用的关键数据存放到灵锐 I 的数据存储区，由于软件运行过程中需要使用这些数据，所以没有灵锐 I，软件就无法正常运行，这样就将软件和灵锐 I 紧密的结合到一起了。您也可以使用灵锐 I 的可读写数据区（区块 0），将软件运行过程中的临时数据存储在这里，这样也可以增加软件与灵锐 I 结合的紧密程度。

二、使用 AES 算法

除了使用灵锐 I 的存储功能保护软件之外，还可以利用 AES 算法使您的软件与灵锐 I 结合的更加紧密，我们建议您使用 AES 算法来提高软件保护的强度。

AES 是一种先进的加密算法，它的特点在于安全并不依赖于算法本身，而是依赖于所使用的密钥，所以只要不知道密钥，就没有办法去完全仿真整个计算过程。灵锐 I 内置了 128 位的 AES 算法，对应输入输出数据的长度是 16 字节，这是一个足够长的数据，可以避免数据侦听和穷举攻击。

利用灵锐 I 的 AES 算法实现验证的过程可以有多种方式，最常见的方式是先计算好足够多的输入/输出数据对并存储于软件中，软件运行时随机抽取部分数据对，与灵锐 I 的计算结果进行比对，只有计算结果相同，才认为灵锐 I 是合法的，这样可使软件与灵锐 I 结合的更加紧密。此外，您还可以利用 AES 算法对部分数据进行加密，当软件运行时再使用灵锐 I 进行解密，这样也可以使软件与灵锐 I 结合的更加紧密。

3.3 软件中使用 API 访问灵锐 I

您可以使用我们提供的灵锐开发设置工具对加密锁进行设置，也可以使用 API 函数完成加密锁的初始化。本章节主要介绍在软件中使用我们提供的 API 函数访问灵锐 I 的流程以及需要使用的头文件和库文件等，然后通过范例向您展示在软件中如何使用 API 函数访问灵锐 I 设备。

3.3.1 软件中使用 API 访问灵锐 I

一、头文件和库文件

在软件中使用 API 访问灵锐 I，需要使用我们提供的头文件和库文件。关于头文件和库文件的位置以及说明如表 3-2 所示。

表 3-2 头文件和库文件

	名称	位置	说明
头文件	living1.h	API\Win_X86\Visual Studio\Include	无
库文件	略	开发包\API	目录中包含了不同语言的库文件

二、软件访问灵锐 I 的流程

软件访问灵锐 I 分为管理权限访问和普通用户权限访问。管理权限是软件开发商拥有的权限，该权限下访问灵锐 I，需要先校验管理密码，校验通过后对锁的操作没有任何限制。普通用户权限是最终用户所拥有的权限，该权限下访问灵锐 I，需要先校验普通用户密码，校验通过后允许的操作包括读数据区、向区块 0 中写入数据、对数据进行加解密以及升级升级包。

管理权限访问灵锐 I 的流程如图 3-1 所示。

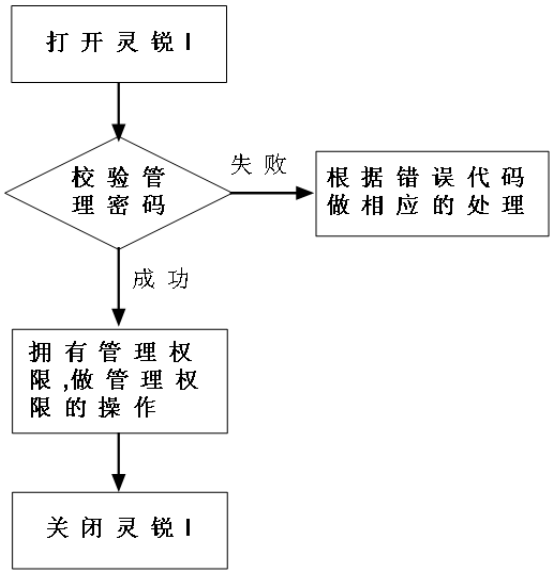

图 3-1 管理权限访问灵锐 I

普通用户权限访问灵锐 I 的流程如图 3-2 所示。

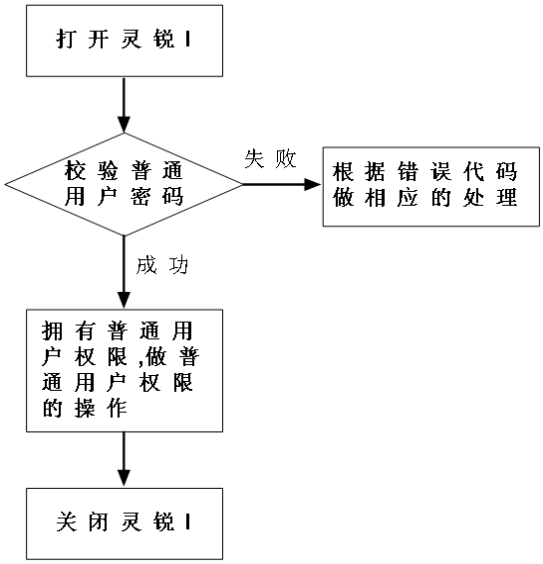

图 3-2 普通用户权限访问灵锐 I

3.3.2 范例

本章节将通过范例来介绍如何在软件中使用 API 访问灵锐 I 设备。

一、管理权限访问灵锐 I 设备

在 VC6 中新建一个 WIN32 控制台程序工程（Win32 Console Application），将图 3-3 所示的代码添加到 VC6 工程中。在本例中，我们假定将工程保存为 C:\livdemo\devep\demo1.dsp，图 3-3 所示代码保存为 C:\livdemo\devep\pc_demo1.c。此范例基本上包括了管理权限下的大部分操作，如更改密码、设置密钥以及向区块 1 写入数据等。

```
#include <windows.h>
#include <stdio.h>
#include <string.h>
#include "living1.h"

int main(int argc,char* argv[])
{
    int handle;
    int res;
    unsigned char data[512]="senselock";
```

```
unsigned char key[20]=
{0x0a,0x0a,0x0a,0x0a,0x0a,0x0a,0x0a,0x0a,0x0a,0x0a,0x0a,0x0a,0x0a,0x0a,0x0a,0x0a,0x0a,0x0a,0x0a,0x0a,
0x0a,0x0a,0x0a};

/*打开灵锐 I*/
res=LIV_open(0x31323334,0,&handle);
if(res==LIV_SUCCESS)
{
    printf("Open success!\n");
}
else
{
    printf("Open failed! Error code=%d\n",res);
    return 1;
}

/*校验管理密码*/
res=LIV_passwd(handle,0,(unsigned char *)"12345678");
if(res==LIV_SUCCESS)
{
    printf("Verify manager password success!\n");
}
else
{
    printf("Verify manager password failed! Error code=%d\n",res);
    LIV_close(handle);
    return 1;
 }

/*更改普通用户密码*/
res=LIV_set_passwd(handle,1,(unsigned char *)"11111111",-1);
if(res==LIV_SUCCESS)
{
    printf("Set user password success!\n");
}
else
{
    printf("Set user password failed! Error code=%d\n",res);
    LIV_close(handle);
    return 1;
 }
```

```
    /*设置远程升级密钥*/
    res=LIV_set_key(handle,0,key);
    if(res==LIV_SUCCESS)
    {
        printf("Set update key success!\n");
    }
    else
    {
        printf("Set update key failed! Error code=%d\n",res);
        LIV_close(handle);
        return 1;
     }

    /*向区块 1 中写入数据*/
    res=LIV_write(handle,1,data);
    if(res==LIV_SUCCESS)
    {
        printf("Write success!\n");
    }
    else
    {
        printf("Write failed! Error code=%d\n",res);
        LIV_close(handle);
        return 1;
     }

    /*关闭灵锐 I*/
    res = LIV_close(handle);
    if(res==LIV_SUCCESS)
    {
        printf("Close success!\n\n");
    }
    else
    {
        printf("Close failed! Error code=%d\n",res);
        return 1;
    }
    return 0;
}
```

图 3-3 管理权限访问灵锐 I

在编写此程序时，需要将我们提供的头文件和库文件拷贝到工程目录中，以便在工程中引用。从程序中可以看出，首先需要将头文件“living1.h”包含进来，然后调用 API 函数 LIV_open 打开灵锐 I 设备，并校验管理密码，然后做一系列管理权限的操作。在本例中涉及到的管理权限操作包括更改普通用户密码、设置远程升级密钥以及向区块 1 中写入数据几项。完成这几项操作都是通过调用我们提供的 API 函数来实现的，有关这几个 API 函数的详细介绍请参考《灵锐 I V1.0.0.0 函数参考》第 1 章。

- 在测试本范例时，需要将“LIV_open(0x31323334,0,&handle);”的第一个参数替换成您自己锁的开发商编号。

- 无论在什么情况下，都不能在最终发行的软件中包含管理权限访问灵锐 I 设备的代码，请务必在加密锁发行之前将管理密码修改为秘密值，且一定要牢记。
- 由于管理权限访问的代码中可能包含了管理密码的“明文形式”，所以要保证管理权限访问代码的保密性。

二、普通用户权限访问灵锐 I 设备

在 VC6 中新建一个 WIN32 控制台程序工程（Win32 Console Application），将图 3-4 所示的代码添加到 VC6 工程中。在本例中，我们假定将工程保存为 C:\livdemo\user\demo2.dsp，图 3-4 所示代码保存为 C:\livdemo\user\pc_demo2.c。此范例基本上包括了普通用户权限下的大部分操作，如获取设备信息、获取开发商编号以及写入数据到可读写数据区等。

```
#include <windows.h>
#include <stdio.h>
#include <string.h>
#include "living1.h"

int main(int argc,char* argv[])
{
```

```
int handle;
int res;
unsigned char data[512];
unsigned char key[20]=
{0x0a,0x0a,0x0a,0x0a,0x0a,0x0a,0x0a,0x0a,0x0a,0x0a,0x0a,0x0a,0x0a,0x0a,0x0a,0x0a,0x0a,0x0a,0x0a,0x0a,
0x0a,0x0a,0x0a};

/*打开灵锐 I*/
res=LIV_open(0x31323334,0,&handle);
if(res==LIV_SUCCESS)
{
    printf("Open success!\n");
}
else
{
    printf("Open failed! Error code=%d\n",res);
    return 1;
}

/*校验普通用户密码*/
res=LIV_passwd(handle,1,(unsigned char *)"12345678");
if(res==LIV_SUCCESS)
{
    printf("Verify user password success!\n");
}
else
{
    printf("Verify user password failed! Error code=%d\n",res);
    LIV_close(handle);
    return 1;
 }

/*读取区块 1 的数据*/
res=LIV_read(handle,1,data);
if(res==LIV_SUCCESS)
{
    printf("Read success!\nRead data: %s\n",data);
}
else
{
```

```
            printf("Read failed! Error code=%d\n",res);
            LIV_close(handle);
            return 1;
        }

    /*关闭灵锐 I*/
    res = LIV_close(handle);
    if(res==LIV_SUCCESS)
    {
            printf("Close success!\n\n");
    }
    else
    {
            printf("Close failed! Error code=%d\n",res);
            return 1;
    }
    return 0;
}
```

图 3-4 普通用户权限访问灵锐 I

在编写此程序时，需要将我们提供的头文件和库文件拷贝到工程目录中，以便在工程中引用。从程序中可以看出，首先需要将头文件“living1.h”包含进来，然后调用 API 函数 LIV_open 打开灵锐 I 设备，并校验普通用户密码，然后做一系列普通用户权限的操作。在本例中涉及到的普通用户权限操作只有读取数据区的内容。完成这些操作都是通过调用我们提供的 API 函数来实现的，有关这几个 API 函数的详细介绍请参考《灵锐 I V1.0.0.0 函数参考》第 1 章。

◆ 在测试本范例时，需要将“LIV_open(0x31323334,0,&handle);”的第一个参数替换成您自己锁的开发商编号。

三、使用 AES 算法保护软件的范例

此处演示的是使用 AES 算法保护软件最常用的方式。在 VC6 中新建一个 WIN32 控制台程序工程（Win32 Console Application），将图 3-5 所示的代码添加到 VC6 工程中。在本例中，我们假定将工程保存为 C:\livdemo\aes\demo3.dsp，图 3-4 所示代码保存为 C:\livdemo\aes\pc_demo3.c。

```
#include <windows.h>
#include <stdio.h>
#include <string.h>
#include <time.h>
#include "living1.h"
int main(int argc,char* argv[])
{
    int handle;
    int res,rnd;
    unsigned char buffer[16];

    /*输入/输出对*/
    unsigned char
in[3][16]={{0x00,0x01,0x02,0x03,0x04,0x05,0x06,0x07,0x08,0x09,0x0a,0x0b,0x0c,0x0d,0x0e,0
x0f},{0x10,0x11,0x12,0x13,0x14,0x15,0x16,0x17,0x18,0x19,0x1a,0x1b,0x1c,0x1d,0x1e,0x1f},{
0x20,0x21,0x22,0x23,0x24,0x25,0x26,0x27,0x28,0x29,0x2a,0x2b,0x2c,0x2d,0x2e,0x2f}};
    unsigned char
out[3][16]={{0x66,0xe6,0x44,0xbf,0xe7,0x6c,0x63,0x75,0xf2,0x7b,0x3a,0x88,0xfd,0xdc,0x13,0
xcb},{0xcf,0x3e,0x0c,0x15,0xaf,0x14,0x9d,0xe6,0x65,0x98,0x5f,0x99,0x15,0x2b,0x16,0x0d},{0
xb9,0x00,0xb4,0x04,0xa8,0x6f,0xb9,0x6f,0xa3,0x84,0xa4,0x7b,0xfa,0x92,0x45,0xaf}};

    /*初始化随机数发生器*/
    srand(time(NULL));

    /*打开灵锐 I*/
    res = LIV_open(0x31323334,0,&handle);
    if(res==LIV_SUCCESS)
    {
        printf("Open success!\n");
    }
    else
    {
```

```
        printf("Open failed! Error code=%d\n",res);
        return 1;
}

/*校验普通用户密码*/
res=LIV_passwd(handle,1,(unsigned char *) "12345678");
if(res==LIV_SUCCESS)
{
        printf("Verify user password success!\n");
}
else
{
        printf("Verify user password failed! Error code=%d\n",res);
        LIV_close(handle);
        return 1;
}

/*随机选取一个码表数据*/
rnd=rand()%(sizeof(in)/(16*2));

/*调用灵锐 I 进行 AES 运算*/
res=LIV_encrypt(handle,in[rnd*(16*2)],buffer);
if(res==LIV_SUCCESS)
{
        printf("Encrypt success!\n");
}
else
{
        printf("Encrypt failed! Error code=%d\n",res);
        LIV_close(handle);
        return 1;
}

/*验证算法结果*/
if(memcmp(buffer,out[rnd*(16*2)],16))
{
         printf("Verify failed!\n");
         LIV_close(handle);
         return 1;
}
```

```
    printf("Verify success!\n");

    /*关闭灵锐 I*/
    res = LIV_close(handle);
    if(res==LIV_SUCCESS)
    {
        printf("Close success!\n\n");
    }
    else
    {
        printf("Close failed! Error code=%d\n",res);
        return 1;
    }
    return 0;
}
```

图 3-5 使用 AES 算法保护软件

- 在测试本范例时，需要将“LIV_open(0x31323334,0,&handle);”的第一个参数替换成您自己锁的开发商编号。
- 由于各软件开发商的灵锐 I 设备内置的 AES 密钥不同，所以在测试本范例时请更新范例中的输入/输出对。

3.4 灵锐 I 的其他应用

灵锐 I 支持 HMAC-SHA1 算法，能够实现“挑战-响应”方式的认证，可以取代传统的“用户名-密码”的方式，实现更加可靠的身份认证功能。其认证原理是：预先在灵锐 I 设备中存放密钥 Kn，认证时服务器端发送随机数（挑战）给客户端的灵锐 I 设备，并验证设备返回的计算结果（响应）是否是由此密钥 Kn 计算出来的，计算结果正确就认为客户端拥有密钥 Kn，此时就可认为客户端是密钥 Kn 对应的客户。其实现原理如图 3-6 所示。

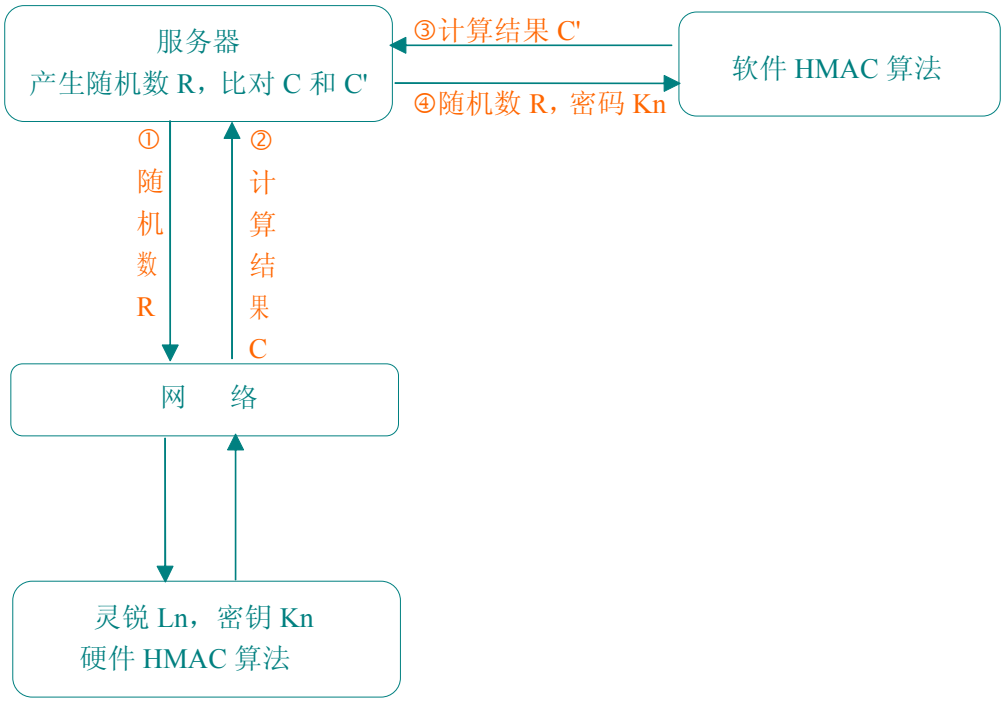

图 3-6 身份认证的原理

第 4 章　灵锐 I 的远程升级功能

4.1 远程升级的原理

如果您的软件和灵锐 I 设备已经发给了最终用户，但是想修改数据存储区的内容，此时不需要最终用户将锁寄回给您进行修改，可以使用灵锐 I 的远程升级功能来实现。灵锐 I 的远程升级功能是基于标准 HMAC 算法的签名过程，以保证升级数据的完整性和合法性。您需要先在灵锐 I 中设置远程升级密钥，生成升级包时使用此密钥对升级包数据进行数字签名，而灵锐 I 只有在验证签名成功之后才会升级内部的数据区，这样升级包在传输过程中有任何变化都会被检测到，确保了升级过程的安全。灵锐 I 的远程升级功能是一种简单的远程设备维护方法，大大减少了您的维护成本。

- 灵锐 I 的远程升级功能只能对区块 1～区块 3 中的某一区块进行整体、覆盖式升级，如果需要升级多个区块，则需要分别签发相应的升级包。
- 在使用远程升级功能时，您需要预先为灵锐 I 设备设置远程升级密钥，签发升级包和升级升级包时都需要使用此密钥。
- 签发升级包时需要最终用户提供被升级灵锐 I 的全球唯一序列号，以实现升级范围的控制。
- 如果要使用灵锐 I 的远程升级功能，则需要保留区块 1 ～区块 3 的前 4 个字节，即需要将前 4 个字节清零，且保证以后的升级包前 4 个字节均为 0x00（十六进制）。
- 在升级过程中请不要拔掉锁。

4.2 生成升级包

使用灵锐 I 的远程升级功能需要先生成升级包，升级包的内容就是更新后的数据，最终用户使用此升级包来更新灵锐 I 数据存储区的内容。

一、使用工具实现

打开灵锐开发设置工具，如图 4-1 所示。首先编辑数据区的内容，即设置需要更新的数据，比如设置区块 1 的数据，然后在“远程升级包”部分输入待升级设备的全球唯一序列号、区块号以及远程升级密钥。

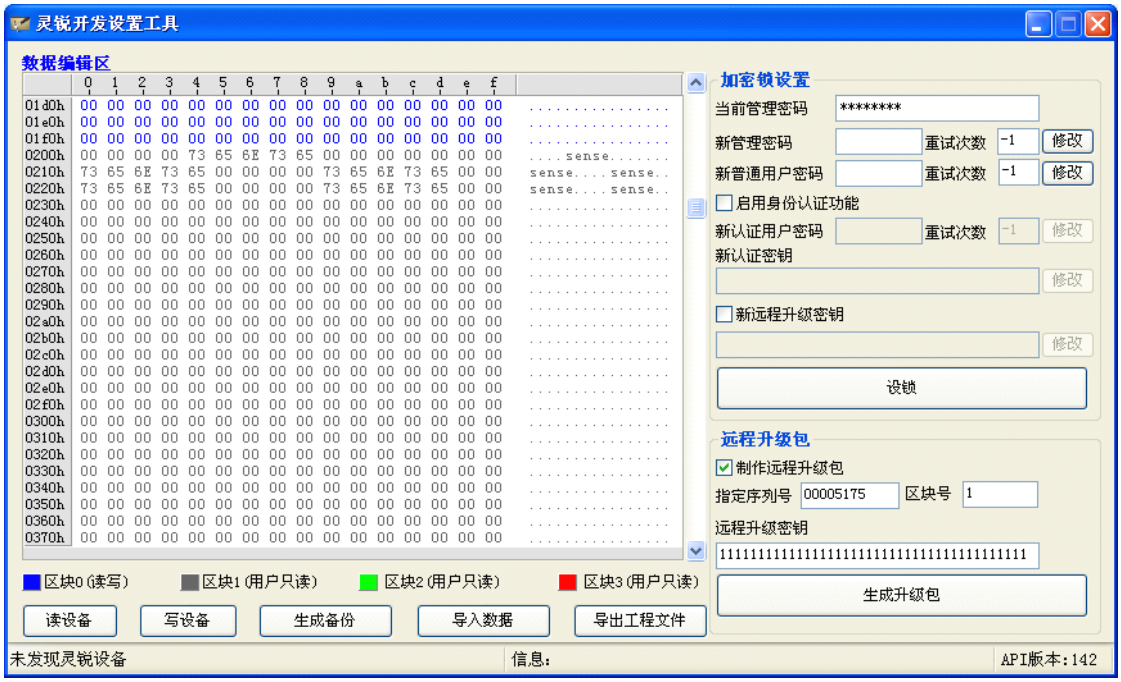

图 4-1 生成升级包

全部设置好之后，点击“生成升级包”按钮，选择保存路径并输入升级包的名称，默认名称格式为“SN_全球唯一序列号_block_区块号”。如果生成成功，会在状态信息栏显示“生成区块 1 升级包成功”，否则将提示相应的错误信息。

- 生成升级包时不需要插入灵锐 I 设备。
- 生成升级包时使用的远程升级密钥必须和预先为待升级设备设置的远程升级密钥相同，否则将不能升级成功。

二、使用 API 方式实现

图 4-2 所示代码是使用 API 的方式生成升级包。有关 API 函数的详细介绍请参考《灵锐 I V1.0.0.0 函数参考》第 1 章。

```
#include <windows.h>
#include <stdio.h>
#include "living1.h"

unsigned long WriteFile1(char *lpszFilePath,unsigned char *lpszContext,unsigned long ulLen)
{
	FILE *hFileHandle=NULL;
	hFileHandle=fopen(lpszFilePath,"a+b");
	if(!hFileHandle)
	{
		return 0;
	}
	fseek(hFileHandle,0,SEEK_END);
	fwrite(lpszContext,ulLen,1,hFileHandle);
	fclose(hFileHandle);
	return 0;
}

int main(int argc, char* argv[])
{
	int  res=0;
	int  iX=0;
	BYTE  bUpdateKey[20];
	BYTE  bUpdateData[512];
	BYTE  bPkg[550];
	BYTE  *bSerialNum =(BYTE*)"00005175"; //待升级设备的全球唯一序列号
	char	bFilePath[]="c:\\a.data";

	/*初始化远程升级密钥*/
	memset(bUpdateKey,0x0a,sizeof(bUpdateKey));

	/*初始化更新的数据*/
	memset(bUpdateData,0x00,sizeof(bUpdateData));
	for(iX=4;iX<sizeof(bUpdateData);iX++)
	{
```

```
        bUpdateData[iX]=iX;
    }
    memset(bPkg,0x00,sizeof(bPkg));

    /*生成升级包*/
    res=LIV_gen_update_pkg(bSerialNum,1,bUpdateData,bUpdateKey,bPkg);
    if(res==LIV_SUCCESS)
    {
        printf("Generate update package success!\n");
    }
    else
    {
        printf("Generate update package failed! Error code=%d\n",res);
        return 0;
    }

    WriteFile1(bFilePath,bPkg,sizeof(bPkg));
    return 0;
}
```

图 4-2 生成升级包

- 生成升级包时不需要插入灵锐 I 设备。
- 在测试本范例时，需要将*bSerialNum =(BYTE*)"00005175";中的全球唯一序列号更改为您待升级设备的序列号。
- 生成升级包时使用的远程升级密钥必须和待升级设备设置的远程升级密钥相同，否则将不能升级成功。

4.3 升级升级包

升级包生成完毕之后，最终用户就可以使用这个升级包升级灵锐 I 的数据区了。您除了给最终用户发送升级包之外，还需要发升级的工具，在我们的开发包中未提供升级升级包的工具，但提供了相关的 API 函数，您可以自己制作升级工具。

图 4-3 所示代码是使用 API 的方式升级升级包。有关 API 函数的详细介绍请参考《灵锐 I V1.0.0.0 函数参考》第 1 章。

```
#include <windows.h>
#include <stdio.h>
#include "living1.h"
BOOL ReadFile1(char *pucFilePath,unsigned long *pulFileLength,unsigned char
*pucFileContent)
{
    FILE *hFileHandle=NULL;
    unsigned long ulRead = 0;
    hFileHandle=fopen((const char*)pucFilePath,"rb");
    if(hFileHandle==NULL)
    {
        return FALSE;
    }
    fseek(hFileHandle,0,SEEK_END);
    *pulFileLength=ftell(hFileHandle);
    fseek(hFileHandle,0,SEEK_SET);
    if(pucFileContent==NULL)
    {
        fclose(hFileHandle);
        return FALSE;
    }
    ulRead=fread(pucFileContent,1,*pulFileLength,hFileHandle);
    fclose(hFileHandle);
    if(ulRead!=*pulFileLength)
    {
        return FALSE;
    }
    return TRUE;
}

int main(int argc,char* argv[])
```

```
{
    int  handle=0;
    int  res=0;
    unsigned long ulFileBufferLen=0;
    BYTE bPkg[550];
    char pkg_path[]="c:\\a.data";

    /*打开灵锐 I*/
    res=LIV_open(0x31323334,0,&handle);
    if(res==LIV_SUCCESS)
    {
        printf("Open success!\n");
    }
    else
    {
        printf("Open failed! Error code=%d\n",res);
        return 0;
    }

    /*校验普通用户密码*/
    res=LIV_passwd(handle,1,(BYTE *)"12345678");
    if(res==LIV_SUCCESS)
    {
        printf("Verify user password success!\n");
    }
    else
    {
        printf("Verify user password failed! Error code=%d\n",res);
        LIV_close(handle);
        return 1;
    }

    ReadFile1(pkg_path,&ulFileBufferLen,NULL);
    ReadFile1(pkg_path,&ulFileBufferLen,bPkg);

    /*升级升级包*/
    res=LIV_update(handle,bPkg);
    if(res==LIV_SUCCESS)
    {
        printf("Update success!\n");
    }
    else
```

```
    {
        printf("Update failed! Error code=%d\n",res);
        LIV_close(handle);
        return 1;
    }

    /*关闭灵锐 I*/
    res = LIV_close(handle);
    if(res==LIV_SUCCESS)
    {
        printf("Close success!\n\n");
    }
    else
    {
        printf("Close failed! Error code=%d\n",res);
        return 1;
    }
    return 0;
}
```

图 4-3 升级升级包

- 在测试本范例时，需要将“LIV_open(0x31323334,0,&handle);”的第一个参数替换成您自己锁的开发商编号。
- 确保待升级锁的全球唯一序列号是生成升级包时指定的序列号，否则升级会失败。

第 5 章　灵锐 I 的批量生产

当确定了存储到灵锐 I 设备中的数据之后，需要对灵锐 I 的数据区进行设置，我们之前介绍的方法都是插入一把灵锐 I 设备，然后使用灵锐开发设置工具进行初始化的操作，但当锁的数量很多时，这样做显然很不方便，灵锐批量设锁工具解决的就是这个问题，您可以利用此工具来批量初始化灵锐 I 设备，可以批量初始化的内容包括数据区、各种密码、密钥以及相应的重试次数。

一、导出方案文件

在批量初始化设备之前，需要先导出方案文件。插入一把灵锐 I 设备，双击开发包 Tools 目录下的“DevelTool.exe”启动灵锐开发设置工具，如图 5-1 所示。

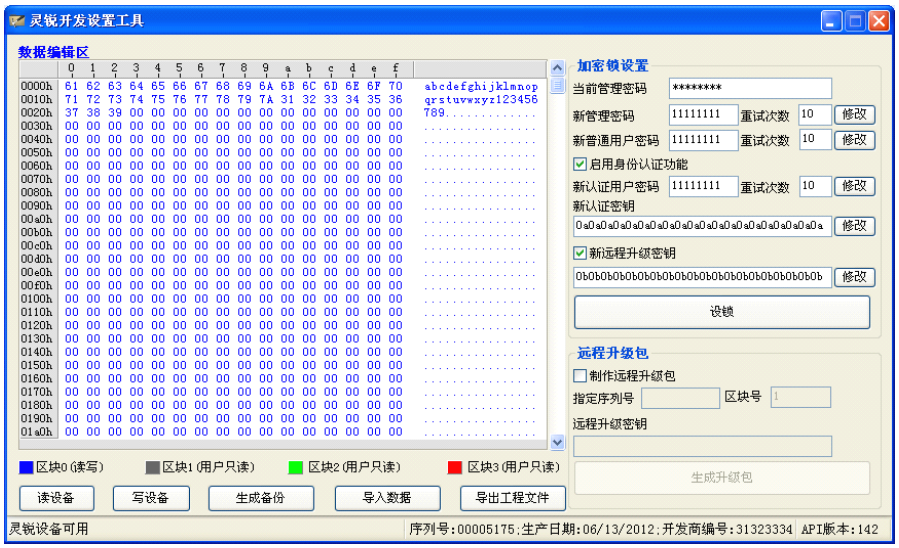

图 5-1 灵锐开发设置工具

首先需要设置数据区、各种密码、密钥以及相应的重试次数等内容，然后点击“导出工程文件”按钮，此时会弹出“另存为”对话框，选择保存路径并输入文件名称后会出现图 5-2 所示界面。

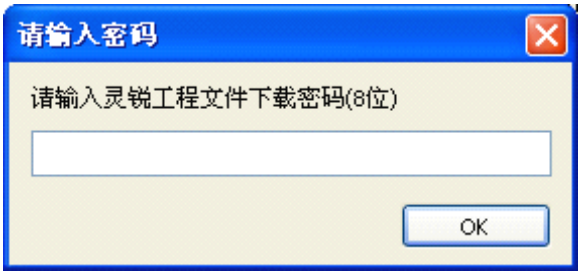

图 5-2 输入下载密码

在此界面中输入 8 字节的工程文件下载密码并点击“OK”按钮，当使用灵锐批量设锁工具打开工程文件时，会用到此密码。如果导出成功，会出现图 5-3 所示的界面，否则将提示相应的错误信息。

图 5-3 导出工程文件成功

◆ 导出工程文件时，必须设置“新管理密码”，其他密码以及密钥可根据实际需要进行设置。

二、批量初始化设备

方案文件导出成功后，即可进行设备的批量初始化，双击开发包 Tools 目录下的“LivBatchTool.exe”启动灵锐批量设锁工具，如图 5-4 所示。

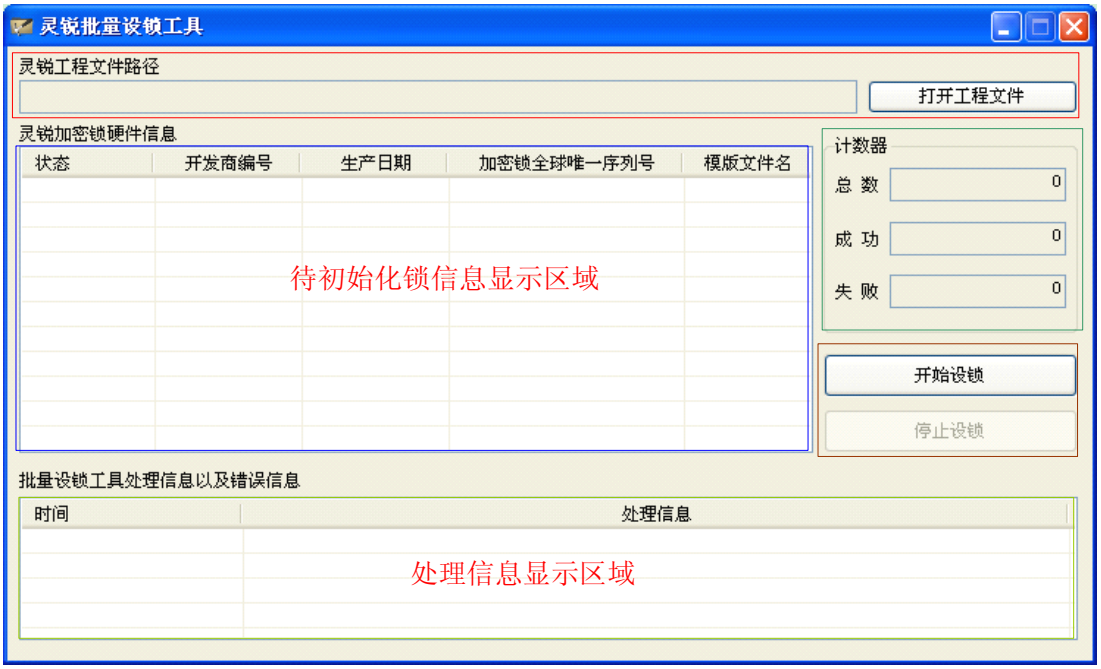

图 5-4 灵锐批量设锁工具

- 打开工程文件

  点击此按钮，选择需要导入的工程文件。

- 开始设锁

  点击此按钮，将开始批量初始化设备。

- 停止设锁

  点击此按钮，将停止初始化的操作。

- 待初始化信息显示区域

  - 状态

    显示初始化的当前状态，共有可用、设锁完成和设锁失败三种状态。

  - 开发商编号

    显示待初始化设备的开发商编号。

  - 生产日期

    显示待初始化设备的生产日期。

  - 加密锁全球唯一序列号

    显示待初始化设备的全球唯一序列号。

  - 模板文件名

    显示当前导入工程文件的名称。

- 初始化锁数量显示区域

  - 总数

    显示初始化设备的总数量。

  - 成功

    显示初始化设备成功的总数量。

  - 失败

    显示初始化设备失败的总数量。

- 处理信息显示区域

  - 时间

    显示初始化设备的时间。

  - 处理信息

    显示初始化设备时的处理情况以及错误等信息。

比如使用之前导出成功的工程文件“project.livf”来批量初始化设备，首先点击“打

开工程文件”按钮打开“project.livf”，将出现图 5-5 所示的界面，在此界面中输入工程文件的下载密码，该密码是图 5-2 中所设置的密码，输入完毕后点击“OK”按钮，会出现图 5-6 所示的提示信息，点击“确定”即可，此时会回到灵锐批量设锁工具的主界面，界面中会多出一些信息，如图 5-7 所示。

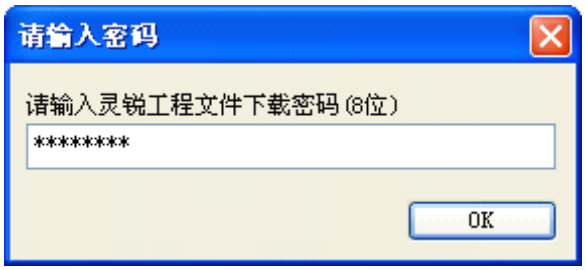

图 5-5 输入密码

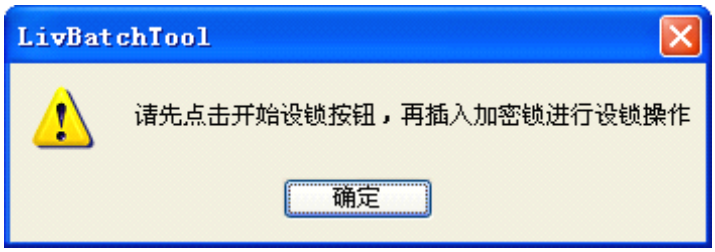

图 5-6 提示信息

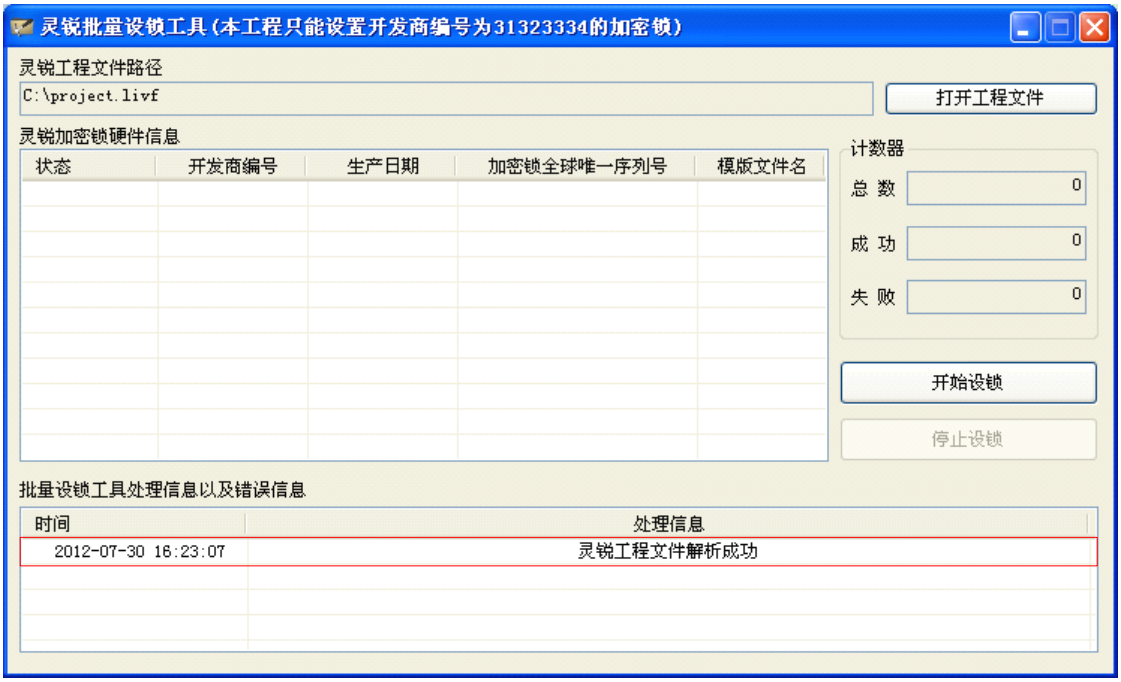

图 5-7 灵锐批量设锁工具

工程文件导入成功之后，就可对设备进行批量初始化了。点击“开始设锁”按钮，然后插上需要初始化的设备，初始化的过程如图 5-8 和图 5-9 所示。如果希望停止设备的初始化，则点击“停止设锁”即可。

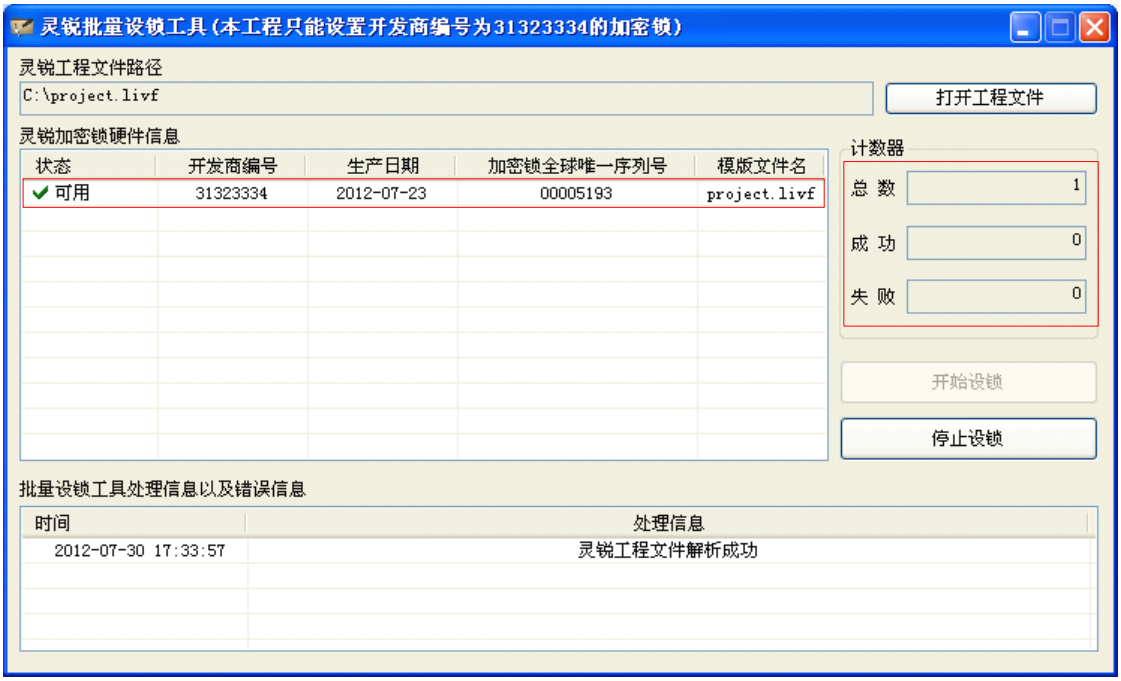

图 5-8 批量初始化设备

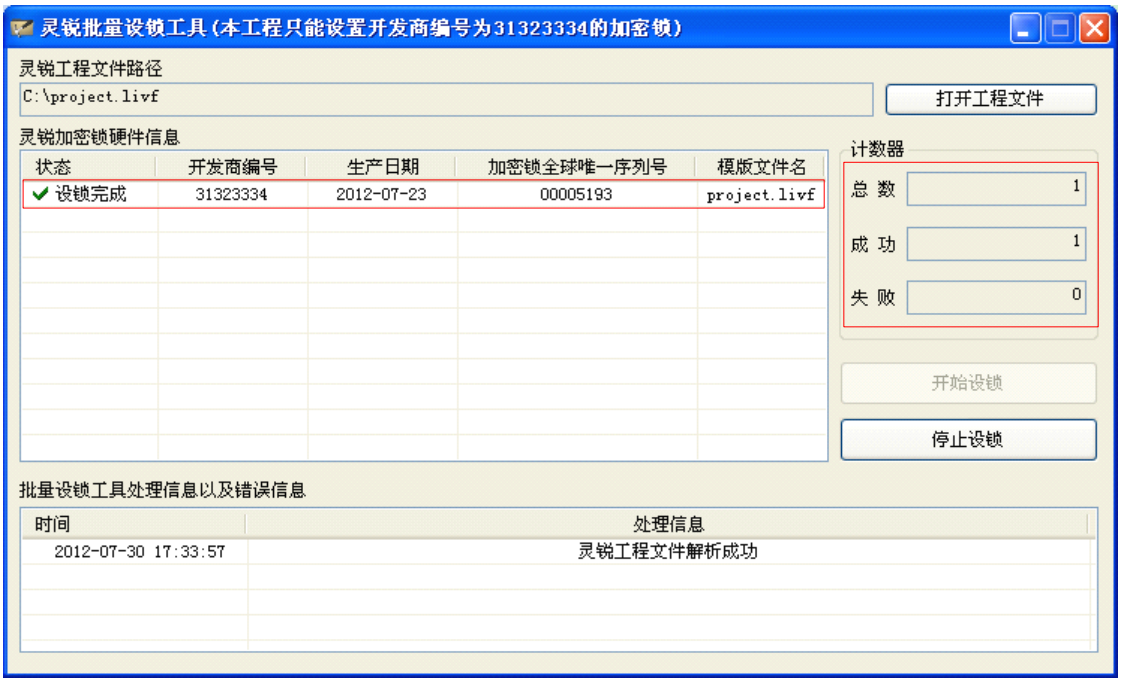

图 5-9 批量初始化设备

当批量初始化设备完成后，会在当前导入方案文件的同目录下生成扩展名为“.ini”和“.log”的两个文件，其中 ini 文件主要用于存储当前方案文件的路径、初始化设备的总数量、成功的总数量以及失败的总数量，log 文件主要用于存储所有初始化成功以及初始化失败设备的全球唯一序列号。

- 待初始化灵锐 I 设备的开发商编号与导出工程文件时所使用设备的开发商编号必须相同，否则将初始化失败。当前打开工程文件所对应的开发商编号可以通过灵锐批量设锁工具的标题栏查看。
- 灵锐批量设锁工具最多可同时对 127 个设备进行初始化。

附录 I

常见问题解答

1. 如何查看灵锐I设备的开发商编号？

答：可以通过灵锐开发设置工具查看（工具底端右侧）。

2. 软件最终用户的电脑上同时插有不同软件开发商提供的灵锐I设备，它们之间会不会冲突？

答：在同一台电脑上可以同时使用多把灵锐I，这是因为我们为不同软件开发商的灵锐I设置了不同的开发商编号，您只需在打开设备时指定开发商编号就可以识别自己的锁。

3. 我们有不同软件都使用灵锐I设备，同时插在一台电脑上使用，它们之间会不会冲突？

答：相同开发商编号的灵锐I可以在一台电脑上同时使用，您只需在调用“LIV_open”函数时使用不同的index值就可遍历所有的灵锐I设备。

4. 如何更改灵锐I的各级密码？

答：可以通过灵锐设置工具修改各级密码，也可以通过API修改各级密码。如果使用API修改，则在成功校验管理密码后，可以通过“LIV_set_passwd”函数修改所有的密码，在成功校验认证密码后，可以通过“LIV_change_passwd”函数修改自身的认证密码。

5. 向灵锐I设备的数据存储区写入数据时需要验证哪个密码？

答：区块0可以在成功校验任意密码后写入数据，区块1、区块2和区块3只能在成功校验管理密码后写入数据。

6. 如何写超过512字节的数据？

答：灵锐I是以区块划分数据区的，区块0、区块1和区块2的大小各为512字节，区块3的大小为384字节，数据区需要整体读取或者写入，如果读写超过区块边界的数据，则必须对不同的区块分别进行操作。

7. 哪些操作需要打开设备？打开设备后是否一定要关闭设备？

答：除了获取软件版本号、软件计算HMAC及制作升级包以外，其他操作都要打开设备，当您

不再使用设备时最好关闭设备，以免再次打开时出错。

8. 为什么获取到的加密数据看起来像是乱码？

答：这是由于灵锐I使用了AES加密技术，其输入输出均为二进制数据，如果需要字符串形式的输入和输出，可以在灵锐I的AES加解密函数基础上进行封装，例如使用BASE64编码。

9. 通过灵锐开发设置工具可以读出锁内的数据，那么其他人是否也可以通过此工具看到我的加密方案？

答：读取区块0的数据需要先校验管理密码，读取区块1～区块3的数据需要先校验普通用户密码，所以只要将这两级密码修改为秘密值，就无法通过工具看到锁内的加密方案了。

10. 每把灵锐I设备的AES密钥是否都是唯一的？

答：各软件开发商的AES密钥不同，即同一软件开发商所有灵锐I设备的AES密钥均相同。

11. 同样的软件，使用试用锁测试时没有问题，但使用正式锁会出现打开设备失败的错误，是什么原因？

答：灵锐I试用锁与正式锁的开发商编号不同，在使用“LIV_open”函数打开设备时，如果第一个参数填写了开发商编号，则需要更改此项为正式锁的开发商编号。

12. 灵锐I是否支持多进程或者多线程操作？

答：灵锐I暂不支持多进程和多线程访问。

附录 II

A. 支持的开发语言

VC++、C++ Builder、C#、Java、Delphi、VB、PB、AutoCAD。

B. 支持的操作系统

Windows 2000、Windows XP（32 位，64 位）、Windows Server 2003（32 位，64 位）、Windows Vista（32 位，64 位）、Windows Server 2008（32 位，64 位）和 Windows7（32 位，64 位）。

C. 硬件技术规格

项目	值	备注
工作电压	DC 4.5-5.5V	无
最大功耗	200mW	无
工作温度	0～70℃	无
数据存储时间	10 年	典型值
擦除/写周期	1 万次	最低值
设备接口	USB 1.1	低速设备，符合 HID 规范

D. 性能测试

项目	数值	备注
AES 加密时间	20ms	16 字节平均值
AES 解密时间	26ms	16 字节平均值
读取时间	179ms	512 字节平均值
写入时间	246ms	512 字节平均值
HMAC 运算时间	185ms	100 字节平均值

注：以上数据仅供参考，实际应用过程中可能存在误差。