

安全密钥管理工具 V.1.03

用户手册

此等资料中所含的包括产品和产品规范在内的所有信息均为资料发布时的产品信息，信息将不时由 Renesas Electronics 公司更改，恕不另行通知。请通过包括 Renesas Electronics 公司网站 (<http://www.renesas.com>) 在内的各种方式了解 Renesas Electronics 公司发布的最新信息。

注意

1. 本文件中电路、软件和其他相关信息的描述仅用于说明半导体产品的操作和应用示例。用户应对产品或系统设计中电路、软件和信息纳入或任何其他用途承担全部责任。对于您或第三方因使用这些电路、软件或信息而引起的任何损失和损害，Renesas Electronics 不承担任何责任。
2. Renesas Electronics 特此声明，对于因使用本文件中所述的 Renesas Electronics 产品或技术信息（包括但不限于产品数据、图纸、图表、程序、算法和应用示例）而引起的侵权或与第三方有关的专利、版权或其他知识产权的任何其他索赔，概不承担任何责任和赔偿。
3. 对 Renesas Electronics 或其他公司的任何专利、版权或其他知识产权均不授予任何明示、暗示或其他形式的许可。
4. 您应负责确定需要从任何第三方获得哪些许可，并在需要时为合法进口、出口、制造、销售、使用、分销或以其他方式处置包含 Renesas Electronics 产品的任何产品获得此类许可。
5. 不得对 Renesas Electronics 产品的全部或部分进行更改、修改、复制或逆向工程。对于因更改、修改、复制或逆向工程而导致您或第三方蒙受的任何损失或损害，Renesas Electronics 不承担任何责任。
6. Renesas Electronics 产品根据以下两个质量等级进行分类：“标准”和“优质”。Renesas Electronics 每种产品的预期应用取决于产品的质量等级，具体如下所示。

“标准”：计算机、办公设备、通信设备、测试和测量设备、视听设备、家用电器、机械工具、个人电子设备、工业机器人等

“优质”：运输设备（汽车、火车、轮船等）；交通管制（交通信号灯）；大型通信设备；关键金融终端系统；安全控制设备等

除非在 Renesas Electronics 数据手册或 Renesas Electronics 其他文档中明确指定为高可靠性产品或用于恶劣环境的产品，否则 Renesas Electronics 产品不适合或不授权用于可能对人类生命构成直接威胁或造成人身伤害（人造生命支持设备或系统；手术植入物等），或者可能造成严重的财产损失（空间系统、海底中继器、核动力控制系统、飞机控制系统、关键设备系统、军事装备等）的产品或系统。对于因使用任何与 Renesas Electronics 数据手册、用户手册或其他 Renesas Electronics 文档不一致的 Renesas Electronics 产品而引起的您或任何第三方所造成的任何损坏或损失，Renesas Electronics 不承担任何责任。
7. 没有任何半导体产品是绝对安全的。尽管 Renesas Electronics 的硬件或软件产品中可能实施了任何安全措施或功能，Renesas Electronics 对因任何漏洞或侵扰（包括但不限于以任何未经授权的方式访问或使用 Renesas Electronics 产品或使用 Renesas Electronics 产品的系统）而产生的任何后果概不负责。RENESAS ELECTRONICS 不担保或保证 RENESAS ELECTRONICS 产品或使用 RENESAS ELECTRONICS 产品创建的任何系统不会被破坏，或者可免于数据损坏、攻击、病毒、干扰、黑客攻击、数据丢失或失窃或其他安全入侵（“漏洞问题”）。RENESAS ELECTRONICS 不承担由任何漏洞问题引起的或与之相关的任何和所有责任或义务。此外，在适用法律允许的范围内，RENESAS ELECTRONICS 不对本文件和任何相关或附带的软件或硬件提供任何和所有明示或暗示的保证，包括但不限于对适销性或特定用途的适用性的暗示保证。
8. 使用 Renesas Electronics 产品时，请参见最新的产品信息（数据手册、用户手册、应用笔记、可靠性手册中的“处理和使用半导体器件的一般说明”等），并确保使用条件符合 Renesas Electronics 在最大额定值、工作电源电压范围、散热特性和安装等方面的规定。对于因在超出上述规定范围的情况下使用 Renesas Electronics 产品而引起的任何失常、故障或事故，Renesas Electronics 不承担任何责任。
9. 尽管 Renesas Electronics 致力于提高 Renesas Electronics 产品的质量和可靠性，但半导体产品具有特定的特性，例如在特定速率下发生故障以及在某些使用条件下出现故障。除非在 Renesas Electronics 数据手册或 Renesas Electronics 其他文档中指定为高可靠性产品或用于恶劣环境的产品，否则 Renesas Electronics 的产品将不受抗辐射设计的约束。用户应负责采取安全措施，以防止人身伤害、火灾造成的伤害，和/或因 Renesas Electronics 产品发生故障或失常而对公众造成的危险，例如硬件和设备的安全设计，包括但不限于冗余、火控和故障预防、针对老化退化的适当处理或任何其他适当的措施。由于对微型计算机软件进行评估非常困难且无实操性，因此用户有责任评估自己生产的最终产品或系统的安全性。
10. 请联系 Renesas Electronics 销售办事处，以获取有关环境事宜的详细信息，例如每个 Renesas Electronics 产品的环境相容性。用户有责任认真、充分地研究有关纳入或使用受控物质的适用法律和法规（包括但不限于欧盟 RoHS 指令），并按照所有适用法律和法规使用 Renesas Electronics 产品。对于因您未遵守适用的法律和法规而造成的损坏或损失，Renesas Electronics 不承担任何责任。
11. Renesas Electronics 产品和技术不得被用于或纳入为任何适用的本国或外国法律、法规所禁止制造、使用或销售的产品或系统范围内。用户应遵守由对当事方或交易拥有管辖权的任何国家/地区的政府颁布和管理的任何可适用的出口控制法律和法规。
12. 应由 Renesas Electronics 产品的购买方或分销商，或者对产品进行分发、处置或以其他方式出售或转让给第三方的任何其他当事方，负责将本文件中阐明的内容和条件提前通知前述第三方。
13. 未经 Renesas Electronics 事先书面同意，不得以任何形式全部或部分重印、再现或复制本文件。
14. 如果对本文件中包含的信息或 Renesas Electronics 产品有任何疑问，请联系 Renesas Electronics 销售办事处。

（注 1）本文件中的“Renesas Electronics”是指 Renesas Electronics Corporation，也包括其直接或间接控制的子公司。

（注 2）“Renesas Electronics 产品”是指 Renesas Electronics 开发或制造的任意产品。

（版本 5.0 2020 年 10 月 1 日）

公司总部

TOYOSU FORESIA, 3-2-24 Toyosu,
Koto-ku, Tokyo 135-0061, Japan
www.renesas.com

商标

Renesas 和 Renesas 徽标是 Renesas Electronics Corporation 的商标。所有商标和注册商标都是各自所有者的财产。

联系信息

有关产品、技术、文档最新版本或离您最近的销售办事处的更多信息，请访问：www.renesas.com/contact/。

有关微处理器和微控制器产品处理的一般预防措施

下方的使用说明适用于 **Renesas** 的所有微处理器和微控制器产品。有关本文档所述产品的详细使用说明，请参见本文档的相关章节以及针对相应产品发布的各项技术更新。

1. 防静电 (ESD) 措施

当暴露于 **CMOS** 器件时，强电场会导致栅极氧化物损坏，并最终使器件的工作性能下降。必须采取措施来尽可能地阻止静电的产生，并在发生静电时将其迅速消除。必须充分进行环境控制。如果环境干燥，应使用加湿器。建议避免使用容易产生静电的绝缘体。半导体器件必须在防静电容器、静电屏蔽袋或导电材料中存储和运输。所有测试和测量工具（包括工作台和地板）都必须接地。操作员还必须使用腕带接地。禁止徒手触摸半导体器件。对于装有半导体器件的印刷电路板，必须采取类似的预防措施。

2. 上电时的处理

产品在上电时仍处于未定义状态。在上电时，**LSI** 内部电路状态未定，寄存器设置和引脚的状态都未定义。对于复位信号已施加到外部复位引脚的成品，从上电开始到复位过程完成之前，引脚的状态并不能确定。同样地，对于以片上上电复位功能复位的产品，从上电开始到达到指定的复位电位之前，都无法确保引脚的状态。

3. 断电状态下的信号输入

器件断电时，请勿输入信号或 **I/O** 上拉电源。输入此类信号或 **I/O** 上拉电源导致的电流注入可能导致故障，此时流经器件的异常电流可能导致内部元件性能下降。请遵循产品文档中所述的电源关闭状态下输入信号的准则。

4. 未用引脚的处理

未使用的引脚应根据本手册“未用引脚的处理”部分给出的说明处理。**CMOS** 产品的输入引脚通常为高阻抗状态。在与开路状态下的未用引脚配合使用时，将在 **LSI** 周围产生额外的电磁噪声，内部将产生相关的直通电流，从而可能因为错误地将引脚状态识别为输入信号而产生故障。

5. 时钟信号

在应用复位后，只有在作业时时钟信号稳定后再释放复位线。当在程序执行过程中切换时钟信号时，等待目标时钟信号稳定。如在复位时外部谐振器（或外部振荡器）生成时钟信号，确保在时钟信号完全稳定后再释放复位线。此外，如在程序执行过程中切换到外部谐振器（或外部振荡器）生成的时钟信号，也需要等待目标时钟信号稳定。

6. 输入引脚上的电压施加波形

输入噪声或反射波引起的波形失真可能会导致故障。例如，如果由于噪声而使 **CMOS** 器件的输入处于 V_{IL} （最大值）和 V_{IH} （最小值）之间，则器件可能会发生故障。当输入电平固定时，以及在输入电平通过 V_{IL} （最大值）和 V_{IH} （最小值）之间时的过渡期间，请注意防止颤动噪声进入器件。

7. 禁止访问保留地址

禁止访问保留地址。保留地址用于未来可能的功能扩展。请勿访问此类地址，否则将无法保证 **LSI** 能正确运行。

8. 产品差异

在从一个产品切换到另一个前（例如，切换到具有不同零件号的产品），确认切换不会引起问题。对于同在一个系列但零件号不同的微处理器单元或单片机产品，其内部存储能力、布局图案和其他特性因素可能有所差异，进而可能会影响特性值、运行裕量、抗干扰性和噪声辐射量等电气特征的范围。在更换为部件编号不同的产品时，应针对给定产品执行系统评估测试。

如何使用本手册

1. 目的和目标读者

本手册旨在帮助用户了解安全密钥管理工具的功能。目标读者为采用安全加密引擎及可信安全引擎（瑞萨单片机内置的安全引擎）系统的设计和开发人员。要使用本手册，您需要掌握单片机以及 Windows 和 Linux 的基础知识。

在使用本软件之前，务必充分确认所用单片机的手册中的内容。

2. 约定

注：文本中标有“注”的项目的脚注。

数字表示：二进制 ... xxxx 或 xxxxB

十进制 ... xxxx

十六进制 ...0xXXXX 或 xxxxH

“ ”：屏幕上可以选择或输入的任何字符或项目

[]：屏幕上命令、对话框、选项卡页、选项或区域的名称

3. 术语

术语	含义
SCE	安全加密引擎 RA 产品家族和 Synergy 平台产品内置的安全加密 IP
TSIP	可信安全引擎 RE、RX 和 RZ 产品家族内置的安全加密 IP
用户密钥	用户应用中使用的加密密钥 AES 密钥、RSA 公钥、RSA 私钥等
UFPK	用户工厂烧录用密钥 用于封装用户密钥和 DLM 密钥的密钥
W-UFPK	使用瑞萨密钥封装服务封装的 UFPK
加密的用户密钥	使用 UFPK 或 KUK 封装的密钥
封装的用户密钥	通过 SCE 或 TSIP 重新封装的用户密钥，可用于 SCE 保护模式或 TSIP
DLM 密钥	产品生命周期管理用密钥，部分 MCU/MPU 支持 DLM Key
KUK	密钥升级用密钥 用于升级用户密钥的密钥
RFP	瑞萨闪存编程器 https://www.renesas.com/rfp
CLI	命令行接口
GUI	图形用户界面
瑞萨密钥封装服务	瑞萨提供的一项服务，通过产品内置的密钥对 UFPK 进行封装，生成 W-UFPK https://dln.renesas.com/keywrap/
瑞萨密钥文件	瑞萨密钥文件 供 RFP 使用的密钥配置文件
HUK	硬件唯一密钥 每个 MCU 唯一的密钥数据
HRK	硬件根密钥 每个 MCU 产品家族特有的密钥数据
PEM	Base64 编码的 X509 ASN.1 密钥文件 该工具支持读取 Openssl genrsa 命令或 ecparam -genkey 命令生成的密钥

目录

- 注意.....5
- 1. 瑞萨密钥管理系统.....10
 - 1.1 信任根简介10
 - 1.2 瑞萨安全引擎和关联密钥简介10
 - 1.3 瑞萨安全密钥安装的优势.....12
 - 1.3.1 密钥封装与密钥加密相比的优势.....12
 - 1.3.2 使用 HUK 的密钥封装的优势.....13
 - 1.4 封装密钥安装程序概述13
 - 1.4.1 安全密钥注入的一般步骤13
 - 1.4.2 安全密钥升级的一般步骤16
- 2. 概述.....18
 - 2.1 特性18
 - 2.2 运行环境18
 - 2.2.1 硬件环境18
 - 2.2.2 软件环境19
- 3. GUI 功能说明20
 - 3.1 主窗口20
 - 3.2 菜单栏.....22
 - 3.2.1 [文件] 菜单22
 - 3.3 [概要] 选项卡23
 - 3.4 [生成 UFPK] 选项卡24
 - 3.5 [生成 KUK] 选项卡26
 - 3.6 [封装密钥] 选项卡28
 - 3.6.1 [密钥类型] 选项卡29
 - 3.6.2 [密钥数据文件] 选项卡32
 - 3.6.2.1 在 [密钥类型] 选项卡中选择了 DLM/KUK/AES/TDES/ARC4/ECC 私钥时32
 - 3.6.2.2 在 [密钥类型] 选项卡中选择了 RSA 公钥时33
 - 3.6.2.3 在 [密钥类型] 选项卡中选择了 RSA 私钥时34
 - 3.6.2.4 在 [密钥类型] 选项卡中选择了 ECC 公钥时35
 - 3.6.3 封装密钥36
 - 3.6.4 IV37
 - 3.6.5 输出38
- 4. CLI 函数说明.....39
 - 4.1 命令行语法39

4.2	命令	40
4.3	genufpk 命令选项	41
4.4	genkuk 命令选项	41
4.5	genkey 命令选项	42
4.5.1	mcu 选项	43
4.5.2	keytype 选项	44
4.5.3	key 选项	46
4.5.3.1	十六进制数据直接输入	46
4.5.3.2	文件输入	50
4.5.3.3	省略 key 选项	52
4.5.4	filetype 选项	53
4.5.4.1	filetype 的 rfp 选项	53
4.5.4.2	filetype 的 csource 选项	53
4.5.4.3	filetype 的 bin 选项	54
4.5.4.4	filetype 的 mot 选项	56
4.5.5	keyfileoutput 选项	58
5.	操作程序	59
5.1	独立	59
5.1.1	Windows	59
5.1.1.1	GUI 版本	59
5.1.1.2	CLI 版本	60
5.1.2	Linux 版本	61
5.1.2.1	GUI 版本	61
5.1.2.2	CLI 版本	62
5.2	e ² studio 插件	63
5.2.1	安装 e ² studio 插件版本	63
5.2.2	卸载 e ² studio 插件版本	68
6.	使用示例	70
6.1	示例 1 – 在具有 TSIP 的 RX 产品家族 MCU 上安装 AES128 密钥	70
6.1.1	使用 GUI 版本	70
6.1.2	使用 CLI 版本	74
6.2	示例 2 – 在具有 SCE9 保护模式的 RA 产品家族 MCU 上安装升级密钥	75
6.2.1	使用 GUI 版本	75
6.2.2	使用 CLI 版本	80
6.3	示例 3 – 在具有 SCE9 保护模式的 RA 产品家族 MCU 上升级 RSA 2048 公钥	82
6.3.1	使用 GUI 版本	82
6.3.2	使用 CLI 版本	86
7.	注意事项	87
7.1	使用 Windows 环境时的显示设置	87

7.2	在Linux环境下使用e ² studio插件版本的注意事项.....	88
附录	89

附图目录

图 1-1 安全加密引擎概览	10
图 1-2 RX 可信安全引擎概览	11
图 1-3 密钥封装与密钥加密	12
图 1-4 使用 HUK 的密钥封装	13
图 1-5 使用 DLM 服务器封装 UFPK	14
图 1-6 使用 UFPK 加密用户密钥	14
图 1-7 通过串行编程接口注入用户密钥	15
图 1-8 通过串行编程接口注入 KUK	16
图 1-9 使用 KUK 加密新用户密钥	16
图 1-10 升级用户密钥	17
图 3-1 独立版本的主窗口	20
图 3-2 e ² studio 插件版本的主窗口	21
图 3-3 [概要] 选项卡	23
图 3-4 [生成 UFPK] 选项卡	24
图 3-5 [生成 KUK] 选项卡	26
图 3-6 [封装密钥] 选项卡	28
图 3-7 [密钥类型] 选项卡	29
图 3-8 选择了 DLM/KUK/AES/TDES/ARC4/ECC 私钥时的 [密钥数据文件] 选项卡	32
图 3-9 选择了 RSA 公钥时的 [密钥数据文件] 选项卡	33
图 3-10 选择了 RSA 私钥时的 [密钥数据文件] 选项卡	34
图 3-11 选择了 ECC 公钥时的 [密钥数据文件] 选项卡	35
图 3-12 封装密钥选项	36
图 3-13 IV 选项	37
图 3-14 输出文件选项	38
图 4-1 手动创建 AES 128 位密钥文件的示例	51
图 4-2 手动创建 RSA 2048 公钥文件的示例	52
图 5-1 安全密钥管理工具 - 从 Windows 启动时的 GUI 对话框	59
图 5-2 安全密钥管理工具 - 从命令提示符执行 CLI 示例	60
图 5-3 安全密钥管理工具 - 从 Linux 启动时的 GUI 对话框	61
图 5-4 安全密钥管理工具 - 从 Linux 终端软件执行 CLI 示例	62
图 5-5 e ² studio “帮助(H)” - “安装新软件...”	63
图 5-6 “安装”对话框	64
图 5-7 “添加资源库”对话框	64
图 5-8 “安装”对话框 - 选择 “Security Key Management Tool”	65
图 5-9 “安装”对话框 - 安装详细信息	65
图 5-10 “信任”对话框	66
图 5-11 项目 “属性”对话框	67
图 5-12 “关于 e ² studio”对话框	68
图 5-13 “e ² studio 安装细节”对话框	68
图 5-14 “卸载”对话框	69
图 6-1 [概要] 选项卡, 使用 RX 产品家族 TSIP 安装 AES128 密钥	70
图 6-2 [生成 UFPK] 选项卡, 使用指定的值生成 UFPK 的示例	71
图 6-3 使用指定的值生成 UFPK 的执行结果示例	71
图 6-4 [封装密钥] - [密钥类型] 选项卡, 以 Motorola 十六进制格式创建 AES128 密钥文件的示例	72

图 6-5 [封装密钥] - [密钥数据文件] 选项卡, 以 Motorola 十六进制格式创建 AES128 密钥文件的示例	73
图 6-6 以 Motorola 十六进制格式创建 AES128 密钥文件的执行结果示例	73
图 6-7 CLI genufpk 命令的执行结果	74
图 6-8 以 Motorola 十六进制格式创建 AES128 密钥文件的 CLI 示例	74
图 6-9 [概要] 选项卡, 在 RA 产品家族 SCE9 保护模式下安装 KUK	75
图 6-10 [生成 UFPK] 选项卡, 使用指定的值生成 UFPK 的示例	76
图 6-11 使用指定的值生成 UFPK 的执行结果示例	76
图 6-12 [生成 KUK] 选项卡, 创建 KUK 密钥文件的示例	77
图 6-13 创建 KUK 文件的执行结果示例	77
图 6-14 [封装密钥] - [密钥类型] 选项卡, 以 RFP 文件格式创建 KUK 文件的示例	78
图 6-15 [封装密钥] 选项卡, 以 RFP 文件格式创建 KUK 文件的示例	79
图 6-16 以 RFP 文件格式创建 KUK 文件的执行结果示例	79
图 6-17 CLI genufpk 命令的执行结果	80
图 6-18 CLI genkuk 命令的执行结果	80
图 6-19 以 RFP 文件格式创建 AES128 密钥文件的 CLI 示例	81
图 6-20 [概要] 选项卡, 在具有 SCE9 保护模式的 RA 产品家族上升级 RSA 2048 公钥	82
图 6-21 [封装密钥] - [密钥类型] 选项卡, 以 C 源文件格式创建 RSA 2048 公钥文件的示例	83
图 6-22 [封装密钥] - [密钥数据文件] 选项卡, 以 C 源文件格式创建 RSA 2048 公钥文件的示例	84
图 6-23 以 C 源文件格式创建 RSA 2048 公钥文件的执行结果示例	85
图 6-24 以 C 源文件格式创建 RSA 2048 公钥文件的 CLI 示例	86
图 7-1 SecurityKeyManagementTool.exe 属性“高 DPI 缩放覆盖”设置	87

1. 瑞萨密钥管理系统

1.1 信任根简介

信任根是高度可靠的硬件、固件和软件组件，用于执行特定关键安全功能 (<https://csrc.nist.gov/projects/hardware-roots-of-trust>)。在物联网系统中，信任根通常由器件硬件中固有的标识和加密密钥组成。它建立唯一不变的防克隆标识，以授权器件存在于物联网网络中。

在许多安全系统中，安全引导是信任根所提供服务的一部分。应用程序的身份验证使用公钥加密。关联密钥是系统信任根的一部分。器件标识由器件私钥和器件证书组成，是许多物联网设备信任根的一部分。

基于上述信任根的探讨，我们可以了解到加密密钥泄露会导致安全系统处于危险状态。信任根的保护包括将密钥可访问性仅限制在加密边界内，同时使用安全存储密钥且最好是防克隆密钥。信任根应锁定，以防止未经授权方对其进行读写访问。

瑞萨用户密钥管理系统可以提供上述全部所需的保护。

1.2 瑞萨安全引擎和关联密钥简介

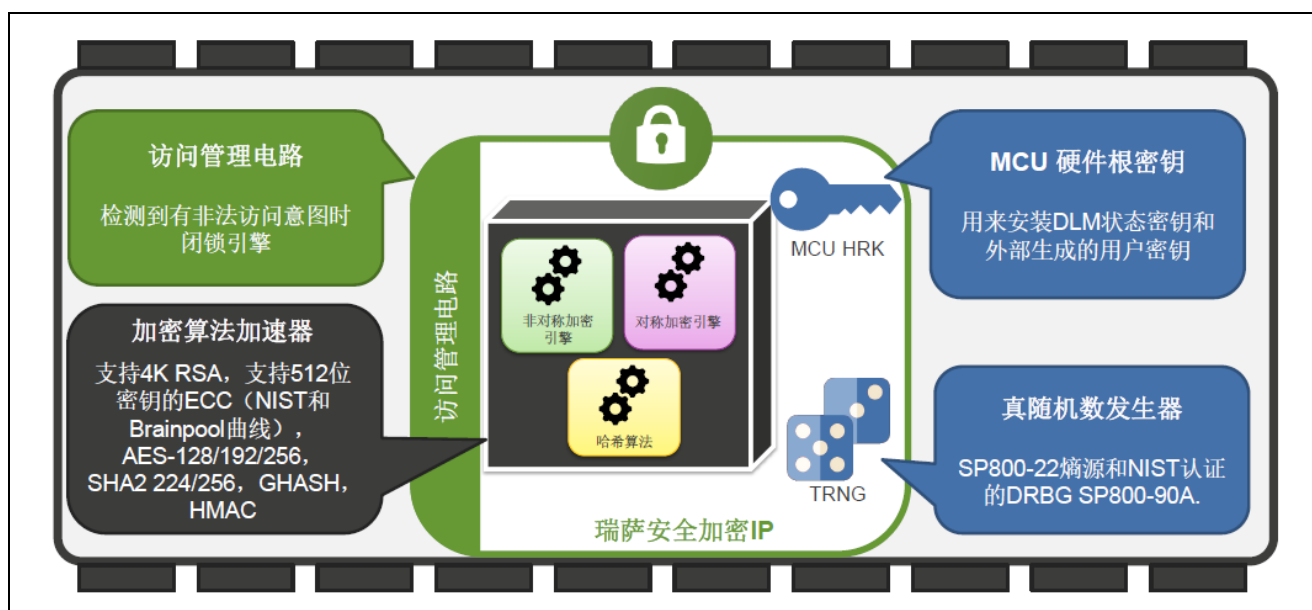


图 1-1 安全加密引擎概览

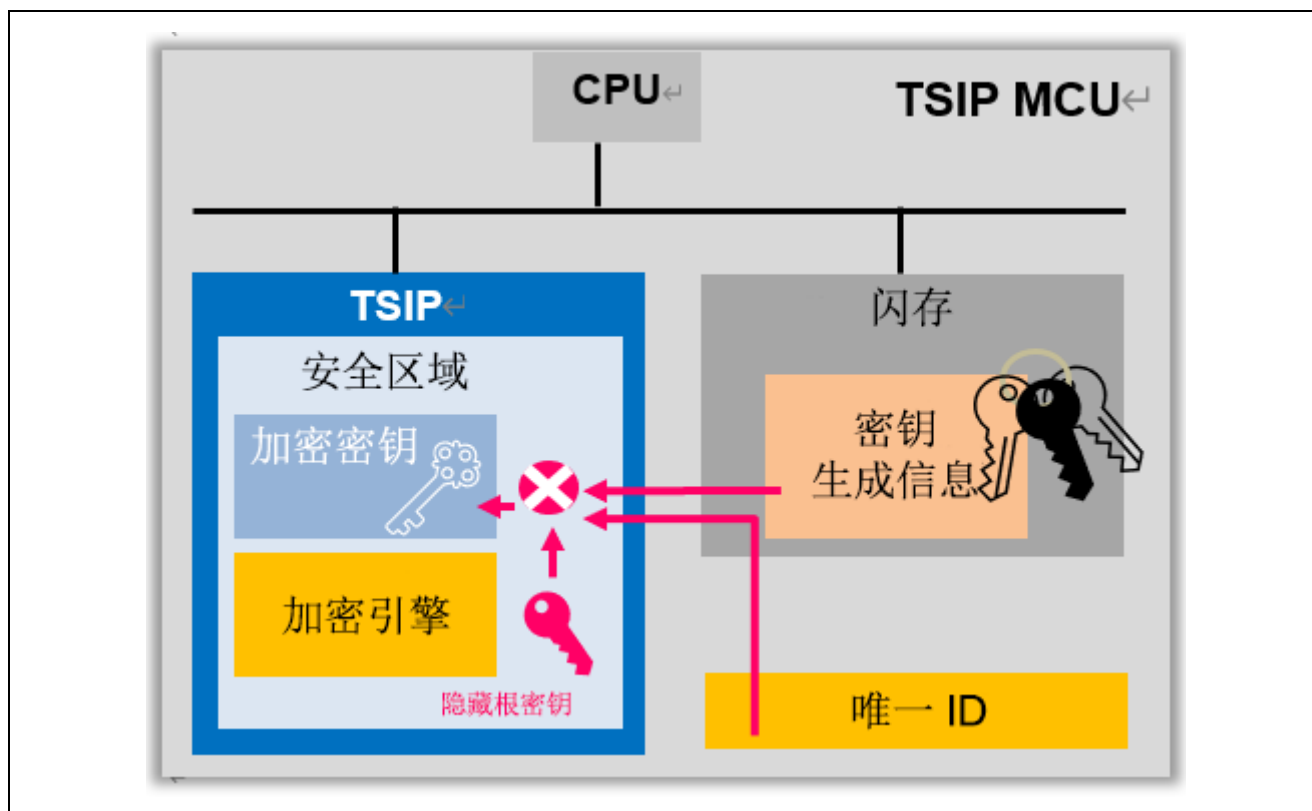


图 1-2 RX 可信安全引擎概览

瑞萨安全引擎包括安全加密引擎 (SCE) 和可信安全引擎 (TSIP)，是 MCU 内部的孤立子系统。加密引擎包括用于对称和非对称加密算法的硬件加速器，以及各种哈希和消息验证代码。其中还包括真随机数发生器 (TRNG)，用于为加密操作提供熵源。加密引擎受访问管理电路保护，该电路可以在有非法外部访问意图时闭锁加密引擎。

根据特定 MCU/MPU，支持特定算法、安全密钥注入和安全密钥升级。有关更多信息，请参见设备的硬件用户手册及以下网页：

表 1-1 MCU/MPU 相关信息

MCU/MPU	类别	URL
RA 产品家族	MCU 驱动程序	https://www.renesas.com/eu/en/software-tool/flexible-software-package-fsp
	应用程序项目	https://www.renesas.com/eu/en/document/apn/installing-and-updating-secure-keys-ra-family
RE 产品家族	MCU 驱动程序和示例项目	https://www.renesas.com/software-tool/trusted-secure-ip-driver
RX 产品家族		
RZ 产品家族		
Synergy 平台	MCU 驱动程序	https://www.renesas.com/eu/en/products/microcontrollers-microprocessors/renesas-synergy-platform-mcus/renesas-synergy-software-package

1.3 瑞萨安全密钥安装的优势

安全密钥安装和升级，结合加密引擎对封装密钥的支持，解决了与使用明文密钥相关的许多漏洞：

- 明文密钥永远不会存储在代码闪存中。在程序存储器被攻破的情况下，可以保护敏感的密钥数据。
- 明文密钥永远不会存储在 **RAM** 中。在系统上可执行恶意代码的情况下，仍会保护敏感的密钥数据。
- 密钥可以安全地存储在代码闪存、数据闪存中，甚至可以复制到外部存储器中，从而实现无数量限制的安全密钥存储。

此外，瑞萨密钥封装技术可防止器件被克隆，如下文所述。

1.3.1 密钥封装与密钥加密相比的优势

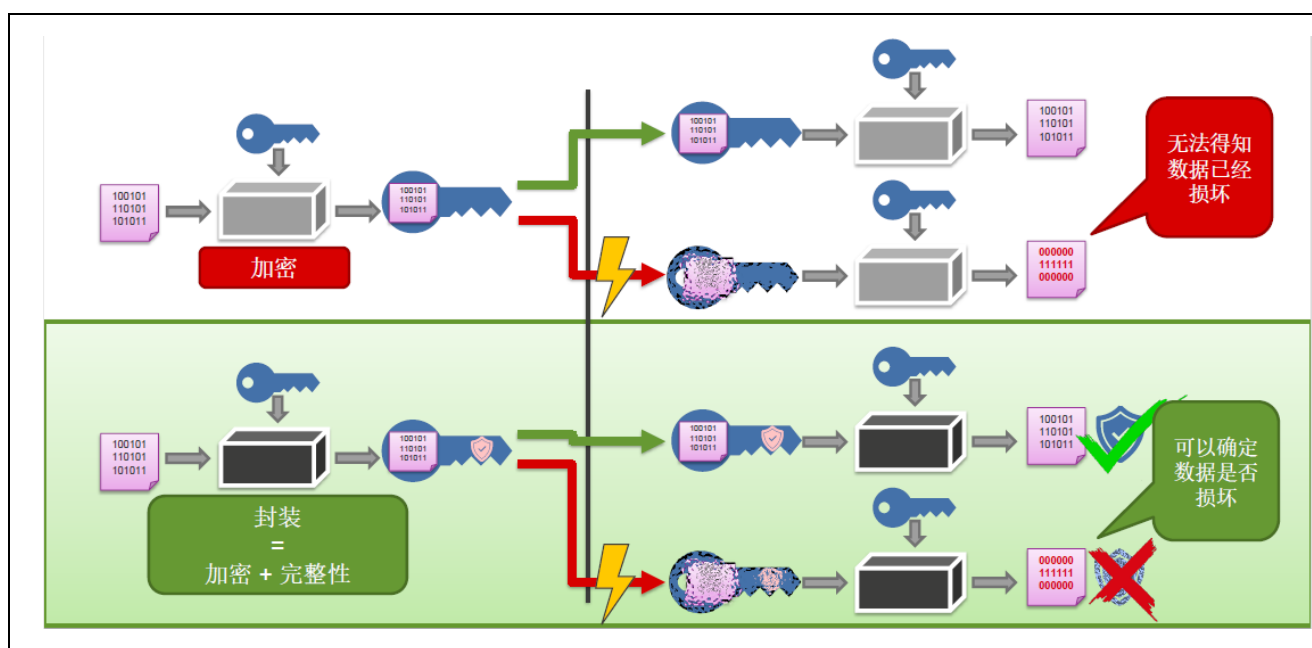


图 1-3 密钥封装与密钥加密

务必要了解安全资产存储时使用封装和加密之间的区别。当数据经过加密并发送到另一接收方时，如果该接收方具有相同的密钥，则可以对数据进行解密。结果即是秘密地交换了信息。但是，如果加密数据的传输出现问题会如何？如果接收方在不知情的情况下接收到已损坏的信息，则解密算法将生成垃圾数据，并且无法得知原始数据已损坏。

利用封装技术，可在加密输出上附加用于完整性检查的消息验证数据，从而解决了此问题。

1.3.2 使用 HUK 的密钥封装的优势

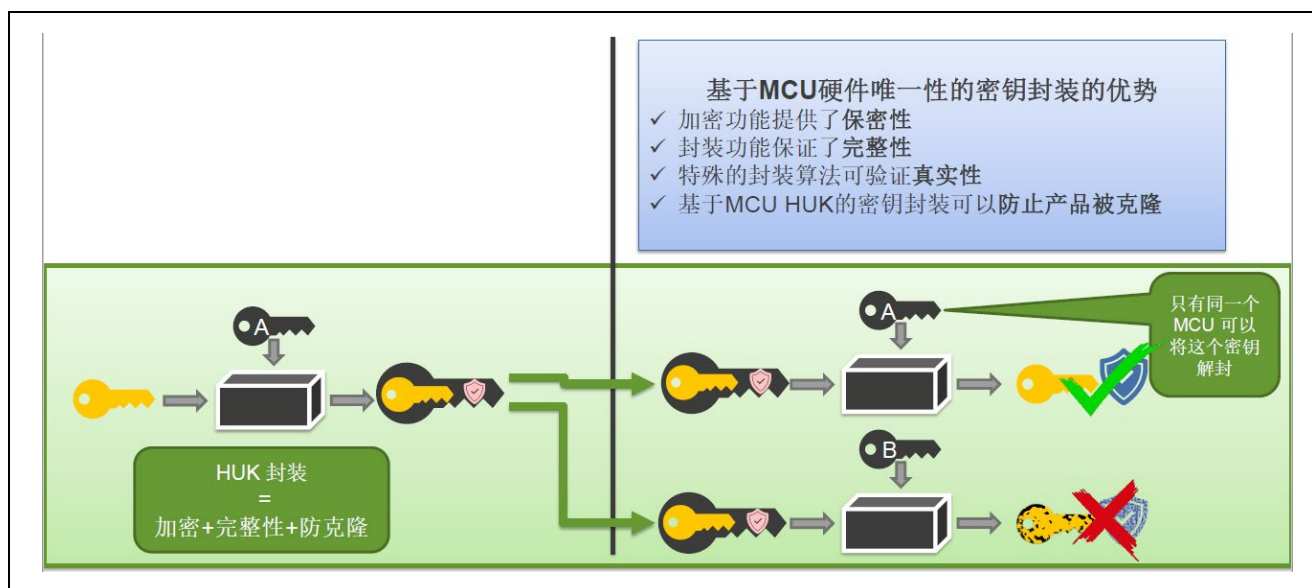


图 1-4 使用 HUK 的密钥封装

使用硬件唯一密钥 (HUK) 封装存储的密钥可提供另一项保护功能 – 防克隆保护。如果封装的密钥传输或复制到其他 MCU/MPU，则该 MCU/MPU 将无法解封也无法使用复制的密钥。即使将整个 MCU 的内容复制到其他器件上，也无法使用或显示密钥。

1.4 封装密钥安装程序概述

本节介绍了封装密钥的注入程序，该程序需要安全密钥管理工具提供的功能。不同 MCU/MPU 支持的密钥类型不同。有关每个器件的应用笔记和驱动程序，参见表 1-1 MCU/MPU 相关信息。

1.4.1 安全密钥注入的一般步骤

安全密钥注入是以 MCU 唯一封装的方式存储应用程序密钥的过程，该过程采用在配置过程中启用不显示明文密钥的机制实现。该过程的具体机制取决于具体的 MCU/MPU。一些器件支持通过编程接口实现安全密钥注入；其他器件支持使用器件上运行的固件进行安全密钥安装。请注意，使用安全密钥管理工具的密钥准备步骤（其中密钥材料以明文显示）必须在安全环境中执行。

密钥注入分三个大的步骤。

1. 安全密钥注入过程的第一步是使用瑞萨的产品生命周期管理 (DLM) 密钥封装服务来封装任意指定的用户工厂烧录密钥 (UFPK)，此步骤使用瑞萨硬件根密钥 (HRK)，生成封装的 UFPK (W-UFPK)。UFPK 是用户选择的 256 位值。可以使用相同的 UFPK 注入任意数量的密钥。安全密钥管理工具可以创建随机的 UFPK，并将其作为适合发送到密钥封装服务的密钥文件。该工具还可以接受指定的 UFPK，并创建适合发送到密钥封装服务的密钥文件。

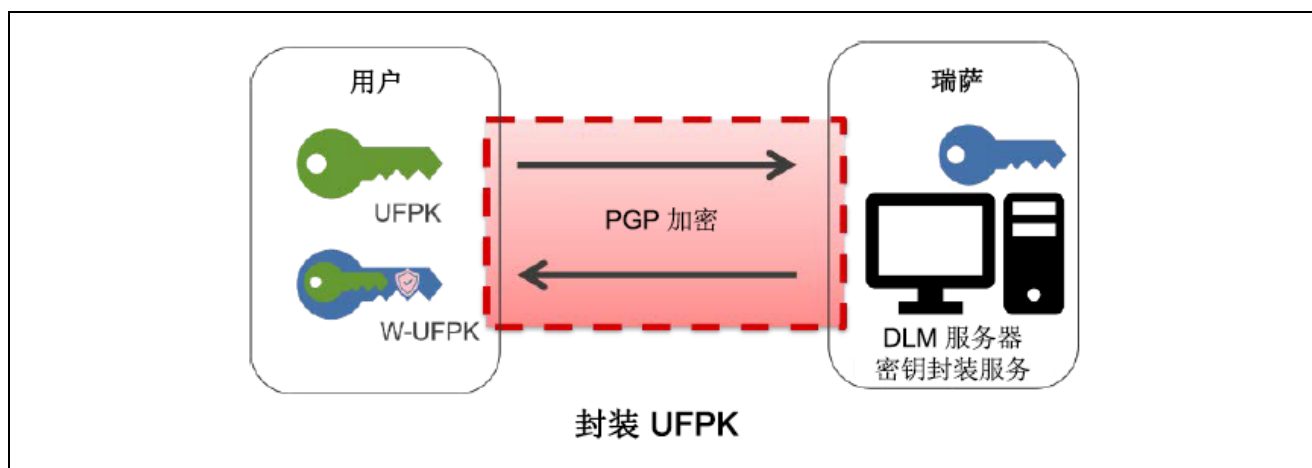


图 1-5 使用 DLM 服务器封装 UFPK

2. 接下来，必须使用 UFPK 来封装用户密钥。安全密钥管理工具可以封装明文密钥数据形式或二进制密钥文件形式的用户密钥，并生成可用于所选器件进行安全密钥安装的文件。请注意，并非所有器件产品家族都支持所有输出文件类型。例如，瑞萨 RA 产品家族仅支持通过编程接口实现安全密钥注入；因此，必须生成用于瑞萨闪存编程器 (RFP) 的安全密钥文件。

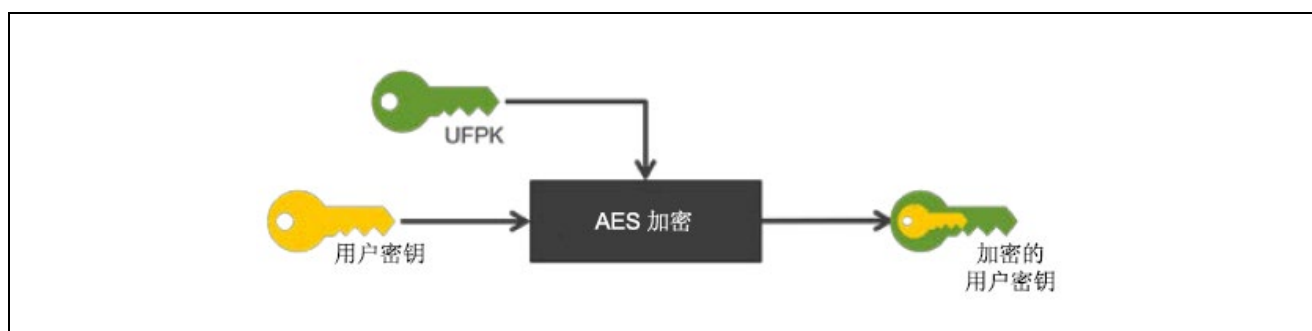


图 1-6 使用 UFPK 加密用户密钥

3. 最后，通过串行编程接口或器件上运行的固件注入用户密钥，具体取决于所选器件支持的密钥注入过程。该过程的输入包括前面步骤中准备的封装 UFPK (W-UFPK) 和封装用户密钥。

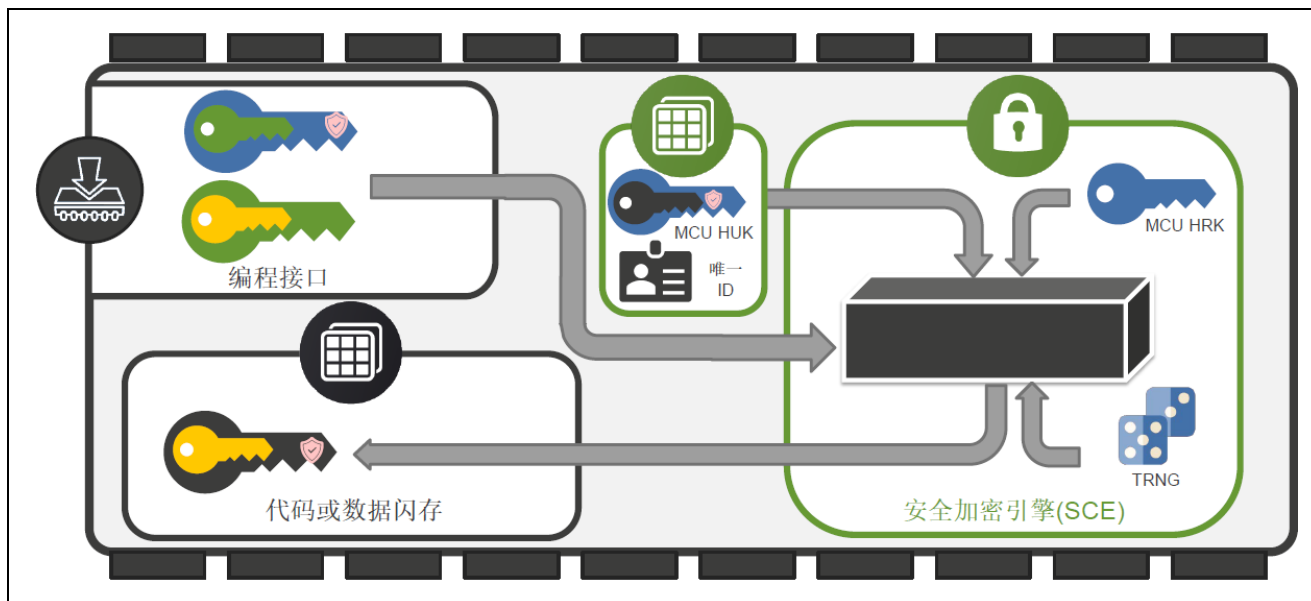


图 1-7 通过串行编程接口注入用户密钥

1.4.2 安全密钥升级的一般步骤

要在系统运行时使用安全密钥安装功能，必须在生产编程/配置过程中注入一个或多个升级密钥 (KUK)。KUK 与其他加密密钥一样，可以存储在代码闪存或数据闪存（如果在 MCU 上可用）中。由于在现场安装新密钥通常是用于替换旧密钥（密钥轮换或重新设置密钥），因此该过程称为“密钥升级”；但是，该过程也可以用于在现场安装新密钥。该过程不会删除之前注入的密钥。

请注意，如果目标器件支持通过编程接口实现密钥注入，则在禁用编程接口之后，无法安装额外的 KUK。在这种情况下，一旦产品位于禁用了编程接口的系统中，则只能通过现存的 KUK 注入新密钥。因此，强烈建议在生产配置过程中注入多个 KUK。这样便允许轮换或撤销 KUK，以遵循架构安全策略或响应密钥泄漏安全漏洞。

密钥升级分三个大的步骤。

1. 第一步是生成并安装 KUK，如第 1.4.1 节安全密钥所述。必须注入“KUK”密钥类型的 KUK。

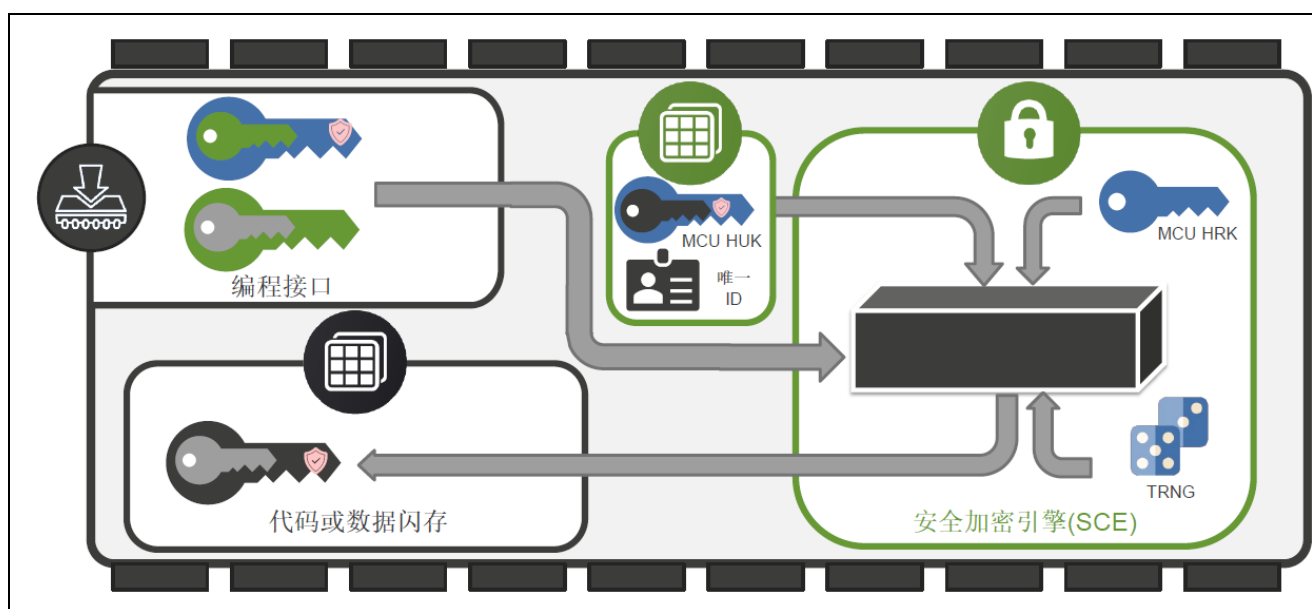


图 1-8 通过串行编程接口注入 KUK

2. 第二步是使用 KUK 封装新用户密钥。这与安全密钥安装类似，但使用的是 KUK 而不是 UFPK。

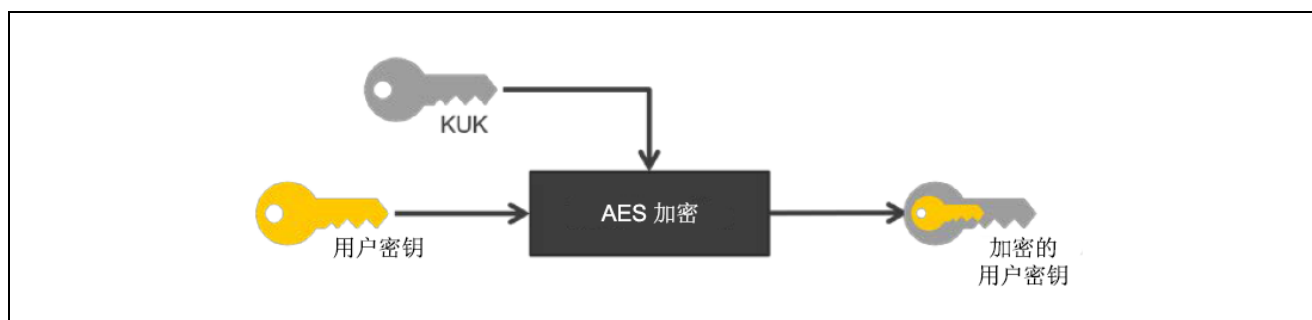


图 1-9 使用 KUK 加密新用户密钥

3. 最后一步是使用相应的器件驱动程序和已注入的 KUK 注入新用户密钥。有关密钥升级 API 及其使用方式的信息，请参见每个 MCU/MPU 器件驱动程序的手册。

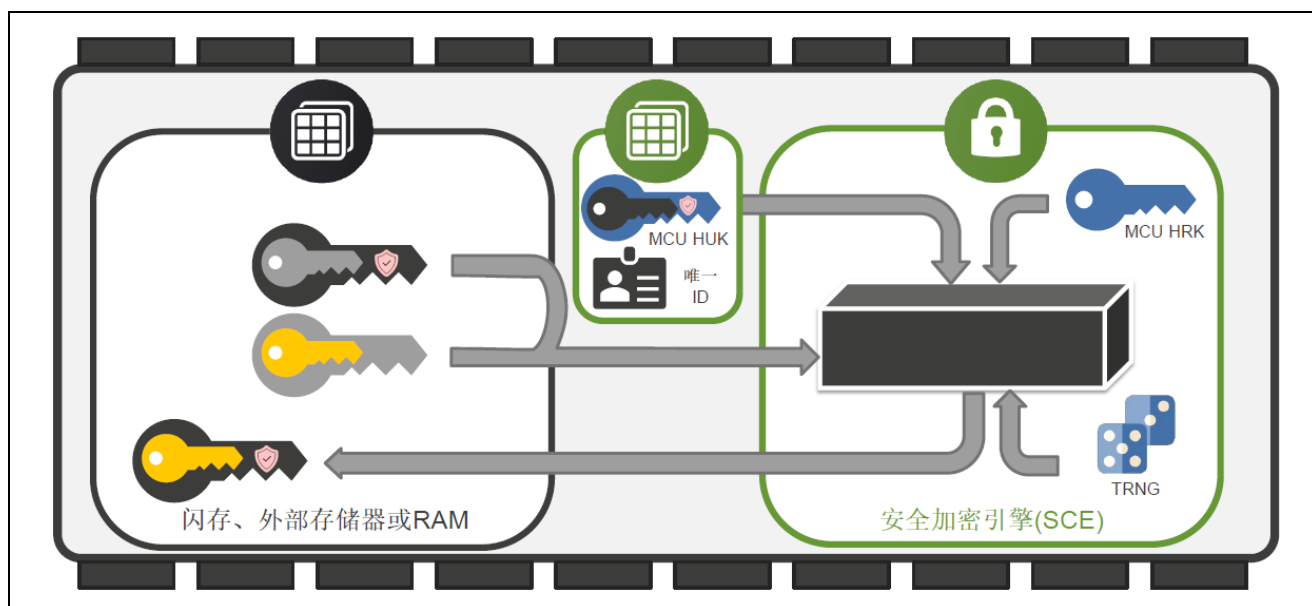


图 1-10 升级用户密钥

2. 概述

安全密钥管理工具用于为安全注入和升级准备应用程序和产品生命周期管理 (DLM) 密钥。

该工具有三个版本：

- 命令行接口 – 从操作系统命令行执行单个命令，或者在批处理文件或脚本中包含多个命令。设计用于密钥管理工具和支持小组开发。
- 独立的图形用户界面 – 直观的 GUI，旨在为固件开发人员提供帮助。在使用第三方 IDE 开发时十分有用。
- e²studio 插件 – 集成到瑞萨 e²studio 中的 GUI 接口，用于简化开发支持。

2.1 特性

安全密钥管理工具支持以下 MCU/MPU 器件：

- RA 产品家族 (DLM 密钥、SCE9 保护/兼容模式、SCE7、SCE5_B 和 SCE5 用户密钥)
- RE 产品家族 (TSIP 和 TSIP Lite 用户密钥)
- RX 产品家族 (TSIP Lite 用户密钥)
- RZ 产品家族 (TSIP 用户密钥)
- Synergy (SCE5 和 SCE7 用户密钥)

该工具支持以下功能：

- 生成和/或准备 UFPK，以提交给瑞萨密的钥封装服务
- 使用 UFPK 封装 DLM 密钥，以实现安全密钥注入（如果适用于所选的 MCU/MPU）
- 使用 UFPK 封装用户密钥，以实现安全密钥注入
- 使用 KUK 封装用户密钥，以实现安全密钥升级

支持以下输出文件格式（如果适用）：

- 瑞萨密钥文件
- C 源文件
- 二进制数据
- Motorola 十六进制文件

2.2 运行环境

2.2.1 硬件环境

(1) 主机 PC

- 处理器：1 GHz 或更快
- 主存储器：1 GB 或更大
- 显示设置：1366 x 768 或更高分辨率

显示比例 100%（推荐）

注：

如果未按上述分辨率或显示比例进行显示，则可能无法显示 GUI 中的所有选项。

2.2.2 软件环境

(1) 支持 OS

- Windows 10（64 位）
- Linux (Ubuntu 20.04 LTS)

(2) e²studio 插件版本 e²studio 运行检查版本

- e²studio 2022-10

3. GUI 功能说明

本章介绍安全密钥管理工具 GUI 版本的界面结构和功能。

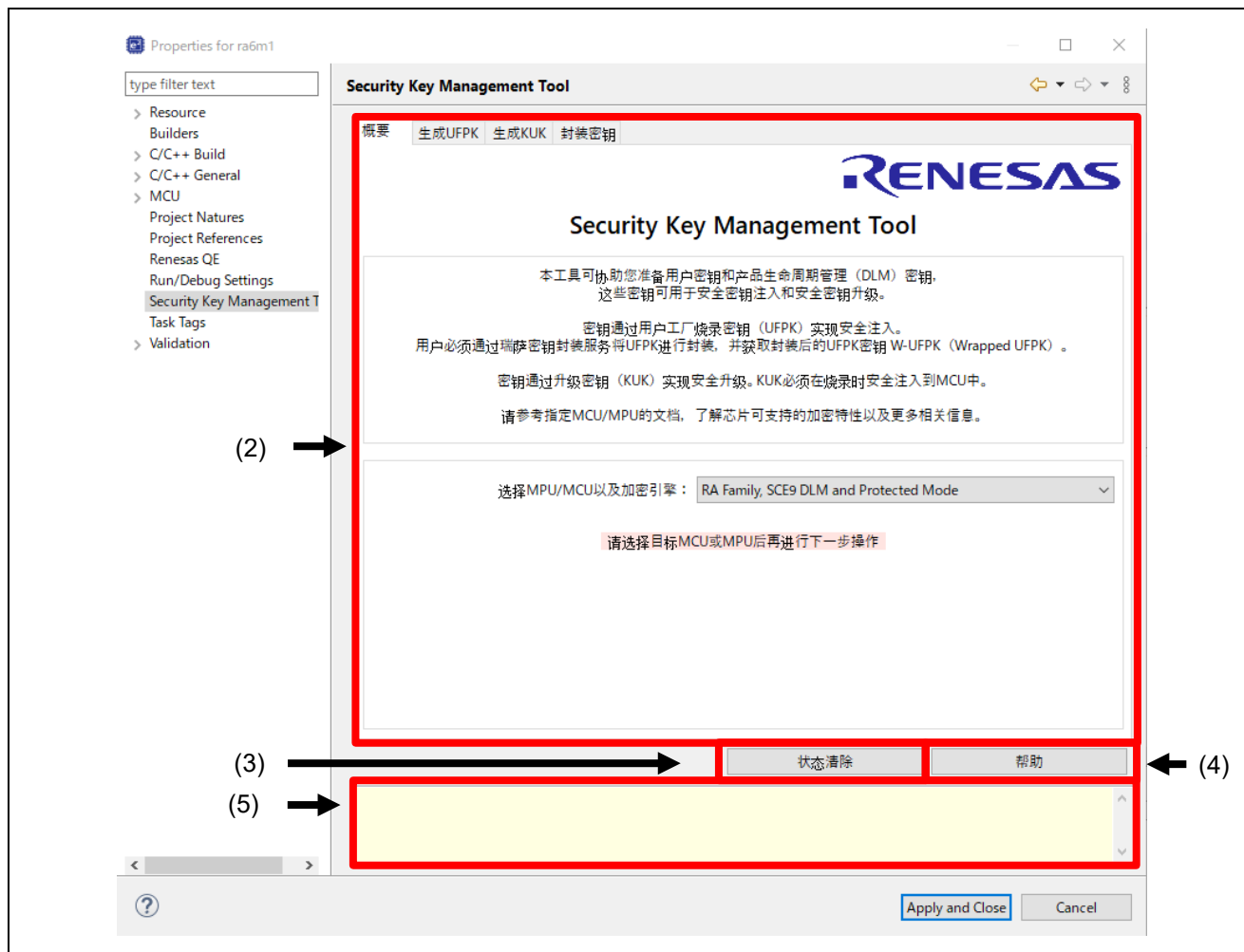
独立版本和 e²studio 插件版本具有相同的 GUI 配置。在此只对独立版本进行了说明。

3.1 主窗口

启动后的主窗口包括以下内容：



图 3-1 独立版本的主窗口

图 3-2 e²studio 插件版本的主窗口

编号	项目	说明
(1)	菜单栏	仅支持独立版本。 有关详细信息，请参见 3.2 菜单栏。
(2)	选项卡窗口	每个选项卡都提供一个界面，用于生成作为安全密钥安装和/或升级过程的一部分的文件。
(3)	状态清除	清除状态区域显示的执行结果。在独立版本中，通过菜单栏使用这个功能
(4)	帮助	显示对理解瑞萨密钥管理系统有用的资源。在独立版本中，通过菜单栏使用这个功能
(5)	状态	显示请求操作的状态。

3.2 菜单栏

这只是独立版本的一个功能。

3.2.1 [文件] 菜单

从与保存设置相关的功能菜单中选择。

- [保存…]

将每个选项卡中设置的输入/输出文件信息保存为 XML 格式的文件。密钥数据不会保存。

- [加载…]

加载 [保存…] 中保存的设置文件，并反映在每个选项卡中。

在 e²studio 插件版本中，按“应用并关闭”按钮将每个选项卡的配置信息保存到 e²studio 项目。

3.2.2 [视图] 菜单

与 GUI 显示相关的菜单。

- [状态清除]

清除状态区域显示的执行结果。

3.2.3 [帮助] 菜单

- [关于 Security Key Management Tool…]

帮助对话框，显示对理解瑞萨密钥管理系统有用的资源。

3.3 [概要] 选项卡

在该选项卡中，选择要使用的目标 MCU/MPU 以及加密引擎。确保先选择目标器件，然后再对任何其他选项卡执行操作。其他选项卡的功能取决于在该选项卡上选择的器件。



图 3-3 [概要] 选项卡

编号	项目	说明
(1)	选择 MPU/MCU 以及加密引擎	选择目标 MCU/MPU 以及加密引擎进行安全密钥安装和/或升级。

3.4 [生成 UFPK] 选项卡

该选项卡以二进制 *.key 文件形式生成用户工厂烧录密钥 (UFPK) 文件。然后必须将该文件发送到瑞萨密钥封装服务(DLM服务器)来获取封装的UFPK (W-UFPK)。UFPK文件还将用于为安全密钥安装准备密钥。

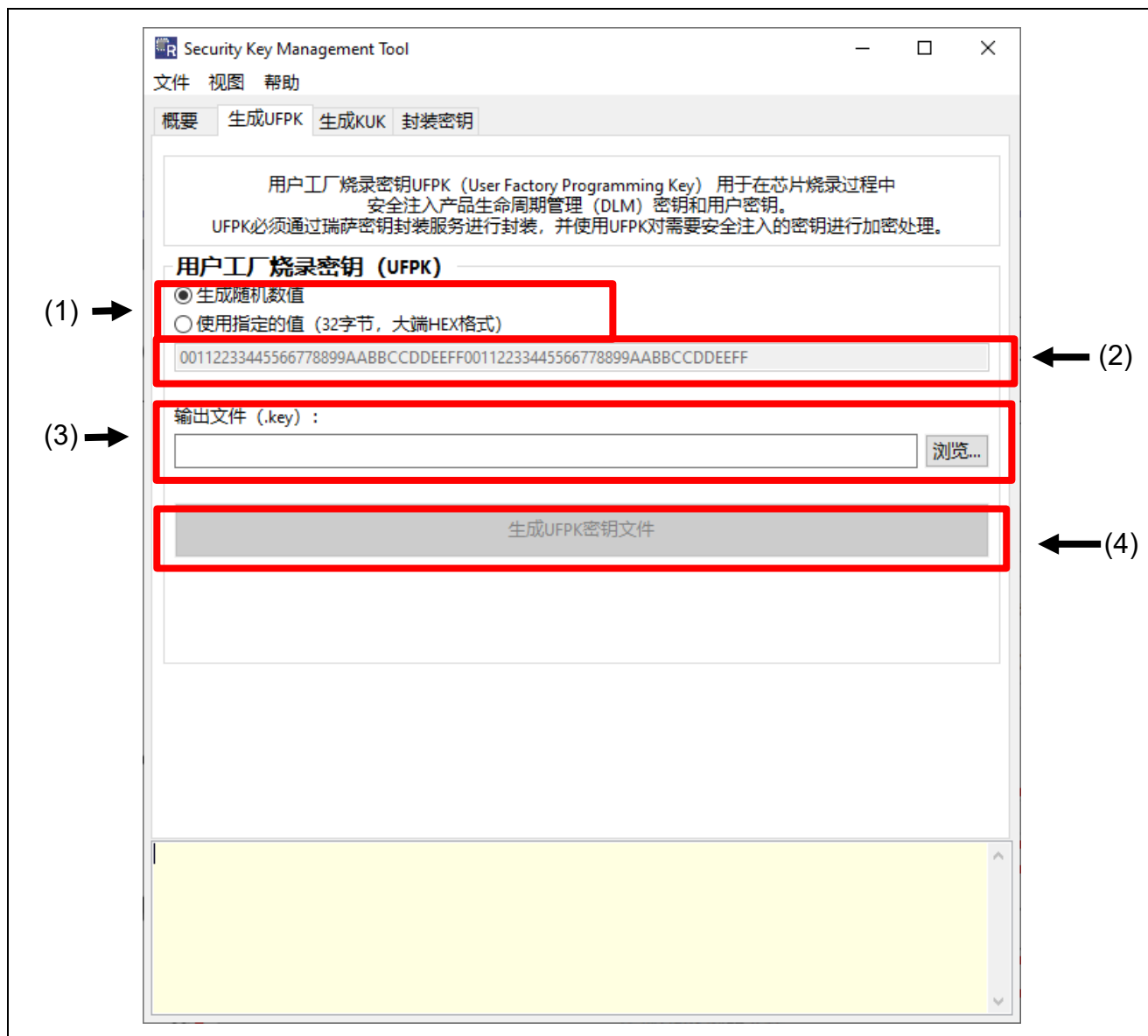


图 3-4 [生成 UFPK] 选项卡

编号	项目	说明
(1)	UFPK 输入格式	选择是使用 (2) 中的输入值作为 UFPK 值还是使用工具生成的随机数值。无法保证该工具随机生成的随机数的具体值。
(2)	UFPK 值	当在 (1) 中选择了 使用指定的值 时，将使用文本框中输入的值作为 UFPK 值。该值必须以大端格式指定为 32 字节十六进制值。
(3)	输出文件 (.key)	选择要输出的 UFPK 文件的路径和文件名。输出文件的扩展名必须设置为 .key。
(4)	生成 UFPK 密钥文件	根据上面输入的信息生成 UFPK 文件。 在 [概要] 选项卡上选择 MCU/MPU 以及加密引擎时启用。

必须将该选项卡中生成的 UFPK 文件发送到瑞萨密钥封装服务 (<https://dlm.renesas.com/keywrap>) 来获取 W-UFPK。有关瑞萨密钥封装服务的更多信息，请参见密钥封装服务的常见问题解答以及每个 MCU/MPU 的附加信息（表 1-1 MCU/MPU 相关信息）。

3.5 [生成 KUK] 选项卡

该选项卡以二进制 *.key 文件形式生成升级密钥 (KUK) 文件。该文件不仅用于安全安装 KUK，还用于准备其他密钥进行安全密钥升级。

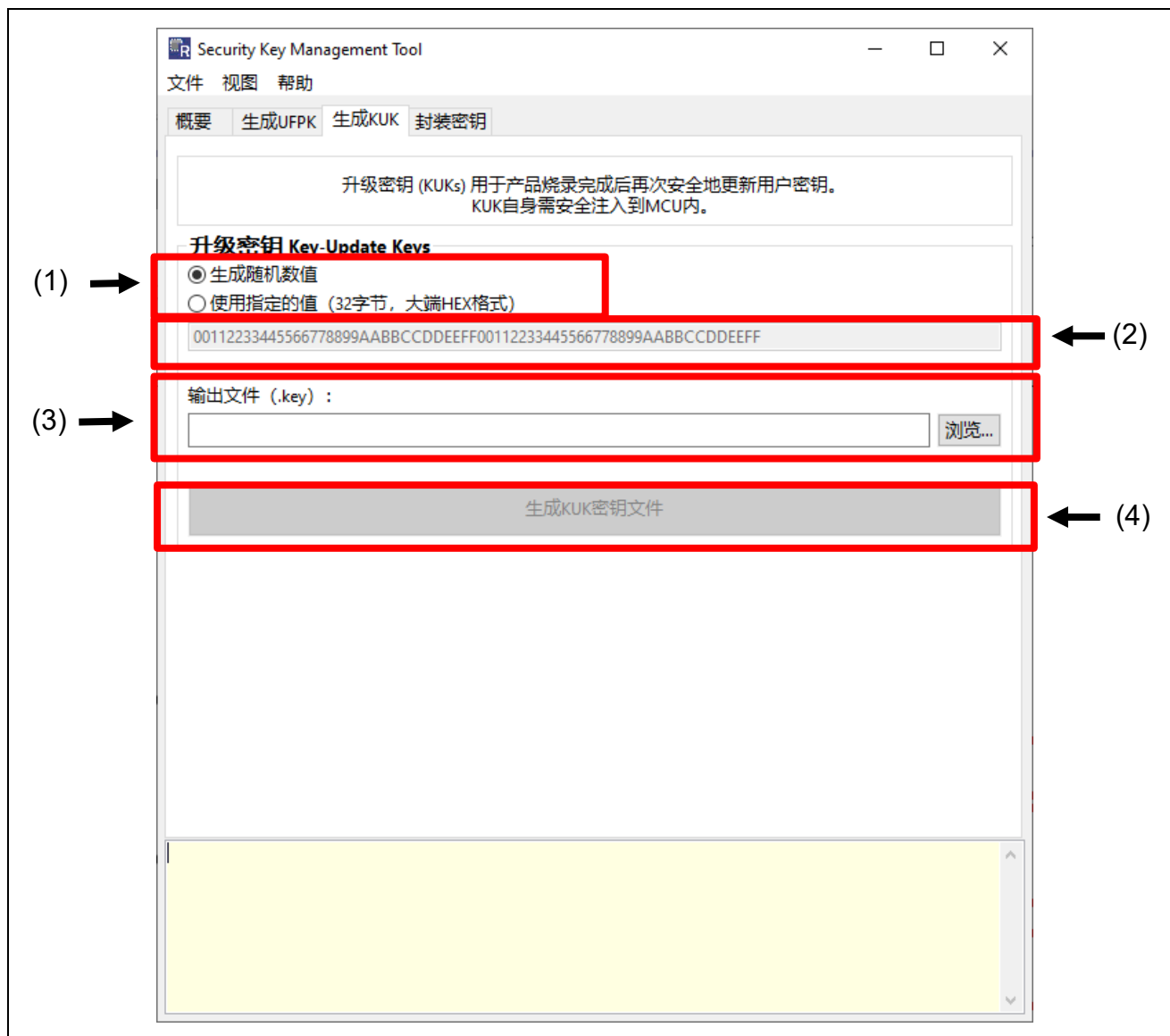


图 3-5 [生成 KUK] 选项卡

编号	项目	说明
(1)	KUK 输入格式	选择是使用 (2) 中的输入值作为 KUK 值还是使用工具生成的随机数值。无法保证该工具随机生成的随机数的具体值。
(2)	KUK 值	当在 (1) 中选择了 使用指定的值 时，将使用文本框中输入的值作为 KUK 值。该值必须以大端格式指定为 32 字节十六进制值。
(3)	输出文件 (.key)	选择要输出的 KUK 文件的路径和文件名。输出文件的扩展名必须设置为 .key。
(4)	生成 KUK 密 钥文件	根据上面输入的信息生成 KUK 文件。 在 [概要] 选项卡上选择 MCU/MPU 以及加密引擎时启用。

3.6 [封装密钥] 选项卡

使用该选项卡加密用户密钥，并生成安全安装或升级所需的文件。

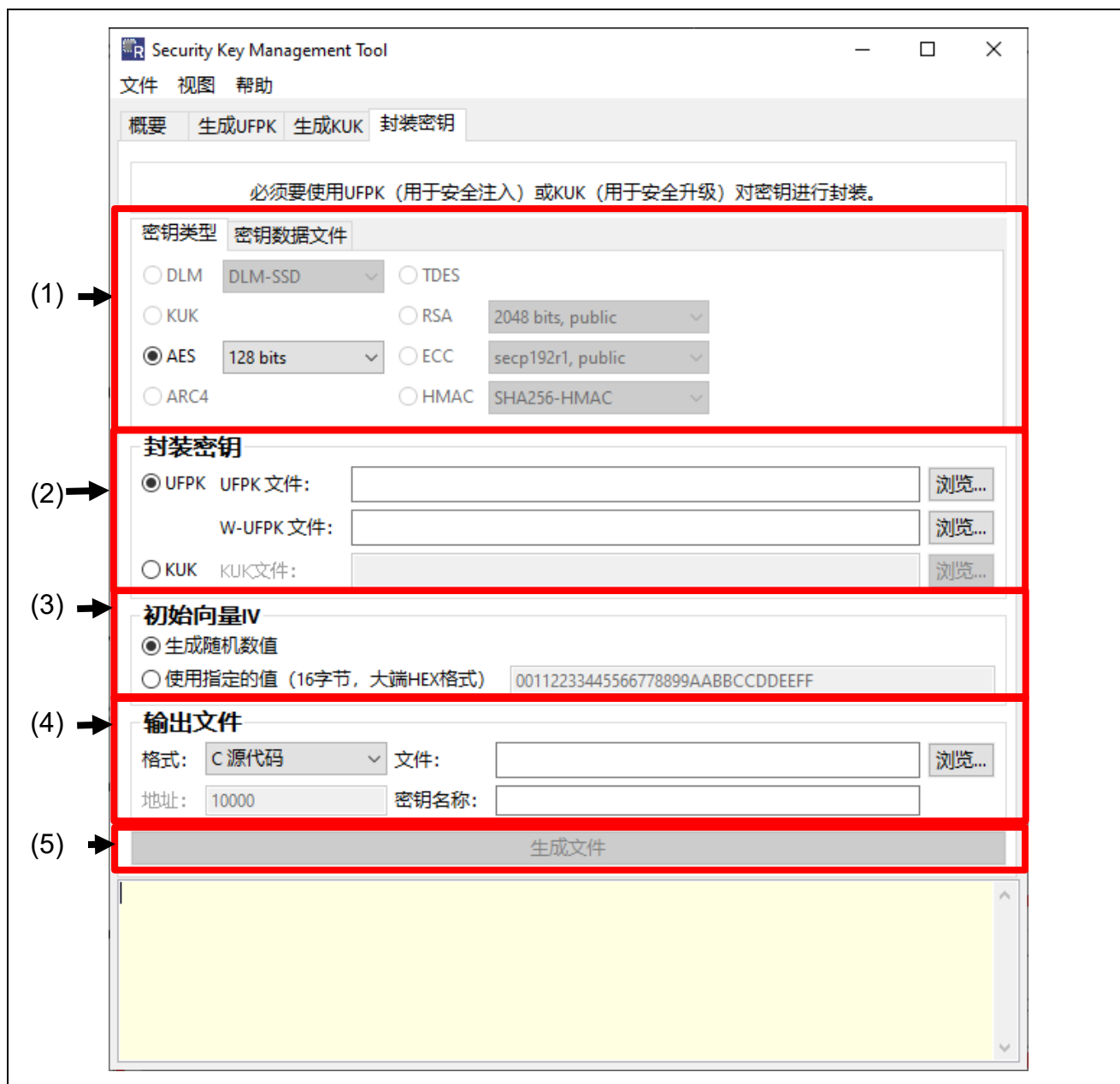


图 3-6 [封装密钥] 选项卡

编号	项目	说明
(1)	[密钥类型] 和 [密钥数据文件] 选项卡	在 [密钥类型] 选项卡中选择了要加密的用户密钥类型之后，在 [密钥数据文件] 选项卡中输入密钥数据文件。 有关 [密钥类型] 选项卡的详细信息，请参见第 3.6.1 节 [密钥类型] 选项卡。 有关 [密钥数据文件] 选项卡的详细信息，请参见第 3.6.2 节 [密钥数据文件] 选项卡。
(2)	封装密钥	设置要用于封装的密钥。 有关设置的详细信息，请参见第 3.6.3 节封装密钥。
(3)	IV	设置要用于封装的初始向量 (IV)。 有关设置的详细信息，请参见第 3.6.4 节 IV。
(4)	输出	选择输出文件格式。请注意，并非所有器件都支持所有可能的输出文件格式。有关详细信息，请参见第 3.6.5 节输出。
(5)	生成文件	使用 (1) 和 (2) 中的信息以 (3) 中指定的格式生成用于安装或升级的封装密钥文件。 在 [概要] 选项卡上选择 MCU/MPU 以及加密引擎时启用。

3.6.1 [密钥类型] 选项卡

使用该选项卡指定为安全安装或升级准备的密钥类型。并非所有器件产品家族都支持所有密钥类型和选项。

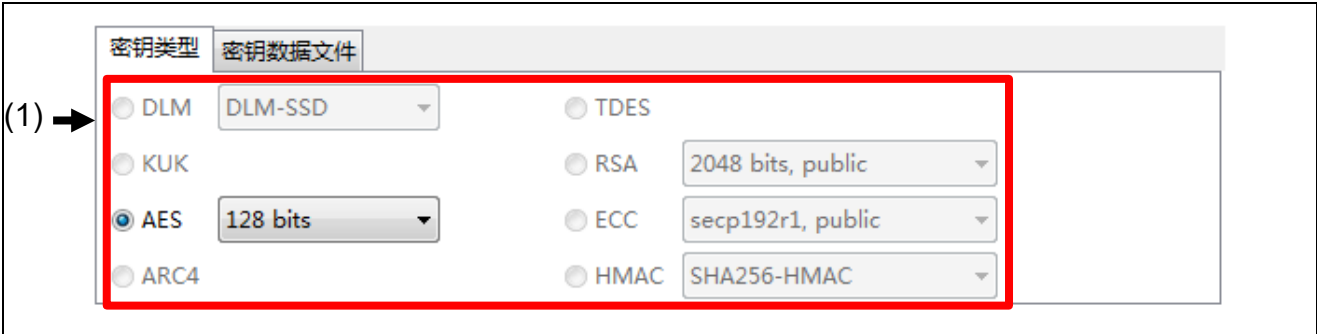


图 3-7 [密钥类型] 选项卡

编号	项目	说明
(1)	密钥类型和密钥长度选择	选择将要安装/升级的密钥的算法和密钥长度。

可以选择下列密钥类型和密钥长度。根据具体的目标 MCU/MPU，支持的密钥类型和密钥长度有所不同。有关支持的密钥类型和密钥长度的详细信息，请参见器件的硬件用户手册和附加器件信息（表 1-1 MCU/MPU 相关信息）。

表 3-1 选择了 DLM 时的选项

选项	说明
DLM-SSD	DLM 中 SSD 状态转换的身份验证密钥。
DLM-NSECSD	DLM 中 NSECSD 状态转换的身份验证密钥。
DLM-RMA-REQ	DLM 中 RMA-REQ 状态转换的身份验证密钥

表 3-2 选择了 AES 时的选项

选项	说明
128 bits	AES 128 位密钥
192 bits	AES 192 位密钥
256 bits	AES 256 位密钥
128 bits, XTS	AES 128 位 XTS 密钥
256 bits, XTS	AES 256 位 XTS 密钥

表 3-3 选择了 RSA 时的选项

选项	说明
1024 bits, public	RSA 1024 位公钥
1024 bits, private	RSA 1024 位私钥
2048 bits, public	RSA 2048 位公钥
2048 bits, private	RSA 2048 位私钥
3072 bits, public	RSA 3072 位公钥
3072 bits, private	RSA 3072 位私钥
4096 bits, public	RSA 4096 位公钥
4096 bits, private	RSA 4096 位私钥
RSA-2048-public-TLS	RSA 2048 位公钥，用于 TLS API

表 3-4 选择了 ECC 时的选项

选项	说明
secp192r1, public	ECC NIST P-192 (secp192r1) 公钥
secp192r1, private	ECC NIST P-192 (secp192r1) 私钥
secp224r1, public	ECC NIST P-224 (secp224r1) 公钥
secp224r1, private	ECC NIST P-224 (secp224r1) 私钥
secp256r1, public	ECC NIST P-256 (secp256r1) 公钥
secp256r1, private	ECC NIST P-256 (secp256r1) 私钥
secp384r1, public	ECC NIST P-384 (secp384r1) 公钥
secp384r1, private	ECC NIST P-384 (secp384r1) 私钥
brainpool P256, public	ECC brainpoolP256r1 公钥
brainpool P256, private	ECC brainpoolP256r1 私钥
brainpool P384, public	ECC brainpoolP384r1 公钥
brainpool P384, private	ECC brainpoolP384r1 私钥
brainpool P512, public	ECC brainpoolP512r1 公钥
brainpool P512, private	ECC brainpoolP512r1 私钥
secp256k1, public	ECC Koblitz 曲线 secp256k1 公钥
secp256k1, private	ECC Koblitz 曲线 secp256k1 私钥

表 3-5 选择了 HMAC 时的选项

选项	说明
SHA1-HMAC	HMAC-SHA1 密钥
SHA224-HMAC	HMAC-SHA224 密钥
SHA256-HMAC	HMAC-SHA256 密钥

3.6.2 [密钥数据文件] 选项卡

使用 [密钥数据文件] 选项卡指定为安全安装或升级准备的明文密钥材料。请注意，该选项卡的外观取决于在 [密钥类型] 选项卡上指定的密钥类型。

3.6.2.1 在 [密钥类型] 选项卡中选择了 DLM/KUK/AES/TDES/ARC4/ECC 私钥时

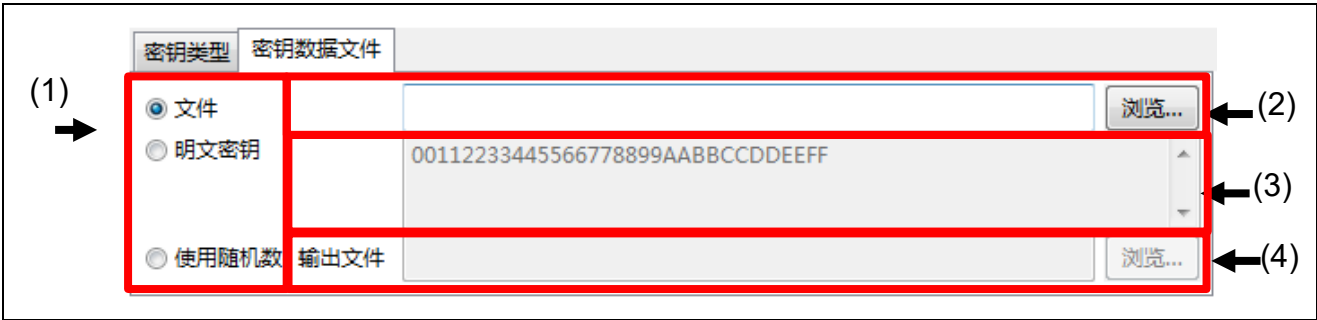


图 3-8 选择了 DLM/KUK/AES/TDES/ARC4/ECC 私钥时的 [密钥数据文件] 选项卡

编号	项目	说明
(1)	输入格式	选择是在 (2) 中提供包含密钥数据的密钥文件，还是在 (3) 中提供密钥的原始明文数据，亦或是通过工具使用 (4) 中指定的输出文件名生成密钥（或密钥对）。
(2)	文件名	在 (1) 中选择文件时，选择包含明文密钥的密钥文件。密钥文件必须为二进制文件，且扩展名为 *.key。
(3)	明文密钥数据	在 (1) 中选择明文密钥时，以十六进制格式输入明文密钥数据。数据量取决于在 [密钥类型] 选项卡上指定的密钥类型。
(4)	输出密钥文件	在 (1) 中选择使用随机数时，工具将生成密钥数据。指定工具生成的明文密钥数据的输出文件。输出明文密钥文件可以是文本文件或二进制文件。 要生成非对称密钥，请选择“私钥”类型。私钥和公钥将同时生成，并且会在指定的文件名中附加 _public（对于公钥）和 _private（对于私钥）。仅针对私钥生成安装/升级文件。可以使用生成的公钥文件作为输入来为公钥创建密钥安装文件。并非支持所有非对称密钥类型。请参见表 3-6 支持的非对称密钥生成。 注： 无法保证该工具随机生成的随机数的具体值。该工具生成的密钥应仅用于原型设计和测试目的。

表 3-6 支持的非对称密钥生成

算法	密钥类型和密钥长度
RSA	RSA-1024、RSA-2048、RSA-3072、RSA-4096
ECC	secp256r1、secp384r1 brainpool P256、brainpool P384、brainpool P512

3.6.2.2 在 [密钥类型] 选项卡中选择了 RSA 公钥时

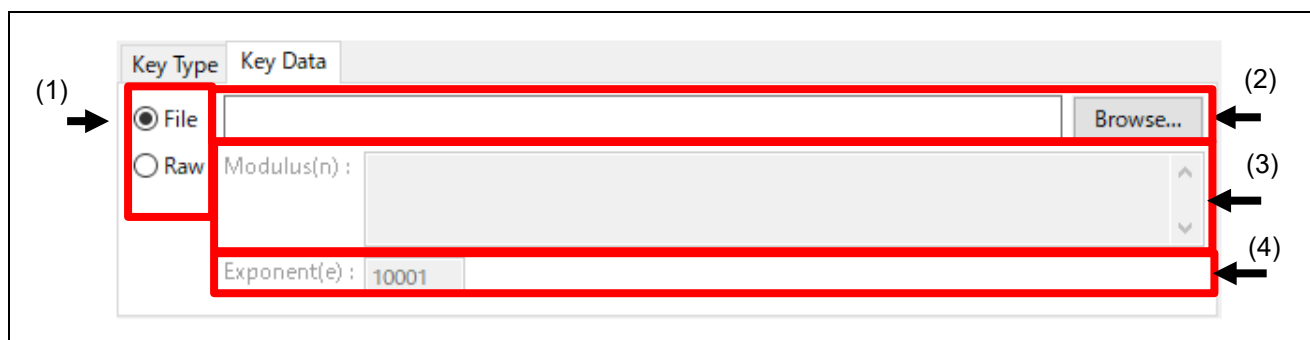


图 3-9 选择了 RSA 公钥时的 [密钥数据文件] 选项卡

编号	项目	说明
(1)	输入格式	选择是在 (2) 中提供包含密钥数据的密钥文件，还是在 (3) 和 (4) 中提供密钥的原始明文数据。
(2)	文件名	在 (1) 中选择 文件 时，选择包含明文密钥的密钥文件。密钥文件必须为二进制文件，且扩展名为 *.key。
(3)	模数 (n)	在 (1) 中选择 明文密钥 时，以十六进制格式输入 RSA 模数 n 的明文数据。
(4)	指数 (e)	在 (1) 中选择 明文密钥 时，以十六进制格式输入 RSA 指数 e 的明文数据。

3.6.2.3 在 [密钥类型] 选项卡中选择了 RSA 私钥时



图 3-10 选择了 RSA 私钥时的 [密钥数据文件] 选项卡

编号	项目	说明
(1)	输入格式	选择是在 (2) 中提供包含密钥数据的密钥文件，还是在 (3) 和 (4) 中提供密钥的原始明文数据，亦或是通过工具使用 (5) 中指定的输出文件名生成密钥对。
(2)	文件名	在 (1) 中选择 文件 时，选择包含明文密钥的密钥文件。密钥文件必须为二进制文件，且扩展名为 *.key。
(3)	模数 (n)	在 (1) 中选择 明文密钥 时，以十六进制格式输入 RSA 模数 n 的明文数据。
(4)	解密指数 (d)	在 (1) 中选择 明文密钥 时，以十六进制格式输入 RSA 解密指数 d 的明文数据。
(5)	输出密钥文件	在 (1) 中选择 使用随机数 时，工具将生成密钥数据。指定工具生成的明文密钥数据的输出文件。输出明文密钥文件可以是文本文件或二进制文件。 要生成非对称密钥，请选择“私钥”类型。私钥和公钥将同时生成，并且会在指定的文件名中附加 _public（对于公钥）和 _private（对于私钥）。仅针对私钥生成安装/升级文件。可以使用生成的公钥文件作为输入来为公钥创建密钥安装文件。并非支持所有非对称密钥类型。请参见表 3-6 支持的非对称密钥生成。 注：该工具生成的密钥应仅用于原型设计和测试目的。

3.6.2.4 在 [密钥类型] 选项卡中选择了 ECC 公钥时

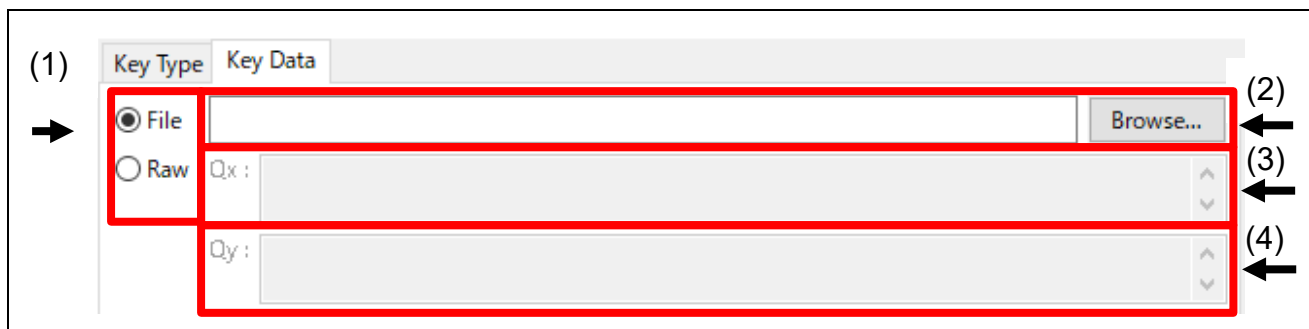


图 3-11 选择了 ECC 公钥时的 [密钥数据文件] 选项卡

编号	项目	说明
(1)	输入格式	选择是在 (2) 中提供包含密钥数据的密钥文件，还是在 (3) 和 (4) 中提供密钥的原始明文数据。
(2)	文件名	在 (1) 中选择 文件 时，选择包含明文密钥的密钥文件。密钥文件必须为二进制文件，且扩展名为 *.key。
(3)	Qx	在 (1) 中选择 明文密钥 时，以十六进制格式输入 ECC 公钥 Qx 的明文数据。
(4)	Qy	在 (1) 中选择 明文密钥 时，以十六进制格式输入 ECC 公钥 Qy 的明文数据。

3.6.3 封装密钥

如果为安全安装准备了新密钥，则必须使用 **UFPK** 对其进行封装，并且必须提供 **W-UFPK**。如果为安全升级准备了新密钥，则必须使用 **KUK** 对其进行封装。



图 3-12 封装密钥选项

编号	项目	说明
(1)	封装密钥类型	选择封装密钥的类型。如果为安全安装准备了新密钥，请选择 UFPK ，并在 (2) 中提供 UFPK 文件，在 (3) 中提供 W-UFPK 文件。如果为安全升级准备了新密钥，请选择 KUK ，并在 (4) 中提供 KUK 文件。
(2)	UFPK 文件	如果在 (1) 中选择了 UFPK ，则输入 UFPK *.key 文件。该文件是在 [生成 UFPK] 选项卡中生成的。
(3)	W-UFPK 文件	如果在 (1) 中选择了 UFPK ，则输入对应于指定 UFPK 的 W-UFPK *.key 文件。该文件必须从瑞萨密钥封装服务中获得。
(4)	KUK 文件	如果在 (1) 中选择了 KUK ，则输入 KUK *.key 文件。该文件是在 [生成 KUK] 选项卡中生成的。

3.6.4 IV

如果为安全安装准备了新密钥，则必须使用 UFPK 对其进行封装，并且必须提供 W-UFPK。如果为安全升级准备了新密钥，则必须使用 KUK 对其进行封装。在这两种情况下，均需要初始向量 (IV)。IV 可以指定，也可以由工具生成。



图 3-13 IV 选项

编号	项目	说明
(1)	IV 格式	选择是使用 (2) 中的输入值作为 IV 值还是使用工具生成的随机数值。无法保证该工具随机生成的随机数的具体值。
(2)	使用指定的值	当在 (1) 中选择了 使用指定的值 时，加密期间将使用此处输入的值作为初始向量。该值必须指定为大端格式 16 字节十六进制值。

3.6.5 输出

该部分指定为安全密钥安装或升级生成的输出文件的类型。请注意，并非所有 MCU/MPU 产品家族都支持所有选项。例如，RA 产品家族 SCE9 保护模式密钥安装仅支持通过器件编程器实现，因此，如果选择 UFPK 作为封装密钥，则仅支持 RFP 输出格式。



图 3-14 输出文件选项

编号	项目	说明
(1)	格式	选择输出的文件格式。有关可能的格式，请参见表 3-7 输出文件格式。
(2)	文件	设置输出文件名和路径。当指定非对称私钥并请求工具生成密钥对时，会在输出的加密密钥文件名中附加“_private”。
(3)	地址	当在 (1) 中选择了 Motorola 十六进制 时，会启用该设置。输入要在 Motorola 十六进制文件中设置的地址。
(4)	密钥名称	当在 (1) 中选择了 C 源文件 时，会启用该功能。该功能指定了 C 源文件和头文件中定义的结构、变量和数据大小值的 <keyname> 部分。有关如何使用 <keyname> 的详细说明，请参见第 53 节 filetype 的 csource 选项。

表 3-7 输出文件格式

选项	说明
C 源文件	输出 C 源文件和头文件。
二进制	输出二进制数据。
Motorola 十六进制	输出 Motorola 十六进制文件。
RFP	以瑞萨密钥文件格式输出密钥数据。

4. CLI 函数说明

4.1 命令行语法

skmt.exe [命令] [选项..]

注：命令、选项和参数不区分大小写。

表 4-1 命令行语法

项目	说明
skmt.exe	可执行文件名。
命令	密钥生成命令。命令以 “/” 或 “-” 开头。
选项 ..	指定的密钥生成命令有零个或多个选项。每个选项以 “/” 或 “-” 开头。

4.2 命令

命令如下表所示。

表 4-2 命令列表

命令	说明
genufpk	生成 UFPK 文件。 成功后，生成的 UFPK 将显示在控制台上。
genkuk	生成 KUK 文件。 成功后，生成的 KUK 将显示在控制台上。
genkey	加密用户密钥并输出文件。 成功后，控制台将显示 IV、W-UFPK 和加密的用户密钥（包括 MAC）。
H	帮助

该工具支持多种文件类型。通过使用表 4-3 中显示的扩展名指定所需的文件类型。支持绝对路径和相对路径。

表 4-3 文件类型和扩展名

文件类型	扩展名	说明
二进制密钥数据	*.key	明文密钥材料的二进制数据文件
瑞萨密钥文件	*.rkey	RFP 用于安全用户和 DLM 密钥安装（如果 MCU/MPU 支持）的文件。
Motorola 十六进制文件	*.mot、 *.srec	Motorola 十六进制文件
二进制数据	*.bin	二进制数据文件
C 源文件、头文件	*.c、*.h	C 源文件和头文件
文本文件	*.txt	用 ASCII 字符编写的文件
PEM 文件	*.pem	Base64 编码的 x509 ANS.1 密钥文件，用于 OpenSSL

4.3 genufpk 命令选项

该命令生成包含用户工厂烧录密钥 (UFPK) 的密钥文件。UFPK 可以指定，也可以由工具生成。

表 4-4 genufpk 选项

选项	参数	说明
ufpk	十六进制数据	为 UFPK 指定 32 字节二进制数据。 该选项为可选项。如果省略该选项，则将为 UFPK 生成随机数值。
output	文件路径	指定输出文件名。 该选项为可选项。如果省略该选项，则执行结果将输出到控制台。
nooverwrite	无	该选项为可选项。指定该选项时，如果存在输出文件，则将出现错误。

注：无法保证该工具随机生成的随机数的具体值。

省略 ufpk 选项的示例：

```
> skmt.exe /genufpk /output "D:\example\ufpk.key"
```

使用 ufpk 选项的示例：

```
> skmt.exe /genufpk
    /ufpk "00112233445566778899aabbccddeeff00112233445566778899aabbccddeeff"
    /output "D:\example\ufpk.key"
```

4.4 genkuk 命令选项

该命令生成包含升级密钥 (KUK) 的密钥文件。KUK 可以指定，也可以由工具生成。

表 4-5 genkuk 选项

选项	参数	说明
kuk	十六进制数据	指定 KUK 使用的 32 字节二进制数据。 该选项为可选项。如果省略该选项，则将使用工具中生成的随机数值作为 KUK。
output	文件路径	指定输出文件名。 该选项为可选项。如果省略该选项，则执行结果将输出到控制台。
nooverwrite	无	该选项为可选项。指定该选项时，如果在输出文件，则将出现错误。

注：无法保证该工具随机生成的随机数的具体值。

省略 kuk 选项的示例：

```
> skmt.exe /genkuk /output "D:\example\kuk.key"
```

使用 kuk 选项的示例：

```
> skmt.exe /genkuk /output "D:\example\kuk.key"
    /kuk "00112233445566778899aabbccddeeff00112233445566778899aabbccddeeff"
```

4.5 genkey 命令选项

该命令生成用于安全密钥安装或升级的文件。

以下选项可与 **genkey** 命令一起使用。如表 4-6 中所述，一些数据输入可以指定为十六进制数据或二进制文件形式。当指定文件时，在文件路径的开头添加“file=”。

表 4-6 genkey 选项

选项	参数	说明
iv	十六进制数据/文件路径	用于加密用户或 DLM 密钥（16 字节）的初始向量 (IV)。该选项为可选项。如果省略该选项，则将使用工具中生成的随机数值作为 IV。
ufpk	文件路径	包含用于安全密钥安装的 UFPK 的密钥文件。请注意，该 UFPK 必须对应于指定的 W-UFPK。当指定 kuk 选项时，无法指定 ufpk 选项。
wufpk	文件路径	包含瑞萨密钥封装服务提供的封装 UFPK 的密钥文件。当指定 kuk 选项时，无法指定 wufpk 选项。
kuk	十六进制数据/文件路径	要用于安全密钥升级的 KUK。当指定 ufpk 选项时，无法指定 kuk 选项。
mcu	ASCII	目标 MCU/MPU。该值必须为第 4.5.1 节 mcu 选项中列出的选项之一。
keytype	ASCII/十六进制数据	为安装或升级准备的密钥类型。可以指定密钥名称或其数字表示。该值必须为第 4.5.2 节 keytype 选项中列出的选项之一。
key	十六进制数据/文件路径	DLM 密钥数据或用户密钥数据。有关要输入的密钥数据格式，请参见第 4.5.3.1 节 十六进制数据直接输入 。该选项为可选项。如果省略该选项，则工具可能会生成密钥数据。但是，在公钥加密模式下，有一些密钥可能无法生成。有关详细信息，请参见第 4.5.3.3 节 省略 key 选项 。
filetype	ASCII	输出文件类型。请参见第 4.5.4 节 filetype 选项。 该选项为可选项。如果省略该选项并指定 output 选项，则将输出为 bin 文件。如果省略 filetype 和 output 选项，则将输出到控制台。
address	十六进制数据	要安装密钥的地址。当指定 mot 为 filetype 时，必须指定该选项。
keyname	ASCII	指定了 C 源文件和头文件中定义的结构、变量和数据大小值的 <keyname> 部分。当指定 csource 为 filetype 时，必须指定该选项。有关如何使用 <keyname> 的详细说明，请参见第 4.5.4.2 节 filetype 的 csource 选项 。
keyfileoutput	文件路径	该选项为可选项。如果省略 key 选项，则将由工具生成 DLM 或用户密钥，使用该选项可指定生成的密钥数据的输出路径。输出二进制数据。
output	文件路径	指定输出文件名。该选项为可选项。如果省略该选项，则执行结果将输出到控制台。
nooverwrite	无	该选项为可选项。指定该选项时，如果存在输出文件，则将出现错误。

注：无法保证该工具随机生成的随机数的具体值。

使用 **ufpk** 选项的示例：

```
> skmt.exe /genkey /ufpk file="D:\example\ufpk.key" /wufpk file="D:\example\ufpk_enc.key"
/mcu "RA-SCE9" /keytype "AES-128" /key "000102030405060708090A0B0C0D0E0F"
/filetype "rfp" /output "D:\example\aes128.rkey"
```

使用 **kuk** 选项的示例：

```
> skmt.exe /genkey /kuk file="D:\example\ufpk.key" /mcu "RA-SCE9" /keytype "AES-128"
/key "000102030405060708090A0B0C0D0E0F" /filetype "csource"
/output "D:\example\aes128.c"
```

4.5.1 mcu 选项

mcu 选项指定 MCU/MPU 类型和加密引擎。

表 4-7 mcu 选项

ASCII	说明
RA-SCE9	使用 RA 产品家族 SCE9 DLM 和保护模式时指定
RA-SCE9-CM	使用 RA 产品家族 SCE9 兼容模式时指定
RA-SCE7	使用 RA 产品家族 SCE7 时指定
RA-SCE5_B	使用 RA 产品家族 SCE5_B 时指定
RA-SCE5	使用 RA 产品家族 SCE5 时指定
RE-TSIPLite	使用 RE 产品家族 TSIP-Lite 时指定
RX-TSIP	使用 RX 产品家族 TSIP 时指定
RX-TSIPLite	使用 RX 产品家族 TSIP-Lite 时指定
RZ-TSIP	使用 RZ 产品家族 TSIP 时指定
Synergy-SCE7	使用 Synergy 平台 SCE7 时指定
Synergy-SCE5	使用 Synergy 平台 SCE5 时指定

4.5.2 keytype 选项

keytype 选项指定为安全安装或升级准备的密钥类型。该选项可以使用 ASCII 或十六进制值。为了便于阅读，建议使用 ASCII 值。

请注意，并非所有 MCU/MPU 都支持所有密钥类型。

表 4-8 用于 DLM 转换的身份验证密钥

ASCII	keytype 值	说明
DLM-SSD	0x01	在 DLM 中转换为 SSD 状态时使用的身份验证密钥。
DLM-NSECSD	0x02	在 DLM 中转换为 NSECSD 状态时使用的身份验证密钥。
DLM-RMA-REQ	0x03	在 DLM 中转换为 RMA_REQ 状态时使用的身份验证密钥。

表 4-9 用户密钥 AES

ASCII	keytype 值	说明
AES-128	0x05	AES-128 位密钥
AES-192	0x06	AES-192 位密钥
AES-256	0x07	AES-256 位密钥
AES-128XTS	0x08	AES-128 位 XTS 密钥
AES-256XTS	0x09	AES-256 位 XTS 密钥

表 4-10 用户密钥 RSA

ASCII	keytype 值	说明
RSA-1024-public	0x0A	RSA 1024 位公钥
RSA-1024-private	0x0B	RSA 1024 位私钥
RSA-2048-public	0x0C	RSA 2048 位公钥
RSA-2048-private	0x0D	RSA 2048 位私钥
RSA-3072-public	0x0E	RSA 3072 位公钥
RSA-3072-private	0x0F	RSA 3072 位私钥
RSA-4096-public	0x10	RSA 4096 位公钥
RSA-4096-private	0x11	RSA 4096 位私钥
RSA-2048-public-TLS	-	RSA 2048 位公钥，用于 TLS

注：“RSA-2048-public-TLS”不能通过 **keytype** 值指定，而必须使用 ASCII 指定。

表 4-11 用户密钥 ECC

ASCII	keytype 值	说明
secp192r1-public	0x12	ECC NIST P-192 (secp192r1) 公钥
secp192r1-private	0x13	ECC NIST P-192 (secp192r1) 私钥
secp224r1-public	0x14	ECC NIST P-224 (secp224r1) 公钥
secp224r1-private	0x15	ECC NIST P-224 (secp224r1) 私钥
secp256r1-public	0x16	ECC NIST P-256 (secp256r1) 公钥
secp256r1-private	0x17	ECC NIST P-256 (secp256r1) 私钥
secp384r1-public	0x18	ECC NIST P-384 (secp384r1) 公钥
secp384r1-private	0x19	ECC NIST P-384 (secp384r1) 私钥
brainpoolP256r1-public	0x1C	ECC brainpoolP256r1 公钥
brainpoolP256r1-private	0x1D	ECC brainpoolP256r1 私钥
brainpoolP384r1-public	0x1E	ECC brainpoolP384r1 公钥
brainpoolP384r1-private	0x1F	ECC brainpoolP384r1 私钥
brainpoolP512r1-public	0x20	ECC brainpoolP512r1 公钥
brainpoolP512r1-private	0x21	ECC brainpoolP512r1 私钥
secp256k1-public	0x22	ECC Koblitz 曲线 secp256k1 公钥
secp256k1-private	0x23	ECC Koblitz 曲线 secp256k1 私钥

表 4-12 用户密钥 SHA-HMAC

ASCII	keytype 值	说明
HMAC-SHA1	0x00	HMAC-SHA1 密钥
HMAC-SHA224	0x1A	HMAC-SHA224 密钥
HMAC-SHA256	0x1B	HMAC-SHA256 密钥

表 4-13 用户密钥 ARC4

ASCII	keytype 值	说明
ARC4	0x00	ARC4 密钥

表 4-14 用户密钥 DES

ASCII	keytype 值	说明
TDES	0x00	三重 DES 密钥

表 4-15 升级密钥

ASCII	keytype 值	说明
key-update-key	0xFF	升级密钥

4.5.3 key 选项

key 选项用于指定需要为安全安装或升级准备的明文 DLM 或用户密钥。明文可以通过直接输入十六进制数据或通过二进制 *.key 文件指定。某些密钥类型还可以由工具生成。

4.5.3.1 十六进制数据直接输入

如果明文密钥是通过直接输入十六进制数据提供的，则必须根据指定的 **keytype** 选项提供所需的字节数，具体请参照下表。

表 4-16 DLM-SSD、DLM-NSECSD、DLM-RMA-REQ

字节	数据
0-15	DLM 密钥数据

表 4-17 AES-128

字节	数据
0-15	AES-128 位密钥数据

表 4-18 AES-192

字节	数据
0-23	AES-192 位密钥数据

表 4-19 AES-256

字节	数据
0-31	AES-256 位密钥数据

表 4-20 AES-128XTS

字节	数据
0-15	AES-128 位密钥 1
16-31	AES-128 位密钥 2

表 4-21 AES-256XTS

字节	数据
0-31	AES-256 位密钥 1
32-63	AES-256 位密钥 2

表 4-22 RSA-1024-public

字节	数据
0-127	RSA 1024 位模数 n
128-131	RSA 1024 位指数 e

表 4-23 RSA-1024-private

字节	数据
0-127	RSA 1024 位模数 n
128-255	RSA 1024 位解密指数 d

表 4-24 RSA-2048-public/RSA-2048-public-TLS

字节	数据
0-255	RSA 2048 位模数 n
256-259	RSA 2048 位指数 e

表 4-25 RSA-2048-private

字节	数据
0-255	RSA 2048 位模数 n
256-511	RSA 2048 位解密指数 d

表 4-26 RSA-3072-public

字节	数据
0-383	RSA 3072 位模数 n
384-387	RSA 3072 位指数 e

表 4-27 RSA-3072-private

字节	数据
0-383	RSA 3072 位模数 n
384-767	RSA 3072 位解密指数 d

表 4-28 RSA-4096-public

字节	数据
0-511	RSA 4096 位模数 n
512-515	RSA 4096 位指数 e

表 4-29 RSA-4096-private

字节	数据
0-511	RSA 4096 位模数 n
512-1023	RSA 4096 位解密指数 d

表 4-30 secp192r1-public

字节	数据
0-23	ECC 公钥 Qx
24-47	ECC 公钥 Qy

表 4-31 secp192r1-private

字节	数据
0-23	ECC 私钥 d

表 4-32 secp224r1-public

字节	数据
0-27	ECC 公钥 Qx
28-55	ECC 公钥 Qy

表 4-33 secp224r1-private

字节	数据
0-27	ECC 公钥 d

表 4-34 secp256r1-public/brainpoolP256r1-public/secp256k1-public

字节	数据
0-31	ECC 公钥 Qx
32-63	ECC 公钥 Qy

表 4-35 secp256r1-private/brainpoolP256r1-private/secp256k1-private

字节	数据
0-31	ECC 私钥 d

表 4-36 secp384r1-public/brainpoolP384r1-public

字节	数据
0-47	ECC 公钥 Qx
48-95	ECC 公钥 Qy

表 4-37 secp384r1-private/brainpoolP384r1-private

字节	数据
0-47	ECC 私钥 d

表 4-38 brainpoolP512r1-public

字节	数据
0-63	ECC 公钥 Qx
64-127	ECC 公钥 Qy

表 4-39 brainpoolP512r1-private

字节	数据
0-63	ECC 私钥 d

表 4-40 HMAC-SHA1

字节	数据
0-19	HMAC-SHA1 密钥

表 4-41 HMAC-SHA224

字节	数据
0-27	HMAC-SHA224 密钥

表 4-42 HMAC-SHA256

字节	数据
0-31	HMAC-SHA256 密钥

表 4-43 ARC4

字节	数据
0-255	ARC4 密钥

表 4-44 TDES

字节	数据
0-7	带奇校验 1 的 56 位 DES 密钥
8-15	带奇校验 2 的 56 位 DES 密钥
16-23	带奇校验 3 的 56 位 DES 密钥

注：为每 7 位密钥数据添加奇校验。

示例：

- DES 密钥数据 = 0x0000000000000000 -> 0x0101010101010101
- DES 密钥数据 = 0xFFFFFFFFFFFFFFFF -> 0xFEFEFEFEFEFEFEFEFE

表 4-45 升级密钥

字节	数据
0-31	升级密钥

十六进制数据的使用示例：

```
> skmt.exe /genkey /ufpk file="D:\example\ufpk.key" /wufpk file="D:\example\ufpk_enc.key"
/mcu "RA-SCE9" /keytype "AES-128" /key "000102030405060708090A0B0C0D0E0F"
/filetype "rfp" /output "D:\example\aes128.rkey"
```

4.5.3.2 文件输入

key 选项支持以下文件格式的文件输入

表 4-46 Key 选项输入文件

文件类型	扩展名	格式
二进制文件	*.bin、*.key	二进制数据文件，格式如第 4.5.3.1 节十六进制数据直接输入所示。
文本文件	*.txt	以十六进制 ASCII 字符表示字节数据的输入文件，格式如第 4.5.3.1 节十六进制数据直接输入所示。
PEM	*.pem	可以读取 OpenSSL 生成的非对称密钥的 PEM 文件格式密钥文件。仅支持表 4-47 支持非对称密钥的 PEM 文件中指定的密钥类型

表 4-47 支持非对称密钥的 PEM 文件

算法	keytype
RSA	RSA-1024-private、RSA-1024-public、RSA-2048-private、RSA-2048-public、RSA-3072-private、RSA-3072-public、RSA-4096-private、RSA-4096-public
ECC	secp256r1-private、secp256r1-public、secp384r1-private、secp384r1-public brainpoolP256r1-private、brainpoolP256r1-public、brainpoolP384r1-private、 brainpoolP384r1-public、brainpoolP512r1-private、brainpoolP512r1-public

指定二进制文件的示例：

```
> skmt.exe /genkey /ufpk file="D:\example\ufpk.key" /wufpk file="D:\example\ufpk_enc.key"
/mcu "RA-SCE9" /keytype "AES-128" /key file="D:\example\aes128.key" /filetype "rfp"
/output "D:\example\aes128.rkey"
```

指定文本文件的示例：

```
> skmt.exe /genkey /ufpk file="D:\example\ufpk.key" /wufpk file="D:\example\ufpk_enc.key"
/mcu "RA-SCE9" /keytype "AES-128" /key file="D:\example\aes128.txt" /filetype "rfp"
/output "D:\example\aes128.rkey"
```

指定 PEM 文件的示例：

```
> skmt.exe /genkey /ufpk file="D:\example\ufpk.key" /wufpk file="D:\example\ufpk_enc.key"
/mcu "RA-SCE9" /keytype "RSA-2048-private" /key file="D:\example\rsa2048.pem" /filetype
"rfp" /output "D:\example\rsa2048.rkey"
```

要手动创建密钥数据，请使用十六进制编辑器或二进制编辑器，并在十六进制编辑器中以大端顺序创建数据。示例如下所示。

- 手动创建 AES 128 位密钥文件的示例：

AES 128 位密钥 00112233445566778899AABBCCDDEEFF

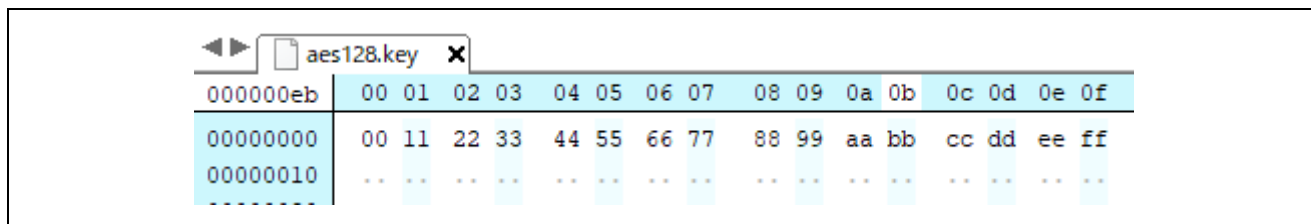


图 4-1 手动创建 AES 128 位密钥文件的示例

- 手动创建 RSA 2048 公钥文件的示例：

```

模数 bad47a84c1782e4dbdd913f2a261fc8b
    65838412c6e45a2068ed6d7f16e9cdf4
    462b39119563cafb74b9cbf25cf544b
    dae23bff0ebe7f6441042b7e109b9a8a
    faa056821ef8efaab219d21d67634847
    85622d918d395a2a31f2ece8385a8131
    e5ff143314a82e21afd713bae817cc0e
    e3514d4839007ccb55d68409c97a18ab
    62fa6f9f89b3f94a2777c47d6136775a
    56a9a0127f682470bef831fbec4bcd7b
    5095a7823fd70745d37d1bf72b63c4b1
    b4a3d0581e74bf9ade93cc4614861755
    3931a79d92e9e488ef47223ee6f6c061
    884b13c9065b591139de13c1ea292749
    1ed00fb793cd68f463f5f64baa53916b
    46c818ab99706557a1c2d50d232577d1
指数 10001
  
```

00000157	00	01	02	03	04	05	06	07	08	09	0a	0b	0c	0d	0e	0f
00000000	ba	d4	7a	84	c1	78	2e	4d	bd	d9	13	f2	a2	61	fc	8b
00000010	65	83	84	12	c6	e4	5a	20	68	ed	6d	7f	16	e9	cd	f4
00000020	46	2b	39	11	95	63	ca	fb	74	b9	cb	f2	5c	fd	54	4b
00000030	da	e2	3b	ff	0e	be	7f	64	41	04	2b	7e	10	9b	9a	8a
00000040	fa	a0	56	82	1e	f8	ef	aa	b2	19	d2	1d	67	63	48	47
00000050	85	62	2d	91	8d	39	5a	2a	31	f2	ec	e8	38	5a	81	31
00000060	e5	ff	14	33	14	a8	2e	21	af	d7	13	ba	e8	17	cc	0e
00000070	e3	51	4d	48	39	00	7c	cb	55	d6	84	09	c9	7a	18	ab
00000080	62	fa	6f	9f	89	b3	f9	4a	27	77	c4	7d	61	36	77	5a
00000090	56	a9	a0	12	7f	68	24	70	be	f8	31	fb	ec	4b	cd	7b
000000a0	50	95	a7	82	3f	d7	07	45	d3	7d	1b	f7	2b	63	c4	b1
000000b0	b4	a3	d0	58	1e	74	bf	9a	de	93	cc	46	14	86	17	55
000000c0	39	31	a7	9d	92	e9	e4	88	ef	47	22	3e	e6	f6	c0	61
000000d0	88	4b	13	c9	06	5b	59	11	39	de	13	c1	ea	29	27	49
000000e0	1e	d0	0f	b7	93	cd	68	f4	63	f5	f6	4b	aa	53	91	6b
000000f0	46	c8	18	ab	99	70	65	57	a1	c2	d5	0d	23	25	77	d1
00000100	00	01	00	01

图 4-2 手动创建 RSA 2048 公钥文件的示例

4.5.3.3 省略 **key** 选项

如果从命令行中省略 **key** 选项，并且为下表中指定的对称算法或非对称算法指定了 **keytype**，则工具将生成随机密钥值。

对于非对称密钥，将生成密钥对。私钥和公钥将同时生成，并且会在指定的文件名中附加 **_public**（对于公钥）和 **_private**（对于私钥）。该选项支持下表中显示的非对称 **keytype** 选项。

建议使用 **keyfileoutput** 选项创建包含明文密钥的密钥文件。

无法保证该工具随机生成的随机数的具体值。该工具生成的密钥应仅用于原型设计和测试目的。

表 4-48 支持非对称密钥生成功能的 **keytype**

算法	keytype
RSA	RSA-1024-private、RSA-2048-private、RSA-3072-private、RSA-4096-private
ECC	secp256r1-private、secp384r1-private, brainpoolP256r1-private、brainpoolP384r1-private、brainpoolP512r1-private

4.5.4 filetype 选项

可以使用 **filetype** 选项指定已准备密钥的输出文件格式。如果省略该选项，则输出二进制数据。如果指定的 **filetype** 与文件扩展名不匹配，则工具将返回错误。

表 4-49 filetype 选项

ASCII	说明
rfp	以瑞萨闪存编程器 (RFP) 可读取的文件格式输出文件。输出文件扩展名必须为 *.rkey。
csource	输出 C 源文件和头文件。输出文件扩展名必须为 *.c。
bin	输出二进制数据文件。输出文件扩展名必须为 *.bin。
mot	以 Motorola 十六进制格式输出二进制数据。输出文件扩展名必须为 *.mot。 在 address 选项中指定具有 8 个十六进制数字 (32 位) 的起始地址。

4.5.4.1 filetype 的 rfp 选项

该选项以瑞萨闪存编程器要使用的瑞萨密钥文件格式输出密钥数据。

RFP 文件输出示例:

```
> skmt.exe /genkey /ufpk file="D:\example\ufpk.key" /wufpk file="D:\example\ufpk_enc.key"
/mcu "RA-SCE9" /keytype "AES-128" /key file="D:\example\aes128.key" /filetype "rfp"
/output "D:\example\abc.rkey"
```

4.5.4.2 filetype 的 csource 选项

该选项输出 C 源文件和头文件。

当使用 **keyname** 选项时，指定的 <keyname> 将用作结构名称、全局变量名称和字节大小定义的一部分。如果不使用 **keyname** 选项，则将使用默认文本。下表显示了这些选项。

表 4-50 keyname 选项用法

	设置 keyname	省略 keyname
结构名称	<keyname>_t	encrypted_user_key_data_t
全局变量名称	g_<keyname>	g_encrypted_user_key_data
encrypted_user_key 字节大小定义值	<KEYNAME>_SIZE *	ENCRYPTED_KEY_BYTE_SIZE

* <keyname> 将转换为大写字母形式

输出 C 源结构如下所示，其中 <keyname> 按照表 4-50 keyname 选项进行替换。

表 4-51 指定 **wufpk** 选项时采用 **csource** 格式的结构输出

名称	类型	说明
g_<keyname>	<keyname>_t	-
keytype	uint32_t	当 mcu 选项为 “RA-SCE9” 或 “RA-SCE5_B” 时输出。否则，输出 0。
shared_key_number	uint32_t	输出 0。当 mcu 选项为 “RX-TSIP”、“RX-TSIP Lite” 或 “RE-TSIP Lite” 时不使用。
wufpk[32]	uint8_t	当指定 wufpk 选项时输出。如果未指定 wufpk 选项，则输出 0。
initial_vector[16]	uint8_t	用于用户密钥加密的初始向量。
encrypted_user_key [<KEYNAME>_SIZE]	uint8_t	加密的用户密钥。 <KEYNAME>_SIZE 表示加密的用户密钥大小。有关加密的用户密钥大小，请参见表 4-55 至表 4-62。
crc[4]	uint8_t	从 keytype 到 encrypted_user_key 的 CRC-32。

表 4-52 指定 **kuk** 选项时采用 “**csource**” 格式的结构输出

名称	类型	说明
g_<keyname>	<keyname>_t	-
keytype	uint32_t	当 mcu 选项为 “RA-SCE9” 或 “RA-SCE5_B” 时输出。否则，输出 0。
shared_key_number	uint32_t	输出 0。当 mcu 选项为 “RX-TSIP”、“RX-TSIP Lite” 或 “RE-TSIP Lite” 时不使用。
initial_vector[16]	uint8_t	用于用户密钥加密的初始向量。
encrypted_user_key [<KEYNAME>_SIZE]	uint8_t	加密的用户密钥。 <KEYNAME>_SIZE 表示加密的用户密钥大小。有关加密的用户密钥大小，请参见表 4-55 至表 4-62。
crc[4]	uint8_t	从 keytype 到 encrypted_user_key 的 CRC-32。

C 源文件输出示例：

```
> skmt.exe /genkey /wufpk file="D:\example\wufpk.key" /wufpk file="D:\example\wufpk_enc.key"
/mcu "RX-TSIP" /keytype "AES-128" /key file="D:\example\aes128.key" /filetype "csource"
/keyname testKey /output "D:\example\abc.c"
```

4.5.4.3 filetype 的 bin 选项

该选项输出二进制数据文件。输出二进制数据的数据数组如下所示：

表 4-53 指定 **ufpk** 选项时采用 **bin** 格式的二进制数据输出

字节	数据
0	Keytype 当 mcu 选项为 “RA-SCE9” 或 “RA-SCE5_B” 时输出。否则，输出 0。
1 - 3	保留（0 填充）
4	shared_key 输出 0。当 mcu 选项为 “RX-TSIP”、“RX-TSIPLite” 或 “RE-TSIPLite” 时不使用。
5 - 7	保留（0 填充）
8 - 39	WUFPK 当指定 wufpk 选项时输出。如果未指定 wufpk 选项，则输出 0。
40 - 55	initial_vector
56 - (56+N-1)	encrypted_user_key
(56+N) - 56+N +3	crc 从 keytype 到 encrypted_user_key 的 CRC-32。

注：“N” 与 <KEYNAME>_SIZE 值相同。

表 4-54 指定 **kuk** 选项时采用 **bin** 格式的二进制数据输出

字节	数据
0	Keytype 当 mcu 选项为 “RA-SCE9” 或 “RA-SCE5_B” 时输出。否则，输出 0。
1 - 3	保留（0 填充）
4	shared_key 输出 0。当 mcu 选项为 “RX-TSIP”、“RX-TSIPLite” 或 “RE-TSIPLite” 时不使用。
5 - 7	保留（0 填充）
8 - 23	initial_vector
24 - (24+N-1)	encrypted_user_key
(24+N) - 24+N +3	crc 从 keytype 到 encrypted_user_key 的 CRC-32。

注：“N” 与 <KEYNAME>_SIZE 值相同。

二进制文件输出示例：

```
> skmt.exe /genkey /ufpk file="D:\example\ufpk.key" /wufpk file="D:\example\ufpk_enc.key"
/mcu "RX-TSIP" /keytype "AES-128" /key file="D:\example\aes128.key" /filetype "bin"
/output "D:\example\abc.bin"
```

4.5.4.4 filetype 的 mot 选项

该选项输出 Motorola 十六进制格式文件。文件格式在表 4-53 和表 4-54 中指定。必须使用 **address** 选项并指定十六进制 8 位（32 位）起始地址来设置数据的地址。

mot 文件输出示例：

```
> skmt.exe /genkey /ufpk file="D:\example\ufpk.key" /wufpk file="D:\example\ufpk_enc.key"
/mcu "RX-TSIP" /keytype "AES-128" /key file="D:\example\aes128.key" /filetype "mot"
/address "FFF00000" /output "D:\example\abc.mot"
```

每个 **keytype** 选项的 **encrypted_user_key** 大小如下所示。

表 4-55 DLM 密钥的 **encrypted_user_key** 大小

keytype 选项	字节大小
DLM-SSD	32
DLM-NSECSD	32
DLM-RMA-REQ	32

表 4-56 AES 密钥的 **encrypted_user_key** 大小

keytype 选项	字节大小
AES-128	32
AES-192	48
AES-256	48
AES-128XTS	48
AES-256XTS	80

表 4-57 RSA 密钥的 **encrypted_user_key** 大小

keytype 选项	字节大小
RSA-1024-public	160
RSA-1024-private	272
RSA-2048-public	288
RSA-2048-private	528
RSA-3072-public	416
RSA-3072-private	784
RSA-4096-public	544
RSA-4096-private	1040
RSA 2048 位公钥，用于 TLS	288

表 4-58 ECC 密钥的 encrypted_user_key 大小

keytype 选项	字节大小
secp192r1-public	80
secp192r1-private	48
secp224r1-public	80
secp224r1-private	48
secp256r1-public	80
secp256r1-private	48
brainpoolP256r1-public	80
brainpoolP256r1-private	48
brainpoolP384r1-public	112
brainpoolP384r1-private	64
brainpoolP512r1-public	144
brainpoolP512r1-private	80
secp256k1-public	80
secp256k1-private	48

表 4-59 HMAC-SHA 密钥的 encrypted_user_key 大小

keytype 选项	字节大小
HMAC-SHA1	48
HMAC-SHA224	48
HMAC-SHA256	48

表 4-60 ARC4 密钥的 encrypted_user_key 大小

keytype 选项	字节大小
ARC4	272

表 4-61 TDES 密钥的 encrypted_user_key 大小

keytype 选项	字节大小
TDES	48

表 4-62 升级密钥的 encrypted_user_key 大小

keytype 选项	字节大小
key-update-key	48

4.5.5 keyfileoutput 选项

如果从命令行中省略 **key** 选项，则工具将生成随机密钥。**keyfileoutput** 选项用于将生成的密钥保存为二进制密钥文件或文本文件。指定的文件扩展名可以是 *.key 或 *.txt。

该工具只能生成对称密钥对和一些非对称密钥对。要生成非对称密钥对，请将私钥指定为 **keytype**。下表显示了可以生成的非对称密钥对类型。

表 4-63 可以生成的非对称密钥类型

算法	keytype
RSA	RSA-1024、RSA-2048、RSA-3072、RSA-4096
ECC	secp256r1、secp384r1 brainpool P256r1、brainpool P384r1、brainpool P512r1

无法保证该工具随机生成的随机数的具体值。该工具生成的密钥应仅用于原型设计和测试目的。

当生成非对称密钥对且指定 **/keyfileoutput** 时，会在私钥的文件名中附加 “_private”，在公钥的文件名中附加 “_public”。例如，如果指定 “/keyfileoutput abc.key /output abc.rkey”，则会生成以下文件：

- 包含明文私钥的 abc_private.key
- 包含明文公钥的 abc_public.key
- 包含私钥安装用加密密钥的 abc_private.rkey
- 包含公钥安装用加密密钥的 abc_public.rkey

要生成用于安装公钥的密钥安装文件，请使用适当的公钥 **keytype** 运行工具，并使用生成的公钥文件作为密钥数据文件。

有关输出密钥数据格式，请参见第 4.5.3.1 节十六进制数据直接输入中定义的格式。

对称密钥文件输出示例：

```
> skmt.exe /genkey /ufpk file="D:\example\ufpk.key" /wufpk file="D:\example\ufpk_enc.key"
/mcu "RA-SCE9" /keytype "AES-128" /filetype "rfp" /keyfileoutput file="D:\example\aes.key"
/output "D:\example\abc.rkey"
```

非对称密钥文件输出示例：

```
> skmt.exe /genkey /ufpk file="D:/example/ufpk.key" /wufpk file="D:/example/ufpk_enc.key"
/mcu "RX-TSIP" /keytype "RSA-1024-private" /filetype "bin"
/keyfileoutput file="D:/example/rsa1024.key" /output "D:/example/rsa1024_private.bin"
```

5. 操作程序

5.1 独立

5.1.1 Windows

运行安装程序 `SecurityKeyManagementTool_installer_v103.exe`，并将其安装到任意文件夹中。

注：安装到层级尽可能浅的文件夹中，并且该文件夹应具有写权限，同时文件夹名称中没有“空格”。如果层级太深或文件夹名称太长，则可能无法运行。

5.1.1.1 GUI 版本

安装文件夹中的 `SecurityKeyManagementTool.exe` 是可执行文件。

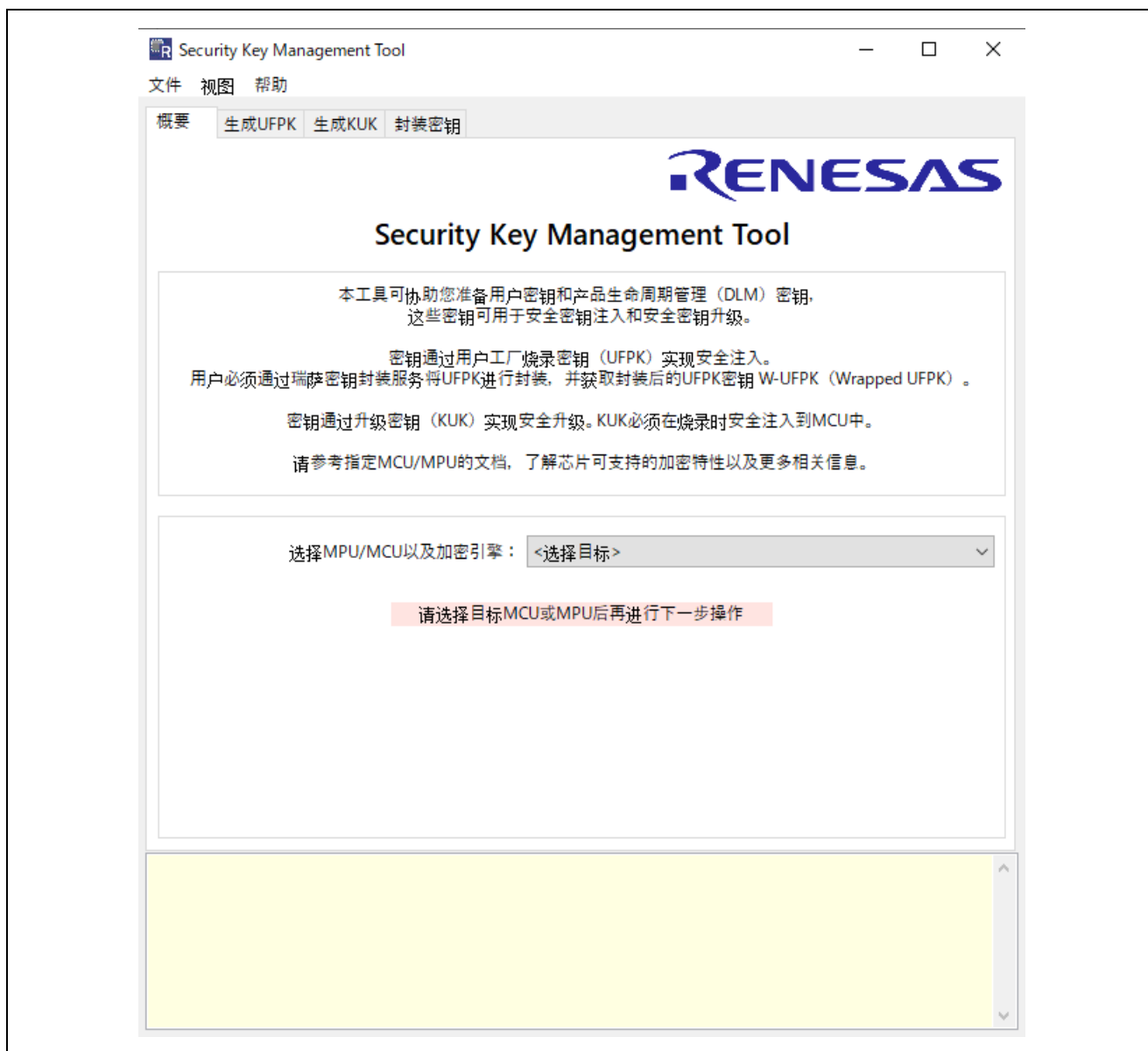


图 5-1 安全密钥管理工具 – 从 Windows 启动时的 GUI 对话框

5.1.1.2 CLI 版本

存储在已安装 **CLI** 文件夹中的文件是执行安全密钥管理工具 CLI 版本所需的全套文件。如果要仅使用该工具的 CLI 版本（例如，在生产环境中），则可以将这些文件和**文件夹**（加粗）移动或复制到另一文件夹中以便于操作。下面列出了执行 CLI 版本所需的全套文件。

- skmt.exe
- clrcompression.dll
- clrjit.dll
- coracle.dll
- mscordacore.dll
- **device**

从命令提示符中输入 skmt.exe 命令。

```
C:\work\skmt\tool>skmt.exe /genkey /ufpk file="C:\work\ufpk.key" /wufpk file="C:\work\ufpk_enc.key" /mcu "RA-SCE9" /keytype "AES-128" /key "000102030405060708090A0B0C0D0E0F" /filetype "rfp" /output "c:\work\aes128.rkey"
Output File: c:\work\aes128.rkey
UFPK: EC6B8FA5C0D5DA5142CCAF3A31AEBEAE2346CFE7EF644B9B6B70523CBA0F5C5C
W-UFPK: 000000004F4C172DEF2789DE4F041294C6B1218D11DC81DC5B37FB72DB899DCFF4BE7158
IV: 46EB124312C83FA226994D17A3F5616A
Encrypted key: DEAE066D5845A01E3FBC0445EC1141F60195D3B3F159D2B624A259BE4965A41D

C:\work\skmt\tool>
```

图 5-2 安全密钥管理工具 - 从命令提示符执行 CLI 示例

5.1.2 Linux 版本

5.1.2.1 GUI 版本

解压缩 Linux 版本包后，将 **SecurityKeyManagementTool** 文件夹放在任意文件夹中。
解压缩时，使用命令 **tar xvzfp xxxxx.tar.gz** 保持执行权限。



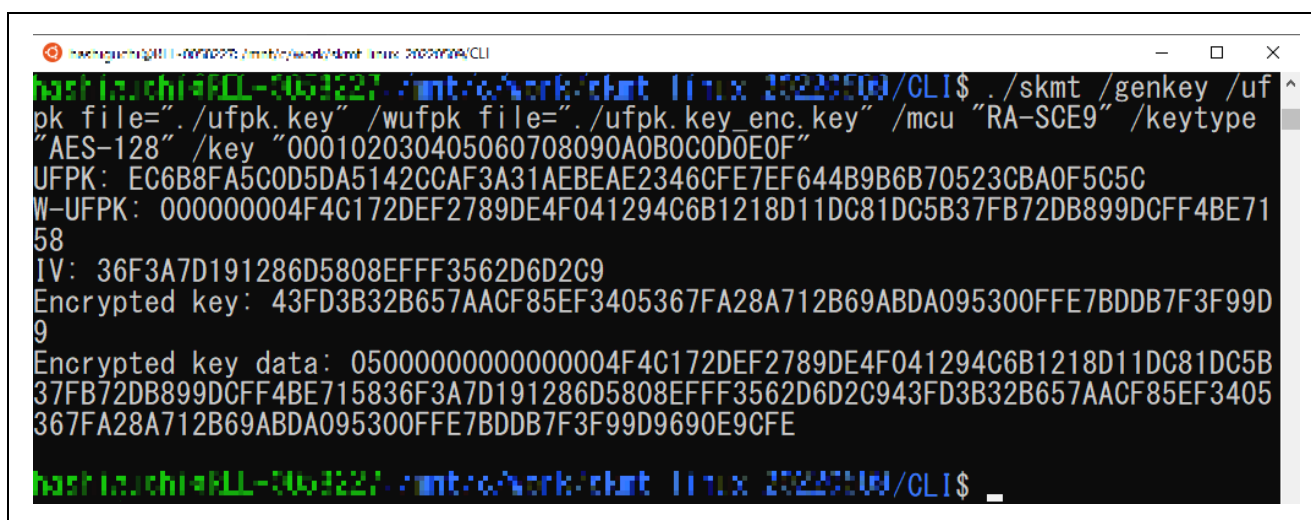
图 5-3 安全密钥管理工具 – 从 Linux 启动时的 GUI 对话框

5.1.2.2 CLI 版本

存储在已安装 **CLI** 文件夹中的文件是执行安全密钥管理工具 **CLI** 版本所需的全套文件。如果要仅使用该工具的 **CLI** 版本（例如，在生产环境中），则可以将这些文件移动或复制到另一文件夹中以便于操作。下面列出了执行 **CLI** 版本所需的全套文件。

- skmt

从终端软件中输入 **skmt** 命令。



```
hash@chi-4LL-063227:~/mnt/work/skmt Linux 20220904/CLI$ ./skmt /genkey /ufpk file="./ufpk.key" /wufpk file="./ufpk.key_enc.key" /mcu "RA-SCE9" /keytype "AES-128" /key "000102030405060708090A0B0C0D0E0F"
UFPK: EC6B8FA5C0D5DA5142CCAF3A31AEBEAE2346CFE7EF644B9B6B70523CBA0F5C5C
W-UFPK: 000000004F4C172DEF2789DE4F041294C6B1218D11DC81DC5B37FB72DB899DCFF4BE7158
IV: 36F3A7D191286D5808EFFF3562D6D2C9
Encrypted key: 43FD3B32B657AACF85EF3405367FA28A712B69ABDA095300FFE7BDDDB7F3F99D9
Encrypted key data: 05000000000000004F4C172DEF2789DE4F041294C6B1218D11DC81DC5B37FB72DB899DCFF4BE715836F3A7D191286D5808EFFF3562D6D2C943FD3B32B657AACF85EF3405367FA28A712B69ABDA095300FFE7BDDDB7F3F99D9690E9CFE
hash@chi-4LL-063227:~/mnt/work/skmt Linux 20220904/CLI$
```

图 5-4 安全密钥管理工具 - 从 Linux 终端软件执行 CLI 示例

5.2 e²studio 插件

Windows 和 Linux 版本的安装和卸载程序相同。请按照以下步骤安装和卸载。

5.2.1 安装 e²studio 插件版本

1. 从瑞萨网站下载 SecurityKeyManagementTool_Plugin_vXXX_Windows.zip 或 SecurityKeyManagementTool_Plugin_vXXX_Linux.zip，并将其放在所选的任意文件夹中。vXXX 为该工具的版本。
2. 启动 e²studio
3. 选择 e²studio 菜单“帮助(H)” – “安装新软件...”菜单，打开“安装”对话框。

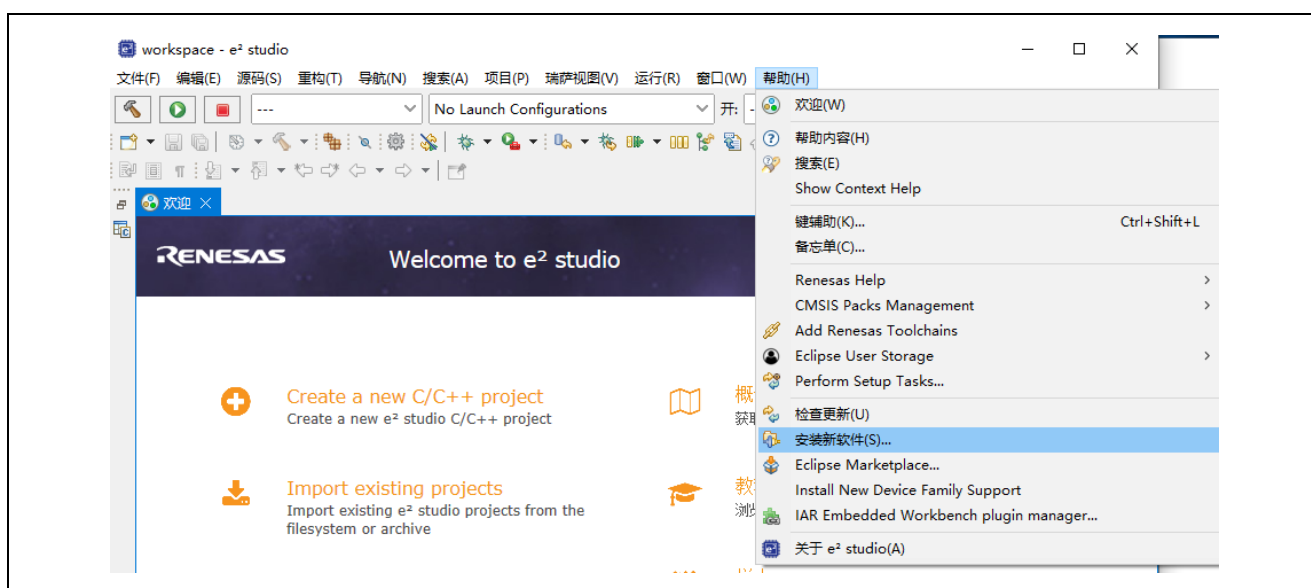


图 5-5 e²studio “帮助(H)” – “安装新软件...”

4. 在“安装”对话框中按“添加”按钮，打开“添加资源库”对话框。

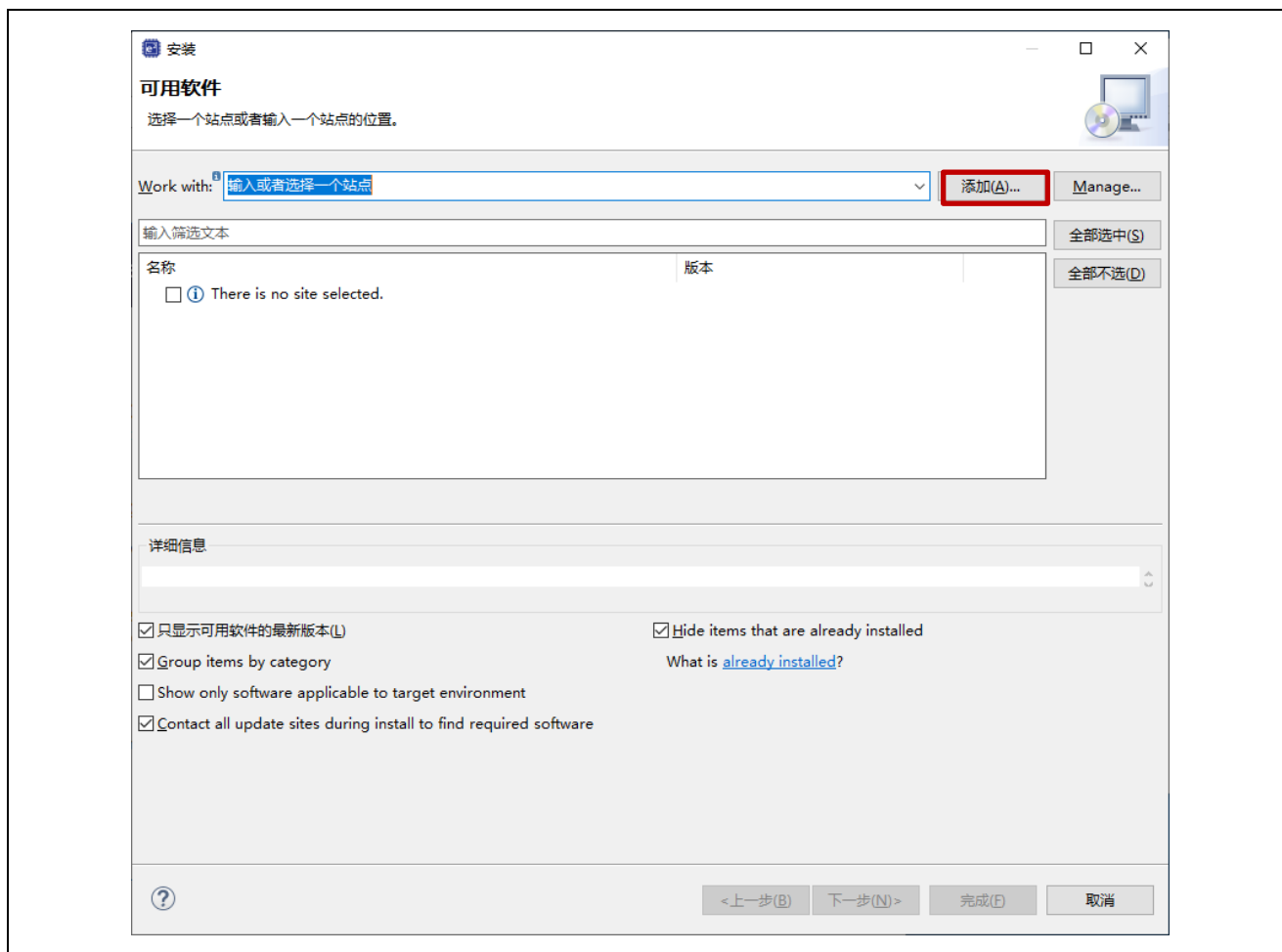


图 5-6 “安装”对话框

5. 在“添加资源库”对话框中单击“归档文件(A)...”，指定在第 1 步中准备的 SecurityKeyManagementTool_Plugin_v103_Window/Linux.zip 文件，然后单击“添加”按钮。

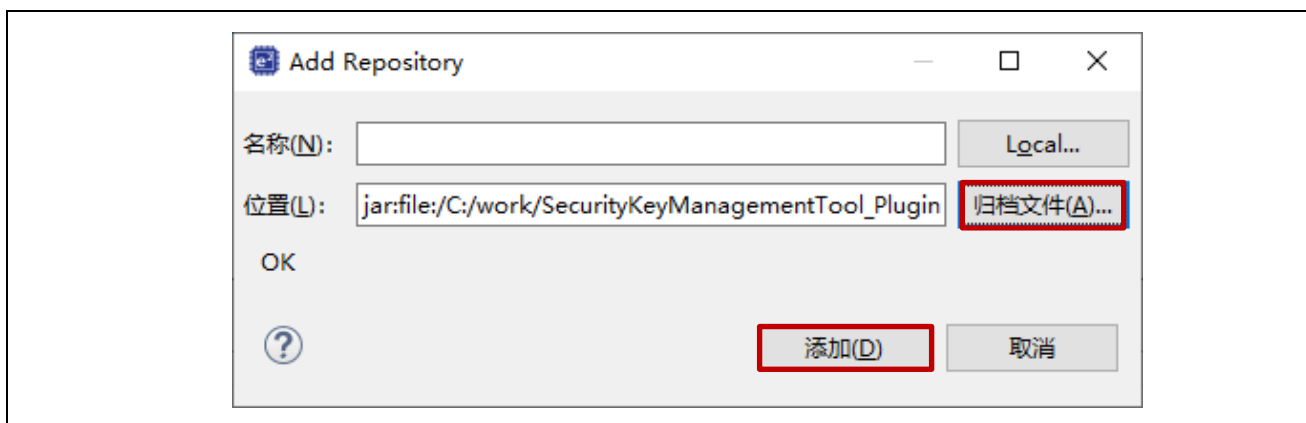


图 5-7 “添加资源库”对话框

6. “Renesas Solution Toolkit” 将添加到“安装”对话框中。选中复选框并按“下一步 >”按钮。不要勾选“Contact all update sites during install to find required software”。如果勾选的话，将会增加安装插件所需的时间

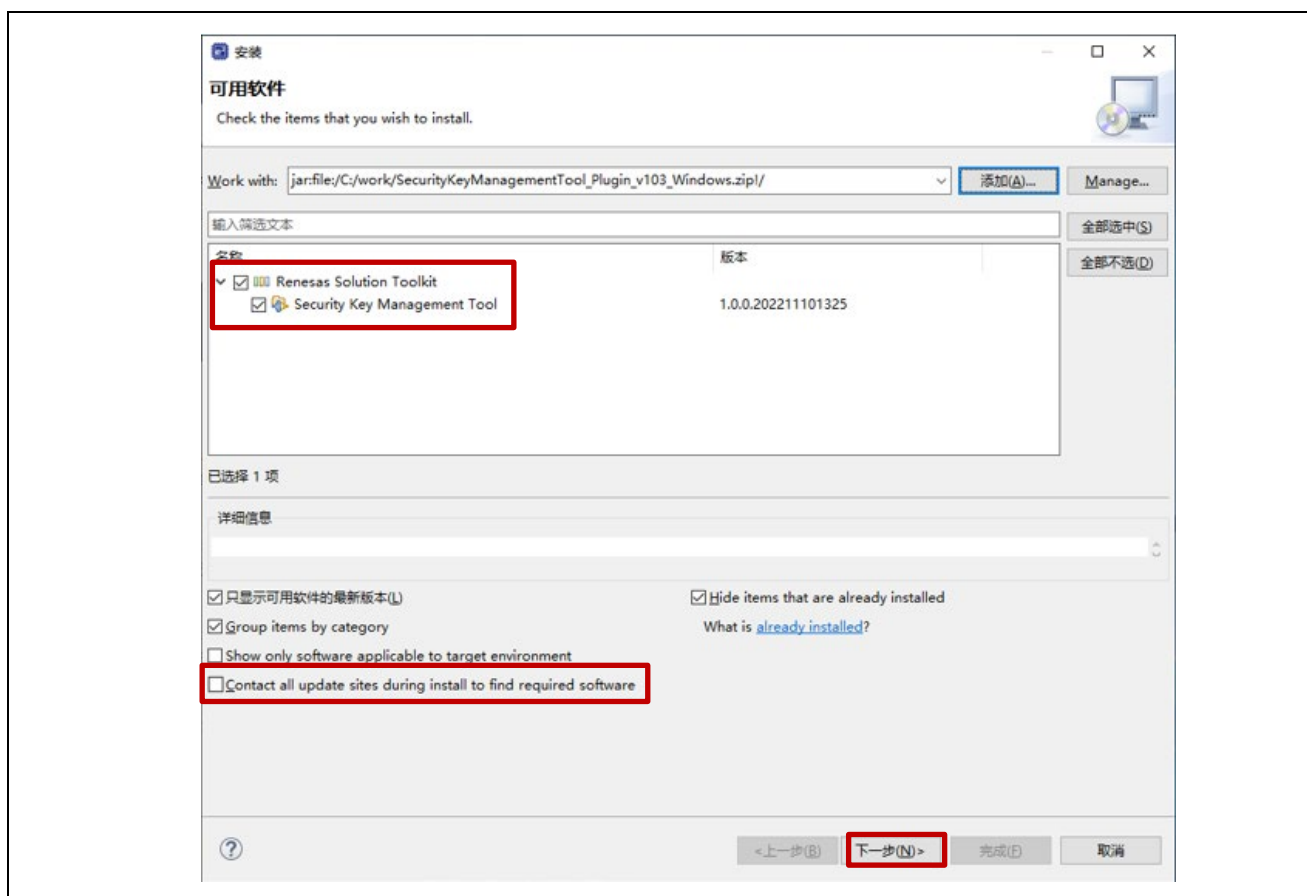


图 5-8 “安装”对话框 – 选择“Security Key Management Tool”

7. 当出现“安装”对话框时，确认安装“Security Key Management Tool”，然后按“下一步”按钮。

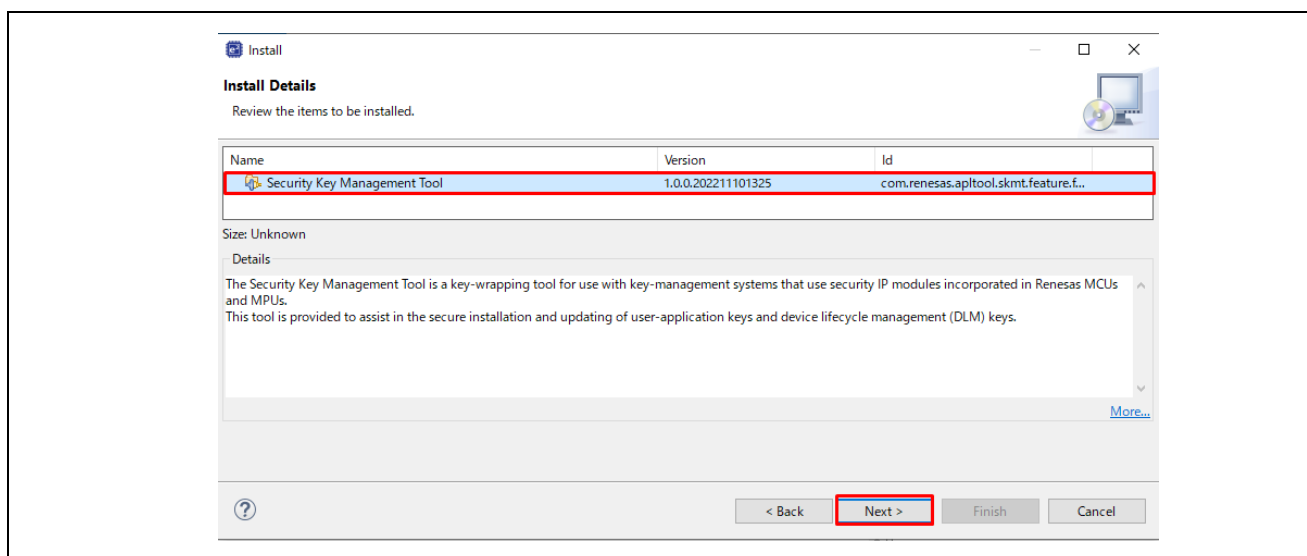


图 5-9 “安装”对话框 – 安装详细信息

8. 随后是许可协议确认屏幕。接受插件下载期间提供的许可条款，可以在对话框中显示的 URL 上查看许可协议的详细内容。选择 “I accept the terms of the license agreement”，然后按 “完成” 按钮。

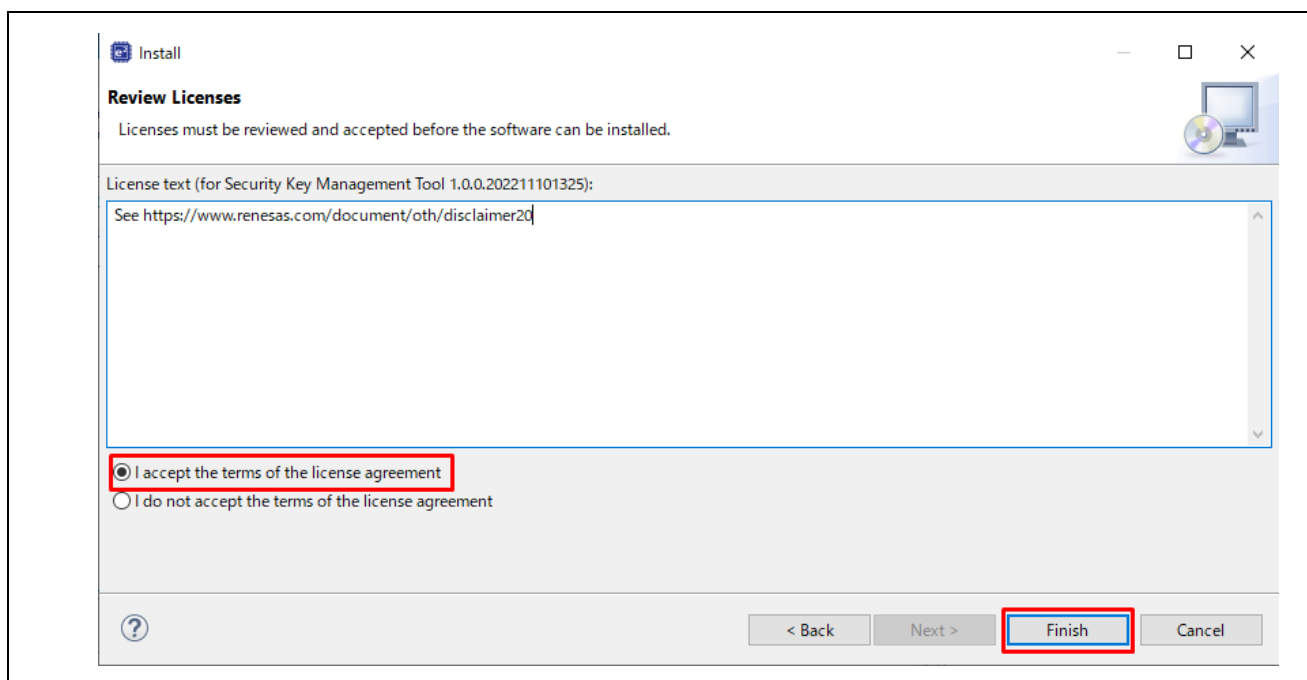


图 5-10 “安装”对话框 – 查看许可协议

9. 当出现 “信任” 对话框时，选中显示的证书，然后按 “信任选定项” 按钮继续安装。

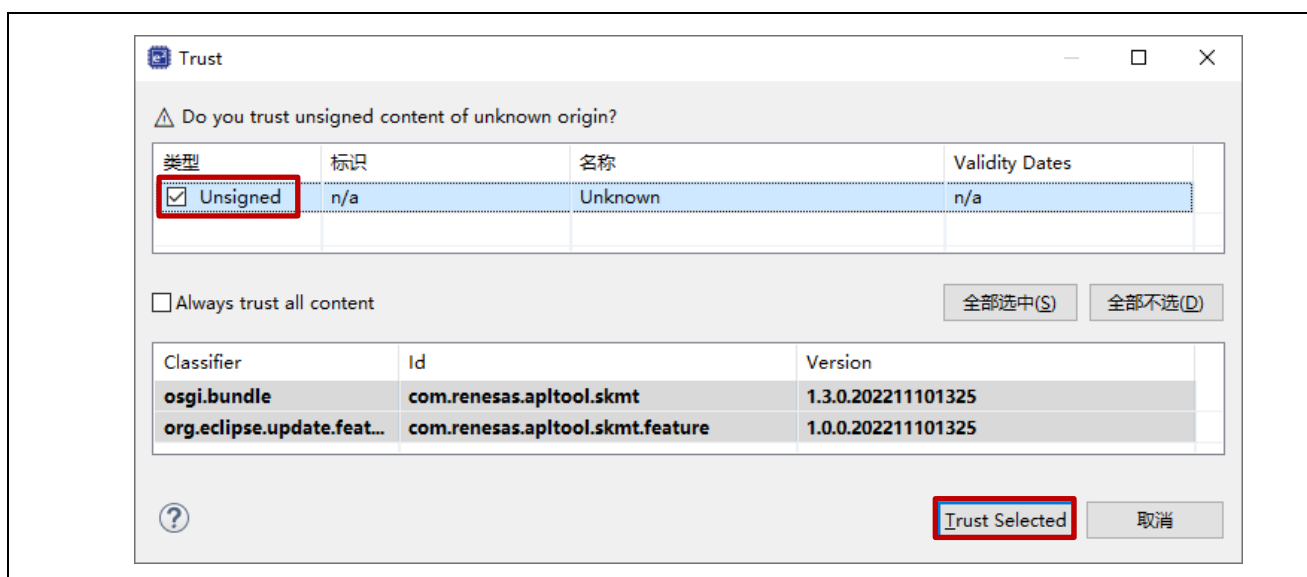


图 5-11 “信任”对话框

10. 当出现提示时，重新启动 e² studio。
11. 安装完成后，安全密钥管理工具将添加到项目的“属性”对话框。

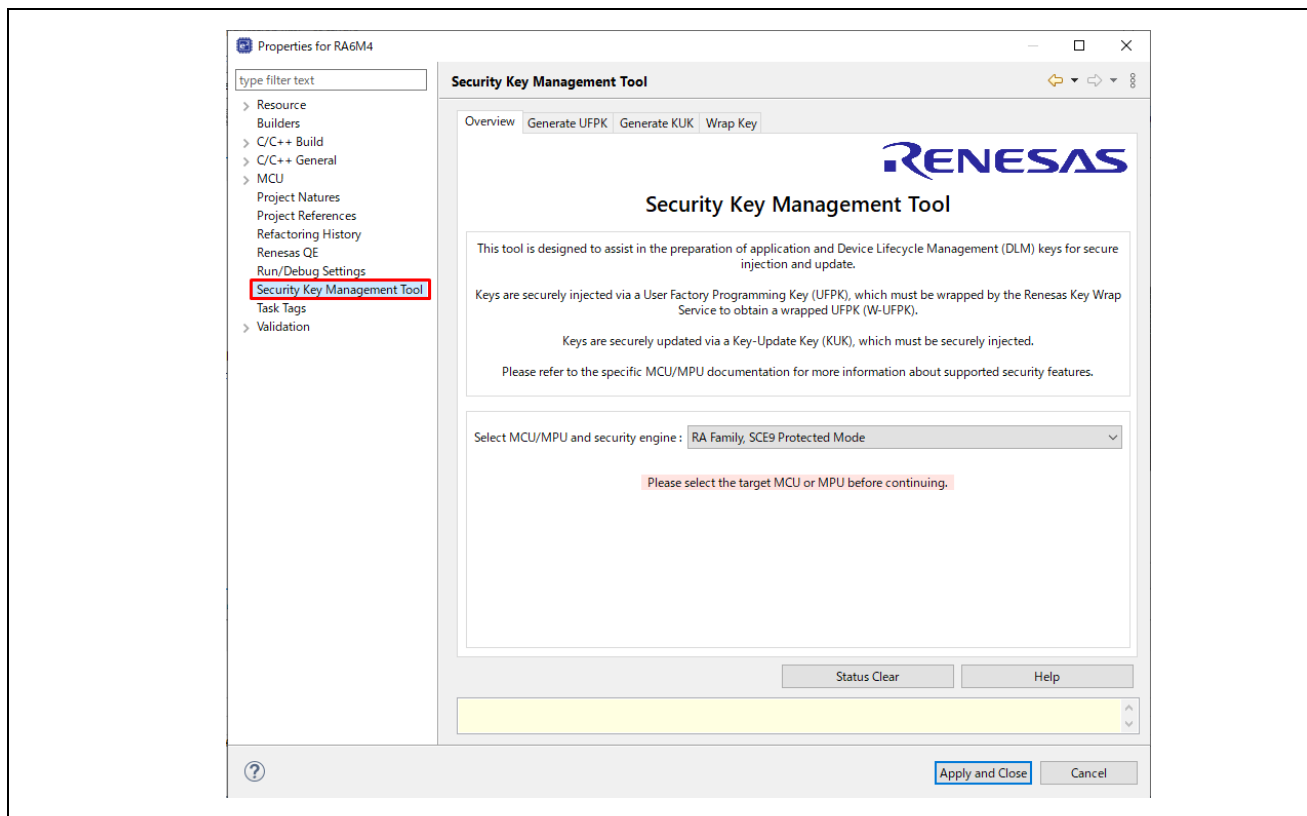


图 5-12 项目“属性”对话框

5.2.2 卸载 e²studio 插件版本

1. 选择 e² studio 菜单“帮助(H)” – “关于 e² studio” 菜单，打开“关于 e² studio”对话框。
2. 在“关于 e² studio”对话框中按“安装细节”按钮。

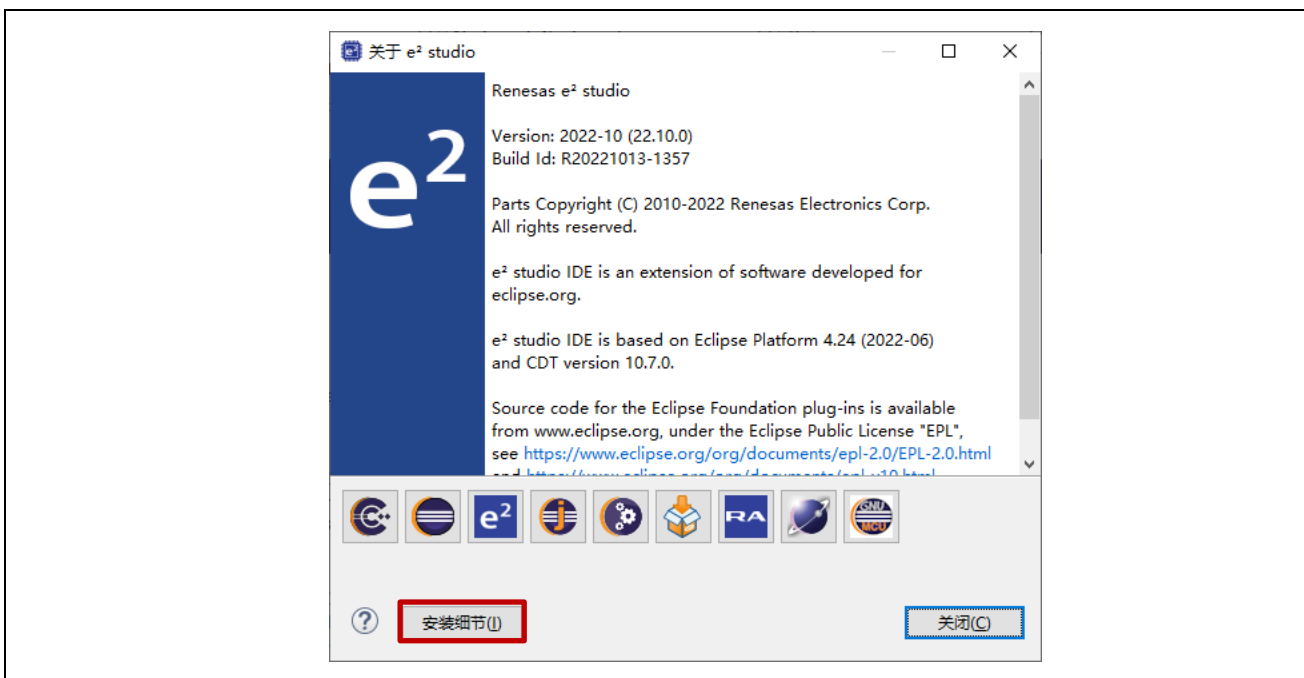


图 5-11 “关于 e²studio”对话框

3. 在“e² studio 安装细节”对话框中，选择“已安装的软件”选项卡 - 安全密钥管理工具，然后按“卸载...”按钮。

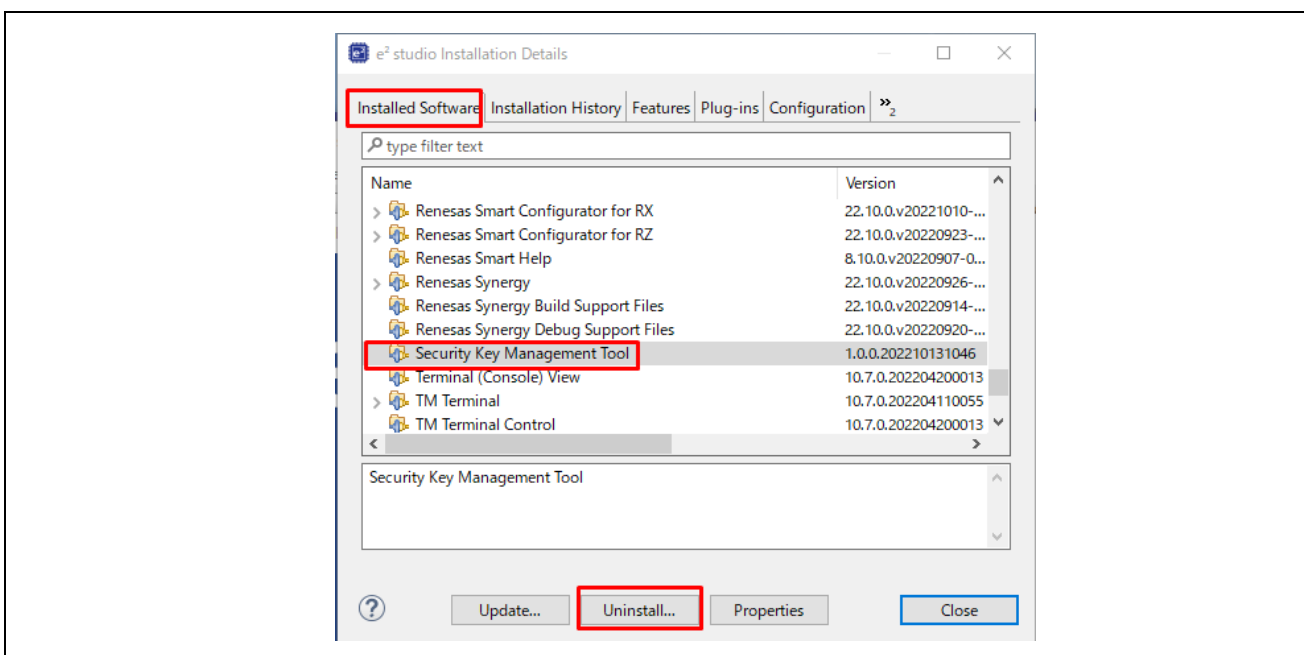


图 5-12 “e² studio 安装细节”对话框

4. 当出现“卸载”对话框时，选择安全密钥管理工具，然后按“完成”按钮。

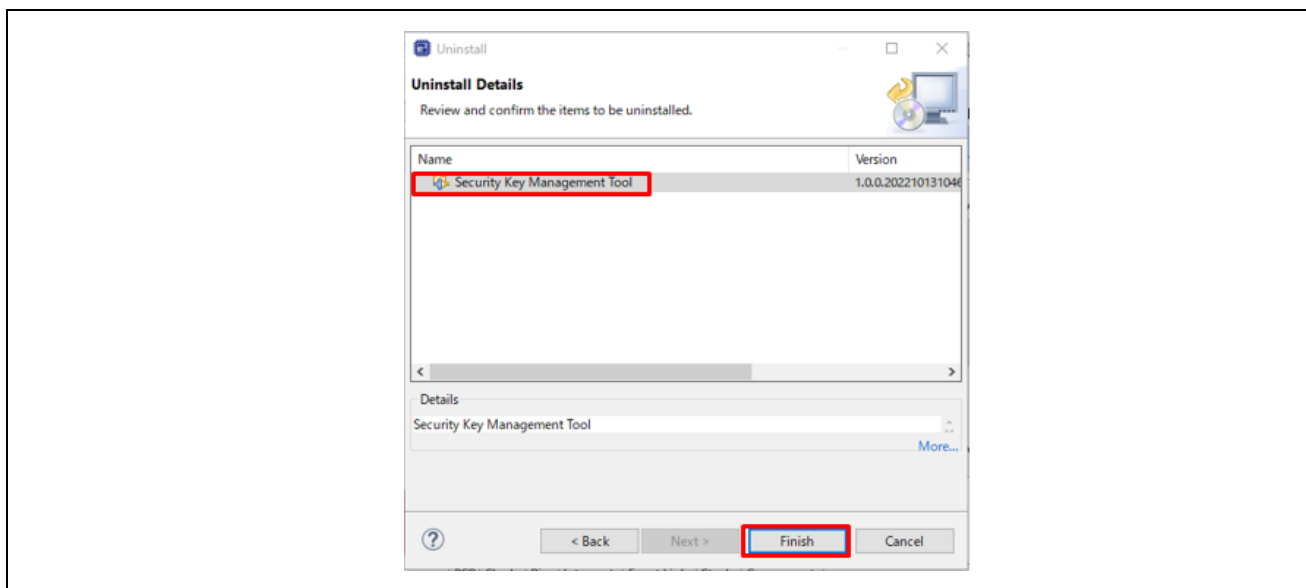


图 5-13 “卸载”对话框

5. 系统将提示重新启动 e² studio。

6. 使用示例

6.1 示例 1 – 在具有 TSIP 的 RX 产品家族 MCU 上安装 AES128 密钥

RX 产品家族 TSIP 支持通过在 MCU 上执行固件来实现安全密钥安装。根据表 1-1 MCU/MPU 相关信息，可以在 TSIP 库中找到所需的驱动程序。

在生产过程中，经常需要使用相同的密钥对器件组进行编程。密钥安装信息可以嵌入到配置代码中，但是如果存在多个具有不同密钥的组，则最好将密钥信息从配置代码中分离出来。这些步骤可用于创建 Motorola 十六进制文件，该文件可与配置代码一起编程，以便安全安装一个或多个密钥。在以下示例中，创建了一个文件来安装一个 AES128 密钥。

6.1.1 使用 GUI 版本

1. 在 [概要] 选项卡上选择 MCU/MPU 以及加密引擎。



图 6-1 [概要] 选项卡，使用 RX 产品家族 TSIP 安装 AES128 密钥

2. 创建 UFPK。在 [生成 UFPK] 选项卡中，输入所需的 UFPK 值，然后输入扩展名为 *.key 的输出文件名。在本示例中，使用文件名 ufpk.key。

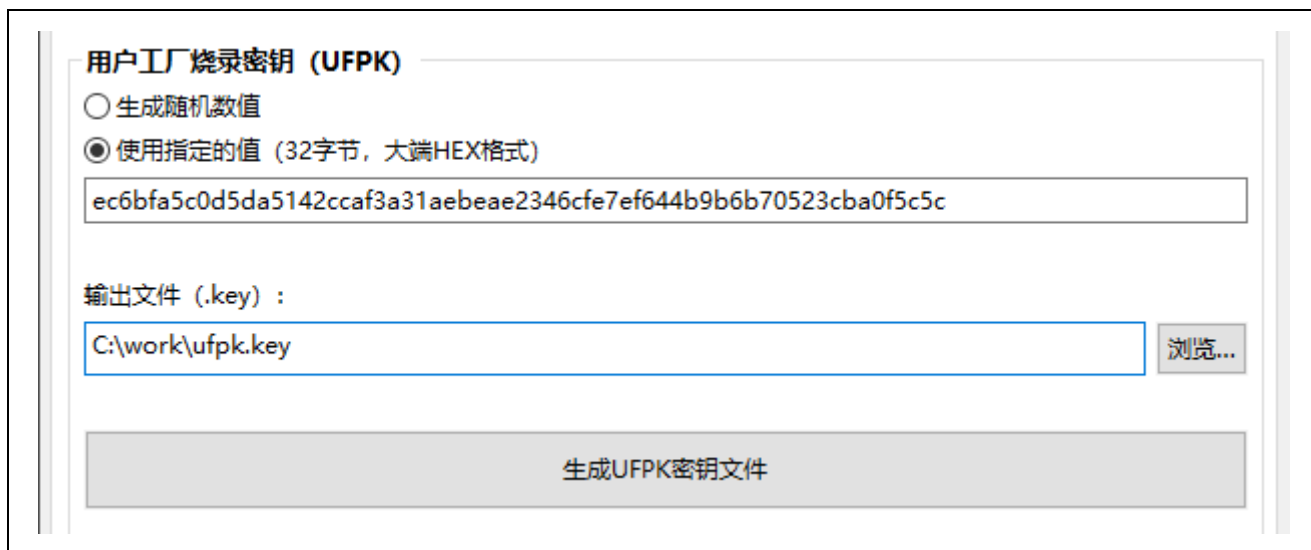


图 6-2 [生成 UFPK] 选项卡，使用指定的值生成 UFPK 的示例

按“生成 UFPK 密钥文件”按钮生成 UFPK 文件。如果文件成功生成，该工具将输出以下执行结果。

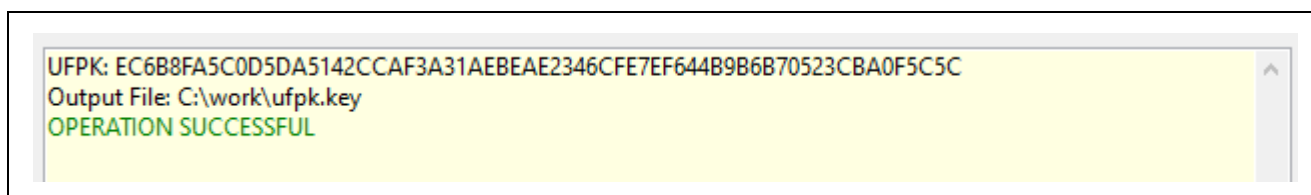


图 6-3 使用指定的值生成 UFPK 的执行结果示例

3. 获取 W-UFPK。将第 2 步中生成的 ufpk.key 文件发送到瑞萨密钥封装服务来获取 W-UFPK。有关更多信息，请参见瑞萨密钥封装服务常见问题解答或器件特定的应用笔记。瑞萨密钥封装服务的 URL 为 <https://dlm.renesas.com/keywrap>。

4. 以 Motorola 十六进制文件格式创建 AES128 密钥文件。在 [封装密钥] 选项卡中的 [密钥类型] 选项卡下，选择 **AES128**，然后在 [密钥数据文件] 选项卡中输入 AES128 数据。对于“封装密钥”，选择“UFPK”，然后选择第 2 步中生成的 UFPK 文件和第 3 步中获取的 W-UFPK 文件。为简单起见，在本示例中为 IV 选择了“生成随机数值”。在“输出文件”面板中，为格式选择“Motorola 十六进制”，输入扩展名为 *.mot 的文件名，然后将地址字段设置为 FFFF0000。

概要 生成UFPK 生成KUK 封装密钥

必须要使用UFPK (用于安全注入) 或KUK (用于安全升级) 对密钥进行封装。

密钥类型 密钥数据文件

☐ DLM DLM-SSD ☐ TDES

☐ KUK ☐ RSA 2048 bits, public

☒ AES 128 bits ☐ ECC secp256r1, public

☐ ARC4 ☐ HMAC SHA256-HMAC

封装密钥

☒ UFPK UFPK 文件: C:\work\ufpk.key 浏览...

W-UFPK 文件: C:\work\ufpk.key_enc.key 浏览...

☐ KUK KUK文件: 浏览...

初始向量IV

☒ 生成随机数值

☐ 使用指定的值 (16字节, 大端HEX格式) 00112233445566778899AABBCCDDEEFF

输出文件

格式: csource 文件: C:\work\aes128.mot 浏览...

地址: 10000 密钥名称:

生成文件

图 6-4 [封装密钥] - [密钥类型] 选项卡，以 Motorola 十六进制格式创建 AES128 密钥文件的示例

概要 生成UFPK 生成KUK 封装密钥

必须要使用UFPK（用于安全注入）或KUK（用于安全升级）对密钥进行封装。

密钥类型 密钥数据文件

☐ 文件 ☐ 明文密钥 ☐ 使用随机数- 输出文件

000102030405060708090a0b0c0d0e0f

封装密钥

☒ UFPK UFPK 文件: C:\work\ufpk.key W-UFPK 文件: C:\work\ufpk.key_enc.key ☐ KUK KUK文件:

初始向量IV

☒ 生成随机数值 ☐ 使用指定的值（16字节，大端HEX格式） 00112233445566778899AABBCCDDEEFF

输出文件

格式: csource 文件: C:\work\aes128.mot 地址: 10000 密钥名称:

生成文件

图 6-5 [封装密钥] - [密钥数据文件] 选项卡，以 Motorola 十六进制格式创建 AES128 密钥文件的示例

单击“生成文件”按钮生成指定的输出文件。如果文件成功生成，该工具将输出以下执行结果。

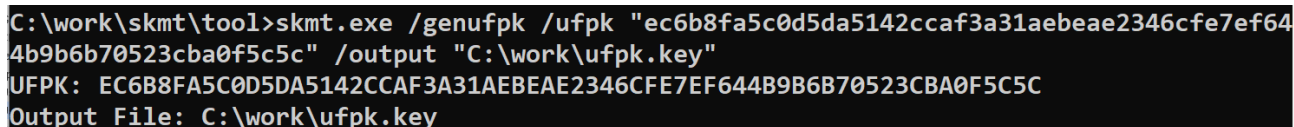
```
Output File: C:\work\aes128.mot
UFPK: EC6B8FA5C0D5DA5142CCAF3A31AEBEAE2346CFE7EF644B9B6B70523CBA0F5C5C
W-UFPK: 00000001102D464621E307E4FF7027D346B9A2DEA8E35A6673DC9685DE0283CBED82DE1E
IV: DBDD88AC0378B58AF7471FEBE17AF392
Encrypted key: 23D335F4BA2BFF6D581F0412C73E8599429BB2AEDBFBBBF18A4E374C39507AEA
OPERATION SUCCESSFUL
```

图 6-6 以 Motorola 十六进制格式创建 AES128 密钥文件的执行结果示例

6.1.2 使用 CLI 版本

1. 使用 **genufpk** 命令创建 UFPK。本示例显示了为 UFPK 使用已知值：

```
> skmt.exe /genufpk  
    /ufpk "ec6b8fa5c0d5da5142ccaf3a31aebeae2346cfe7ef644b9b6b70523cba0f5c5c"  
    /output "C:\work\ufpk.key"
```

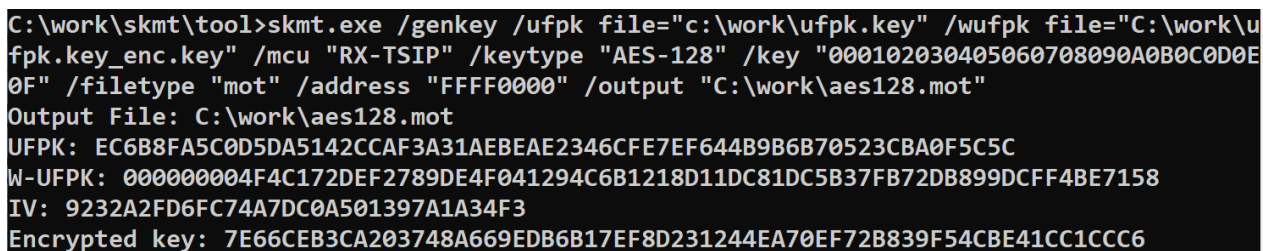


```
C:\work\skmt\tool>skmt.exe /genufpk /ufpk "ec6b8fa5c0d5da5142ccaf3a31aebeae2346cfe7ef644b9b6b70523cba0f5c5c" /output "C:\work\ufpk.key"  
UFPK: EC6B8FA5C0D5DA5142CCAF3A31AEBEAE2346CFE7EF644B9B6B70523CBA0F5C5C  
Output File: C:\work\ufpk.key
```

图 6-7 CLI genufpk 命令的执行结果

2. 获取 W-UFPK。将第 1 步中生成的 `ufpk.key` 文件发送到瑞萨密钥封装服务来获取 W-UFPK。有关更多信息，请参见瑞萨密钥封装服务常见问题解答或器件特定的应用笔记。瑞萨密钥封装服务的 URL 为 <https://dlm.renesas.com/keywrap>。
3. 使用第 1 步中生成的 UFPK 文件和第 2 步中获取的 W-UFPK 文件，通过 **genkey** 命令创建 Motorola 十六进制格式的 AES128 密钥文件：

```
> skmt.exe /genkey /ufpk file="c:\work\ufpk.key" /wufpk file="C:\work\ufpk.key_enc.key"  
    /mcu "RX-TSIP" /keytype "AES-128" /key "000102030405060708090A0B0C0D0E0F"  
    /filetype "mot" /address "FFFF0000" /output "C:\work\aes128.mot"
```



```
C:\work\skmt\tool>skmt.exe /genkey /ufpk file="c:\work\ufpk.key" /wufpk file="C:\work\ufpk.key_enc.key" /mcu "RX-TSIP" /keytype "AES-128" /key "000102030405060708090A0B0C0D0E0F" /filetype "mot" /address "FFFF0000" /output "C:\work\aes128.mot"  
Output File: C:\work\aes128.mot  
UFPK: EC6B8FA5C0D5DA5142CCAF3A31AEBEAE2346CFE7EF644B9B6B70523CBA0F5C5C  
W-UFPK: 000000004F4C172DEF2789DE4F041294C6B1218D11DC81DC5B37FB72DB899DCFF4BE7158  
IV: 9232A2FD6FC74A7DC0A501397A1A34F3  
Encrypted key: 7E66CEB3CA203748A669EDB6B17EF8D231244EA70EF72B839F54CBE41CC1CCCC6
```

图 6-8 以 Motorola 十六进制格式创建 AES128 密钥文件的 CLI 示例

6.2 示例 2 – 在具有 SCE9 保护模式的 RA 产品家族 MCU 上安装升级密钥

具有 SCE9 保护模式的 RA 产品家族 MCU 支持通过编程接口实现安全密钥安装。瑞萨闪存编程器 (RFP) 支持该功能。

使用编程接口实现安全密钥安装的优势在于，MCU 上无需特殊的配置代码。密钥和应用程序代码可以在同一编程过程中安装。在本示例中，安装一个 KUK，同时在现场启用了密钥升级。

6.2.1 使用 GUI 版本

1. 在 [概要] 选项卡上选择 MCU/MPU 以及加密引擎。



图 6-9 [概要] 选项卡，在 RA 产品家族 SCE9 保护模式下安装 KUK

2. 创建 UFPK。在 [生成 UFPK] 选项卡中，输入所需的 UFPK 值，然后输入扩展名为 *.key 的输出文件名。在本示例中，使用文件名 ufpk.key。



图 6-10 [生成 UFPK] 选项卡，使用指定的值生成 UFPK 的示例

按“生成 UFPK 密钥文件”按钮生成 UFPK 文件。如果文件成功生成，该工具将输出以下执行结果：

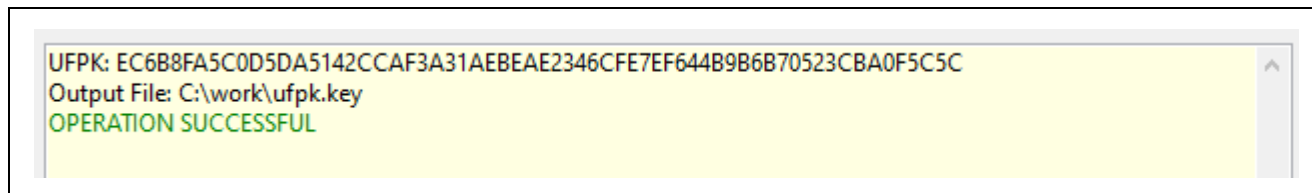


图 6-11 使用指定的值生成 UFPK 的执行结果示例

3. 获取 W-UFPK。将第 2 步中生成的 ufpk.key 文件发送到瑞萨密钥封装服务来获取 W-UFPK。有关更多信息，请参见瑞萨密钥封装服务常见问题解答或器件特定的应用笔记。瑞萨密钥封装服务的 URL 为 <https://dlm.renesas.com/keywrap>。

4. 创建 KUK 密钥文件。在 [生成 KUK] 选项卡中，选择是使用工具为 KUK 生成随机数值还是为 KUK 输入 256 位值。输入扩展名为 *.key 的输出文件名。在本示例中，使用文件名 kuk.key。



升级密钥 Key-Update Keys

☐ 生成随机数值

☒ 使用指定的值 (32字节, 大端HEX格式)

d0aec19426cbc0e2fb403866b9b465a6c0d05b7a60362d5f435f9a3e98c79084

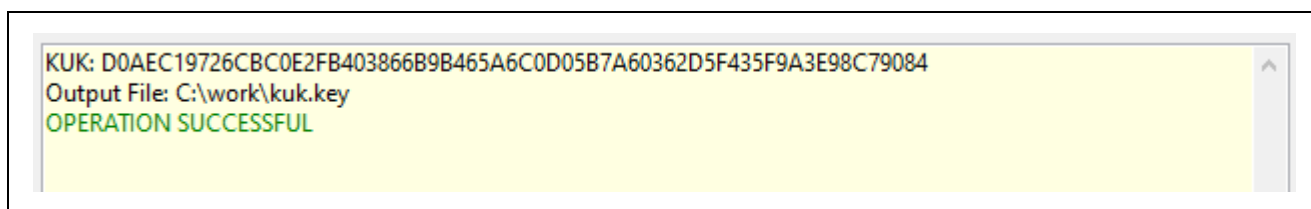
输出文件 (.key) :

C:\work\kuk.key 浏览...

生成KUK密钥文件

图 6-12 [生成 KUK] 选项卡，创建 KUK 密钥文件的示例

单击“生成 KUK 密钥文件”按钮生成 KUK 密钥文件。如果文件成功生成，该工具将输出以下执行结果。



KUK: D0AEC19726CBC0E2FB403866B9B465A6C0D05B7A60362D5F435F9A3E98C79084
Output File: C:\work\kuk.key
OPERATION SUCCESSFUL

图 6-13 创建 KUK 文件的执行结果示例

5. 为 KUK 安装创建 RFP 文件。在 [封装密钥] 选项卡中的 [密钥类型] 选项卡中，选择 **KUK**。在 [密钥数据文件] 选项卡中，输入上一步中生成的 KUK 密钥文件名（本例中为 `kuk.key`）。对于“封装密钥”，选择“**UFPK**”，然后选择第 2 步中生成的 UFPK 文件和第 3 步中获取的 W-UFPK 文件。为简单起见，在本示例中为 IV 选择了“生成随机数值”。在“输出文件”面板中，为格式选择“**RFP**”，然后输入扩展名为 `*.rkey` 的文件名。

概要 生成UFPK 生成KUK 封装密钥

必须要使用UFPK（用于安全注入）或KUK（用于安全升级）对密钥进行封装。

密钥类型 密钥数据文件

☐ DLM DLM-SSD ☐ TDES

☒ KUK ☐ RSA 2048 bits, public

☐ AES 128 bits ☐ ECC secp256r1, public

☐ ARC4 ☐ HMAC SHA256-HMAC

封装密钥

☒ UFPK UFPK 文件: C:\work\ufpk.key 浏览...

W-UFPK 文件: C:\work\ufpk.key_enc.key 浏览...

☐ KUK KUK文件: 浏览...

初始向量IV

☒ 生成随机数值

☐ 使用指定的值（16字节，大端HEX格式） 00112233445566778899AABBCCDDEEFF

输出文件

格式: rfp 文件: C:\work\kuk.rkey 浏览...

地址: 10000 密钥名称:

生成文件

图 6-14 [封装密钥] - [密钥类型] 选项卡，以 RFP 文件格式创建 KUK 文件的示例

概要 生成UFPK 生成KUK 封装密钥

必须要使用UFPK (用于安全注入) 或KUK (用于安全升级) 对密钥进行封装。

密钥类型 密钥数据文件

☒ 文件 C:\work\kuk.key 浏览...

☐ 明文密钥 000102030405060708090a0b0c0d0e0f

☐ 使用随机数- 输出文件 浏览...

封装密钥

☒ UFPK UFPK 文件: C:\work\ufpk.key 浏览...

W-UFPK 文件: C:\work\ufpk.key_enc.key 浏览...

☐ KUK KUK文件: 浏览...

初始向量IV

☒ 生成随机数值

☐ 使用指定的值 (16字节, 大端HEX格式) 00112233445566778899AABBCCDDEEFF

输出文件

格式: rfp 文件: C:\work\kuk.rkey 浏览...

地址: 10000 密钥名称:

生成文件

图 6-15 [封装密钥] 选项卡, 以 RFP 文件格式创建 KUK 文件的示例

单击“生成文件”按钮生成指定的输出文件。如果文件成功生成, 该工具将输出以下执行结果。

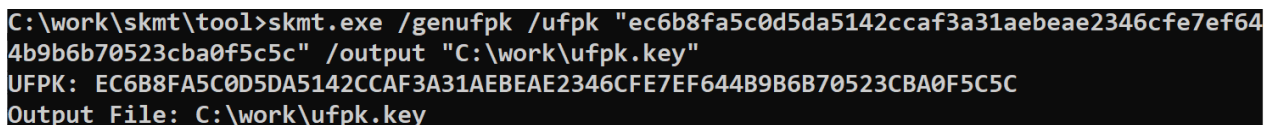
```
W-UFPK: 00000000971BF948953C5A92626AC7D70CFD8C4803B7FC978A9028CF1CD0CC32BD2F7F97
IV: A7136C9E53E737BF2B5864AEEDA43D58
Encrypted key:
9B00211834DB5492DB9CE4C8707D1C38D5032B5538CBFDEC8A0D3B65FD802F81475D14B70DE4C89DE8A8ABCF28FA
899F
OPERATION SUCCESSFUL
```

图 6-16 以 RFP 文件格式创建 KUK 文件的执行结果示例

6.2.2 使用 CLI 版本

1. 使用 **genufpk** 命令创建 UFPK。本示例显示了为 UFPK 使用指定的值：

```
> skmt.exe /genufpk  
    /ufpk "ec6b8fa5c0d5da5142ccaf3a31aebeae2346cfe7ef644b9b6b70523cba0f5c5c"  
    /output "C:\work\ufpk.key"
```

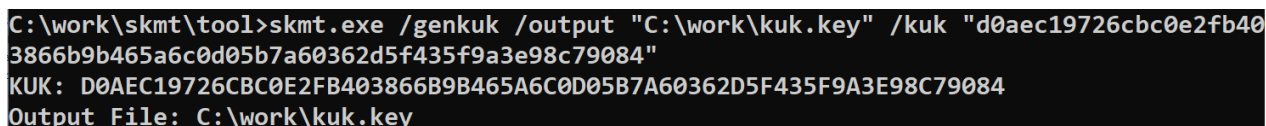


```
C:\work\skmt\tool>skmt.exe /genufpk /ufpk "ec6b8fa5c0d5da5142ccaf3a31aebeae2346cfe7ef644b9b6b70523cba0f5c5c" /output "C:\work\ufpk.key"  
UFPK: EC6B8FA5C0D5DA5142CCAF3A31AEBEAE2346CFE7EF644B9B6B70523CBA0F5C5C  
Output File: C:\work\ufpk.key
```

图 6-17 CLI genufpk 命令的执行结果

2. 获取 W-UFPK。将第 1 步中生成的 `ufpk.key` 文件发送到瑞萨密钥封装服务来获取 W-UFPK。有关更多信息，请参见瑞萨密钥封装服务常见问题解答或器件特定的应用笔记。瑞萨密钥封装服务的 URL 为 <https://dlm.renesas.com/keywrap>。
3. 使用 **genkuk** 命令创建 KUK 密钥文件。本示例显示了为 KUK 使用指定的值。

```
> skmt.exe /genkuk /output "C:\work\kuk.key"  
    /kuk "d0aec19726cbc0e2fb403866b9b465a6c0d05b7a60362d5f435f9a3e98c79084"
```



```
C:\work\skmt\tool>skmt.exe /genkuk /output "C:\work\kuk.key" /kuk "d0aec19726cbc0e2fb403866b9b465a6c0d05b7a60362d5f435f9a3e98c79084"  
KUK: D0AEC19726CBC0E2FB403866B9B465A6C0D05B7A60362D5F435F9A3E98C79084  
Output File: C:\work\kuk.key
```

图 6-18 CLI genkuk 命令的执行结果

4. 使用第 1 步中生成的 UFPK 文件、第 2 步中获取的 W-UFPK 文件和第 3 步中生成的 KUK 文件，通过 **genkey** 命令创建要用于 KUK 安装的 RFP 文件：

```
> skmt.exe /genkey /ufpk file="c:\work\ufpk.key" /wufpk file="C:\work\ufpk.key_enc.key"  
/mcu "RA-SCE9" /keytype "key-update-key" /key file="C:\work\kuk.key" /filetype "rfp"  
/output "C:\work\kuk.rkey"
```

```
C:\work\skmt\tool>skmt.exe /genkey /ufpk file="c:\work\ufpk.key" /wufpk file="C:\work\ufpk.key_enc.key" /mcu "RA-SCE9" /keytype "key-update-key" /key file="C:\work\kuk.key" /filetype "rfp" /output "C:\work\kuk.rkey"  
Output File: C:\work\kuk.rkey  
UFPK: EC6B8FA5C0D5DA5142CCAF3A31AEBAE2346CFE7EF644B9B6B70523CBA0F5C5C  
W-UFPK: 000000004F4C172DEF2789DE4F041294C6B1218D11DC81DC5B37FB72DB899DCFF4BE7158  
IV: 4B44A010D7BDF764200BC2424650F976  
Encrypted key: 043565AE00E185EBAFF505290DF1976B1A00132FE1F0B25F32C3C83BF92C83A7644968FF5869821C3EBAD7EB41AA509B
```

图 6-19 以 RFP 文件格式创建 AES128 密钥文件的 CLI 示例

6.3 示例 3 – 在具有 SCE9 保护模式的 RA 产品家族 MCU 上升级 RSA 2048 公钥

可以使用之前安装的 KUK 在现场升级密钥。根据表 1-1 *MCU/MPU 相关信息*，可以在器件的支持软件驱动程序中找到所需的驱动程序。

可以通过多种方式在现场执行密钥升级。一种方式是将使用 KUK 封装的密钥作为固件升级的一部分包括在内。有一种简单的方式是将数据作为 C 源文件嵌入。在本示例中，使用之前安装的 KUK（例如，在前一个示例中创建和安装的 KUK）升级一个 RSA 2048 公钥。

6.3.1 使用 GUI 版本

1. 在 [概要] 选项卡上选择 MCU/MPU 以及加密引擎。



图 6-20 [概要] 选项卡，在具有 SCE9 保护模式的 RA 产品家族上升级 RSA 2048 公钥

2. 以 C 源文件格式创建 RSA 2048 公钥文件。在 [封装密钥] 选项卡中的 [密钥类型] 选项卡下，选择 **RSA** 和 “**2048 bits, public**”，然后在 [密钥数据文件] 选项卡中输入 RSA 2048 公钥数据。对于 “封装密钥”，选择 **KUK**，然后将 “**KUK 文件**” 设置为第 6.2 节示例 2 – 在具有 SCE9 保护模式的 RA 产品家族 MCU 上安装升级密钥中生成的文件。为简单起见，在本示例中为 IV 选择了 “生成随机数值”。在 “输出文件” 面板中，为格式选择 “**C 源文件**”，然后输入扩展名为 *.c 的文件名。

必须要使用UFPK (用于安全注入) 或KUK (用于安全升级) 对密钥进行封装。

密钥类型 密钥数据文件

☐ DLM DLM-SSD ☐ TDES

☐ KUK ☒ RSA 2048 bits, public

☐ AES 128 bits ☐ ECC secp256r1, public

☐ ARC4 ☐ HMAC SHA256-HMAC

封装密钥

☐ UFPK UFPK 文件: 浏览...

W-UFPK 文件: 浏览...

☒ KUK KUK文件: C:\work\kuk.key 浏览...

初始向量IV

☒ 生成随机数值

☐ 使用指定的值 (16字节, 大端HEX格式) 00112233445566778899AABBCCDDEEFF

输出文件

格式: csource 文件: C:\work\rsa2048public.c 浏览...

地址: 10000 密钥名称: rsa2048public

生成文件

图 6-21 [封装密钥] - [密钥类型] 选项卡，以 C 源文件格式创建 RSA 2048 公钥文件的示例

概要

生成UFPK

生成KUK

封装密钥

必须要使用UFPK（用于安全注入）或KUK（用于安全升级）对密钥进行封装。

密钥类型

密钥数据文件

☐ 文件

☒ 明文密钥

模数(n) :

1b4a3d0581e74bf9ade96cc46146817553931a79d92e9e488ef47223ee6f6c061884b13c9065b591139de13c1ea2927491ed00fb793cd68f463f5f64baa53916b46c818ab99706557a1c2d50d232577d1

指数(e) :

10001

封装密钥

☐ UFPK

UFPK 文件:

浏览...

W-UFPK 文件:

浏览...

☒ KUK

KUK文件:

C:\work\kuk.key

浏览...

初始向量IV

☒ 生成随机数值

☐ 使用指定的值（16字节，大端HEX格式）

00112233445566778899AABBCCDDEEFF

输出文件

格式:

csource

文件:

C:\work\rsa2048public.c

浏览...

地址:

10000

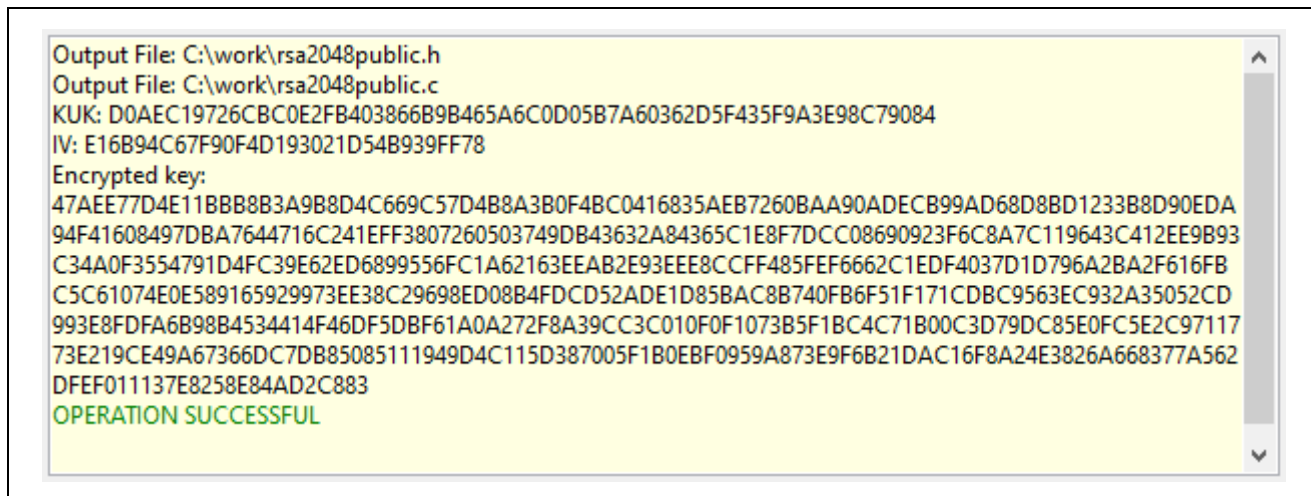
密钥名称:

rsa2048public

生成文件

图 6-22 [封装密钥] – [密钥数据文件] 选项卡，以 C 源文件格式创建 RSA 2048 公钥文件的示例

单击“生成文件”按钮生成指定的输出文件。如果文件成功生成，该工具将输出以下执行结果。



```
Output File: C:\work\rsa2048public.h
Output File: C:\work\rsa2048public.c
KUK: D0AEC19726CBC0E2FB403866B9B465A6C0D05B7A60362D5F435F9A3E98C79084
IV: E16B94C67F90F4D193021D54B939FF78
Encrypted key:
47AEE77D4E11BBB8B3A9B8D4C669C57D488A3B0F4BC0416835AEB7260BAA90ADECB99AD68D8BD1233B8D90EDA
94F41608497DBA7644716C241EFF3807260503749DB43632A84365C1E8F7DCC08690923F6C8A7C119643C412EE9B93
C34A0F3554791D4FC39E62ED6899556FC1A62163EEAB2E93EEE8CCFF485FEF6662C1EDF4037D1D796A2BA2F616FB
C5C61074E0E589165929973EE38C29698ED08B4FDCD52ADE1D85BAC8B740FB6F51F171CDBC9563EC932A35052CD
993E8FDA6B98B4534414F46DF5DBF61A0A272F8A39CC3C010F0F1073B5F1BC4C71B00C3D79DC85E0FC5E2C97117
73E219CE49A67366DC7DB85085111949D4C115D387005F1B0EBF0959A873E9F6B21DAC16F8A24E3826A668377A562
DFEF011137E8258E84AD2C883
OPERATION SUCCESSFUL
```

图 6-23 以 C 源文件格式创建 RSA 2048 公钥文件的执行结果示例

6.3.2 使用 CLI 版本

1. 使用 **genkey** 命令以 C 源文件格式创建 RSA 2048 公钥文件：

```
> skmt.exe /genkey /kuk file="C:\work\kuk.key" /mcu "RA-SCE9" /keytype "RSA-2048-public"
/key "bad47a84c1782e4dbdd913f2a261fc8b65838412c6e45a2068ed6d7f16e9cdf4
462b39119563cafb74b9cbf25cfd544bdae23bff0ebe7f6441042b7e109b9a8a
faa056821ef8efaab219d21d6763484785622d918d395a2a31f2ece8385a8131
e5ff143314a82e21afd713bae817cc0ee3514d4839007ccb55d68409c97a18ab
62fa6f9f89b3f94a2777c47d6136775a56a9a0127f682470bef831fbec4bcd7b
5095a7823fd70745d37d1bf72b63c4b1b4a3d0581e74bf9ade93cc4614861755
3931a79d92e9e488ef47223ee6f6c061884b13c9065b591139de13c1ea292749
1ed00fb793cd68f463f5f64baa53916b46c818ab99706557a1c2d50d232577d1
00010001"
/filetype "csource" /output "C:\work\rsa2048public.c" /keyname "rsa2048public"
```

使用第 6.2 节示例 2 – 在具有 SCE9 保护模式的 RA 产品家族 MCU 上安装升级密钥中生成的 KUK 文件。

```
C:\work\skmt\tool>skmt.exe /genkey /kuk file="C:\work\kuk.key" /mcu "RA-SCE9" /keytype
"RSA-2048-public" /key "bad47a84c1782e4dbdd913f2a261fc8b65838412c6e45a2068ed6d7f16e9cdf
4462b39119563cafb74b9cbf25cfd544bdae23bff0ebe7f6441042b7e109b9a8afaa056821ef8efaab219d2
1d6763484785622d918d395a2a31f2ece8385a8131e5ff143314a82e21afd713bae817cc0ee3514d4839007
ccb55d68409c97a18ab62fa6f9f89b3f94a2777c47d6136775a56a9a0127f682470bef831fbec4bcd7b5095
a7823fd70745d37d1bf72b63c4b1b4a3d0581e74bf9ade93cc46148617553931a79d92e9e488ef47223ee6f
6c061884b13c9065b591139de13c1ea2927491ed00fb793cd68f463f5f64baa53916b46c818ab99706557a1
c2d50d232577d100010001" /filetype "csource" /output "C:\work\rsa2048public.c" /keyname
"rsa2048public"
Output File: C:\work\rsa2048public.h
Output File: C:\work\rsa2048public.c
KUK: D0AEC19726CBC0E2FB403866B9B465A6C0D05B7A60362D5F435F9A3E98C79084
IV: BED48489C13FF9173A6F7649DEB6EB47
Encrypted key: FBB110AAAA2260E6033B2D9839A71121C137ABE34D1FE59D50C7DB2F0B568AA2D5C0A0CF
92CA7D093A624E931BFC6115A09DBED02CA3E85909940D2FE2921C5589D4D858A349F54FFFEBF8A18D94139
909BF57A0DEB219A99C640993990B3B837132F404CCCCB821AF1D28C8895C968863E293E22A87365BEC0748
1B856275BEC8E44EC9BFA392F586CEBB674C73230834C0B7989011E8DCEA6C71DE314D381788A129A42C541
27CA2FBD315A3F8BA9960B25C7B6A999D915443358C755A52D77608CFE4A48B9EEB46CFDA0AB96CEC209EB2
01F2EA2C2382564C30A3622E57071247B2E386160C5D21A00119ED4DD118B07554E72BB844FFA2AE9EF7C3D
28D9D0C116490388A554EC39D7D515C89C8459A42FD4495B6DF14ADB69D082D356DED
```

图 6-24 以 C 源文件格式创建 RSA 2048 公钥文件的 CLI 示例

7. 注意事项

7.1 使用 Windows 环境时的显示设置

在 Windows 环境中，如果独立 GUI 或 e² studio 插件版本设置为建议设置以外的显示设置，则可能无法显示整个对话框。

可通过以下方式改进显示。

1. 右键单击可执行文件：
独立版本：SecurityKeyManagementTool.exe
e² studio 插件版本：e2studio.exe
e2studio.exe 位于 e² studio 安装的 **eclipse** 文件夹中。
2. 选择**属性**，然后选择**兼容性**选项卡。在**兼容性**选项卡上，单击**更改高 DPI 设置**按钮。
3. 在**高 DPI 缩放覆盖**部分，选中**覆盖高 DPI 缩放行为**复选框，并从下拉列表中选择**系统**。然后单击两个对话框中的**确定**。

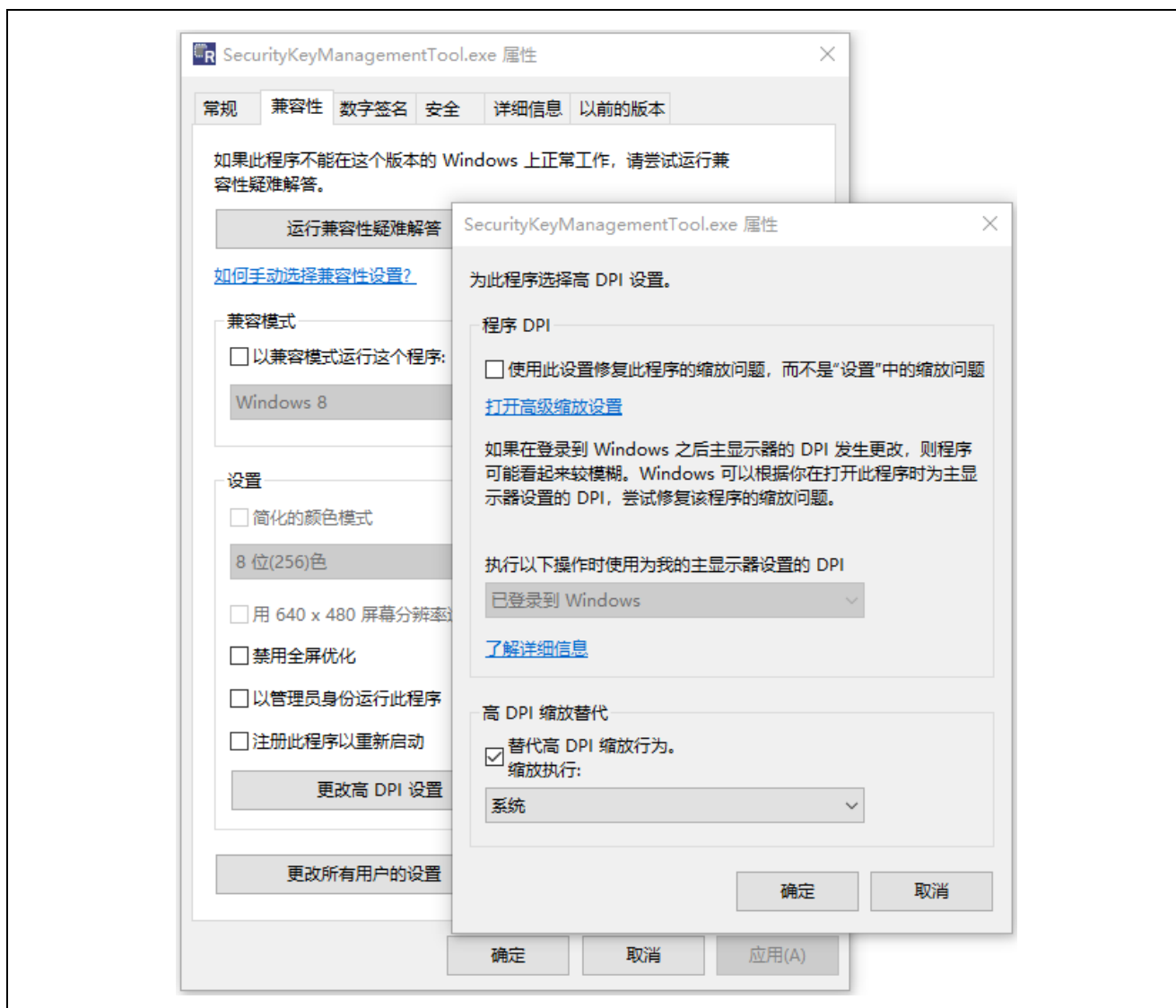


图 7-1 SecurityKeyManagementTool.exe 属性“高 DPI 缩放覆盖”设置

7.2 在 Linux 环境下使用 e²studio 插件版本的注意事项

如果您在 Linux 环境下使用 e² studio 插件版本设置为建议设置以外的显示设置，则可能无法显示整个对话框。安装后必须对命令行可执行文件设置执行权限。在 e²studio 中安装插件版 Security Key Management Tool 后，为 e²studio 安装文件夹中的以下文件赋予执行权限。

```
/eclipse/plugins/com.renesas.apltool.skmt_X.X.X.XXXXXXXXXXXXXX/cli/linux/skmt
```

附录

A. .NET

该工具使用 .NET Foundation 提供的开源软件 .NET。
.NET 许可证见下文。

.NET 许可证：

<https://github.com/dotnet/runtime/blob/main/LICENSE.TXT>

.NET Foundation：

<https://dotnetfoundation.org/>

B. Inno Setup

该工具的 windows 版本安装程序是使用 Inno Setup 创建的。
Inno Setup 许可证见下文。

Inno Setup 许可证：

<https://jrsoftware.org/files/is/license.txt>

Inno Setup：

<https://jrsoftware.org/isinfo.php>

C. 瑞萨密钥文件（Key File）的格式

1) 文本格式

项目	定义
字符	仅限 ASCII 字符 (不能使用 Unicode 和多字节字符)
空格	TAB(0x09) / Space(0x20)
每行最大长度	80 个字节
换行	CRLF(0x0D,0x0A) / CR(0x0D) / LF(0x0A)
Base64 编码	Chars = 英文字母 / 数字 / "+" / "/" Pad = "=" 行长度= 64 chars (最末行除外)

2) 文件结构

密钥文件由一下三部分组成。

```
<Header>
<Base64Lines>
<Footer>
```

<Header> 和 <Footer> 是以下列出的固定字符串。

Header = "-----BEGIN RENESAS KEY-----"

Footer = "-----END RENESAS KEY-----"

<Base64Lines> 由多行字符组成，代表密钥数据结构，具体定义如下：

密钥数据结构是 Base64 编码的二进制数据。

Footer 后面的行是空行，每一行数据后面的空格都将被忽略。

3) 文件扩展名

".rkey"

4) 密钥数据

a) 结构

密钥数据按照以下格式和大小存储，大端字符顺序。

名称	类型	大小	说明
标识符	Char[4]	4 字节	固定的四个字符"REK1"
套件版本	Integer	4 字节	数据格式版本. 当前必须设定为 1。
保留区域	Byte[7]	7 字节	Reserved. Must be 0。
密钥类型	Byte	1 字节	[DLM 密钥] 必须设为 0。 [用户密钥] Keytype 值。 (参考 4.5.2 keytype)
加密的密钥长度	Integer	4 字节	"Encrypted Key"的长度 (N 字节)
W-UFPK	Byte[36]	36 字节	从瑞萨 DLM 服务器获得的 W-UFPK 文件中的值 最前面 4 个字节是 Shared Key Number 剩余 32 个字节是 WUFPK 的值。
初始向量	Byte[16]	16 字节	I 封装用户密钥使用的初始向量 IV。
加密的密钥	Byte[N]	N 字节	通过 UFPK 加密后的用户密钥数据+MAC 值
数据 CRC	Byte[4]	4 字节	除了这四个字节以外的其他所有数据的 CRC 校验值。 Initial Value = 0xFFFFFFFF Magic number = 0x04C11DB7

版本更新历史		安全密钥管理工具V.1.03用户手册	
版本	日期	描述	
		页码	概要
1.00	2022 年 12 月 29 日	—	初版发行

安全密钥管理工具 V.1.03 用户手册

出版日期： 版本 1.00 2022 年 12 月 29 日

出版方： Renesas Electronics Corporation

安全密钥管理工具 V.1.03