Rockchip Anti Copy Board 开发指南

文件标识：RK-KF-YF-877

发布版本：V1.0.0

日期：2023-12-01

文件密级：□绝密 □秘密 □内部资料 ■公开

瑞芯微电子股份有限公司

Rockchip Electronics Co., Ltd.

地址： 福建省福州市铜盘路软件园A区18号

网址： www.rock-chips.com

客户服务电话： +86-4007-700-590

客户服务传真： +86-591-83951833

客户服务邮箱： fae@rock-chips.com

前言

概述

本文档主要介绍 Rockchip 防抄板技术的原理及使用步骤。

读者对象

本文档（本指南）主要适用于以下工程师：

技术支持工程师

软件开发工程师

修订记录

版本号	作者	修改日期	修改说明
V1.0.0	林平	2023-12-01	初始版本

目录

1. 概述

Rockchip 芯片提供防抄板技术，保护客户的固件、私有数据、核心代码。

防抄板技术主要用于防止客户的固件和私有数据被非授权用户非法拷贝并使用，避免被抄板造成商业上的损失。

本文档介绍防抄板技术的三种方案：

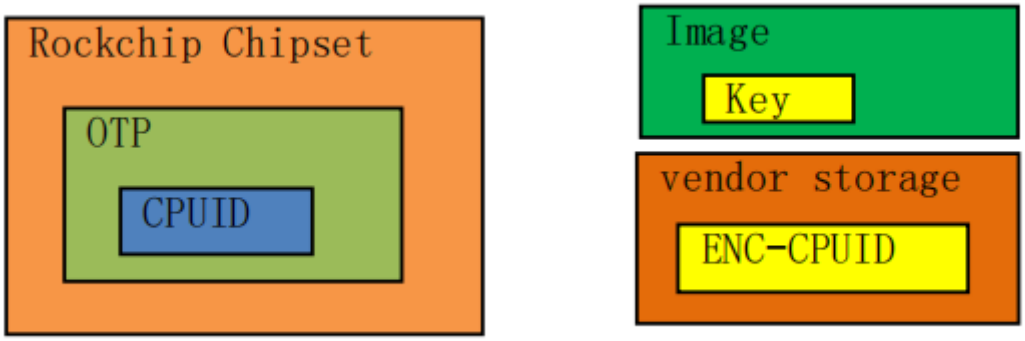
- 初级方案
- 中级方案
- 高级方案

客户可自由选择使用其中一种方案。

2. 初级方案

2.1 原理

Rockchip 芯片在生产时会烧写 CPUID 到 OTP，每颗芯片都拥有唯一的 CPUID，客户可以读取 CPUID，使用对称密钥 Key 加密得到 ENC-CPUID，客户可以将 ENC-CPUID 存储到 vendor storage 分区，系统启动后从 vendor storage 分区读取 ENC-CPUID 并解密得到 DEC-CPUID，比较 DEC-CPUID 和 CPUID 是否匹配，如果匹配失败则认为非法，重新启动设备，从而达到防抄板目的。



本方案安全性取决于对密钥 Key 的保护，客户应该避免明文密钥直接固化在代码中，建议混淆密钥后再固化到代码中，防止非法用户反汇编获取到明文密钥。

另外，ENC-CPUID 存储于 vendor storage 分区，擦除 flash 会清空 vendor storage 分区导致 ENC-CPUID 丢失，所以擦除 flash 后需要重新烧写 ENC-CPUID。

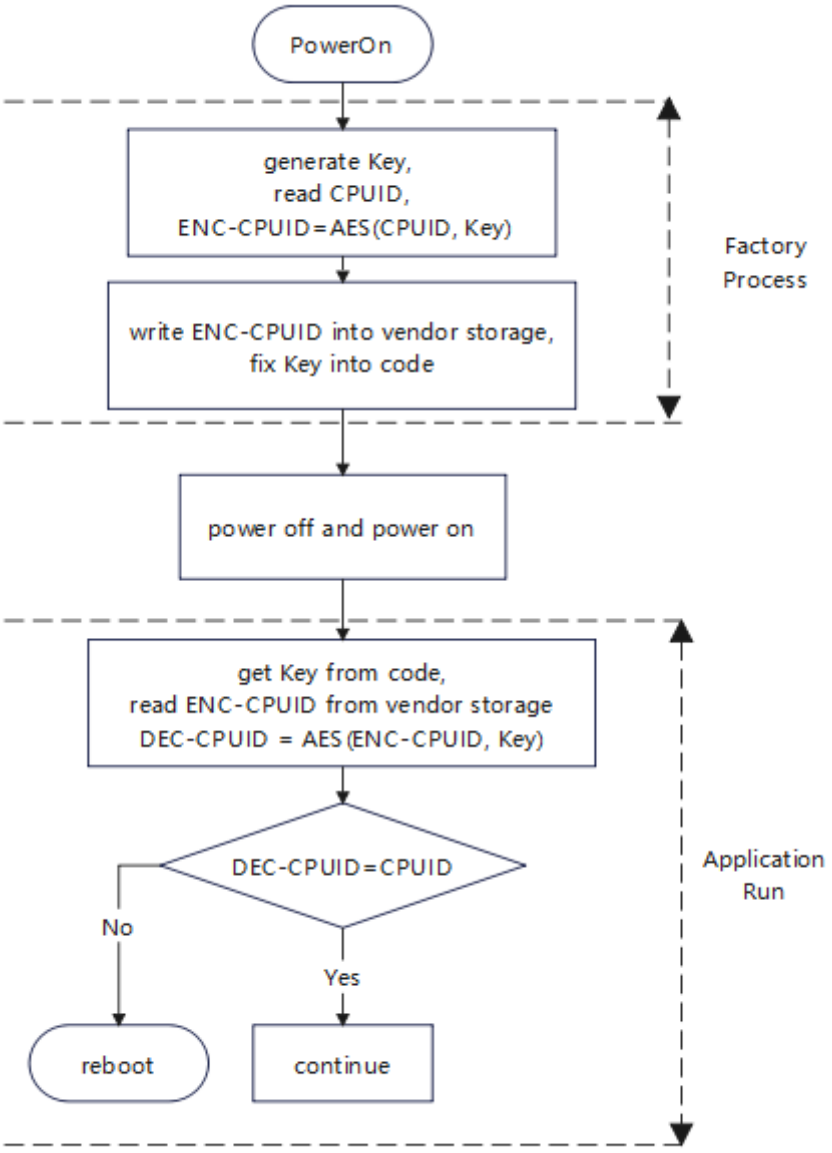
2.2 应用步骤

初级方案的应用步骤如下：

1. 客户生成自定义私有对称密钥 Key，从 OTP 读取 CPUID，使用对称密钥 Key （AES 算法）加密得到 ENC-CPUID。
2. 写入 ENC-CPUID 到 vendor storage 分区，混淆密钥增加安全性，将混淆密钥 Key 固化在应用代码中。
3. APP 运行起来后从 vendor storage 分区读取得到 ENC-CPUID，使用密钥 Key 解密 ENC-CPUID 得到 DEC-CPUID，读取 CPUID。
4. 比较 DEC-CPUID 和 CPUID 是否匹配，APP 根据匹配结果选择正常运行或者立即重启。

2.3 完整流程图

初级方案的完整流程图如下所示：



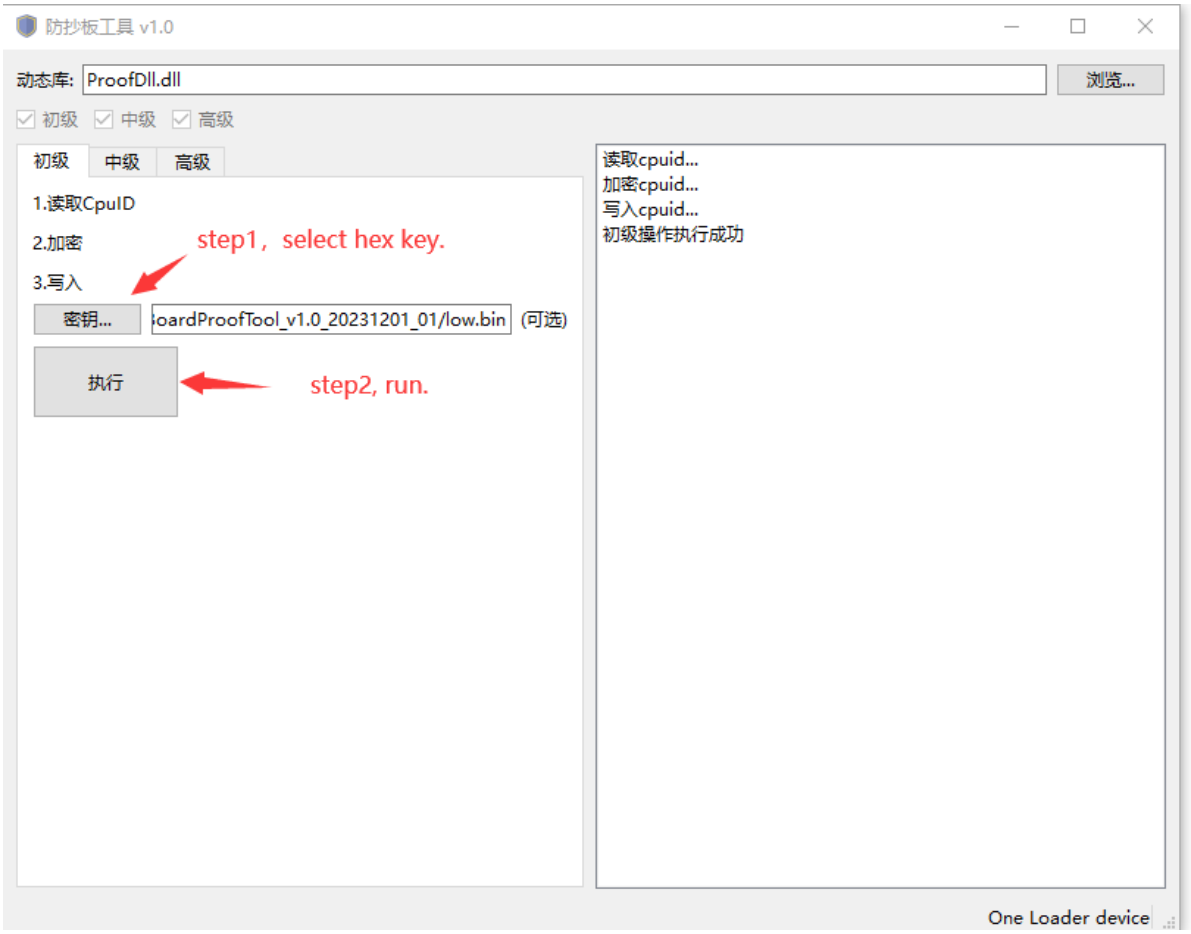
2.4 参考案例

1. 使用 BoardProofTool 工具写入 ENC-CPUID。

设备开机时长按 recovery 键进入Loader状态。

客户先创建 low.bin 文件，以16进制格式编辑该文件，写入32字节密钥。

打开 BoardProofTool.exe 工具，点击 “密钥” 选择 low.bin 文件，点击 “执行” 即可。



BoardProofTool 工具底层调用 U-Boot 中 rkusb_do_read_otp 函数读取到 CPUID，工具自动使用密钥加密 CPUID 得到 ENC-CPUID，

工具调用 U-Boot 中 vendor_storage_write 函数写入 ENC-CPUID。

2. UserSpace 层使用 rk_anti_copy_board 写入 ENC-CPUID。

rk_anti_copy_board 源码位于 rk_tee_user/v2/host/rk_anti_copy_board 目录下。

安卓编译命令：

```
cd /home1/xxxx/rk_android_13
source build/envsetup.sh
lunch rk3568_t-userdebug
cd /home1/xxxx/rk_android_13/external/rk_tee_user/v2
./build.sh ta
mm

//copy to device
adb push
Y:\rk_android_13\out\target\product\rk3568_t\vendor\bin\rk_anti_copy_board
/vendor/bin
```

Linux编译命令：

```
cd /home1/xxxx/rk_px30_linux/external/security/rk_tee_user/v2
rm -rf out/
./build.sh 6432

//copy to device
adb push
Y:\rk_px30_linux\external\security\rk_tee_user\v2\out\rk_anti_copy_board\rk_anti_
copy_board /usr/bin
```

测试命令：

```
root@RK3588:/# rk_anti_copy_board low_gen
generate enc_cpuid success!
```

3. UserSpace 层使用 rk_anti_copy_board 验证设备合法性。

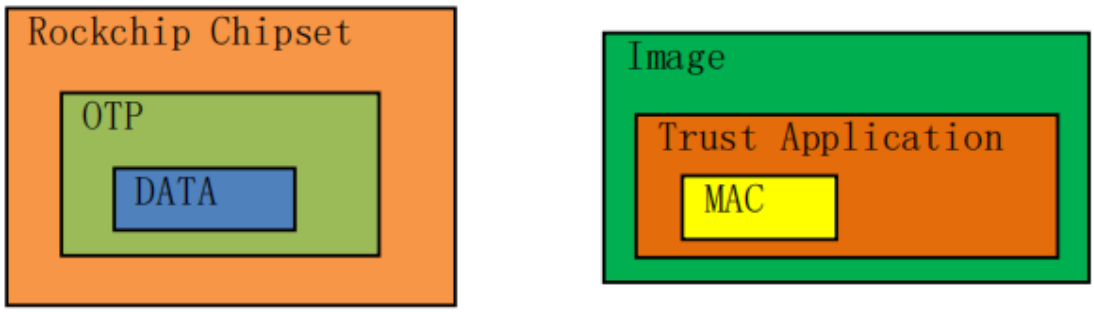
测试命令：

```
root@RK3588:/# rk_anti_copy_board low_verify
verify success!
```

3. 中级方案

3.1 原理

Rockchip 芯片在 OTP 中有预留空间供客户存储防抄板私有数据，客户生成自定义私有数据 DATA 和 MAC，客户自定义特定函数func，其中 DATA 和 MAC 满足特定函数转换关系，比如 MAC = func(DATA)，客户可以在 OTP 区域写入防抄板私有数据 DATA，将 MAC 固化在可信应用 TA(Trust Application) 代码中。系统启动后调用可信应用 TA 从 OTP 中读取到防抄板私有数据 DATA，可信应用 TA 判断 DATA 和 MAC 间是否存在给定的函数转换关系，如果匹配失败则认为非法，重新启动设备，从而达到防抄板目的。



本方案安全性取决于对防抄板私有数据 DATA 的保护，内核端无法读取该数据，只有可信应用 TA(Trust Application) 可以读取到该数据，客户应该参考《Rockchip_Developer_Guide_TEE_SDK_CN.md》文档中“TA签名” 章节，使用自己的密钥对可信应用 TA(Trust Application) 进行签名，避免非法 TA 运行而造成防抄板私有数据 DATA 泄露。

3.2 应用步骤

中级方案的应用步骤如下：

1. 客户生成自定义私有数据 DATA，选择特定转换函数 func，计算得到 MAC = func(DATA)。
2. 写入防抄板私有数据 DATA 到 OTP 中，开发可信应用 TA，将 MAC 固化在 TA 代码中。
3. APP 运行起来后调用可信应用 TA，可信应用 TA 从 OTP 读取到防抄板私有数据 DATA，可信应用 TA 计算 DATA 和 MAC 是否匹配并返回匹配结果。
4. APP 根据匹配结果选择正常运行或者立即重启。

3.3 完整流程图

中级方案的完整流程图如下所示：

3.4 参考案例

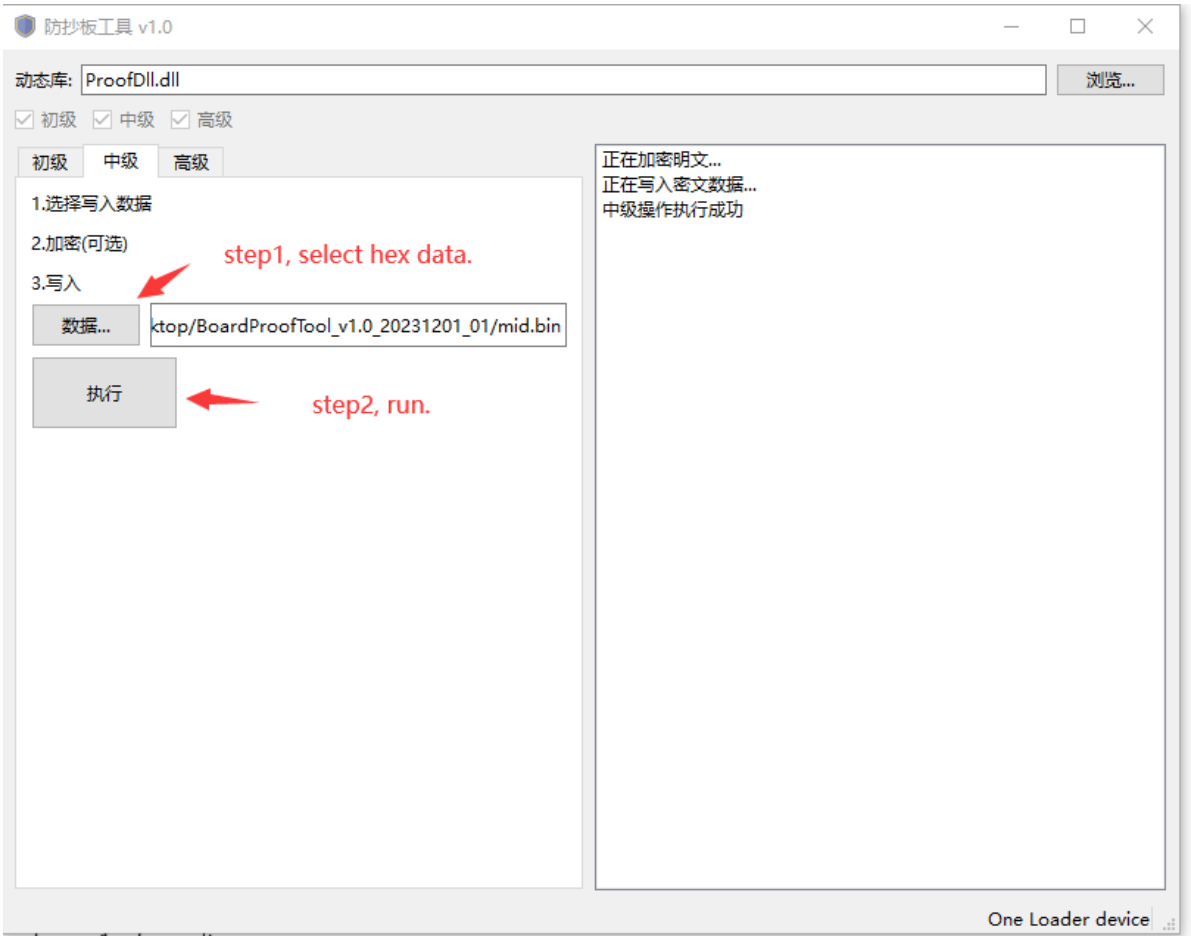
1. 使用 BoardProofTool 工具写入防抄板私有数据 DATA 到 OTP 中。

设备开机时长按 recovery 键进入Loader状态。

客户先创建 mid.bin 文件，以16进制格式编辑该文件，写入16字节防抄板私有数据。

打开 BoardProofTool.exe 工具，点击 “数据” 选择 mid.bin 文件，点击 “执行” 即可。

（注意：OTP是一次性可编程存储，每台设备只能写入一次防抄板私有数据）



2. UserSpace 层使用 rk_anti_copy_board 写入防抄板私有数据 DATA 到 OTP 中。

rk_anti_copy_board 源码位于 rk_tee_user/v2/host/rk_anti_copy_board 目录下。

安卓编译命令：

```
cd /home1/xxxx/rk_android_13
source build/envsetup.sh
lunch rk3568_t-userdebug
cd /home1/xxxx/rk_android_13/external/rk_tee_user/v2
./build.sh ta
mm

//copy to device
adb push Y:\rk_android_13\hardware\rockchip\optee\v2\arm64\libteec.so
/vendor/lib64
adb push Y:\rk_android_13\hardware\rockchip\optee\v2\arm64\tee-supplicant
/vendor/bin
adb push
Y:\rk_android_13\external\rk_tee_user\v2\out\ta\rk_anti_copy_board\3d4fc699-2065-
7bb9-33c7-b6529b43c91a.ta /vendor/lib/optee_armtz
adb push
Y:\rk_android_13\out\target\product\rk3568_t\vendor\bin\rk_anti_copy_board
/vendor/bin
```

Linux编译命令：

```
cd /home1/xxxx/rk_px30_linux/external/security/rk_tee_user/v2
rm -rf out/
./build.sh 6432

//copy to device
adb push Y:\rk_px30_linux\external\security\bin\optee_v2\lib\arm64\libteec.so.1
/lib64
adb push Y:\rk_px30_linux\external\security\bin\optee_v2\lib\arm64\tee-supplicant
/usr/bin
adb push
Y:\rk_px30_linux\external\security\rk_tee_user\v2\out\ta\rk_anti_copy_board\3d4fc
699-2065-7bb9-33c7-b6529b43c91a.ta /lib/optee_armtz
adb push
Y:\rk_px30_linux\external\security\rk_tee_user\v2\out\rk_anti_copy_board\rk_anti_
copy_board /usr/bin
```

测试命令：

```
root@RK3588:/# rk_anti_copy_board mid_gen
save data into OTP success!
```

3. UserSpace 层使用 rk_anti_copy_board 验证设备合法性。

测试命令：

```
root@RK3588:/# tee-supplicant &
root@RK3588:/# rk_anti_copy_board mid_verify
I/TA: Hello!
I/TA: verify success!
I/TA: Goodbye!
```

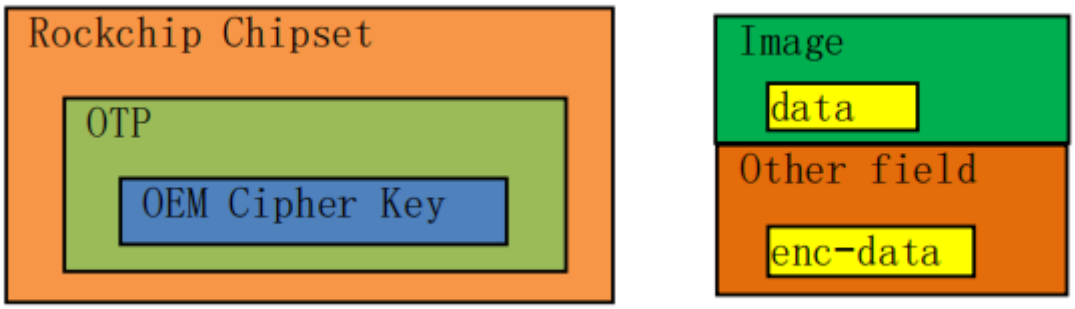
4. 高级方案

4.1 原理

Rockchip 芯片在 OTP 中有预留空间供客户存储自定义的私有密钥，使用方法请参考《Rockchip_Developer_Guide_OTP_CN.md》中 "OEM Cipher Key" 章节，

防抄板技术依赖于芯片 OTP 内存储的用户私有密钥 OEM Cipher Key，此密钥在工厂阶段生产时写入，为保证密钥不泄露，系统只提供烧写接口没有读取接口，部分平台还支持 Hardware Read 功能用于锁定密钥，锁定密钥后该密钥无法被CPU读取，安全性更高。

客户可以自行定制明文数据 data，使用私有密钥 OEM Cipher Key 并采用 AES 算法加密 data，得到密文数据 enc-data 并存放到 flash 上指定位置，也可以固化在代码中，在应用程序代码中设置明文数据 data。系统启动后，先使用私有密钥 OEM Cipher Key 解密密文数据 enc-data 得到解密数据 dec-data，将解密数据 dec-data 和应用程序代码中明文数据 data 进行对比。如果明文数据 data 或者密文数据 enc-data被篡改，或者 flash 固件被非法拷贝到其他非授权芯片平台，则会出现解密数据 dec-data 和明文数据 data 不匹配，匹配失败则认为非法，重新启动设备，从而达到防抄板目的。

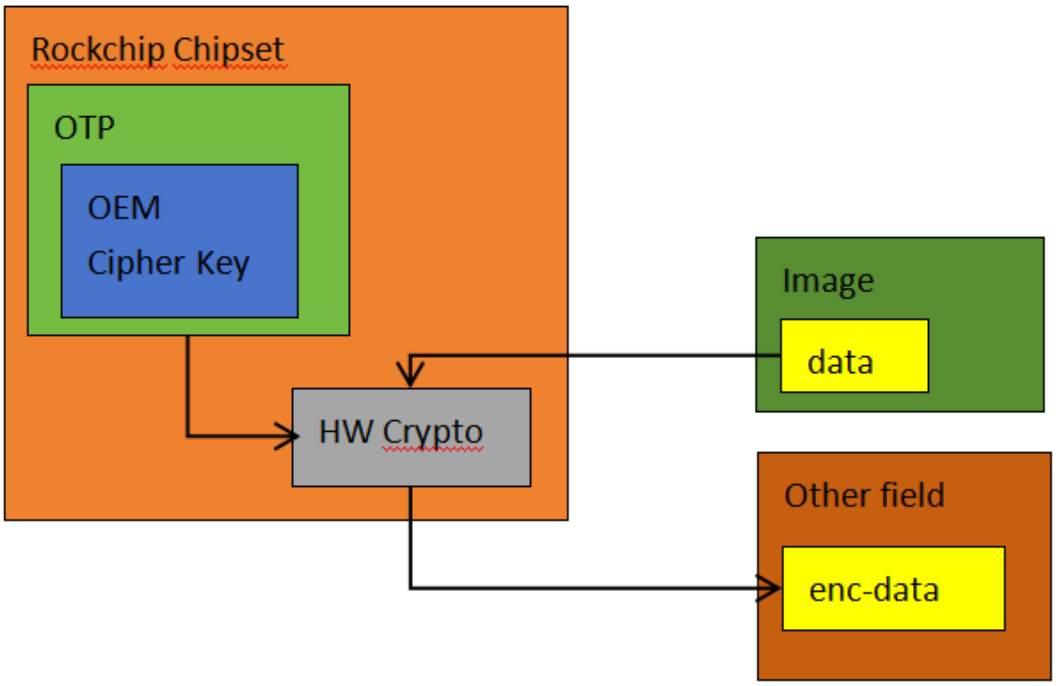


本方案优点：用户私有密钥 OEM Cipher Key被锁定后不可被 CPU 读取，具有更高的保密性、完整性和不可篡改等安全特性。

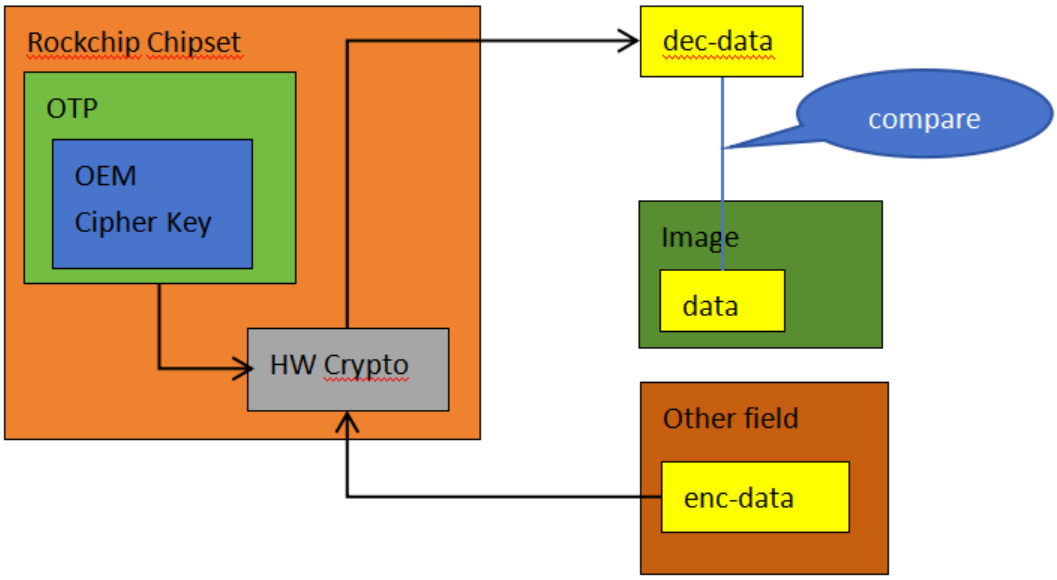
4.2 应用步骤

高级方案的应用步骤如下：

1. 客户生成自定义私有密钥 OEM Cipher Key，往 OTP 中写入 OEM Cipher Key 并锁定该密钥。
2. 在 APP 代码中设置明文数据 data。
3. 在 APP 代码中使用私有密钥 OEM Cipher Key 加密明文数据 data，得到密文数据 enc-data 并存放到 flash 上指定位置，或者固化到代码中。

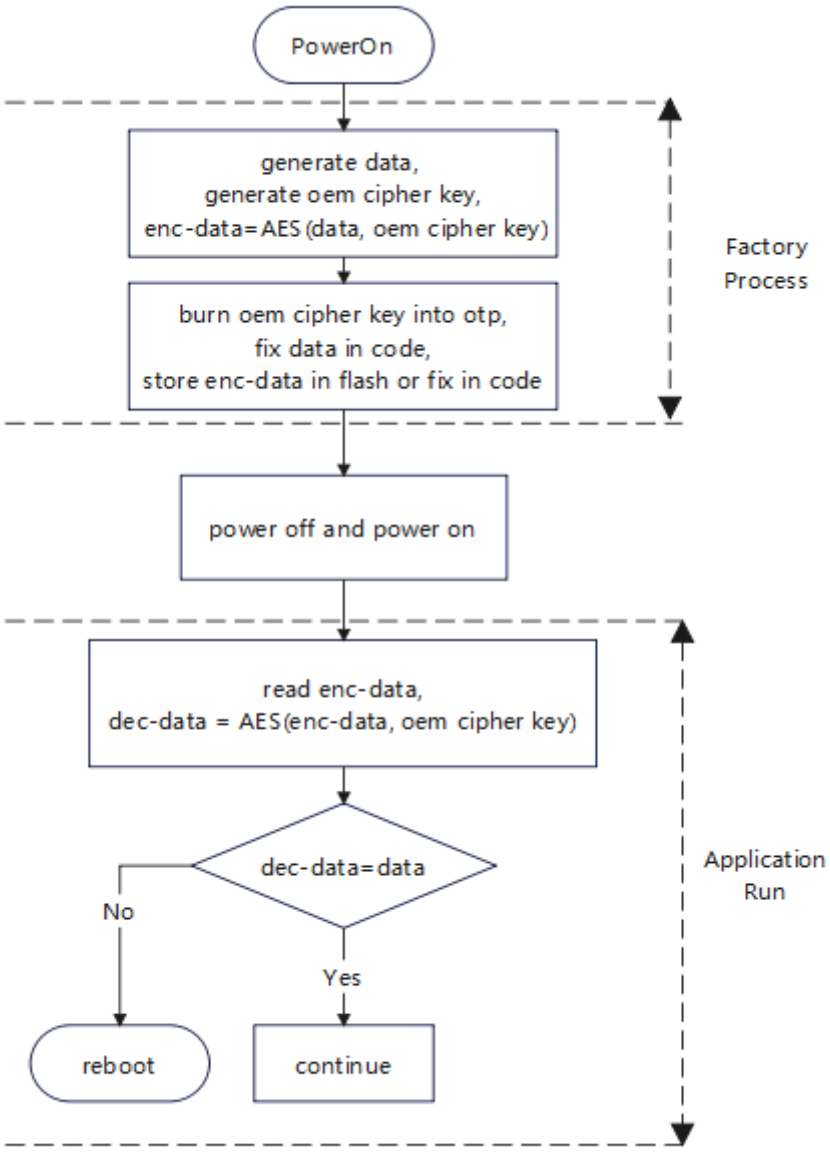


4. 系统启动后，APP 使用私有密钥 OEM Cipher Key 解密 flash 中密文数据 enc-data 得到解密数据 dec-data，将解密数据 dec-data 和应用程序代码中明文数据 data 进行对比。根据匹配结果选择正常运行或者立即重启。



4.3 完整流程图

中级方案的完整流程图如下所示：



4.4 参考案例

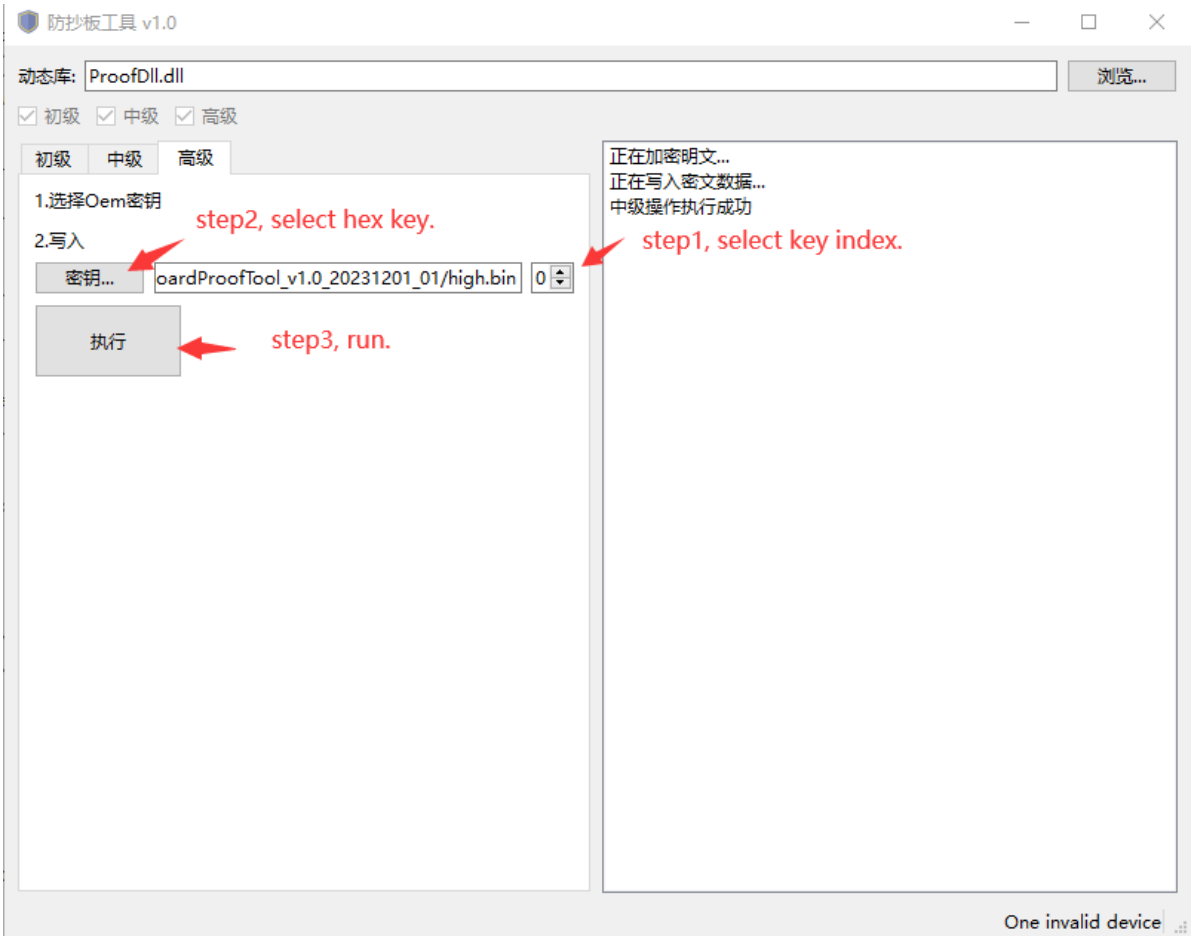
1. 使用 BoardProofTool 工具写入自定义私有密钥 OEM Cipher Key。

设备开机时长按 recovery 键进入Loader状态。

客户先创建 high.bin 文件，以16进制格式编辑该文件，写入16字节自定义私有密钥 OEM Cipher Key（也支持24或32字节长度）。

打开 BoardProofTool.exe 工具，选择Key Index，点击 “密钥” 选择 high.bin 文件，点击 “执行” 即可。

（注意：OTP是一次性可编程存储，同一个Key Index，每台设备只能写入一次自定义私有密钥 OEM Cipher Key）



2. UserSpace 层使用 rk_anti_copy_board 写入自定义私有密钥 OEM Cipher Key。

rk_anti_copy_board 源码位于 rk_tee_user/v2/host/rk_anti_copy_board 目录下。

编译方法与初级方案一致，这里不再赘述。

测试命令：

```
root@RK3588:/# rk_anti_copy_board high_gen
```

3. UserSpace 层使用 rk_anti_copy_board 验证设备合法性。

测试命令：

```
root@RK3588:/# rk_anti_copy_board high_verify
```

4.5 方案扩展

客户可以对私有密钥 OEM Cipher Key 的使用方法进行扩展。

例如使用 OEM Cipher Key 对关键数据进行加密存储，系统启动后当需要使用关键数据时将其解密。

也可以使用 OEM Cipher Key 对核心代码进行加密，系统启动后，需要运行核心代码时，先解密再执行。

5. 扩展安全

客户可以将防抄板技术与以下安全技术配合，可以进一步减低被抄板的分险，提高系统安全性。

- 安全启动。启动时校验固件的合法性，保证运行的固件未被篡改，防止非法程序运行。安全启动流程及应用步骤请参考 Secure Boot 相关文档。
- 固件加密。将固件进行加密再烧写，启动时才解密运行，防止非授权用户获取 flash 上明文固件，防止固件被反汇编分析。