life.augmented

AN5056
应用笔记

X-CUBE-SBSFU STM32Cube扩展包集成指南

引言

X-CUBE-SBSFU安全启动和安全固件更新解决方案允许使用新固件版本更新STM32微控制器内置程序、添加新功能并更正潜在问题。更新过程以安全方式执行，以防止未经授权的更新以及访问设备上的机密数据。

安全启动（可信根服务）是一种不可变代码，总是在系统复位后执行。它检查STM32静态保护、激活STM32运行时保护措施，然后在每次执行前验证应用程序代码的真实性和完整性，以确保无效或恶意代码不能运行。

安全固件更新应用程序通过使用Ymodem协议的UART接口接收固件镜像。它会在安装代码之前检查其真实性和完整性。固件更新可更新整个固件镜像，或者部分镜像。示例代码可灵活配置，可配置使用对称或非对称加密方案，采用明文或密文方案。工程以两种方案提供：

- 单固件镜像配置，以使固件镜像尺寸最大化
- 双固件镜像配置，以确保安全镜像安装，并启用物联网器件中常用的OTA固件更新功能。

安全密钥管理服务通过PKCS #11 API（基于KEY ID的API）向用户应用程序提供加密服务，这些API在受保护和隔离的环境中执行。用户应用程序密钥存储在受保护和隔离的环境中，以便进行安全更新：真实性检查、数据解密和数据完整性检查。

STSAFE-A100是一种防篡改安全元件（通过硬件通用标准EAL5+认证），用于托管X509证书和密钥，并在安全启动和安全固件更新过程中执行用于固件镜像身份确认的验证。

X-CUBE-SBSFU用户手册（UM2262）讲解X-CUBE-SBSFU入门知识并详细介绍SBSFU功能。该应用笔记描述了如何调整X-CUBE-SBSFU并将其与用户应用程序集成；它回答了诸如以下问题：

- 如何将X-CUBE-SBSFU移植到另一块板？
- 如何根据用户的需求微调X-CUBE-SBSFU配置？
- 如何生成新的固件加密密钥？
- 如何调试X-CUBE-SBSFU？
- 如何调整SBSFU？
- 如何调整用户的应用程序？

注： *在该应用笔记中，使用IAR™EWARM IDE作为示例，为项目配置提供指南。安全启动和安全固件更新应用程序被称为SBSFU。*

目录

表格索引

图片目录

1 概述

*表 1*和*表 2*给出了相关缩略语和术语的定义，帮助您更好地理解本文档。

表1. 缩略语列表

缩略语	说明
AES	高级加密标准
DAP	调试访问端口
ECDSA	椭圆曲线数字签名算法
GCM	AES Galois/计数器模式
HAL	硬件抽象层
IDE	集成开发环境
FWALL	防火墙
MPU	存储器保护单元
OTFDEC	动态解密
PEM	隐私增强邮件
PCROP	专有代码读保护
RDP	读出器件保护
SB	安全启动
SE	安全引擎
SFU	安全固件更新
SBSFU	安全启动和安全固件更新流程
UART	通用异步收发器
WRP	写保护

表2. 术语列表

术语	说明
固件映像	二进制映像（可执行文件），作为用户应用程序由设备来运行。
固件头文件	元数据包，描述要安装的固件映像。它包含 固件信息和密码信息。
mbedTLS	TLS和SSL协议的mbed实现，以及各自的 加密算法。
*sfb*文件	包含固件头文件和固件映像的二进制文件。

X-CUBE-SBSFU安全启动和安全固件更新扩展包可在基于Arm®(a) Cortex®-M处理器的STM32 32位微控制器上运行。

2 相关文档

1. *面向STM32H7系列的STM32CubeH7入门*（UM2204）
2. *面向STM32G4系列的STM32CubeG4入门*（UM2492）
3. *面向STM32L0系列的STM32CubeL入门*（UM1754）
4. *面向STM32L1系列的STM32CubeL1 MCU软件包入门*（UM1802）
5. *面向STM32WB系列的STM32CubeWB入门*（UM2550）
6. *适用于STM32L4系列和STM32L4+系列的STM32CubeL4入门*用户手册(UM1860)
7. *适用于STM32F4系列的STM32CubeF4 MCU软件包入门*用户手册（UM1730）
8. *适用于STM32F7系列的STM32CubeF7 MCU软件包入门*用户手册（UM1891）
9. *适用于STM32G0系列的STM32CubeG0入门*用户手册（UM2303）
10. *X-CUBE-SBSFU STM32Cube扩展包入门*用户手册（UM2262）
11. *STM32Cube扩展包开发指南*用户手册（UM2285）
12. *STM32Cube扩展包开发清单*用户手册（UM2312）
13. *STM32CubeProgrammer 软件描述*用户手册（UM2237）
14. *STM32F3系列*、*STM32F4系列*、*STM32L4系列和STM32L4+系列Cortex®-M4编程手册*（PM0214）
15. *STM32F7系列和STM32H7系列Cortex®-M7处理器编程手册*（PM0253）
16. *STM32L0系列和STM32G0系列Cortex®-M0+编程手册*（PM0223）
17. *认证*、*最先进的外设安全措施*、*以及*STSAFE-A100的物联网器件数据表（DS12911）。

a. Arm是Arm Limited（或其子公司）在美国和或其他地区的注册商标。

3 移植X-CUBE-SBSFU到另一块板上

X-CUBE-SBSFU补充了STM32Cube™软件技术，使不同STM32微控制器之间更易移植。它附带一组在给定的STM32板上实现的示例，可作为有用的起点将X-CUBE-SBSFU移植到另一块STM32板。本文档中将NUCLEO-L476RG和NUCLEO-L432KC板作为示例。

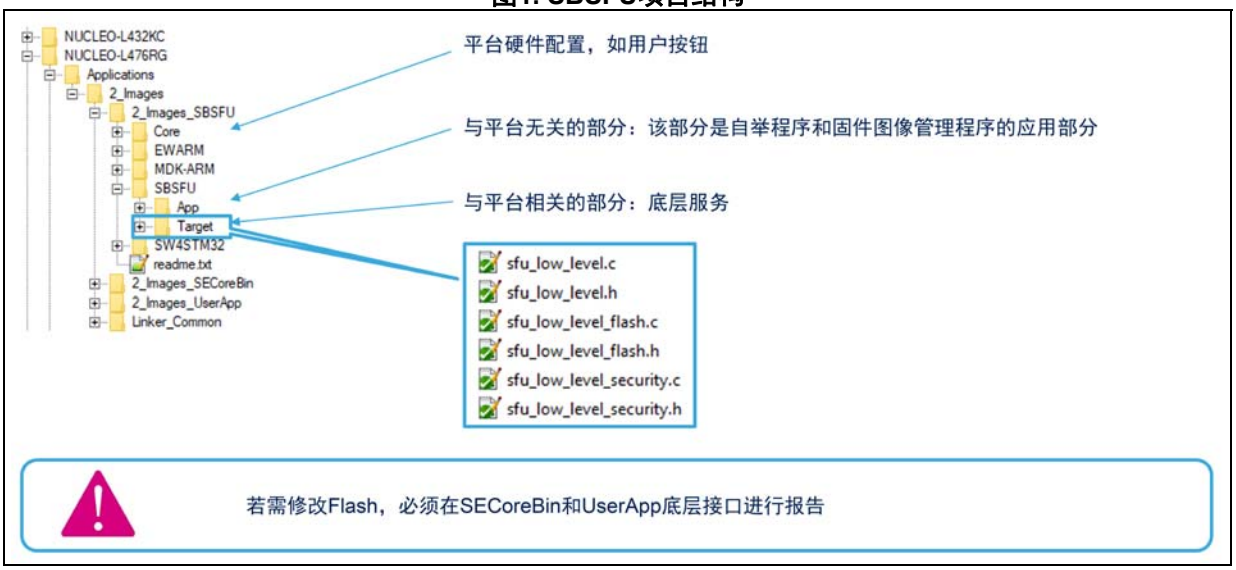
3.1 硬件适应

需要进行一些更改，以使X-CUBE-SBSFU适应另一块板：

1. GPIO配置，用于与主机PC进行UART通信（在文件*sfu_low_level.h*中）
2. Flash配置：NUCLEO-L432KC 给出了单存储区Flash接口的例子，而NUCLEO-L476RG则基于双存储区（在文件*sfu_low_level.c*中）
3. 按钮配置：NUCLEO-L476RG给出了基于按钮的例子，而NUCLEO-L432KC使用GPIO模拟虚拟按钮（在文件*app_hw.h*中）
4. Tamper GPIO引脚配置（在文件*sfu_low_level_security.h*中）
5. DAP - 调试端口配置（在文件*sfu_low_level_security.h*中）
6. I^2C总线配置，用于与STSAFE-A100进行通信（在文件*stsafea_service_interface.c of B-L475E-IOT01A\Applications\2_Images_STSAFE\2_Images_SECoreBin*中）。

*图 1*显示SBSFU项目结构，以及期望移植更改的文件位置。

图1. SBSFU项目结构



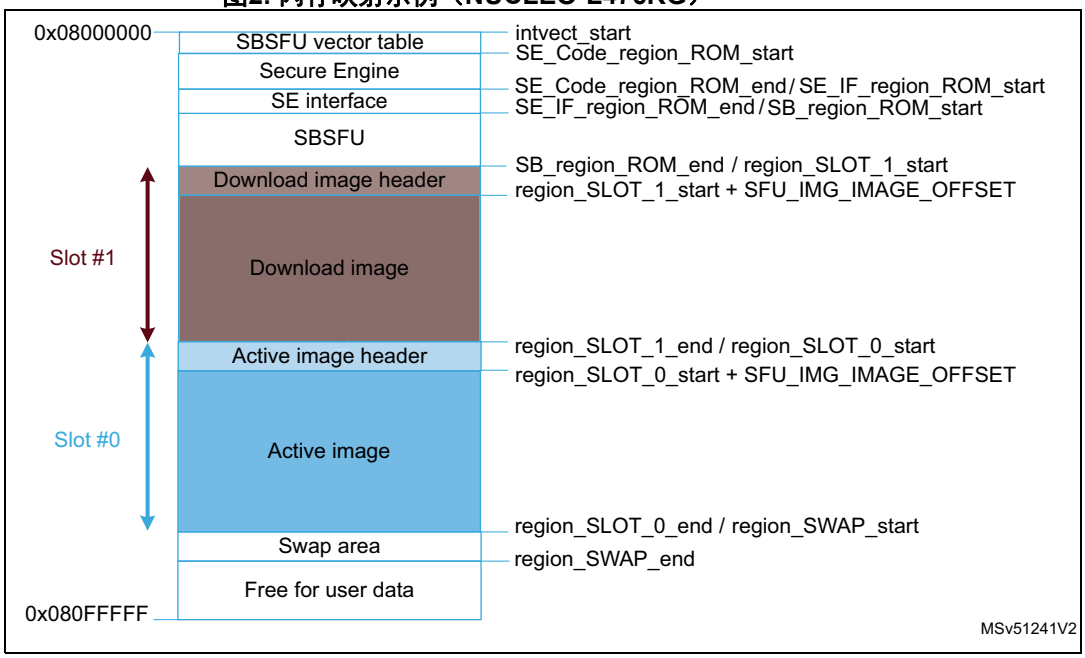
3.2 内存映射定义

正如X-CUBE-SBSFU用户手册（参见[5]）中所强调，一个关键因素是器件闪存内所有元素的布局：

- 安全引擎：受保护的环境，以管理所有关键数据和操作。
- SBSFU：安全启动和安全固件更新流程
- 插槽 #0：该插槽包含了活动固件（固件头文件+固件）。
- 插槽 #1：此插槽用于存储下载的固件（固件头文件+加密固件），以便在下次重启时安装。
- 交换区：用来在安装过程中交换插槽#0和插槽#1的内容的Flash区域。

*图 2*通过NUCLEO-L476RG示例讲解闪存映射。

图2. 闪存映射示例（NUCLEO-L476RG）



三个项目（SECoreBin、SBSFU、UserApp）共享的链接器文件定义在*Linker_Common*文件夹中分组，如*图 3*中所示：

- *mapping_fwimg.icf*：包含固件镜像定义，比如插槽 #0、插槽 #1、以及交换区
- *Mapping_sbsfu.icf*：*contains*：包含SBSFU定义，例如SE_Code_region、SE_Key_region、以及SE_IF_region
- *Mapping_export.h*：*从mapping_sbsfu.icf和mapping_fwimg.icf*：从*mapping_sbsfu.icf*和*mapping_fwimg.icf* 导出符号到SBSFU应用程序

当添加更多代码后，每个区域可能要做相应的扩展，或者转移到另一个位置，这些都是允许的，只要最终的设置满足应用的安全需求。

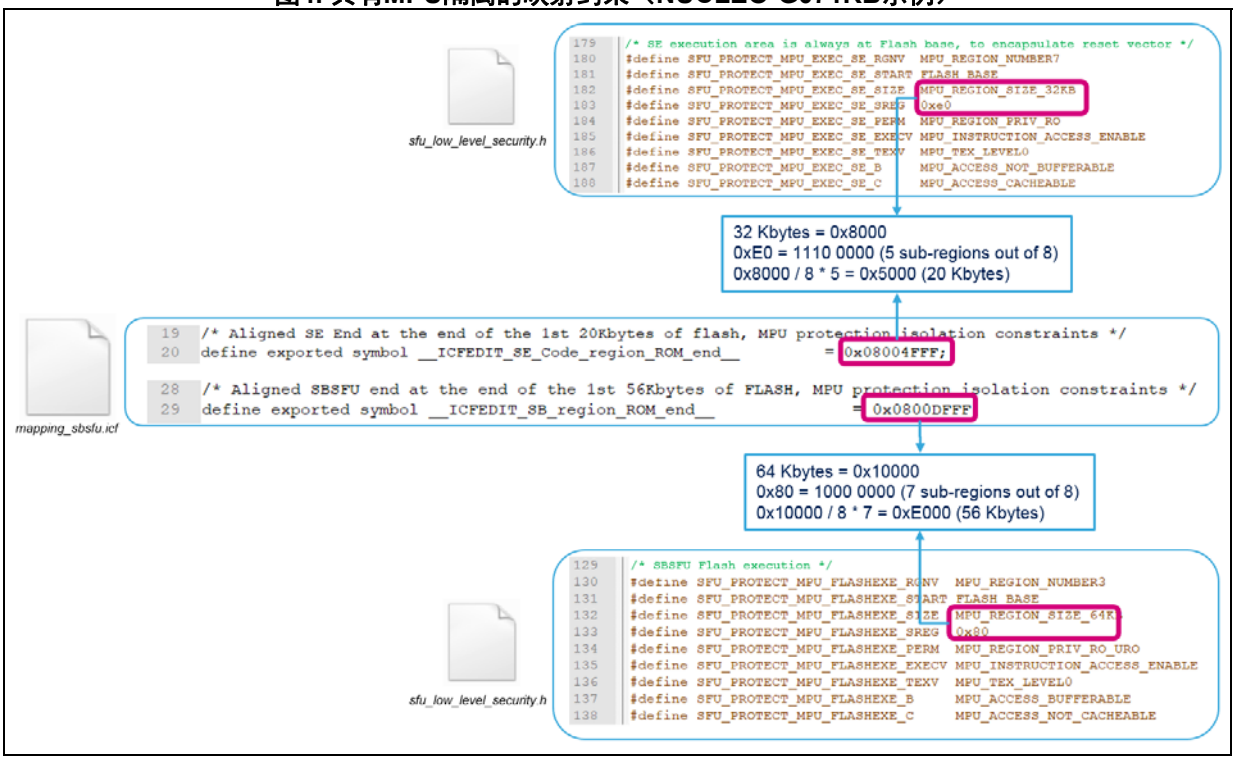
图3. 链接器文件架构

安全外设配置（RDP、WRP、PCROP、FWALL、安全用户存储区（如果当前型号拥有））是基于SBSFU链接符号自动计算的；但是MPU配置不一样，因为存在以下约束：

- 每个MPU区域的基址必须是MPU区域大小的倍数。
- 每个MPU区域可以划分为8个子区域，以调整大小。

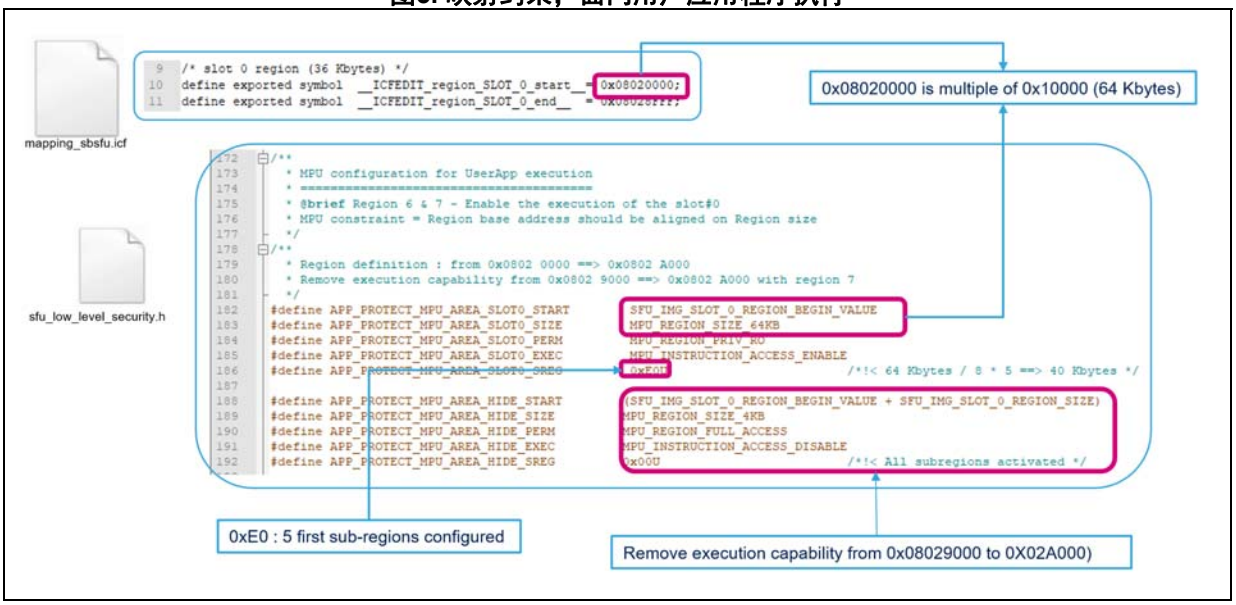
*图 4*中说明了具有MPU隔离的映射约束。

图4. 具有MPU隔离的映射约束（NUCLEO-G071RB示例）



另一个典型用例是插槽#0区域的MPU配置，以授权执行用户应用程序。*图 5：映射约束，面向用户应用程序执行*演示如何遵守NUCLEO-L073RZ上的MPU约束。

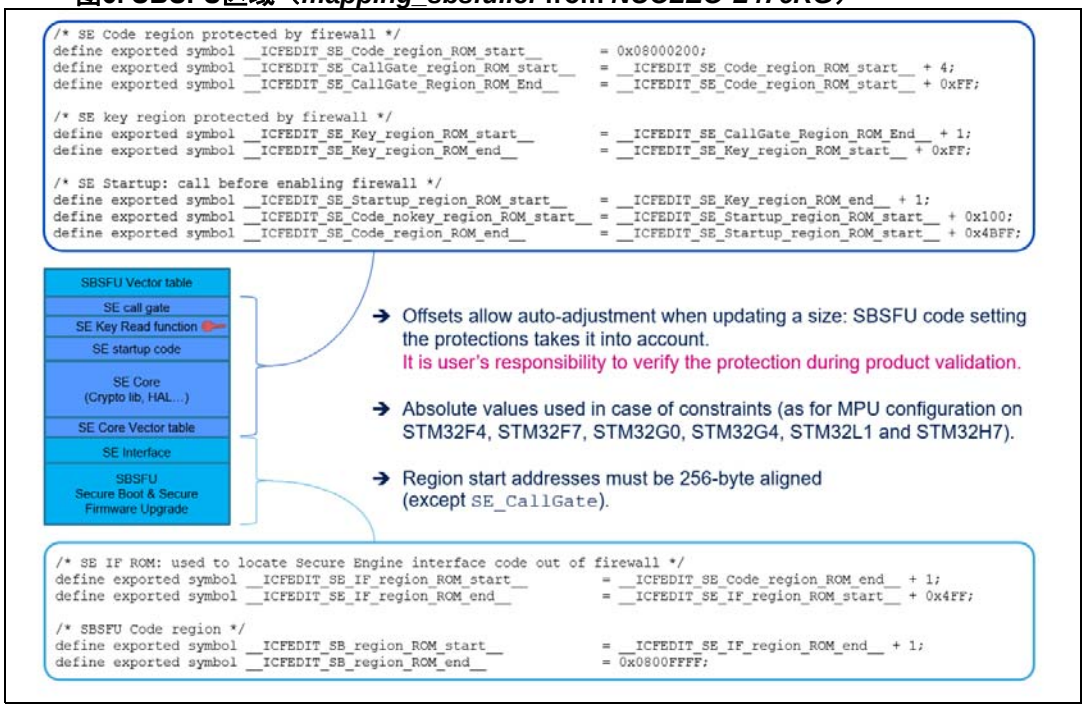
图5. 映射约束，面向用户应用程序执行



3.2.1 SBSFU区域定义参数

*图 6*在*mapping_sbsfu.icf*文件中给出这些参数，用于SBSFU区域的配置。

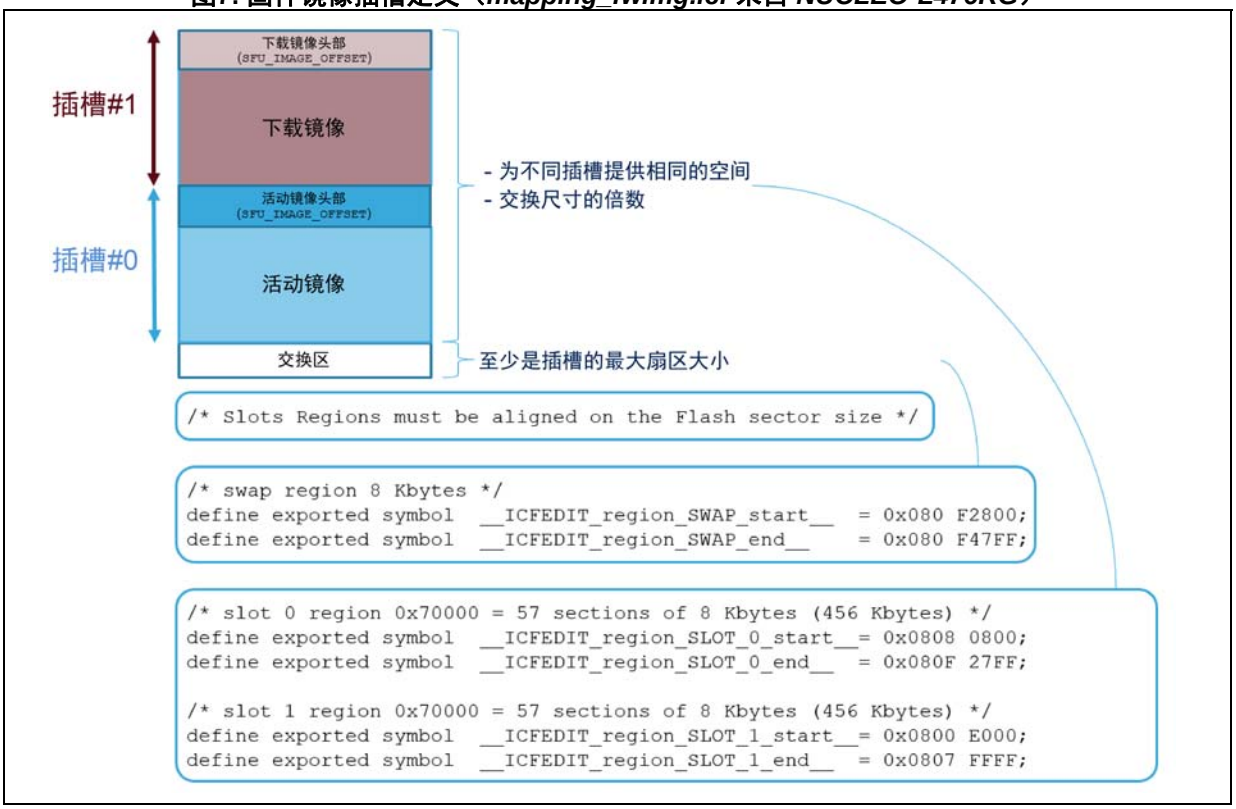
图6. SBSFU区域（*mapping_sbsfu.icf* from *NUCLEO-L476RG*）



3.2.2 固件镜像插槽定义参数

*图 7*在*mapping_sbsfu.icf*文件中给出这些参数，用于镜像区域的配置。

图7. 固件镜像插槽定义（*mapping_fwimg.icf* 来自 *NUCLEO-L476RG*）



遵守SBSFU约束要求满足以下条件：

- 插槽区域必须与闪存扇区大小对应，对于STM32L4系列中的设备，闪存扇区大小为2048字节（0x800）
- 交换区的最小尺寸是4 Kb，至少与最大扇区的尺寸相当
- SLOT_0 的尺寸必须是交换区尺寸的倍数
- SLOT_0 和SLOT_1的尺寸必须相等，除非使用局部更新功能

对于STM32L4双存储区闪存器件，特定于防火墙的约束有：

- 防火墙代码段必须在bank1中，防火墙非易失性数据（包括插槽#0的头部）段必须在bank2中。
- 非易失性数据段必须与防火墙代码段重叠，以确保即使交换了存储区，机密也始终受到保护。
  *图 8：双存储区产品的防火墙配置约束*和*图 9：交换存储区之后的防火墙配置*举例说明NUCLEO-L476RG上的防火墙配置，以及交换存储区的后果。

图8. 双存储区产品的防火墙配置约束

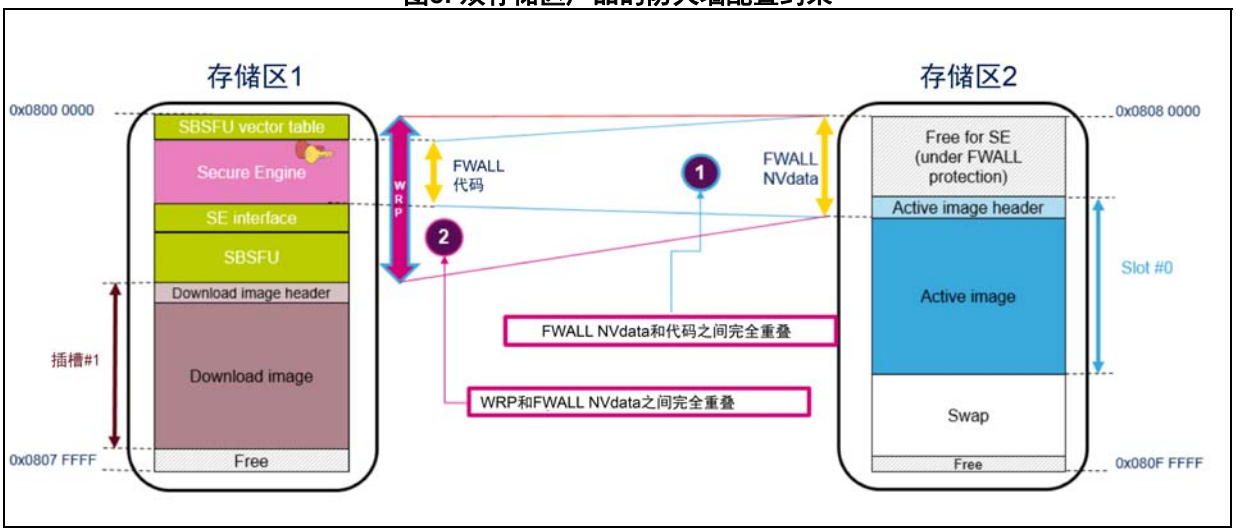
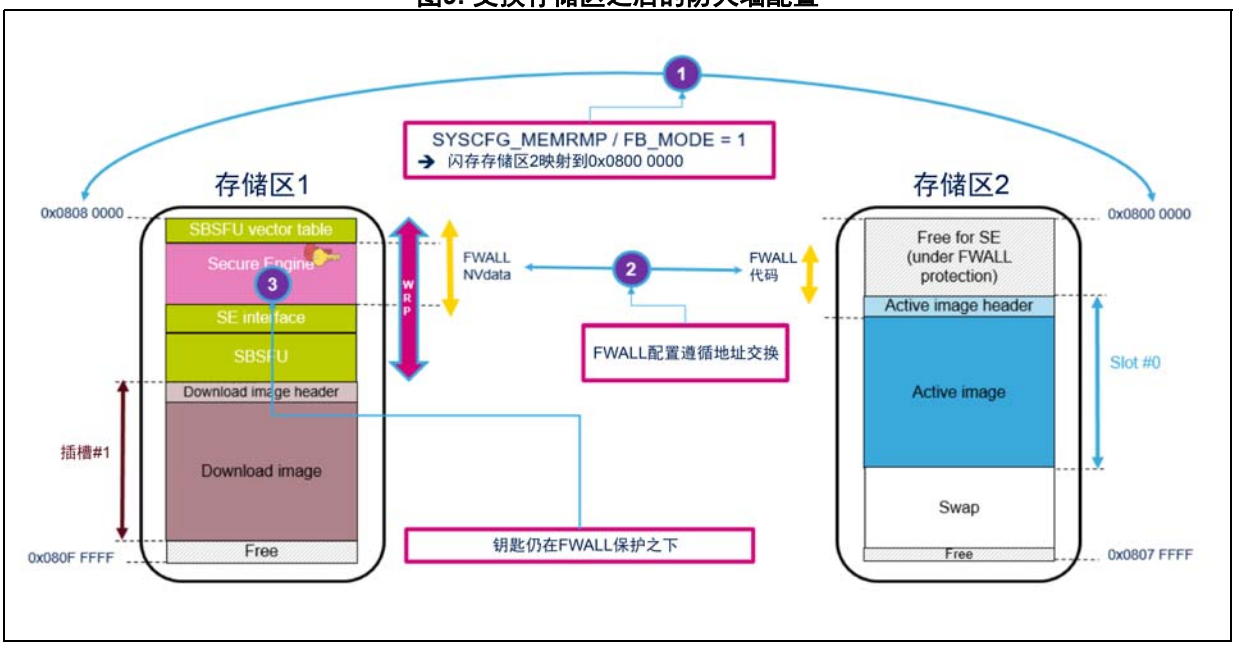


图9. 交换存储区之后的防火墙配置



对于STM32G0系列、STM32G4系列、以及STM32H7系列，存在一种约束：插槽 #0 必须恰好映射在SBSFU代码之后，这样才能受到安全内存的保护。

SFU_IMAGE_OFFSET值取决于STM32微控制器系列：

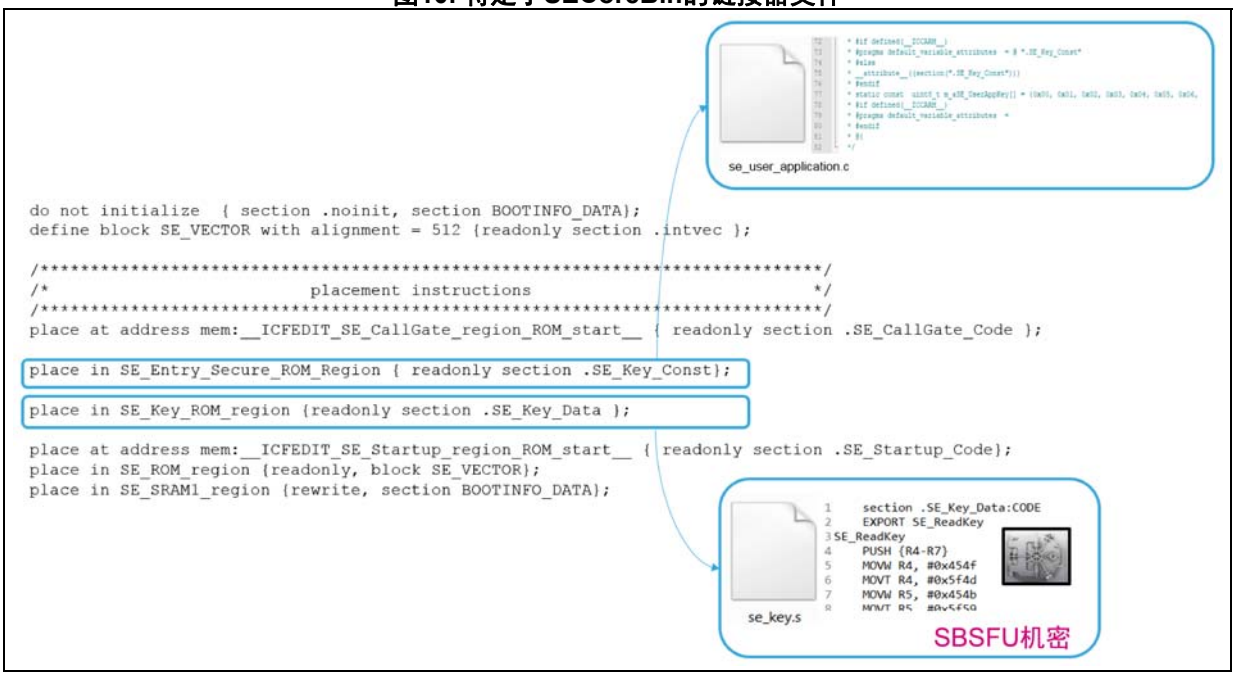
- 对于STM32L4系列、STM32L0系列、STM32L1系列、STM32WB 系列、以及STM32F4系列，使用默认值：512字节。
- 对于STM32F7系列：1024字节。
  （因为采用Cortex®-M7，向量表必须与1024字节对应）。
- 对于STM32G0系列：2048字节。
  安全用户内存结束地址与闪存扇区大小一致。
- 对于STM32G4系列：4096字节。
  安全用户内存结束地址与闪存扇区大小一致。
- 对于STSAFE-A变体：2048字节。
  为了包含X509证书，镜像头部的长度为2048字节。

注： *如果系列器件带有基于MPU的隔离或基于防火墙的隔离，Slot#0配置的MPU约束必须按照图 5所示加以验证。*

3.2.3 特定于项目的链接器文件

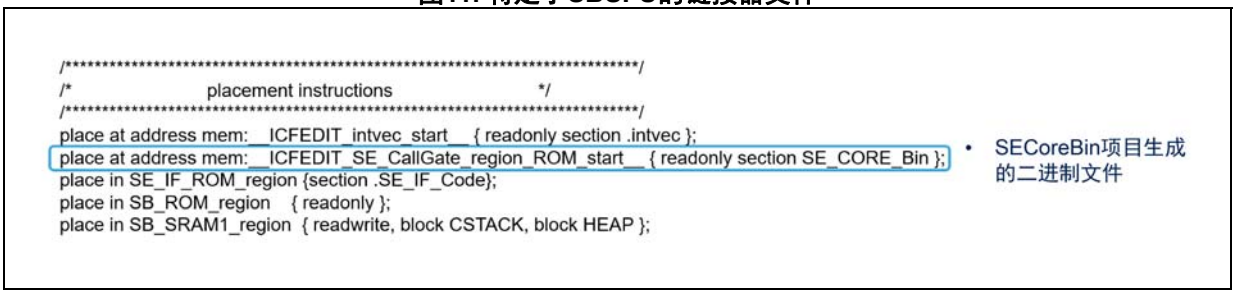
SECoreBin放置关键代码和关键数据（如机密），如图 10中所示。

图10. 特定于SECoreBin的链接器文件



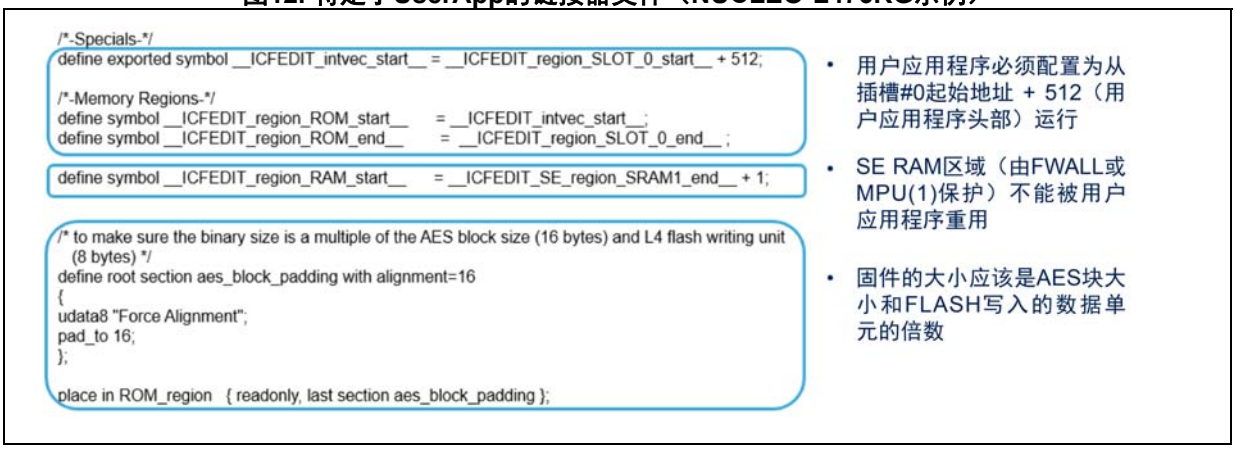
SBSFU链接器文件负责SBSFU应用程序位置，包括SECoreBin二进制文件，如*图 11*中所示。

图11. 特定于SBSFU的链接器文件



UserApp必须配置为在插槽 #0（SLOT_0 起始地址 + SFU_IMG_IMAGE_OFFSET），如*图 12*中所示，其中面向STM32L4系列的SFU_IMG_IMAGE_OFFSET位512字节。

图12. 特定于UserApp的链接器文件（NUCLEO-L476RG示例）



1. 取决于ST32微控制器系列

4 SBSFU配置

4.1 要配置的特性

X-CUBE-SBSFU支持：

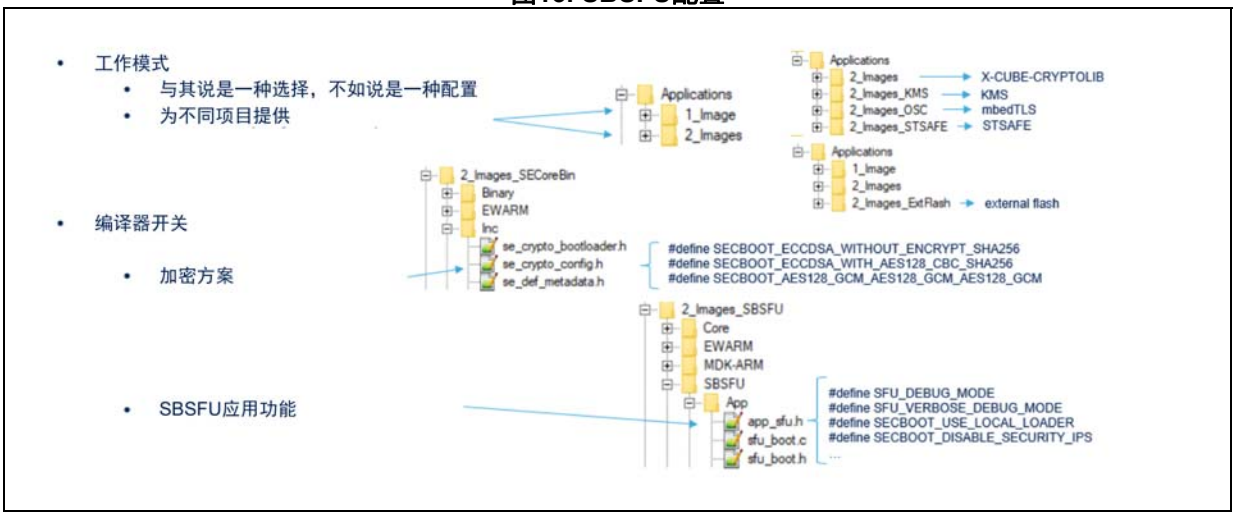
- 两种操作模式：双镜像和单镜像
- 3种加密方案，使用对称和非对称加密操作
- 两种加密中间件：
  - ST的中间件：X-CUBE-CRYPTOLIB库，集成在*1_Image*和*2_Images*变体中。
  - 第三方中间件：mbedTLS（开源代码）加密服务。为*2_Images_OSC*变体中的32L496GDISCOVERY、B-L475E-IOT01A、32F413HDISCOVERY和32F769IDISCOVERY板提供了示例。
- STSAFE-A100安全元件用于托管X509证书和密钥。为*2_Images_STSAFE* 变体中的B-L475E-IOT01A板提供了示例。
- KMS中间件。为*2_Images_KMS* 变体中的B-L475E-IOT01A板提供了示例。
- 外部闪存，带OTFDEC。为*2_Images_ExtFlash*变体中的STM32H7B3I-DK板提供一个示例，该示例使用了一种采用AES-CTR固件加密的特定方案。

以下配置选项可通过编程开关来设置：

- 可以移除本地加载器，以减少内存占用（仅双镜像）
- 可以激活Verbose开关，使调试更容易
- 可以禁用调试模式（在SBSFU执行期间，终端上不再有printf）以减少内存占用
- 可以关闭安全IP，使调试更容易

*图 13*呈现的SBSFU配置解决方案带相关文件和编译开关。

图13. SBSFU配置



4.2 加密方案选择

X-CUBE-SBSFU提供了同时使用非对称和对称加密的三种加密方案。

- 用于固件验证的ECDSA非对称加密和用于固件解密的AES-CBC对称加密
- 用于固件验证（未加密固件）的ECDSA非对称加密技术
- 用于固件验证和解密的AES-GCM对称加密技术

如*图 14*中所描述，通过SECBOOT_CRYPTO_SCHEME编译开关在这些方案中进行选择。

图14. 切换加密方案

注： *对于STSAFE变体，选择了SECBOOT_X509_ECDSA_WITHOUT_ENCRYPT_SHA256加密方案。*

对于带动态解密（OTFDEC）的外部闪存，选择SECBOOT_ECCDSA_WITH_AES128_CTR_SHA256 加密方案。

4.3 安全配置

SBSFU示例采用了STM32安全保护配置，允许保护机密免受外部和内部攻击。

STM32的各种安全外设可由用户决定是否单独停用，以实现不同的保护级别（例如，对于STM32L4系列器件，防火墙和PCROP允许激活防止内部攻击的保护措施）。任何STM32安全配置修改都需要在系统产品级别进行安全保护评估，以确保根据产品约束和规范适当地设置保护。

在开发阶段，为了简化调试，可能需要禁用所有IP。

*图 15*显示文件*app_sfu.h*中针对STM32L4系列和STM32L0系列提供的各种安全配置解决方案。

图15. STM32L4系列和STM32L0系列 安全配置（*app_sfu.h*）

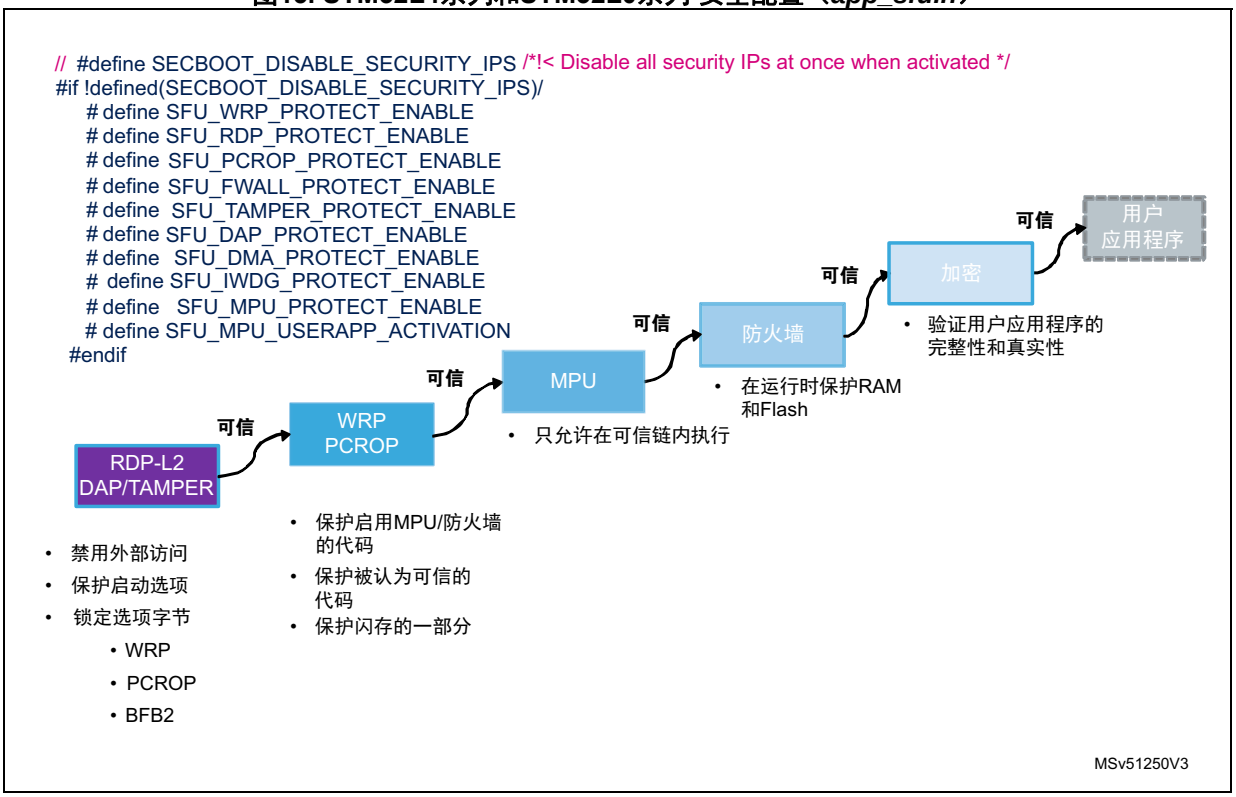


*图 16*显示文件*app_sfu.h*中针对STM32F4系列、STM32F7系列和STM32L系列提供的各种安全配置解决方案。

图16. STM32F4系列、STM32F7系列和STM32L1系列安全配置（*app_sfu.h*）

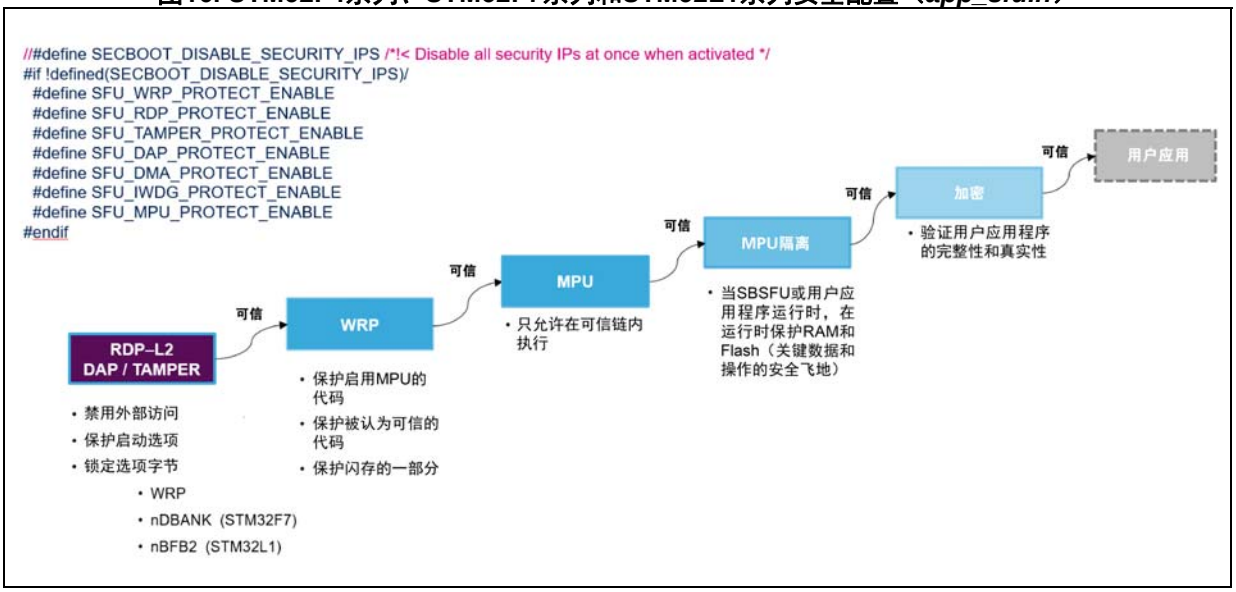


*图 17*显示文件*app_sfu.h*中针对STM32WB系列提供的各种安全配置解决方案。

图17. STM32G0系列、STM32G4系列和STM32H7系列安全配置（*app_sfu.h*）

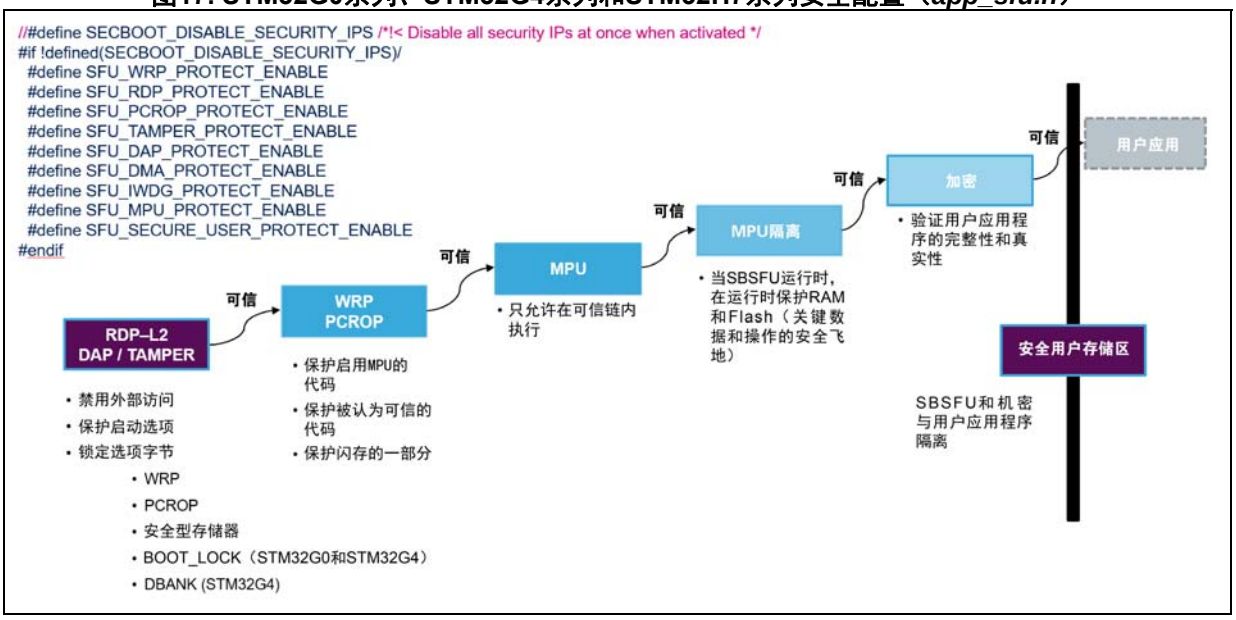
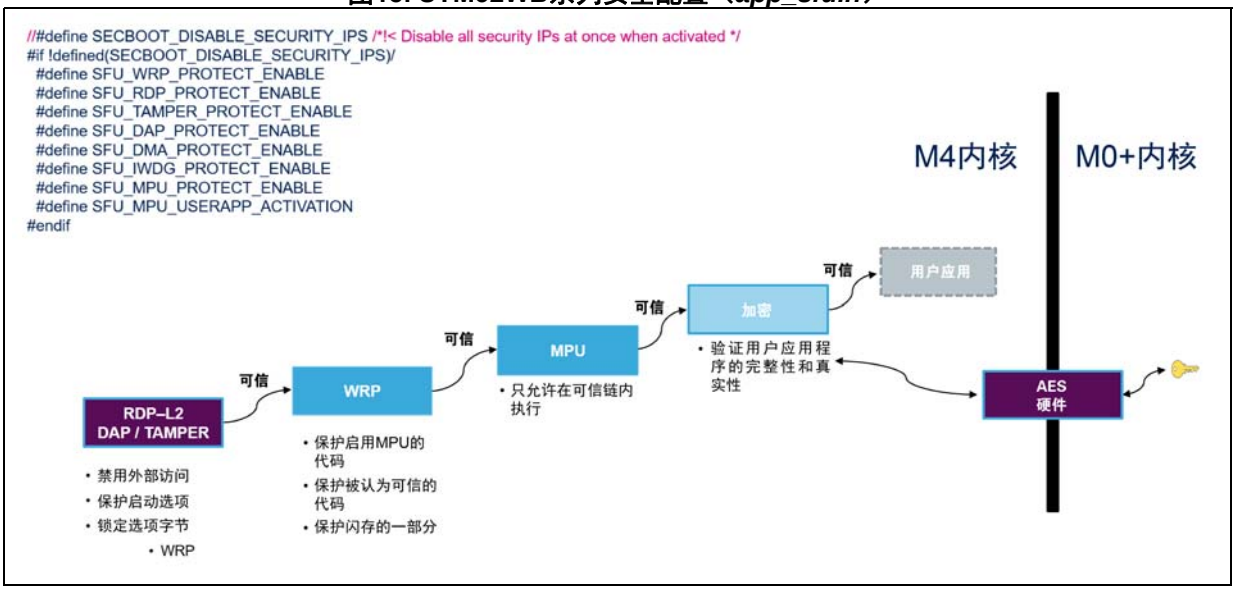


*图 18*显示文件*app_sfu.h*中针对STM32WB系列提供的各种安全配置解决方案。

图18. STM32WB系列安全配置（*app_sfu.h*）



4.4 开发或生产模式配置

进行任何代码修改之前的第一步通常是在开发模式下配置SBSFU项目，以启用IDE调试设施和添加SBSFU调试跟踪：

1. 禁用所有安全保护：SFU_xxx_PROTECT_ENABLE
2. 禁用SFU_FINAL_SECURE_LOCK_ENABLE
3. 激活SFU_FWIMG_BLOCK_ON_ABNORMAL_ERRORS_MODE
4. 激活SECBOOT_OB_DEV_MODE
5. 可以选择激活verbose模式：SFU_VERBOSE_DEBUG_MODE（如需详细了解对映射的影响，请参阅*第 5节：生成加密密钥*）

在开发阶段的最后，SBSFU项目必须配置为生产模式，以便最终版本：

1. 激活所有需要的安全保护：SFU_xxx_PROTECT_ENABLE
2. 禁用verbose模式：SFU_VERBOSE_DEBUG_MODE
3. 禁用SFU_FWIMG_BLOCK_ON_ABNORMAL_ERRORS_MODE
4. 禁用SECBOOT_OB_DEV_MODE
5. 激活SFU_FINAL_SECURE_LOCK_ENABLE以配置RDP级别2。在STM32H7系列器件上，如果启用了SFU_FINAL_SECURE_LOCK_ENABLE，也会配置安全用户内存。

读保护级别2是强制性的，以实现最高级别的保护并实现可信根。用户负责在产品制造阶段被编程的最终软件中将其激活。

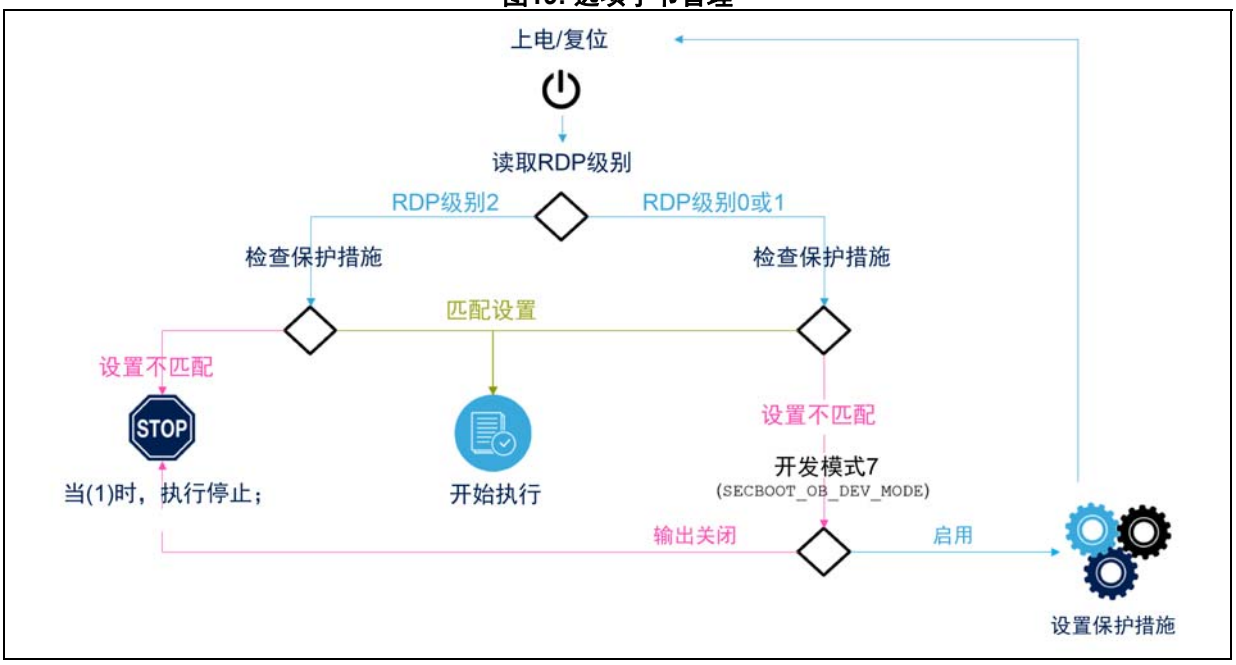
在生产模式下，安全启动检查选项字节值（RDP、WRP、PCROP、安全用户内存）和块的执行，以防检测到错误配置。少数其他选项字节必须配置（具体取决于平台），例如:

- 为采用双存储区闪存的STM32L4系列和STM32L0系列禁用BFB2
- 为STM32F7系列启用nDBANK
- 为STM32L1系列启用nBFB2
- 为STM32G0系列和STM32G4系列启用BOOT_LOCK
- 在STM32G4系列上禁用DBANK

注意： 在生产阶段对软件进行编程之后，选项字节必须通过STM32CubeProgrammer（STM32CubeProg）配置为生产模式值。如果不这样做，器件仍然是非安全。请参阅*[13]*，获取STM32CubeProgrammer的使用方法。

*图 19*显示如何在SBSFU启动时管理选项字节：

图19. 选项字节管理



5 生成加密密钥

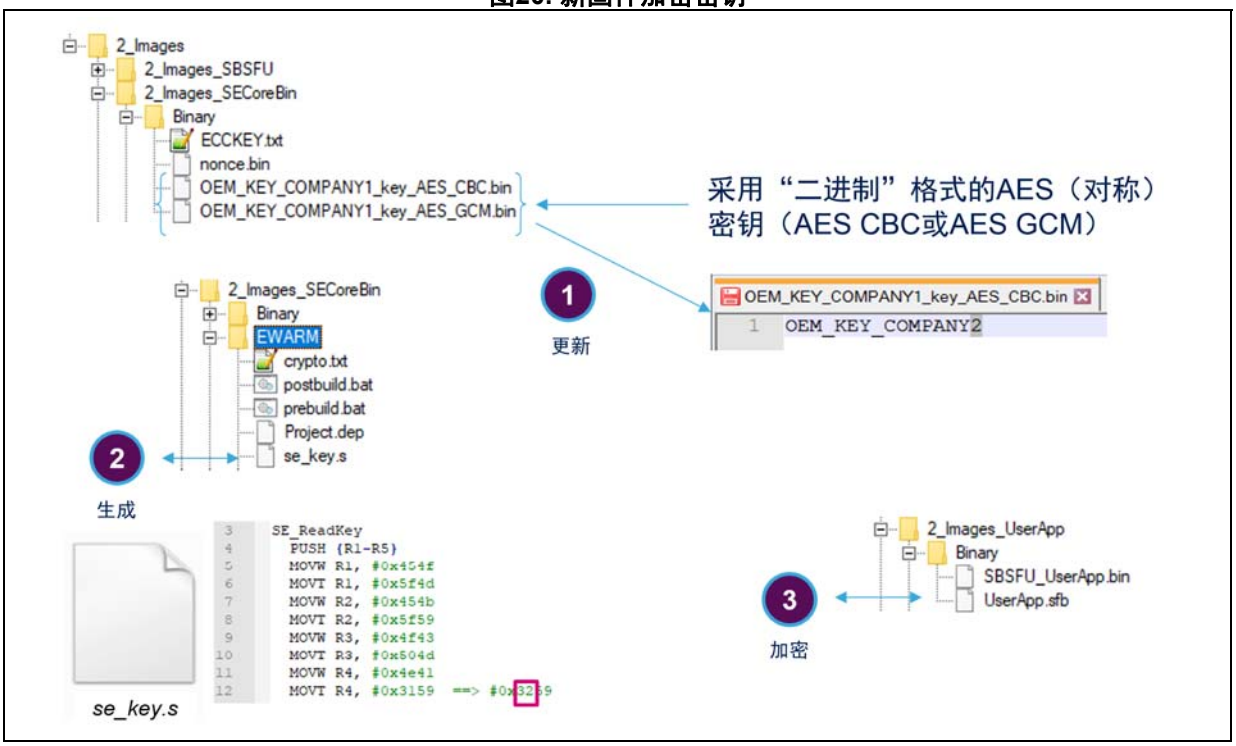
5.1 生成新的固件AES加密密钥

密钥生成和固件加密是在编译过程中通过*prebuild.bat*和*postbuild.bat*脚本自动进行的（请参阅*[5]*，获取 有关构建过程的详细描述）。

*图 20*显示修改固件加密密钥的几个步骤：

1. 在文件*OEM_KEY_COMPANY1_keys_AES_xxx.bin*中更改密钥值
2. 编译SECoreBin：执行*prebuild.bat*，生成*se_key.s*
3. 编译UserApp：执行*postbuild.bat*，加密UserApp

图20. 新固件加密密钥



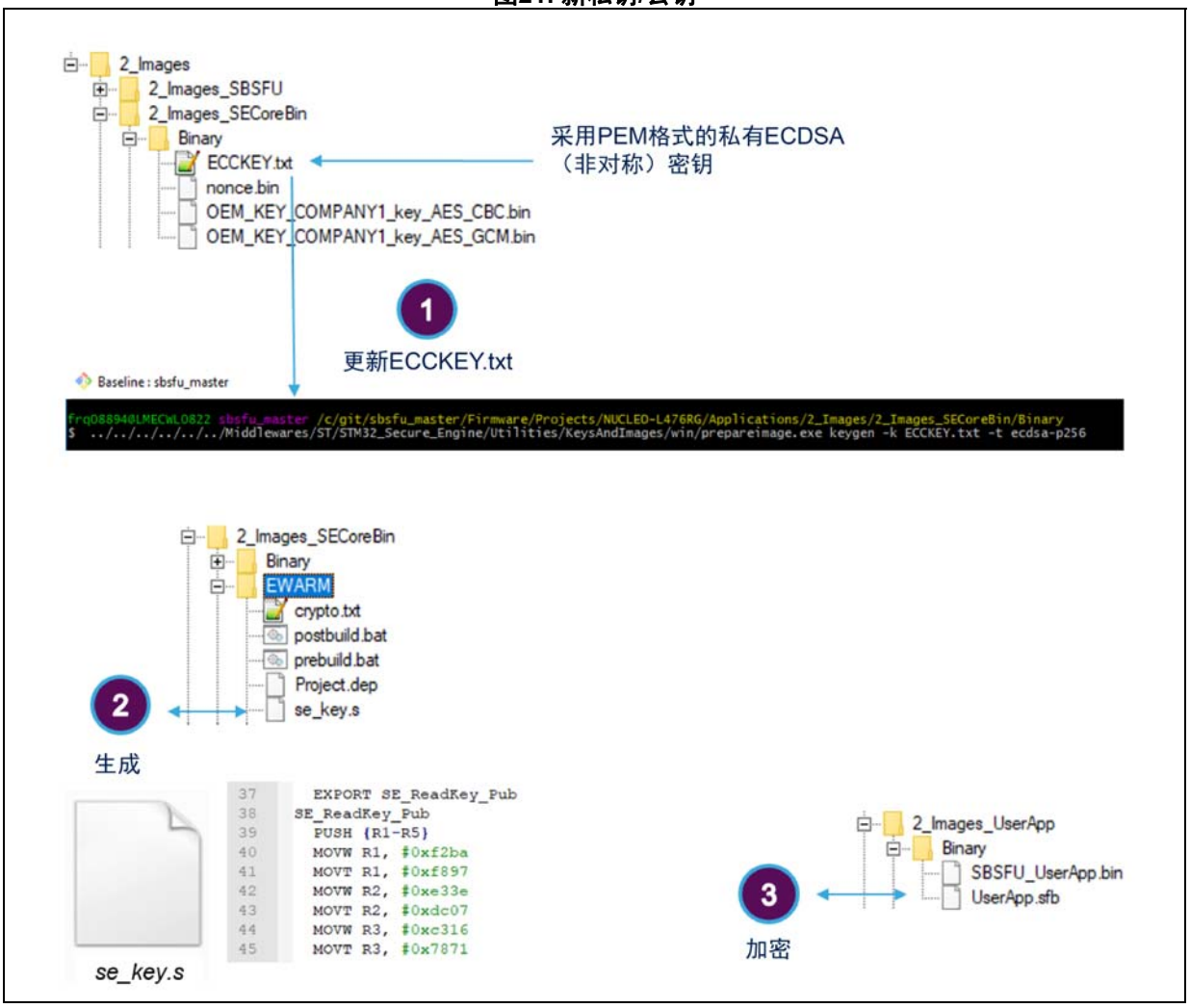
5.2 生成用于固件验证的新的公钥/私钥对（ECDSA）

至于AES加密密钥，如果私有密钥（*ECCKEY.txt*）发生更改，公共密钥（SE_ReadKey_Pub()）将自动修改。

*图 21*显示修改用于ECDSA非对称加密固件验证的私有密钥和公共密钥的几个步骤：

1. 在文件*ECCKEY.txt*中更改密钥值
2. 编译SECoreBin：执行*prebuild.bat*，生成*se_key.s*
3. 编译UserApp：执行*postbuild.bat*，加密UserApp

图21. 新私钥/公钥



5.3 STM32WB系列特殊性

对于STM32WB系列，AES加密密钥不是通过*prebuild.bat*脚本进行处理，而是配置在M0+内核中。*SECoreBin/readme.txt*中描述了配置过程。

5.4 KMS特殊性

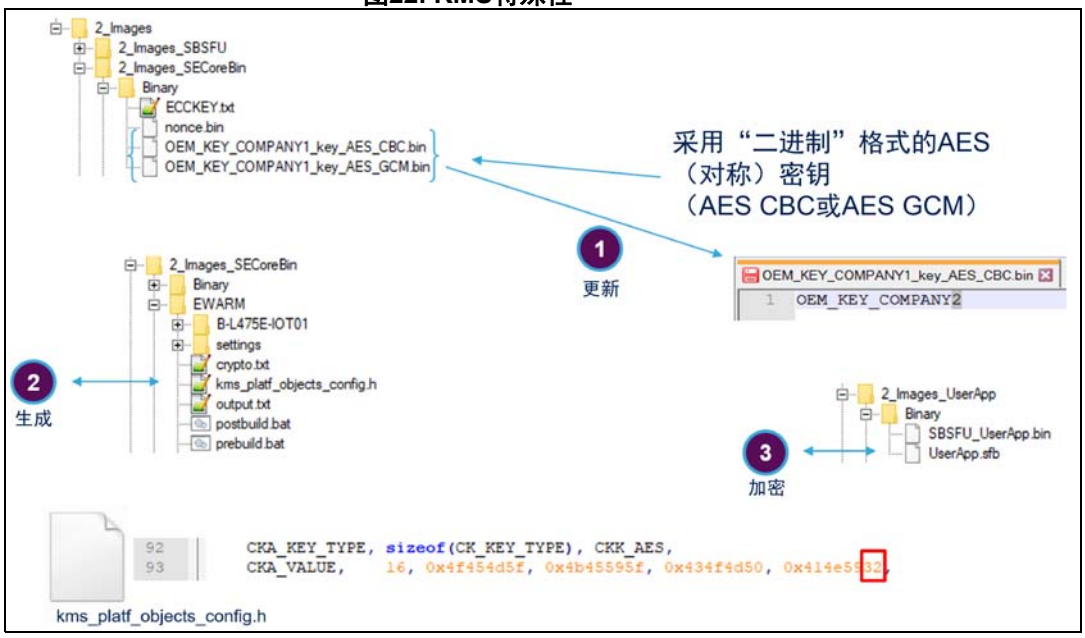
有了KMS中间件集成，SBSFU密钥不再存储在受到PCROP保护的扇区中。它们作为静态嵌入式密钥存储在KMS代码中。

*图 22*显示一个固件加密密钥修改的示例：

1. 在文件*OEM_KEY_COMPANY1_keys_AES_xxx.bin*中更改密钥值
2. 编译*SECoreBin*：执行*prebuild.bat*，生成*kms_platf_objects_config.h*
3. 编译*UserApp*：执行*postbuild.bat*，加密*UserApp*

相同的过程适用于固件ECDSA验证密钥、BLOB AES加密密钥和BLOB ECDSA验证密钥。

图22. KMS特殊性



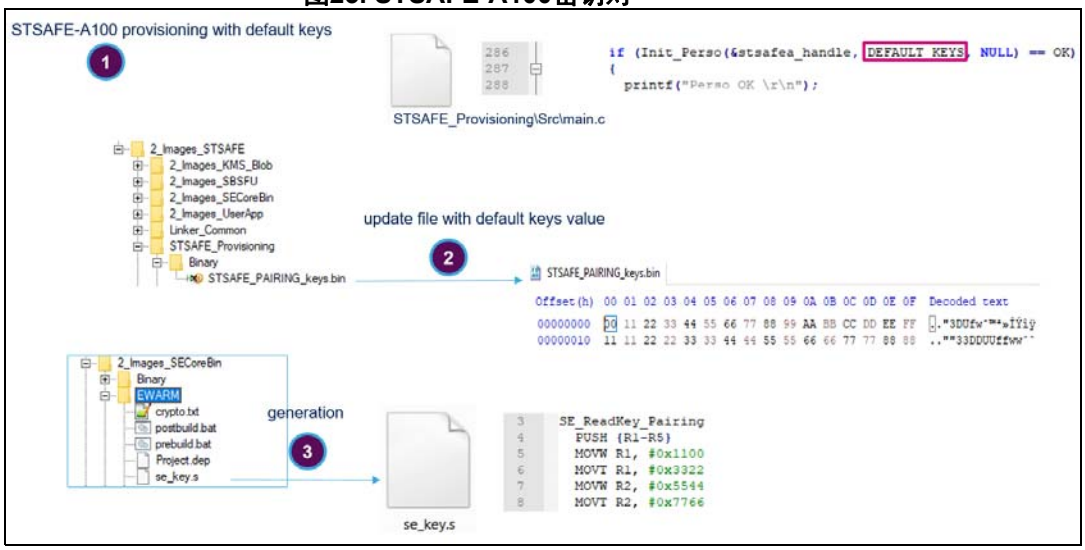
5.5 STSAFE-A100特殊性

正如在UM2262的附录G中所解释的，STM32和STSAFE-A100必须使用密钥对进行配置。*STSAFE_Provisioning/readme.txt*中描述了STSAFE-A100配置过程。

图xx显示了一个密钥对配置示例：

1. 采用默认密钥对配置STSAFE-A100
2. 相应地更新*STSAFE_PAIRING_keys.bin*
3. 编译*SECoreBin*：执行prebuild.bat，生成*se_key.s*。

图23. STSAFE-A100密钥对

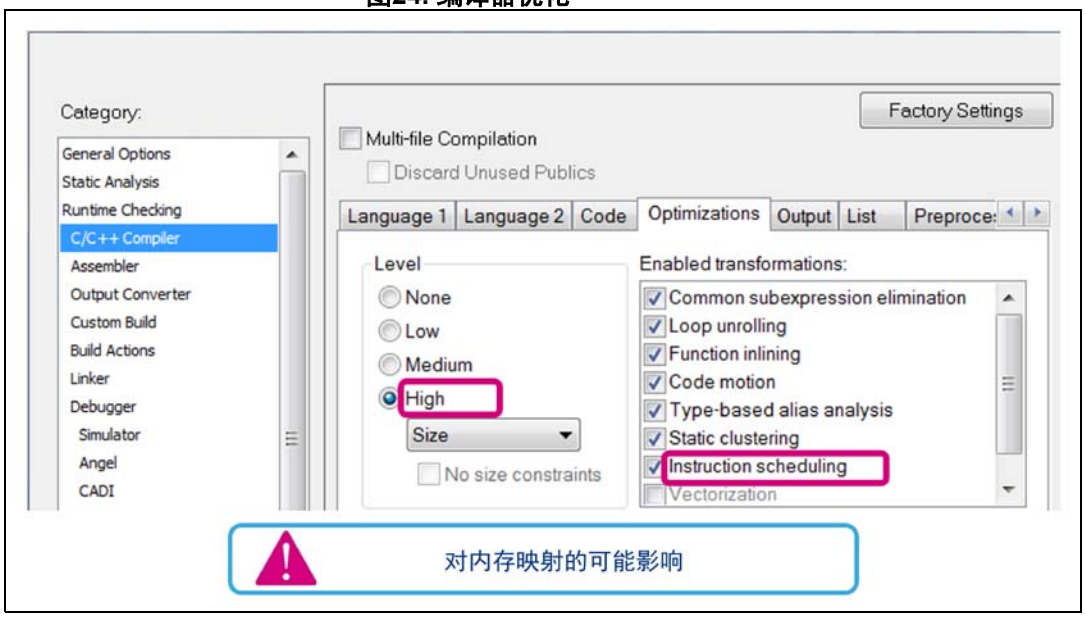


6 调试技巧

6.1 编译器优化级别

在交付项目时，针对大小方面进行了最高级别编译器优化。这样的优化会使调试变得复杂。更改编译器优化级别可能影响内存映射。

图24. 编译器优化



6.2 内存映射调整

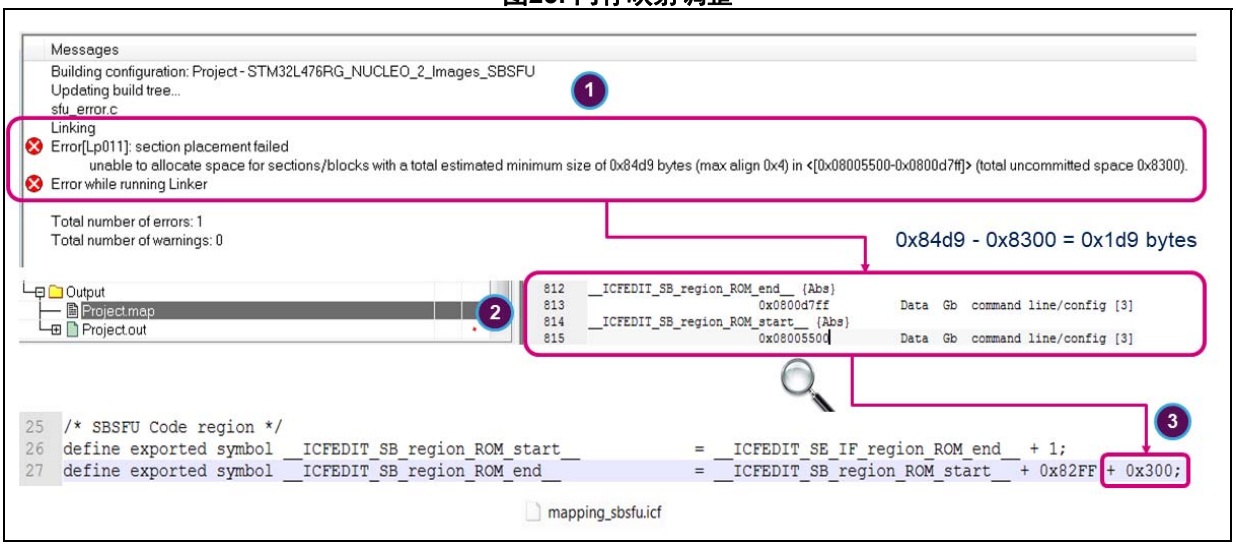
当更改编译器优化级别或使用verbose编译开关激活开发模式时，用户可能必须调整SBSFU内存映射，例如减少固件镜像插槽以避免重叠。

注意： 安全外设配置（RDP、WRP、PCROP、FWALL、安全用户内存（如果可用于系列））是基于SBSFU链接器符号自动计算的；但是MPU配置不一样，因为存在*第 3.2 节：内存映射定义*”详细说明的约束。暂时禁用MPU保护对于调试来说是一种有效的变通方法。

*图 25*根据一个示例描述内存调整的3个步骤：

1. 通过分析链接器消息来确定GAP：0x1d9字节
2. 通过咨询*project.map*文件确定相关区域：__ICFEDIT_SB_region_ROM_start__
3. 在文件*mapping_sbsfu.icf*中应用修改：0x300字节

图25. 内存映射调整



必须检查内存映射调整对安全外设配置的影响，尽管它是自动计算的。例如，使用STM32CubeProgrammer（STM32CubeProg）检查WRP配置，如*图 26*中所示。

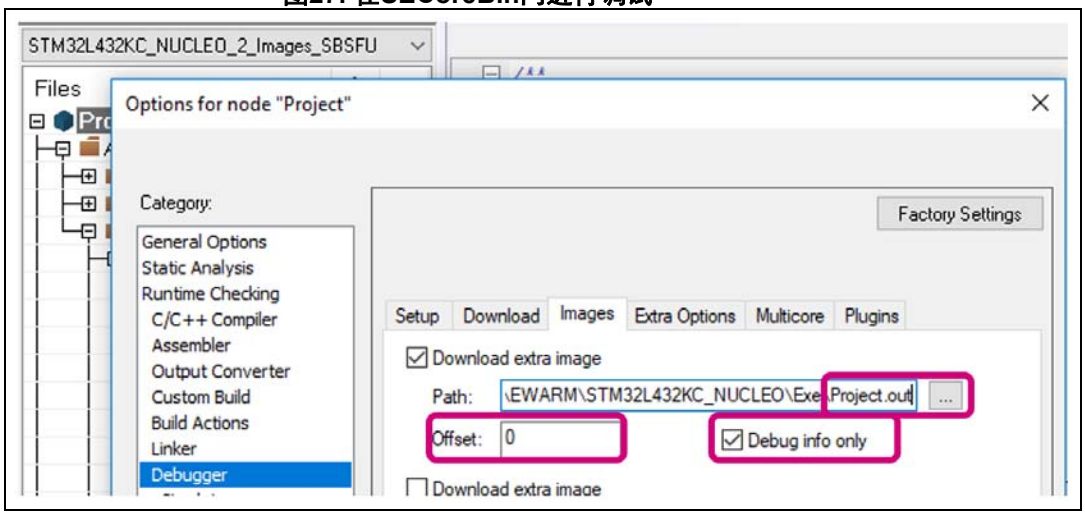
图26. 检查WRP保护

6.3 调试SECoreBin

为了在SECoreBin内部进行调试，必须更改SBSFU项目选项以加载SECoreBin符号。这是在调试器菜单中执行的，如*图 27*中所示：

- 浏览并选择文件*Project.out*
- 设置*偏移*为0
- 勾选*仅调试信息*复选框

图27. 在SECoreBin内进行调试



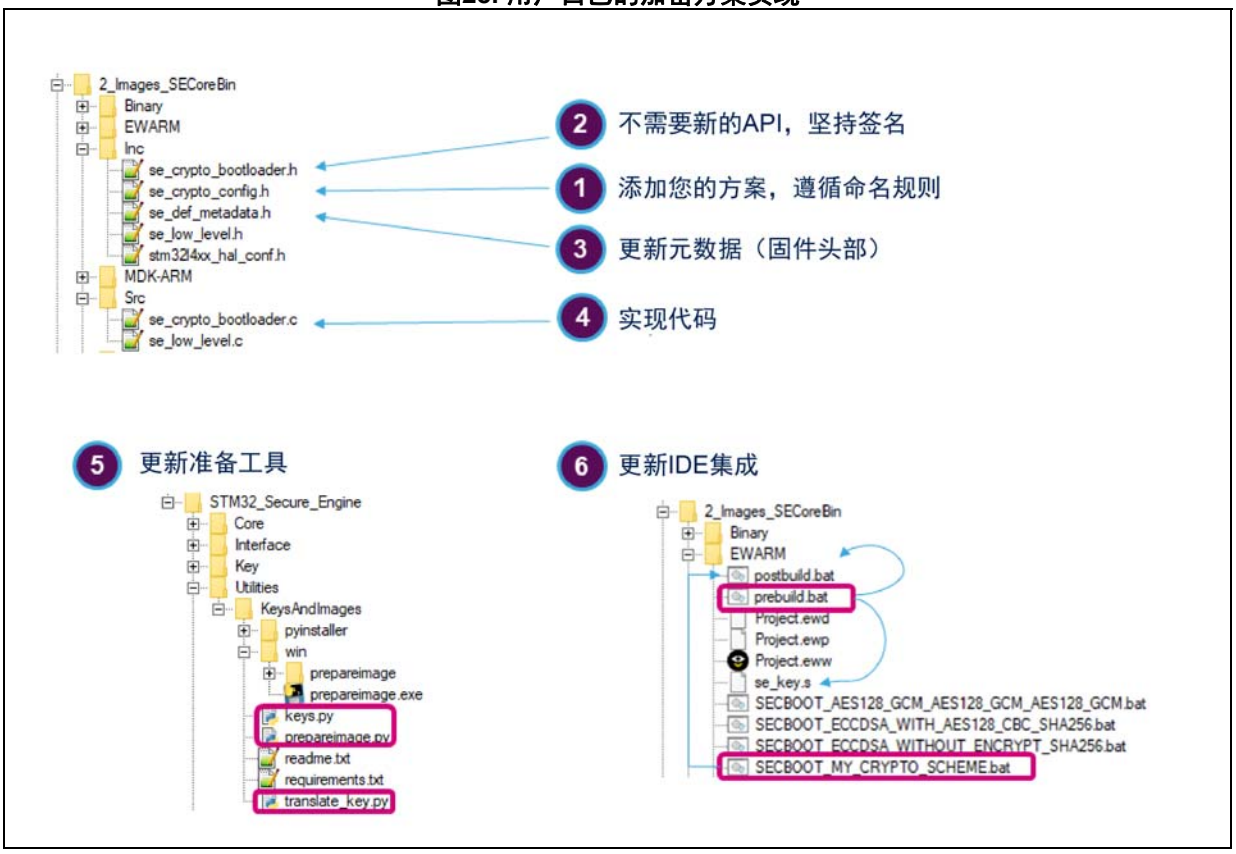
7 调整SBSFU

7.1 实现一种新的SBSFU加密方案

X-CUBE-SBSFU附带一些预定义的加密方案（请参见*第 4.2 节：加密方案选择第 18 页*）。还可以使用用户自己的加密方案来扩展软件包。

为了实现新的SBSFU加密方案，请遵循*图 28*中已说明并在下面描述的步骤。

图28. 用户自己的加密方案实现



更新在器件端运行的代码：

1. **步骤1**：为SECBOOT_CRYPTO_SCHEME定义新的值。
2. **步骤2**：仔细查看引导程序所需的API签名。加密服务必须具有相同的签名，以避免更新SBSFU代码。
3. **步骤3**：定义一个新的SE_FwRawHeaderTypeDef结构，并遵守约束，以保持兼容现有SBSFU代码。
4. **步骤4**：在*se_crypto_bootloader.c*中实现加密服务代码。

更新在主机端运行的工具，以准备密钥和固件镜像：

5. **步骤5**：更新准备工具以支持新的加密方案（*prepareimage.py*; *translate_key.py*; *keys.py*）。
6. **步骤6**：更新IDE集成以生成适当的密钥和固件镜像。
   - 需要一个新的批处理文件，以使用适当的命令调用准备工具；*prebuild.bat*复制该批处理文件，用于创建*postbuild.bat*。
   - 必须更新*prebuild.bat*以考虑到新的加密方案并生成适当的密钥和*postbuild.bat*。

7.2 优化内存映射

有几个选项可以减少SBSFU代码量，以便最大化用户应用程序插槽的大小。*表 3*总结了其中一些选项。

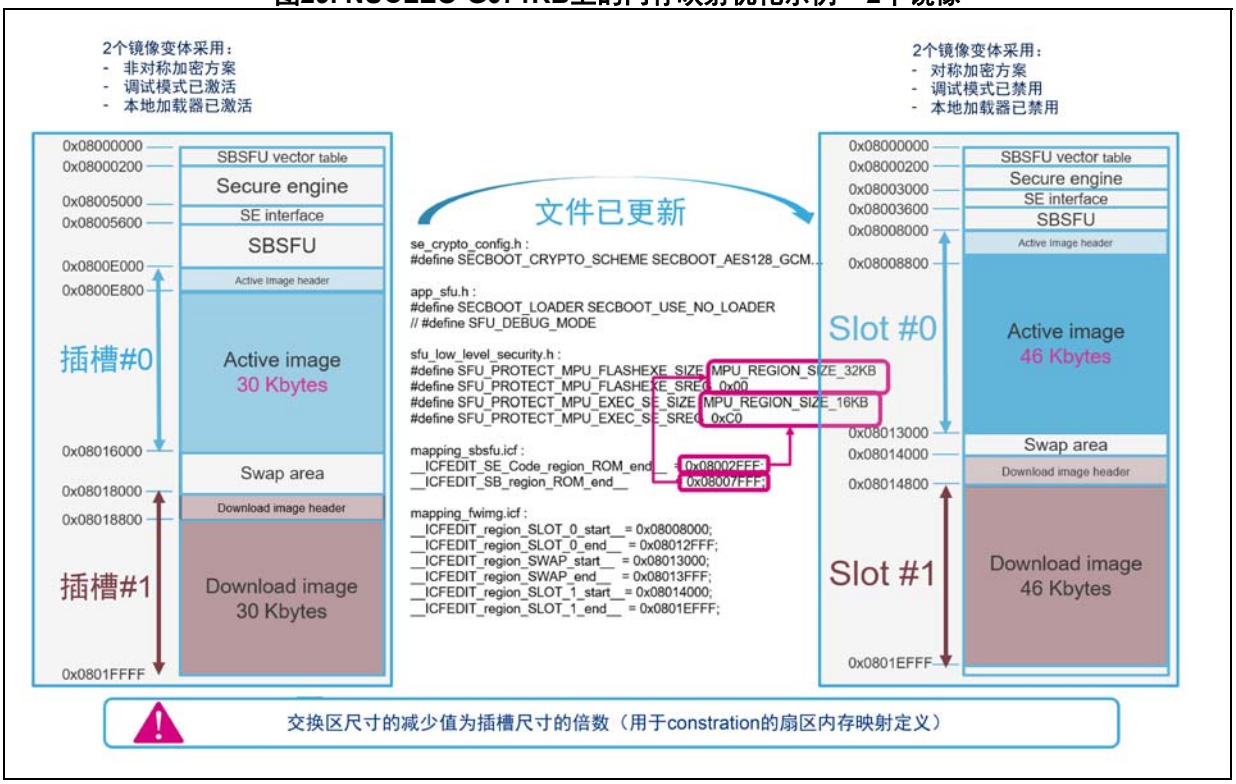
表3. 裁减SBSFU代码量

选项	描述 / 结果	增益
选择1-镜像变体	不再可能从用户应用程序下载新的固件镜像。	插槽的大小翻倍 vs. 2-镜像项目
选择AES-GCM对称加密方案	共享对称密钥机密存储在器件中。	~ 9 KB
禁用SFU_DEBUG_MODE	在SBSFU执行过程中，终端不再显示任何信息	~ 9 KB
禁用SECBOOT_USE_LOCAL_LOADER	SBSFU应用程序内不再有本地加载程序。它不兼容1-镜像变体。	~3 KB
实现硬件解密	选择集成加密硬件IP的STM32器件。	取决于用户的实现
如果在STM32上运行的所有代码都是完全可信且稳定的，那么您可以为STM32F4/F7/G0/G4/H7/L1移除基于MPU的安全引擎内部隔离	移除MPU区域的对齐约束。	最高可达12 KB，具体取决于产品

总增益取决于*第 3.2 节：内存映射定义第 9 页*中描述的映射约束。

作为示例，*图 29*突出强调了要进行的映射修改。从使用对称加密方案的两个镜像开始，SFFU_DEBUG_MODE 和SECBOOT_USE_LOCAL_LOADER开关被禁用，导致用户应用程序的大小增加了16 KB。

图29. NUCLEO-G071RB上的内存映射优化示例 – 2个镜像



在文件夹*NUCLEO-G031K8\Applications\1_Image*中，为NUCLEO-G031K8提供了另一个内存优化示例，在该板上可用的64 KB中，32 KB被分配给用户应用程序。

8 调整用户应用程序

8.1 如何使应用程序兼容SBSFU

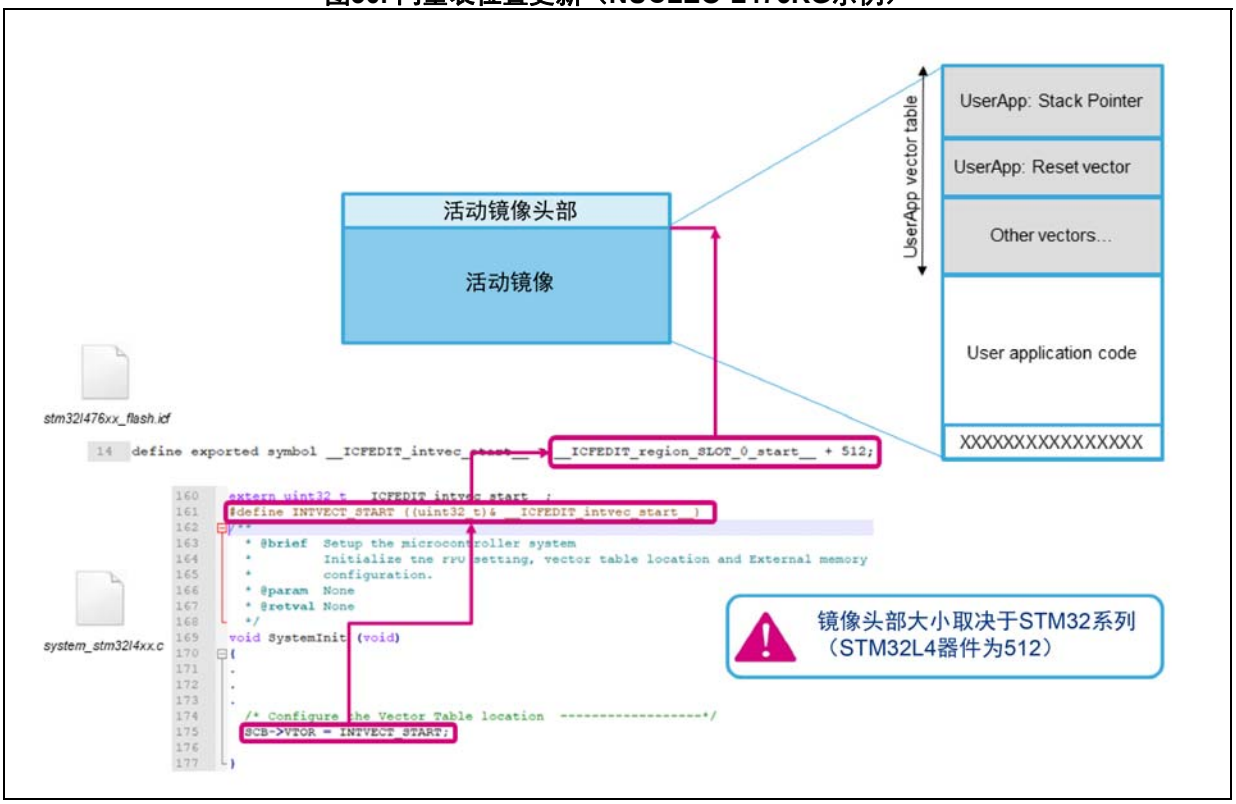
首先，必须修改用户应用程序的映射，以允许应用程序在插槽#0中运行：

- 由向量表启动的代码段必须配置为从插槽#0运行，就在镜像头部之后：__ICFEDIT_region_SLOT_0_start__ + 512 (SFU_IMG_OFFSET = STM32L4系列为512）
- 数据段必须在安全引擎保护区域之后启动：（__ICFEDIT_SE_region_SRAM1_end__ + 1）

请参考*第 3.2 节：内存映射定义第 9 页*获取更多关于内存约束的详细信息。

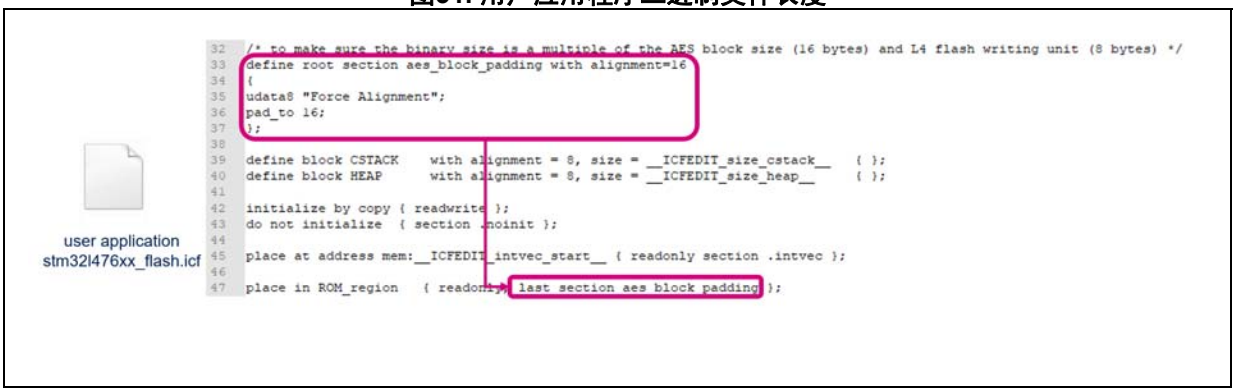
然后，在系统初始化时，必须将VTOR设置为向量表的新位置，如*图 30*中所示。

图30. 向量表位置更新（NUCLEO-L476RG示例）



对于用户应用程序加密，用户应用程序二进制文件长度必须是16字节的倍数。*图 31*演示如何更新链接器文件以验证此约束。

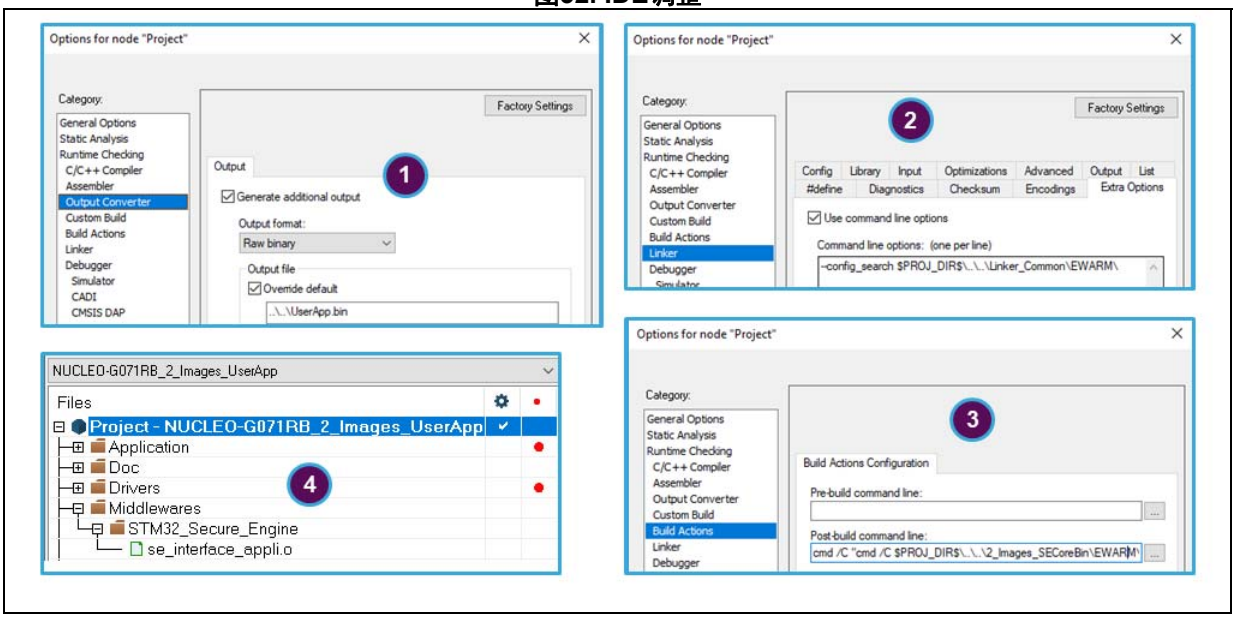
图31. 用户应用程序二进制文件长度



最后，如UserApp示例中所述，必须更新IDE配置，以便：

1. 生成*UserApp.bin*文件
2. 包括链接器通用文件的搜索路径
3. 调用*postbuilb.bat*以生成 *UserApp.sfb*和*SBFU_UserApp.bin*
4. 集成*se_interface_appli.o*，以访问安全引擎运行时服务（如果有）

图32. IDE调整



如UM2262中所述，还存在一些额外的约束，具体取决于STM32系列：

- STM32F4系列、STM32F7系列、以及STM32L1系列：基于MPU的安全引擎隔离完全依赖于这样一个事实 - 访问安全引擎服务需要软件执行的特权级别。用户应用程序必须考虑这种情况，并信任以特权模式运行的任何代码段。

- STM32G0系列、STM32G4系列、以及STM32H7系列：在安全状态下，任何对安全内存区域的访问（取、读、编程、擦除）都被拒绝，并生成一个总线错误。因此，用户应用程序没有可用的安全引擎运行时服务。

注： *IWDG在SBSFU执行期间启动。它必须在6秒间隔内刷新。*

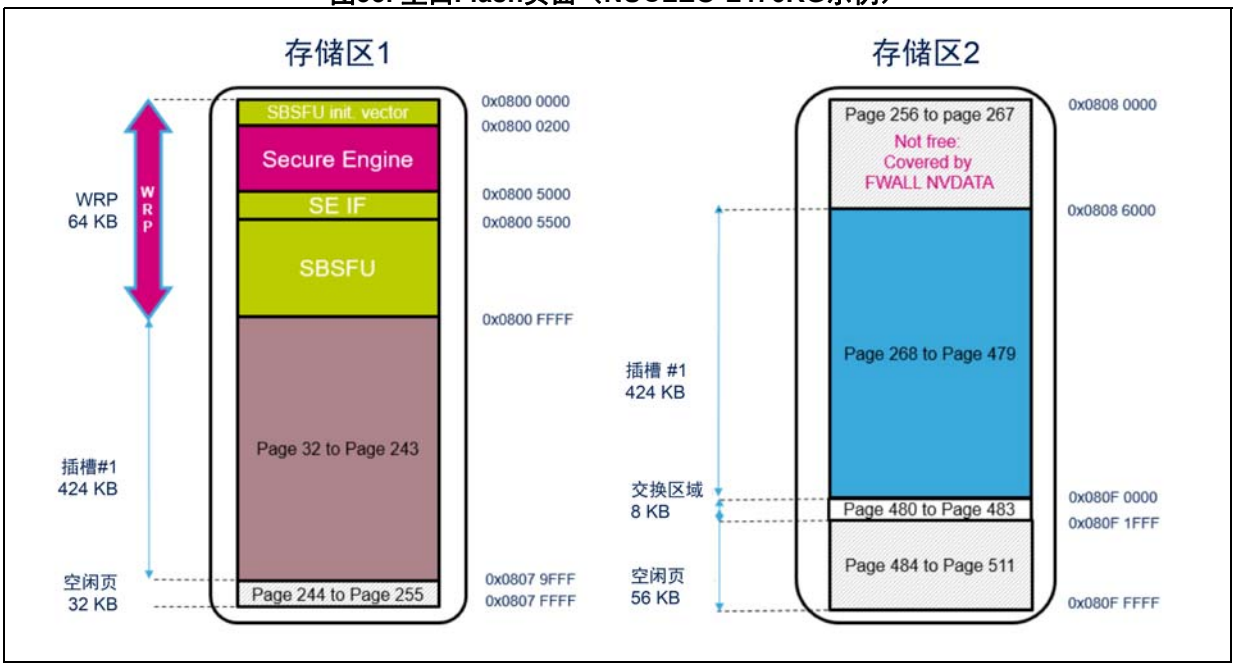
8.2 使用闪存存储用户数据

在闪存页面（或闪存扇区）存储用户数据可能有一些限制：

- 在SBSFU代码区域之外
- 不在镜像插槽（插槽 #0，插槽 #1）中
- 不在交换区域中

*图 33*提供一个基于NUCLEO-L476RG的内存映射示例，此处的闪存（从489页到511页）供用户存储数据、安装文件系统或模拟EEPROM。

图33. 空白Flash页面（NUCLEO-L476RG示例）

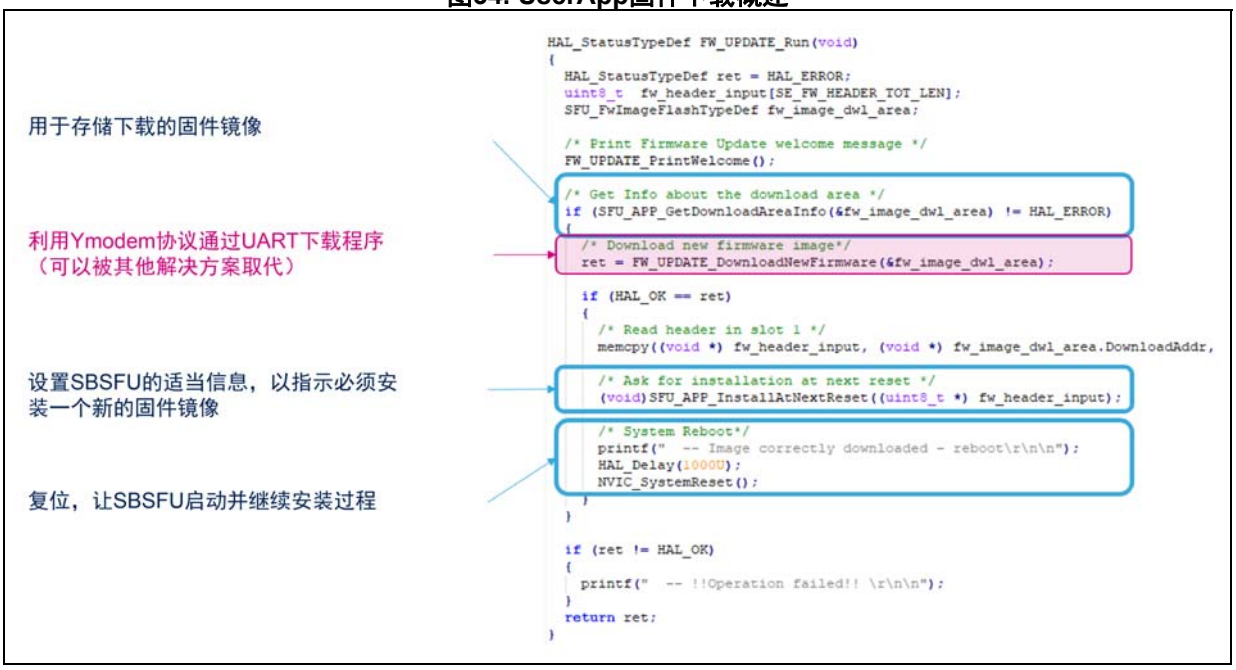


8.3 更改用户应用程序中的固件下载功能

只有在双镜像操作模式下才存在这种可能性。

X-CUBE-SBSFU UserApp项目中提供了一段基于YMODEM协议（通过UART）的示例代码。下载流程在文件*fw_update_app.c* 中，如*图 34*中所示。

图34. UserApp固件下载概述



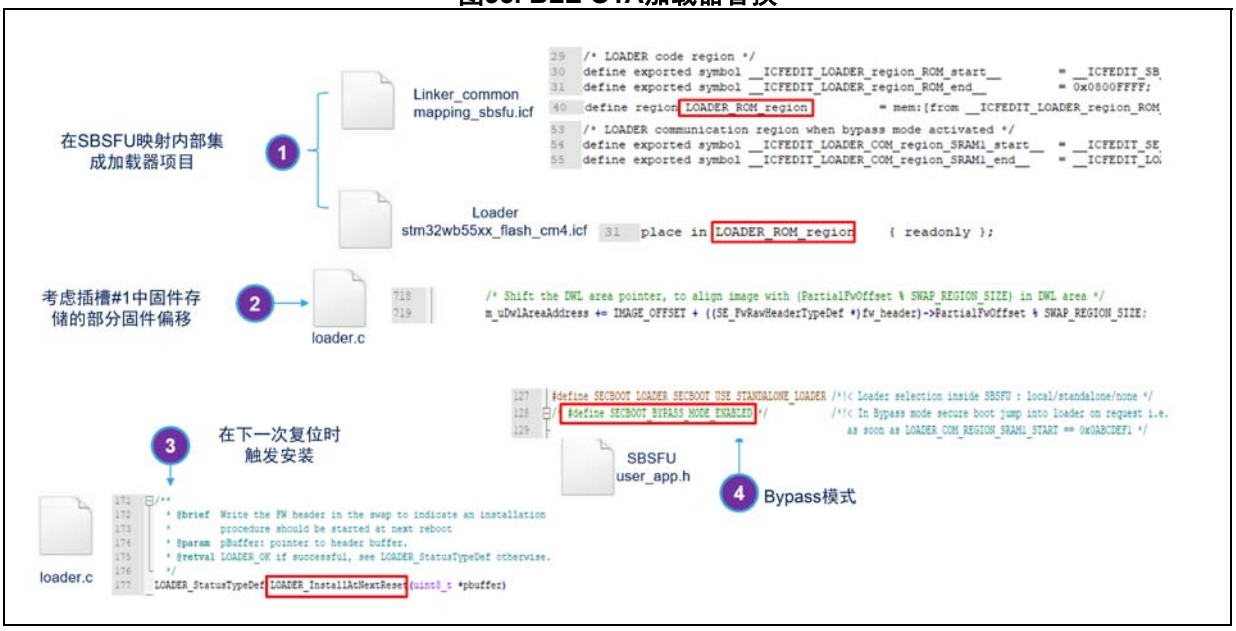
8.4 如何用BLE OTA加载器替换独立加载器

对于STM32WB系列，STM32WB cube软件包中提供一个BLE OTA加载器应用程序示例。

*图 35*显示替换独立加载器时要遵循的规则列表：

1. 在SBSFU公共映射定义内集成加载器项目
2. 下载的固件存储必须考虑部分镜像偏移
3. 新固件被下载后，在交换区写入头文件，然后在下一次复位时触发安装
4. 如果加载器被设计为通过M0+内核更新BLE堆栈，SECBOOT_BYPASS_MODE_ENABLED开关可以被激活。

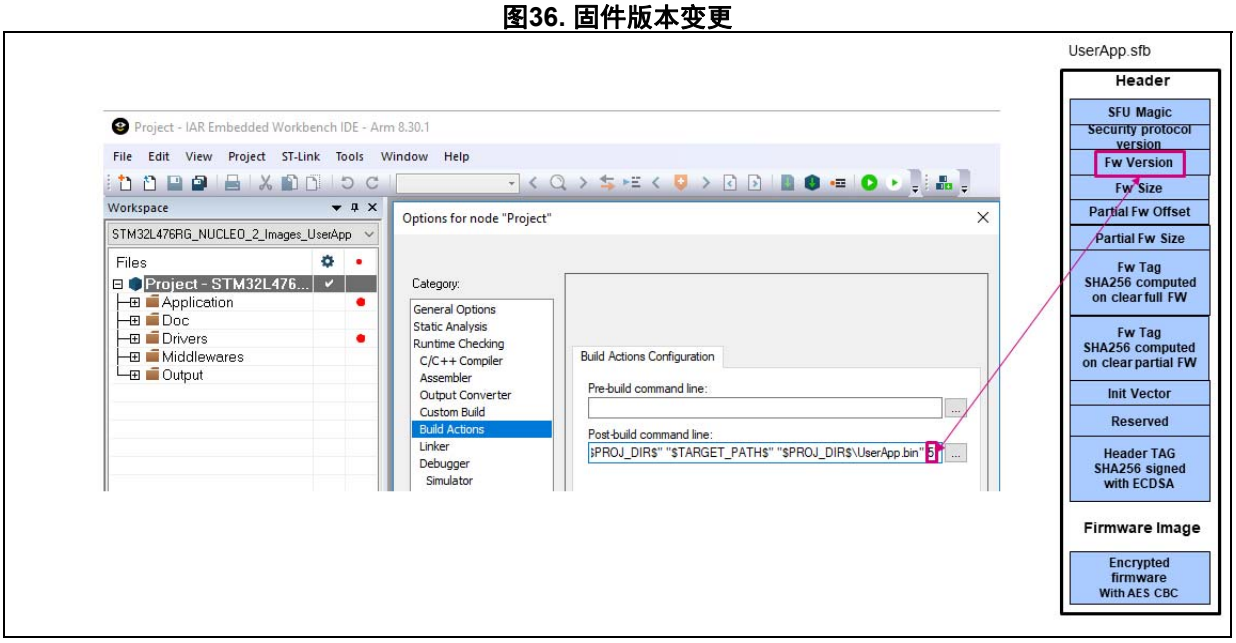
图35. BLE OTA加载器替换



8.5 如何变更固件版本

固件版本是用*postbuild.bat*脚本生成的固件头文件的一部分。在下面的示例中，版本为5。

图36. 固件版本变更



9 版本历史

表4. 文档版本历史

日期	版本	变更
2017年12月20日	1	初始版本。
2018年8月31日	2	文件结构和内容全部更新： – 重新关注*简介*中的集成主题 – 适应于非对称和对称加密方案 – 适应于单镜像和双镜像模式
2018年12月18日	3	产品范围扩展到STM32F4系列、 STM32F7系列和STM32G0系列： – 更新了*第1章：概述*、*第2章：相关文档*、*第3.2节：内存映射定义*、*第4.3节：安全配置*、*节：图15显示文件app_sfu.h中针对STM32WB系列提供的各种安全配置解决方案。*、以及*第8.1节: 如何使应用程序SBSFU兼容* – 添加了*第7章：调整SBSFU* 安全库提供扩展到mbedTLS： – 更新了*第4.1节：要配置的特性*
2019年9月6日	4	更新了*简介*。 产品范围扩展到STM32H7系列、 STM32G4系列、STM32L0系列、STM32L1系列、以及STM32WB系列。 更新了*第2章：相关文档*。 更新了*第3.1节：硬件调整* 更新了*第3.2节：内存映射定义* 修改了*第3.2.1节：SBSFU区域定义参数*和*第3.2.2节：固件镜像插槽定义参数* 更新了*第17页上的第4.1节* 更新了*第4.3章节：安全配置*（更新了图形并添加了*图18：STM32WB系列安全配置（app_sfu.h）* 在*第18页上的第4.2节*中添加了注释。 在*第4.4节：开发或生产模式配置*中修改了选项字节配置。 添加了*第5.3节：STM32WB系列特殊性*、*第5.4节：KMS特殊性*、以及*第5.5节：STSAFE-A100特殊性*。 在*第7.2节：优化内存映射*中更新了*表3* 添加了*第8.4节：如何用BLE OTA加载器替换独立加载器*和*第8.5节：如何变更固件版本*。

表4. 文档版本历史（续）

日期	版本	变更
12-Aug-2021	5	添加了OTFDEC信息到*第 4.1 节：要配置的特性*和*第 4.2 节：加密方案选择*中（添加一条注释） 更新了*第 3.2.2 节：固件镜像插槽定义参数*。 增加了*图 8：双存储区产品的防火墙配置约束*和*图 9：交换存储区之后的防火墙配置*。 更新了*图 11：特定于SBSFU的链接器文件*、*图 12：特定于UserApp的链接器文件（NUCLEO-L476RG示例）*和*图 13：SBSFU配置*。 更新了*第 4.4 节：开发或生产模式配置*、*第 6.2 节：内存映射调整*、*第 7.2 节：优化内存映射* 移除了*图28：NUCLEO-G031K8上的内存映射优化示例–1个镜像*

表5. 中文文档版本历史

日期	版本	变更
2021年8月8日	1	中文初始版本。

重要通知 - 请仔细阅读

意法半导体公司及其子公司 （“ST”）保留随时对 ST 产品和 / 或本文档进行变更、更正、增强、修改和改进的权利，恕不另行通知。买方在订货之前应获取关于 ST 产品的最新信息。 ST 产品的销售依照订单确认时的相关 ST 销售条款。

买方自行负责对 ST 产品的选择和使用， ST 概不承担与应用协助或买方产品设计相关的任何责任。

ST 不对任何知识产权进行任何明示或默示的授权或许可。

转售的 ST 产品如有不同于此处提供的信息的规定，将导致 ST 针对该产品授予的任何保证失效。

ST 和 ST 徽标是 ST 的商标。若需 ST 商标的更多信息，请参考 *www.st.com/trademarks*。所有其他产品或服务名称均为其各自所有者的财产。

本文档中的信息取代本文档所有早期版本中提供的信息。