STM32 微控制器安全简介

引言

本应用笔记介绍了 STM32 微控制器的安全基础知识。

微控制器中的安全性涵盖了几个方面，其中包括固件知识产权保护、设备私有数据保护以及服务执行保证。

在物联网的背景下，安全性更加重要。大量联网器件成为了攻击者的主要目标，一些远程攻击都通过器件通信通道的弱点对其进行攻击。对于物联网，信息安全的要求从机密性和身份认证扩展到了通信渠道，这些通信渠道往往要求加密。

本文档旨在通过对不同攻击类型采取对策来帮助构建安全系统。

在第一部分，在快速概述不同类型的威胁之后，会提供一些典型的攻击示例，以展示攻击者如何利用嵌入式系统中的不同弱点。

接下来的几节重点介绍了保护系统免受这些攻击的一套硬件和软件防护。

最后几节列出了 STM32 系列中可用的所有安全功能，并提供了用于构建安全系统的指南。

表 1. 适用产品

类型	产品系列
微控制器	STM32F0 系列、STM32F1 系列、STM32F2 系列、STM32F3 系列、STM32F4 系列、STM32F7 系列、STM32G0 系列、STM32G4 系列、STM32H7 系列、STM32L0 系列、STM32L1 系列、STM32L4 系列、STM32L4+系列、STM32L5 系列、STM32U5 系列、STM32WB 系列、STM32WL 系列

1 概述

下表提供了本文所用的缩略词及其意义的非详尽列表。

表 2. 词汇表

术语	定义
AES	高级加密标准
CCM	内核耦合存储器（SRAM）
CPU	中央处理单元 - 微控制器的核心
CSS	时钟安全系统
DoS	拒绝服务（攻击）
DPA	差分功率分析
ECC	错误代码校正
FIA	故障注入攻击
FIB	聚焦离子束
GTZC	全局 TrustZone®控制器
HDP	安全隐藏保护
HUK	硬件唯一密钥
IAP	应用内编程
IAT	初始认证令牌
IoT	物联网
IV	初始化向量（加密算法）
IWDG	独立看门狗
MAC	消息认证码
MCU	微控制器单元（基于 STM32 Arm® Cortex®-M 的器件）
MPCBB	基于存储器保护块的控制器
MPCWM	基于存储器保护水印的控制器
MPU	存储器保护单元
NSC	非安全可调用
NVM	非易失性存储器
OTFDEC	动态解密
OTP	一次性可编程
PCROP	专有代码读保护
PKA	公钥算法（又称 aka 非对称算法）
PSA	平台安全架构
PVD	可编程电压检测器
PWR	电源控制
ROM	只读存储器 – STM32 中的系统 Flash
RoT	可信根
RDP	读保护
RSS	根安全服务

术语	定义
RTC	实时时钟
SAU	安全归属单元
SB	安全启动
SCA	侧信道攻击
SDRAM	同步动态随机存取存储器
SFU	安全固件更新
SPA	简单功率分析
SPE	安全处理环境
SRAM	静态随机访问存储器（易失性）
SST	安全存储
SWD	串行线调试
TF-M	可信固件-M
WRP	写保护

参考文档

各器件的参考手册详细介绍了可用的安全特性以及存储器保护实现的信息。

每个 Arm® Cortex®版本均提供有编程手册，可用于说明 MPU（内存保护单元）：

- *STM32 Cortex®-M33 MCU 编程手册*（PM0264）
- *STM32F7 系列和 STM32H7 系列 Cortex®-M7 处理器编程手册*（PM0253）
- STM32 Cortex® *-M4 MCU 和 MPU 编程手册*(PM0214)
- *STM32F10xxx/20xxx/21xxx/L1xxxx Cortex®-M3 编程手册*（PM0056）
- *STM32L0、STM32G0、STM32WL 和 STM32WB 系列 Cortex®-M0+编程手册(PM0223)*

有关某些安全特性的详细说明，请参考以下用户手册和应用笔记（可从 *www.st.com* 获取）：

- 用户手册 UM1924“*STM32 加密库*”：介绍了 STM32 加密库的 API；随附 X-CUBE-CRYPTOLIB 扩展包。
- 用户手册 UM2262 “*X-CUBE-SBSFU STM32Cube 扩展包入门*”：介绍了 ST 的 SB（安全启动）和 SFU（安全固件更新）解决方案；随附 X-CUBE-SBSFU 扩展包。
- 应用笔记 AN4246、AN4701、AN4758、AN4968“*STM32xx 微控制器上专有代码读出保护*”：说明如何为 STM32L1、F4、L4 和 F7 系列分别设置和使用 PCROP 固件；随附 X-CUBE-PCROP 扩展包。
- 应用笔记 AN4838“*STM32 MCU 中内存保护单元（MPU）的管理*”：介绍了如何在 STM32 产品中管理 MPU。
- 应用笔记 AN5185“*STM32WB ST 固件升级服务*”

提示 *Arm 是 Arm Limited（或其子公司）在美国和/或其他地区的注册商标。*

2 概述

2.1 安全保护的目的

为何需要采取保护措施

微控制器中的安全性是指保护内置的固件、数据以及系统功能。而对于数据保护有需求的场合，尤其是密钥和个人数据最为重要。

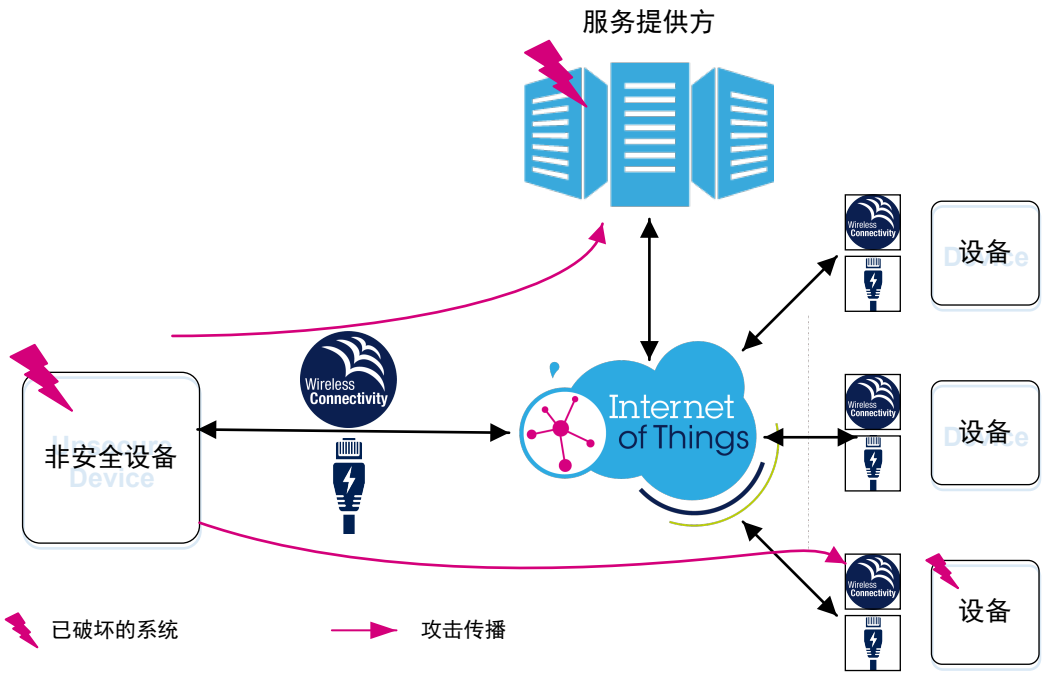
固件代码也是重要的资产。如果攻击者能够访问二进制代码，便可对程序进行逆向工程，尝试找到其它漏洞，绕过许可和软件限制。攻击者可以复制任何自定义算法，甚至可以使用它来烧写盗版硬件。即使是开源软件，也有必要证明代码的真实性，而且并未被恶意固件替代。

考虑到系统层面的保护，如环境，包含气体，火灾或侵蚀，检测报警或监控摄像头，拒绝服务攻击（DoS 攻击）是另一种主要威胁。系统功能须具有稳健、可靠的特性。

即使安全要求增加了系统的复杂性，也不得低估其地位。如今，越来越多的技术熟练的攻击者将基于微控制器构建的系统做为潜在目标，希望以此实现经济获益。这些收益可能会非常高，特别是在攻击可大规模传播（比如在 IoT 环境中）的情况下。即便系统无法达到完全安全，也可以提高攻击的成本。

事实上，IoT 或智能设备已提高了对安全的要求。互联设备可远程访问，因此对黑客来说极具吸引力。这个连接性本身可能为攻击提供了一个基于协议弱点的入口。一旦成功实施攻击后，一台被破解的设备会破坏整个网络的完整性（请参见下图）。

图 1. 已破坏的连接设备的威胁



什么需要被保护

安全不能局限于特定的目标或资产。如果代码二进制文件一旦被公开，则很难保护数据，对应攻击行为和保护机制很难被区分。但是对资产和风险进行汇总仍然非常有用。

下表介绍的是被攻击者当做目标的部分资产列表。

表 3. 要保护的资产

目标	资产	风险
数据	传感器数据（例如医疗数据或位置日志） 用户数据（例如 ID、PIN、密码或帐户） 事务日志 密钥	未经授权出售个人数据 非法使用 监视 勒索
控制器件（bootloader、恶意应用）	设备正常功能 设备/用户身份	拒绝服务 针对服务提供商的攻击 对服务进行欺诈性访问（云）
用户代码	设备硬件架构/设计 软件专利/架构 技术专利	器件假冒 软件假冒 软件篡改 访问安全区

弱点、威胁和攻击

保护机制需要处理不同的威胁。目的在于消除攻击中可能利用的弱点。第 3 节 攻击类型中概括介绍了主要攻击类型（从基本攻击到最高级的攻击）。

下面介绍有关安全的三要素：

- 资产：需要保护的内容
- 威胁：需要保护器件/用户免于的内容
- 弱点：保护机制中存在的漏洞或缺陷

总而言之，攻击是指一种利用系统漏洞访问资产的实现。

3 攻击类型

本节介绍微控制器可能面临的不同类型的攻击，从最基本的攻击到非常复杂且成本高昂的攻击。最后一部分举例介绍针对 IoT 系统发起的典型攻击。

对微控制器的攻击主要分为以下几类：

- 软件攻击：利用软件漏洞（例如错误或协议缺陷）。
- 硬件非侵入性攻击：侧重于 MCU 接口和环境信息。
- 硬件侵入性攻击：直接访问硅片的破坏性攻击

3.1 攻击类型介绍

安全的一条重要规则是攻击总是无处不在。

首先，没有任何保护措施能够完全避免意外攻击。无论采取哪种安全措施对系统进行保护，都会在器件使用过程中发现安全漏洞并加以利用。最后一点需要考虑如何更新设备固件以提高其安全性（请参考第 5.2.2 节 安全固件更新（SFU））。

其次，在配有正确设备的实验室条件下，可检索微控制器内容甚至是设计架构细节。这些技术在 第 3.3 节 硬件攻击 中进行了简短的介绍。

从攻击者的角度来看，如果期望收益/攻击成本的比率足够高，那么攻击便有利可图。收益取决于窃取的资产价值以及攻击的可重复性。成本则取决于成功实现攻击所花费的时间和金钱（设备成本），以及所获取的必要技能。

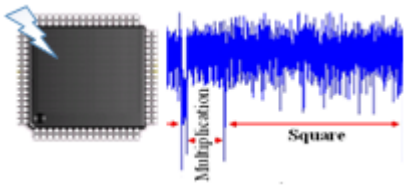
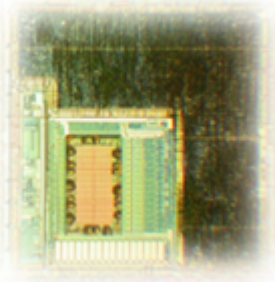
攻击类型

虽然还可按分组和类别对攻击作进一步细分，但其基本类别如下：

- 软件攻击利用漏洞，协议弱点或不可信的代码片段来执行。对通信渠道的攻击（拦截或篡夺）属于此类。软件攻击是最主要的攻击类型。它们的成本可能非常低。它们可以广泛传播，反复进行，破坏力巨大。无需物理访问器件，即可远程执行攻击。
- 硬件攻击需要对设备进行物理访问。最典型的硬件攻击是利用未受保护的调试端口实施攻击。但一般而言，硬件攻击颇为复杂，成本可能非常高昂。进行攻击时需要使用特定的材料，而且对电子工程技能有一定的要求。在板级或芯片级实施的非侵入性攻击不会破坏器件，而在器件硅片级实施的侵入性攻击则会破坏器件封装。在大多数情况下，通过此手段找到一种可用于广泛远程攻击的方法和信息，此类攻击手段才是可利可图的。

下表概括介绍了每种攻击类型的成本及使用的技术。

表 4. 攻击类型和成本

攻击类型	软件	硬件非侵入性	硬件侵入性
-			
范围	远程或本地	本地板级和器件级	本地器件级
技术	软件 bug 协议缺陷 木马 窃听	调试端口 电源噪声攻击 故障注入 侧信道分析	探测 激光 FIB 逆向工程
成本/专业知识	范围很大，具体视目标安全故障而定	成本极低。只需要中等复杂的设备和知识便可实施。	非常昂贵。需要专业设备和非常专业的技术。
目标	访问机密资产（代码和数据）。 非法使用 拒绝服务	访问机密数据或器件内部特性（算法）。	对设备进行逆向工程（硅片知识产权） 访问隐藏的硬件和软件机密（Flash 访问）

3.2 软件攻击

软件攻击会让 CPU 执行一段代码（恶意程序），从而对系统发起攻击。恶意软件的目的是控制器件，以获取对任何系统资源（例如 ID、RAM 和 Flash 内容或外设寄存器）的访问权限或修改其功能。

这类攻击能够成为最常见的器件威胁的原因有以下几点：

- 此类攻击除了个人计算机外不需要其它特殊设备，因此攻击成本较低。
- 很多黑客可以合作，分享各自的专业知识和技巧，如果存在安全漏洞，这类攻击很容易成功。此外，如果攻击成功，攻击协议可通过网络快速扩散

举例来说，恶意程序可以被注入到器件中或者可能已存在于主应用固件中（内部威胁：主应用使用了未经验证或者不值得信赖的库）。

恶意软件包括很多类型，可能很小且易于隐藏。

下文举例介绍了恶意软件的用途：

- 修改器件配置（如选项字节或存储器属性）。
- 禁用保护。
- 读取存储器并转储其内容，以实现固件和数据复制。
- 跟踪或记录器件数据。
- 访问加密项。

- 打开信道/接口。
- 修改或阻止器件功能。

除非用户应用完全可信、不存在 bug 且与外界隔绝，无法通过任何方式与外界通信，否则必须考虑软件攻击。

恶意软件注入

可通过各种方法将一段代码注入系统中。恶意软件的大小取决于目标，但可能会非常小（只有几十字节）。要执行恶意软件，必须将其注入到器件内存（RAM 或 Flash）。恶意软件被注入后，面临的挑战就是通过 CPU 运行恶意软件，也就是 PC（程序计数器）须跳转到恶意软件执行。

恶意软件的注入方法可以分为以下几类：

- 基本器件访问/“开门”：
  - 调试端口：JTAG 或 SWD 接口
  - Bootloader：如果可访问，可通过它的任何可用接口来读/写存储器内容。
  - 从外部存储器执行

  简单的硬件机制很容易碰到这些恶意软件注入，将在第 4 节 器件保护中加以介绍。
- 应用下载：
  - 固件更新过程：可能会传输恶意软件，而不是新固件。
  - 能够下载新应用的操作系统。

  此类对策基于器件与服务器之间的身份验证或直接采用代码身份验证。身份验证依赖于加密算法。
- 利用通信端口漏洞和错误：
  - 执行数据。有时可能会将恶意软件作为数据潜入其中，并利用不正确的边界检查执行恶意软件。
  - 基于栈的缓冲区溢出、基于堆的缓冲区溢出、跳转到 libc 攻击以及仅数据攻击

  这第三类显然难以避免。大多数嵌入式系统应用均采用 C/C++等低级语言进行编码。由于这些语言会导致的存储器管理错误可能被攻击者所利用（栈/堆/缓冲区溢出），因此这些语言被视为不安全语言。一般的观点是最大限度地减少不受信任或未验证的固件部分，从而尽可能减小攻击受面。一种解决方案是隔离不同进程的执行和资源。例如，TF-M 包括这种机制。
- 使用携带后门的不可信任的库

  最后一类是有意引入恶意软件，帮助破坏器件。现在，很多固件开发都依赖于网络上共享的软件，复杂的软件可隐藏木马病毒。在前一类中，应对这种威胁的方法是尽可能隔离进程执行并保护重要代码和数据，以减少攻击受面。

暴力破解

这种类型的攻击针对基于共享秘密的身份验证。安全器件在访问服务之前可能需要进行会话身份验证（比如在云中），可利用人机界面（HMI）和自动进程连续尝试获取密码。

下面列出了一些有趣的对策：

- 使用单调计数器（通过定时器实现，或者在可能的情况下使用备份域实现）限制登录尝试次数。
- 延长两次连续登录尝试之间的延迟。
- 增加询问-响应的机制，打断自动尝试。

3.3 硬件攻击

硬件攻击需要对器件或通常并行多个器件进行物理访问。

两种类型的攻击在成本、时间以及必备的专业知识方面有所不同：

- 非侵入性攻击只需要对器件进行外部访问（板级攻击），实施成本不会很高（设备成本为数万美元）。
- 侵入性攻击可直接访问器件芯片（拆包后）。此类攻击使用专业实验室中配备的先进设备进行。此类设备成本高昂（超过 10 万美元，通常在数百万美元范围内），目标是非常有价值的数据（密钥或 ID）甚至技术专利。

通用型微控制器并不是应对最高级物理攻击的最佳候选产品。如果需要达到最高级别的保护，建议将安全元件与通用型微控制器配合使用。安全元件是通过特定硬件最新安全标准认证的专用微控制器。

请参考意法半导体安全微控制器网页（www.st.com/en/secure-mcus.html）。

3.3.1 非侵入性攻击

非侵入性攻击或板级攻击会试图绕过保护而不会造成物理损坏（器件功能保持正常）。仅使用可访问的接口和器件环境。这些攻击要求采用适当的精密设备和工程技术（例如信号处理）。

调试端口访问

这是可对器件执行的最基本的攻击。禁用调试功能必须是需要考虑的第一级保护。的确，通过 JTAG 或 SWD 协议访问调试端口或扫描链允许访问器件的全部内部资源：CPU 寄存器、嵌入式 Flash、RAM 和外设寄存器。

对策：

- 通过禁用或熔断调试端口 读保护 (RDP)

串行端口访问

通信端口（如 I2C,SPI）的访问隐藏了被攻击者利用的漏洞。通信端口可作为器件入口点进行监视或使用。根据相关协议的实现方式（例如存储器地址范围、目标外设或读/写操作），攻击者可能获得对器件资源的访问权。

对策：

- 软件：
  - 相关协议操作必须限制在固件级执行，以免读取或写入敏感资源。
  - 将通信栈与敏感数据隔离。
  - 必须检查数据传输长度，以免缓冲区溢出。
  - 器件与目标之间的通信可使用共享密钥加密。
- 硬件：
  - 物理通信接口可嵌入多层板内，从而使其更难以访问。
  - 须禁用未使用的接口。

故障注入：时钟和电源干扰/故障攻击

故障注入在于在数据手册限定的参数范围之外使用器件，从而在系统中产生故障。一次成功的攻击可以以不同方式修改程序行为，例如破坏程序状态、破坏存储器内容、停止进程执行（“卡死故障”）、跳过指令、修改条件跳移或提供未授权访问。

典型的威胁包括篡改时钟（冻结或故障）和电源（欠压/过压或故障）。由于故障可能是无意的，因此故障对策与实现安全保护的对策相同，即冗余、错误检测和监控。

对策：

- 软件：
  - 检查函数返回值。
  - 分支时进行严格比较。
  - 用质数增加每个分支中的专用变量，确保关键部分没有跳过任何代码并检查期望值。
  - 采用特殊值用作“真”和“假”（避免与 0 或-1 比较，尝试使用高汉明距离的复杂值）。
- 硬件：
  - 使用时钟安全系统(CSS)（若提供）。
  - 使用内部时钟源。
  - 使用内部调压器。
  - 使用存储器错误检测（ECC 和奇偶校验）。

侧信道攻击（SCA）

执行固件时，攻击者可以观察器件的运行特征（如功耗、电磁辐射、温度或活动时间）。这种观察可以带来足够的信息来检索秘密资产，比如数据值和/或算法实现。基于侧信道的攻击对加密器件很有效，可破解出系统使用的密钥。SPA（简单功率分析）和 DPA（差分功率分析）是利用功耗进行侧信道攻击的典型示例。

对策：

- 软件：
  - 限制密钥使用：尽可能使用会话随机密钥。
  - 使用对行为进行随机化处理的受保护加密库（例如延迟或伪指令）。
- 硬件：
  - 安全元件（STSAFE）中可采用监控屏蔽，但通常不在通用微控制器中嵌入高效的硬件对策（除了 STM32U5 系列中的 SAES 示例）。

3.3.2 硅片侵入性攻击

此类攻击的成本非常高；在此过程中，会考虑通过一切方式从已被破坏的器件中提取信息。攻击者需要获取大量设备才能成功。此类攻击通过专业实验室中配备的昂贵设备实施，对技能、知识和时间都有较高的要求。

侵入性攻击最开始会拆除器件封装。如果不拆除保护层，也可以执行初步分析，但深入研究器件交互（探测）则需要拆除保护层。可通过化学蚀刻、钻孔或激光切割机拆除封装。一旦器件被打开，便可执行探测或修改攻击。

多个专用于提供安全保护的 ST 微控制器可提供对抗此类处理的稳健性。这些不属于 STM32 系列，不在本文的讨论范围之内。请参考 ST 安全硬件平台（www.st.com/en/secure-mcus.html）。

逆向工程

逆向工程的目的是了解器件的内部构造并分析其功能。对于具有数百万个栅极的器件来说，这是一项极具挑战性的任务。

第一步是创建微控制器映射。要完成这一步骤，可使用光学显微镜生成器件表面的高分辨率相片。通过蚀刻处理剥掉器件的金属层后，可在第二步中对更深层进行分析。

读取数据

使用电子显微镜，可以看到由电荷表示的数据。可以读取整个器件的存储器。

微探测和内部故障注入

微探测是指在金属层级与器件进行交互。会使用薄型电极与器件表面直接建立电气接触，以便攻击者能够在器件运行时对其进行观察、操作和干扰。

器件修改

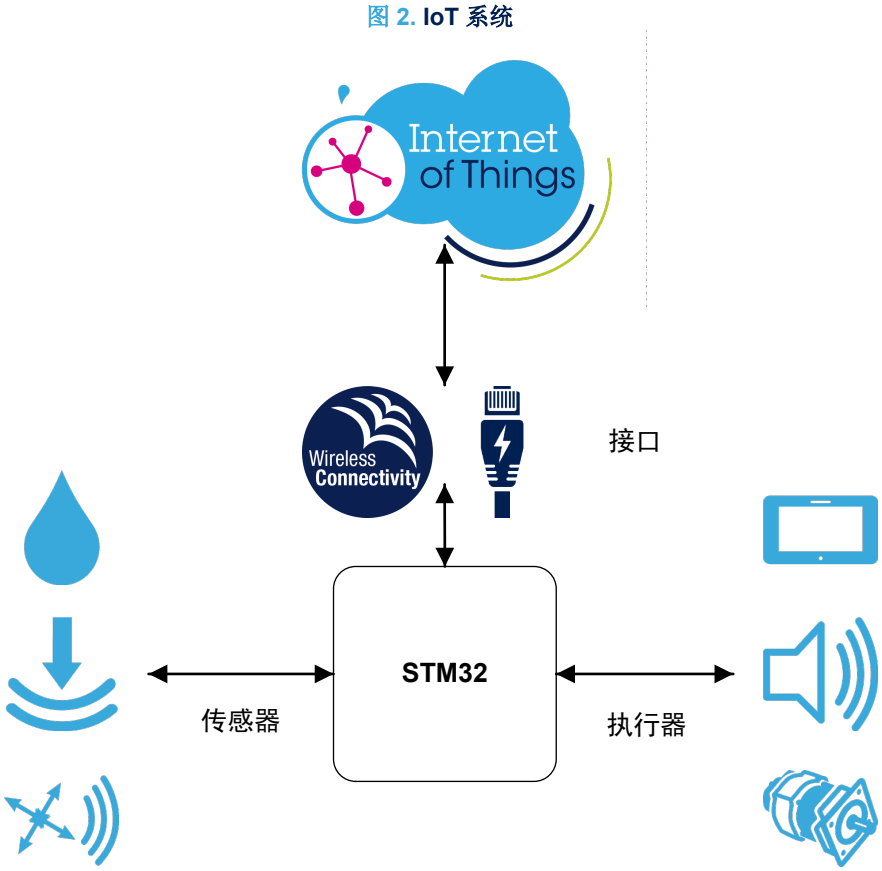
可以使用更加复杂的工具执行攻击。例如，聚焦离子束（FIB）工作站会简化深层金属和多晶硅线路的手动探测。通过切割或创建新的互联线路甚至是新的晶体管，它们还可用于修改器件结构。

3.4 IoT 系统攻击示例

最后一节举例介绍针对 IoT 系统发起的典型攻击。幸运的是，大部分攻击均可通过启用安全特性（硬件对策）和安全应用架构（软件对策）加以应对。相应对策会在后续章节中详细介绍。

IoT 是围绕着 STM32 微控制器以及连接系统（以太网、WiFi®、低功耗蓝牙®、LoRa®等）以及传感器和/或执行器构建的（请参见下图）。微控制器负责处理应用、数据获取以及与云服务的通信。微控制器还负责通过固件更新和完整性检查进行系统维护。

图 2. IoT 系统



3.5 攻击目标清单

以下几节列出了可能的攻击目标：

初始配置

安全链的信任根的加密数据需要以受控可信的方式注入到 SoC。无论是密钥、证书还是哈希初始值，都必须保持不变和/或机密。一旦在器件中完成编程，对应的数据保护机制则也须相应启动。随后只有授权进程可访问它。

- *风险*：固件损坏或篡改
- *对策*：
  - 使用可信的生产环境
  - 使用安全数据配置服务（SFI）
  - 数据保护机制
  - 安全应用隔离
  - 使用 OTP 存储器

启动修改

此类攻击的目的是使用 bootloader 接口访问器件内容。攻击的目的是修改启动模式和/或启动地址，以抢占用户应用，并通过 bootloader（通过 USB DFU、I2C 或 SPI）、调试端口或通过注入在 RAM 中的固件控制 CPU。启动模式和启动地址由器件配置或者输入引脚控制，须对其进行保护。

- *风险*：可完全访问微控制器内容

- *对策*：
  - 唯一启动入口
  - 禁用 bootloader 和调试（请参见读保护 (RDP)）

安全启动（SB）或可信固件-M（TF-M）

稳健的系统在启动主应用程序之前会检查其完整性和真实性。作为器件的信任根，这部分用户固件不得更改且不可绕过。

要成功发动攻击，需要绕过验证以及直接跳转到恶意软件执行不受信任的应用。此类攻击可通过故障注入等硬件技术实现，也可通过将预期的哈希值替换为恶意软件的哈希值的方式实现（参照本章开头的*初始配置*一节）。

- *风险*：器件欺骗或应用修改
- *对策*：
  - 唯一启动入口可避免绕过验证
  - “不可变代码”可避免 SB 代码被修改
  - 安全存储固件签名和/或标签值
  - 环境事件检测（电源故障、温度或时钟速度等）

固件更新

固件更新过程允许产品所有者更新固件版本，以确保在器件使用期间提供最佳的用户体验。但在固件更新过程中，攻击者可利用这一机会在器件中加入自己的固件或已有固件的已损坏版本。

固件更新过程必须通过固件身份验证和完整性验证进行保护。成功的攻击需要访问加密过程和密钥（参照本章开头的*初始配置*一节）。

- *风险*：器件固件损坏
- *对策*：采用身份验证和完整性检查的 SFU 应用。除了对固件进行签名之外，还可对固件加密，以提高其保密性。

通信接口

bootloader 或应用使用串行接口（例如 SPI、I2C 或 USART）与器件交换数据和/或命令。攻击者通过拦截通信可以使用该接口作为器件入口点。此外，固件协议也可能存在 bug（如溢出）。

- *风险*：访问器件内容
- *对策*：
  - 使物理总线在板上难以找到。
  - 隔离软件通信栈，避免其访问重要数据和操作。
  - 加密通信。
  - 禁用不需要使用的接口。
  - 仔细检查输入

调试端口

调试端口可用于访问器件的全部内容：内核和外设寄存器、Flash 和 SRAM 内容。用于开发应用时，将其保持激活状态可用于研究今后可能出现的问题。这是攻击者尝试对器件进行物理访问的第一个突破口。

- *风险*：完全访问器件
- *对策*：禁用器件调试特性（请参考读保护 (RDP)功能）。

外部外设访问

IoT 设备根据全局应用控制传感器和执行器。攻击者可通过修改传感器发出的数据或分流发送到执行器的输出数据的方式篡改系统。

- *风险*：不正确的系统行为。
- *对策*：利用防篡改功能检测板级系统入侵

敏感固件和数据

固件的某些部分需要进行特殊保护：例如加密算法或第三方库。此外，如果所选数据被视为重要资产（密钥），可能需要对其加强保护。

须对内存内容进行保护，以免受到外部访问（例如通信接口）和内部访问（其它软件进程）。存储器属性和防火墙是针对进程和数据隔离的主要保护措施。

- *风险*：固件拷贝或数据窃取
- *对策*：
  - 仅执行访问权限（XO）。
  - 防火墙
  - 存储器保护单元
  - 安全区域
  - 外部存储器的加密

SRAM

SRAM 属于器件运行的存储器。它嵌入了运行时缓冲区和变量（例如栈或堆），并可嵌入固件和密钥。在非易失性存储器中时，保密信息可以在加密后存储，当加载到 SRAM 时，需要以平面图的形式显示以供使用。同时，SRAM 通常保留通信缓冲区。出于这两个原因，攻击者可能会将攻击的重点放在 SRAM 上。攻击者至少可针对该存储器发起三种类型的攻击：代码（恶意软件）注入、通过缓冲区溢出破坏存储器以及通过临时存储的变量获取保密数据。

- *风险*：缓冲区溢出、数据窃取或器件控制
- *对策*：
  - 防火墙
  - 存储器保护单元
  - 安全区域

随机数生成

随机数通常用于会话密钥、加密随机数或初始化向量（IV）生成的加密过程。弱随机数发生器可能会使任何安全协议易受攻击。

软件攻击将利用随机序列的隐藏周期性或结构来猜测密钥并破坏通信保密性。硬件攻击试图禁用 RNG 或削弱输出的统计随机性。

稳健的随机数发生器依赖于熵源（模拟）的质量。

- *风险*：降低加密协议的安全性
- *对策*：
  - 使用硬件真随机数发生器
  - 对 RNG 输出进行测试，核验产生的随机数的统计属性。

通信堆栈

连接协议（例如蓝牙、以太网、Wi-Fi 或 LoRa）具有复杂的通信固件栈。这些栈通常以开源形式提供，不得始终将其视为可信栈。某一潜在的漏洞可能被大量利用。

- *风险*：通过网络访问器件（内容、控制）
- *对策*：
  - 通信进程隔离
  - 服务器身份验证
  - 安全更新固件以修复错误

通信窃听

设备与 IoT 服务之间的数据交换可直接被兼容的射频设备窃听或通过网络窃听。黑客的目的可能是检索数据、获取设备 ID 或访问服务器。

所有通信协议都可以采用加密的形式。通常会考虑使用多个加密步骤来保护不同层（设备、网关、应用）之间的通信。

- *风险*：监听和网络欺骗。
- *对策*：使用加密版通信栈（如以太网的TLS）

4 器件保护

本节描述的这些安全保护通过硬件机制控制。设置方法为通过选项字节配置器件、或通过硬件组件动态设置：

- 存储器保护：主要的安全特性，用于保护代码和数据免受内部和外部的攻击
- 软件隔离：用于避免内部攻击的进程间保护
- 接口保护：用于保护器件入口点，比如串行或调试端口
- 系统监控：检测器件外部入侵尝试或异常行为

4.1 用于 Armv8-M 架构的 TrustZone®

基于 Armv6 或 Armv7 架构（Cortex-M0、M3、M4 和 M7）的微控制器主要依靠实施软件来隔离固件和资源。这些机制将在本文档的后面部分进行介绍，它们具有稳健性，灵活性不足，无法同时执行安全固件和非安全固件。

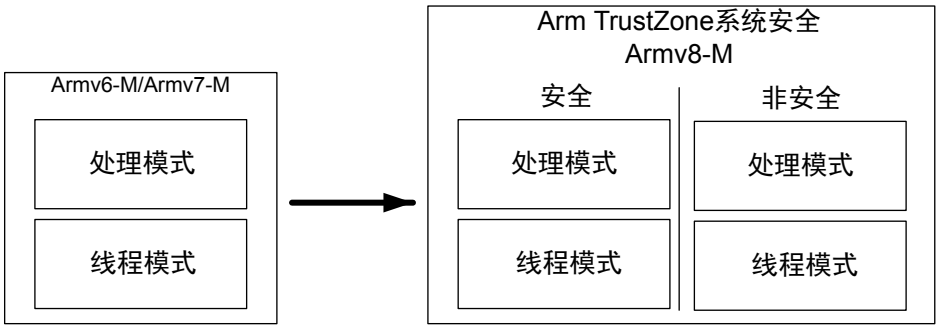
Armv8-M 架构为 Arm 微控制器带来了新的安全范例。它可以在微控制器系统级实现 TrustZone 技术，可以在运行时通过可靠的隔离来开发可信固件。

TrustZone 技术依赖于具有双寄存器组（用于安全域和非安全域）的处理器（Cortex-M23 或 Cortex-M33），还依赖于在整个系统（外设和存储器）中传播安全属性的总线基础设施（AHB5）。

TrustZone 可以在运行时实现强大而灵活的安全控制。从安全域切换到非安全域（反之亦然）非常简单，几乎没有周期损失。无需类似于应用处理器 Cortex-A 核那样的专门的 TrustZone 管理程序。

安全模式与现有模式、线程和处理程序相互交错。因此，在每种安全模式下都可以有一个线程或处理模式（请参见下图）。

图 3. Armv8-M TrustZone 执行模式



在 Armv8 TrustZone 上运行的典型固件架构上，由非安全域执行应用和 OS 任务，而由安全域执行安全应用和系统信任根机制。

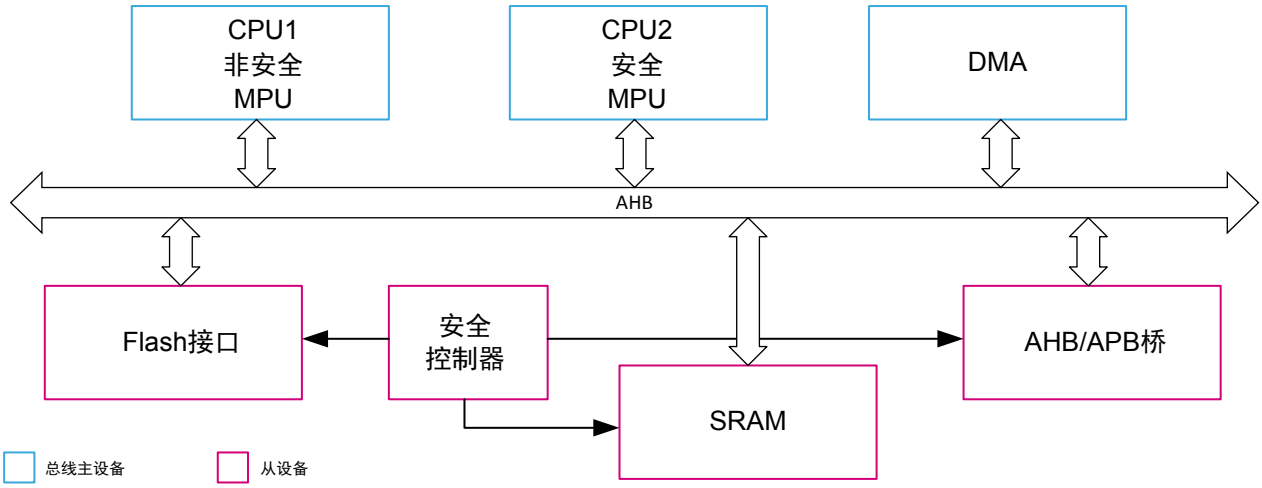
4.2 双核安全

在双核产品中，一个内核可以负责安全事务，而另一个内核负责非安全事务。有些产品（特别是双核 STM32WL 器件）通过硬件支持将安全属性传播到内存和外设，从而确保能够实现稳健的运行时隔离。

为双核 STM32WL 器件增加了专用安全控制器，以便于隔离。

在 Flash 存储器接口配置中定义了专用于安全 CPU2 的安全非易失性存储器，而不是使用属性单元。关键外设（如 DMA）必须传递安全上下文（参照 UM2643 *面向 STM32WL 系列的 STM32CubeWL 入门*，详细了解如何使用此混合架构）。

图 4. 简化的双核系统架构图



4.3 存储器保护

考虑系统安全时，存储器保护是最重要的因素。存储器包含敏感代码和数据，不得通过任何非预期接口（调试端口）或未授权进程（内部威胁）进行访问。

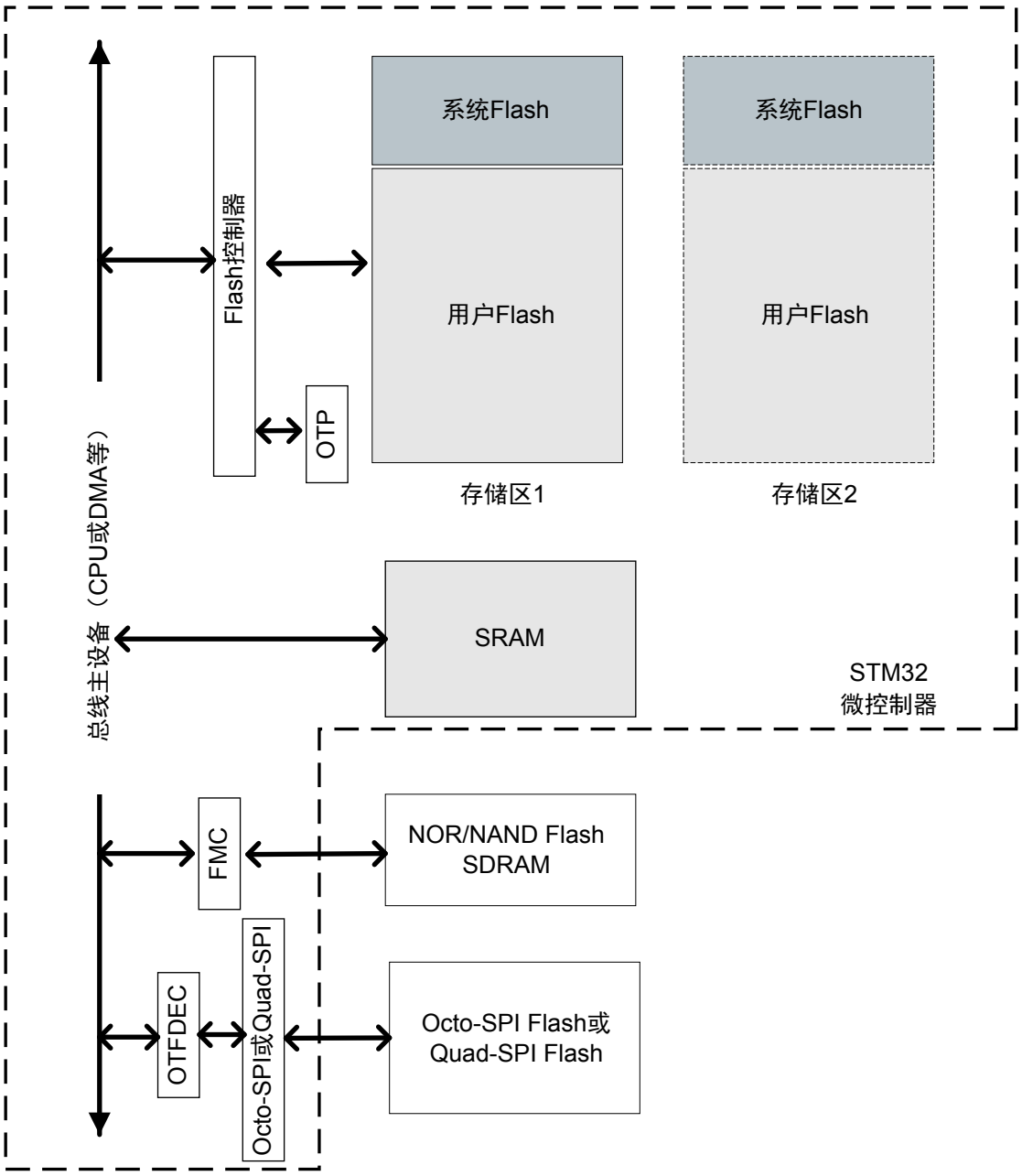
根据要保护的资产（代码或数据），可设置各种机制在授权访问源处（外部端口、内部进程）或对要保护的存储器类型（Flash、SRAM 或外部存储器）建立保护。

可通过存储器接口（如 Flash 控制器）、总线控制器 IP（防火墙）或通过内核 MPU（若可用）执行部分访问过滤。有关专有保护（安全隐藏保护、PCROP、WRP、RDP）的详细信息，请参考第 6 节 STM32 安全特性。

内置 Flash、内置 SRAM 和外部存储器的用途有很大的区别。它们各自的保护机制区别如下。

下图简单介绍了微控制器中的存储器访问架构。

图 5. 存储器类型



下表总结了各类存储器的特殊性以及典型保护特性。

表 5. 存储器类型和相关保护功能

存储器	类型	说明	保护器件
系统 Flash	.内部 .NVM .ROM	Flash 的 ROM 部分。内置器件 bootloader 和其它意法半导体服务。	无法更新（擦除/写入）。 某些部分不可读取。
用户 Flash 存储器	.内部 .NVM	用户应用的 Flash	内部保护： • RDP • WRP（不适用于 SRAM） • TrustZone • PCROP（不适用于 SRAM） • OTP（不在 SRAM 中） • 防火墙 • 安全隐藏保护（不适用于 SRAM） • MPU
SRAM	.内部 .易失性	栈、堆或缓冲区的工作存储器。可用于执行从内部或外部非易失性存储器下载的固件。	
NAND、NOR、Octo-SPI 或 Quad-SPIFlash	.外部 .NVM	用于应用或数据存储的其它存储器	密码 写保护（Flash 器件上） TrustZone
SDRAM	.外部 .易失性	用于应用执行的其它 RAM	密码

4.3.1 系统 Flash

在 STM32 MCU 中，系统存储器为内置且只可读（ROM），专用于 ST bootloader。某些器件可能在该区域包含其他安全服务（RSS）。为确保其真实性和完整性，不能对此部分进行修改。由于 bootloader 不包含任何敏感算法，因此是可读的。RSS 的某些部分是隐藏的，不能被用户读取。

系统 Flash 的保护属性为不可修改。

4.3.2 用户 Flash 存储器

用户 Flash 是用于存储固件和非易失性数据的主要用户存储器，属于嵌入式 Flash 的组成部分，可通过一系列存储器保护特性对其进行保护，且这些特性适用于所有 STM32 MCU。

外部攻击

与外部 Flash 不同的是，嵌入式 Flash 可轻松实现外部攻击保护。通过 RDP 禁用调试端口访问和控制连接接口访问可充分隔绝外部攻击。

相关保护机制：RDP 可禁用调试端口访问

内部攻击

对存储器的内部读访问或写访问可能是由注入到器件 SRAM 中或在不受信任库内的恶意软件发起的，因此须仅允许通过验证的进程访问关键代码和数据。

相关保护：PCROP、MPU、防火墙、安全隐藏保护和 TrustZone

保护未使用的存储器

至始至终都须默认对 Flash 设置写保护（即使是未使用的区域也不例外），以防止修改或注入代码。最好的做法还是在未使用的存储器内填入已知值，比如软件中断（SWI）操作码、非法操作码或 NOP。

相关保护机制：MPU 或 WRP

误码校正 (ECC)

Flash 可采用 ECC 实现错误检测和校正（最多使用 2 位进行错误检测，使用 1 位进行错误校正）。它更被视为一种功能安全特性，还可以做为一种互补保护以防故障注入。

4.3.3 内部 SRAM

内置 SRAM 属于器件运行的存储器。它在运行时用于堆栈/全局缓冲区和变量。SRAM 可作为字节、半字（16 位）或全字（32 位）以最大系统时钟频率访问，没有等待状态。

代码执行

如果某段代码需要更快的性能，那么可以将其从用户 Flash 或者外部 Flash 拷贝到 SRAM 中执行。在 SRAM 中执行代码的另一个原因是，如果固件本身是加密的并存储在外部 flash 上，且 MCU 也没有提供动态解密的组件时，在加密固件开始被执行之前，必须将其解密到内部 SRAM 中。并在 SRAM 中对执行的代码地址范围进行精确的 MPU 属性配置。如果 SRAM 中无需执行任何代码，那么建议通过 MPU 准确配置其属性为永不执行，以防运行任意恶意程序的可能。

相关保护：MPU 或防火墙

SRAM 清除

SRAM 可能包含一些可从中获取机密信息的敏感数据或临时值。一个典型的例子是，位于受保护区域的密钥需要传输时，是以明文的形式临时存储在 SRAM 中。强烈建议在函数操作敏感数据结束之后立即显式地清除工作缓存和变量。

提示 *复位时，STM32 MCU 允许自动擦除 SRAM（请参考各器件的参考手册）。对于某些产品来说，设置 RDP 时，部分 SRAM 已受到保护，不会受到外部访问或不受信任的启动（SRAM 启动）。*

写保护

可通过写保护隔离部分区域，以免被另一进程破坏或避免受到溢出攻击。溢出攻击是指写入多于目标缓冲区大小的数据（例如在数据传输期间通过接口端口写入）。如果未执行边界检查，超出缓冲区的存储器地址会被破坏，木马可能通过这种方式被注入。只有主要用于代码执行的 SRAM 区域才具有这种保护（该保护功能对于数据不适用）。SRAM 写保护功能仅在部分 STM32 系列器件上的 SRAM2 区域提供（参照第 6.1 节 安全特性概述和产品的参考手册）。

相关保护：MPU、TrustZone 或 SRAM 写保护（仅在某些 STM32 系列上提供）

奇偶校验和 ECC

SRAM 上奇偶校验功能可按字（32 位）控制潜在的错误。为存储器内容（数据总线宽度为 36 位）的每个字节额外添加一位，以提高其稳健性，使其达到 Class B 或 SIL 标准的要求。ECC 具有 SECDED 功能，更为复杂，但仅适用于某些 MCU 产品线的 SRAM。它们无法被禁用，并在默认情况下提升故障保护。

4.3.4 外部 Flash

外部 Flash 通过专用接口（NAND、NOR、Octo- 或 Quad-SPI）连接到微控制器。与内置 Flash 一样，外部 Flash 也包含代码和数据，但外部存储却引入了保密性（内容保密）和身份验证（器件保护）等问题。硬件保护限制为写锁定，以避免擦除或修改内容。进一步的保护由加密算法提供。内容至少须进行签名，以避免执行未经认证的固件。仅当内容需要保密时，才需要进行加密。

嵌入式代码可在原位置执行，也可以在执行之前加载到器件 SRAM 中。仅当器件具有实时解密功能时，才能在加密固件原位置执行。在另一种情况下，固件必须在加载到 SRAM 后解密。 如果解密后的代码, 或其中部分, 未受到 RDP 的保护 (RDP2), 则违反了代码的机密性。之外，还建议将加密与完整性保护进行结合。

相关保护：OTFDEC

4.3.5 STM32 存储器保护概述

STM32 提供了各种安全特性，以涵盖考虑到的各种用例。下表列出了其各自的范围并在第 6 节 STM32 安全特性中进行说明。

表 6. STM32 嵌入式存储器保护特性的范围

特征	外部攻击保护	内部攻击保护	Flash	SRAM
RDP	有	无	有	有
防火墙	无	有	有	有
MPU	无	有	有	有
PCROP	有	是（读/写）	有	无
WRP(1)	有	有	有	无
HDP	有	有	有	是（对于执行）(2)
TrustZone	有	有	有	有

1. *RDP 级别 ≠ 2 时，可取消设置写保护。*
2. *SRAM 仅在执行安全代码时受安全区保护。退出安全区之前须将其清空。*

4.4 软件隔离

软件隔离是指防止不同进程相互影响的运行时机制（进程间保护）。这些进程可按顺序执行，也可同时执行（例如操作系统的任务）。SRAM 中的软件隔离可确保每个进程各自的栈和工作数据不会被其他进程访问。进程间保护也可以扩展到 Flash 中的代码和非易失性数据。

软件隔离的目标：

- 防止一个进程监视另一个敏感进程的执行。
- 保护进程执行，以免因内存泄露或溢出导致栈破坏（未实施正确的内存管理）。

可以通过下表中列出并在第 6 节 STM32 安全特性中详细介绍的不同机制来实现这种储存器保护。

表 7. 软件隔离机制

保护	类型	隔离
MPU	动态	通过权限属性 (1)
防火墙	静态	通过总线地址硬件设置
安全隐藏保护	静态	复位时抢占进程
双核	静态	通过内核 ID(2)
TrustZone	静态和动态	通过从内核传播到所有资源的安全属性

1. *属性保护仅适用于 CPU 访问，不考虑用于其它总线主设备（例如 DMA）。*
2. *读取 CPUID 指示当前正在执行代码的 CPU。HAL_GetCurrentCPUID 函数中可以找到示例。*

4.5 调试端口及其它接口保护

调试端口可提供对内部资源（内核、存储器和寄存器）的访问，应在最终产品中被禁用。它是最基本的外部攻击，只需通过安全、不可变的固件禁止 JTAG（或 SWD）端口（请参考第 5.2.1 节 安全启动（SB））、或最好通过永久禁用功能（RDP2 中的 JTAG 熔断）便可避免受到攻击。

之外，还可以通过其它串行接口访问内部资源。若存在 bootloader，则可通过 I2C、SPI、USART 或 USB-DFU 访问器件内容。如果接口在运行时开放，应限制其传输协议的访问能力（工作模式、地址访问范围等）。

相关 STM32 特性：

- 读保护 (RDP)
- 禁用未使用的端口。
- 禁止 bootloader 访问（通过 STM32 器件中的 RDP 配置）。

4.6 启动保护

启动保护会针对系统中的第一条软件指令进行保护。如果攻击者成功修改器件启动地址，便可执行自己的代码，以绕过初始保护配置，或者执行不安全的间谍程序来访问器件存储器内容。

微控制器可进行启动配置，以选择在用户应用、bootloader 应用还是在 SRAM 所在固件中启动。启动保护依赖于受信任代码的单一入口点，而受信任代码可以是用户应用，也可以是安全服务区（RSS，若提供）。

相关 STM32 特性：

- 读保护 (RDP)
- 唯一启动入口
- 安全隐藏保护（HDP）
- TrustZone

4.7 系统监控

可设置对器件电源和环境的监控，以免发生故障，并采取相应的对策。一些安全机制如入侵检测即是专门用于此类安全保护。还有其它主要用于功能安全方面的机制，也可以用在信息安全方面。比如，当检测到掉电或者外部时钟源意外停止时（功能安全），可能意味着当前系统正在被攻击（信息安全）。

入侵检测用于检测系统级/板级入侵。可在 MCU 引脚上检测到开盖，并会触发相应的操作。内部入侵传感器能够检测到异常的电压、温度或其他参数。

时钟安全系统用于防止外部振荡器出现故障。如果检测到外部时钟上的故障，微控制器会切换为使用内部时钟，以安全执行操作。固件通过中断信号可对时钟故障事件做出反应。

电源和电压监控用来检测异常电平。低于某个电压值时，无法保证正常运行，而这可能是故障注入攻击的征兆。

器件温度可通过内部温度传感器来测量。温度信息会通过内部 ADC 通道反馈给器件。监控应用可根据温度范围采取相应措施。温度升高可能是故障注入攻击方案的一部分。

相关 STM32 特性：

- 入侵保护（使用 RTC 组件）
- 时钟安全系统
- 电源监控
- 温度传感器

5 安全应用

为了创建安全的系统，需要在实现安全固件架构时使用硬件功能。行业标准解决方案是 Arm 为 IoT 生态系统提出的 PSA。意法半导体专有解决方案包括安全启动（SB）和安全固件更新（SFU）。

本节在介绍以下典型安全应用之前定义了信任根和信任链的概念：

- 安全启动。
- 安全固件更新（SFU）
- 安全存储
- 加密服务

这些应用与密码学算法有着紧密联系。所有加密方案均基于密钥、公钥和哈希这三个概念。附录 A 中介绍了加密的基础知识。加解密算法 - 主要概念.

提示

- *用户手册 UM2262“X-CUBE-SBSFU STM32Cube 扩展包入门”提供了实现 SB 和 SFU 的示例（www.st.com/en/product/x-cube-sbsfu）。*
- *用户手册 UM2671“STM32CubeL5 TF-M 应用程序入门”介绍了实现 STM32L5 系列 MCU 的 TF-M 的示例。*
- *用户手册 UM2851“STM32CubeU5 TF-M 应用程序入门”介绍了实现 STM32U5 系列 MCU 的 TF-M 的示例。*

5.1 信任根和信任链

信任根和信任链的原理对许多（即使不是全部）安全系统来说是通用的。它显然可以随意扩展，具有固有的高效性和灵活性。

信任链是由一系列组件构成的，每个组件的安全均通过其它组件来保障。信任根是总体安全所依赖的安全链的起点。

一个安全启动的实现，它本身必须作为器件的唯一入口点，在复位后开始启动，且在安全模式下，它不可被修改。然后它对后续功能进行身份验证，并执行固件的下一部分，使固件能够安全地证实后续安全链节点所需的其他功能。例如，它配置易失性存储器保护，以便其可在安全存储服务中被使用。

5.2 意法半导体专有 SBSFU 解决方案

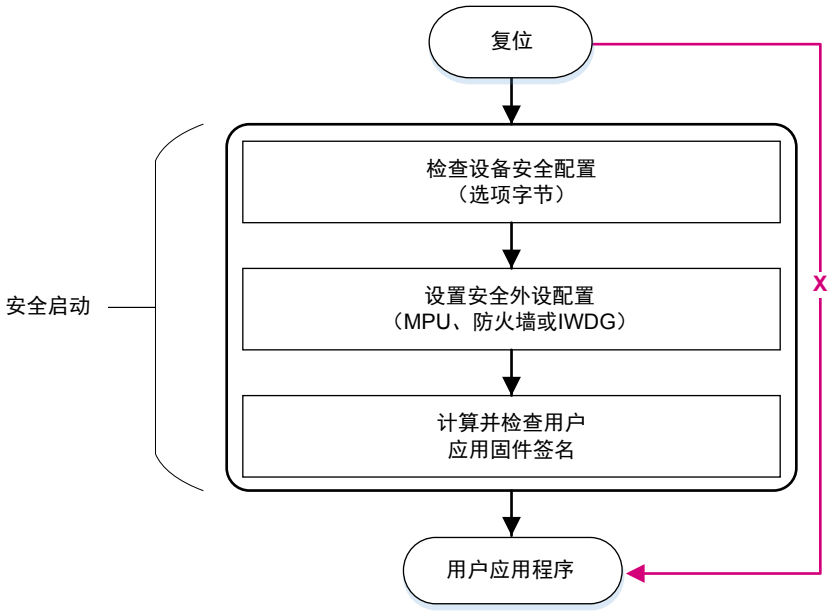
5.2.1 安全启动（SB）

SB 应用在复位时会先于用户应用执行。它提供了第一阶段的安全功能，随后会负责保护系统全局信任链。

SB 主要功能：

- 检查 STM32 安全配置和设置运行时保护。
- 声明所执行的用户应用程序映像的完整性和真实性（见下图）。

图 6. 安全启动流程图



检查器件安全性

这一部分的 SB 应用会检查静态配置是否正确，并会设置动态配置。静态安全配置是通过选项字节（RDP、PCROP、WRP 和安全隐藏保护）定义的。动态保护需要进行编程（防火墙、MPU、入侵检测和 IWDG）。

完整性和真实性检查

检查固件完整性时，会对应用映像采用哈希算法（使用 MD5、SHA1 或 SHA256 哈希算法）并将摘要与预期值进行比较，以此来确认应用固件无错误。

通过固件所有者与器件之间共享的密钥对固件进行加密得到的标签，并与预期标签进行对比，通过这种方式可实现真实性检查。此共享密钥存储在器件的受保护区域内。

保护属性

SB 固件须具有以下属性，以执行其角色的任务：

- 必须是器件唯一入口（无旁路）。
- 其代码必须是不可变的。
- 必须可访问敏感数据（例如证书或应用签名）。

SB 最敏感的部分将利用进程和数据隔离特性，如防火墙、MPU 或安全隐藏保护。实现取决于 STM32 系列的可用特性。

5.2.2 安全固件更新（SFU）

SFU 实现了安全的现场固件更新，可以将新固件映像下载到设备。

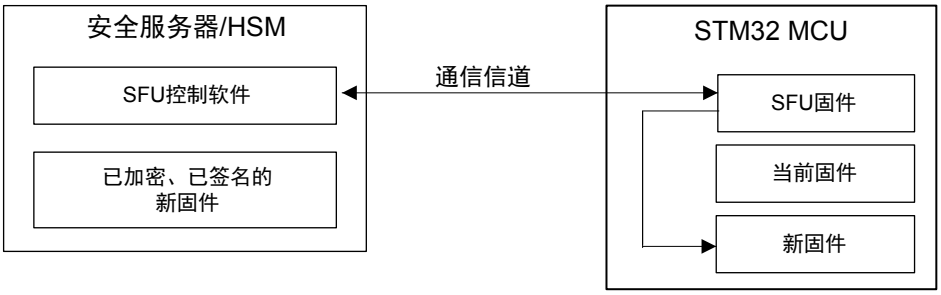
固件更新属于敏感操作，必须对器件所有者和应用（OEM）所有者这两方进行保护：

- 从器件所有者来看，目标是避免下载错误的固件（有意或无意），从而导致损坏器件。
- 从应用所有者（OEM）来看，需要保护其固件不被克隆或加载到未授权器件上。

架构

SFU 传输涉及两个实体：固件所有者和要更新的设备（请参见下图）。由于容易被窃听，通信通道通常被视为不安全，整体安全责任由发送方（固件所有者服务器）和接收方（设备）共同承担。

图 7. 安全的服务器/设备 SFU 架构



应用

在 OEM 侧，安全服务器负责向认证的设备端发送加密的（如果需要）和签名过的固件。
在器件上运行的 SFU 负责以下各项：

- 在安装加载的映像之前对其进行身份验证和完整性检查
- 如果需要保密，则对新固件进行解密
- 检查新的固件版本（防回滚机制）

5.3 Arm TF-M 解决方案

当安全的 Cortex M-33 内核与 Armv8-M 架构一起引入时，Arm 可信固件已经存在了一段时间。它提供了符合 PSA 标准的更紧凑的 TF-M 开源实现方式，作为参考安全固件框架。
对于利用 ARMv8 架构的 ST MCU（如 STM32L5 器件），将 SBSFU 替换为 TF-M 解决方案。
有关 TF-M 本身的文档，请使用 Arm 资源以及代码注释。
有关 STM32L5 系列 MCU 上 TF-M 集成的指南，请参阅固件示例用户手册。
用户在从 SBSFU 迁移到 TF-M 时，可能会发现一个有用的快速比较表，其中比较了一些基本功能上的差异：

表 8. 基本功能差异

特征	SBSFU v2.3.1	STM32L5 上的 TF-M
RoT 服务	信任传播	HUK 和证明信息管理
Bootloader	有	无
固件加密	多个选项，包括 OTFDEC	不包含
密钥管理	NMV 中的密钥，静态代码	易失性密钥，可更新代码
安全存储	密钥管理	是，基于 HUK
初始认证	无	有
支持安全元件	STSAFE-A110	无

这两种解决方案有着不同的要求和不同的架构。它们并不意味着可以互换或存在竞争关系。
在 UM2671“*STM32 TF-M 用户手册*”中提供了有关在 STM32 器件上构建 TF-M 的更多详细信息。
在应用笔记 AN5447 中进行了详细的比较（*X-CUBE-SBSFU 与 TF-M 比较*部分）。

5.4 产品认证

不同的安全应用程序通常需要某些认证，以证明其能够以安全的方式执行。独立机构或政府机构通过评估和/或测试向 MCU 或基于 MCU 的系统授予认证状态。

STM32 微控制器相关认证和评估包括但不限于：

- PSA - 平台安全架构 -由 Arm 管理，专注物联网安全、MCU 认证、三级评估
  - STM32L4 系列通过一级认证，STM32L5 系列（带 TF-M）通过二级认证，STM32U5（带 TF-M）通过三级认证
  - 如要达到 Arm PSA 的认证安全级别，请参考用户手册《*STM32U585 针对通过 SESIP 配置文件实现 PSA Level 3 认证™的安全指南*》（UM2852）。
- SESIP - 物联网平台安全评估标准 - 几个主要安全评估实验室采用的国际方法，分五个级别
  - 使用 SBSFU 或 TF-M 的系统（STM32L4、L4+、L5 和 U5 系列器件）符合 level 3 标准
- PCI - 支付卡信息 - 关注销售终端（POS）应用的重要安全标准
  - 有良好的系统（例如，使用 STM32L4 系列器件）成功评估记录
- CC - 通用标准 - 通用认证标准要求高标准的开发、测试和文档编制质量。

6 STM32 安全特性

本节介绍了可结合在一起实现之前章节中提出的安全概念以及可实现高级安全功能的所有 STM32 特性。

6.1 安全特性概述

静态和动态保护

可根据保护特性属于静态还是动态进行区分：

- 静态保护是指通过选项字节设置的特性。其配置在掉电后仍会保留。

  静态保护包括 RDP、PCROP、WRP、BOR、OTP 和安全隐藏保护（若可用）。
- 动态（或运行时）保护在复位时不会保留其状态。应在每次启动时对其进行配置（例如在安全启动（SB）期间）。

  由 STM32 提供的动态保护包括 MPU、入侵检测和防火墙。

  其它动态保护可能同时与信息安全和功能安全相关。一个异常环境行为有可能是意外造成的（功能安全），也有可能是为了攻击而蓄意制造的（信息安全）。针对这种情况的对应保护措施有时钟和电源监控系统，存储器完整性位校验，以及独立看门狗（IWDG）。

STM32 系列的安全特性

以下表格按照 STM32 系列列出了可用特性。

表 9. **STM32Fx 系列的安全特性**

特征	STM32F0	STM32F1	STM32F2	STM32F3	STM32F4	STM32F7
Cortex 内核	M0	M3	M3	M4	M4	M7
RDP 附加保护	+ 备份寄存器	仅限 RDP 级别 2 (1)	+ 备份 SRAM	+ 备份寄存器	+ 备份 SRAM	+ 备份 SRAM
Flash WRP	按扇区（4 KB）	按页（4 KB 或 8 KB）	按扇区（16K、64K 或 128 KB）	按扇区（4 KB）	按扇区（16K、64K 或 128 KB）	按扇区（16 KB、64 KB、128 KB 或 256 KB）
SRAM WRP	无	无	无	无	无	无
PCROP	无	无	无	无	按扇区	按扇区
HDP	无	无	无	无	无	无
防火墙	无	无	无	无	无	无
MPU	无	有(2)	有	有	有	有
OTP	无	无	有	有	512 字节	528 - 1040 字节
OTFDEC	无	无	无	无	无	无
唯一启动入口(3)	无	无	无	无	无	无
安全启动：系统 Flash 存储	无	无	无	无	无	无
内部篡改检测	无	无	无	无	无	无
硬件加密加速器	无	无	AES, HASH	无	AES, HASH	AES, HASH
硬件加密：TRNG	NA	NA	SP800-90-A	NA	SP800-90-A	SP800-90-A
安全软件：安全固件安装	无	无	无	无	无	无
安全软件：SBSFU	无	无	无	无	有	有
安全软件：TF-M	无	无	无	无	无	无
安全软件：KMS	无	无	无	无	无	无

1. *STM32F1 系列中的 RDP 仅可用于 Flash 保护。RDP 置位（RDP1）或取消置位（RDP0）。RDP 级别 2 未实现。*
2. *仅 XL 密度部件具有 MPU。*
3. *“无 UBE”意味着启动只能依靠 RDP 级别 2。“有 UBE”意味着存在专用的启动锁定服务。*

表 10. **STM32Lx 和 STM32Ux 系列的安全特性**

特征		STM32L0	STM32L1	STM32L4 STM32L4+	STM32L5	STM32U5
Cortex 内核		M0	M3	M4	具有 TrustZone 的 M33	具有 TrustZone 的 M33
RDP 附加保护		+ EEPROM	+ EEPROM	+ 备份寄存器 + SRAM2	四个 RDP 级别 + 备份寄存器 + SRAM2	四个 RDP 级别 + 备份寄存器 + SRAM2
Flash WRP		按扇区（4 KB）	按扇区（4 KB）	按区域（粒度为 2 KB） 每个存储区一个区域	最多四个具有 2 KB 或 4 KB 粒度的保护区域	每个存储块 2 个区域，由区域设置定义
PCROP		按扇区	按扇区	按区域（粒度为 8 字节） 每个存储区一个区域	无	无
HDP[1]		无	无	无	TrustZone 安全域内最多两个安全隐藏区域（HDP）	每个存储块一个安全隐藏区域（HDP），位于 TrustZone 安全域内
防火墙		有	无	有	基于 TrustZone	基于 TrustZone
MPU		有	有	有	有	有
唯一启动入口[2]		无	无	无	有	有
内部篡改检测		无	无	无	有	有
IWDG		有	有	有	有	有
器件 ID（96 位）		有	有	有	有	有
硬件加密加速器	对称	AES	AES	AES	AES, OTFDEC	AES，安全的 AES，OTFDEC
	非对称	无	无	无	PKA	PKA
	支持	无	无	HASH, TRNG	HASH, TRNG	HASH, TRNG
RAM（带保护功能）		无	无	SRAM2（粒度为 1 KB）	SRAM2（粒度为 1 KB）	SRAM2（粒度为 1 KB）

1. *根据产品的不同，HDP 被称为安全用户存储器、粘性区域或安全存储器。*
2. *“无 UBE”意味着启动只能依靠 RDP 级别 2。“有 UBE”意味着存在专用的启动锁定服务。*

表 11. STM32H7、STM32G0、STM32G4、STM32WB 和 STM32WL 系列的安全特性

特征		STM32H7 系列			STM32G0	STM32G4	STM32WB	STM32WL
		STM32H72x/73x	STM32H74x/75x	STM32H7Ax/Bx				
Cortex 内核		M7			M0+	M4	M4 和 M0+	M4 和 M0+(1)
RDP 附加保护 (2)		+ 备份 SRAM + 备份寄存器 + OTFDEC	+ 备份 SRAM + 备份寄存器	+ 备份 SRAM + 备份寄存器 + OTFDEC	+ 备份寄存器	+ 备份寄存器 + CCM-SRAM	+ 备份寄存器 + SRAM2	+ 备份寄存器 + SRAM2
WRP		按扇区（128 KB）		按一组 x4 8-Kb 字节扇区	按区域（粒度为 2 KB） 提供两个区域	按页（2 KB 或 4 KB）	按区域（粒度为 4 KB） 提供两个区域	按区域（粒度为 2 KB） 提供两个区域
PCROP		按区域（粒度为 256 字节）	按区域（粒度为 256 字节） 每个存储区一个区域		按区域（粒度为 512 字节） 提供两个区域	按区域（具有 64 位或 128 位粒度） 最多两个区域 (3)	按区域（粒度为 2 KB） 最多两个区域	按区域（粒度为 1 KB） 提供两个区域
HDP(4)		是（安全用户存储器，粒度为 256 字节）			是（可保护存储区）	是（可保护存储区）	是（仅限于 CM0 +固件）	有
防火墙		无			无	无	无	有
MPU		有			有	有	是（CM4）	有
唯一启动入口(5)		是（安全访问的唯一入口点）			是（启动锁定特性）	有	无	是（启动锁定特性）
内部篡改检测		有			有	有	无	有
IWDG		有			有	有	有	有
器件 ID（96 位）		有			有	有	有	有
硬件加密 (2)	对称	AES			AES	AES	AES	AES
	非对称	无			无	无	PKA	PKA
	支持	HASH, TRNG			TRNG	TRNG	TRNG	TRNG
SRAM 写保护		无			无	CCM SRAM（粒度为 A-Kb）	SRAM2（粒度为 1-KB）	SRAM2（粒度为 1-KB）

1. *M0+内核不可用于所有 STM32WL 器件。*
2. *取决于器件产品编号。*
3. *区域数量取决于器件类别和双/单排配置。*
4. *根据产品的不同，HDP 被称为安全用户存储器、粘性区域或安全存储器。*
5. *“无 UBE”意味着启动只能依靠 RDP 级别 2。“有 UBE”意味着存在专用的启动锁定服务。*

6.2 读保护 (RDP)

读保护属于全局 Flash 保护机制，可防止嵌入的固件代码被复制、逆向工程、使用调试工具读取或在 SRAM 中注入代码。用户须在将二进制代码烧录进内置 Flash 后设置这一保护。

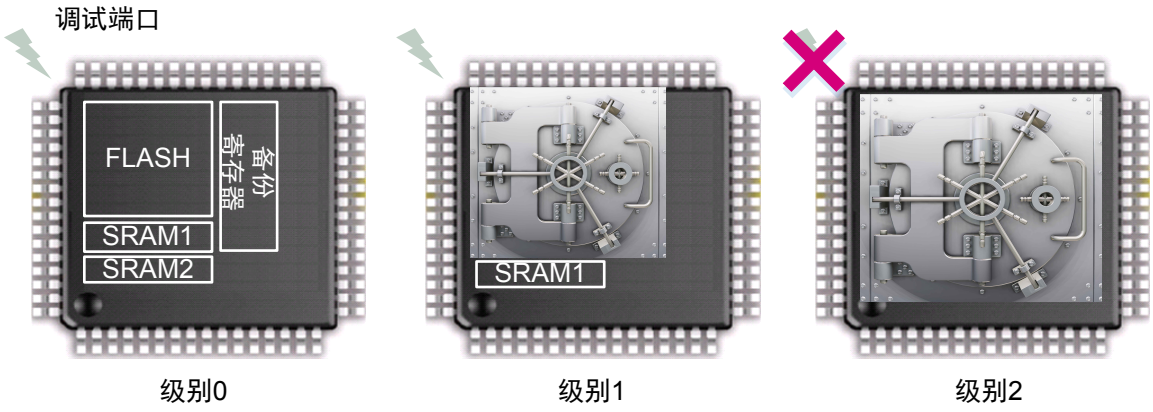
读保护应用于 STM32 系列的：

- 主 Flash
- 选项字节（仅限级别 2）

根据 STM32 所属的系列，可提供的附加保护包括：

- 用于实时时钟（RTC）的备份寄存器
- 备份 SRAM
- EEPROM

图 8. **RDP 保护示例（STM32L4 系列）**



RDP 级别（0、1 和 2）的定义如下：

- 级别 0：该级别为默认 RDP 级别。Flash 完全打开，可在所有启动配置（调试特性、从 RAM、从系统内存 bootloader 启动或从 Flash 启动）下进行全部内存操作。该配置模式不提供保护，仅适用于开发和调试。
- 级别 1：禁止通过调试特性（如串行线或 JTAG）进行 Flash 访问（读取、擦除、编程）或 SRAM2 访问，即使通过 SRAM 或系统内置 bootloader 启动的情况也不例外。在这些情况下，任何对受保护区域的读请求都会生成总线错误。

  但是，如果系统从主 FLASH 上启动，通过用户代码访问主 FLASH 和 SRAM2 则是允许的。
- 级别 2：激活 RDP 级别 2 时，级别 1 下提供的所有保护均有效，MCU 受到全面保护。RDP 选项字节和所有其他选项字节都会被冻结，不能再修改。JTAG、SWV（单线查看器）、ETM 和边界扫描全部禁用。

在基于 ARMv8 架构构建的器件上，可以使用第四个 RDP 级别：

- 级别 0.5：仅限非安全调试。可以对非安全 Flash 进行所有的读写操作（前提是未设置写保护）。禁止对安全区域的调试访问。仍然可以调试访问非安全区域。

RDP 级别回退

RDP 始终可以升级。级别回退可能会导致以下结果：

- 从 RDP 级别 1 回退到 RDP 级别 0 会导致 Flash 批量擦除以及 SRAM2 和备份寄存器擦除。
- 从 RDP 级别 1 回退到 RDP 级别 0.5 会导致部分 Flash 擦除：仅擦除非安全部分。
- 从 RDP 级别 0.5 回退到 RDP 级别 0 会导致 Flash 批量擦除以及 SRAM2 和备份寄存器擦除。

在 RDP 级别 2 时，无法进行回退。

受 RDP 保护的 STM32 微控制器内部 Flash 内容更新

在 RDP 级别 1 或级别 2 时，不能再通过外部访问（bootloader 或通过 SRAM 启动）修改 Flash 内容，但始终可通过内部应用进行修改。可通过 SFU 应用或（即使从安全角度来说并不建议这样做）通过简单的应用内编程进程（IAP）进行修改。

下表总结了 RDP 保护。

表 12. RDP 保护

区域	RDP 级别	从用户 Flash 启动			调试或者从 SRAM 或 bootloader 启动		
		读	写	擦除	读	写	擦除
Flash 主存储器	0	有	有	有	有	有	有
	1	有	有	有	无	无	无
	2	有	有	有	N/A	N/A	N/A
系统存储器	0	有	无	无	有	无	无
	1	有	无	无	无	无	无
	2	有	无	无	N/A	N/A	N/A
选项字节	0	有	有	有	有	有	有
	1	有	有	有	有	有	有
	2	有	无	无	N/A	N/A	N/A
其它受保护资产 (1)	0	有	有	有	有	有	有
	1	有	有	N/A	无	无	无
	2	有	有	N/A	N/A	N/A	N/A

1. *备份寄存器/SRAM*

RDP 的适用情况

在消费类产品中，RDP 至少须设为级别 1。该级别可阻止通过调试端口或通过 bootloader 发动的基本攻击。但在 RDP 级别 1 下，如果回退到 RDP 级别 0 时将会引起全片擦除，此风险将导致拒绝服务。

如果要实现更高的安全级别应用（如不可变代码），那么 RDP 级别 2 是必须采用的。它的缺点是一旦使能后，器件配置无法再更新，比如芯片从客户返回原厂时。

支持 ARMv8 的产品提供新的 RDP 级别 0.5。它用于调试非安全应用，同时保护安全区域边界内的内容避开调试访问。有关该保护的更多信息，请参见应用笔记 AN5347（第 10 节“*使用 TrustZone®的开发建议*”）。

提示 *RDP 可用于所有 STM32 系列。*

6.3 一次可编程(OTP)

OTP 是 Flash 存储器中一个专用的隔离区域，只能被写入或锁定，防止任何修改。通常 OTP 是一个比用户 Flash 存储更小的区域。

该特性对于生命周期管理、预设置、个性化或配置非常有用。一旦 OTP 被写入，就不得不在不损坏芯片的情况下擦除数据。对写入数据的读取没有限制。

提示 *OTP 可用于大多数 STM32 系列，参照第 2 节 概述获取详细信息。*

6.4 TrustZone

下一节介绍了 TrustZone 架构的主要功能。有关更多信息，请参见应用笔记 AN5347“*STM32L5 系列 TrustZone®功能*”和参考手册 RM0438“*STM32L552xx 和 STM32L562xx 基于高级 Arm®的 32 位 MCU*”。

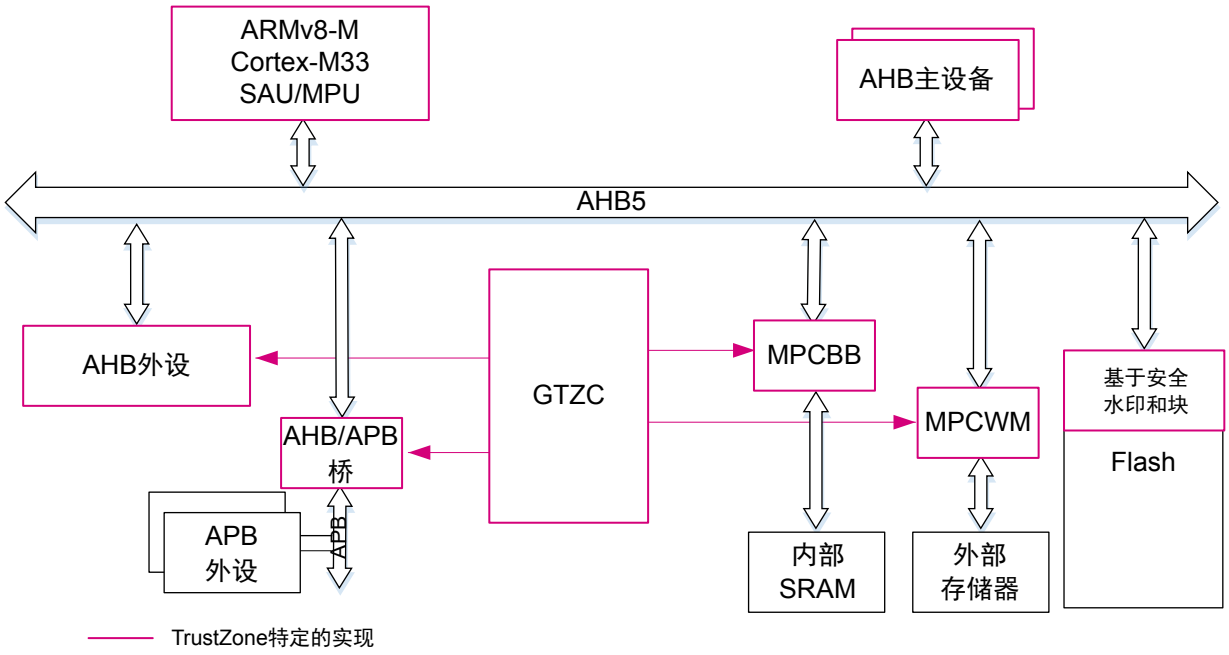
ARMV8-M TrustZone 架构在系统级定义了两个域：安全域和非安全域。完整的存储器映射空间分为安全区域和非安全区域。这包括所有存储器类型（Flash、SRAM 存储器和外部存储器），以及可以共享（每个域具有特定环境）或专用于一个域或另一个域的所有外设。

在系统级，安全域和非安全域之间的隔离依赖于以下硬件机制（请参见下图）。

- 具有用于安全域和非安全域的双组寄存器的特定内核架构（ARMV8-M Cortex-M33），以及用于声明地址范围安全状态的安全属性单元（SAU）。
- 实现定义属性单元（IDAU）是 SAU 的补充。
- 传播任何事务的安全和特权属性的总线基础设施（AHB5）

- 专用硬件模块，用于管理两个域之间的划分（GTZC 用于定义内部 SRAM 和外部 FSMC/OCTOSPI 存储器以及外设的安全属性）

图 9. 在系统级实现 TrustZone



6.4.1 内核状态

内核状态取决于当前运行代码的区域。当代码从安全区域运行时，内核处于安全状态。否则，内核处于非安全状态。

6.4.2 安全属性单元（SAU）

SAU 是耦合到内核（作为 MPU）的硬件单元。SAU 负责设置 AHB5 事务的安全属性。事务的安全属性通过存储器映射的资源（存储器区域或外设）的目标地址进行确定。根据 SAU 配置，将地址标记为安全、非安全可调用（NSC）或非安全。NSC 是安全域的子域，可通过其定义非安全代码的网关，以在特定入口点访问安全域。

SAU 可以通过安全固件进行配置。它可以在启动时配置为固定配置，也可以通过安全固件动态修改。

提示 *经修改后，安全属性不能低于硬件通过 IDAU（实现定义安全属性）设置的默认属性（按安全性顺序：安全> NSC>不安全）。请参阅参考手册中每种器件的实现细节。*

地址别名

由于资源的安全属性是取决于其固定地址，但是，根据应用需求，一个内存映射的资源可以设为安全的，也可以设为非安全的，这里就产生了一个明显的矛盾。为了克服这种矛盾，为每个内存映射资源分配两个地址：通过其中一个地址只能在安全模式下访问此资源，通过另一个地址却只能在非安全模式下访问这同一资源。这种机制被称为地址别名。

通过地址别名，所有外设访问可在仅有的两个区域内进行分组，而不是在多个分散的区域中分组。最后，IDAU 在以下区域中拆分内存映射资源：

- 外设安全/非安全区域
- Flash 安全/非安全区域
- SRAM 安全/非安全区域

有关详细配置，请参阅器件的参考手册。

6.4.3 存储器和外设保护

SAU 定义了事务安全属性，总线基础设施可将该属性传播到目标处。目标（存储器和外设）由硬件机制进行保护，这些机制根据安全属性和特权属性对访问进行过滤。

TrustZone 系统架构中有两种外设：

- 支持 TrustZone 的外设：直接连接到 AHB 或 APB 总线，具有特定的 TrustZone 行为（如安全寄存器子集）。这些外设包含访问过滤控件
- 安全外设：由受 GTZC 控制的 AHB/APB 防火墙门进行保护，以定义其安全属性。

支持 TrustZone 的外设是具有总线主控角色（DMA）、GTZC、Flash 控制器的外设以及其他在系统中起基本作用的外设：PWR、RTC 和系统配置块。剩余的系统外设则是安全外设。

GTZC

GTZC 定义了安全外设、内置 SRAM 存储器和外部存储器的访问状态：

- 可以使用 TZSC 将外设设置为安全或非安全（专有），特权或非特权。
- 内置的 SRAM 由 MPCBB 模块保护，保护区域是基于块的，每块大小 256 字节。
- 外部存储器是基于区域（水印：开始地址，长度）来保护的。受保护区域的数量取决于存储器的类型（NAND，NOR 或者 OCTOSPI）。
- 非法访问事件会导致 TZIC 生成安全中断。

提示 *Flash 安全属性是通过安全水印选项字节和/或基于 Flash 接口块的寄存器定义的。*

6.5 Flash 写保护（WRP）

写保护特性用于保护指定存储器区域的内容，以免其被擦除或更新。

对于 Flash 技术，更新须视为用零填充。

例如，可对 Flash 的某一页或某一扇区设置写保护，以避免其在固件或数据更新过程中被修改。还可以对未使用的存储区域设置写保护，以避免其中被注入恶意软件。写保护的粒度与其页大小或者扇区大小相关。

WRP 的适用情况

特别是在预期会在应用中写 flash 的情况下，写保护必须使用。比如预期执行数据存储或代码更新操作的情况。WRP 可避免因功能不安全进行错误访问而导致意外溢出的情况。

提示 *所有 STM32 系列均提供写保护功能。*

6.6 只执行固件(PCROP)

STM32Flash 的某些部分可配置为具有“仅执行”属性。存储在采用此配置的区域中的固件仅可通过 CPU 指令总线获取。禁止对此区域进行读取或写入。这种保护不仅适用于内部访问（固件），也适用于外部访问（调试端口）。在 STM32 中，此特性为专有代码读保护（PCROP）。

PCROP 是通过选项字节设置的静态保护。受保护区域数及其粒度划分取决于 STM32 系列（请参考表 9、表 10 和表 11）。使用 PCROP 时，在编译具有只执行属性的固件时必须多加留意（请参考用户编译器选项）。

PCROP 的适用情况

PCROP 用于保护第三方固件（知识产权）以及用户固件最敏感的部分。

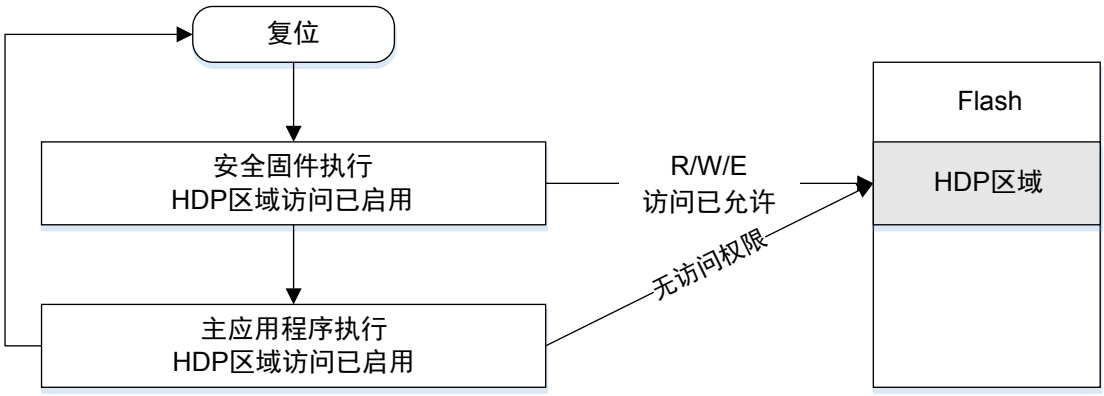
提示 *表 1 中列出的所有 STM32 系列均配备 PCROP，支持 TrustZone 的器件除外，该系列采用另一种保护机制作为替代。*

6.7 安全隐藏保护（HDP）

某些 STM32 器件支持 HDP 存储器概念。HDP（在 STM32L5 系列上称为安全隐藏保护）在 STM32H7 系列上又称为安全用户存储器，或在 STM32G0 系列上称为安全存储器。

HDP 区域是 Flash 的组成部分，仅可在器件复位后访问一次。HDP 针对的是内置或操作了机密数据以及启动时应安全执行的敏感型应用。一旦应用程序被执行，HDP 区域将关闭，无法以任何方式对其进行访问（请参见下图）。

图 10. HDP 保护的固件访问



HDP 是通过选项字节设置的静态保护。一旦设定后，无论启动引脚设置何种启动配置、无论使用哪一启动地址，CPU 都会从内置在这一区域内的固件上启动。

HDP 的适用情况

HDP 适用于仅在复位后执行的代码，如用于信任根的安全启动。

STM32H7、STM32G0、STM32G4 和 STM32L5 系列具有此项保护，其实现方式和名称略有不同（有关详细信息，请参阅专用参考手册）。

6.8 防火墙

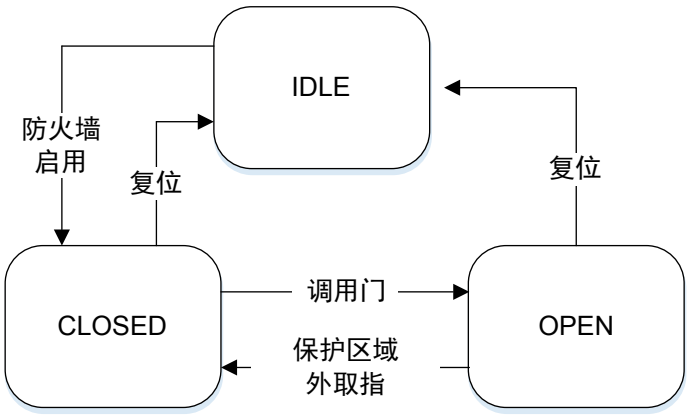
防火墙是一种硬件保护外设，它控制着总线事务以及过滤对代码区（Flash）、易失性数据区域（SRAM）以及非易失性数据区域（Flash）这三个特殊区域的访问。受保护的代码可以通过单入口点进行访问（调用门机制在下文有说明）。任何试图跳过并尝试不通过入口点而执行代码段中函数的操作都会导致系统复位。

防火墙属于动态保护的组成部分，需要在启动时例如通过 SB 应用进行设置。

调用门机制

防火墙通过调用一种“调用门”机制打开：必须使用单入口点来打开门并执行由防火墙保护的代码。如果访问受保护的代码而不通过调用门机制，则会产生系统复位。如果在受保护区域以外取指，防火墙会关闭（参见下图）。

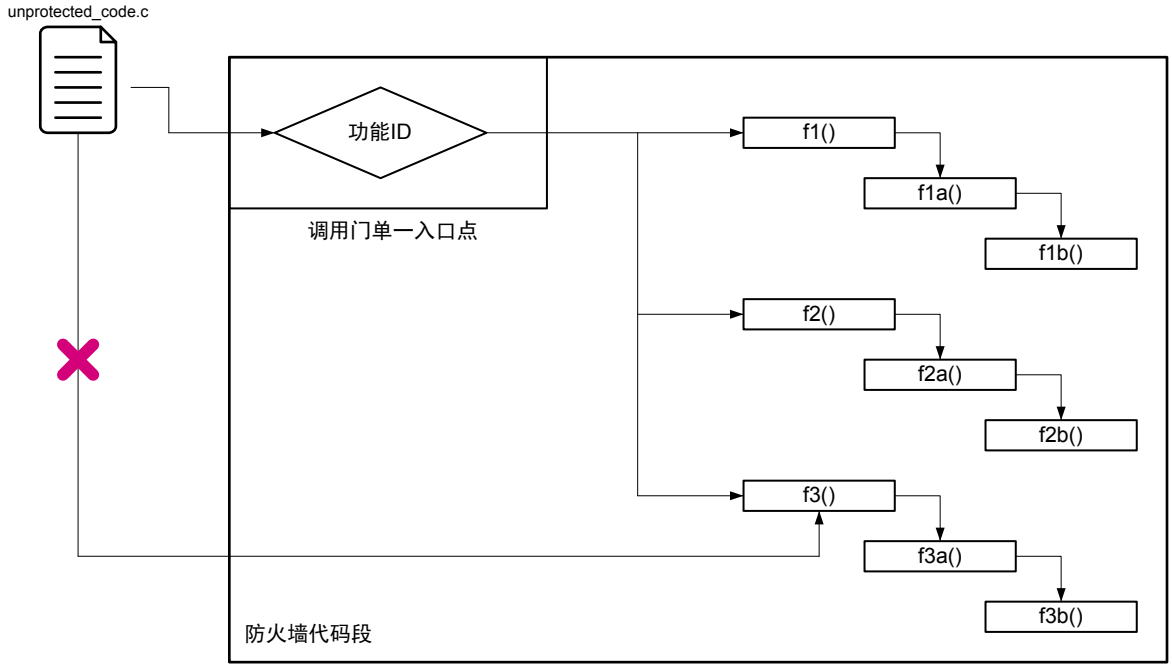
图 11. 防火墙 FSM



因为遵循调用门序列的唯一方法是通过单个调用门入口点，因此，必须提供某种机制来支持应用从未受保护的代码区中调用多个受防火墙保护的函数（例如，加密和解密功能）。可以使用参数指定要执行的功能（例如 `CallGate(F1_ID)` 或 `CallGate(F2_ID)`）。根据参数，来内部调用正确的函数。

该机制如下图所示。

图 12. 防火墙应用示例



防火墙的适用情况

防火墙保护代码和数据。只要遵循 call-gate 机制，便始终可调用受保护的代码。

提示 *防火墙仅在 STM32L0 和 STM32L4 系列上可用。*

提示 *参照 STM32L0/L4 FIREWALL 概述应用笔记（AN4729）了解更多关于防火墙的信息。*

6.9 存储器保护单元(MPU)

MPU 属于存储器保护机制，允许为器件的任何存储器映射资源（Flash、SRAM 和外设寄存器）定义特定的访问权限。此类保护在运行时进行动态管理。

提示 *MPU 设置的访问权限仅适用于 CPU 访问，其它总线主设备请求（如 DMA）不会由 MPU 进行过滤，必须在不需要使用的情况下禁用。*

区域访问属性

MPU 将存储器映射划分为多个区域，每个区域都有自己的访问属性。访问权限可设为*可执行、不可执行（XN）*、*读写（RW）*、*只读（RO）*或*不可访问*。

提示 *MPU 还可为各区域设置其它属性：可共享、可缓存和可缓冲。本应用笔记不涉及 MPU 的整体复杂性，本章仅作为介绍和高级概述。请参考对应的器件编程手册或应用笔记 AN4838《管理 STM32 MCU 中的存储器保护单元（MPU）》。*

特权级模式和非特权级模式

Arm Cortex-M 架构在访问属性之上定义了两种执行模式，允许进程在特权级模式或非特权级模式下运行。对于每一区域，均可为每种模式单独设置访问属性。

下表显示了混合模式和访问属性支持的不同用例。

表 13. 由 MPU 管理的属性和访问权限

特权级模式属性	非特权级模式属性	说明
永不执行（XN）(1)		代码执行属性
无访问权限	无访问权限	所有访问都会产生权限故障。
RW	无访问权限	仅通过特权级软件访问
RW	RO	由非特权级软件写入会产生权限故障。
RW	RW	完全访问权限
RO	无访问权限	仅通过特权级软件读取
RO	RO	只读，由特权级或非特权级软件读取

1. *永不执行（XN）属性按区域设置，对两种模式均有效。可通过该属性避免 SRAM 代码注入等。*

在特权级模式下执行的代码可访问其它特定指令（MRS），也可以访问 Arm 核心外设寄存器（例如 NVIC、DWT 或 SBC）。这一特性适用于要求访问非特权级固件不可访问的敏感资源的操作系统内核或安全代码段。

安全进程隔离策略

复位后，特权模式是任何进程的默认模式。因此 SB 应用会在特权级模式下执行。这是为了将安全进程（如 SB、OS 内核、密钥管理器或 SFU）与不安全或不受信任的进程（用户应用）隔离。

表 14. 进程隔离

固件类型	模式	资源访问
安全固件（如 SB 或 OS 内核）	特权级	完全访问权限
其余所有固件	非特权级	受 MPU 控制的访问：不可访问、RO、RW

操作系统内核可动态操作 MPU 属性，根据当前运行的任务授予对特定资源的访问权限。每次操作系统由一个任务切换为另一任务时，访问权限可能会更新。

MPU 的适用情况

运行时使用 MPU 隔离敏感代码，并且/或根据器件当前执行的进程管理对资源的访问权限。该功能非常适用于包含安全设计的高级嵌入式操作系统。

提示 *除 STM32F0 系列外，所有 STM32 系列均配备 MPU（更多详细信息请参见各个编程手册）。*

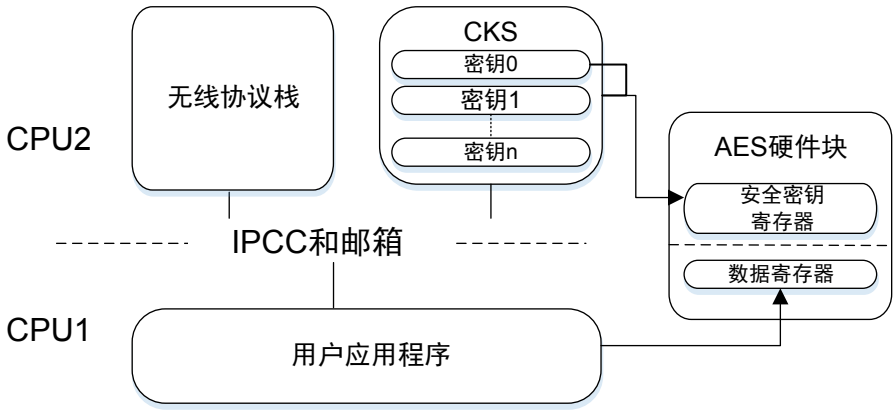
6.10 客户密钥存储（CKS）

STM32WB 系列为双核器件，其中一个内核（CPU1）用于用户应用，另一个内核（CPU2）专用于无线实时执行方面（低功耗蓝牙或线程协议）。CPU2 使用的 Flash 不受 CPU1 或外部访问的影响。通过邮箱和进程间通道控制硬件块（IPCC）确保两个内核之间的通信。

除了执行无线协议线之外，CPU2 还为与专用 AES 硬件块一起使用的加密密钥提供安全存储服务（请参见下图）。该硬件块的密钥寄存器只能由 CPU2 访问，可防止 CPU1 上运行的不可信进程或调试端口访问密钥。

在安全区域内设置密钥之后，用户应用可以使用引用密钥的索引（而不是密钥本身），通过调用安全加载服务来使用。

图 13. 具有 CKS 服务的双核架构



CKS 的适用情况

当用户应用依赖 AES 加密或解密时，必须使用 CKS。预置密钥可以存储在安全区域中，因此其他内部进程或外部访问都无法读取其值。

提示 *CKS 仅在 STM32WB 系列上可用。*

6.11 防篡改（TAMP）/备份寄存器（BKP）

防篡改属于系统级保护，用于检测系统上的物理篡改行为。外部篡改事件通过专用器件引脚的电平转换进行检测。内部篡改传感器可以检查电压、温度或时钟。该事件可用于唤醒内核，以采取相应的动作（例如存储器擦除或报警）。

该外设包含备份域寄存器，其由 VBAT 维持，以及实时时钟（RTC）。如果检测到入侵尝试，这些寄存器可能会复位。

在之前的器件上，该外设被称为备份寄存器（BKP）。在最近的器件上，它具有附加功能，如单调计数器或用于 TrustZone 安全区域的安全部分。

防入侵适用情况

应在系统入侵检测时使用（例如消费产品密封外壳中）。单调计数器可防止篡改 RTC。

提示 *外部篡改检测可用于所有 STM32 系列。*

6.12 时钟安全系统(CSS)

CSS 设计为检测外部时钟源（例如晶振）故障。时钟源缺失可能是有意的，也可能是无意的。无论是哪种情况，器件都必须采取相应的措施进行恢复。在这种情况下 CSS 向内核触发一个中断。

对于外部时钟源驱动主系统时钟的情况，CSS 会将系统切换为内部时钟源。

CSS 的适用情况：

CSS 必须在使用外部时钟的情况下使用。

提示 *CSS 可用于所有 STM32 系列。*

6.13 电源监控

某些攻击可能针对微控制器电源，以引起可能导致安全对策失效的错误。掉电本身可能意味着器件当前正在受到攻击，攻击都通过掉电来尝试冻结器件状态以便后续访问内部存储器内容。

STM32 器件嵌入的可编程电压检测器（PVD）可检测到电压下降。PVD 允许配置电压阈值下限，如果电压低于该值，则会生成中断，以便实施相应的措施。

PVD 的适用情况

如果敏感应用正在运行，并很有可能在工作存储器中保存某些机密数据（SRAM），则须使用 PVD。如果检测到掉电，可启动存储器清除。

提示 *PVD 可用于所有 STM32 系列。*

6.14 存储器完整性硬件检查

错误代码校正（ECC）和奇偶性校验是与存储器内容相关的安全位。ECC 与存储字相关联，通过 ECC 可以从单个错误中恢复或者可以在每个 Flash 或 SRAM 存储字上检测多达两个错误位，存储字可以是 32 位至 256 位，具体取决于存储器类型。简单的奇偶校验可在未实施 ECC 的 SRAM 字中检测单个错误位。有关 ECC 的更多详细信息，例如可在 AN5342 应用笔记中找到。

ECC 和奇偶校验位的适用情况

出于安全考虑，通常使用 ECC 和奇偶校验。ECC 也可用于防止某些侵入性硬件攻击。

6.15 独立看门狗 (IWDG)

IDWG 是自由运行的递减计数器，可用于在计数器达到给定超时值时触发系统复位。IDWG 可用于解决运行代码中的故障或死锁。IDWG 由自己的独立低速时钟（LSI）提供时钟源，因此，即使主时钟发生故障，独立看门狗也会处于工作状态。

IWDG 的适用情况

IWDG 可用于跳出死锁。还可用于控制关键代码的执行时间（例如加密或 Flash 编程）。

提示 *IDWG 可用于所有 STM32 系列。*

6.16 器件 ID

每个 STM32 器件都具有唯一的 96 位标识符，可对任何环境中的任何器件提供单独引用。用户永远不能改变这些位。

器件的唯一标识符可用于直接验证器件身份，也可用于通过主 OEM 密钥导出唯一密钥。

6.17 密码

如第 5 节 所述，加解密算法对于确保嵌入式系统的安全至关重要。加解密算法可确保数据或代码的保密性、完整性和真实性。为了高效支持这些功能，大多数 STM32 系列的 MCU 均提供了硬件加密外设。这些硬件模块可加快加密运算（哈希或对称算法等）的速度。对于没有这种特定硬件加速功能的器件，STM32 加密固件库（CryptoLib）提供了一套大型加密算法的软件实现。

6.17.1 硬件加速器

STM32 器件中提供下列加密硬件外设：

- 真随机数发生器
  - 基于硬件的外设，用于提供物理噪声源。用于生成强会话密钥。

提示 *如需详细了解 TRNG 验证，请参照应用笔记使用 NIST 统计测试套件验证 STM32 微控制器随机数生成（AN4230）。*

- AES 加速器
  - 加密/解密
  - 128 位或 256 位密钥
  - 多种链模式（例如 ECB，CBC，CTR 或 GCM）
  - 支持 DMA
- PKA 加速器
  - 通过 GF(p)操作加速 RSA，DH 和 ECC 算法，实现快速模乘运算
  - 内置 Montgomery 域向内和向外变换
- HASH 加速器
  - MD5, SHA1, SHA224, SHA256
  - 符合 FIPS 规范（FIPS Pub 180-2）
  - 支持 DMA

提示 *如果将 AES 模块用于加密或解密，必须保护访问包含密钥的寄存器，并在使用完后清空（MPU）。*

6.17.2 CryptoLib 软件库

STM32 X-CUBE-CRYPTOLIB 是在所有 STM32 器件上运行的软件库，可访问：www.st.com/en/product/x-cube-cryptolib 免费下载。STM32 X-CUBE-CRYPTOLIB V3.1.5 提供加解密算法的纯软件实现，针对 Cortex M0、M0+、M3、M33、M4 和 M7 内核进行编译。

X-CUBE-CRYPTOLIB 支持以下算法：

- DES、3DES（采用 ECB 和 CBC 模式）
- AES（采用 ECB、CBC、OFB、CCM、GCM、CMAC、KEY 封装、XTS 模式）
- 哈希函数：MD5、SHA-1、HA-224、SHA-256、SHA-384、SHA-512
- 其它：ARC4、ChaCha20、Poly1305、Chacha20-Poly1305
- RSA 签名（采用 PKCS#1v1.5）
- ECC，采用密钥生成、标量乘法（ECDH 的基础）& ECDSA + ED25519 和 Curve 25519

提示 *X-CUBE-CRYPTOLIB V3.1.5 支持所有 STM32 系列。*

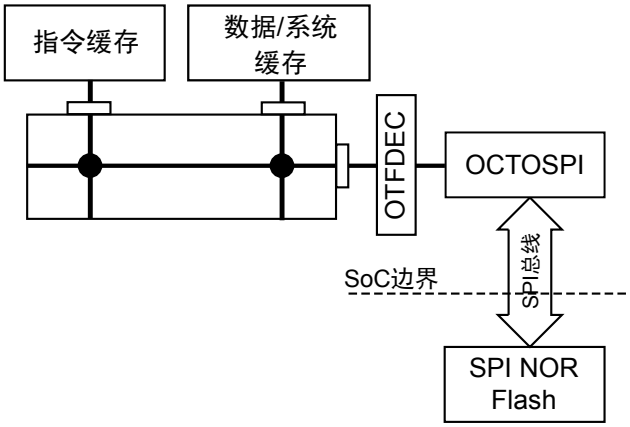
6.18 动态解密引擎（OTFDEC）

外部存储器内容（代码和数据）无法使用传统的读/写保护进行保护。保护其内容的方法是对其进行加密，并在使用之前在器件内对其进行解密。

一种方法是在 SRAM 中下载外部存储器内容，对其进行解密并执行代码和/或使用数据。该方法有以下两个缺点：引入了可能无法接受的延迟；并且根据内容的不同，可能会使用大量的 SRAM。

OTFDEC 可以以低延迟损失直接解密内容，无需分配 SRAM。OTFDEC 是一个可以基于读取请求地址信息动态解密的硬件模块。它与 Octo-SPI 接口一起使用（请参见下图）。

图 14. SoC 中 OTFDEC 的典型用法



OTFDEC 采用 AES-128 的 counter 模式，并使用 128 位密钥来实现 12 个 AHB 周期以内的延迟。最多可以定义四个独立且不重叠的加密区域（4 KB 粒度），每个区域都有自己的密钥。

OTFDEC 的适用情况

OTFDEC 适用于当系统采用外部存储器的情况。若 MCU 是带 TrustZone 功能，此时仅在安全模式下可以设置解密对应的密钥。请参见 AN5281 以获得更多信息。

提示 *OTFDEC 仅在 STM32L5 和 STM32H7 系列上可用。*

7 指南

安全系统可以利用许多支持安全性的硬件功能。一些可用于所有系统，只需少量更改应用程序代码即可激活并完全起作用。RDP 特性便是其中的一种安全机制，可通过禁用调试端口阻止对 Flash 的基本访问。须根据用户应用以及要求达到的安全级别选择其它特性。

本节将根据系统用例帮助定义适配的安全功能集。

用例主要分为四组：避免外部（1）和内部（2）威胁、安全维护（3）以及其它与加密相关的用例（4）（参见下表）。

表 15. 安全用例

1 保护器件免受外部威胁：RDP 保护、入侵检测、器件监控。	
-	1.1 器件配置（选项字节，始终不应修改）
	• 使用 RDP 级别 2。此操作可防止设备受到任何外部访问。
	1.2 禁用器件的调试功能。
	• 使用 RDP 级别 2 可永久性禁用调试。
	1.3 防止器件丢失外部时钟源（晶振）。
	• 使能时钟安全系统（CSS）。
	1.4 检测系统级入侵。
	• 使用 RTC 外设的入侵检测功能。
	1.5 保护器件免受代码注入。
	• 使用 RDP。 • 通过 MPU、防火墙或 HDP 隔离通信端口协议。 • 限制通信端口协议访问范围。 • 对空存储区（Flash 和 SRAM）使用写保护。
2. 保护代码免受内部威胁：TrustZone、PCROP、MPU、防火墙和 HDP	
-	2.1 保护代码免受克隆。
	• 使用 RDP 级别 1 或 2 防止代码受到外部访问。 • 对代码最敏感的部分使用 PCROP，使其免受内部访问。 • 使用 OTFDEC 保护存储在外部存储器中的代码。
	2.2 如何保护保密数据免受其它进程的访问
	• 使用防火墙保护代码和数据。 • 使用 MPU 防止保密数据区被读取。 • 如果数据仅应在复位时使用，则使用 HDP。 • 如果可用，则使用 TrustZone 的安全域。
	2.3 使用未完全通过验证或完全受信任的库时保护代码和数据。
	• 使用 PCROP 保护用户最敏感的代码。 • 使用防火墙保护用户敏感应用（代码、数据和执行）。 • 使用 MPU 并取消不受信库的权限。 • 使用 IWDG 避免任何死锁。 • 如果可用，则使用 TrustZone 的安全域。
3. 器件安全检查与维护：完整性检查、SB、SFU	
-	3.1 检查代码完整性。
	• 将固件代码进行哈希处理，并与预期值进行比较。 • 在 Flash 上启用 ECC，在 SRAM 上启用奇偶校验。
	3.2 安全检查或嵌入式固件身份验证

-	• 通过加密实现 SB。 • 保护 SB 应用的保密数据（请参考之前的章节）。 • 保证 SB 应用的唯一启动入口： – 使用 HDP（若提供）。 – 使用 RDP 级别 2 并禁用启动引脚选择。
	3.3 现场安全固件更新。
	• 通过加密实现 SFU 应用。 • 针对 SFU 保密数据应用相关安全存储器保护（请参考之前的章节）。
4. 通信和身份验证：加密	
-	4.1 安全通信。
	• 使用或者实现一套依赖于保密性和真实性的安全通信协议栈（例如为以太网使用 TLS）。
	4.2 在 STM32 上使用意法半导体提供的 AES/DES/SHA 加解密算法函数。
	• 仅使用意法半导体官方提供的 STM32 X-CUBE-CRYPTOLIB。
	4.3 AES/DES/SHA 加密功能加速。
	• 采用带硬件加密外设的器件并配合官方 STM32 X-CUBE-CRYPTOLIB 一起使用。 • 使用 OTFDEC 访问外部存储器中的 AES 加密代码，而不会出现延迟损失。
	4.4 生成真随机数据。
	• 使用嵌入在 STM32 器件中的硬件真随机数发生器。
	4.5 唯一标识 ST 微控制器。
	• 使用 STM32 微控制器的 96 位 UID。
	4.6 对产品器件进行身份验证。
	• 在器件中嵌入共享密钥并交换加密报文。
	4.7 对产品设备进行唯一身份验证。
	• 在器件植入私钥和证书并交换密文。
	4.8 对通信服务器进行身份验证。
	• 在器件中嵌入共享密钥并交换加密报文。 • 在器件中嵌入公钥并交换加密报文。

8 结论

仅通过在硬件中启用安全功能的方法无法保证系统的安全性。安全性必须根植于整套解决方案的架构之中。

应识别出威胁，并与其他安全功能协同地正确设计出对策并实施。

由于安全性需要使用大量资源，因此重要的是要正确评估风险并有效使用资源，同时牢记攻击成本以及受保护资产的价值。

信任根的概念非常有用，因为它使用了一种分析功能更强大的方法，这与一切都取决于其他一切的整体方法不同。

使用 STM32 系列微控制器，内置物联网安全性可以达到较高的经济性和高效性。

附录 A 加解密算法 - 主要概念

完整性、身份验证和保密性

加密的目的有三个：

- 保密性：保护敏感数据免受未授权的读访问
- 身份验证：保证消息发送方的身份
- 完整性：在传输过程中检测任何消息损坏

为了达到这些目的，所有安全数据流或多或少依赖于以下算法的复杂组合：

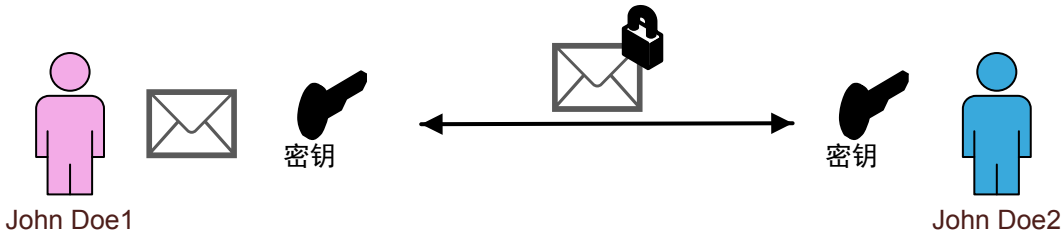
- 密钥/对称加密算法
- 公钥/非对称加密算法
- 哈希算法

下面几节将描述这些算法。

A.1 密钥算法

此系列算法使用发送方和接收方之间共享的密钥将明文加密，从而确保保密性。由于加密和解密使用相同密钥，此技术被称为对称加密。

图 15. 对称加密算法



这类算法的固有弱点是双方共享密钥。这在安全环境（例如，制造工厂）中可能不是问题，但是当双方距离遥远时，密钥传输将面临挑战。

在所有密钥算法中，基于块的算法十分常见，因为它们可以通过硬件或软件的并行实现进行高效加速。典型的 AES（高级加密标准）算法基于 128 位的明文分组。它们使用 128、192 或 256 位密钥生成长度相同的加密分组。链接连续分组的不同方式被称为“运算模式”。其中包括密码分组链接模式（CBC）、计数器模式（CTR）和伽罗瓦计数器模式（GCM）。

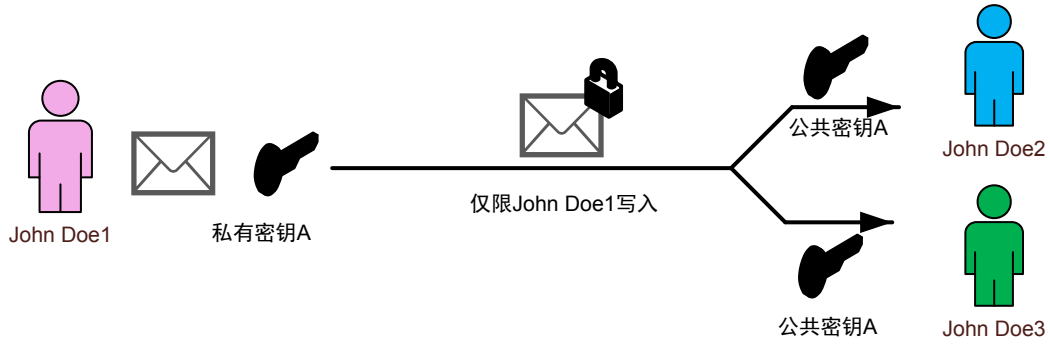
由于这些算法都是确定的，它们总是将输入数据与随机值（称为随机数）混合，随机数作为初始向量仅用于一次会话。

A.2 公钥算法（PKA）

这类算法基于一对密钥。其中的私钥绝不会与任何远程系统进行交换，而另一个公钥则可与任何方面共享。两个密钥之间的关系是不对称的（不对称加密）：

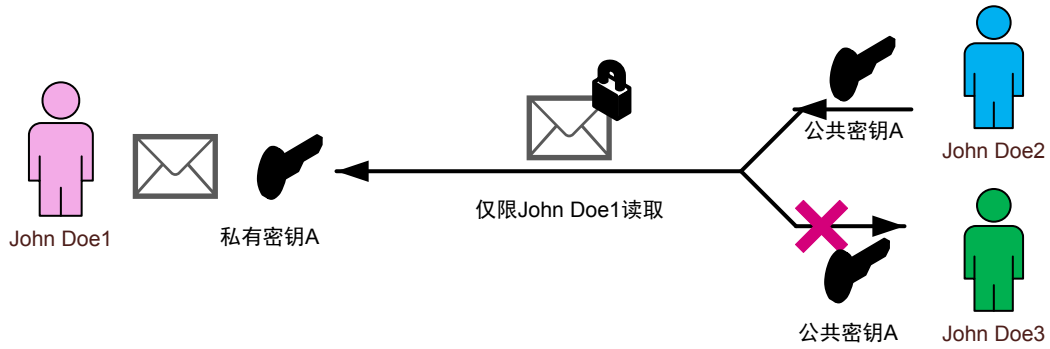
- 通过私钥加密的消息可以被拥有公钥的任意方读取。此机制确保了发送方的严格身份验证，因为私钥从未被共享。数字签名就是基于这一机制。

图 16. 签名



- 通过公钥加密的消息只能被私钥所有者读取。

图 17. 公钥加密



公钥算法的主要用途是进行身份验证。

公钥算法还用于解决对称加密的“密钥共享”问题。但是，其代价是运算更加复杂，增加了计算时间和内存占用量。

RSA 和椭圆曲线加密（ECC）是最常用的非对称算法。

混合加密

常用的安全传输协议（例如，Bluetooth 和 TLS）同时依赖于两种类型的算法。这种机制被称为混合加密：

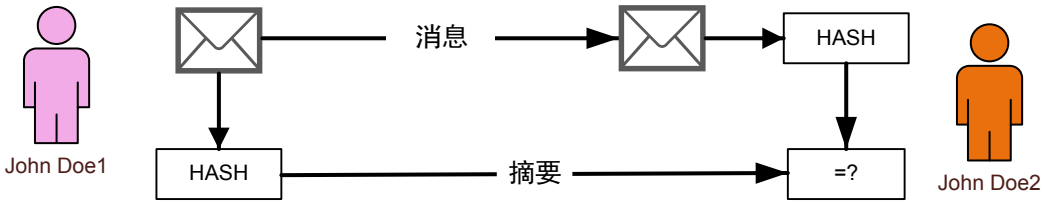
- 不对称加密最初用于解决对称密钥共享问题。由公钥所有者将会话密钥传输给私钥所有者。
- 然后，通过对称算法使用会话密钥提供传输保密性。

A.3 哈希算法

哈希算法可保证消息的完整性。它们从消息生成唯一的固定长度的比特流，称之为摘要。输入消息中的任何差异都会导致生成的摘要完全不同。摘要不能逆向用于检索输入消息。

哈希算法可独立于消息加密单独使用。

图 18. 消息哈希



它与传统 CRC 的区别在于更复杂的运算和大得多的摘要长度（最长 512 位，而不是 16 或 32 位）带来的稳健性。例如，CRC 被保留用于数据传输期间的快速完整性检查。摘要长度几乎实现了唯一性，并确保不会发生冲突。

典型的算法是 MD5（128 位摘要）、SHA-1（160 位摘要）、SHA-2（224、256、384 或 512 位摘要）和 SHA-3（224、256、384 或 512 位摘要）。

A.4 MAC 或签名和证书

MAC 和证书

消息认证码（MAC）和签名对消息进行哈希加密，在检验其完整性的基础上增添了身份认证功能。MAC 与签名不一样的地方在于 MAC 使用了一个对称密钥算法来生成 MAC（图 19），而签名则是使用发送者的私钥来生成签名（图 20）。

签名增加了身份验证的不可否认性：

- 私钥不应撤回（其使用生命周期要远大于当前传输操作周期），而密钥的使用寿命则是有限的（仅限此次传输期间）。
- 用于签名的私钥绝不会共享，其安全性要高于密钥。

图 19. 通过密钥算法生成 MAC

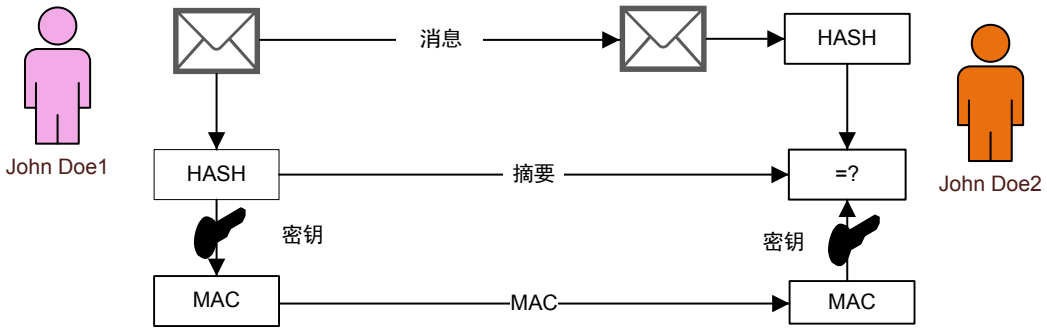
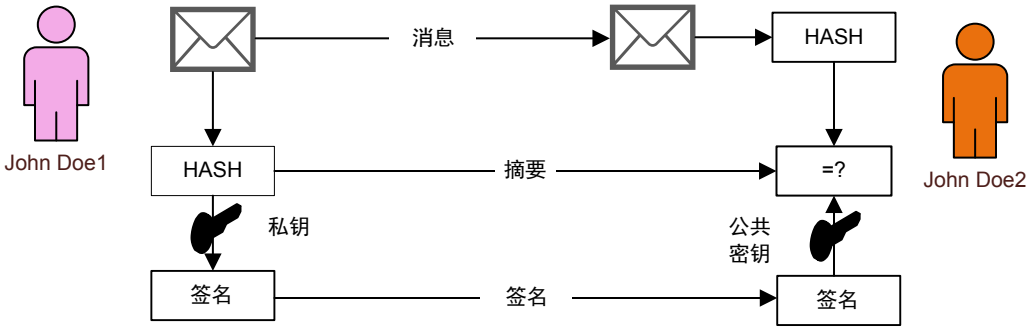


图 20. 通过公钥算法生成签名



证书

证书与公钥算法有关。它对不对称加密中的公钥进行验证。它验证非对称传输中的公钥。它用来对抗攻击者的非法操作，比如意图使用攻击者自己的假冒公钥替代正确的公钥。一个证书包含了经过证书授权中心（CA）私钥签名的公钥。此 CA 被认为完全可信。

除了公钥，证书还包含版本号、有效期和一些 ID。

版本历史

表 16. 文档版本历史

日期	版本	变更
2018 年 10 月 17 日	1	初始版本。
2019 年 2 月 25 日	2	更新了： • 表 1.适用产品 • 第 1 节 概述 • 表 11.STM32H7、STM32G0、STM32G4 和 STM32WB 系列的安全特性 • 图 9.RDP 保护示例（STM32L4 系列） • 第 6.6 节 防火墙 增加了： • 第 6.8 节加密密钥存储（CKS）
2019 年 10 月 7 日	3	更新了： • *表 1.适用产品* • 本节介绍重命名为“概述” • *表 2.词汇表* • 本节硬件保护重命名为器件保护 • *图 4.存储器类型* • *表 5.存储器类型和相关保护功能* • *第 4.2.4 节 外部 Flash 存储器* • *表 6.STM32 嵌入式存储器保护特性的范围* • *表 7.软件隔离机制* • *第 4.5 节 启动保护* • *第 5 节 安全应用程序：表 9、表 10 和表 11* • *第 6.2 节 读保护（RDP）* • *第 6.7 节 安全隐藏保护（HDP）* • *第 6.17 节 加密* • *第 7 节 指南* • 删除了所有图形上的某些颜色 增加了： • *第 4.1 节 面向 Armv8-M 架构的 TrustZone®* • *第 6.4 节 TrustZone* • *第 6.18 节 动态解密引擎（OTFDEC）*
2020 年 2 月 21 日	4	• 更新了“*前言*”一节 • 在*表 2 中增加了首字母缩略词。词汇表* • 更新了*第 2 节 概述*和*第 3 节 攻击类型* • 对*第 3.4 节 物联网系统攻击示例*进行了结构调整（增加了*第 3.5 节 攻击目标列表*） • 更新了*第 4 节 设备保护* • 对*第 5 节 安全应用程序*进行了更新和结构调整 • 增加了*第 5.3 节 Arm TF-M 解决方案* • 更新了*第 6 节 STM32 安全特性*、*第 7 节 指南*，以及*第 8 节 结论*
2020 年 11 月 6 日	5	更新了： • 文档范围，增加了 STM32WL 系列 • *表 1.适用产品* • *第 1 节 概述* • *第 3.1 节 攻击类型简介* • *第 3.2 节 软件攻击* • *第 3.3.1 节 非侵入式攻击* • *第 3.3.2 节 硅片侵入性攻击*

日期	版本	变更
		• *第 4.1 节 面向 Armv8-M 架构的 TrustZone®* • *表 5.存储器类型和相关保护功能* • *第 5.3 节 Arm TF-M 解决方案* • *表 8.基本功能差异* • *第 6.1 节 安全特性概述（包括所有表格更新）* • *第 6.2 节 读保护（RDP）* • *第 6.4 节 TrustZone* 增加了： • *第 4.2 节 双核安全* • *第 6.3 节 一次可编程(OTP)*
2021 年 7 月 7 日	6	更新了： • 文档范围，增加了 STM32U5 系列 • 表 1. 适用产品 • 第 3.3.1 节 非侵入性攻击 • 第 4.3.3 节 内部 SRAM • 第 4.3.4 节 外部 Flash • 第 5 节 安全应用 • 表 9. STM32Fx 系列的安全特性 • 表 10. STM32Lx 和 STM32Ux 系列的安全特性 • 表 11. STM32H7、STM32G0、STM32G4、STM32WB 和 STM32WL 系列的安全特性 • 第 6.3 节 一次可编程(OTP) • 第 6.6 节 只执行固件(PCROP) • 第 6.8 节 防火墙 • 第 6.9 节 存储器保护单元(MPU) • 第 6.17 节 密码 • 第 6.17.1 节 硬件加速器 • 第 6.17.2 节 CryptoLib 软件库 增加了： • 第 5.4 节 产品认证

目录

1 概述 2
2 概述 4
2.1 安全保护的目的 4
3 攻击类型 6
3.1 攻击类型介绍 6
3.2 软件攻击 7
3.3 硬件攻击 9
3.3.1 非侵入性攻击 9
3.3.2 硅片侵入性攻击 10
3.4 IoT 系统攻击示例 10
3.5 攻击目标清单 11
4 器件保护 14
4.1 用于 Armv8-M 架构的 TrustZone® 14
4.2 双核安全 14
4.3 存储器保护 15
4.3.1 系统 Flash 17
4.3.2 用户 Flash 存储器 17
4.3.3 内部 SRAM 17
4.3.4 外部 Flash 18
4.3.5 STM32 存储器保护概述 18
4.4 软件隔离 19
4.5 调试端口及其它接口保护 19
4.6 启动保护 19
4.7 系统监控 19
5 安全应用 21
5.1 信任根和信任链 21
5.2 意法半导体专有 SBSFU 解决方案 21
5.2.1 安全启动（SB） 21
5.2.2 安全固件更新（SFU） 22
5.3 Arm TF-M 解决方案 23

5.4 产品认证 24
6 STM32 安全特性 25
6.1 安全特性概述 25
6.2 读保护 (RDP) 28
6.3 一次可编程(OTP) 30
6.4 TrustZone 30
6.4.1 内核状态 31
6.4.2 安全属性单元（SAU） 31
6.4.3 存储器和外设保护 31
6.5 Flash 写保护（WRP） 32
6.6 只执行固件(PCROP) 32
6.7 安全隐藏保护（HDP） 32
6.8 防火墙 33
6.9 存储器保护单元(MPU) 34
6.10 客户密钥存储（CKS） 35
6.11 防篡改（TAMP）/备份寄存器（BKP） 36
6.12 时钟安全系统(CSS) 36
6.13 电源监控 36
6.14 存储器完整性硬件检查 37
6.15 独立看门狗 (IWDG) 37
6.16 器件 ID 37
6.17 密码 37
6.17.1 硬件加速器 38
6.17.2 CryptoLib 软件库 38
6.18 动态解密引擎（OTFDEC） 38
7 指南 40
8 结论 42
附录 A 加解密算法 - 主要概念 43
A.1 密钥算法 43
A.2 公钥算法（PKA） 44
A.3 哈希算法 44

A.4 MAC 或签名和证书 45
Revision history 47

表一览

表 1. 适用产品 . . . 1
表 2. 词汇表 . . . 2
表 3. 要保护的资产 . . . 5
表 4. 攻击类型和成本 . . . 7
表 5. 存储器类型和相关保护功能 . . . 17
表 6. STM32 嵌入式存储器保护特性的范围 . . . 18
表 7. 软件隔离机制 . . . 19
表 8. 基本功能差异 . . . 23
表 9. STM32Fx 系列的安全特性 . . . 26
表 10. STM32Lx 和 STM32Ux 系列的安全特性 . . . 27
表 11. STM32H7、STM32G0、STM32G4、STM32WB 和 STM32WL 系列的安全特性 . . . 28
表 12. RDP 保护 . . . 30
表 13. 由 MPU 管理的属性和访问权限 . . . 35
表 14. 进程隔离 . . . 35
表 15. 安全用例 . . . 40
表 16. 文档版本历史 . . . 47

图一览

图 1. 已破坏的连接设备的威胁 4
图 2. IoT 系统 11
图 3. Armv8-M TrustZone 执行模式 14
图 4. 简化的双核系统架构图 15
图 5. 存储器类型 16
图 6. 安全启动流程图 22
图 7. 安全的服务器/设备 SFU 架构 23
图 8. RDP 保护示例（STM32L4 系列） 29
图 9. 在系统级实现 TrustZone 31
图 10. HDP 保护的固件访问 33
图 11. 防火墙 FSM 33
图 12. 防火墙应用示例 34
图 13. 具有 CKS 服务的双核架构 36
图 14. SoC 中 OTFDEC 的典型用法 39
图 15. 对称加密算法 43
图 16. 签名 44
图 17. 公钥加密 44
图 18. 消息哈希 45
图 19. 通过密钥算法生成 MAC 45
图 20. 通过公钥算法生成签名 45