AN12291

内置 Flash 的 NXP LPC MCU 的软件知识产权(IP)绑定

版本 1.0 — 2018 年 11 月 6 日 应用笔记

目录

1 简介

本应用笔记描述了保护 LPC 32 位 ARM 微控制器(MCU)上的嵌入式应用程序免受逆向工程、未经授权的固件篡改、过多生产和假冒的技术。通过将固件和配置信息绑定到 MCU 的 SRAM 的唯一物理属性来提供保护。软件知识产权（IP）通过使用从物理不可克隆功能(PUF)派生的加密密钥的加密过程绑定到特定的设备。由于这些密钥是不可克隆的，而且是设备唯一的，所以存储的加密信息只能由授权的设备读取。这种设备独有的加密使攻击者难以逆向工程或修改固件。整个过程支持：

- 设备唯一的、不可克隆的标识，可以通过加密方式进行身份验证。
- 代码认证以确保固件只能由 OEM 安装和更新。
- 设备独有的加密密钥，以保护存储的信息不被克隆、反向工程或修改。
- 可选版本号验证，以确保只安装新的固件更新，以防止回滚攻击。
- 可以保护其他机密信息以支持应用程序服务。此信息可以是附加密钥、信任根或配置信息。该敏感数据具有与固件相同的强设备独有保护，并强力防止提取。
- PUF API 还可以生成额外的密钥，以支持使用对称密钥、公钥和/或公钥证书的强设备身份验证。

介绍了利用 PUF 设计和实现安全嵌入式应用程序的过程。该解决方案是基于软件的，但对系统设计和制造测试过程有影响。提供软件模块与生产测试设备和应用程序集成。可以使用 IP 绑定工具套件来保护固件，并用于创建工厂安装的映像和用于空中升级的映像。

1.1 物理不可克隆功能

就像雪花一样，每个半导体器件的原子结构都是不同的。这些原子结构上的差异在未初始化 SRAM 的通电状态中表现出来，它为每个设备形成一个独特的模式，就像硅指纹。从 SRAM PUF 派生的密钥不存储在芯片上，但只在需要时从芯片中提取。它们只在短时间内存在于芯片中。当 SRAM 没有通电时，芯片上就没有密钥。它使解决方案更加安全，免被逆向工程和提取密钥。

1.2 IP 绑定概述

软件 IP 与设备绑定时会使用 UDS (Unique-Device Secret)对存储的信息进行加密。UDS 密钥源于 SRAM PUF，提供安全保障的基础。UDS 还可以用于获取设备的唯一标识符，用于设备的识别和认证。可以从这个初始信任根（RoT）派生出各种各样的密钥（包括对称的和不对称的），供嵌入式应用程序使用。本应用说明主要介绍 UDS 在 IP 绑定解决方案中的应用，对 OEM 固件和配置信息进行保护。开发人员还可以将基于 PUF 的安全性用于其应用程序中的其他安全性需求。

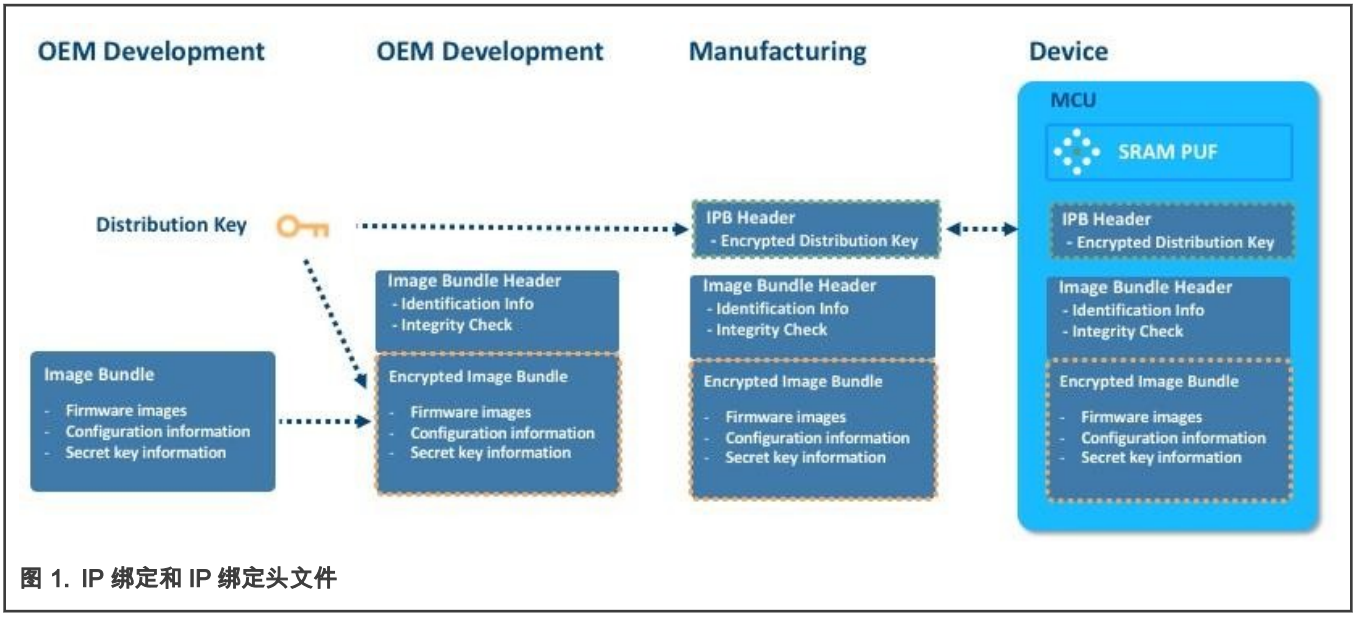

图 1. IP 绑定和 IP 绑定头文件

将映像包绑定到设备的加密过程如 图 1 所示。OEM 创建一个固件映像，并可能将映像与额外的机密和配置信息捆绑在一起。映像由 OEM 使用对称分布密钥进行保护。加密过程向映像包添加一个头文件，该头文件标识受保护的包，并提供完整性检查，在解密时验证完整性检查。在生产设备中，通过创建 IP 绑定头文件，每个设备都被唯一授权解密绑定包。此头包含以设备的 SRAM PUF 唯一生成的密钥加密的分发密钥。为了保护制造中的威胁，分发密钥应该安全地存储和管理在硬件安全模块（HSM）中。由于头文件是使用 PUF 派生的密钥创建的，因此它封装的密钥不会被绑定设备以外的任何设备提取或使用。

对于同一型号的所有设备，头文件的分配密钥将被设置为相同的密钥。它允许将相同的加密映像包安装到一系列设备中。映像中 IP 的安装和分发受到保护，后续可能使用相同的分发密钥创建升级映像。

映像包中的头文件包含一个版本号。升级过程可能会检查这个版本号，并强制执行单调递增的版本号，以防止回滚攻击。

IP 绑定头文件也可以配置为每个设备包含一个唯一的分发密钥。它提供了控制将映像分发到单个设备的能力。

2 实现 IP 绑定

本节提供其他详细信息和示例代码片段，以帮助开发人员创建和部署 IP 绑定解决方案。

2.1 设计

PUF 设备的唯一密钥为系统设计提供了一个强大的信任根(RoT)。要使用 PUF，嵌入式应用程序必须在引导过程中使用未初始化的 SRAM 来初始化和提取密钥。使用 PUF 的 SRAM 内存分配的一个例子如 图 2 所示。

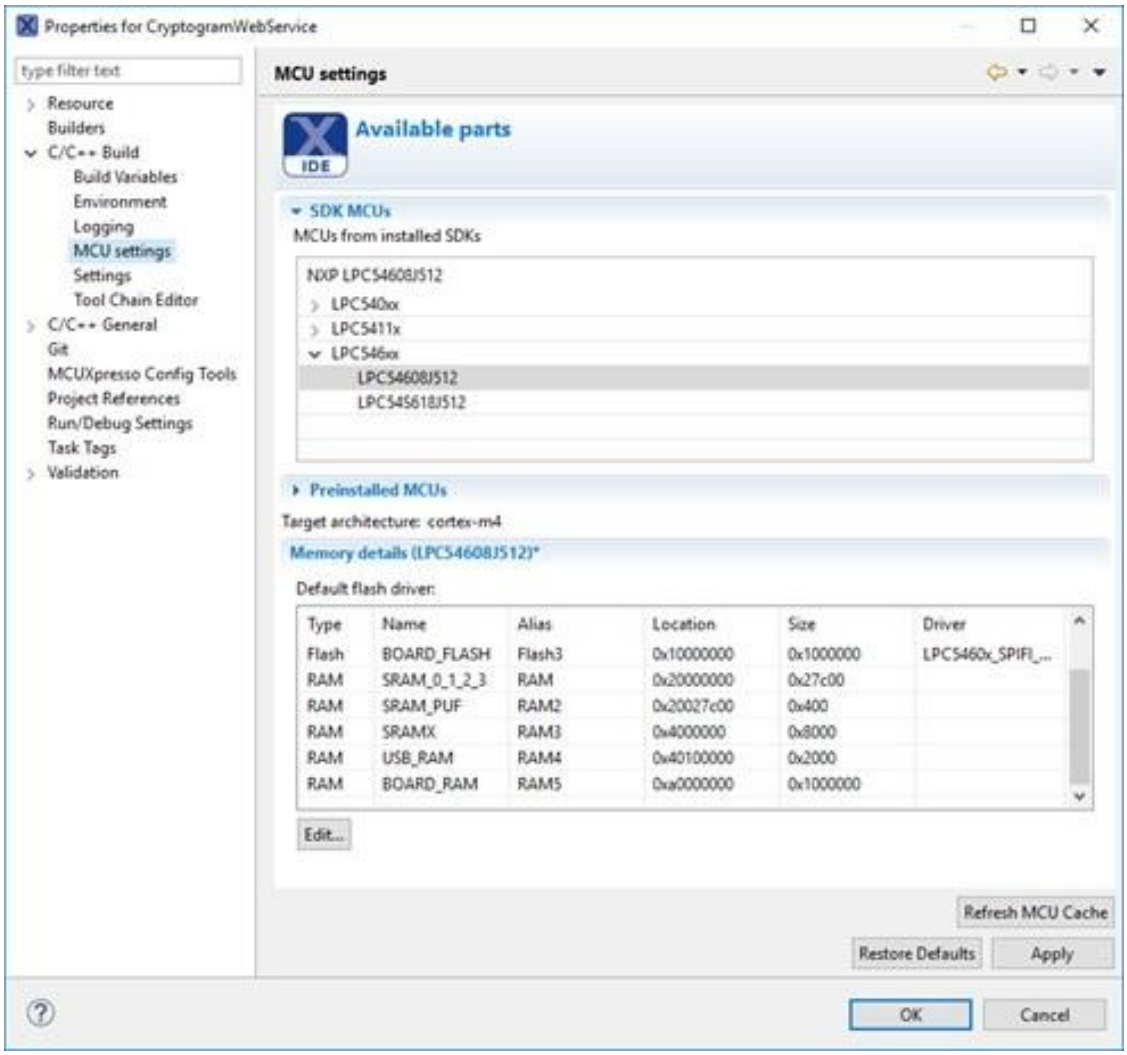

图 2. SRAM PUF 内存分配

设计还必须确保单片机的内存使用和 OTP 设置与安全设计相适应。设备的安全性取决于其整体配置，设计师应确保正确使用 NXP LPC MCU 提供的锁定机制来保护系统。SWD 和调试功能必须通过适当的 OTP 设置禁用。

2.2 创建受保护的映像包

使用正常的构建流程创建和测试映像。在对映像进行保护之前，它可能与其他配置和密钥信息捆绑在一起。它允许应用程序唯一的信任根和身份验证密钥安全地传输并与固件映像一起安装。

映像包使用 IP 绑定（IPB）工具通过分发密钥加密。IP 绑定工具套件是一个命令行工具，用于保护固件映像和提供额外的密钥。在 OEM 的构建/发布过程中，它被用于软件工具链。此外，它还可以在无线更新服务器上使用，对更新后的代码映像进行实时保护。

该工具套件使用来自单个芯片的 IPB_HEADER 来提取其配置的保护配置文件（认证、加密、防回滚），以便选择要应用的保护选项。然后，它接受一个不受保护的代码映像和可选的附加密钥，并使用分发密钥保护代码和密钥，从而产生一个受保护的代码映像。后者由 IPB 固件处理。当防回滚激活时，该工具从文件中读取 32 位的映像版本号，并增加该文件中的版本号。

2.3 OEM 设备配置

在 OEM 设备中，通过使用用于测试和编程的制造工具来实现产品的个性化；如 图 3 所示。当没有被充分信任的生产设施来保护主分发密钥时，使用主分发密钥的加密处理应该在硬件安全模块（HSM）中进行。

制造工具通过向设备中注入并运行注册映像来启动个性化流程。这段代码是暂时的，只运行一次。

注册映像的功能包括：

- 设备测试功能。
- PUF 初始化和设备唯一密钥创建。
- 提取设备唯一公钥。
- 外部 flash 中 IP 绑定头文件设备的创建和存储。
- 加载受保护的映像包到 flash。
- 获取包装的固件分发密钥，并将设备唯一的 IPB 头写入 NV 内存。
- 锁定设备以防止使用 SWD 和边界扫描，并防止设备的任何后续供应。

注册映像是特定于单片机的，可以包括特定于应用程序的测试和初始化。PUF 初始化和 IP 绑定头功能必须与现有的注册映像软件集成。注册映像的代码示例为这个集成提供了 PUF 初始化和 IP 绑定函数的示例代码。

PUF 唯一密钥用于为设备创建一个唯一标识符。它可以被提取和存储，以便后续跟踪产品。

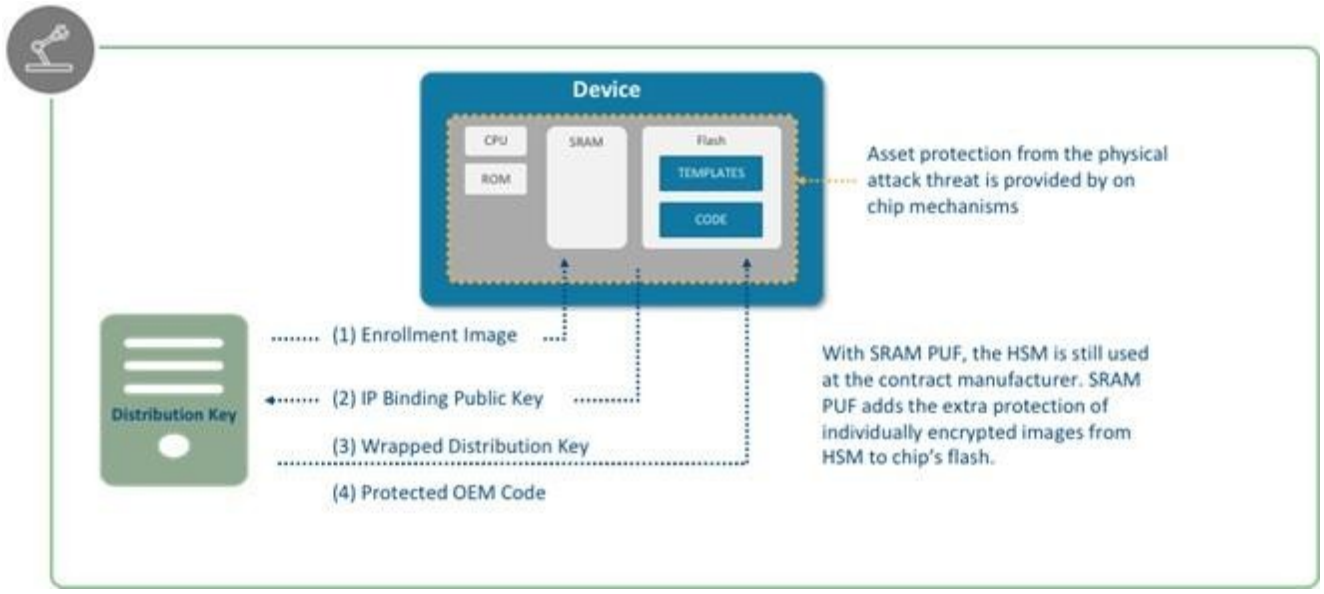

图 3. OEM 设备配置

2.4 使用 SRAM PUF 安全启动

在启动时，设备使用 SRAM PUF 创建设备唯一的密钥。然后，该设备固件加密密钥用于解密 IP 绑定头文件，IP 绑定头文件提供用于解密和验证映像包的分发密钥，然后运行未包装的固件。如 IP 绑定固件所述，IP 绑定固件模块支持这些功能的集成，该模块提供所有必需的 PUF 和加密功能和加密功能。

安全引导的代码示例提供了演示如何集成安全引导进程的示例代码。

3 格式规范

本节介绍 IP 绑定头文件和映像绑定头文件的格式。IP 绑定头文件使用 PUF 派生的密钥加密，该密钥对单个设备是唯一的。映像包头文件对于加密的固件映像是唯一的。加密的映像包可以与多个设备一起使用，并且每个设备都可以通过给它一个唯一的 IP 绑定头文件来启用这个包，这个头包含一个为特定设备加密的公共分发密钥。

3.1 IP 绑定头文件

IP 绑定头文件是设备唯一的字节序列，它支持对提供的固件映像进行解密。头文件包含三个字段：

- 描述是否需要防回滚保护的安全选项。
- 加密了的分配密钥。
- 消息认证码 MAC (Message Authentication Code)，用于提供强大的安全性完整性检查。

这些字段的格式描述如下：

```
IPB_HEADER
{
  uint32_t security_options; /* anti_rollback flags */
  uint8_t encrypted_distribution_key[32];
  uint8_t IPB_HDR_MAC[32]; /* MAC of prior two fields using PUF derived key*/
}
```

这些选项是在创建 IP 绑定头文件时设置的。这些选项包括回滚策略和支持在调试和开发中使用空加密的能力。示例代码提供了示例代码和用法。

3.2 映像包头文件格式

映像包头文件的格式描述如下：

```
IMG_HEADER
{
  security_version[4]
   version_counter[4]
   image_size[4]
   additional_keys_size[4]
   jump_offset[4]
   hash_encrypted_additional_keys[32] /* SHA256 */
   hash_encrypted_image[32]    /* SHA256 */
   IMG_HDR_MAC[32] /* MAC of everything up to here */
 }
```

开发人员可以使用存储在受保护映像包中的附加密钥来保护信任根和特定于应用程序的机密。

4 IP 绑定固件

IP 绑定固件是解决方案的设备端，在单个固件模块中实现，该模块编译为针对目标平台的二进制库。IP 绑定固件对 SRAM 进行身份验证和解密，并防止回滚二进制代码映像和附加密钥。认证、解密、回滚保护是可以通过 IP 绑定工具单独激活或去激活的功能，参见 IP 绑定工具包和被编程到设备中的 IPB_HEADER 数据结构。固件映像在内部与设备唯一的密钥源接口。

5 IP 绑定工具说明

5.1 IP 绑定工具包

IP 绑定工具套件 v1.0 是一个命令行工具，用于保护固件映像和提供附加密钥。在 OEM 的构建/发布过程中，它被用于软件工具链。此外，它还可以在无线更新服务器上使用，对更新后的代码映像进行实时保护。

该工具套件使用来自单个芯片的 IPB_HEADER 来提取其配置的保护配置文件（认证、加密、防回滚），以便选择要应用的保护选项。然后，它接受一个不受保护的代码图像和可选的附加密钥，并使用分发密钥保护代码和密钥，从而产生一个受保护的代码图像。后者由 IP 绑定固件处理。当防回滚激活时，该工具从文件中读取 32 位的映像版本号，并增加该文件中的版本号。

6 示例代码

下面的示例提供了适合用于利用 SRAM PUF 将软件 IP 绑定到设备的 PUF 设备密钥开发安全 MCU 产品的代码。

6.1 注册映像的代码示例

以下代码支持在初始设备注册期间初始化和使用基于 PUF 的密钥。

注册映像包括以下内容:

- 烧写集成了 IPB 固件的固件映像到单个芯片的 NVM。
- 启动芯片并调用 bk_init()，然后调用 bk_enroll()来生成芯片唯一和注册唯一的激活码（AC）。
- 使用安全配置文件配置选项调用 ipb_bind_header()，以生成芯片唯一的 IPB_HEADER。

> **注意**
>
> 再次注册将产生一个不同的 AC，不能与不匹配的 IPB_HEADER 一起使用——ipb_bind_header()需要在新的 AC 中重复使用。

- AC 和 IPB_HEADER 存储在片上 NVM 中。
- 可选地，AC 和 IPB_HEADER 被发送到芯片外部以存储在数据库中，用于备份恢复或没有 NVM 的基于服务器的激活。
- 永久启用适合使用案例的硬件访问控制机制。例如禁用 SWD/debug 接口、禁用外部 flash 写和擦除功能、更改 NVM 变量以禁用实现芯片注册流程的命令。

下面的代码应该与制造测试系统注入设备的测试和初始化软件集成在一起。

```
18    include/iidbroadkey.h
19    #define BK_SRAM_PUF_SIZE_BYTES    (1024)
20    #define BK_AC_SIZE_BYTES    (788)
21
22      /*
23    * REQUIREMENT:
24    *    Modify the Memory Map to allocate SRAM slice (RAM2)
25    *    (identify/allocate/isolate SRAM memory)
26    */
27        static __attribute__ ((section(".noinit.$RAM2"))) uint8_t sram_puf[BK_SRAM_PUF_SIZE_BYTES];
28        static uint8_t activation_code[BK_AC_SIZE_BYTES]__attribute__ ((aligned (4)));
29
30
31    APP_PUF_store_activation_code( activation_code ) - Application dependent code to write
Activation Code (ex. Non-volatile memory)
32
33    APP_IPB_load_distribution_key( distribution_key ) - Application dependent function to provide
Application dependent Distribution Key
34
35    APP_IPB_store_counter( 0 )    - Initialize Monotonic Counter 36
37
38
39    /* Initialize SRAM dedicated for PUF */
40    ret_code = bk_init( sram_puf, BK_SRAM_PUF_SIZE_BYTES );
41    if (IID_SUCCESS != ret_code)    break; 42
43    ret_code = bk_enroll( activation_code );
44    if (IID_SUCCESS != ret_code)    break; 45
46    /* Application-Specific method for storage (i.e. flash, OTP)
47    ret_code = APP_PUF_store_activation_code( activation_code );
48    if (IID_SUCCESS != ret_code)    break; 49
50        /* Application-Specific method for Distribution Key management (cryptogram)
51    ret_code = APP_IPB_load_distribution_key( distribution_key );
52    if (IID_SUCCESS != ret_code)    break; 53
```

```
54    ipb_header_t    ipb_header;
55    ret_code = ipb_bind_header(
56        (IP_BIND_AUTHENTICATE | IP_BIND_ENCRYPT | IP_BIND_NON_ROLLBACK),
57    distribution_key,
58    (ipb_header_t * const )&ipb_header );
59    if (IID_SUCCESS != ret_code)    break; 60
61
62    /* Initialize Monotonic counter
63    * Application-Specific method for storage (i.e. NVRAM) 64    */
65    ret_code = APP_IPB_store_counter( 0 );
```

6.2 安全引导的代码示例

下面的代码演示了如何使用和集成 IP 绑定固件。此代码集成到引导过程中，以支持 PUF 初始化过程和受保护固件的解密和验证。

```
66    static __attribute__ ((section(".noinit.$RAM2"))) uint8_t sram_puf[BK_SRAM_PUF_SIZE_BYTES];
67
68    #typedef struct PUF_boot_params {
69    const ipb_header_t * const ipb_header;
70    const uint8_t    * const image_address;
71    uint8_t    * const image_target_address;
72    uint8_t    * const additional_keys_target_address;
73    uint32_t    * const version_counter;
74    uintptr_t    * const execution_address;
75    } PUF_boot_params_t; 76
77    PUF_boot_params_t    params;
78    Bool    bool_counter_updated;
79
80
81
82    APP_IPB_load_params( &params ) – Application dependent code to load boot params 83
84    APP_PUF_load_activation_code( activation_code ) – Application dependent code to read Activation
Code (ex: Non-volatile memory)
85
86    APP_PUF_test_activation_code( activation_code ) – Application dependent code to validate
activation code. For example, the activation code might be stored in flash and thus it might have
initial clear values (i.e. zero or all-ones).
87
88
89    /* Load variables required by ipb_process_image()
90       * Application-Specific method for storage (nvram, header, etc.)
91      *
92      * Flash Storage:
93      * IPB_header – stored in OTP by Provisioning Image
94      * bound_image – stored in Flash; ipb_process_image output to SRAM
95      * counter – stored in NOR Flash
96      *
97      * Flashless Storage:
98      * IPB_header – stored in OTP by Provisioning Image
99      * bound_image – stored in SRAM at boot; ipb_process_image no output(XIP)
100      * counter – stored in NOR Flash
101     */
102    ret_code = APP_IPB_load_params( &params );
103    if (IID_SUCCESS != ret_code)    break;
104
105    /* Initialize SRAM dedicated for PUF */
106    ret_code = bk_init( sram_puf, BK_SRAM_PUF_SIZE_BYTES );
107    if (IID_SUCCESS != ret_code)    break;
108
```

```
109    /* Load Activation code
110    * Application-Specific method for storage
111    */
112    ret_code = APP_PUF_load_activation_code( activation_code );
113    if (IID_SUCCESS != ret_code)    break;
114
115    /* Test if Activation Code is Valid
116    * Application-Specific Method for Test (i.e. not all 0x0 or 0xFF)
117    *    Start Only, Enroll is during Enrollment 118    */
119    ret_code = APP_PUF_test_activation_code( activation_code );
120    if (IID_SUCCESS != ret_code)    break;
121
122    /* System has valid Activation Code; Enroll would alter IPBindingHeader */
123    ret_code = bk_start( activation_code );
124    if (IID_SUCCESS != ret_code)    break;
125    ret_code = ipb_process_image(
126            params->ipb_header,
127            params->image_address,
128            params->image_target_address,
129            params->additional_keys_target_address,
130            params->version_counter,
131            &bool_counter_updated,
132            params->execution_address );
133    if (IID_SUCCESS != ret_code)    break;
```

6.3 安全更新细节

安全更新的过程与安全引导的过程类似。加密的固件映像以与设备使用的初始映像相同的方式创建。固件分发过程还必须为每个授权使用固件的设备创建和分发一个新的 IP 绑定头文件。一旦被保护的固件绑定包和绑定头文件被发送到设备，绑定包将以与初始图像相同的方式解密。当映像安全选项需要回滚保护时，目标设备必须在非易失性内存中维护一个版本号。当需要回滚保护时，将检查存储的版本号，并且只在新映像包含较大的版本号时加载它。在接收新映像时，设备必须更新其存储的参考计数器。

```
134    /* Store Monotonic counter
135    * Application-Specific method for storage (i.e. NVRAM)
136    */
137    if( counter_updated == true )
138    ret_code = APP_IPB_store_counter( params->version_counter );
```

6.4 附加保护信息检索

受保护的映像包可能包括其他受保护的机密。这些机密可能是设备唯一的配置信息或用于应用程序的密钥。该信息块位于映像数据的第一个位置，大小为头部信息提供的 additional_keys_size。

7 总结

该应用笔记描述了使用 NXP LPC54S0xx MCU 系列提供的 PUF 技术保护嵌入式应用程序的技术。

来自 PUF 的物理上不可克隆的密钥为设备的识别和设备之间信息的安全分发提供了强大的信任基础。IP 绑定使用设备唯一的公钥作为保护固件和配置信息的基础。基于 PUF 的信任根也可以应用于设备内的其他加密服务。PUF 派生的密钥也可以用于安全地为协议（如 TLS、IPsec 和 SSH）派生设备唯一密钥。

NXP 已从 Intrinsic-ID 获得 PUF 技术许可。有关 PUF 技术应用和 IP 绑定的补充信息，可以通过访问 Intrinsic-ID 网站 https://www.intrinsic-id.com/ 或发送邮件至 info@intrinsic-id.com.

8 修订记录

版本号	日期	说明
1.0	2018 年 11 月 6 日	初始版本

How To Reach Us

Home Page:

nxp.com

Web Support:

nxp.com/support

Limited warranty and liability — Information in this document is provided solely to enable system and software implementers to use NXP products. There are no express or implied copyright licenses granted hereunder to design or fabricate any integrated circuits based on the information in this document. NXP reserves the right to make changes without further notice to any products herein.

NXP makes no warranty, representation, or guarantee regarding the suitability of its products for any particular purpose, nor does NXP assume any liability arising out of the application or use of any product or circuit, and specifically disclaims any and all liability, including without limitation consequential or incidental damages. “Typical” parameters that may be provided in NXP data sheets and/or specifications can and do vary in different applications, and actual performance may vary over time. All operating parameters, including “typicals,” must be validated for each customer application by customer's technical experts. NXP does not convey any license under its patent rights nor the rights of others. NXP sells products pursuant to standard terms and conditions of sale, which can be found at the following address: nxp.com/SalesTermsandConditions.

Right to make changes - NXP Semiconductors reserves the right to make changes to information published in this document, including without limitation specifications and product descriptions, at any time and without notice. This document supersedes and replaces all information supplied prior to the publication hereof.

Security — Customer understands that all NXP products may be subject to unidentified or documented vulnerabilities. Customer is responsible for the design and operation of its applications and products throughout their lifecycles to reduce the effect of these vulnerabilities on customer’s applications and products. Customer’s responsibility also extends to other open and/or proprietary technologies supported by NXP products for use in customer’s applications. NXP accepts no liability for any vulnerability. Customer should regularly check security updates from NXP and follow up appropriately. Customer shall select products with security features that best meet rules, regulations, and standards of the intended application and make the ultimate design decisions regarding its products and is solely responsible for compliance with all legal, regulatory, and security related requirements concerning its products, regardless of any information or support that may be provided by NXP. NXP has a Product Security Incident Response Team (PSIRT) (reachable at PSIRT@nxp.com) that manages the investigation, reporting, and solution release to security vulnerabilities of NXP products.

For more information, please visit: http://www.nxp.com

For sales office addresses, please send an email to: salesaddresses@nxp.com

Date of release: 2018 年 11 月 6 日

Document identifier: AN12291

arm