# AN13086

## EdgeLock$^{TM}$ SE05x to enhance the MCU boot sequence security

Rev. 1.0 — 13 April 2021 Application note

**Document information**

| Information | Content |
|---|---|
| Keywords | EdgeLock SE05x, binding, secure, boot |
| Abstract | This application note describes how to enhance the security of the boot process by using EdgeLock SE05x secure element to verify firmware images. |

# Revision history

**Revision history**

| Revision number | Date | Description |
|---|---|---|
| 1.0 | 2021-04-13 | Initial version |

# 1 Introduction

Internet of Things (IoT) devices have reached a widespread use in many different applications, including industrial and automotive equipment requiring high reliability and robustness. Many of these IoT devices may have access to sensitive end-user data, critical sensor information and intellectual property in the form of software implementations and algorithms. Specifically in industrial applications, operation continuity and correct functioning is essential.

To ensure the correct operation of IoT devices and prevent threats such as theft, cloning and counterfeiting, one fundamental step is to ensure the integrity and authenticity of the firmware embedded into the IoT devices. In order to run only trusted and authorized software, IoT devices need to implement a mechanism to cryptographically verify the integrity and authenticity of the firmware image, so that invalid or malicious code is prevented from being executed.

A dedicated Secure Element (SE) such as EdgeLock SE05x can be used as a trust anchor to validate the authenticity and the integrity of the firmware and ensure that only signed software is executed in the IoT devices. The SE protects mission-critical cryptographic keys and provides cryptographic services to the devices. EdgeLock SE05x is Common Criteria EAL 6+ certified up to OS level and provides security against physical and logical attacks aimed, for example, at extracting security keys.

This application note describes how EdgeLock SE05x can be leveraged to enhance the security of the boot process. To find out more about other additional security features please refer to NXP Security Primitives.

# 2 Secure boot of the device firmware using EdgeLock SE05x

Secure boot is a security mechanism that has been designed to protect the boot chain of a system against the execution of malicious or counterfeited code. It ensures that only authentic and authorized software is executed on the device, from the IoT device bootloader up to the user applications.

One fundamental piece of code in the boot chain of a device is the device firmware. It contains routines fundamental for the operation of the IoT device, such as system initialization routines and the application logic. By compromising the firmware, an attacker can infiltrate malicious or counterfeited software, exfiltrate or manipulate sensitive data and cause the intentional malfunctioning of the device. Protecting firmware integrity and authenticity during the boot process is therefore essential to guarantee that data will be protected and that the device will work and behave as intended by the manufacturer.

Thanks to its anti-tamper features, policy enforcement capabilities and support of modern cryptographic algorithms with high key lengths, EdgeLock SE05x provides all the tools required to support the verification of the integrity and authenticity of the firmware in a connected MCU. In fact, EdgeLock SE05x can be used as a secure trust anchor for the firmware validation keys and as a secure cryptoprocessor for carrying out related cryptographic operations.

This section describes how to implement the secure boot of the device firmware using EdgeLock SE05x:

- Section 2.1 describes how to implement the secure boot of the firmware by verifying a cryptographic signature of the firmware with a public key securely stored in EdgeLock SE05x. This solution requires the OEM to securely generate, manage and store private credentials in an own managed PKI infrastructure;
- Section 2.2 describes how to strengthen the secure boot of the firmware by implementing the binding of the host MCU to EdgeLock SE05x. Binding is a process which establishes a pairing between the IoT device host MCU and EdgeLock SE05x, so that the MCU is only able to use the services offered by the paired EdgeLock SE05x and EdgeLock SE05x is only able to provide services to the paired MCU.

For more information about the EdgeLock SE05x product features, documentation and EdgeLock SE05x Plug & Trust middleware, refer to www.nxp.com/SE050.

## 2.1 Asymmetric firmware verification

The device firmware integrity and authenticity can be ensured during the boot process by verifying a digital signature of the firmware image prior to execution. In this configuration, the system's firmware image is hashed and signed during manufacturing with a private key from a private-public key pair (ECC or RSA). The private key never leaves the OEM facilities. The public key instead is shipped with the device. During boot, the MCU uses the public key to verify the firmware signature before loading the firmware image. If the private key is securely stored in the OEM facilities and remains secret, the successful verification of the signature is an undeniable proof that the firmware is indeed authentic and hasn't been tampered with.

However, if the public key remains vulnerable and unprotected in the IoT device, attackers could potentially replace it with a public key of a key pair that they have control of and in this way perform operations such as loading a modified or completely different firmware image. EdgeLock SE05x ensures both the secure storage of the public key that is used to verify the firmware image, as well as the secure and fast execution of the signature verification algorithm. Table 1 lists the expected execution times for some common public key verification algorithms. Finally, it is possible to effectively protect the public key against unauthorized manipulations and alterations by taking advantage of EdgeLock SEO5x anti-tampering protections against physical and side channel attacks; and also by taking advantage of secure object policies to make the public key unchangeable and non-deletable without authentication.

Table 1. SE processing time of public key verification in EdgeLock SE05x

| Algorithm | SE processing time |
|---|---|
| RSA 4096 | < 50 ms |
| ECDSA with NIST-P256 curve | < 55 ms |
| ECDSA with NIST-P521 curve | < 110 ms |
| EdDSA | < 50 ms |

The secure boot of the device firmware using asymmetric cryptography and EdgeLock SE05x only requires the components shown in Figure 1:

- **EdgeLock SE05x secure element**: it is connected to the IoT device MCU and stores the public key that will be used to verify the authenticity of the firmware image prior to execution. The public key must be injected during manufacturing and must be made unchangeable and non-deletable (if no authentication is performed) by setting the appropriate EdgeLock SE05x policies. The update rights of the public key can be configured by setting the appropriate policies so that authenticated users and/or services can securely update the public key if needed;
- **Device bootloader:** it is responsible for loading into memory the software components that have to be executed, including the device firmware image, when the device boots up. The device bootloader might consist of one or several stages (e.g. a primary bootloader, followed by a secondary bootloader). The device bootloader must be configured to establish a communication with EdgeLock SE05x during one of its stages and perform the firmware image validation before finally loading the firmware image. Note that, If an adversary is able to revert this configuration, the SE we would not gain any benefit. Refer to Section 2.2 to increase the boot sequence security.
- **Firmware image hash and signature**: The signature of the firmware hash must be computed in the OEM facilities using the private key associated to the public key stored in EdgeLock SE05x. The bootloader sends the hash computed on the current version of the device firmware and the pre-loaded signature attached to this firmware. Then, the EdgeLock SE05x verifies the signature of the received hash using the public key stored in the EdgeLock SE05x and returns the success or failure of this verification. To avoid potential replay attacks, the MCU-SE communication must be secured, refer to Section 2.2 for details on how to establish a secure channel with the SE (i.e. platform SCP).
  **Note**: *The hash of the current device firmware is computed in the MCU each time the device boots, since computing it on the SE brings no security benefit, and the speed is thereof conditioned by the implementation of the MCU bootloader. In addition*

The next sections will describe the operations that must be performed to enable the secure boot of the firmware using EdgeLock SE05x.

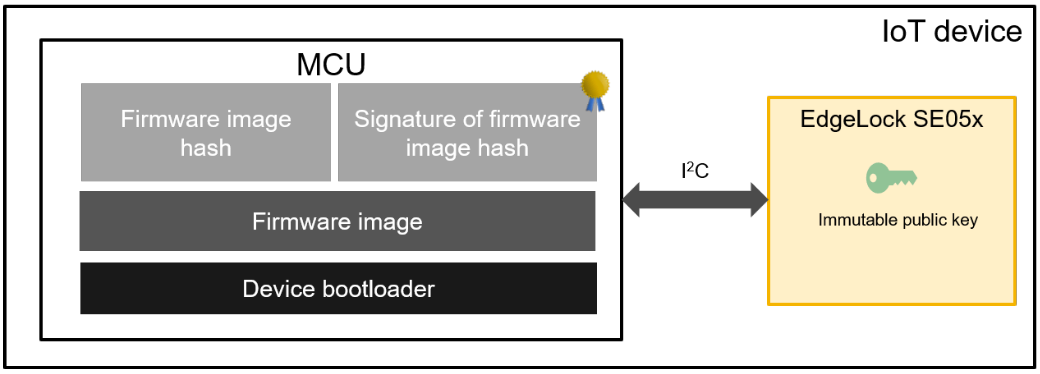

Figure 1. Asymmetric firmware verification using EdgeLock SE05x (components)

### 2.1.1 IoT device preparation

Implementing the secure boot of the device firmware using asymmetric cryptography requires the OEM to perform a set of preliminary steps during device manufacturing as shown in Figure 2.

1. The OEM must first generate a private-public key pair. The private key must be stored in a secure location and shall remain secret at all times. The private key will be used only to sign the firmware image hash. The public key, as the name implies, can be distributed and will be used by the device to verify the signature of the firmware image. The generated key pair must be supported by EdgeLock SE05x and can therefore be either an RSA key pair (512 up to 4096 bits) or an ECC key pair (NIST, Brainpool, Edwards and Montgomery curves are supported);
2. The OEM must compute the firmware image hash and sign it using the signature algorithm corresponding to the type of the key pair. The resulting signature must be stored in the device MCU;
3. The public key must be injected in EdgeLock SE05x. When creating the secure object in the secure element it is fundamental to add at least the policy for unauthenticated users shown in Table 2 so that the public key cannot be changed or deleted if no authentication is performed.

Table 2. Mandatory policy for unauthenticated users for public key object

| Policy | Description | Value |
|---|---|---|
| ALLOW_READ | If true object can be read | True |
| ALLOW_WRITE | If true object can be updated | False |
| ALLOW_VERIFY | If true object can be used for signature verification | True |
| ALLOW_DELETE | If true object can be deleted | False |

4. Finally, the OEM shall include in the boot process the validation of the firmware image using EdgeLock SE05x as described in Section 2.1.2.

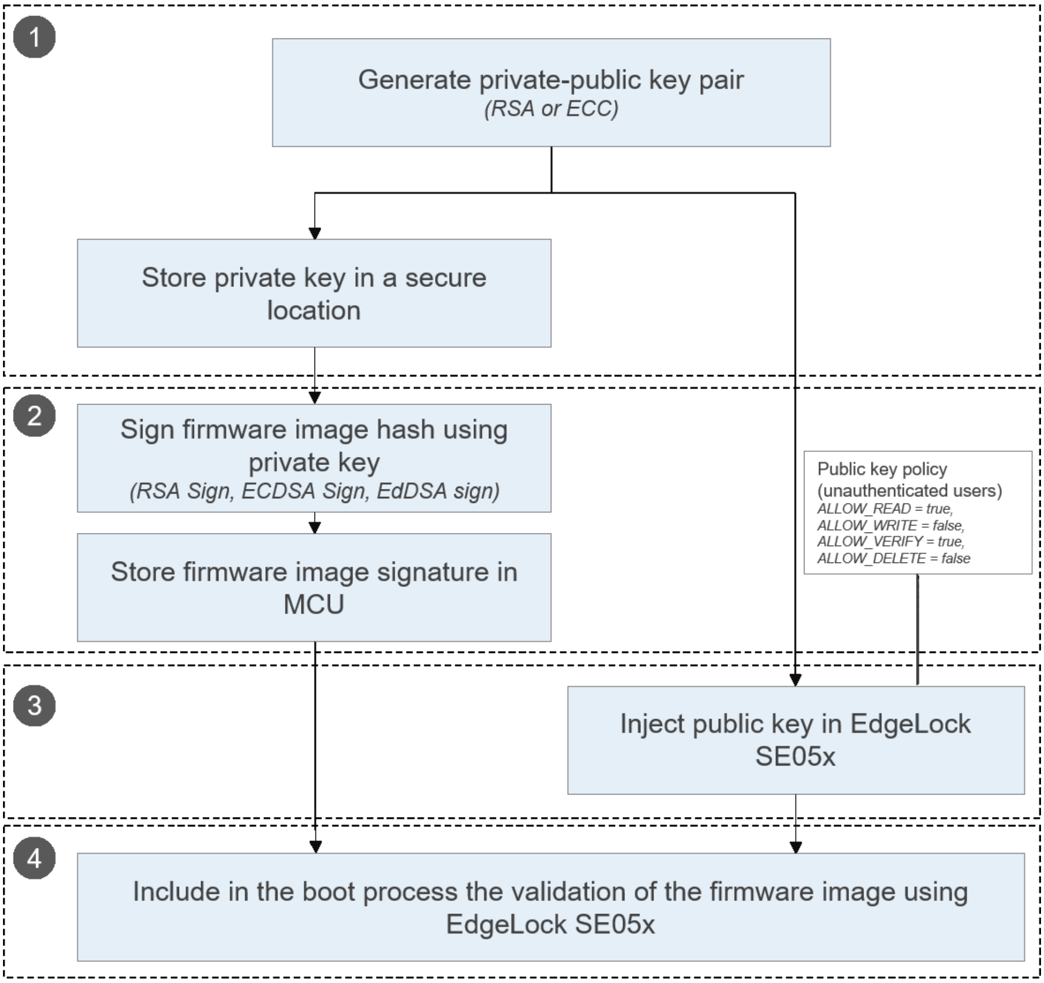
1
Generate private-public key pair
(RSA or ECC)
Store private key in a secure location
2
Sign firmware image hash using private key
(RSA Sign, ECDSA Sign, EdDSA sign)
Store firmware image signature in MCU
Public key policy
(unauthenticated users)
ALLOW_READ = true,
ALLOW_WRITE = false,
ALLOW_VERIFY = true,
ALLOW_DELETE = false
3
Inject public key in EdgeLock SE05x
4
Include in the boot process the validation of the firmware image using EdgeLock SE05x

**Figure 2. IoT device preparation steps for asymmetric firmware verification**

### 2.1.2 Firmware verification during boot

Once the IoT device has been prepared in the OEM's manufacturing facilities as described in Section 2.1.1, the device bootloader logic shall be adapted to include the validation of the firmware image before executing it as shown in Figure 3:

1. When the IoT device is powered up, the device bootloader is executed first. The device bootloader might consist of one or several stages. During one of these stages – but before loading the firmware image – the bootloader requests to the EdgeLock SE05x to perform the signature verification of the firmware image.
   ***Note:*** *The host device must ensure that the bootloader is not alterable ; otherwise a skilled attacker might be able to bypass the whole secure boot process.*
2. The bootloader fetches from memory the pre-injected firmware image signature and computes the hash of the firmware image. These two pieces of information are then sent to EdgeLock SE05x for verification. Note that, you can rely on EdgeLock SE05x Plug & Trust middleware to facilitate integration with EdgeLock SE05x.
3. EdgeLock SE05x uses the pre-injected public key to verify the signature. With EdgeLock SE05x Plug & Trust middleware this operation can be easily performed by first setting the algorithm, operation and key to be used for verification (*sss_asymmetric_context_init ()*) and then by calling the

*sss_asymmetric_verify_digest ()* API with the hash and signature of the firmware as input parameters;

4. The result of the signature verification as reported by EdgeLock SE05x is evaluated by the bootloader. If the verification succeeds, then the firmware image is loaded into memory and executed; otherwise execution is blocked. Note that, secure coding best practices must be applied to prevent fault injection attacks (e.g. glitch attacks) changing the decision.

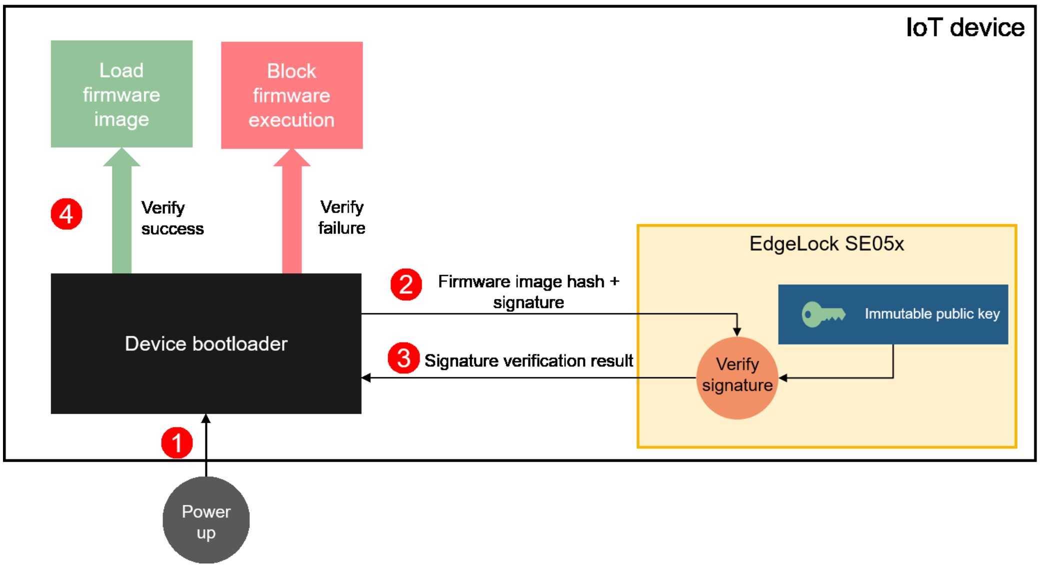

Figure 3. Secure firmware boot using asymmetric cryptography

## 2.2 Secure element binding to increase boot sequence security

For improved security, EdgeLock SE05x provides manufacturers the option to bind the MCU of the IoT device to the secure element as shown in Figure 4.

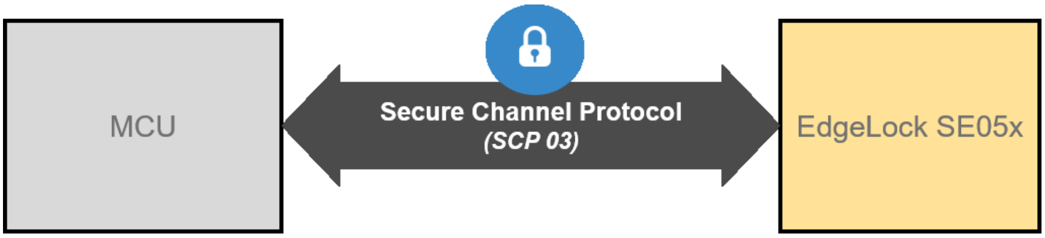

Figure 4. Binding of a host MCU to EdgeLock SE05x

Binding is a process to establish a pairing between the IoT device host MCU and EdgeLock SE05x, so that the MCU is only able to use the services offered by the paired EdgeLock SE05x and EdgeLock SE05x is only able to provide services to the paired MCU. A mutually authenticated, encrypted channel will ensure that both parties are indeed communicating with the intended recipients and that local communication is protected against local attacks, including man-in-the-middle attacks aimed at intercepting the communication between the MCU and the SE and physical tampering attacks aimed at replacing the MCU or the SE. EdgeLock SE05x natively supports Global Platform Secure Channel Protocol 03 (SCP03) for this purpose.

EdgeLock SE05x can be bound to the host MCU by injecting in both the MCU and EdgeLock SE05x the same unique SCP AES key-set and by enabling EdgeLock SE05x Platform SCP feature. EdgeLock SE05x Platform SCP feature allows the MCU to protect the communication between a host processor and the secure element using an SCP03 channel established at the platform level. If Platform SCP is made mandatory in EdgeLock SE05x, the MCU would first be required to establish a mutually authenticated SCP session with the secure element before any other operation can be performed (including the operations related to the validation of the firmware image). All subsequent commands must be sent encrypted and MACed using the SCP session keys.

For additional information to perform the secure element binding, refer to AN12662 and AN12514.

# 3 Overview of EdgeLock SE05x Plug & Trust middleware secure boot demo

The EdgeLock SE05x Plug & Trust middleware contains a demo that showcases an example implementation of secure boot using LPC55S69 MCU and EdgeLock SE05x. The demo leverages EdgeLock SE05x as a trust anchor to verify a firmware image (secure application) before finally loading it. A 2048-bit RSA key pair is used to sign and verify the firmware image. The example also performs the secure binding of the MCU with the SE, so that the MCU is only able to use the services offered by the paired EdgeLock SE05x, and EdgeLock SE05x is only able to provide services to the paired MCU.

The secure boot demo supports LPC55S69 MCUs and uses the Physically Unclonable Function (PUF) hardware of LPC55S69 MCU to securely store Platform SCP keys. Platform SCP keys are used to bind the MCU to the SE. For more information on how PUF can be used to implement binding between the MCU and EdgeLock SE05x, please refer to AN12662.

For detailed instructions on how to prepare and run the secure boot demo, please refer to the EdgeLock SE05x Plug & Trust middleware documentation (*/simw-top/doc/demos/lpc55s/ex/puf_se05x_sbl/Readme.html*). A high-level overview of the main steps required to run the demo is given below.

Before running the demo, the following actions must be completed:

1. Inject in the SE the public key (`K_PUB_OEM`, 2048-bit RSA key) that will be used to verify firmware images. The key ID (`K_PUB_OEM_ID`) of the public key must be updated in `puf_se05x_sbl_s.c`;
2. Enroll the PUF in LPC55S69 using the *Key injection to PUF* example. Please refer to EdgeLock SE05x Plug & Trust middleware documentation for more information on how to run the *Key injection to PUF* example (`/simw-top/doc/demos/lpc55s/ex/puf_inject_scp03/Readme.html`);
3. Provide new Platform SCP keys in `puf_se05x_sbl_s.c`. These keys will be used when SCP key rotation is performed during the first boot. After key rotation is performed, usage of EdgeLock SE05x services will always require authentication using the new Platform SCP keys;
4. The following projects must be compiled: `puf_se05x_sbl_s`, `sbl_app_s`, `sbl_app_ns`. Before flashing the images, they must be signed using the private key associated to `K_PUB_OEM`. The *elftosb* tool can be used for signing the images.

An overview of the secure boot demo flow is shown in Figure 5. The demo implements a three step boot process:

1. First, the MCU bootloader verifies and loads a Secondary Bootloader (SBL) image.
2. **First boot:** the SBL performs all the steps required to bind the MCU to the SE. As a result, default Platform SCP keys are rotated, new keys are stored in PUF hardware and PUF Key Codes are stored in flash for later usage. At the end of the process, a flag is set to disable key rotation in subsequent boots.
   **Next boots:** the SBL opens an SCP03 session with the SE using the Platform SCP keys stored in PUF and uses the SE to verify the secure application image using the public key pre-provisioned in the SE (`K_PUB_OEM`). If the verification is successful, the secure application image is loaded and PUF Key Codes are handed over to the secure application so that they can later be used to establish a secure SCP03 session with the SE.

3. Finally, the verified secure application opens an SCP03 session with the SE using the Platform SCP keys stored in PUF.

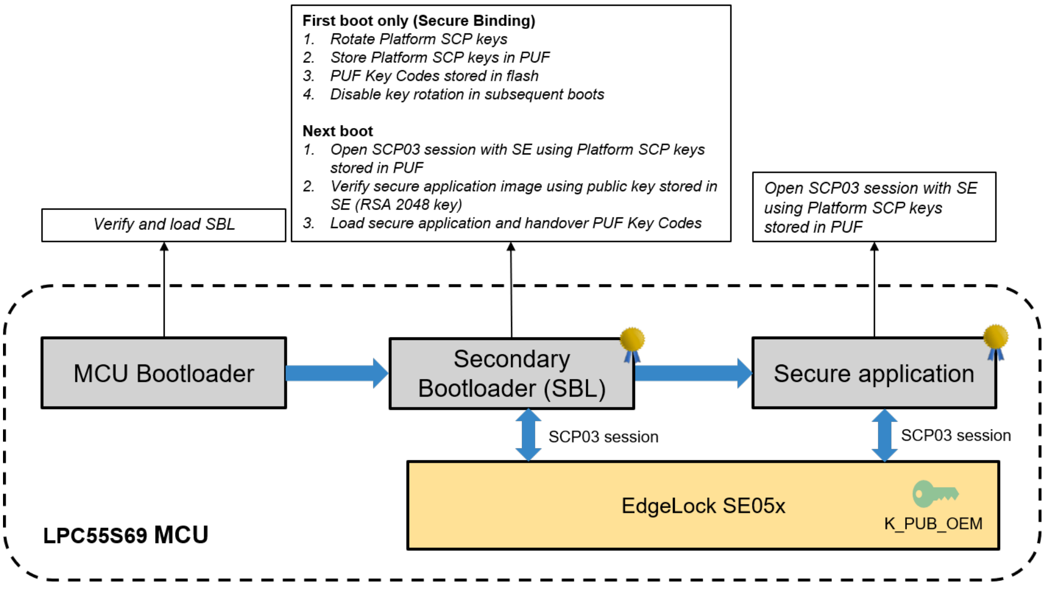
First boot only (Secure Binding)
1. Rotate Platform SCP keys
2. Store Platform SCP keys in PUF
3. PUF Key Codes stored in flash
4. Disable key rotation in subsequent boots
Next boot
1. Open SCP03 session with SE using Platform SCP keys stored in PUF
2. Verify secure application image using public key stored in SE (RSA 2048 key)
3. Load secure application and handover PUF Key Codes
Verify and load SBL
Open SCP03 session with SE using Platform SCP keys stored in PUF
MCU Bootloader
Secondary Bootloader (SBL)
Secure application
SCP03 session
SCP03 session
EdgeLock SE05x
K_PUB_OEM
LPC55S69 MCU

Figure 5. Overview of secure boot demo flow

# 4 Legal information

## 4.1 Definitions

**Draft** — A draft status on a document indicates that the content is still under internal review and subject to formal approval, which may result in modifications or additions. NXP Semiconductors does not give any representations or warranties as to the accuracy or completeness of information included in a draft version of a document and shall have no liability for the consequences of use of such information.

## 4.2 Disclaimers

**Limited warranty and liability** — Information in this document is believed to be accurate and reliable. However, NXP Semiconductors does not give any representations or warranties, expressed or implied, as to the accuracy or completeness of such information and shall have no liability for the consequences of use of such information. NXP Semiconductors takes no responsibility for the content in this document if provided by an information source outside of NXP Semiconductors. In no event shall NXP Semiconductors be liable for any indirect, incidental, punitive, special or consequential damages (including - without limitation - lost profits, lost savings, business interruption, costs related to the removal or replacement of any products or rework charges) whether or not such damages are based on tort (including negligence), warranty, breach of contract or any other legal theory. Notwithstanding any damages that customer might incur for any reason whatsoever, NXP Semiconductors' aggregate and cumulative liability towards customer for the products described herein shall be limited in accordance with the Terms and conditions of commercial sale of NXP Semiconductors.

**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.

**Suitability for use** — NXP Semiconductors products are not designed, authorized or warranted to be suitable for use in life support, life-critical or safety-critical systems or equipment, nor in applications where failure or malfunction of an NXP Semiconductors product can reasonably be expected to result in personal injury, death or severe property or environmental damage. NXP Semiconductors and its suppliers accept no liability for inclusion and/or use of NXP Semiconductors products in such equipment or applications and therefore such inclusion and/or use is at the customer's own risk.

**Applications** — Applications that are described herein for any of these products are for illustrative purposes only. NXP Semiconductors makes no representation or warranty that such applications will be suitable for the specified use without further testing or modification. Customers are responsible for the design and operation of their applications and products using NXP Semiconductors products, and NXP Semiconductors accepts no liability for any assistance with applications or customer product design. It is customer's sole responsibility to determine whether the NXP Semiconductors product is suitable and fit for the customer's applications and products planned, as well as for the planned application and use of customer's third party customer(s). Customers should provide appropriate design and operating safeguards to minimize the risks associated with their applications and products. NXP Semiconductors does not accept any liability related to any default, damage, costs or problem which is based on any weakness or default in the customer's applications or products, or the application or use by customer's third party customer(s). Customer is responsible for doing all necessary testing for the customer's applications and products using NXP Semiconductors products in order to avoid a default of the applications and the products or of the application or use by customer's third party customer(s). NXP does not accept any liability in this respect.

**Export control** — This document as well as the item(s) described herein may be subject to export control regulations. Export might require a prior authorization from competent authorities.

**Evaluation products** — This product is provided on an "as is" and "with all faults" basis for evaluation purposes only. NXP Semiconductors, its affiliates and their suppliers expressly disclaim all warranties, whether express, implied or statutory, including but not limited to the implied warranties of non-infringement, merchantability and fitness for a particular purpose. The entire risk as to the quality, or arising out of the use or performance, of this product remains with customer. In no event shall NXP Semiconductors, its affiliates or their suppliers be liable to customer for any special, indirect, consequential, punitive or incidental damages (including without limitation damages for loss of business, business interruption, loss of use, loss of data or information, and the like) arising out the use of or inability to use the product, whether or not based on tort (including negligence), strict liability, breach of contract, breach of warranty or any other theory, even if advised of the possibility of such damages. Notwithstanding any damages that customer might incur for any reason whatsoever (including without limitation, all damages referenced above and all direct or general damages), the entire liability of NXP Semiconductors, its affiliates and their suppliers and customer's exclusive remedy for all of the foregoing shall be limited to actual damages incurred by customer based on reasonable reliance up to the greater of the amount actually paid by customer for the product or five dollars (US$5.00). The foregoing limitations, exclusions and disclaimers shall apply to the maximum extent permitted by applicable law, even if any remedy fails of its essential purpose.

**Translations** — A non-English (translated) version of a document is for reference only. The English version shall prevail in case of any discrepancy between the translated and English versions.

**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.

## 4.3 Trademarks

Notice: All referenced brands, product names, service names and trademarks are the property of their respective owners.

**NXP** — wordmark and logo are trademarks of NXP B.V.

**EdgeLock** — is a trademark of NXP B.V.

# Tables

## Figures

## Contents

Please be aware that important notices concerning this document and the product(s) described herein, have been included in section 'Legal information'.

For more information, please visit: http://www.nxp.com
For sales office addresses, please send an email to: salesaddresses@nxp.com

**Date of release: 13 April 2021**
**Document identifier: AN13086**