UM3520

User manual

STM32C0 security guidance for SESIP level 3 certification

Introduction

This document describes how to prepare an STM32C0 microcontroller to make a secure system solution compliant with SESIP level 3.

The security guidance that is described in this document applies to any boards based on the devices listed in the table below for die revision.

Table 1. **Applicable products**

Type	Product series
Microcontroller	STM32C0 series

1 General information

This document applies to STM32C0 Arm®-based MCU.
Arm is a registered trademark of Arm Limited (or its subsidiaries) in the US and/or elsewhere.

Table 2. **Specific acronyms**

Acronym	Description
HDP	Secure HiDe protection aka securable memory area
HW	Hardware
IoT	Internet of Things
MCU	Microcontroller
TOE	Target of evaluation
RDP	Readout protection
RMA	Return material for analysis
SCA	Side-channel attack
SESIP	Security evaluation standard for IoT platforms
SFR	Security-functional requirement

1.1 Platform reference

Table 3. **Platform reference**

<table>
<tr><th>Reference</th><th colspan="4">Value</th></tr>
<tr><td>Platform name</td><td colspan="4">STM32C0 Arm® Cortex®-M0+ 32-bit MCU</td></tr>
<tr><td>Platform version</td><td colspan="4">Revision 1</td></tr>
<tr><td>Platform identification</td><td>Commercial name:
• STM32C011
• STM32C031
• STM32C051
• STM32C071
• STM32C09x</td><td>Die identifier:
• 443
• 453
• 44C
• 493
• 44D</td><td>Rev identifier:
• 0x1001
• 0x1001
• 0x1000
• 0x1001
• 0x1000</td><td>On chip flash:
• 32 KB
• 32 KB
• 64 KB
• 128 KB
• 256 KB</td></tr>
<tr><td>Platform type</td><td colspan="4">General purpose microcontroller device for IoT, industrial, or consumer applications.</td></tr>
</table>

1.2 Included guidance documents

The following documents are included with the platform:

Table 4. **Guidance documents**

Category	Name	Reference
User manual	UM3520 - STM32C0 security guidance for SESIP level 3 certification	SG
Product reference manual	RM0490 - Reference manual STM32C0 series advanced Arm®-based 32-bit MCUs	RM

2 Reference documents

Table 5. Reference documents

Reference	Document title and revision
[RM]	Reference manual for STM32C0 series (RM0490), revision 5
[ST]	TN1590: STM32C0 SESIP Security Target for PSA Certified™ RoT Component Level 3, revision 1.0
[UM2237]	User manual STM32CubeProgrammer software description (UM2237), revision 22
[UM2609]	STM32CubeIDE user guide (UM2609), revision 9
[IEEE1149]	EEE 1149.1 – 2013
[IHI0031]	Arm Debug Interface Architecture Specification ADIv5.0 to ADIv5.2
[SM]	PSA Certified Platform Security Model 1.1 JSADEN014

3 TOE preparative procedures

This chapter describes the procedures to prepare the environment and the TOE before starting to use the device or before testing the IoT product:

- Secure acceptance: procedures to check the device to be tested
- Secure preparation of the operational environment: procedures to set up the environment needed to manage and test the IoT product.
- Secure installation: procedure to program and configure the IoT product to be tested.

3.1 Secure acceptance

Secure acceptance is the process in which the user securely receives the TOE and verifies its genuineness.

The TOE is distributed as an STM32xx MCU device, with corresponding firmware packages that can be obtained from www.st.com. Refer to the cover page for the applicable devices.

How to accept an STM32C0xxxx MCU device

`STM32C011: By reading, with STM32CubeProgrammer (for more details, refer to [UM2237]) the DBG_IDCODE register value as defined in RM:`

- DBG identity code register (DBG_IDCODE)
  - Base address: `0x4001 5800`
  - Address offset: `0x00`
  - Reset value: `0x1001 6443`

`STM32C031: By reading, with STM32CubeProgrammer (for more details, refer to [UM2237]) the DBGMCU_IDCODE register value as defined in RM:`

- DBG identity code register (DBG_IDCODE)
  - Base address: `0x4001 5800`
  - Address offset: `0x00`
  - Reset value: `0x1001 6453`

`STM32C051: By reading, with STM32CubeProgrammer (for more details, refer to [UM2237]) the DBGMCU_IDCODE register value as defined in RM:`

- DBG identity code register (DBG_IDCODE)
  - Base address: `0x4001 5800`
  - Address offset: `0x00`
  - Reset value: `0x1000 644C`

`STM32C071: By reading, with STM32CubeProgrammer (for more details, refer to [UM2237]) the DBG_IDCODE register value as defined in RM:`

- DBG identity code register (DBG_IDCODE)
  - Base address: `0x4001 5800`
  - Address offset: `0x00`
  - Reset value: `0x1001 6493`

`STM32C091 and STM32C092: By reading, with STM32CubeProgrammer (for more details, refer to [UM2237]) the DBG_IDCODE register value as defined in RM:`

- DBG identity code register (DBG_IDCODE)
  - Base address: `0x4001 5800`
  - Address offset: `0x00`
  - Reset value: `0x1000 644D`

In the following sections, STM32C0 designates STM32C011, STM32C031, and STM32C0X1 products indifferently.

3.2 Secure installation and preparation of the operational environment (AGD_PRE.1.2C)

This section describes all the steps necessary for secure installation of the TOE and for the secure preparation of the operational environment in accordance with the security objectives for the operational environment as described in [ST].

3.2.1 Hardware and software setup procedures

To set up the hardware environment, the development board must be connected to a personal computer via a USB cable. This connection with the PC allows the user to:

- Configuring platform non-volatile memory (flash)
- Debugging when the protections are disabled.

The ST-LINK firmware programmed on the development board must be the V2J42M27 version or newer.

Software tools for programming STM32 microcontrollers

The STM32CubeProgrammer (STM32CubeProg) is an all-in-one multi-OS software tool for programming STM32 microcontrollers. It provides an easy-to-use and efficient environment for reading, writing, and verifying device memory through both the debug interface (JTAG and SWD) and the bootloader interface (UART and USB).

The STM32CubeProgrammer offers a wide range of features to program STM32 microcontroller internal memories (such as flash memory, RAM, and option bytes) as well as external memories. STM32CubeProgrammer also allows option programming and upload, programming content verification, and microcontroller programming automation through scripting.

The STM32CubeProgrammer is delivered in GUI (graphical user interface) and CLI (command-line interface) versions.

For more details about STM32CubeProgrammer, refer to [UM2237].

The latest release of the STM32CubeProgrammer software package is recommended.

STM32CubeC0 firmware package

The STM32CubeC0 package includes necessary drivers, middleware, template projects, example applications and compiler toolchain for software development on STM32C0 product. Download and install the latest package release from any of these:

- https://github.com/STMicroelectronics/STM32CubeC0
- or https://www.st.com/en/embedded-software/stm32cubec0.html

Terminal emulator

A terminal emulator software is needed to control the correctness of the installation if the example described in Appendix A: Building is used. The example in this document is based on Tera Term, an open-source free software terminal emulator that can be downloaded from the https://osdn.net/projects/ttssh2/ webpage. Any other similar tool can be used instead.

3.2.2 Secure preparation

The STM32C0 product preparation is done in three steps to get a complete installation with fully activated security configuration. The three steps must be entirely done as security protections are only configured at the very last step:

Step 1: Software build

- The non-platform required secure boot and root parameters are linked at address 0x0800 0000 which is the hardcoded start address of the securable memory area.
- All the other parts of the application code not belonging to the secure boot must be linked out of the securable memory area.
- A detailed building procedure is provided in Appendix A: Building, based on the secure boot code example available in the STM32CubeC0 firmware package.

Step 2: Software programming:

- Copy the images generated in the previous step into the STM32C0 internal flash memory.
- The non-platform required secure boot is loaded in the securable memory at address 0x0800 0000.
- All the other parts of the application code not belonging to the secure boot are loaded out of the securable memory area at the destination address according to the build parameters.

Step 3: STM32C0 static security protection programming

- Program SEC_SIZE[5:0] option byte with the index of the last page including the secure boot and root parameters programmed in step 2.
- Program BOOT_LOCK=1
- Program a write protected memory area (WRP) inside the securable memory area to contain the Immutable Platform Root of Trust (Cf. [SM]: 1.3.2 Immutable Platform Root-of-Trust Protection).
  - WRP1A_STRT: index of the first page of the write protected memory area.
  - WRP1A_END: index of the last page of the write protected memory area.
- Program read protection in level 2: RDP=0xCC

A detailed programming procedure is provided in Appendix B: Programming, based on the secure boot code example available in the STM32CubeC0 firmware package.

After achieving those secure installation steps, the platform is in its secured configuration.

4 Operational user guidance

4.1 User roles

The user role integrator is the only role involved in the preparative TOE procedure. The integrator is responsible to:

- Receive the TOE
- Develop (or subcontract) the embedded firmware required by the product
- Perform the preparative procedures as described in TOE preparative procedures
- Integrate the TOE into a finalized product.

The integrator has full access to the TOE security features, as the STM32C0 MCU devices are delivered in RDP0 state without any features activated for NVM protection. The integrator also has full access to the tools needed to program the TOE.

Once the TOE is in certified configuration, the user-operational guidance is described in Operational guidance for the integrator roles.

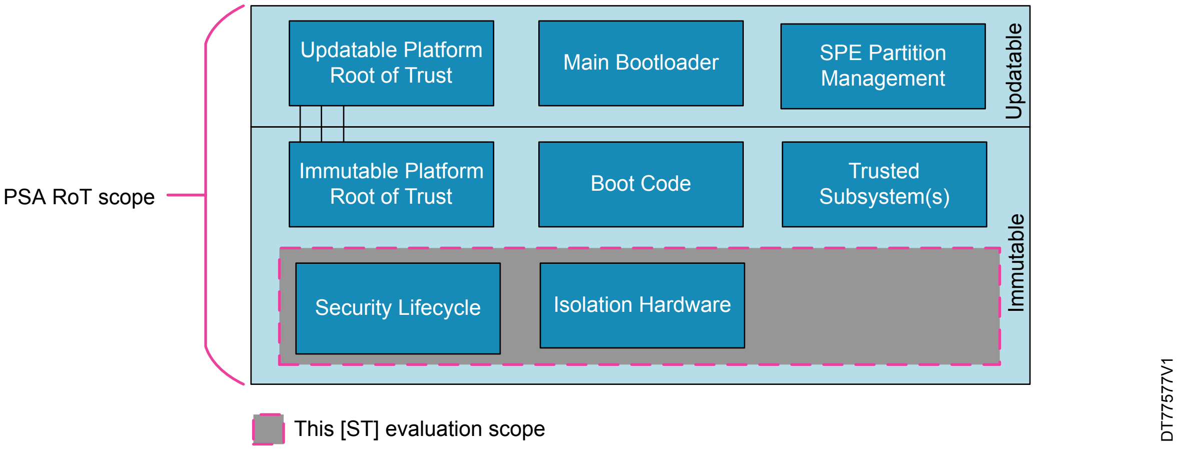
> The integrator has full access to the TOE security features, as the STM32C0xx MCU devices are delivered in RDP0 state without any protection feature activated. The integrator also has full access to the tools needed to program the TOE.

4.2 Operational guidance for the integrator roles

4.2.1 User-accessible functions and privileges (AGD_OPE.1.1C)

The main task of the integrator is to integrate the TOE into a final product. To this end, the system integrator has access to interfaces that are unavailable to other users, as described in Available interfaces and methods of use (AGD_OPE.1.2C and AGD_OPE.1.3C). The integrator cannot change any parts inside the TOE but must configure the TOE to make it functional in the final secure state. The integrator can change parts outside the TOE without compromising the security of the TOE as shown in Figure 1. TOE perimeter.

Figure 1. TOE perimeter



Follow the procedures described in Section 3.1: Secure acceptance to check if the TOE is acceptable for the secure configuration. The secure configuration of the TOE may be impacted when changing some parts of the TOE but may also be impacted when changing some parts located outside the TOE scope. This section describes changes that the integrator can make and highlights what is covered in the evaluation scope and what may impact the secure configuration of the TOE.

Product firmware

The integrator must first load the product firmware prior to setting up the security configuration of the final product.

The integrator must split the product firmware into two separated areas:

- Securable memory area in which the integrator can locate its protected assets. This area could contain, for example, a first stage secure boot code with the root parameters composed of private and immutable personalization data or cryptographic elements depending on the integrator product security requirements.
- Non-securable memory area (the rest of the flash memory space) where the integrator can locate the next boot stages plus the application firmware.

The integrator shall define the securable memory area size within a number N of 2 KB pages and shall link the product firmware with the securable memory area starting at address 0x0800 0000 and the non-securable memory area starting at address 0x0800 0000 + N*0x800.

RDP Level

The secure platform shall be deployed in RDP level 2.

The RDP value indicates the protection level of the product as explained in [RM] §4.5.1:

- RDP level 0 if RDP=0xAA
- RDP level 2 if RDP=0xCC
- RDP level 1 if RDP is not 0xAA neither 0xCC

A detailed programming procedure is provided in Appendix B: Programming.

(HDP) Securable memory area

The platform shall be deployed with the securable memory enabled. The integrator shall link the Immutable Root of Trust of the product firmware inside the securable memory area and configure the associated options bytes as follows:

- SEC_SIZE[6:0]: if different from 0x0, it represents the number of 2 KB pages required for the size of the securable memory area.

Additionally, the product firmware shall activate the read protection just before jumping out of the securable memory area by setting the SEC_PROT BIT to 0x1. Cf [RM] §4.7.5.

Boot configuration

The integrator shall enforce the boot mode configuration where the product firmware can only start from the main user flash memory by asserting the option byte BOOT_LOCK to 1.

4.2.2 Available interfaces and methods of use (AGD_OPE.1.2C and AGD_OPE.1.3C)

To exercise the functions and privileges described in User-accessible functions and privileges (AGD_OPE.1.1C), the integrator interacts with the TOE interfaces described in this section:

- Physical chip interface
- Flash registers and option bytes
- SWD debug interface

In the context of the certification, the TOE implements several mechanisms to secure the initialization of the platform and increase robustness against physical attacks.

Physical chip interface

After providing power supply and clocks and de-asserting the reset sign (Refer to [RM] §4, §5 to get details about the power-on and reset procedure), the platform firmware starts at a unique boot hardcoded address.

Method of use:

- Activate power supplies and clocks of the platform.
- Reset the device.

Parameters:

- Not applicable

Actions:

- The processor first executes the product code located in the HDP1 memory area.

Errors:

- The platform firmware does not start properly if its vector table of the integrator firmware is not located at the boot address in the user flash.

Flash option bytes

The integrator in the secure boot shall read the flash option bytes to confirm that the platform is started in its secure state:

- Securable memory area definition
- RDP level 2 life cycle state
- Write protection on securable area pages (Cf. [RM] §4.5.3)
- Forced in a unique boot entry address by BOOT_LOCK

Method of use:

- The processor performs read access in the flash control registers.

Parameters:

- FLASH_SECR.SEC_SIZE[6:0] = number of/last index of the 2 KB page of securable memory/HDP area
- FLASH_SECR.BOOT_LOCK = 1
- FLASH_OPTR.RDP[6:0] = 0xCC for RDP2 life-cycle state
- FLASH_SECR.WRP1A_STRT = 0
- FLASH_SECR.WRP1A_END = last index of the 2 KB page including the Immutable Root of Trust

Actions:

- Verify the value of each enumerated configuration byte.

Errors:

- The actions fail if the content read from option bytes differs from the programmed values. In that case, the integrator firmware must cancel the boot operation by resetting the platform or looping infinitely (for example).

Flash registers

When the product firmware ends up with the code located in the securable memory area, it jumps to the code located in the non-securable area. The jump procedure shall activate the securable area to be inaccessible by any read or write operation until the next reset.

Method of use:

- Configure SEC_PROT = 0x0 in the FLASH_CR register (Cf. [RM] §4.7.5)

Parameters:

- FLASH_CR.SEC_PROT

Actions:

- Write SEC_PROT with 0x1.
- Verify securable memory activation by reading a 32-bit word which is programmed to a non-zero value, anywhere in the securable memory area.
- This should generate a bus error and the read is 0x0.

Errors:

- The verification value is not zero, meaning the securable memory area is not properly activated. In that case, the product firmware shall be restarted.

SWD debug interface

Standard JTAG with SWD interface allows debugging of the TOE and integrator application. It is used according to IEE1149 and IHI0031.

When RDP is level 0, the debug interface is active/open and the security settings described in the TSFI “FLASH. Option bytes” are configured.

When RDP is Level 2, all debug features are disabled. Any SWD connection is ignored.

Method of use:

- RDP level 0: through JTAG/SWD I/Os
- PA14: SWCLK in pull-down
- PA13: SWDIO in pull-up
- RDP level 2: None

Parameters:

- Refer to section 30.5.2 SWD protocol sequence in the [RM].

Actions:

- Performing read/write transactions to flash interface using core access port and debug port.

Errors:

- RDP0: Connection not established, option byte programming failure
- RDP2: Not applicable

Note: A detailed programming procedure is provided in Appendix B: Programming.

4.2.3 Security-relevant events (AGD_OPE.1.4C)

Once configured according to Secure installation and preparation of the operational environment (AGD_PRE.1.2C), the platform detects any unauthorized access and any unexpected configuration:

- Erroneous values found in the option bytes of the security configuration at boot time:
  - Cancel the boot sequence by resetting or blocking the platform (integrator defined implementation)
- Illegal access in the (HDP) securable memory area:
  - Read/Write causes a bus error. The integrator firmware can implement an error handler.
- Attempt to reprogram the security option bytes in RDP level 2:
  - Status FLASH_SR.WRPERR raised with a possible interrupt toward a user implementation defined error handler
- Closed SWD access violation:
  - The connection request is not transmitted to the access port and debug port. The request is ignored.

4.2.4 Security measures (AGD_OPE.1.6C)

This section describes for the integrator user role the security measures to be followed to fulfill the security objectives for the operational environment as described in the [ST] (§2.1).

To achieve TRUSTED_INTEGRATOR and LIFECYCLE, the following measures must be taken:

- Follow all guidelines described and referenced in Section 3.2: Secure installation and preparation of the operational environment (AGD_PRE.1.2C).
- Follow all guidelines described in Section 4.2.1: User-accessible functions and privileges (AGD_OPE.1.1C) and Section 4.2.2: Available interfaces and methods of use (AGD_OPE.1.2C and AGD_OPE.1.3C) regarding the implementation of the non-platform required firmware described in [ST] (§1.4.5).
- In the manufacturing phase, the integrator must securely provision the TOE immutable data specific to the integrator or specific to the product as stated in the product firmware HDP securable memory area of Section 4.2.1: User-accessible functions and privileges (AGD_OPE.1.1C).
- Once the integrator finishes the production of a final user application, he must set the STM32C0 hardware static protections as stated in Section 3.2: Secure installation and preparation of the operational environment (AGD_PRE.1.2C). To protect the complete product including the user application, the final product state must be RDP2

The trusted integrator personalizes the TOE and uses the TOE security functionalities so that the IoT product meets its final certification target. The integrator is trusted and does not attempt to thwart the TOE security functionalities, nor attempt to bypass them.

4.2.5 Modes of operation (AGD_OPE.1.5C)

This section identifies all possible modes of operation of the platform (including operation following failure or operational error), their consequences, and implications for maintaining secure operation.

Normal boot mode

After reset, the platform is forced to boot into the non-platform required secure firmware. The other alternate boot modes have all been unauthorized by the security configuration dictated by Section 3.2: Secure installation and preparation of the operational environment (AGD_PRE.1.2C).

Appendix A Building

The following steps describe the operation to build a secure boot example code and it demonstration application based on STM32CubeC0 MCU Full Package for the STM32C0 series under Windows®:

- Install the package in a folder
- Open the folder ".\Projects\NUCLEO_C071RB\Applications\ROT\OEMiSB_Boot\STM32CubeIDE"
- Double-click on ".project" file to automatically open STM32CubeIDE with import and setup of the secure boot example
- Open the folder ".\Projects\NUCLEO_C071RB\Applications\ROT\OEMiSB_Appli\"
- Double-click on ".project" file to automatically import and set up the demonstration application in STM32CubeIDE
- Open the file ".\Projects\NUCLEO_C071RB\Applications\ROT\OEMiSB_Boot\Inc\boot_cfg.h"
  - If the building is for the production version, ensure that the expected RDP level is 2
    - Line 28: #define OEMISB_OB_RDP_LEVEL_VALUE OB_RDP_LEVEL_2
  - If the building is for a development version, keep OB_RDP_LEVEL_0
  - Optionally align the expected secure boot area size that must fit HDP securable memory area of the hardware configuration.
    - Line 25: #define FLASH_BOOT_AREA_SIZE (0x2000UL)
- In folder ".\Projects\NUCLEO_C071RB\Applications\ROT\OEMiSB_Boot\STM32CubeIDE\
  - Open the link file "STM32C071RBTX_FLASH"
  - Ensure that the boot code is generated at flash origin address 0x8000000 (line 50)
- In folder ".\Projects\NUCLEO_C071RB\Applications\ROT\OEMiSB_Appli\STM32CubeIDE\
  - Open the link file "STM32C071RBTX_FLASH"
  - Ensure that the boot code is generated at flash origin address 0x8000000 + FLASH_BOOT_AREA_SIZE (line 51)
- Menu->Project->Build All or CTRL+B
- Wait for the expected message: "Build Finished. 0 errors, 0 warnings"

Appendix B Programming

- Change directory: stm32cube_fw_u0\Projects\NUCLEO_C071RB\ROT_Provisioning\OEMiSB>
- Execute: /provisioning.bat
- Follow the instructions and verify the log messages (apply Building):

```
•    run config Appli with windows executable
•    =====
•    ===== Provisioning of OEMiSB boot path
•    ===== Application selected through env.bat:
•    ===== Applications/ROT/OEMiSB_Appli
•    =====
•
•    Step 1 : Product configuration
•       * Define final RDP value
•           [ 0 | 1 | 2 ]: 2
•
•       * Define data area size in Kbytes
•           [ 0 | 8 | 16 | 24 | 32 | 40 | 48 | 56 | 64 ]: 64
•
•    Step 2 : Projects generation
•       * Boot project generation
•           Open the OEMiSB_Boot project with preferred toolchain and rebuild all files.
•           Press any key to continue...
•
•       * Application project generation
•           Open the OEMiSB_Appli project with preferred toolchain and rebuild all files.
•           Press any key to continue...
•
•    Step 3 : Product programming
•       * Remove protection and flash erase
•
•       * Project flash programming
•           - OEMiSB application programming
•           - OEMiSB application SHA256 programming
•           - OEMiSB boot programming
•
•       * Configure Option Bytes:
•           - Write Protection
•           - Securable Memory Area Protection
•           - Boot Lock
•           Press any key to continue...
•
•       * Setting the final RDP Level 0
•
•    =====
•    ===== The board is correctly configured.
•    =====
```

- Connect Tera Term console on serial port “STMicroelectronics ST-LINK Virtual COM Port” configured at 115200 bauds, one start bit, one stop bit, no parity
- Reset the platform
- Verify the application aliveness message:

```
•    =                                                                     =
•    =                          OEMiSB User App                            =
•    =======================================================================
```

Revision history

Table 6. **Document revision history**

Date	Version	Changes
17-Jun-2025	1	Initial release.

Contents

1 General information 2

1.1 Platform reference 2

1.2 Included guidance documents 2

2 Reference documents 3

3 TOE preparative procedures 4

3.1 Secure acceptance 4

3.2 Secure installation and preparation of the operational environment (AGD_PRE.1.2C) 5

3.2.1 Hardware and software setup procedures 5

3.2.2 Secure preparation 5

4 Operational user guidance 7

4.1 User roles 7

4.2 Operational guidance for the integrator roles 7

4.2.1 User-accessible functions and privileges (AGD_OPE.1.1C) 7

4.2.2 Available interfaces and methods of use (AGD_OPE.1.2C and AGD_OPE.1.3C) 8

4.2.3 Security-relevant events (AGD_OPE.1.4C) 10

4.2.4 Security measures (AGD_OPE.1.6C) 10

4.2.5 Modes of operation (AGD_OPE.1.5C) 10

Appendix A Building 11

Appendix B Programming 12

Revision history 13

List of tables 15

List of figures 16

List of tables

Table 1. Applicable products 1
Table 2. Specific acronyms 2
Table 3. Platform reference 2
Table 4. Guidance documents 2
Table 5. Reference documents 3
Table 6. Document revision history 13

List of figures

Figure 1. TOE perimeter 7