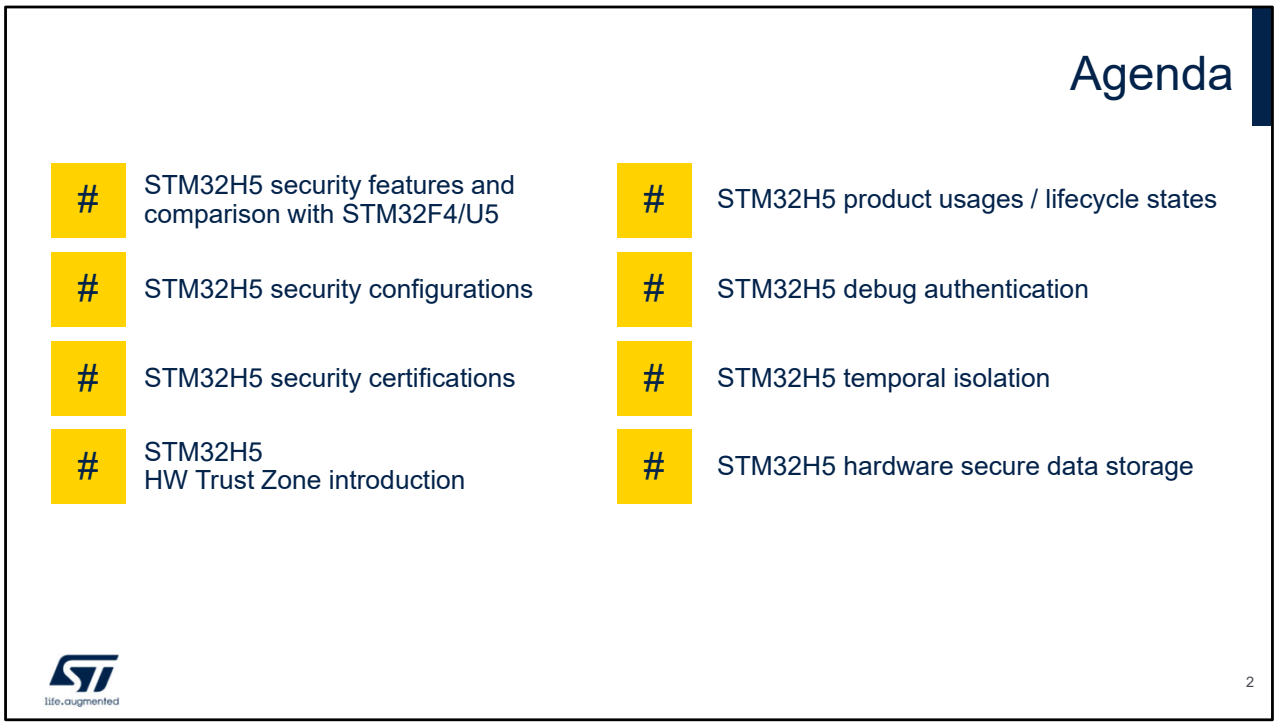
Hello, and welcome to this overview of the STM32H5 security architecture.



The following topics of the STM32H5 are going to be explained:

- Security features of STM32H5 and comparison with STM32F4 and STM32U5
- Configurations
- Certifications
- Hardware trustzone introduction
- Product usages and lifecycle states
- Debug authentication
- Temporal isolation
- Hardware secure data storage.

Let us start with the description of security features implemented in STM32H5 and compare them with the ones present in STM32F4 and STM32U5.

STM32H5 Security features and comparison vs STM32F4/U5

- Security target PSA level3 & SESIP3 certifications

- Security:
  - Debug Authentication Control
  - Device Life Cycle
  - Hardware Unique Key (HUK): to get a secure storage resistant to logical, side and physical attack
  - Root of Trust, Secure Boot and Secure firmware Install
  - HW Secure storage
  - Resource isolation using TrustZone
  - Cryptographic functions:
    - 2x AES 256, one with SCA resistance
    - AES and PKA, side attack resistant by HW. ECC up to 640 bits and RSA up to 4160 bits
    - HASH: SHA-1, SHA-2 (up to 512)
    - TRNG
  - On The Fly Decryption on External OctoSPI Flash
  - Active tamper detections

life.augmented

4

STM32H5 targets PSA Level3 and SESIP3, passing tests for logical, board, and basic physical resistance that confirm a substantial level of cyber protection.

- PSA Level3 stands for Platform Security Architecture level 3. It establishes trust through a multi-level assurance program for chips containing a security component called a PSA Root of Trust (PSA-RoT) that provides trusted functionality to the platform. The multi-level scheme has been designed to help device makers and businesses get the level of security they need for their use case.
- SESIP3 stands for Security Evaluation Standard for IoT Platforms. The SESIP, published by GlobalPlatform, defines a standard for trustworthy assessment of the security of the IoT platforms, such that this can be re-used in fulfilling the requirements of various commercial product domains.

SESIP3 Assurance (SESIP3) is a traditional white-box vulnerability analysis. The evaluation is structured around a time-limited source code analysis combined with a time-limited penetration testing effort.

The following security features are implemented in the STM32H5:

- Debug authentication control
- Device life cycle
- Hardware unique key, called HUK
- Root of Trust, Secure Boot and Secure firmware Install
- Hardware Secure storage
- Resource isolation using TrustZone
- Cryptographic functions:
    - 2x AES 256, one with SCA resistance
    - AES and PKA, side attack resistant by hardware
    - HASH: SHA-1, SHA-2 (up to 512)
    - True Random Number Generator
- On The Fly Decryption on External OctoSPI Flash
- Active tamper detections

STM32H5 Security features and comparison vs STM32F4/U5

- Provide IPs, functions & services to better protect your assets
- Offer products & services security assurance level to ensure required trust
  - Focusing on 2 de-facto industry standard product certification schemes:

SESIP	**Security Evaluation Standard for IoT Platforms (SESIP)** ➢ Published by Global Platform for IoT devices
psacertified	**Platform Security Assurance by ARM® (PSA)** ➢ Focusing to protect IoT devices

STM32 Trust

- Our targets:
  - All STM32 with SBSFU have means to offer at minima PSA L1 equivalent level
  - All new secure products target at minima PSA Level3 and SESIP3
    - Starting from STM32L5

life.augmented

5

Security has many layers and multiple levels.
Its implementation depends on various factors such as the product capabilities, the application requirements, the security assurance level requirements.
Therefore, STM32Trust based its offer upon a set of scalable security functions, services and ecosystem offer.
Security Assurance is a key item within STM32Trust offer. It represents the trust ST can provide to our customers into our product ability to help secure our customer assets.
The choice of PSA and SESIP was made as an independent source of evaluation for building that trust, lowering the necessary investments of our customers in Security Engineering.

These will also help some of our customers quickly engineer

their platforms or devices for proprietary or standardized Security Assurance levels required by their ecosystems.
In particular SESIP, from Global Platform, is bridging the gap with many other certification bodies, by aligning the protection profiles with the application requirements.

Our portfolio offer products with or without security. We also have very old products that were not designed with and for security.
Our target is to provide a Security assurance level for all our products embedding security (Hardware or software). Legacy products embedding Secure Boot and Secure Firmware Update (SBSFU) will provide PSA Level1 & SESIP1 equivalent levels at minima, and if certifications stamps are not present evaluations will be done and can be retrieved.
Starting from STM32L5, ST decided to raise the bar and offers PSA Level3 & SESIP3 on all new secure products, allowing to move forward on security assurance to a much higher level.

STM32H5 Security features and comparison vs STM32F4/U5

Feature	STM32F4	STM32U5	STM32H5	Comments
Native secure boot	No	No	**ST- iROT**	**ST-iROT = ST secure boot code ROM'ed**
ROT solution	No	TFM + SBSFU (x-cube package)	OEM-iROT+TFM (as example)	**Customer provisioning of ROT**
SFI (secure FW install)	No	Native		**On specific part number**
Device life cycle	3 levels (RDP)	4 levels (RDP) **Password based regression**	6 levels (Product states) **+2 Debug authentications**	**Debug authentication :** ➢ Control of debug re-opening + regression
FW IP protection	PCROP	Device life cycle (RDP1/2) TrustZone (RDP 0.5) TZ + MPU (e.g. TFM level3)	Trustzone +MPU **Secure Manager** (Installable Services) or TFM level3	
Temporal Isolation	No	Secure HDP – 1 level	3 temporal levels	**STM32H5 isolates NVM code/data and Keys compartments**
Runtime isolation	No	TrustZone Privilege /Non-privileged (Peripherals + **Memories**)		
HW crypto accelerator	CRYP / HASH / RNG	AES / HASH / TRNG AES DPA / PKA DPA (RSA 4K)	AES / HASH / TRNG AES DPA / PKA DPA(RSA 4K)	**STM32H5:** ➢ **SAES (all modes)** ➢ **SHA1 to SHA2, up to 512-bit**
HUK	No	HUK	HUK	
HW-Secure storage	No	Based on HUK 1 S / 1 NS	**5 secure storage domains** ➢ 4 HDPL Secure + 1 NS ➢ Includes Flash Secure Storage	**STM32U5: Software management of key storage** **STM32H5: Native support of key storage inside FLASH interface (enabling constrained debug feature)**
Anti -Tamper	Non-active tamper	4x active pair of tamper pins Volt. &Temp. monitoring (Vbat) Total tamper I/Os: 8	**2x** active pair of tamper pins. Volt. &Temp. monitoring **(Vbat)** Total tamper I/Os: **11**	**All devices include clock, volt. & temp monitoring**
Certification Target	No	PSA Level3 SESIP3 (ST PP)	PSA Level3 SESIP3 (ST PP)	**Consecutive hardware robustness improvements from STM32L5 to STM32H5**

life.augmented

6

This table compares the security features of three microcontrollers: STM32F4, STM32U5 and STM32H5.
The new features offered by STM32H5 are highlighted.
First, ST iROT (immutable root of trust) secure software is a secure software in System Flash memory used to manage Secure Boot & Secure Firmware Update of the first Updatable code in the system.
The STM32H5 implements a device life cycle state machine, supporting 6 states and two debug authentication mechanisms.
In the STM32H5 devices, the hardware and software resources used to boot can be isolated. This is called temporal isolation.
Temporal isolation uses secure hide protection (HDP) levels
According to Arm PSA security model, the number of temporal

isolation levels for STM32H5 is 3 (HDPL0, 1 and 2), while HDPL3 Secure / non-secure protection is based on runtime isolation.
Globally 11 passive and active tampers are available on I/Os.

STM32H5 Security features and comparison vs STM32F4/U5

Unit	Feature	Available when export control ?
AES	Key size: 128 / 256 Modes: ECB + CBC + CTR + GCM/GMAC + CCM	N
SAES	SCA resistant AES Key size: 128 / 256 Modes: ECB + CBC + CTR + GCM/GMAC + CCM	N
HASH	SHA1, SHA2, SHA2-384, SHA2-512	Always available
PKA	SCA resistant RSA 2048, 3184, 4096, EDCSA	Limited to EDCSA Signature verification when export control
RNG	NIST SP 800-90B compliant entropy source	Always available
OTFDEC	Uses AES-128 in counter mode to achieve the lowest possible latency	N

AES, HASH, PKA Crypto IPs are clocked with system clock
System clock for STM32H5 is 250 MHz, compared to 110 MHz for L5 or 120 MHz for U5 → Crypto perf = multiplied by 2

SAES is now clocked at 250 MHz on STM32H5
It was clocked with a 48 MHz on STM32U5 → Crypto perf = multiplied by 5

life.augmented

7

This table lists the security acceleration units present in the STM32H5.
Some of them are disabled in products satisfying export control rules.
Symmetric cryptography is accelerated by means of AES and Secure AES modules. Secure AES, or S.A.E.S is resistant against Side Channel Attack (SCA).
The following hash algorithms are supported in hardware: SHA1, SHA2, SHA2-384 and SHA2-512.
The Public Key Accelerator (PKA) is intended for the computation of cryptographic public key primitives, specifically those related to RSA, Diffie-Hellmann or ECC (elliptic curve cryptography) over GF(p) (Galois fields).
The RNG is a true random number generator that provides full entropy outputs to the application as 32-bit samples.

OTFDEC allows on-the-fly decryption of the AHB traffic based on the read request address information.
Typical usage is decrypting code and data read from external non-volatile memory.
The performance of AES and SAES units is boosted on STM32H5, due to frequency increase.

Let us describe now the configuration of the platform security.

STM32H5 Platform Security Configurations

Reference	Embedded flash size	Crypto accelerators	Configurable features	
			TZEN	Root of Trust
STM32H573	2 MB	All crypto units are enabled (Export control required)	TZEN = 0 : When no runtime isolation is needed. TZEN = 1: if runtime isolation needed ➢ Based on TrustZone	❖ ST-iROT integrated in the System Flash (ONLY on Crypto parts (H57x) & TZEN=1) ➢ iROT part of certification scope ➢ Taking care of Key management, Secure Boot, Secure Firmware Update, Product Lifecycle, .. ❖ OEM-iROT ➢ OEMs can define their own iROT with limited dependencies on ST code ➢ Should manage similar feature than a ST-iROT (Secure Boot, Secure Firmware Update, Product Lifecycle, ….)
STM32H56X	2 MB	Not subject to export control/ Crypto disabled		
STM32H503	128 KB	Aligned with TZEN=0		

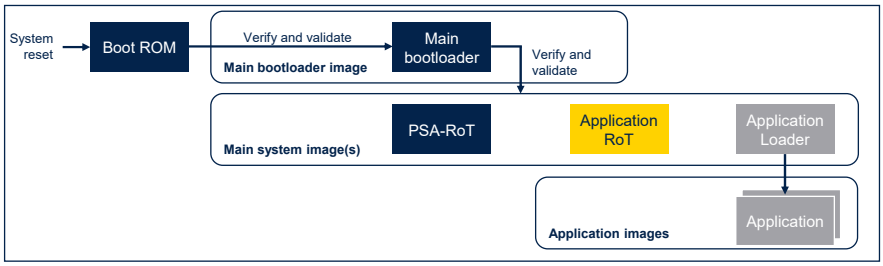
life.augmented

9

The configuration of the platform security depends on the reference of the STM32H5.
The STM32H573 supports all features. Trustzone can be enabled or disabled through an option byte.
When Trustzone is enabled, the root of trust firmware can either be the one designed by STMicroelectronics, which is certified, or one developed by the OEM.
STM32H56x does not embed cryptographic accelerators and is therefore not subject to export control.
STM32H57x embeds cryptographic accelerators and is therefore subject to export control.
Note that TrustZone can be enabled in STM32H56x, but an OEM iROT is required.
TrustZone is deactivated by default in all STM32H563/H573 and STM32H562 devices.
The STM32H503 does not support Trustzone.

STM32H5 Platform Security Configurations

- Extract from ARM PSA Security Model:
  - The secure boot flow must start with an inherently trusted Boot ROM in the Immutable PRoT; this is the trust anchor for the boot validation chain
    - Both the following are recommended:
      - The Boot ROM is small, simple, and verifiable
        - This minimizes the risk of a vulnerability that cannot be corrected once on the chip
      - The complex steps are handled by a main bootloader, which is subject to validity check by the Boot ROM, because it can be corrected through a secure firmware update process



life.augmented

10

All devices must support a secure boot flow to ensure only authorized software can be executed on the device.
Secure boot uses cryptography to verify the next stage code and any metadata.
The example boot flow represented in the figure shows the Main Bootloader verifying and validating the Platform and Application Root of Trust codes.

1. A device must always boot from a fixed address in the Boot ROM following system reset.
2. The Boot ROM verifies and validates all images that are associated with the next stage before executing any next stage code.
3. The next stage verifies and validates all data and images that are associated with the following stage before executing any of the following stage code.

This process is repeated until all the chained images have been verified and validated.

STM32H5 Platform Security Configurations

- Definitions
  - Security services: Services natively present (immutable) in the STM32 (In System Flash)
    - Covers:
      - Bootloader: Manage local connectivity (SPI, I2C, I3C, …) allowing device provisioning
      - RSS: ST immutable code allowing to install securely an extension (RSS-e)
      - ST-DA: ST Debug Authentication – Allows secure control of Regressions and Debug re-opening
      - RSS-Lib: Library providing basic natives services (HDPL incrementation, OBKey provisioning)
      - ST-iROT (Only H57x) : ST Immutable Root Of Trust
  - ROT (Root Of Trust): Primary root of trust of a device
    - ROT: immutable ROT and uROT for Updatable ROT
    - iROT Covers secure boot and firmware update
      - Can be natively implemented ➔ST-iROT
      - Can be implemented by the OEM, using write protection mechanisms to set it as immutable ➔OEM-iROT

life.augmented

11

The platform comes with native security services, embedded in the system memory to manage the root of trust services. Native root of trust services take care of platform security, they include :

- Bootloader
- Root Security Services (RSS)
- ST Debug Authentication
- RSS-library
- ST-iROT.

The Root of Trust is split into two parts:

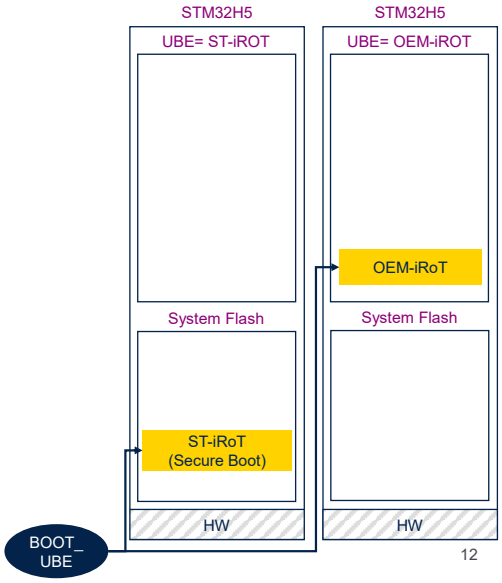
- The iROT that is immutable
- The Updatable ROT (uROT).

Security services provide the root of trust of the platform managing the verification and the update of the first updatable code, which is the uROT.

STM32H5 Platform Security Configurations

- H5 configuration
  - Secure Boot Entry point
    - ST-iROT: Immutable Secure Boot Natively present
    - OEM-iROT
      - Immutable Secure Boot to be developed & installed by OEMs
      - or any code in User Flash to boot from

BOOT_UBE (STM32H57x only): Select Secure Boot entry point between ST-iROT or UserFlash (implementing an OEM-iROT as example)



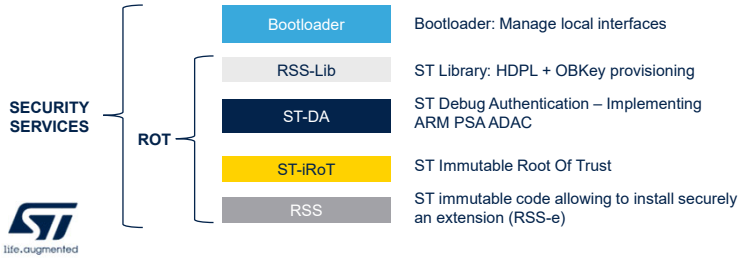
life.augmented

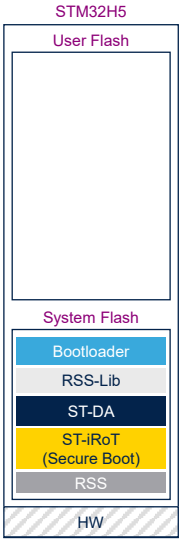
12

The first step in the boot sequence is the jump to entry point of the iROT, called the Unique Boot Entry (UBE).
The secure boot address is selectable through an option byte.
Either the ST-iROT is used or an OEM-iROT that has to be installed.
ST-iROT is natively present only in the flash of STM32H57X.

STM32H5 Platform Security Configurations

- STM32H5 configuration
  - STM32H5 comes with Security Services
    - Secure Boot Secure firmware update ➔ ST-iROT
    - Debug Authentication Control (in field Lifecycle management) ➔ST-DA
    - Initial Installation services ➔RSS





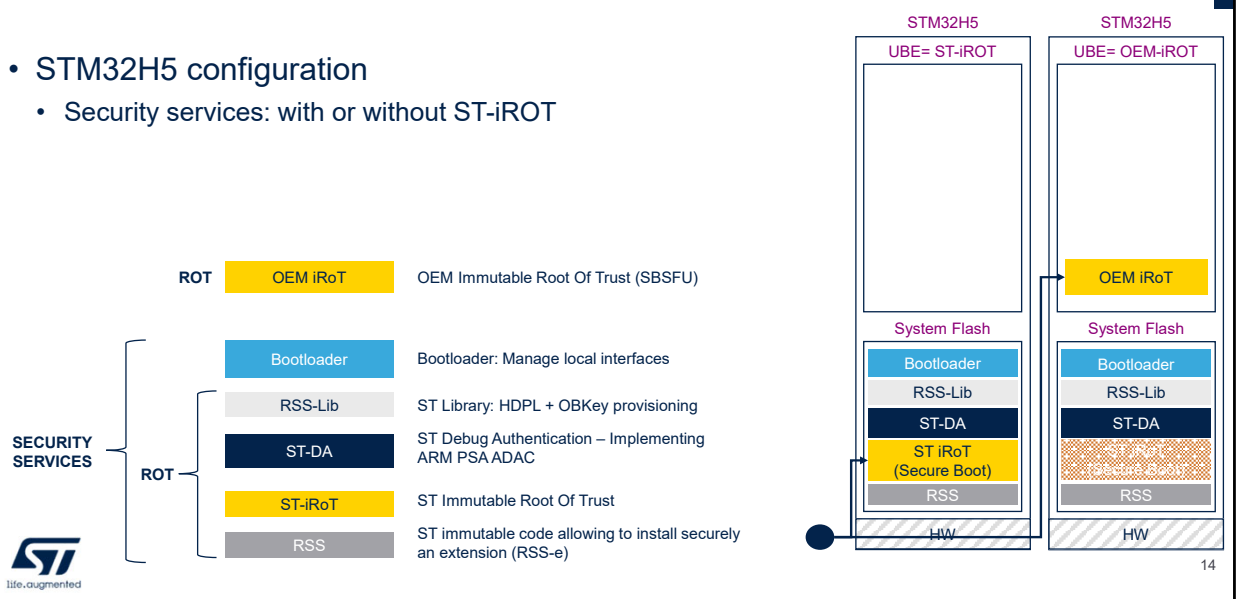
13

The following security services, stored in system flash, are available:

- Root Security Services, allowing to install securely an extension
- ST-iROT
- ST-Debug Authentication, enabling to re-open the debug when the product is in the field
- RSS-library
- Boot loader.

STM32H5 Platform Security Configurations

- STM32H5 configuration
  - Security services: with or without ST-iROT

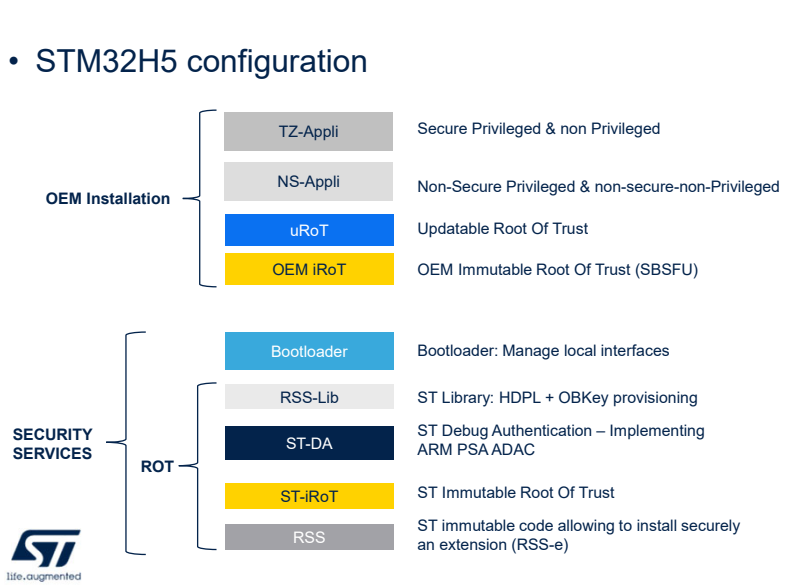


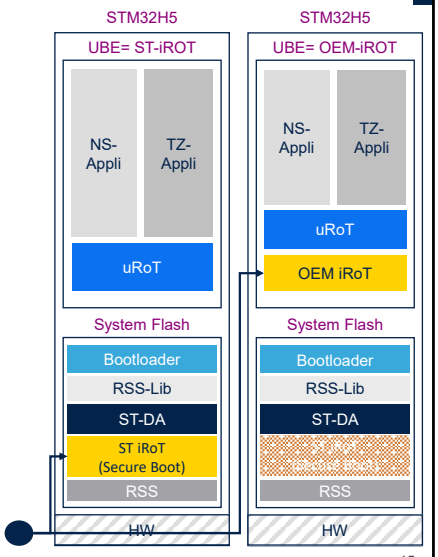
The STM32H573 offer two options to execute the immutable code after a reset:

- Security services when ST-iROT is selected managing the secure boot of the next boot level
- Proprietary boot entry when the OEM manages the full chain (OEM-iROT), to be installed in user flash memory with proper security activation.

STM32H5 Platform Security Configurations

- STM32H5 configuration





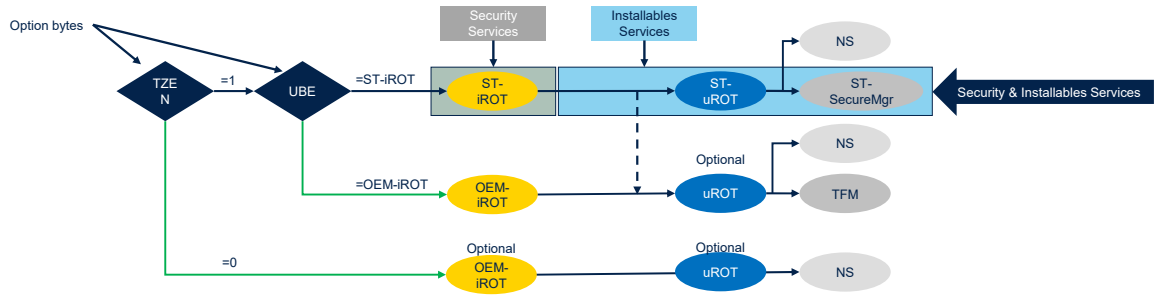
15

The next step consists in verifying the updatable root of trust (uRoT).
Then the uROT verifies the trustzone application image and the non-secure application image.
The secure boot chain is now completed.

STM32H5 Platform Security Configurations

- H5 configurations

STM32H57x



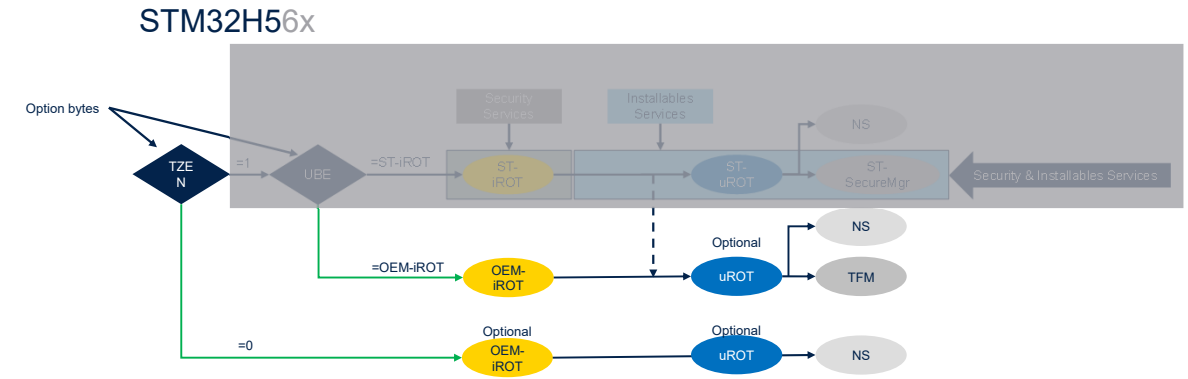
life.augmented

16

The STM32H57X supports all the steps of the secure boot described in the previous slides.
Trustzone can be enabled or disabled.
When enabled, either the ST-iROT or OEM-iROT is executed after the reset.

STM32H5 Platform Security Configurations

- H5 configurations

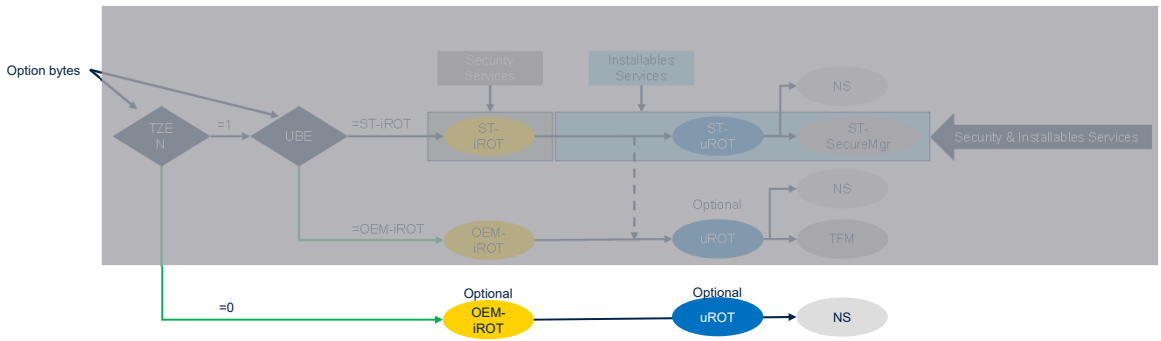


In the STM32H56X, AES and SAES are not active and ST-iROT is not present.
Trustzone can be enabled or disabled.
When enabled, the OEM-iROT is executed after the reset, capable of verifying the non-secure image and the secure image, which is typically based on trusted-firmware for Cortex-M (TF-M).

STM32H5 Platform Security Configurations

- H5 configurations

STM32H50x



life.augmented

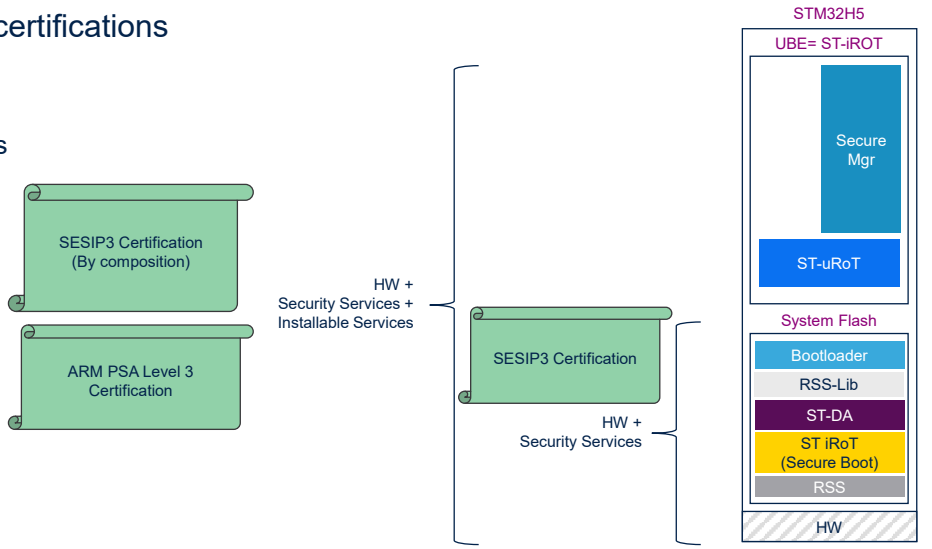
18

The STM32H50X does not support trustzone.
Only a non-secure image can be executed.
OEM-iROT and uROT are optional in this case.

The security framework designed by STMicroelectronics for the STM32H5 is certified by two independent authorities: SESIP and PSA.

STM32H5 Security Target Certifications

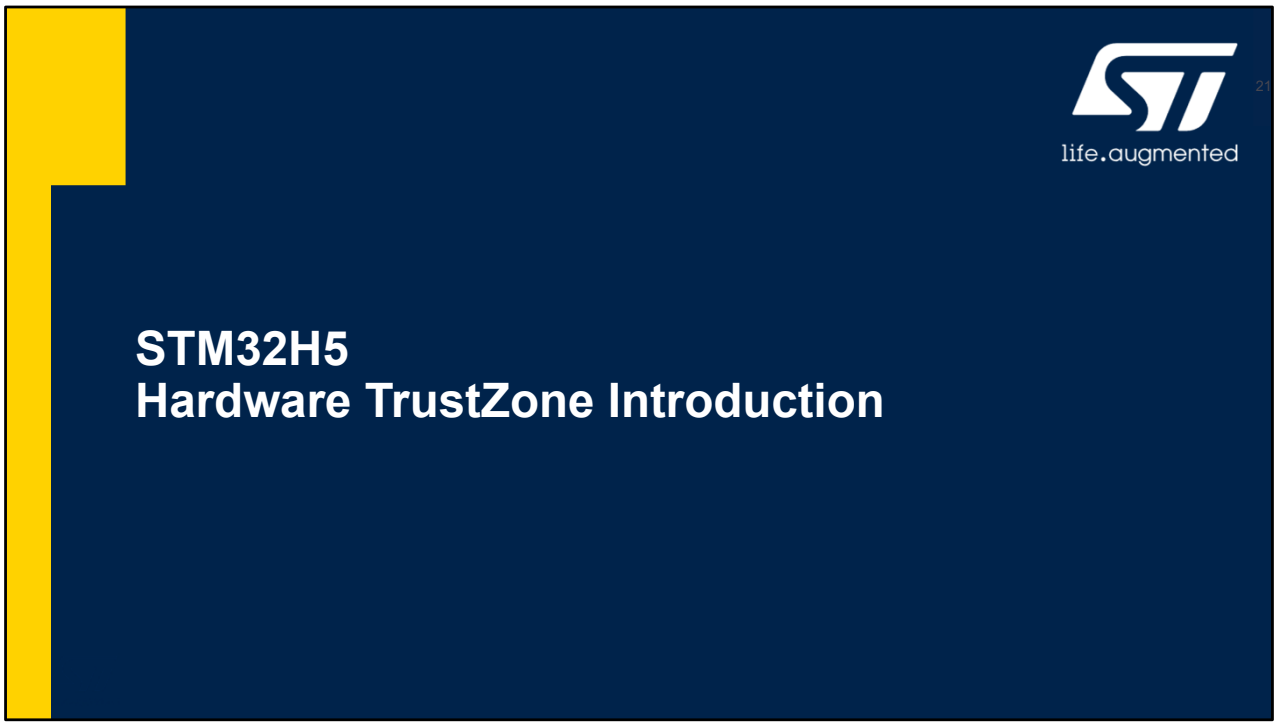
- STM32H5 target certifications
  - Security Services
    - SESIP3
  - Installable Services
    - SESIP3
    - ARM PSA Level3



life.augmented

20

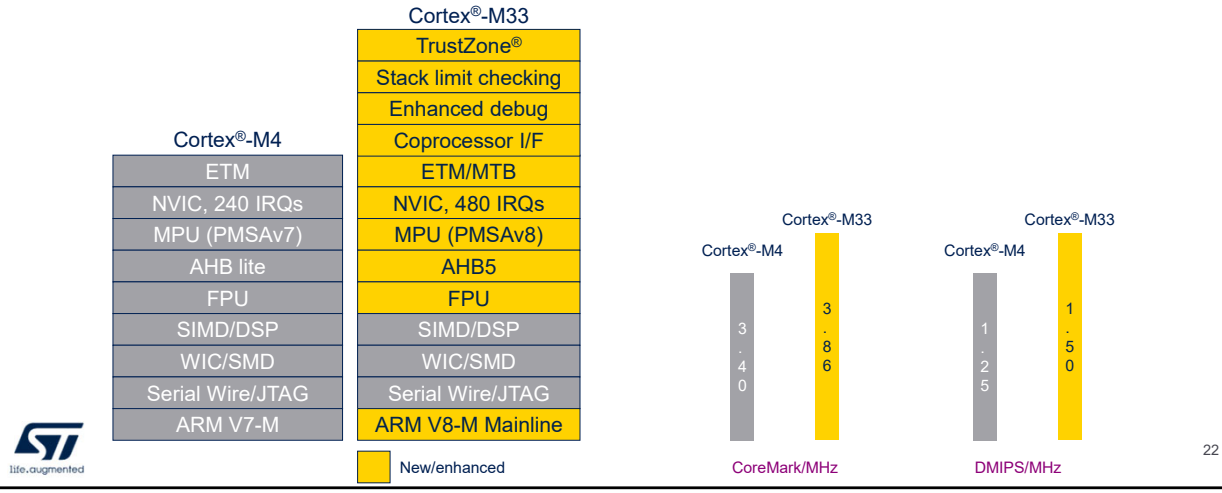
Arm PSA and SESIP certifications cover hardware and software implementations on the STM32H5 that protect code and users from security attacks.
The hardware of the STM32H5 targets the SESIP3 certification.
Regarding the firmware, the Security services target the SESIP3 certification, and the installable services target the PSA level 3 and SESIP3 certifications.
Thanks to the PSA Certified and SESIP3 certifications, STMicroelectronics can offer greater assurances against physical and remote attacks.



TrustZone reduces the potential for attack by isolating the critical security firmware and private information, such as secure boot, firmware update, and keys, from the rest of the application.

STM32H5
Hardware TrustZone Introduction

- Cortex-M33 enhancements vs. Cortex-M4

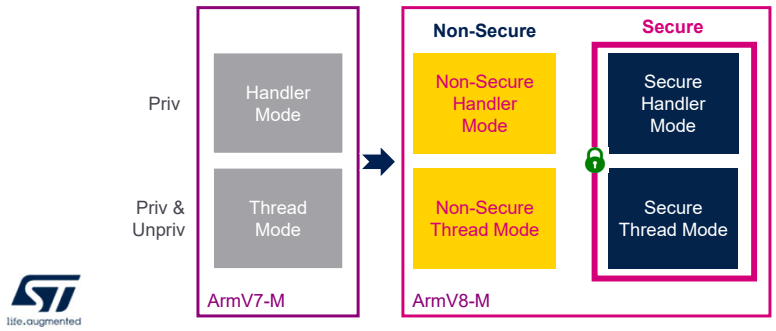


This slide highlights the differences between Arm Cortex-M4 and Cortex-M33 cores.
The V8-M mainline is a superset of the V7-M architecture.
TrustZone is a new feature enabling the core to switch between two security states: secure and non-secure.
The AHB buses used to communicate with the STM32H5 bus matrix are compatible with the AHB5 specification, which supports the secure attribute.
TrustZone is by default disabled in the STM32H5 and in this case, the Cortex-M33 offers the same kind of features as the Cortex-M4, some of them being enhanced.
An interesting new capability is runtime stack overflow checking, which is achieved by programming stack limit registers.
Note that the PMSAv8 MPU present in the Cortex-M33 is not

compatible with the PMSAv7 MPU present in the Cortex-M4. In terms of performance, the Cortex-M33 beats the Cortex-M4 as indicated by the results of CoreMark and DMIPS benchmarks.

STM32H5
Hardware TrustZone Introduction

- Security state in ARMv8-M (Cortex-M33)
  - Addition of an extra processor state → secure / non secure
- Thread mode ➔ Privileged or unprivileged mode



life.augmented

23

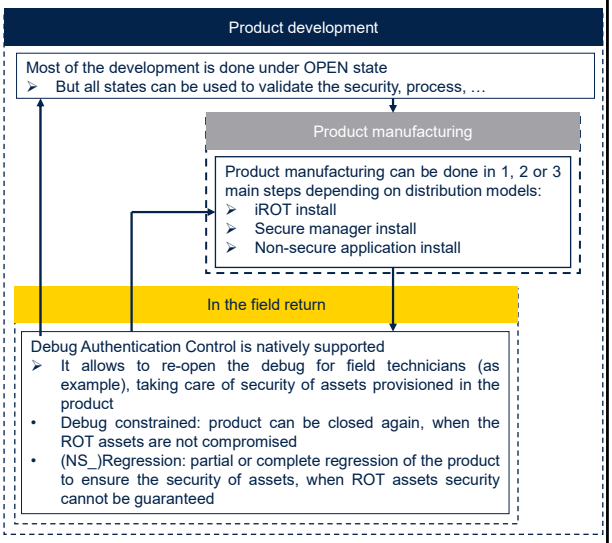
The ARM V8-M architecture implements TrustZone for Cortex-M cores.
As explained in the figure, the V7-M cores support one kernel running in privileged state, in charge of switching unprivileged tasks.
When the security features of the core is enabled, there are 2 orthogonal security states: secure and non-secure.
The non-secure kernel and applications run in non-secure state while secure kernel and applications run in secure state.
Security state of the processor depends on the address at which the instruction was fetched.
Each security state supports both privileged and unprivileged user access.
When TrustZone feature is not enabled the programmer's model only include non-secure state.

For each security state the processor can operate in Thread or Handler mode.
At reset or return from an exception, the processor enters in Thread mode – both privileged and unprivileged programs can run in Thread mode.

Let us describe now product usages and Lifecycle states.

STM32H5
Product usages / Lifecycle states

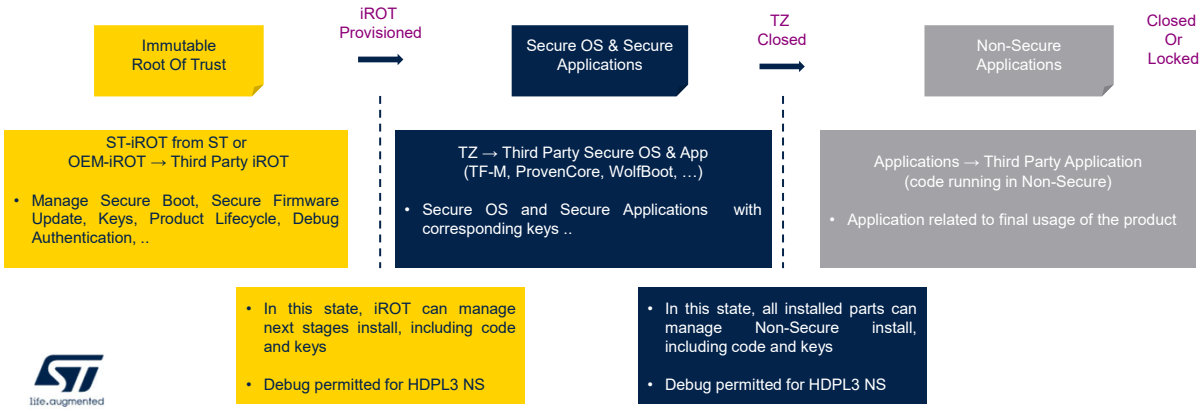
- The product is used in different contexts
  - Product development
  - Product Provisioning considering different cases (trusted/non-trusted environments)
  - In the Field management



The product life-cycle allows to control access to different assets (code and data) of the product, including during development, manufacturing, and after sales.
It allows to provision the product with different distribution models, taking care on the code and data provisioned.
The full view of the life cycle includes the debug authentication part, to help product maintenance to manage field returns.
The debug authentication control is ensured thanks to a protocol based on Arm PSA ADAC specification

STM32H5
Product usages / Lifecycle states

- Multi-OEMs model
  - New Lifecycle consider 3 Main bricks to be installed in the product that could rely on 3 different parties



The STM32H5 life-cycle is designed to support provisioning of the product with up to 3 different parties.

- The First party is the Immutable Root-of-Trust (iROT).
- The Second party is the secure operating system and secure applications.
- The Third party is the non secure application.

We are differentiating the Initial provisioning, from the full product provisioning.

Typically, for an install with 3 parties, we can consider:

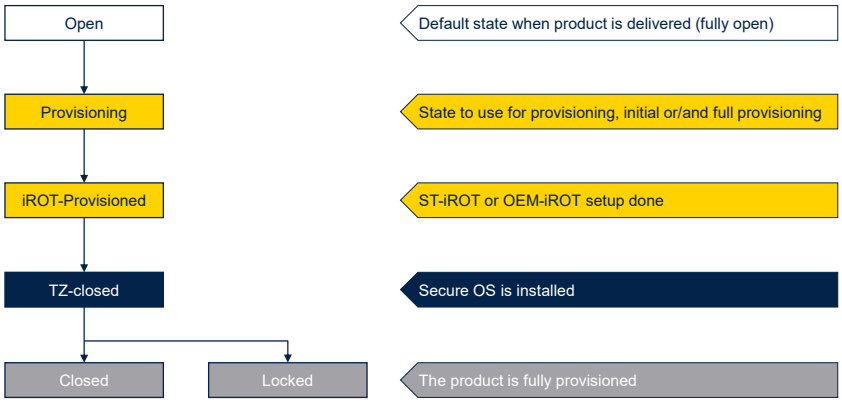
- iROT (firmware and data) to be provisioned in Provisioning PRODUCT_STATE, then to move to iROT-Provisioned.
- Then SecureOS & Secure application being installed (update mechanism of iROT), then PRODUCT_STATE

passed in TZ-Closed

- Then the Non-Secure application being installed (update mechanism of uROT (part of SecureOS)), then PRODUCT_STATE passed in Closed.

STM32H5
Product usages / Lifecycle states

- Simplified Lifecycle / TZEN=enabled



life.augmented

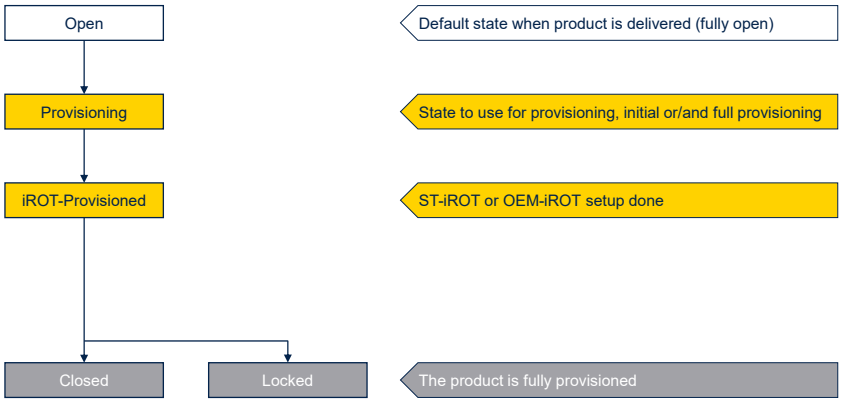
27

This slide introduces the life cycle state diagram, assuming that trustzone is enabled.
It is simplified, because it does not include the field return additional states.
The list of product states from top to bottom is:
- Open, default state when no firmware is installed
- Provisioning
- iROT-Provisioned
- TZ-closed
- Closed
- Locked.

The supported transitions can be requested through the debug interface or via the system bootloader.

STM32H5
Product usages / Lifecycle states

- Simplified Lifecycle / TZEN=disabled

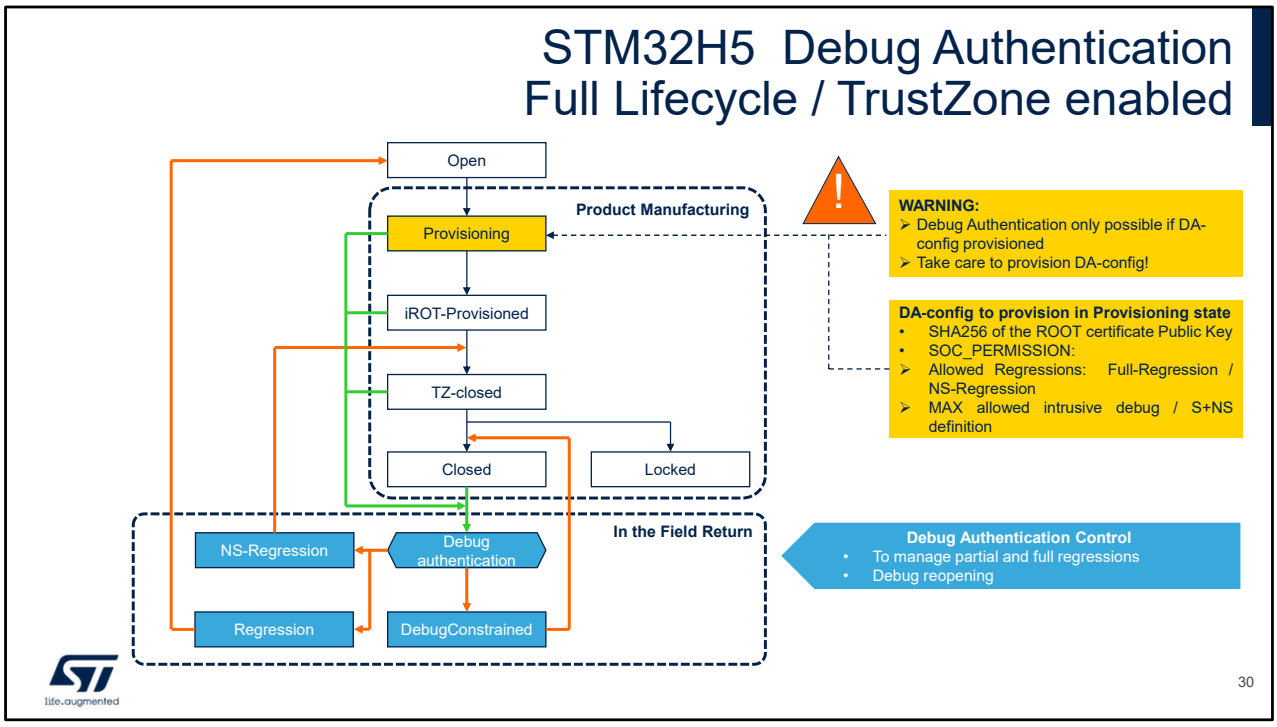


life.augmented

28

When TrustZone is disabled, the state TZ-Closed does not exist.

When product state is closed, the debug authentication procedure allows a trusted debugger to reopen access without compromising sensitive information.



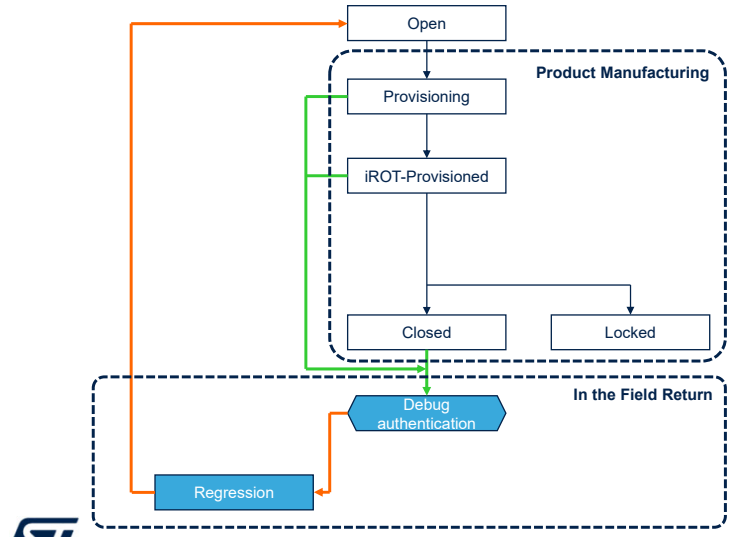
To control re-opening or manage regressions of the debug, the device imposes a debug authentication protocol.
If the device is in CLOSED (product life cycle) state, the debug state is CLOSED.
The debug authentication procedure allows a trusted debugger to reopen access without compromising sensitive information called the Root Of Trust (ROT).
Reopening the debug is possible only if sensitive asset security is ensured and TrustZone is enabled. This is called Constrained Debug, as constraints ensure the security of the ROT information.
Alternatively, a partial or full regression mechanism can be used when security of sensitive information cannot be guaranteed.
This is called Regression, as regression ensures removal of

the sensitive information before reopening the debug.

- Partial regression corresponds to releasing non-secure code and assets; the intermediate state which allows partial regression management is called NS-Regression.
- Full regression corresponds to releasing all code and assets; the intermediate state allowing full regression management is called Regression.

For STM32H57x microcontrollers, the debug authentication configuration must be done only when the product state is “Provisioning”, it cannot be performed when product state is “Open”.

STM32H5 Debug Authentication Full Lifecycle / TrustZone disabled



In TZEN = 0, Authentication method is PASSWORD

- Debug Authentication state can be launched when a reset is done, with "STDA" posted in the DBGMCU mailbox (green arrows)
- This state is NOT a specific PRODUCT_STATE
  - But it allows to execute the ST-DA embedding the ARM PSA ADAC protocol through JTAG or SWD, with a host
  - It is executed in HDPL=1
  - The only permitted action is to launch a full regression

Debug Authentication Control

- To manage full regressions

life.augmented

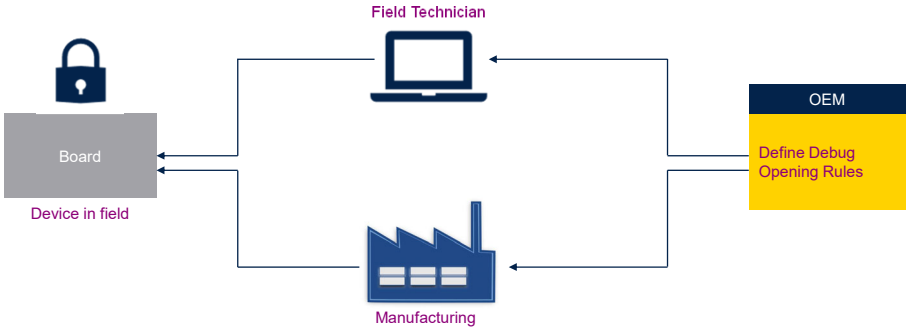
31

When trustzone is disabled, the only control, which is allowed, is a full-regression

Debug authentication control is based on passwords; only the HASH of the password has to be provisioned.

STM32H5 Debug Authentication

- **Objective**:
  - Provide to OEMs a way to allow Field Technicians to reopen the devices without compromising the security of sensible assets



life.augmented

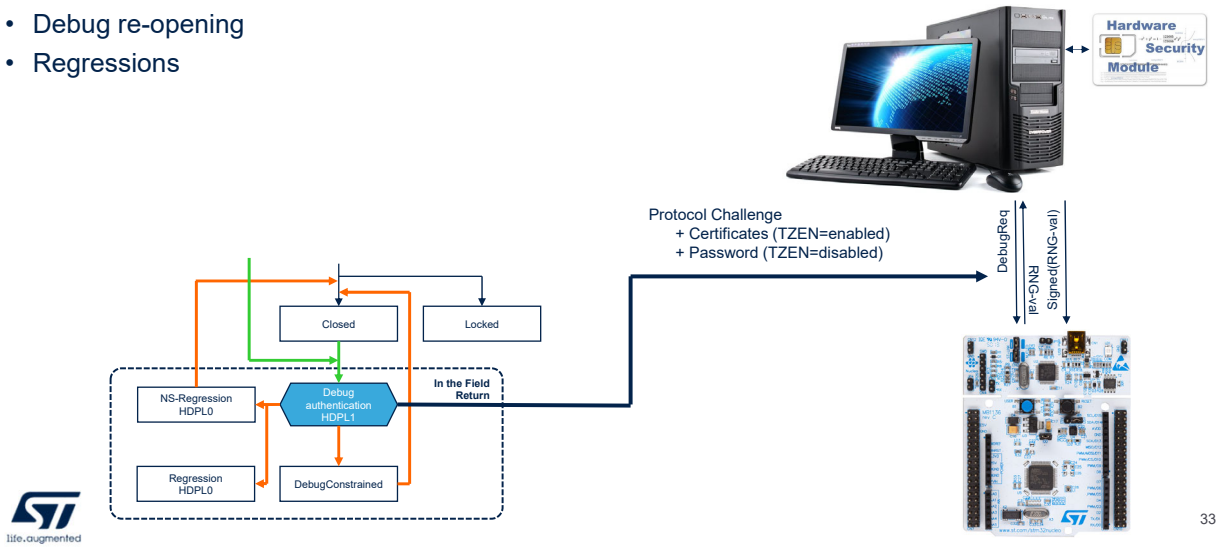
32

The following scenarios illustrate some of the flexibility of the authentication mechanism combined with certificates:

- Manufacturing equipment with a device-class certificate (and matching key stored in a hardware security module) is able to authenticate itself to the devices on the production line to initialize them (flash, credentials, root of trust. . .).
- A developer uses a device-locked certificate with a local key to debug their application on the device
- A technician connects diagnostics equipment to a device, the authentication token is generated in the cloud to unlock access and perform maintenance.

STM32H5 Debug Authentication

- Debug Authentication allows control of
  - Debug re-opening
  - Regressions



When the product state is Debug Authentication, the authenticated debug sequence occurs.

1. The external host requests to launch the debug authentication protocol, via the DBGMCU access port mailbox; the rest of the device is kept under reset.
2. The System, Boot and Security (SBS) selects the STMicroelectronics RSS-DA (debug authentication library) boot address, and requests the CPU to be released from reset
3. The CPU running RSS-DA library executes the debug authentication protocol in the system flash memory. If the device is closed, the access port control to the Cortex-M33 core debug is closed until RSS-DA acknowledges the authentication sequence start request
4. The authentication method depends on TrustZone

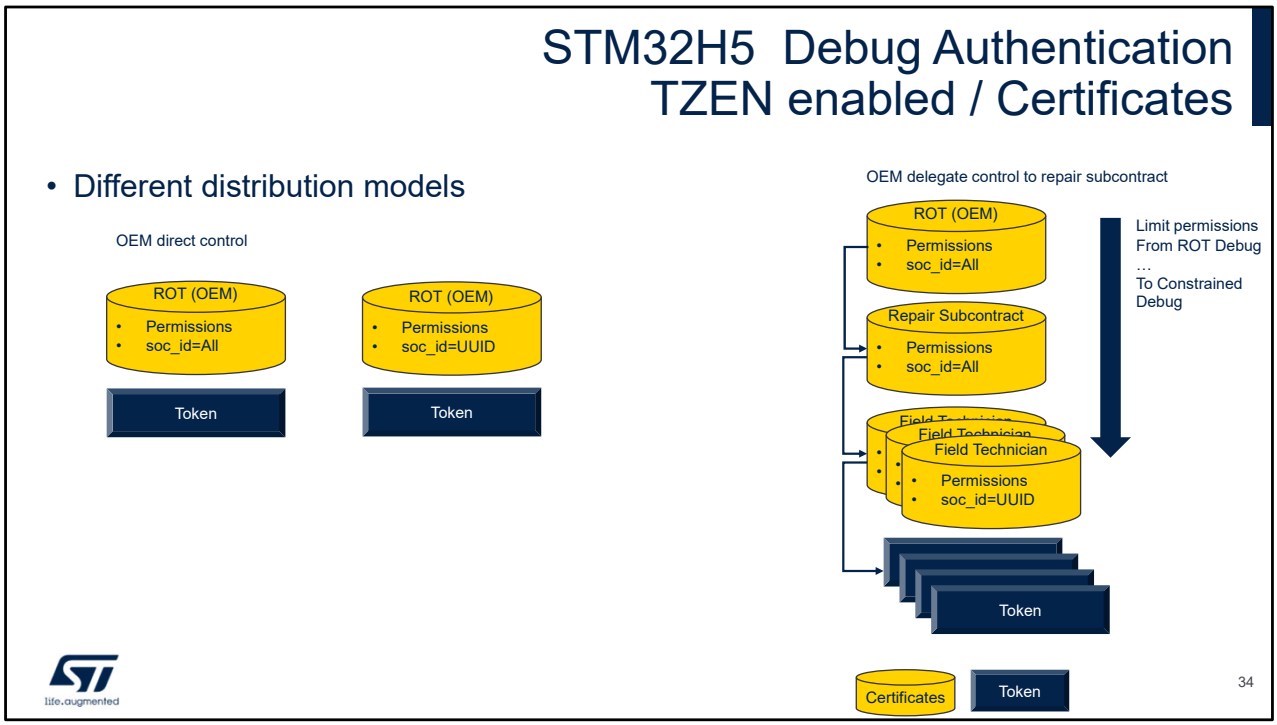
activation:

   1. When TrustZone is activated, the authentication method is based on certificates.
   2. When TrustZone is disabled, the authentication method is based on password. This method only allows the full regression of the product to be controlled

1. Above reopenings are effective only when HDPL in SBS_HDPLSR has a value equal or superior to the value programmed in DBG_AUTH_HDPL field in SBS_DBGCR register.

In case of authentication failure, the user is informed through the host interface.

The debug authentication library in system flash memory is available only when HDPL = 0 or 1. Only this library can perform the steps 3 and 4 described above.



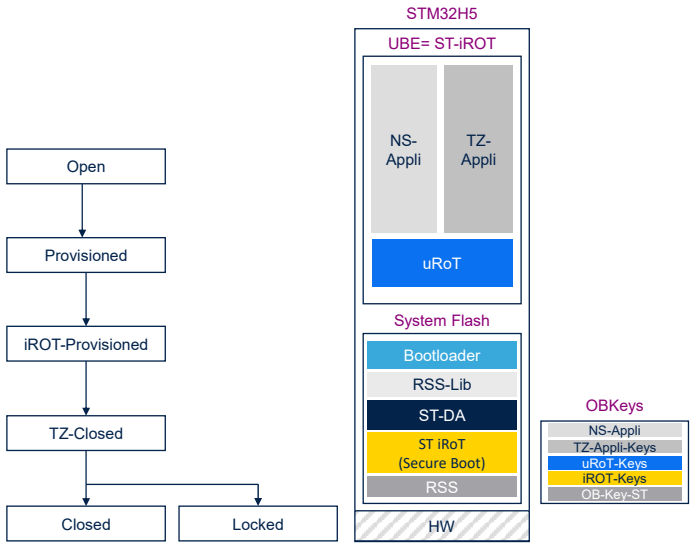
When TrustZone is activated, the authentication method is based on certificates.
As soon as a debug certificate chain is fully verified by the device, if the certificate concerns a debug permission, the RSS-DA programs the debug opening of the Cortex-M33.
Alternatively, the certificate can authorize partial or full regression, allowing debug on a regressed part.
The default mechanism for authentication relies on a challenge-response protocol:
The response to the challenge is a signed authentication token, also called the debug token.
A chain of certificates links the key used to sign to the authentication token to a set of one or more trusted anchors (roots of trust).
Based on the vendor needs, the chain can be of arbitrary

length, ending in a key directly linked to a programmed root authority.
In the figure on the left, the chain is limited to the OEM, while on the right it also includes the repair subcontractor and field technicians.
Adding intermediate steps in the certificate chain adds some overhead to the authentication process in the form of extra verification operations and increased data size.
However, intermediate certificates also limit the exposure of the most sensitive keys, allowing that those keys can be used less often and protected with added security.
The chain can be more or less long, depending on number of stakeholders in the product maintenance.
The Arm PSA-ADAC specification defines a certificate format to build trust chains and offer the flexibility to deal with complex scenarios.

In the STM32H563/H573 and STM32H562 devices, the hardware and software resources used to boot can be isolated.
This is called temporal isolation

STM32H5 Temporal Isolation

- STM32H5 configuration



life.augmented

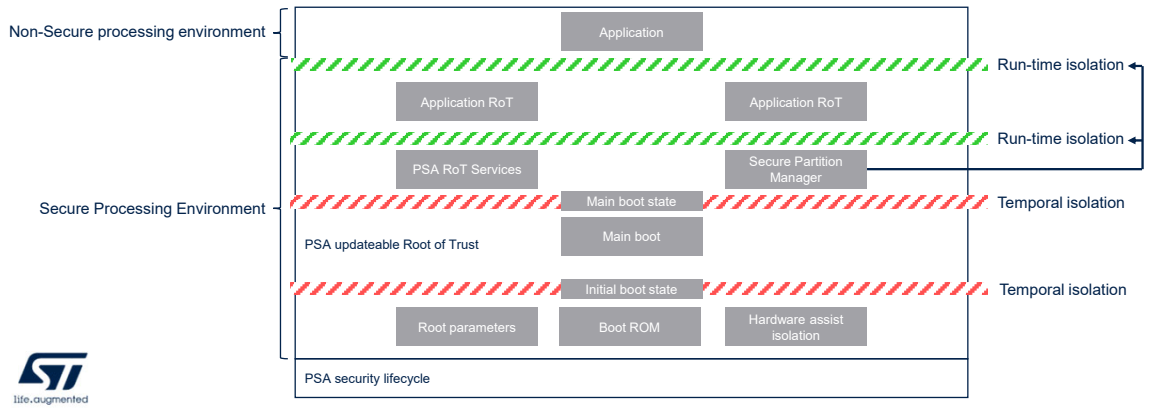
36

The hardware PRODUCT_STATE mechanism automatically controls the accesses to secrets provisioned in the device.
In Open state, there is no special protections.
In Provisioning and iROT-Provisioned state, all areas protected with HDPL1 cannot be dumped, debugged or traced. The iROT can setup a higher level of protection to prevent access from uROT to its own resources.
In TZ-Closed state, all peripherals and memories mapped as secure cannot be dumped, debugged or traced.
In Closed state, no debug is possible except through debug authentication.
In Locked state, all data and code stored in the device or encrypted in external flash memory cannot be dumped clear-text, debugged or traced.
Consequently, access to particular resource varies over time:

this is the temporal isolation.

STM32H5 Temporal Isolation

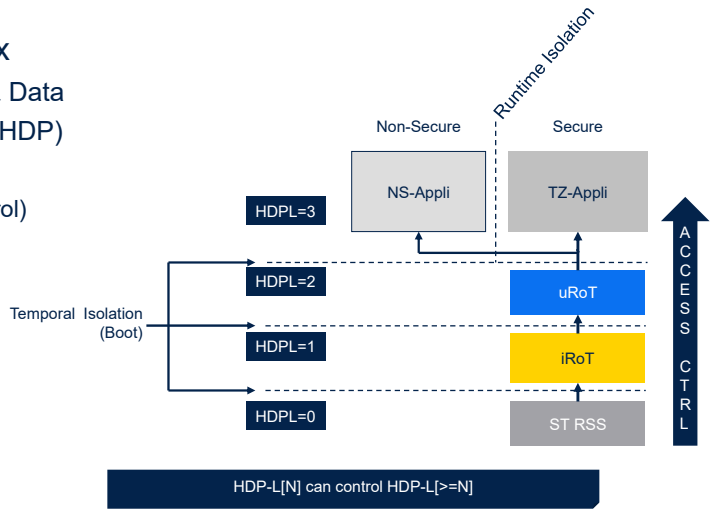
- STM32 isolation follows ARM PSA security model specification
  - Temporal isolation for boot stages
  - TrustZone isolation for runtime time services isolation



Temporal isolation protects sensitive device assets that are used during the secure boot process, from access by later stages.
Run-time isolation is based on STMicroelectronics solution around HDPL.
In the Arm's PSA security model, temporal isolation is used for boot stages, while run-time isolation applies to components that execute following secure boot and initialization of the SPE Partition Management function.

STM32H5 Temporal Isolation

- Temporal Isolation Levels: HDPLx
  - To manage access control on Code & Data
  - Code protected with Hide protection (HDP)
  - Data:
    - Flash OB-Keys (Physical Access Control)
      - 5 secure storage areas
        - HDPL0 ➔ST (never erased)
        - HDPL1 ➔iROT (ST-iROT or OEM-iROT)
        - HDPL2 ➔uROT
        - HDPL3 + Secure ➔Trust Zone
        - HDPL3 + NS ➔Non secure appli
    - Data can be wrapped with DHUK
      - Based on HUK + Version counter
      - Different for each HDPLx



This figure explains the hierarchy of access.
The various programs involved in boot-time and run-time are presented on the right:

- The ST Root Secure Service (RSS)
- The ST Immutable Root Of Trust (iROT)
- The ST Updatable Root Of Trust (ST-uROT) or proprietary one (uROT)
- The Secure and non-secure applications.

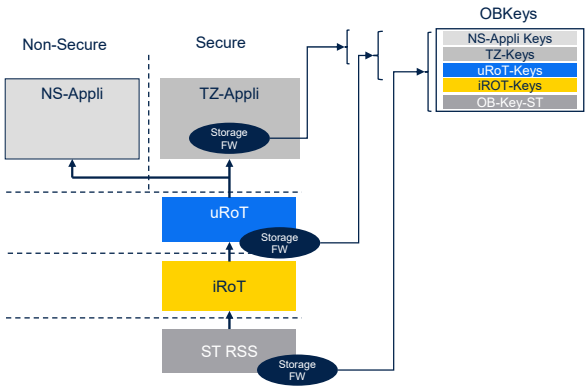
Each of these programs is assigned a HDPL.
Access to the hide protection area can be denied by progressing the HDPL level in the System configuration, boot and security (SBS).
For example, the part of the flash containing the RSS becomes hidden when the RSS transitions the HDPL from 0 to 1.

The HDPL level can be only cleared by a system reset, there is no means to deactivate the area protected by HDP.
The protected HDP area is defined by setting its size using start and end sectors in a similar way as the secure watermark.
The current HDPL also determines the access control rules and keys used for encryption / decryption.
Non-volatile data and code of a particular HDPL can be encrypted with the Derived Hardware Unique Key (DHUK), that includes the monotonic counter, making sensitive resources undecipherable from higher HDPLs.

The HDPL selects the secure storage domain (OBK-HDPL) accessible from current HDPL, or greater ones.

STM32H5 Secure Data Storage

- HW Secure Data Storage / Use cases
  - For application (runtime isolation)
    - Data Storage firmware
      - Isolated in secure privileged
      - Provide services to secure non-privileged
      - Provide services to non-secure
  - Key provisioning
    - During OEM product manufacturing (Provisioning state)
      - SFI or RSS-e when manufacturing not trusted
      - Using RSS-Lib when manufacturing is trusted
    - Key Provisioning using application services
      - Based on SKP Secure Key Provisioning services embedded in FW
        - Embedded in uROT, TZ-application



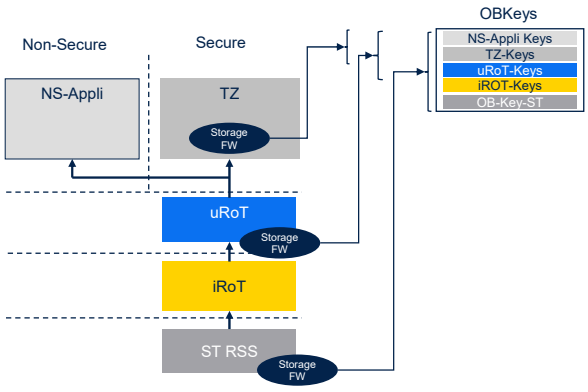
life.augmented

40

Hardware secure storage control improves the security, by isolating programs running at different privilege and security levels.
It also protects OBKeys against accesses and utilization from lower level HDPLs.
Finally, hardware secure storage enables secure provisioning of firmware, by different OEMs.
Note that even if we are talking about Key storage, any kind of data can be stored.

STM32H5 Secure Data Storage

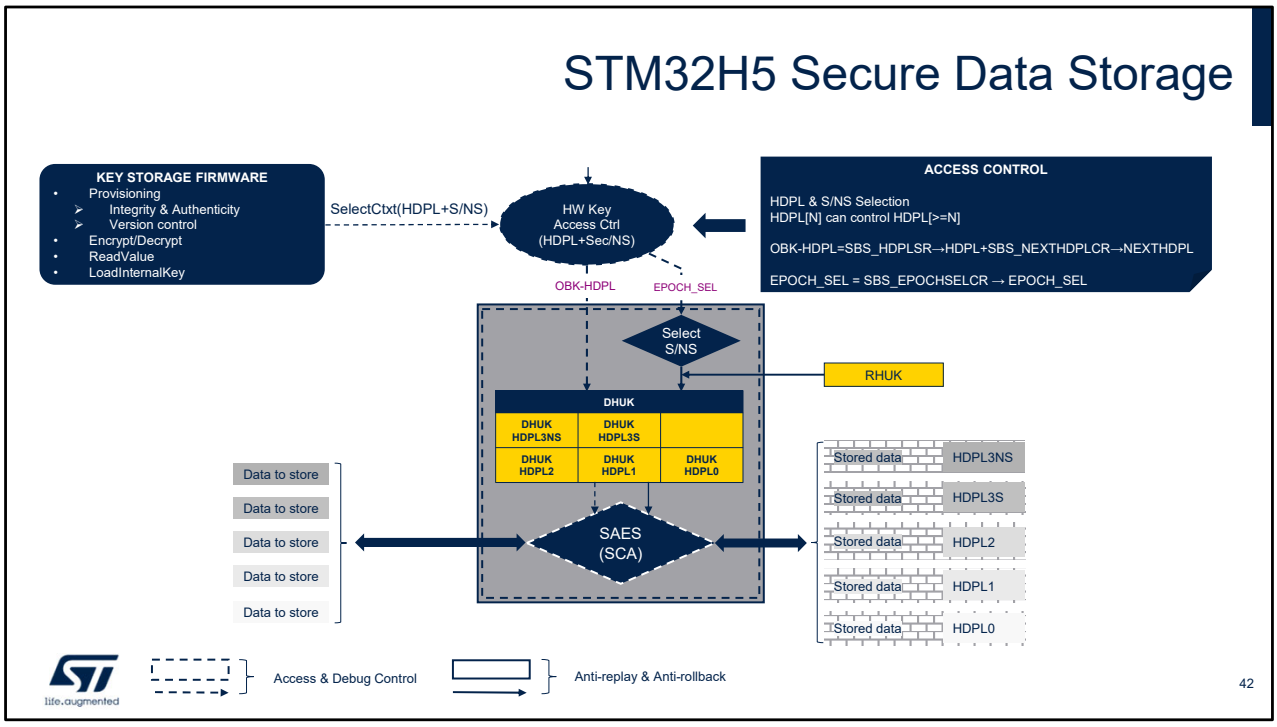
- HW Secure Data Storage / Features
  - Three levels of protection
    - TrustZone, Access Ctrl, Crypto isolation
  - In Flash Memory: OBKey area
    - Capacity 8 KB (swapped)
    - Splitted in 5 domains:
      - HDPL0, 1,2,3S, 3NS
      - HW Access Control
  - Stored Encrypted
    - The OBK keys can be stored encrypted based on SAES
      - Per domain encryption ➔Cryptography isolation
      - Anti-rollback ➔thanks to EPOCH counter of regressions
      - Anti-cloning ➔Thanks to RHUK (unique per device)



life.augmented

41

First of all, let's generalize that when we are talking about keys O.B.Keys, we are considering also Data like: Mapping, configuration, status, etc
At the receipt of an OBK access (read/write/execute), secure and privilege attributes are first checked, followed by OBK-HDPL.
The OBK-HDPL value must exactly match the HDPL assigned to the key location, otherwise an error is raised.
OBK storage is split into 5 domains, one per HDPL from 0 to 2 and two for HDPL3, one for secure keys, one for non secure keys.
The OBK keys can be stored encrypted based on SAES, with per-domain encryption.
The encryption mechanism also ensures anti-rollback and anti-cloning.



This figure represents the data protection performed by the flash security mechanisms, based on SAES and DHUK.
To distinguish two types of protection mechanisms, the following convention is used:

- Dotted lines identify mechanisms related to access and debug control
- Solid lines identify mechanisms related to anti-replay and anti-rollback.

The key storage firmware selects the current context composed of HDPL and Secure / Non-secure state, which is passed to the hardware key access control unit.
This unit selects the OBK-HDPL and EPOCH according to this current context and SBS register settings.
OBK-HDPL selects the area of secure storage of the Flash memory and selects the corresponding DHUK in SAES.

The EPOCH selector has to be set by the application wishing to use the secure storage.
sbs_epoch_s and sbs_epoch_ns are 24-bit values coming from the Flash memory and representing regression counters respectively for secure and non-secure states.
Then, data to be stored to flash memory will first be checked against accessible domains and if no violation is detected, are encrypted according to the selected DHUK

Thank you

life.augmented

Thanks for attending this presentation.
You can also refer to the following complementary presentations:

- Secure data storage
- Product Lifecycle
- Debug Authentication.