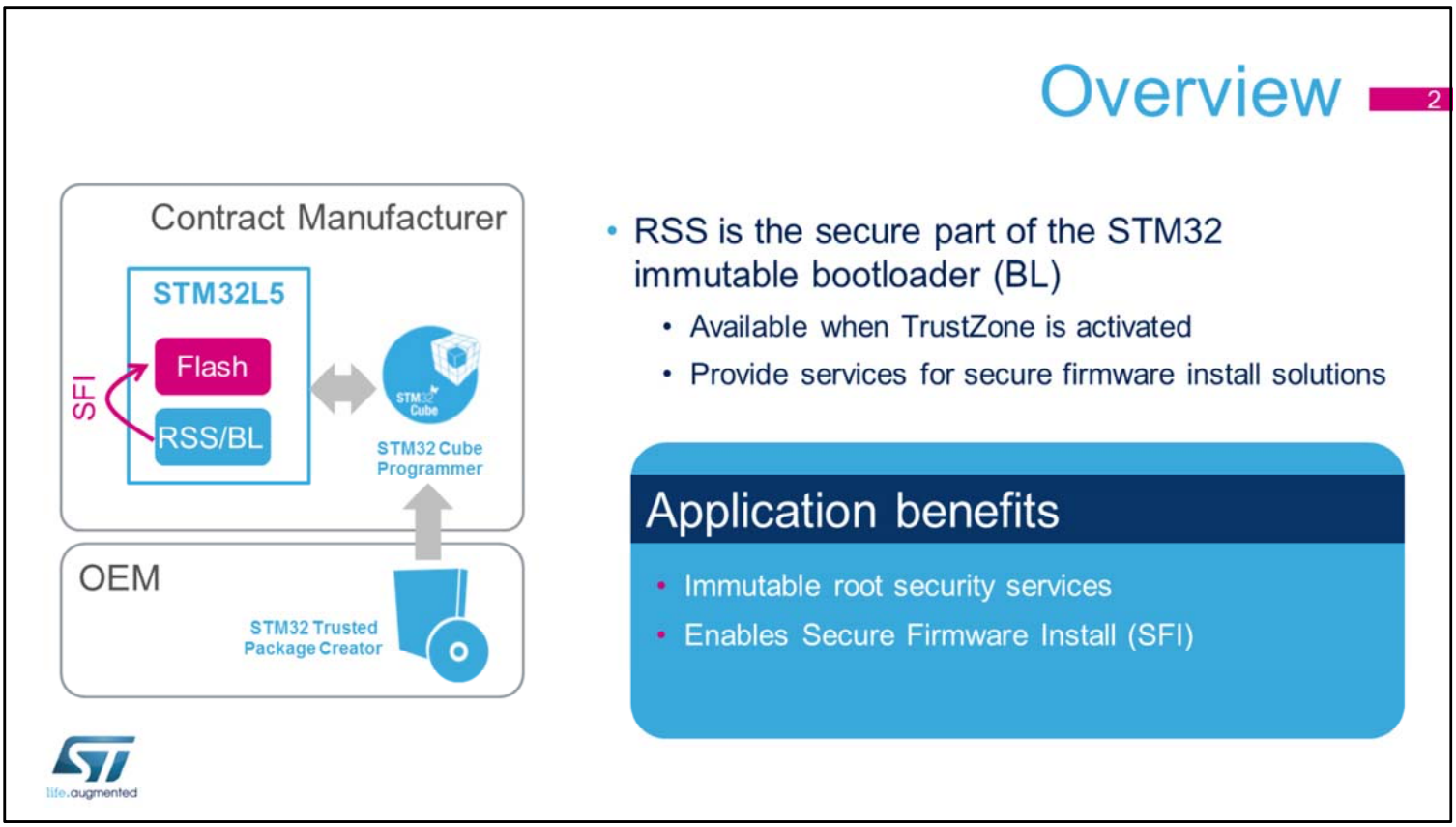
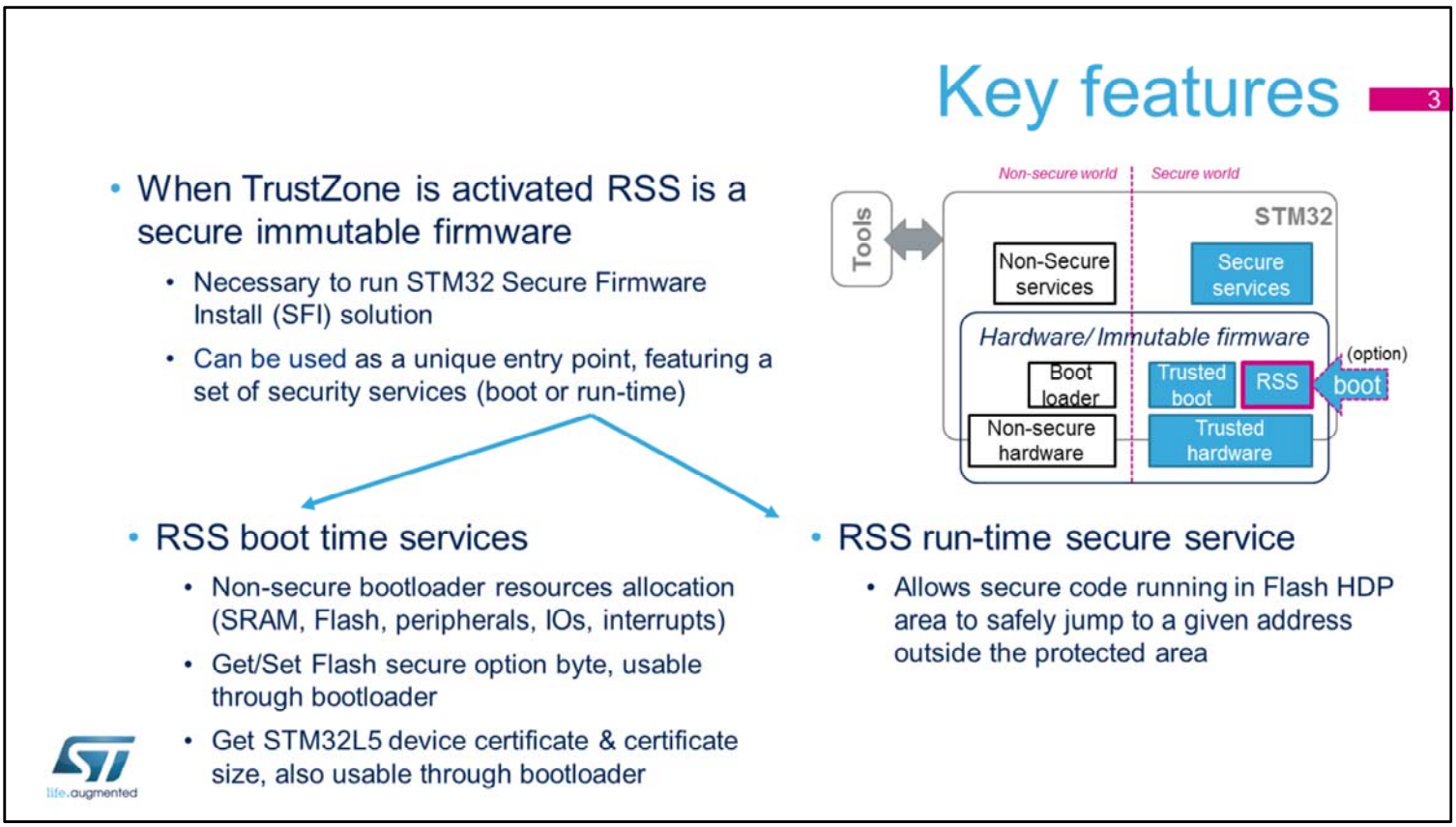
Hello and welcome to this on-line training module dedicated to the advanced security features of the STM32L5: the Root Security Services (RSS).
It is strongly advised to have already viewed the on-line training module “Memory protections”.



The Root Security Services (RSS) is the secure part of the STM32L5 immutable bootloader. It is available only when TrustZone is activated on the device.
RSS provides immutable root security services, used for example to run STM32 secure firmware install (SFI) solution in a untrusted environment.

For more information on TrustZone and protected memories please refer to the on-line training module “STM32L5 Memory Protection features”.



When TrustZone is activated RSS is the secure immutable firmware provisioned by ST during STM32L5 production (along with a dedicated device unique key pair).
After reset this immutable firmware is used as an a unique entry point, featuring a set of security services that are available at boot time, and sometimes also at run-time. RSS includes the necessary features to run STM32 Secure Firmware Install (SFI) solution (see following slides).

Boot time services of the RSS include:

- Non-secure bootloader resources allocation (SRAM, Flash, peripherals, IOs, interrupts)
- Functions to get or set Flash secure option bytes, usable through bootloader
- Function to get STM32L5 device certificate and certificate size, also usable through bootloader

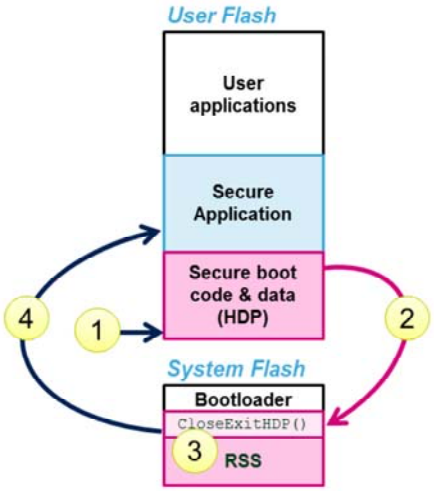
RSS also provides services for both secure and non-secure running firmwares. One of those secure service allows

secure firmware running in Flash HDP area to safely jump to a given address outside the protected area.

All those services are detailed in the next three slides.

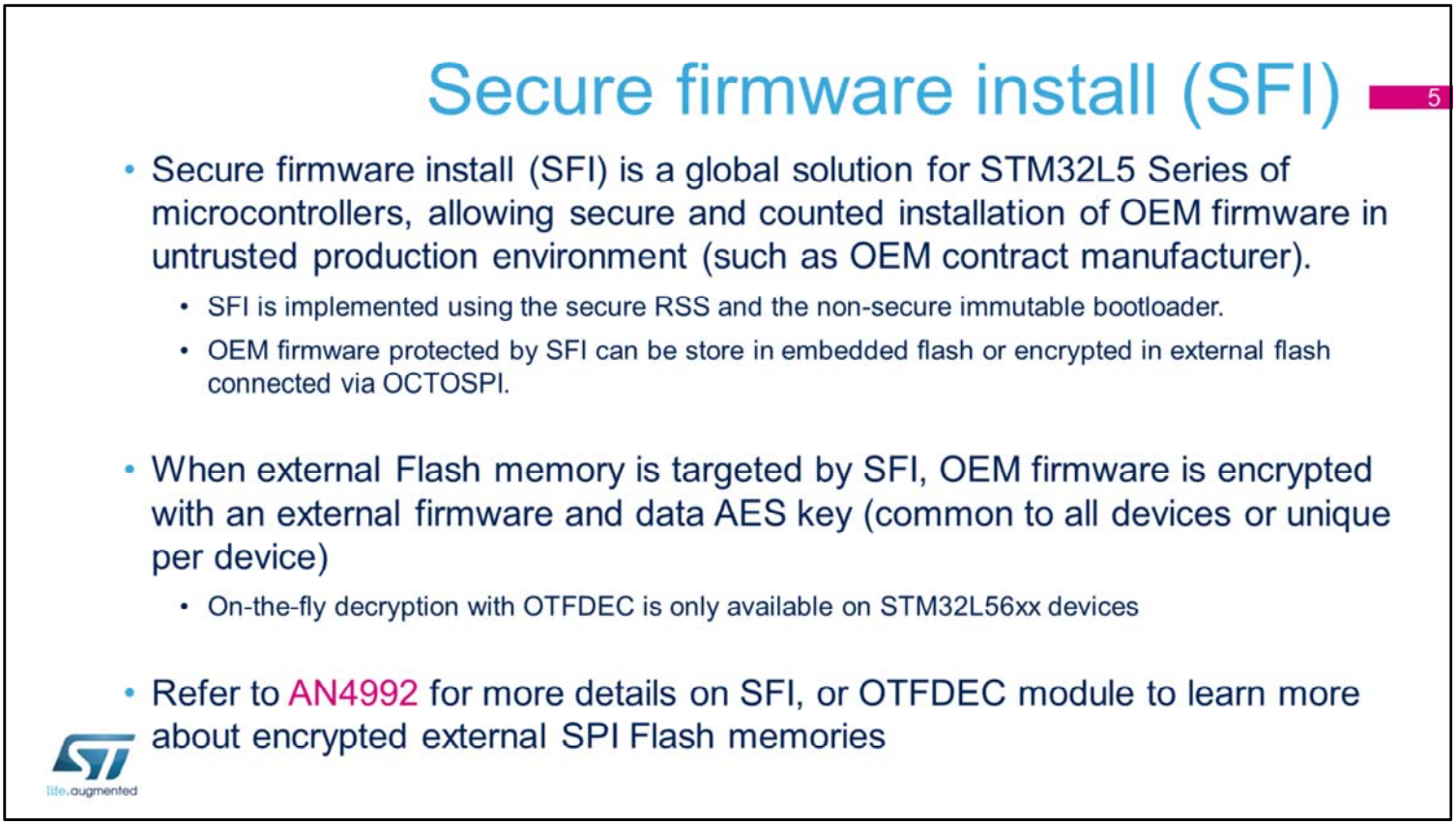
Jumping outside the HDP area

- HDP area is activated by secure code, setting the HDPEN option bit
- Typical sequence to jump outside of it:
  1. Device boots, executing sensitive code in HDP area
  2. Boot code calls HDP exit function in RSS lib
  3. RSS hides secure HDP area until next reset, then branches to (secure) application code
  4. Secure firmware can no longer access the HDP area
- Refer to RM0438 for more details on HDP close & exit function



Secure Hide protection (HDP) is an additional protection mechanism within the TrustZone secure domain. The code embedded in HDP is executed first, and at the end of its execution, it jumps to secure user application. The code and data protected by HDP can no longer be accessed until the next system reset. HDP area is activated by secure code, setting the HDPEN option bit.

How RSS helps jumping outside the HDP area? The boot code embedded in HDP executes after reset. At the end of its execution it calls RSSLIB_sec_CloseExitHDP to close Flash HDP secure memory area and to jump to the reset handler embedded within the vector table which address is passed as input parameter. This function resets STM32L5 in case of failure (e.g. bad input parameter value).



Secure firmware install (SFI) is a global solution for STM32L5 Series of microcontrollers, allowing secure and counted installation of OEM firmware in untrusted production environment (such as OEM contract manufacturer).
SFI is implemented using the secure RSS and the non-secure immutable bootloader. OEM firmware protected by SFI can be store in the device's embedded flash or encrypted in external flash connected via OCTOSPI.

STM32L5 SFI solution consists in having the whole OEM firmware and the option bytes encrypted with an AES secret key, thanks to STM32 Trusted Package Creator tool. This is done during OEM firmware development. Confidentiality of this AES secret key is ensured using an STM32 device unique key pair, with the private key readable only by the RSS.

When external Flash memory connected via OCTOSPI is

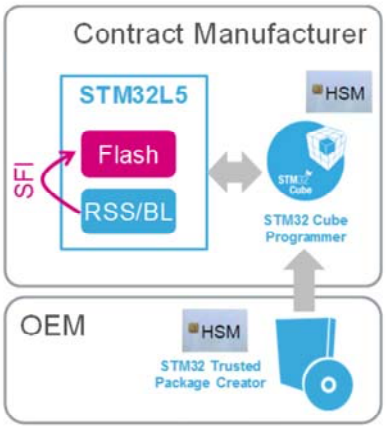
targeted by SFI, OEM firmware code must be encrypted with an external firmware and data AES key. This key can be:

- Common to all devices (in this case tools could perform the encryption), or
- Unique per device (in this case firmware is encrypted inside the device)

NB: On-the-fly decryption of encrypted firmware stored in SPI flash memories is only available on STM32L56xx devices.

For more information please refer to application note AN4992 for secure firmware install (SFI) solutions, or the STM32L5 OTFDEC training module for encrypting and decrypting external Flash firmware.

SFI security features



- Only genuine STMicroelectronics STM32 microcontrollers can install the protected firmware via SFI. It can be done in a non-trusted environment/facility.
- The number of STM32 devices on which the firmware have been installed can be counted inside the HSM
- Authenticity, integrity and confidentiality of the OEM internal firmware (and option bytes) are checked before embedded Flash is programmed with decrypted firmware (and option bytes).
- SFI can re-encrypt OEM external firmware using AES key(s) dedicated to OTFDEC peripheral. Those keys can be globally managed (by the tools), or they can be device specific (e.g. locally computed using the true RNG peripheral).

life.augmented

Only genuine STMicroelectronics STM32L5 microcontrollers can install the protected firmware via SFI. It can be done in a non-trusted environment/facility.
The number of STM32 devices on which the firmware have been installed can be counted inside the hardware security module (HSM) associated with the SFI process (see next slide).

OEM firmware and the option bytes are encrypted thanks to STM32TrustedPackageCreator tool, during OEM firmware development. OEM also uses this tool to program the Hardware security module (HSM) with its AES secret key, its nonce, and a maximum installation counter. OEM contract manufacturer uses STM32CubeProgrammer + provisioned HSM to initiate SFI process and send encrypted SFI image to the STM32L5 device.

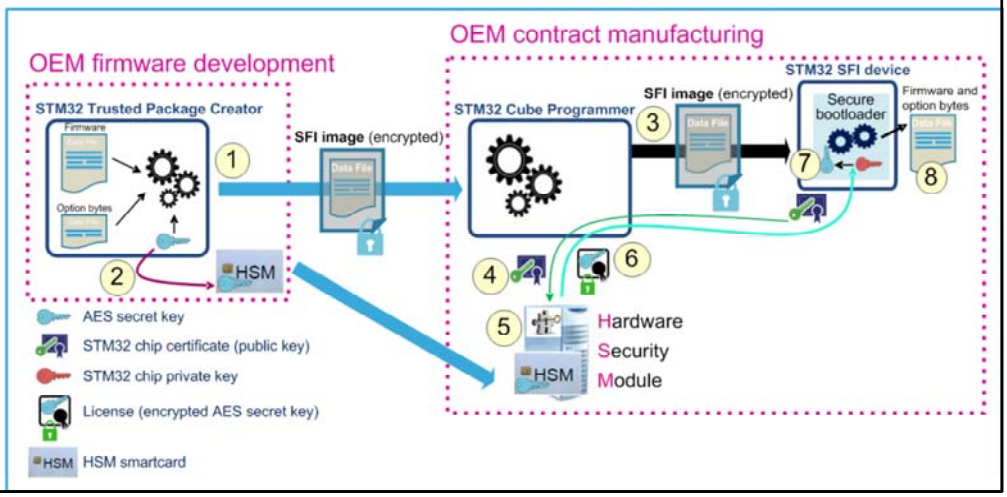
Authenticity, integrity and confidentiality of the OEM internal

firmware (and option bytes) are checked before embedded Flash is programmed with decrypted firmware (and option bytes).

When external Flash memory is targeted by SFI process RSS can re-encrypt OEM external firmware using an AES key dedicated to the OTFDEC peripheral. Those OTFDEC keys can be globally managed by the tools, or they can be device specific (e.g. locally computed using the true RNG peripheral). A mix of both is also possible.

SFI to internal flash

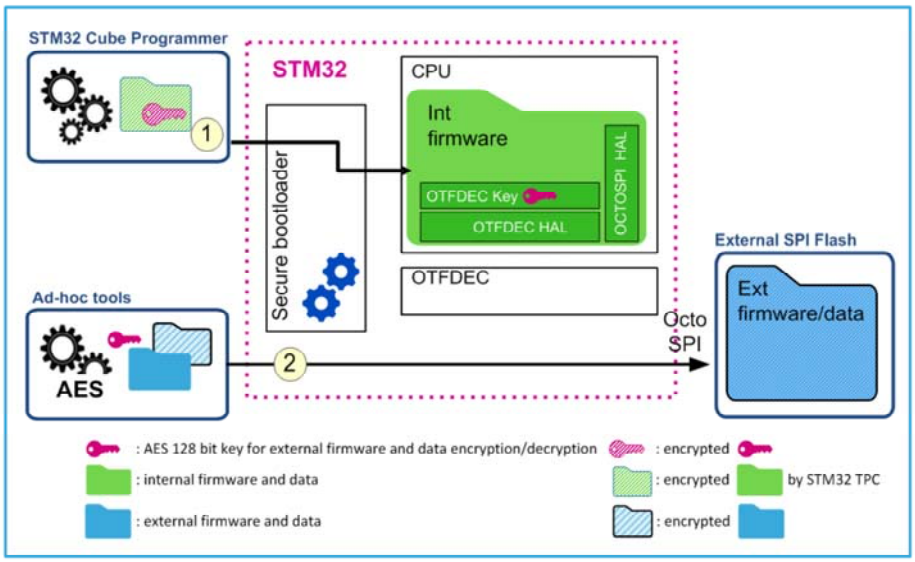
1. SFI image (encrypted) available from *STM32 Trusted Package Creator*
2. OEM programs HSM with AES secret key
3. SFI process launch
4. Device certificate retrieval
5. STM32 device authentication in HSM
6. HSM provides license to STM32
7. RSS retrieves OEM AES secret key encrypted in license
8. Encrypted firmware and option bytes decryption then programming



Secure firmware install to internal Flash memory goes as follow (numerical steps are represented on the schematic):
(1) SFI image (encrypted) available from STM32 Trusted Package Creator
(2) OEM programs HSM with AES secret key
(3) SFI process launch
(4) Device certificate retrieval
(5) STM32 device authentication in HSM
(6) HSM provides the license to STM32 device
(7) RSS retrieves OEM AES secret key encrypted in the license
(8) Encrypted firmware and option bytes are transferred, decrypted then programmed

SFI to internal & external flash (1)

1. Secure programming of internal Flash memory, using SFI
2. Encryption then programming of external firmware and data



NB: OTFDEC peripheral is only available in STM32L56xx devices

The cryptographic engine responsible for the on-the-fly external Flash memory decryption (OTFDEC) supports AES standard cryptographic algorithm. Thanks to this standard algorithm, OEM can encrypt external firmware and data on host before programming to the external Flash memory, without using STM32 secure bootloader.

This slide shows that the secure programming of internal Flash memory (1) and the encryption plus programming of external firmware and data (2) could be done in two separated flows. The first flow uses secure bootloader, while the second uses the OEM host for programming the external Flash memory.

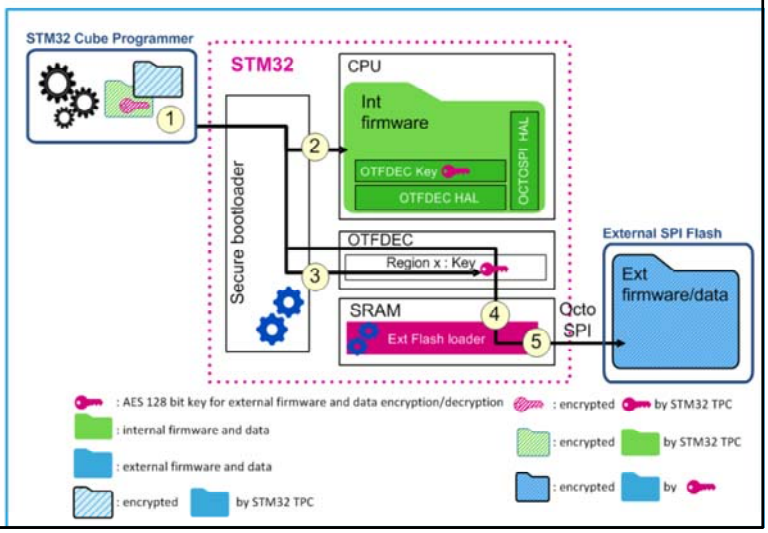
Afterward, during each secure boot, the secure internal firmware first copies the AES firmware and data key(s) in write-only OTFDEC key registers, then activates the OTFDEC region tied to those keys. At this point the CPU can

seamlessly read/fetch data/code from external Flash memory once the OCTOSPI driver has been initialized.

SFI to internal & external flash (2)

1. Create an SFI image, with:
   - Internal firmware and data (including external Flash memory drivers)
   - External firmware and data *AES key*
   - External firmware and data
2. Internal Flash memory programming
3. External firmware and data *AES key* programming in OTFDEC peripheral
   - Alternatively such key(s) can be managed locally to the device, not globally in the flashing tools.
4. External Flash memory chunk encryption
5. External Flash memory programming by the user's firmware

*NB: OTFDEC peripheral is only available in STM32L5**6**xx devices*



This slide represents the sequence where the STM32 secure bootloader handles both internal firmware installation and external firmware installation with a global external Flash memory AES key and the help of an external Flash memory loader. The numerical steps are represented on the schematic.

(1) Create an SFI image, with a) internal firmware and data (including external Flash memory drivers), b) external firmware and data AES key, and c) external firmware and data

(2) Internal Flash memory programming, as described in slide before.

(3) External firmware and data AES key programming in OTFDEC peripheral. Alternatively to what is drawn on the slide this key can be managed locally to the device, not globally in the flashing tools.

(4) External Flash memory chunk encryption

(5) External Flash memory programming by the user's

firmware

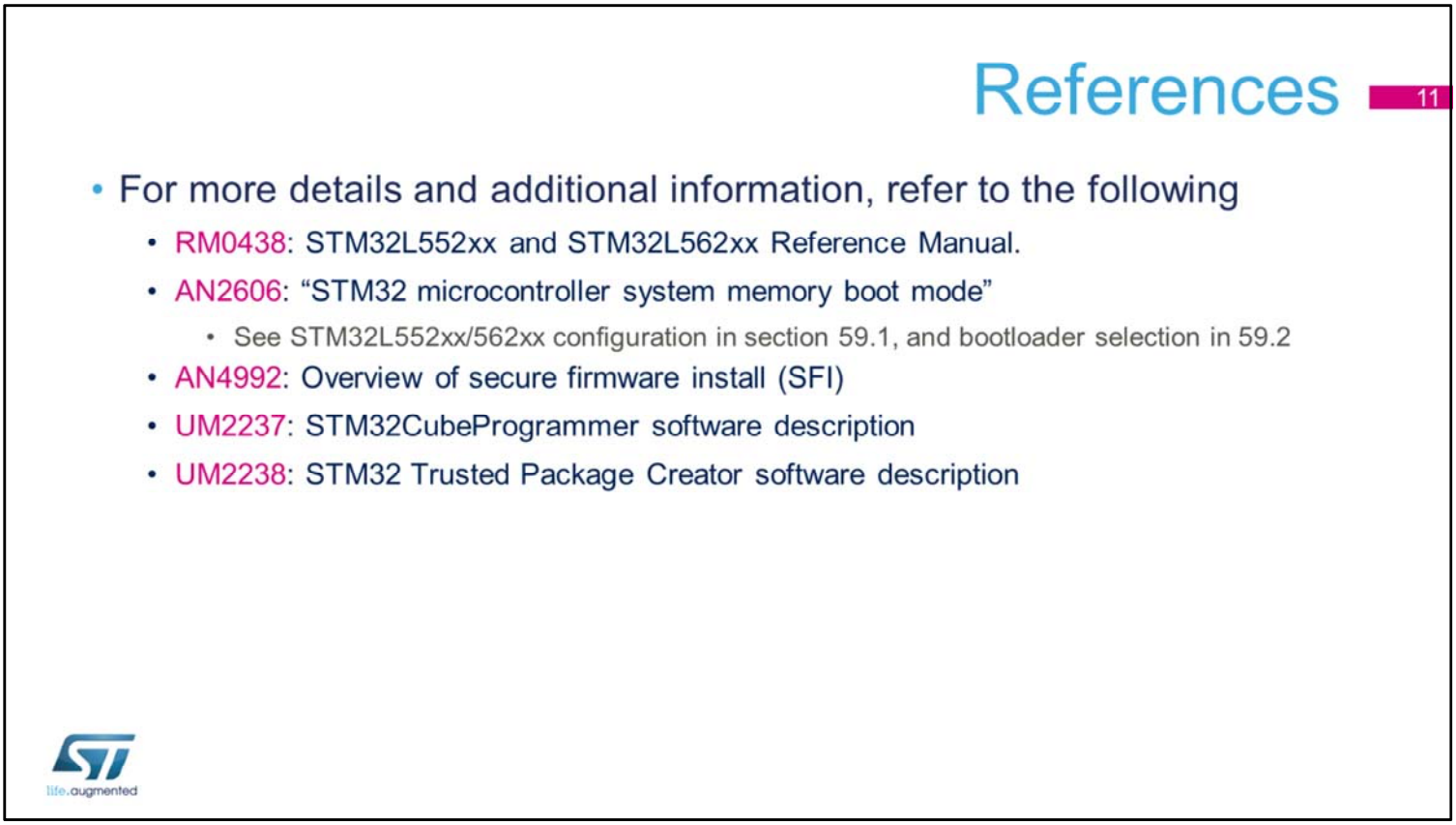
Afterward, during each secure boot, the secure internal firmware first copies the AES firmware and data key(s) in write-only OTFDEC key registers, then activates the OTFDEC region tied to those keys. At this point the CPU can seamlessly read/fetch data/code from external Flash memory once the OCTOSPI driver has been initialized.

Related peripherals and trainings

- Refer to these trainings linked to RSS
  - STM32L5-Security-Memories Protection
    - Protecting code and/or data from external and/or internal attacks
  - STM32L5-System-Boot
    - Booting the device
  - STM32L5-Security-TrustZone
    - STM32L5 implementation of Arm TrustZone ® technology
  - STM32L5-Memory-Flash
    - Embedded Flash memory features
- Refer to those peripherals trainings linked to RSS and SFI
  - Global TrustZone Controller (GTZC)
  - OctoSPI interface (OCTOSPI)
  - On-the-fly Decryption Engine (OTFDEC)
  - Random Number Generator (RNG)
    - For optional external firmware and data AES key generations

life.augmented

Please refer to MemProtect, Flash, Boot or TrustZone training if you want to know more on those topics.
Also find a list of peripherals related to the RSS and the SFI.
Please refer to GTZC, RNG, OCTOSPI or OTFDEC trainings for more information if needed.



For more details, please refer to:

- Application note AN2606 about STM32 microcontroller system memory boot mode. See in particular STM32L552xx/562xx configuration in section 59.1, and bootloader selection in section 59.2.
- Application note AN4992 about Overview of secure firmware install (SFI)
- User manuals for STM32CubeProgrammer and STM32 Trusted Package Creator are also available on the ST website.