Security with VisionSDK on TDA2x-HS and TDA3x-GP Prime devices

Automotive Processor Business Unit

ABSTRACT

This document summarizes the modifications needed in VisionSDK and Starterware based bootloader to run VisionSDK framework on TDA2x/TDA2Ex HS devices and TDA3x GP Prime devices.

Contents

Figures

Tables

Abbreviations

SBL: Secondary Boot Loader.

PPA: Primary Protected Application.

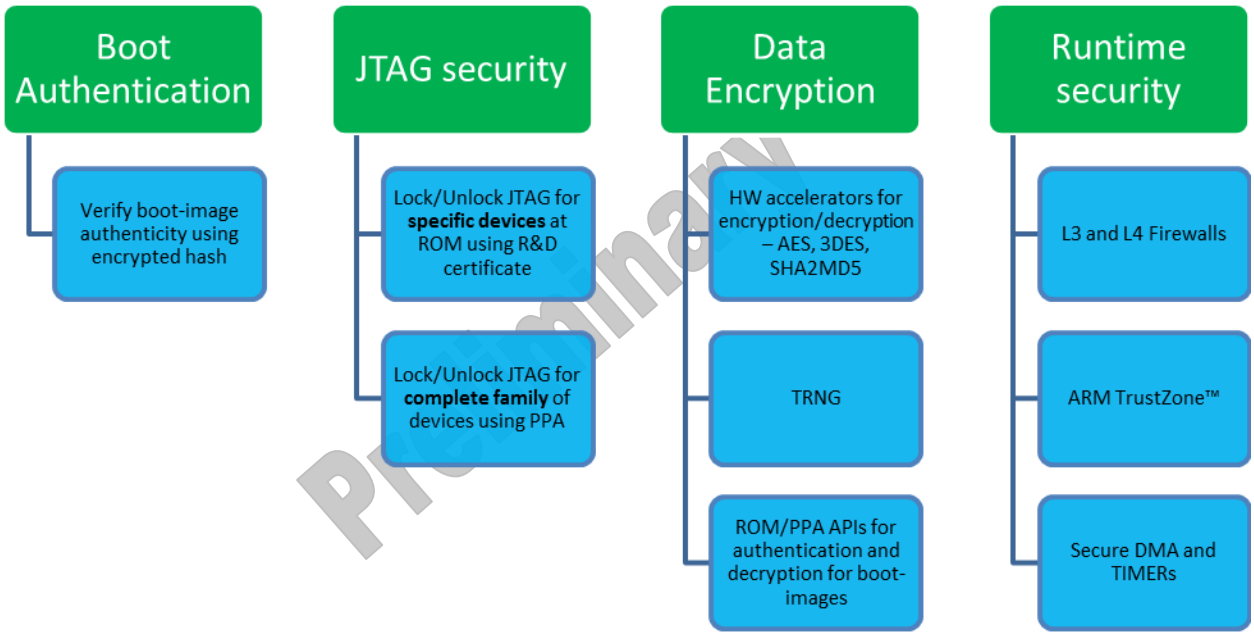
ISW: Initial Software

1 Introduction

1.1 TDA2x and TDA2Ex

TDA2x/TDA2Ex family provides high-security (HS) variants. The following figure gives an overview of the various security features available on these devices.

Figure 1. Security features on HS devices



Security features on TDA2x/TDA2Ex are identical. All further documentation is applicable for both TDA2x and TDA2Ex families unless explicitly stated otherwise. Security Addendums for the TRMs of these devices are available on CDDS. These documents are available under a special NDA.

To enable VisionSDK on HS devices, users need to install an additional package - MSHIELD-DK. This document is intended only as a userguide for using VisionSDK with MSHIELD-DK. Details on MSHIELD-DK are limited only to a few relevant features used in context of VisionSDK.

MSHIELD-DK primarily provides

1. Reference PPA source code, build system and relevant documentation
2. Image signing tools to sign SBL and AppImages.

For further details on MSHIELD-DK, users are encouraged to refer to the documentation available released as part of the MSHIELD-DK package.

VisionSDK v2.11 and higher requires MSHIELD-DK version 4.5.3 GA available on CDDS under a special NDA. Contact your TI FAE for further details.

The MSHIELD-DK package is intended for a wide variety of TI SoC families like DRA7x/TDA2x/AM57x and will contain references to all these SoCs. MShield-DK package will be renamed to SECDEV in later releases.

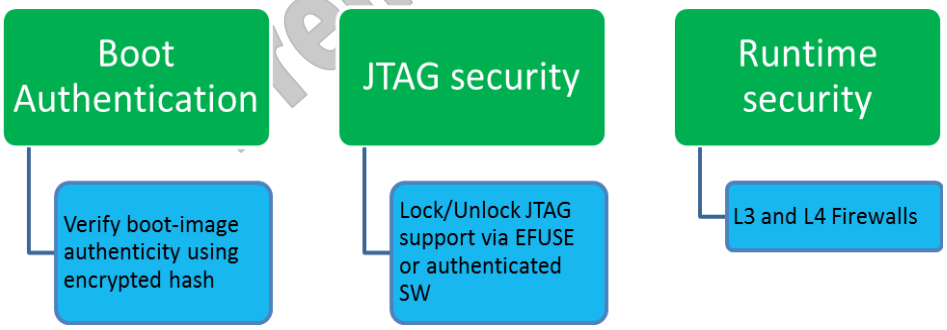
In this document, we shall discuss following topics:

- Boot flow on TDA2x/TDA2Ex HS devices
- Build flow updates for Starterware Bootloader and MLO/X-Loader generation scripts to enable boot authentication
- Updates to AppImage generation scripts to enable extended authentication
- Setting up MSHIELD-DK for TDA2x/TDA2Ex to create PPA and signed images for bootloader and AppImage

1.2 TDA3x

TDA3x provides only partial security features as compared to TDA2x high-security (HS) variants. These are indicated in the figure below. These TDA3x variants are called GP-Prime.

Figure 2. Security features on TDA3x GP Prime devices



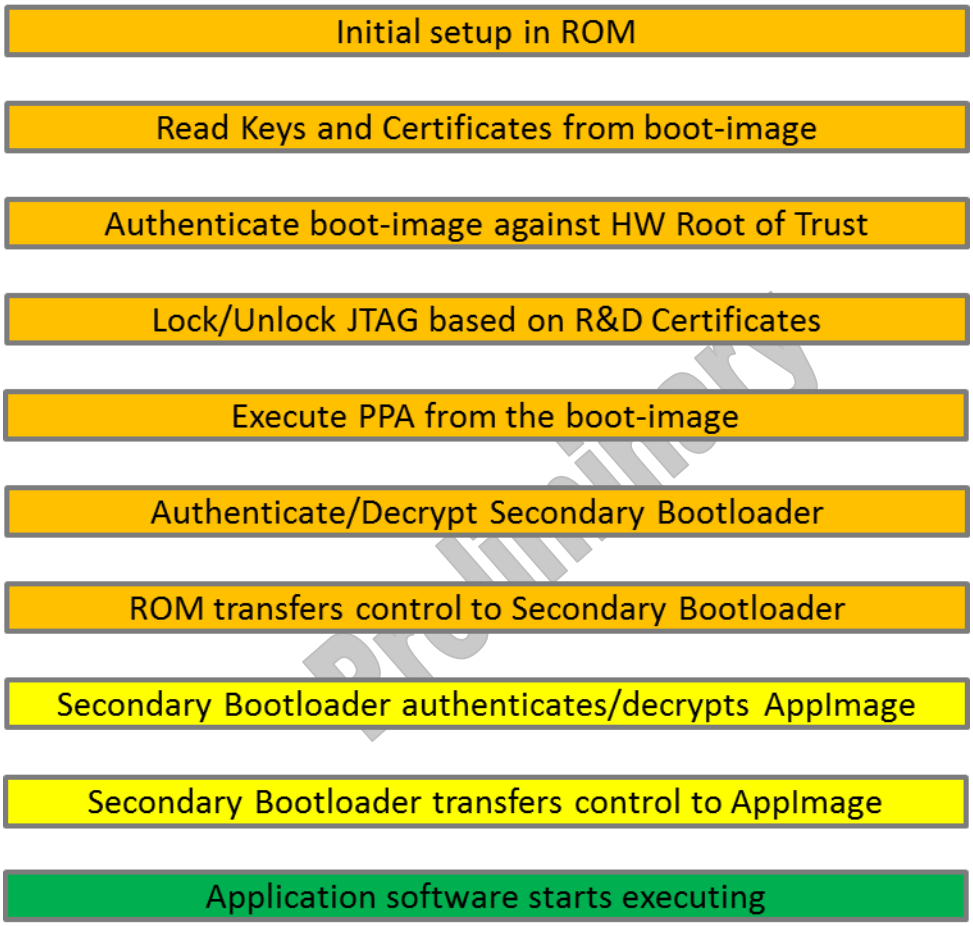
To enable VisionSDK on GP Prime devices, users need to install an additional package – **Starterware Security Add-on** available with all VisionSDK 2.11 and higher. This document is intended only as a userguide for using VisionSDK with this add-on pacakge. Refer to the documentation within the add-on package for more details.

This package is available only under a special NDA. Contact your FAE for more details.

2 Boot flow

2.1 Boot flow of TDA2x HS devices

Figure 3. Boot flow in HS devices



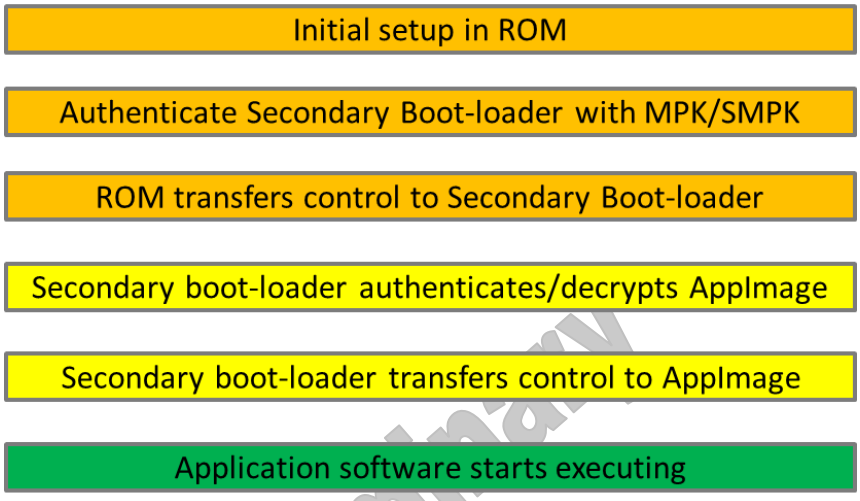
The above figure gives a brief overview of boot-flow in HS devices. The Orange colored sections indicate ROM execution. The Yellow colored indicate execution of Secondary Boot Loader (typically called as SBL or MLO or X-LOADER). MShield-DK documentation refers to SBL as ISW (Initial software). The Green colored indicate execution of final application in the system.

Boot-image in this figure means a signed image consisting of Keys Certificates, PPA and ISW Certificates, PPA and ISW binary images. In this document, we shall not discuss the structure of boot-image, but only indicate how to use MShield-DK/SECDEV package to generate this boot-image.

PPA refers to Primary Protected Application. This code is copied from boot-media to Secure RAM in A15. This application is typically intended for Secure environment initialization, patching Secure ROM code, and provide user-defined Secure services. The MShield-DK/SECDEV package provides a reference PPA code which implements features like boot-decryption, optional JTAG unlock, and provides services like decryption and firewall configurations.

2.2 *Boot flow of TDA3x GP Prime devices*

Figure 4. Boot flow in HS devices



The above figure gives a brief overview of boot-flow in GP Prime devices. The Orange colored sections indicate ROM execution. The Yellow colored indicate execution of Secondary Boot Loader (typically called as SBL or MLO or X-LOADER). In case of TDA3x, ROM authenticates the SBL against MPK or SMPK.

TDA3x does not provide HW acceleration for authentication/decryption. The SBL needs to implement these features in software to authenticate further software loaded onto the system.

The SBL can optionally decide to open up JTAG access to the system if needed.

Refer to TDA3x GP Prime documentation for further details on GP Prime device features.

3 Differences in SoC behavior

3.1 TDA2x/TDA2Ex

In case of HS devices, the reset behavior of TDA2x/TDA2Ex differs in few ways

- JTAG access
  - In case of secure devices, JTAG access is blocked by default. This is the intended setting for end-applications as well.
  - However, during development/maintenance flow, it might be necessary to enable JTAG access for debugging software.
  - JTAG access can be enabled via PPA or by using appropriate R&D Certificates
  - Enabling JTAG access via PPA can open up JTAG access for any devices
  - R&D Certificates is a more restrictive and, therefore, more secure option to unlock JTAG on one specific sample. This flow is currently not included in this document, but will be available in future releases.
- L3/L4 Firewalls
  - The default PPA opens up access to all non-secure L3/L4 peripherals to all masters by default.
  - However, access to L3/L4 firewall configuration registers are restricted only to secure software. Secure software can execute only on A15 in secure mode.
  - Users may choose to open up any relevant firewalls using PPA or create PPA services to open/configure firewalls.
- OCMC1 RAM
  - In case of HS devices, the first 4kB of OCMC_RAM1 is reserved and not accessible to non-secure world.
  - Secure ROM enforces this by using region 0 and region 1 of OCMC1 firewall.
  - Any new PPA code or services included by users ***must*** ensure this constraint is strictly ensured to avoid creating security holes.
- Secondary Boot Location entry point
  - For GP devices, the 4kB restriction of OCMC_RAM1 is not present. Typically, SBL will use the start address (0x4030_0000) of OCMC_RAM1 as an entry point.
  - In case of HS devices, due to structure of signed boot-image and the 4kB restriction, the lowest address in OCMC_RAM1 that SBL can use is 0x4030_1350.

3.2 TDA3x

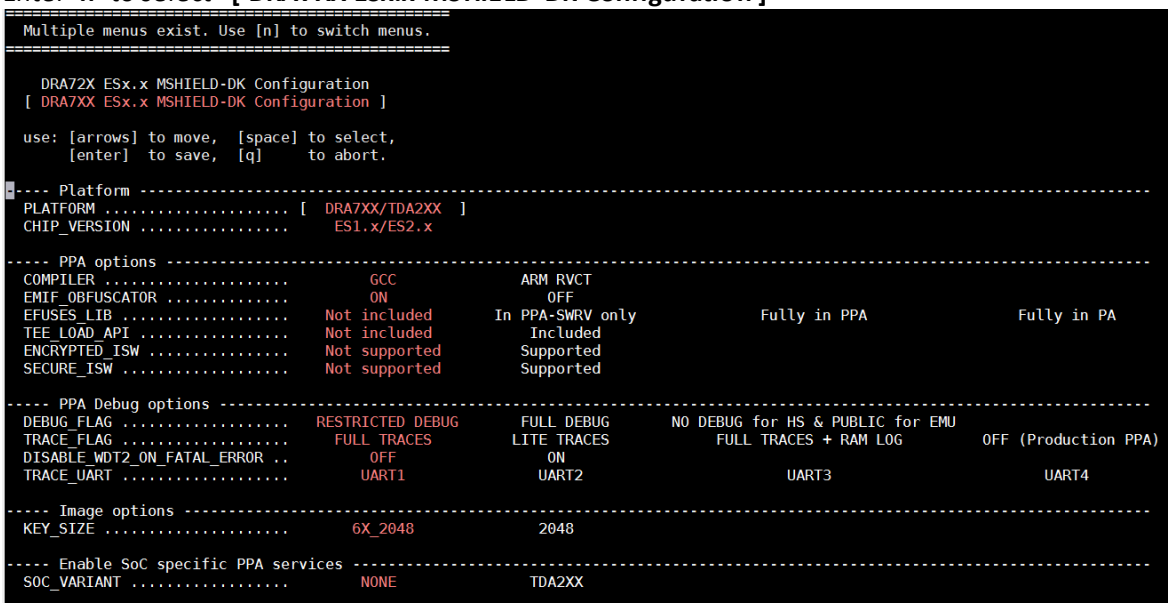
TDA3x GP Prime don't differ significantly from normal GP devices beyond the initial secure boot flow. Additionally, GP Prime devices allow JTAG access to be closed at boot-time based on EFUSE or at run-time using a control-module register.

4 Setting up MShield-DK/SECDEV for TDA2x/TDA2Ex

- **MShield-DK/SECDEV package is supported only on Linux/Ubuntu machines**. This applies to building PPA, signing boot-images and application images.
- Update **secdev_PATH** variable in **vision_sdk/tools_path.mk** to point to the MSHIELD-DK/SECDEV installation.
- GCC compiler installation details from MShield-DK/SECDEV documentation
  - If you need to re-generate the PPA or PA using GCC, these are the compilers tested
    DRA7xx/72x: GCC 4.8.2 arm-linux-gnueabihf-gcc cross compiler under Ubuntu 14.04 64-bit (arm-linux-gnueabihf-gcc (Ubuntu/Linaro 4.8.2-16ubuntu4) 4.8.2)
    DRA7xx/72x: GCC 4.8 arm-eabi-gcc bare-metal cross compiler from TI Android SDK (arm-eabi-gcc (GCC) 4.8)
  - The CROSS_COMPILE environment variable will be used, if it is defined, to prefix the gcc build calls. If it is not defined, the value 'arm-linux-gnueabihf-' will be used. The arm-linux-gnueabihf-gcc compiler can be installed under Ubuntu using the 'sudo apt-get gcc-arm-linux-gnueabihf' command.
- We will refer to the MShield-DK/SECDEV directory by the environment variable MSHIELD_DK_DIR.
- Setting up TDA2x configuration
  - Run the following commands:

```
cd $MSHIELD_DK_DIR/src
perl config.pl
```

  - Enter **'n'** to select **"[DRA7XX ESx.x MSHIELD-DK Configuration]"**



– Different configurations can be selected using “Arrow” keys to move and “Space” to select. Hit “Enter” to save the configuration.

Table 1. MShield-DK/SECDEV Options Summary

Option	Summary
PLATFORM	Only one option. Sets up base configuration for DRA7x and TDA2x platforms for PPA generation.
CHIP_VERSION	Only one option. Currently all SR versions are supported in same configuration.
COMPILER	Select GCC compiler. ARM RVCT support is deprecated and present only for legacy users.
EMIF_OBFUSCATOR	Set this to “ON”. This enables obfuscation of EMIF signals using a 16bit key to prevent snooping or error injection using the EMIF interface. This has no effect of view of EMIF within the SoC or when using debugger interface.
EFUSES_LIB	Not verified for TDA2x. Refer to MShield-DK documentation for details.
TEE_LOAD_API	Not verified for TDA2x. Refer to MShield-DK documentation for details.
ENCRYPTED_ISW	Will assume the boot-image includes SBL in encrypted form.
SECURE_ISW	Keep as “Not supported”. This transfers control to SBL in while keeping the A15 in secure-mode. Refer to MShield-DK documentation for relevant use-cases.
DEBUG_FLAG	“RESTRICTED DEBUG” prevents any debugger connection via the JTAG interface. “FULL DEBUG” allows users to connect to CCS for all software. “NO DEBUG for HS & PUBLIC for EMU” prevents JTAG access when SECURE software is executing, but allow JTAG access for rest of the software.
TRACE_LOG	Select trace level as needed.
DISABLE_WDT2_ON_FATAL_ERROR	Disable/Enable watchdog timer after a fatal error.
TRACE_UART	Select which UART interface to use for ROM/PPA traces. TI EVM supports only UART1 interface.
KEY_SIZE	Keep as 6X_2048. Refer to MShield-DK documentation for details.
SOC_VARIANT	**Select “TDA2XX” if you working with RTOS on A15**. This enables some PPA services to enable firewall usage from RTOS environment. **Caution:** If you are using HLOS like Linux on a TDA2x device, select SOC_VARIANT = “NONE”. This is because the PPA services introduced by “TDA2XX” option should only be used when all software is authenticated. This may not be guaranteed in an HLOS environment.

– If you enable ENCRYPTED_ISW, you need to modify **reference/dra7xx/config/KEY_6X_2048/isw_m.cfg**

- Uncomment the lines:

```
ISW_TYPE = "0x00000001"
ISW_ENC_IV = "000102030405060708090a0b0c0d0e0f"
ISW_ENC_KEY = "0828F4E5A4C5B20F119A631517C623CCA2752BBD639D150A12F26EC5923B3E86"
```

For TDA2x, only ISW_TYPE=1 is verified. This selects the key used for encryption ISW/SBL encryption. The key can be the device MEK, a key derived from the device MEK, the device CEK, or a key derived from the device CEK. Refer to MShield-DK documentation for details on these options. The default keys in this file are TI Dummy Keys.

5 Setting up Starterware Security Add-on for TDA3x

- Starterware Security Add-on package is supported only on Linux/Ubuntu machines. This applies to signing boot-images and application images.
  - Linux/Ubuntu installations need to install **python2.7** and **openssl** to use the signing scripts
  - Standard linux commands like xxd, sed, cat, dd are used.
  - /dev/urandom is used to generate random IV for optional encryption process.

6 Starterware updates for Secondary Bootloader (SBL) for TDA2x

6.1 Memory map

SBL should use OCMC_RAM1 from address 0x4030_1350 onwards. The entry point will be set to 0x4030_1350. The GP-Header in SBL which indicates entry point and size of SBL is not needed when generating the boot-image using MShield-DK.

- If entry point is provided as 0xFFFFFFFF to the **tiimage** utility, the GP Header will not be appended at the top of the MLO. This change is available only after VisionSDK 2.10.

6.2 Building SBL

make -s -j sbl_sd

SBL binary generated in starterware needs to be signed and merged with PPA using MShield-DK package. This is discussed in Chapter 8.

6.3 PPA service calls for TDA2X

- The MShield-DK/SECDEV package provides several services for application software to use the security features in the SoC. SBL uses three such services – two PPA services to enable "Freedom From Interference" using Firewalls and one ROM service to enable extended authentication (Authentication of AppImage). These are implemented in **starterware_XX/bootloader/sbl_lib/src/sbl_lib_tda2xx_hs.c**. These functions are intended to be executed only of A15 core in HS devices.

Table 2. PPA services available in SBL

SERVICE	SBL_LIB function name	Description
PPA_SERV_HAL_OPEN_FFI_FW	SBLLibHSOpenL3Firewalls	Enable access to EMIF, MA_MPU, OCMC2 and OCMC2 firewalls to be configured by all cores
PPA_SERV_HAL_SETUP_OCMC1_FW	SBLLibHSConfigureOcmc1Firewall	Configure OCMC1 firewall regions 2-7. Configuration of OCMC1 firewall is restricted to secure world to enforce no-access to first 4kB of OCMC1 which is reserved in case of HS devices.
API_HAL_KM_VERIFYCERTIFICATESIGNATURE_INDEX	SBLLibHSAuthenticateBinary	This uses a ROM API to authenticate an input binary data blob by checking if it is signed with a valid key.

- SBLLibHSOpenL3Firewalls() is executed as part of default SBL.
- SBLLibHSAuthenticateBinary() is used as part of default SBL to authenticate AppImage.
- Users may link "sbl_lib" library to their application to use SBLLibHSConfigureOcmc1Firewall() or SBLLibHSAuthenticateBinary() functions at run-time. **Note:** These functions are supported only on A15.

7 Starterware updates for Secondary Bootloader (SBL) for TDA3x

7.1 Boot Authentication

SBL binary generated in starterware needs to be signed using Starterware Security Add-on package. This is discussed in Chapter 9.

8 Boot Authentication and Extended Authentication on TDA2x

8.1 Boot Authentication

In case of HS devices, a boot-image consist of PPA and SBL "signed" with appropriate certificates to ensure that Secure ROM can authenticate it. This step is called as "Boot Authentication". In this chapter, we discuss steps to generate "signed" boot-images (referred to as MLO/XLOADER in this section) using unsigned images generated using the VisionSDK build flow.

- Setup MShield-DK/SECDEV for TDA2XX
  - ```
    cd $MSHIELD_DK_DIR/src
    perl config.pl
    ```
  - Refer to Chapter 3.2 for details
- Build PPA using the following command
  - ```
    cd $MSHIELD_DK_DIR/scripts
    ./ift-ppa-gen.sh dra7xx
    ```
- Create boot image using following commands:
  - SD-BOOT (Input file: **unsigned_MLO**, Output file: **MLO** – Both are in little-endian format)

    ```
    cd $MSHIELD_DK_DIR/scripts
    ./create-boot-image.sh MLO unsigned_MLO MLO
    ```
    - Note: Boot-Image for SD-BOOT must have the **MLO** as the first argument
  - QSPI-BOOT (Input file: **unsigned_XLOADER**, Output file: **XLOADER** – Both are in little-endian format)

    ```
    cd $MSHIELD_DK_DIR/scripts
    ./create-boot-image.sh X-LOADER unsigned_XLOADER XLOADER
    ```
    - Note: Boot-Image for QSPI-BOOT must have the **XLOADER** as the first argument
  - In case of starterware, the **qspiFlashWriter** application expects input in big-endian format. This can be done in Linux using the following command
    - ```
      xxd -p -c 4 inp_LE | sed "s/\(..\)\(..\)\(..\)\(..\)/\4\3\2\1/" | xxd -r –p > out_BE
      ```

8.2 Extended Authentication

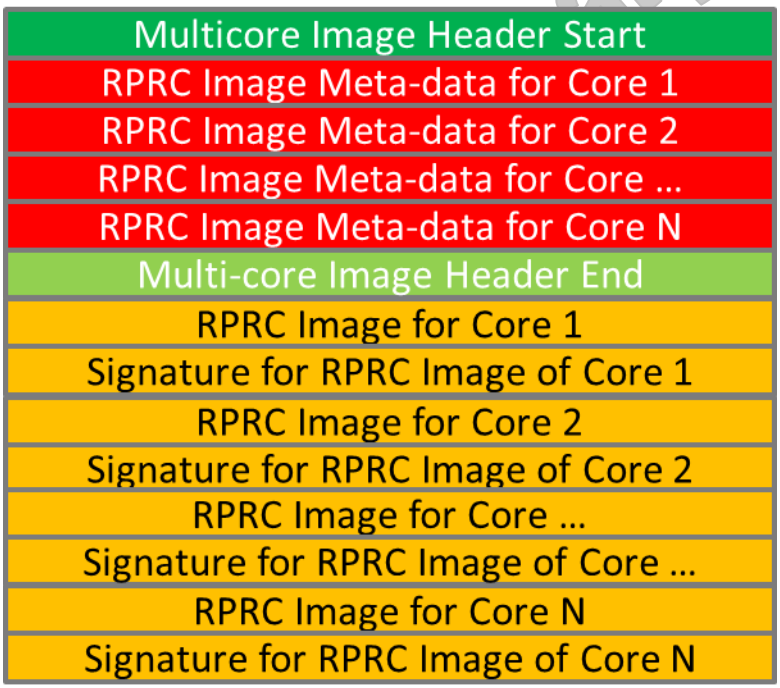
- Secure ROM is responsible for authentication of ISW or SBL only. SBL has the responsibility to authenticate any further software it loads into the system. This is known as "Extended authentication". SBL need not include any authentication software – it can use a ROM/PPA service to authenticate any data-blob as needed. In this section, we discuss how to generate a "signed" AppImage as required by TI SBL.
      ```

- MShield-DK provides an image-signing tool to sign any binary image
  - A signed image contains the original data padded with zero to reach a 4 byte aligned size, followed by a 280 byte of signature.
  - This can be generated using the following command

```
cd $MSHIELD_DK_DIR/scripts
./ift-image-sign.sh dra7xx input_binary output_binary
```

- AppImage formats can be customized by users as needed. VisionSDK uses a new AppImage format defined in starterware_/include/sbl_lib/sbl_lib.h. This “Version 2” format uses the structures suffixed with **v2** in this file.
  - TI SBL will not support AppImage built with older “Version 1” format (used in VisionSDK versions 2.9 or older) on HS devices.
- The new AppImage structure is as follows

Figure 5. AppImage structure



  - The RPRC image contains the code and data sections to be loaded on different CPUs in the system. The SBL will abort boot in case any of the RPRC image cannot be authenticated.
  - The RPRC image meta-data contains the size and offset of each RPRC image to enable authentication
- VisionSDK provides scripts to consolidate all these different steps. This is done using the **MulticoreImageGen_tda2xx.sh** or **MulticoreImageGen_tda2ex.sh** scripts. Before running these scripts users must ensure

- MShield-DK is installed in the VisionSDK installation directory. The **mshield_dk** folder must be at same level as **vision_sdk, ti_components,** etc.
- MShield-DK is setup for TDA2XX builds using

```
cd $MSHIELD_DK_DIR/src
perl config.pl
```

- SBL is built and corresponding signed binaries can be generated using following commands
  - `make -s sbl_sd sbl_qspi MAKECONFIG=tda2xx_evm_bios_all HS_DEVICE=yes`
  - `make -s sbl_sd sbl_qspi MAKECONFIG=tda2ex_evm_bios_all HS_DEVICE=yes`
- Signed AppImage is generated using
  - make -s appimage MAKECONFIG=tda2xx_evm_bios_all HS_DEVICE=yes
  - make -s appimage MAKECONFIG=tda2ex_evm_bios_all HS_DEVICE=yes
- Signed SBL images are generated in
  - build/scripts/qspi_tda2xx-evm_hs
  - build/scripts/qspi_tda2ex-evm_hs
  - build/scripts/sd_tda2xx-evm_hs
  - build/scripts/sd_tda2ex-evm_hs
- Signed AppImages are generated in
  - vision_sdk/binaries/tda2xx_evm_bios_all/vision_sdk/bin/tda2xx-evm/sbl_boot_hs/
  - vision_sdk/binaries/tda2ex_evm_bios_all/vision_sdk/bin/tda2ex-evm/sbl_boot_hs/
- There are no changes for VisionSDK compilation for HS devices.

- **Note**: NOR XIP builds for HS devices are not verified in VisionSDK 2.10 release

9 Boot Authentication and Extended Authentication on TDA3x

For TDA3x, refer to Starterware Security Add-on documentation for more details on building SBL images and AppImage. This section provides documentation on infrastructure provided within VisionSDK to generate SBL images and AppImages.

The signing process is supported only on Linux/Ubuntu machines. It depends on following tools

- OpenSSL (1.0.1f or higher)
- Python (2.7x)
- Standard linux commands like xxd, sed, cat, dd are used. /dev/urandom is used to generate random IV for optional encryption process.

9.1 Boot Authentication

To build SBL and generate signed SBL binary images, following commands can be used depending on boot-mode needed. Before building SBL make sure SBL_LIB_CONFIG_ENABLE_CRC is 0 for your SBL LIB configuration in <starterware_install_dir>/bootloader/sbl_lib/sbl_lib_config_tda3xx.h

- make -s -j sbl_qspi MAKECONFIG=tda3xx_evm_bios_all HS_DEVICE=yes
- make -s -j sbl_qspi_sd MAKECONFIG=tda3xx_evm_bios_all HS_DEVICE=yes

This will generate binaries for TDA3x GP Prime devices in the following folders:

- build/scripts/qspi_tda3xx-evm_hs/
- build/scripts/qspi_sd_tda3xx-evm_hs/

sbl_qspi_BE or **sbl_qspi_sd_BE** should be used when flashing with qspiFlashWriter_m4_release.xem4

9.2 Extended Authentication

There are no changes for VisionSDK compilation for HS devices. However, the AppImage generation steps needs to change.

SBL built using the Starterware security add-on for TDA3x GP Prime devices expects each RPRC image to be appended with a SHA2-HMAC digest. The overall structure is similar to one shown in Figure 5. The key used for this digest is expected to be present in first four rows of CEK. By default, these rows will be zero.

Users are expected to do one of following two things:

- Refer to documentation in <starterware_install_dir>/security/docs/ to flash the CEK using the efuse_app example in <starterware_install_dir>/security/examples
- Update <starterware_install_dir>/security/tools/tda3xx/image_enc_key.txt to use a string of 16 “0”s to use the default

- Signed AppImage are generated using the command:
  - make -s appimage MAKECONFIG=tda3xx_evm_bios_all HS_DEVICE=yes
- Signed AppImages are generated at following location
  - vision_sdk/binaries/tda3xx_evm_bios_all/vision_sdk/bin/tda3xx-evm/sbl_boot_hs/

9.3 Enabling AppImage encryption

By default, only AppImage authentication is enabled as the reference decryption library provided is not optimized for performance.

To enable encryption following steps should be done before building SBL and AppImage

- To enable support for AppImage encryption enabled the macro **SECURITYLIB_APPIMAGE_DECRYPTION** in
  - <starterware_install_dir>/include/security/tda3xx/tda3xx_gp_prime.h
- Set **SECURITYLIB_APPIMAGE_DECRYPTION=yes** in
  - <vision_sdk>/build/makerules/build_multi_core_image_gen.mk

Key used for encryption is expected same as that used for Authentication only mode and is expected to be present in first four rows of CEK. Refer to previous sub-section, for details on changing this.