Freescale Semiconductor
User's Guide

Document Number: AHDCP2XUG
Rev. L5.0.0_1.0.0-ga, 06/2015

Android™ HDCP2.x User's Guide

Contents

1 Overview

This document explains the steps to enable the HDCP2.X function based on the Android™ Extended Wi-Fi Display Sink Release package. It describes how to enable the High Assurance Boot (HAB) for an Android Image, how to generate and provision keyblob for HDCP2.X key, and how to enable the HDCP2.X functions based on i.MX Android Extended Wi-Fi Display Sink Release package. The reader should have read the *Android™ User's Guide* (AUG), *i.MX Android™ Extended Wi-Fi Display Sink Release Notes* (AEWDSRN), and *i.MX 6 Linux High Assurance Boot (HAB) User's Guide* (IMX6HABUG).

2 Preparation

Unpack the imx-android-hdcp2.x.tar.gz file by using the following commands:

```
$ cd ~ (or any other directory you like)
$ tar xzvf L5.0.0_1.0.0-HDCP2.x.tar.gz
$ cd L5.0.0_1.0.0-HDCP2.x
$ tar xzvf HDCPLib.tar.gz
$ tar xzvf mxc_secureboot.tar.gz
$ tar xzvf Patches.tar.gz
```

3 Patching i.MX Android HDCP2.x Package

3.1 Enabling security boot in the Android platform's U-Boot

Rebuild the Android platform's U-Boot image with the patches to myandroid/bootloader/bootable/u-boot. The patches are included in L5.0.0_1.0.0-HDCP2.x/Patches/android, which can be applied as:

```
$ cd myandroid/bootloader/bootable/u-boot
$ git apply ~/L5.0.0_1.0.0-HDCP2.x/Patches/android/bootloader/bootable/u-boot/0001-Enable-
CONFIG_SECURE_BOOT-for-android-bootloader.patch
```

3.2 Enabling security boot in MFG Tool's U-Boot

Rebuild the MFG Tool's U-Boot image with the patches to myandroid/bootloader/bootable/u-boot. The patches are included in L5.0.0_1.0.0-HDCP2.x/Patches/mfgtool, which can be applied as:

```
$ cd myandroid/bootloader/bootable/u-boot
$ git apply ~/L5.0.0_1.0.0-HDCP2.x/Patches/mfgtool/u-boot/0001-Enable-CONFIG_SECURE_BOOT-for-
android-bootloader.patch
```

3.3 Disabling non-security features in boot image

Rebuild boot.img with the following patches in myandroid/device/fsl.

All these patches are included in L5.0.0_1.0.0-HDCP2.x/Patches/android, which can be applied as:

```
$ cd myandroid/device/fsl
$ git apply ~/L5.0.0_1.0.0-HDCP2.x/Patches/android/device/fsl/0001-Disable-runtime-config-on-
dm_verity.patch
$ git apply ~/L5.0.0_1.0.0-HDCP2.x/Patches/android/device/fsl/0002-Change-shell-to-be-
executed-in-shell-user.patch
```

3.4 Disabling DM_VERITY runtime configurable feature

The patch 0002-Disable-runtime-config-on-dm_verity.patch is used to disable DM_VERITY runtime configurable feature in boot.img. It has the following changes. Users should follow the *Android™ Frequently Asked Questions* to generate the keys to sign the verity table of system.img.

```
diff --git a/sabresd_6dq/BoardConfig.mk b/sabresd_6dq/BoardConfig.mk
index 516c8e5..86800e4 100644
--- a/sabresd_6dq/BoardConfig.mk
+++ b/sabresd_6dq/BoardConfig.mk
@@ -80,7 +80,7 @@ TARGET_SELECT_KEY := 28
# we don't support sparse image.
TARGET_USERIMAGES_SPARSE_EXT_DISABLED := false
-DM_VERITY_RUNTIME_CONFIG := true
+DM_VERITY_RUNTIME_CONFIG := false
# uncomment below lins if use NAND
#TARGET_USERIMAGES_USE_UBIFS = true
```

NOTE

With DM_VERITY enabled, the boot.img and recovery.img files include the verity metadata of system.img. Anything changed in system.img needs a rebuild on boot.img and recovery.img. Otherwise, boot.img and recovery.img fail to load system.img.

3.5 Enabling the CAAM driver in kernel DTS

Rebuild boot.img with the following patches in myandroid/kernel_imx.

All these patches are included in L5.0.0_1.0.0-HDCP2.x/Patches/android, which can be applied as:

```
$ cd myandroid/kernel_imx
$ git apply ~/L5.0.0_1.0.0-HDCP2.x/Patches/android/kernel_imx/0001-enable-caam-keyblob-in-
dts.patch
Enable caam driver in MFG Tool's kernel dts
Rebuild the zImage with below patches in myandroid/kernel_imx
All these patches are included in L5.0.0_1.0.0-HDCP2.x/Patches/mfgtool, which can be applied
as:
$ cd myandroid/kernel_imx
$ git apply ~/L5.0.0_1.0.0-HDCP2.x/Patches/android/kernel_imx/0001-MA-6261-Create-caam-
keyblob-driver-for-general-memor.patch
$ git apply ~/L5.0.0_1.0.0-HDCP2.x/Patches/android/kernel_imx/0002-enable-caam-keyblob-in-
dts.patch
```

3.6 Adding HDCP support based on Android extended Wi-Fi display sink package

The HDCPLib.tar.gz package is used to add HDCP support in Android Extended Wi-Fi Display Sink. The following are the steps to add the HDCP support:

1. Follow Section 3.3 "Patch Freescale extended features code" in the *Android™ User's Guide* (AUG) to patch the Android Extended Wi-Fi Display Sink.
2. Use the following commands to patch HDCP librarys to wfd-proprietary:

   ```
   tar zxvf ~/L5.0.0_1.0.0-HDCP2.x/HDCPLib.tar.gz  -C ~/myandroid/device/wfd-proprietary/
   ```
3. Follow Section 3.4 "Building Android images" in the *Android™ User's Guide* (AUG) to build Android images.

4 Android High Assurance Boot (HAB)

4.1 Generating key pair and public key's fuse file, and enabling HAB for i.MX 6DualQuad/6DualLite

To generate key pair and public key's fuse file, and enable HAB for i.MX 6DualQuad/6DualLite, perform the following steps:

1. Download the CST Tool as described in the *i.MX 6 Linux High Assurance Boot (HAB) User's Guide* (IMX6HABUG).
2. Follow the steps from 1 to 6, and 18 in "4 Test Procedure" in the *i.MX 6 Linux High Assurance Boot (HAB) User's Guide* (IMX6HABUG).

4.2 Signing the Android boot/recovery image and U-Boot image

1. Prepare tools for HAB:

```
$ cd ~
$ tar xzvf mxc_secureboot.tar.gz
# Assuming the cst tools installed into ~/cst-2.2
$ cd ~/mxc_secureboot
$ cp mxc_secureboot/template ~/cst-2.2/. -r
$ cp mxc_secureboot/u-boot ~/cst-2.2/. -r
$ cp mxc_secureboot/zImage ~/cst-2.2/. -r
$ cp mxc_secureboot/bootimg ~/cst-2.2/. -r
```

2. Sign the U-Boot image:

```
$ cp ~/myandroid/out/target/product/sabresd_6dq/u-boot-imx6q.imx ~/cst-2.2/u-boot/u-
boot.imx
$ cd ~/cst-2.2/u-boot
$ ./mk_secure_uboot
$ mv u-boot-signed-pad.imx u-boot-imx6q-signed-pad.imx
```

3. Sign the recovery image:

```
$ cp ~/myandroid/out/target/product/sabresd_6dq/recovery-imx6q.img ~/cst-2.2/bootimg/
boot.img
$ cd ~/cst-2.2/bootimg
$ ./mk_secure_bootimg 0x14007800
$ mv boot-signed-pad.img recovery-imx6q-signed-pad.img
```

4. Sign the boot image:

```
$ cp ~/myandroid/out/target/product/sabresd_6dq/boot-imx6q.img ~/cst-2.2/bootimg/
boot.img
$ cd ~/cst-2.2/bootimg
$ ./mk_secure_bootimg 0x14007800
$ mv boot-signed-pad.img boot-imx6q-signed-pad.img
```

5. Sign the zImage:

```
$ cp ~/myandroid/kernel_imx/arch/arm/boot ~/cst-2.2/zImage
$ cd ~/cst-2.2/zImage
$ ./mk_secure_zimage 0x10800000
```

NOTE

- The mk_secure_bootimg script only takes boot.img as the input file, and boot-signed-pad.img as the output file. When being used to sign recovery.img, the recovery.img needs to be renamed as boot.img.
- The parameter 0x14007800 can be calculated as the description in mxc_secureboot/V2012/README.

4.3 Downloading Android images to the SD Card or eMMC with the MFG tool

To download Android images to the SD Card or eMMC with the MFG tool, perform the following steps:

1. The bootloader u-boot-imx6q-sabresd.imx in mfgtools\Profiles\Linux\OS Firmware\firmware needs to be built with the patches in L5.0.0_1.0.0-HDCP2.x/Patches/mfgtool/uboot.
2. The zImage in mfgtools\Profiles\Linux\OS Firmware\firmware needs to be built with the patches in L5.0.0_1.0.0-HDCP2.x/Patches/mfgtool/kernel_imx.
3. Sign u-boot-imx6q-sabresd.imx and zImage in mfgtools\Profiles\Linux\OS Firmware\firmware as steps 11 and 12 in the link: community.freescale.com/docs/DOC-96451
4. See the *Android™ Quick Start Guide* (AQSUG) to download the signed Android boot image and U-Boot image.

NOTE

- The code base for U-Boot and kernel in the MFG tool should be from the i.MX 6 Linux 3.10.53_1.1.0 GA release.
- The default u-boot-imx6q-sabresd.imx in the default Android release MFG tool does not enable the security boot.
- User need to follow the *i.MX 6 Linux High Assurance Boot (HAB) User's Guide* (IMX6HABUG) to build a security boot enabled U-Boot for MFG tools.
- Users need to sign the U-Boot and zImage in the MFG tool.
- The process of signing U-Boot of the MFG tool is different from signing U-Boot for booting device from the SD Card or eMMC.

5 HDCP Keyblobs

There are two keys for each HDCP Sink device as shown in the following figure.

- 128 bit Secret Global Constant (lc 128). The length of Secret Global Constant (lc 128) is 36 bytes, which includes its SHA1.
- Device Certification, which includes the Device Public Certification and Device Private Certification. The Device Certification is 862 bytes, which includes its SHA1.

The Wi-Fi Display Sink needs to access two keys to support the Wi-Fi Display Source with HDCP enabled. It is high risk to store these two keys directly in storage. i.MX 6Quad/6DualLite has the hardware security module to encrypt key to keyblob and decrypt keyblob to key. Device only needs to store the HDCP keyblobs in storeage. The HDCP keys can only be decrypted by the same i.MX 6Quad/6DualLite, which can be used to generate the keyblob. The decryption and encryption process is performed with the hardware security module, which is acted as the black box to users.

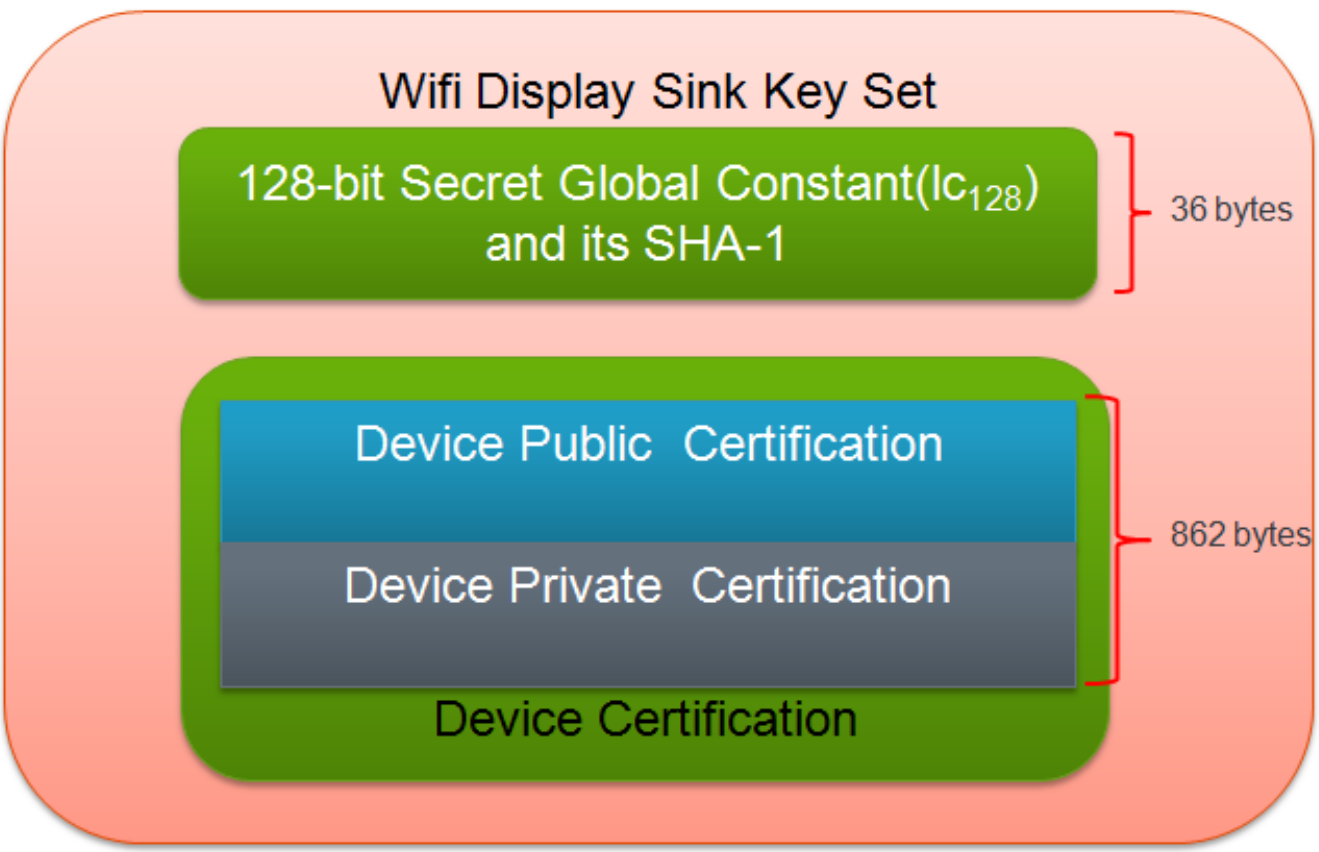

Figure 1. HDCP keyblobs

5.1 Generating HDCP key files

The user should request HDCP keys for Sink device from the Digital Content Protection, LLC (DCP). The user should have the HDCP keys for Sink device as the following format:

HDCP Sink Key text format

KSV: xxxxx

Content: xxxx

To generate HDCP key files, perform the following steps:

1. Copy the first 72 characters in Content to the file lc_128.txt. and the remainder in Content to the hdcp2_key.txt.
2. Use the following commands to generate the key file in Linux OS:

```
$ xxd -r -p lc_128.txt hdcpv2trans_license_constant.bin
$ xxd -r -p hdcp2_key.txt hdcpv2trans_kp.bin
```

5.2 Generating and provisioning HDCP Keyblobs

The mfgtools-HDCP2x.zip file is used to generate and provision HDCP keyblobs into the system. The keyblobs are stored in the device partition. The setps are as follows:

1. Unzip the mfgtools-HDCP2x.zip file into a Windows® OS computer.
2. Put the HDCP keys in the following folder. Make sure to have the same file name as follows:
    - mfgtools-HDCP2x\Profiles\Linux\OS
        - Firmware\files\android\hdcp-keys\hdcpv2trans_kp.bin
    - mfgtools-HDCP2x\Profiles\Linux\OS
        - Firmware\files\android\hdcp-keys\hdcpv2trans_license_constant.bin
3. Make your board into download mode.
4. Click the VBS according to your board type.

Table 1. Board types and VBS

Board type	Boot storage	VBS
1 GHZ i.MX 6DualQuad SABRE-SD	eMMC	mfgtool2-android-mx6q-sa bresd-emmc-hdcp2x.vbs
1.2 GHZ i.MX 6DualQuad SABRE-SD	eMMC	mfgtool2-android-mx6q-sa bresd-emmc-1.2g-hdcp2x.vbs
i.MX 6DualLite SABRE-SD	eMMC	mfgtool2-android-mx6dl-s abresd-emmc-hdcp2x.vbs
1 GHZ i.MX 6DualQuad SABRE-SD	SD card	mfgtool2-android-mx6dl-s abresd-sd-hdcp2x.vbs
1.2 GHZ i.MX 6DualQuad SABRE-SD	SD card	mfgtool2-android-mx6q-sa bresd-sd-1.2g-hdcp2x.vbs
i.MX 6DualLite SABRE-SD	SD card	mfgtool2-android-mx6dl-s abresd-sd-hdcp2x.vbs

NOTE

- The steps in this section should be done after the Android images are downloaded into the device with the MFG tool. Otherwise, the process of Android images downloading will erase the HDCP keyblobs in the device partition.
- Users should follow the steps 1 - 3 in Downloading Android images to the SD Card or eMMC with the MFG tool to generated the bootloader and zImage for the MFG tool.

How to Reach Us:

Home Page:
freescale.com

Web Support:
freescale.com/support

Document Number: AHDCP2XUG
Rev. L5.0.0_1.0.0-ga
06/2015