AN13303 SNxxx/PN557- NFC Host SW Integration Guideline

Rev-2.2 — 10/22/2024 Application Note

Document information

Info	Content
Keywords	NFC, Android

Revision history

Rev	Date	Description
1.0	2023-12-15	Initial version for Android 15 NXP NFC Host SW Integration Guide
2.0	2024-08-30	Android 15 Observe mode config options
2.1	2024-09-18	Android 15 eUICC SMB debug, KM2.0 configuration support
2.2	2024-10-16	Updated Android 15 Observe mode config options

1. Introduction

NXP’s NFC controller SNxxxT/U and PN557 are designed to work with Android open source.

Below Table shows the NXP’s development and validation platform setup.

Chip Type	Platform	NFC/SE Board
SN110	RB3	Iguana Lite Board
Sn220 & later	RB3	Komodo
PN557	RB3	PN8x Daughter Board

2. Abbreviations

NFC	Near Field Communication
OEM	Original Equipment Manufacturer
HW	Hardware
IC	Integrated Circuit
SWP	Single Wire Protocol
GPIO	General Purpose Input / Output
I2C	Inter-Integrated Circuit
SPI	Serial Peripheral Interface
SW	Software
SE	Secure Element
OMAPI	Open Mobile Application Programming Interface
AOSP	Android Open Source Project
HAL	Hardware Abstraction Layer
eSE	Embedded Secure Element
OS	Operating System
SEMS	Secure Element Management Service
LS	Loader Service
GSMA	GSM Association
GSM	Global System for Mobile
NFCC	NFC Controller
SMB	System Mail Box
HIDL	HAL interface definition language
UICC	Universal Integrated Circuit Card
ISO	International Organization for Standardization
DH	Device Host
DTA	Device Test Application
NA	Not Applicable
MPOS	Mobile Point of Sale
TEE	Trusted Execution Environment

3. Scope

This document provides guidelines for setting up NXP's new generation NFC/SE monolithic platform SNxxxT/U and NFC only PN557 in Android 15 build environment. It is a reference guideline for basic system integration. OEM integration may have variations based on actual system integration.

4. General steps for Android NFC integration

For the NFC software integration with Android, it is hereby assumed that NFC IC HW integration is done in a platform with following checks.

- Schematic reviewed with NXP
- HW IC interface like I2C/SPI, SWP (if used) working.
- Antenna designed and reviewed
- Antenna connection working
- GPIO connections checked

Fig. 4, shows the basic flow for Android NFC SW bring up. Following sections describe these steps in detail.

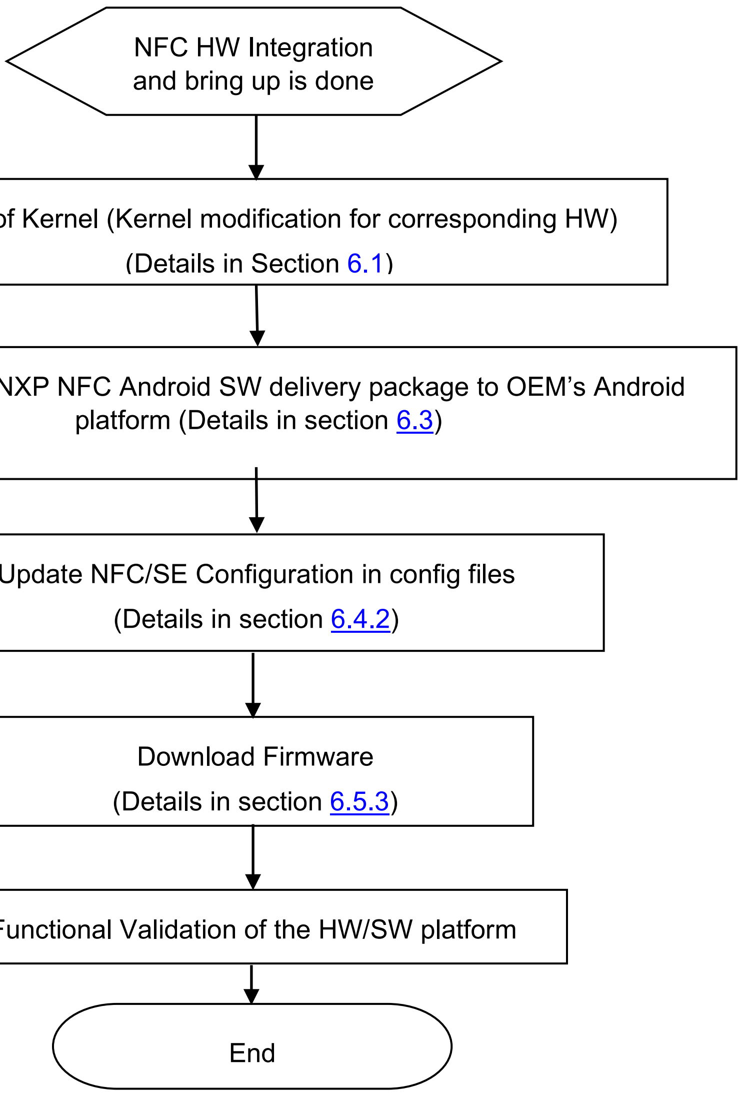
NFC HW Integration and bring up is done
Porting of Kernel (Kernel modification for corresponding HW) (Details in Section 6.1)
Integration of NXP NFC Android SW delivery package to OEM's Android platform (Details in section 6.3)
Update NFC/SE Configuration in config files (Details in section 6.4.2)
Download Firmware (Details in section 6.5.3)
Functional Validation of the HW/SW platform
End

Figure 4: Android NFC SW bring up flow

5. Architecture Overview

Fig. 5, describes the architecture of Android 15 based NXP delivery package.

OMAPI implementation is part of the AOSP from Android P version onwards and NXP does not make any modification in Android OMAPI service layer.

Note: SEHal, WeaverHal, KeyMint Hal, AuthSecret HAL and SPIDriver are not applicable and shall not be integrated for NFC only product PN557.

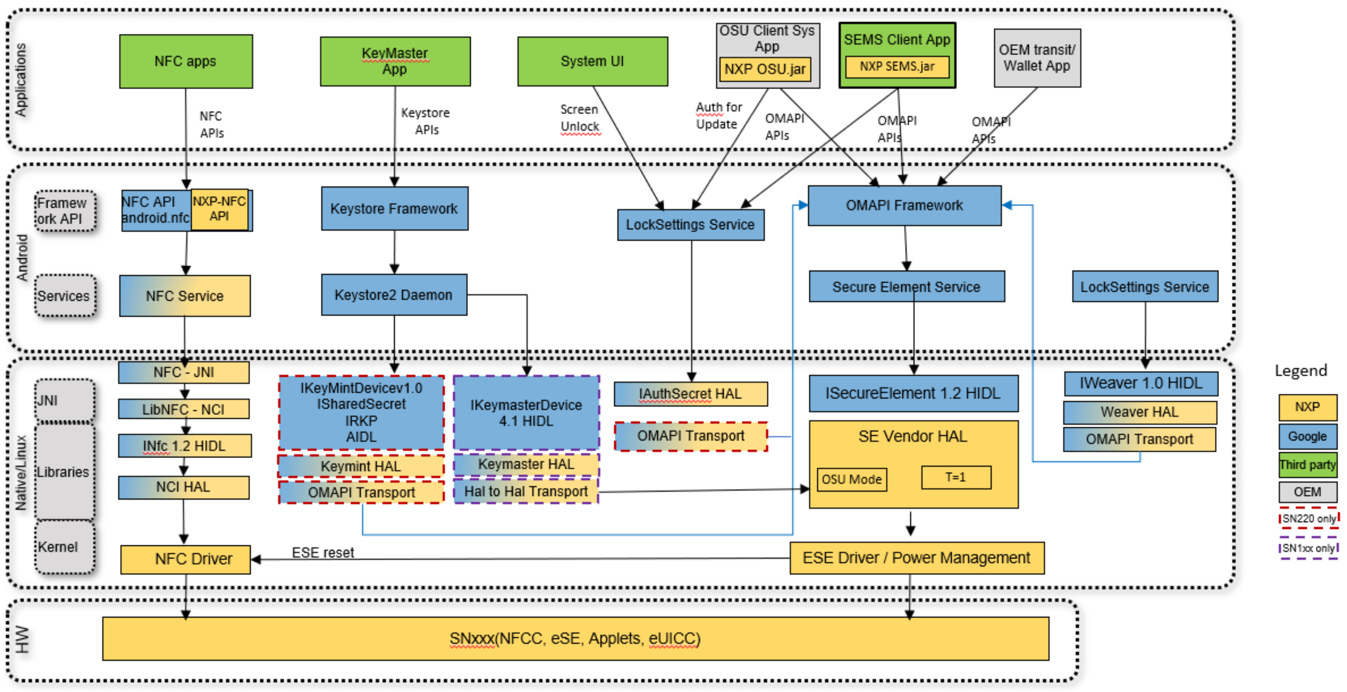

Figure 5: Secure NFC MW architecture

6. Setup of Android NFC

6.1 Android Kernel driver setup for NXP-NFCC and eSE

6.1.1 SNxxx

The db845c platform kernel can be downloaded by the below command:

```
repo init -u https://android.googlesource.com/kernel/common
repo sync -j8
```

Additional information regarding db845 kernel:
git branch: 5.10
git commit: ea897dd10d35334b86d85e4cf23169d550220921

Steps to perform in platform's kernel root directory to integrate NXP specific I2C and SPI drivers for accessing NFCC and eSE.

1. Download NFC I2C & SPI drivers from below git hub location:
   https://github.com/NXPNFCProject/NXPNFC_I2CDriver/tree/br_android_ncihalx_comm_15
   https://github.com/NXPNFCProject/NXPESE_SPIDriver/tree/br_android_ncihalx_comm_15
2. Create nxp folder inside common/drivers/
3. Copy nfc from NXPNFC_I2CDriver and keep inside common/driver/nxp
4. Copy ese from NXPNFC_SPIDriver to common/driver/nxp
5. Include the driver Makefile folder path in the higher level Makefile in hierarchy
6. Include the Kconfig source to the higher level Kconfig in hierarchy
7. Add the DTS changes required in your platform DTS file

```
clock-frequency = <1000000>;
sn-i2c@28 {
    compatible = "nxp,sn-nci";
    reg = <0x28>;
    nxp,sn-irq = <&gpio26 0 0>;
    nxp,sn-ven-rstn = <&gpio26 1 0>;
    nxp,sn-dwl-req = <&gpio26 2 0>;
 };
p61@0 {
    compatible = "nxp,p61";
    reg = <0>;
    nxp,p61-irq = <&gpio2 3 0>;
    nxp,p61-rst = <&gpio2 5 0>;
    nxp,trusted-se = <&gpio26 4 0>;
    spi-max-frequency = <20000000>;
    nxp,nfcc = "2-0028";        };
```

8. Set the kernel configuration to build driver as static or dynamic in the platform config file
    a. Static Linking with kernel image
       CONFIG_ NXP_NFC_I2C=y
       CONFIG_ NXP_ESE_P73=y

    b. Dynamic as module(.ko)
       CONFIG_ NXP_NFC_I2C=m
       CONFIG_ NXP_ESE_P73=m

9. Compile the kernel using corresponding cross compiler and copy the generated Image.gz-dtb file to the ANDROID_ROOT/device/vendor/platform-kernel

Note: It is recommended to apply the patches manually.
Steps 2-6 are only required for building driver in-tree during building kernel.
Max frequency supported can be adjusted in dts as in step 7.
Ex. For SNxxx based platforms:
For I2c: clock-frequency = <3400000>
For SPI: spi-max-frequency = <25000000>

6.1.2 PN557

The db845c platform kernel can be downloaded by the below command:

```
repo init -u https://android.googlesource.com/kernel/common
repo sync -j8
```

Additional information regarding db845 kernel:
git branch: 5.10
git commit: ea897dd10d35334b86d85e4cf23169d550220921

Steps to perform in platform's kernel root directory to integrate NXP specific I2C driver for accessing NFCC

1. Download NFC I2C driver from below git hub location:
   https://github.com/NXPNFCProject/NXPNFC_I2CDriver

2. Create nxp folder inside kernel/driver/

3. Copy nfc from NXPNFC_I2CDriver and keep inside kernel/driver/nxp

4. Include the driver Makefile folder path in the higher level Makefile in hierarchy

5. Include the Kconfig source to the higher level Kconfig in hierarchy

6. Add the DTS changes required in your platform DTS file

```
ldo11: LDO11 { /* Low Speed Connector */
        regulator-name = "VOUT11_1V8_2V95";
        regulator-min-microvolt = <1825000>;
                        regulator-always-on;
        regulator-enable-ramp-delay = <240>;
};
 clock-frequency = <1000000>;
 sn-i2c@28 {
    compatible = "nxp,sn-nci";
    reg = <0x28>;
    nxp,sn-irq = <&gpio26 0 0>;
    nxp,sn-ven-rstn = <&gpio26 1 0>;
    nxp,sn-dwl-req = <&gpio26 2 0>;
  };
```

7. Set the kernel configuration to build driver as dynamic in the platform config file
   CONFIG_NXP_NFC_I2C=m

8. Compile the kernel using corresponding cross compiler and copy the generated Image.gz-dtb file to the ANDROID_ROOT/device/vendor/platform-kernel

6.2 Setup of Android NFC for DB845C

6.2.1 Downloading Android source code

Use following command to get source code for Android-<x>.<y>:

```
repo init -u https://android.googlesource.com/platform/manifest -b android-<x>.<y>

repo sync -f
```

Note: x & y represents Android major & minor versions

For detailed steps to download Android source code refer Android website:

http://source.android.com/source/downloading.html

6.2.2 Building the source code

Use android build instructions from Android website for building android OS image:
http://source.android.com/source/building.html

Build name for RB3 development board is **DB845C.** For device specific build (e.g. RB3), additional steps as described in link below needs to be followed.
https://source.android.com/setup/build/running

Information about the public APIs supported by Android NFC are available on following links:
http://developer.android.com/reference/android/nfc/package-summary.html

http://developer.android.com/reference/android/nfc/tech/package-summary.html

6.2.3 Building driver out of kernel tree(for arm64 arch)

Following are the steps to build the NFCC and ESE driver out-of-tree with the ACK, validated with the launch kernel android13-5.10 for A15.

1. Follow the steps mentioned in 6.1 for kernel sync & NXP NFC/SE driver integration.
2. Run below command for kernel out of tree compilation according to target platform. Command for db845c.
   - BUILD_NUMBER= BUILD_CONFIG=common/build.config.db845c ./build/build.sh -j$(nproc)

   SKIP_MRPROPER=1 SKIP_CP_KERNEL_HDR=1 are optional parameters to optimize the kernel build time during incremental build.
3. Copy the generated ko files to the android build environment for example "device/linaro/dragonboard-kernel/android-5.10" for db845c.
4. Compile NXP Driver code, under vendor/nxp/drivers/ for nfc and spi

6.3 Android NXP NFC SW Delivery Package

6.3.1 Android NXP NFC Package Description

Project/Repository	Repository Link	Branch
NFC_NCIHAL_base	https://github.com/NXPNFCProject/NFC_NCIHAL_base	br_android_ncihalx_comm_15
NFC_NCIHAL_Nfc	*https://github.com/NXPNFCProject/NFC_NCIHAL_Nfc*	br_android_ncihalx_comm_15
NFC_NCIHAL_libnfc-nci	https://github.com/NXPNFCProject/NFC_NCIHAL_libnfc-nci	br_android_ncihalx_comm_15
nfcandroid_nfc_hidlimpl	https://github.com/NXPNFCProject/nfcandroid_nfc_hidlimpl	br_android_ncihalx_comm_15
nfcandroid_se_hidlimpl	https://github.com/NXPNFCProject/nfcandroid_se_hidlimpl	br_android_ncihalx_comm_15
nfcandroid_secureelement	https://github.com/NXPNFCProject/nfcandroid_secureelement	br_android_ncihalx_comm_15

nfcandroid_weaver_hidlimpl	https://github.com/NXPNFCProject/nfcandroid_weaver_hidlimpl	br_android_ncihalx_comm_15
nfcandroid_keymint_hidlimpl	https://github.com/NXPNFCProject/nfcandroid_keymint_hidlimpl	br_android_ncihalx_comm_15
nfcandroid_nxp_ese_clients	https://github.com/NXPNFCProject/nfcandroid_nxp_ese_clients	br_android_ncihalx_comm_15
NXPNFC_Reference	https://github.com/NXPNFCProject/NXPNFC_Reference	br_android_ncihalx_comm_15
NXPNFC_I2CDriver	https://github.com/NXPNFCProject/NXPNFC_I2CDriver	br_android_ncihalx_comm_15
NXPESE_SPIDriver	https://github.com/NXPNFCProject/NXPESE_SPIDriver	br_android_ncihalx_comm_15
NFC_NCIHAL_docs	https://github.com/NXPNFCProject/NFC_NCIHAL_docs	br_android_ncihalx_comm_15
nfc-NXPNFCC_FW	https://github.com/NXP/nfc-NXPNFCC_FW	master
NXPAndroidDTA	https://github.com/NXPNFCProject/NXPAndroidDTA	master
nfcandroid_frameworks	https://github.com/NXPNFCProject/nfcandroid_frameworks.git	br_android_ncihalx_comm_15

Table 1: Android NXP NFC Package Description

6.3.2 Integration of NXP NFC Modules for SNxxx & PN557

Modify/Add AOSP directories in-place with NXP GitHub sources as per the following table:

Module	NXP GitHub sources	Integration Path	Description	Applicable Chip type
NFC Interface and Public APIs	NFC_NCIHAL_base /core/java/ android/nfc	$ANROID_ROOT/frameworks/ base/core/ java/android/nfc	NFC Interfaces & Public APIs for Google AOSP	SNxxx & PN557
NFC JNI and JAVA implementation of NCI stack	*NFC_NCIHAL_Nfc* /nci	$ANDROID_ROOT/packages/ apps/Nfc/nci	Includes Java files and JNI for NCI NFC stack. It is modified minimally to adapt new features provided by NXP.	SNxxx & PN557
	NFC_NCIHAL_Nfc	$ANDROID_ROOT/packages/ apps/Nfc	It is a derived module originally from AOSP. It is modified minimally to adapt new features provided by NXP.	SNxxx & PN557
NCI based NFC stack implementation	*NFC_NCIHAL_libnfc-nci*	$ANDROID_ROOT/system/nfc	NCI based NFC stack. It is a derived module originally from AOSP (Android Open Source Project). It is modified to adapt new features provided by NXP	SNxxx & PN557

HAL implementation for NFC	nfcandroid_nfc_hidlimpl	$ANDROID_ROOT/hardware/nxp/nfc	Hardware abstraction layer for NXP specific controllers. This directory includes the configuration files also as below. 1.libnfc-nci.conf (to be pushed to vendor/etc on target) 2.libnfc-nxp-sn100x_example.conf (to be pushed to vendor/etc on target as libnfc-nxp.conf. 3.libnfc-nxp_RF-sn100x_example.conf(to be pushed to /vendor/ on target) NOTE: these configuration files are example files. Contact NXP support engineer for creating exact file for your platform.	SNxxx & PN557
HAL implementation for Secure Element	nfcandroid_se_hidlimpl	$ANDROID_ROOT/hardware/nxp/secure_element	Hardware abstraction layer implementation for Secure Element.	SNxxx
HAL implementation for Weaver	nfcandroid_weaver_hidlimpl	$ANDROID_ROOT/hardware/nxp/weaver	Hardware abstraction layer implementation for Weaver.	SNxxx
HAL implementation for keymint	nfcandroid_keymint_hidlimpl	$ANDROID_ROOT/hardware/nxp/keymint	Hardware abstraction layer implementation for Keymint	SNxxx
HAL implementation for authsecret	nfcandroid_keymint_hidlimpl/authsecret	$ANDROID_ROOT/hardware/nxp/keymint/authsecret	Hardware abstraction layer implementation for authsecret	SNxxx
SE Service	nfcandroid_secureelement	$ANDROID_ROOT/packages/apps/SecureElement	AOSP Secure Element Service	SNxxx
eSe Client Library	nfcandroid_nxp_ese_clients	$ANDROID_ROOT/hardware/nxp/secure_element_extns	NXP eSE client library implementation	SNxxx
Vendor APIs	nfcandroid_frameworks	$ANDROID_ROOT/vendor/nxp/frameworks	NXP vendor framework APIs for NXP extension interfaces, SEMS & GSMA interfaces.	SNxxx & PN557
NFC I2C Driver	NXPNFC_I2CDriver/nfc	$KERNEL_ROOT/drivers/nxp/nfc	NFCC I2C Interface	SNxxx & PN557
NFC SPI Driver	NXPESE_SPIDriver/ese	$KERNEL_ROOT/drivers/nxp/ese	NFCC SPI Interface	SNxxx
Nxp Nfc Documentation	NFC_NCIHAL_docs	NA	NXP framework Java Docs	SNxxx & PN557
NFCC Firmware	nfc-NXPNFCC_FW	$ANDROID_ROOT/system/vevendor/lib64	NFCC FW binary	SNxxx & PN557
DTA	NXPAndroidDTA	$ANDROID_ROOT/system/nfc-dta/	Device Test Application (DTA) used for NFC Forum testing.	SNxxx & PN557
SePolicy	NXPNFC_Reference//nxp/SNxxx/sepolicy	$ANDROID_ROOT//vendor/nxp/SNxxx/sepolicy	SE Policy updates for NFC and SE service	SNxxx & PN557

Table 2 : Android NXP NFC Integration

6.3.3 Android NFC Apps and Lib on Target

Projects	Compiled Files	Location in target device
NFCNCIHAL_base/core/java/android/nfc	Will be part of framework.jar	/system/framework
NFC_NCIHAL_Nfc	lib/ NfcNci.apk oat/ libnfc_nci_jni.so	/system/app/NfcNci /system/lib64/
nfcandroid_secureelement	oat/ SecureElement.apk	/sytem/app/SecureElement
NFC_NCIHAL_libnfc-nci	libnfc_nci.so	/system/lib64
nfcandroid_nfc_hidlimpl	nfc_nci_nxp_snxxx.so android.hardware.nfc-service.nxp	/vendor/lib64 /vendor/bin/hw/
nfcandroid_nfc_hidlimpl/extns	vendor.nxp.nxpnfc_aidl-V1-ndk.so	/vendor/lib64
nfcandroid_se_hidlimpl	ese_spi_nxp_snxxx.so android.hardware.secure_element-service.nxp	/vendor/lib64 /vendor/bin/hw/
nfcandroid_keymint_hidlimpl	libjc_keymint.nxp.so libjc_keymint_transport.nxp.so android.hardware.security.keymint-service.strongbox.nxp android.hardware.authsecret-service.nxp	/vendor/lib64 /vendor/lib64 /vendor/bin/hw /vendor/bin/hw
nfcandroid_weaver_hidlimpl	ese_weaver.so android.hardware.weaver-service.nxp	/vendor/lib64 /vendor/bin/hw
nfcandroid_nxp_ese_clients	se_extn_client.so	/vendor/lib64
Nfcandroid_frameworks	com.nxp.nfc.jar com.nxp.sems.jar com.nxp.osu.jar	/system/framework /product/framework /product/framework

Table 3 : Android NXP NFC Apps & Library Info on Target

6.3.4 Android Platform Modifications

6.3.4.1 Android platform specific patches

Follow Step 1 to enable the following:

- Enable NFC, host card emulation and HCE-Felica features.
- Provide permission to i2c(nxp-nci) and spi(p73) driver for NFC Hal and SE Hal
- Assign object type for i2c(nxp-nci) and spi(p73) devices for providing se policy permissions
- Android SE Policy changes (these changes help in defining types, classes, permissions and rules for Nfc, SE, Strongbox & Weaver Hal service)
    1. Integrate all required sepolicy. Reference SE policy changes are available in below link https://github.com/NXPNFCProject/NXPNFC_Reference/tree/br_android_ncihalx_comm_15/nxp/SNxxx/sepolicy

       Make sure to add corresponding HAL SEPolicy dirs to device makefile. Example as below

```
BOARD_SEPOLICY_DIRS += vendor/$(NXP_VENDOR_DIR)/SNxxx/sepolicy \
                vendor/$(NXP_VENDOR_DIR)/SNxxx/sepolicy/authsecret \
                vendor/$(NXP_VENDOR_DIR)/SNxxx/sepolicy/keymint \
                vendor/$(NXP_VENDOR_DIR)/SNxxx/sepolicy/nfc \
                vendor/$(NXP_VENDOR_DIR)/SNxxx/sepolicy/se \
                vendor/$(NXP_VENDOR_DIR)/SNxxx/sepolicy/weaver
```

6.3.4.2 Android Source Build

To perform a full build, execute the following command from android root directory:

- ➢ cd $ANDROID_ROOT/
- ➢ make api-stubs-docs-non-updatable-update-current-api
- ➢ make system-api-stubs-docs-non-updatable-update-current-api
- ➢ make -j$(nproc)

6.4 Host SW Source Package Compilation

6.4.1 Compilation Flags

NXP_EXTNS=TRUE	Enable NXP extensions

Table 4: Compilation Flags

6.4.2 Configuration Files

Host specific configuration are available in the below path and all the configs are self-explanatory and some of the configs are listed below

SN110 config path:

https://github.com/NXPNFCProject/nfcandroid_nfc_hidlimpl/tree/br_android_ncihalx_comm_15/snxxx/halimpl/conf/SN1xx/sn110/gen-config-files

SN100 config path:

https://github.com/NXPNFCProject/nfcandroid_nfc_hidlimpl/tree/br_android_ncihalx_comm_15/snxxx/halimpl/conf/SN1xx/sn100/gen-config-files

SNXXX config path: Folder with chiptype name on below path <SNXXX>/gen-config-file

Ex. Sn220/gen-config-file

https://github.com/NXPNFCProject/nfcandroid_nfc_hidlimpl/tree/br_android_ncihalx_comm_15/snxxx/halimpl/conf/

PN557 config path:

https://github.com/NXPNFCProject/nfcandroid_nfc_hidlimpl/tree/br_android_ncihalx_comm_15/snxxx/halimpl/conf/PN557/gen-config-files

SN300 Config path:

https://github.com/NXPNFCProject/nfcandroid_nfc_hidlimpl/tree/br_android_ncihalx_comm_15/snxxx/halimpl/conf/SN300/gen-config-files

PN560 Config path:

https://github.com/NXPNFCProject/nfcandroid_nfc_hidlimpl/tree/br_android_ncihalx_comm_15/snxxx/halimpl/conf/PN560/gen-config-files

6.5 Feature Integration guideline

6.5.1 OMAPI Secure Element terminal configuration

Assignment of terminal number to each SE interface (SPI) is based on system configuration in **libnfc-nxp-<snxxx>-example.conf**. These terminals are mapped to OMAPI framework SEService readers list. This section is not applicable for PN557.

Terminal Naming should start from eSE1 and continue in ascending order

(This is as per OMAPI SE service implementation)

Only terminal which are mapped in configuration file are reflected as readers available in SE service.

For Example: -

Order below is just an example

NXP_SPI_SE_TERMINAL_NUM="**eSE1**" -> eSE domain accessed via SPI interface

Additionally, from Android 11 onwards it is mandatory to enable terminals as per the system configuration in vendor/etc/vintf/manifest/secure_element-service-nxp.xml.xml

Based on number of terminals getting enabled in config file corresponding number of terminal instances need to be updated in manifest.xml as shown below

```
<manifest version="1.0" type="device">
  <hal format="aidl">
    <name>android.hardware.secure_element</name>
    <version>1</version>
    <fqname>ISecureElement/eSE1</fqname>
  </hal>
 </manifest>
```

6.5.2 NFC DTA Setup

6.5.2.1 NFC DTA Source

Information of NXPAndroidDTA Project repositories in the GitHub are as below:

NFC DTA source can be downloaded from the below link:

https://github.com/NXPNFCProject/NXPAndroidDTA

Copy NFC DTA source to /system/nfc-dta/ folder

6.5.2.2 Build NFC DTA

After building DTA, it generates 64-bit DTA binaries. To install DTA on the android device, ensure that adb is installed on the system and USB cable is connected between the system and the android device.

6.5.2.3 NFC DTA Binaries

1. The generated binary files should be pushed to the target devices as per the below table.

Project	Compiled Files	Location in target device
/system/nfc-dta/	libdta.so libosal.so libdta_jni.so libmwif.so	/system/lib64
/system/nfc-dta/	NxpDTA.apk	/system/app/NxpDTA (Create folder “NxpDTA” under /system/app in target device)

Table 5: DTA specific binaries

After updating the required files, the “NXP Device Test Application” appears in the main menu.

Setting to be done before running DTA APK are as below

1. Switch off the default NFC service option in Settings.

   Settings->Connected Devices >NFC as OFF (Un-ticked) and reboot the device (using ‘adb reboot’).

2. Set Screen time out settings or Stay Awake option should be ticked.

   Screen time out should be updated in the IUT settings to avoid the DTA RF signal loss. Because once the device goes to sleep mode immediately RF will be stopped from device, to avoid this device screen timeout should be increased to 30 minutes or device should powered. The following path can be used for updating the screen timeout setting.

   Main menu -> Settings -> Developer Options -> Stay Awake.

   Settings -> Display -> Sleep -> select 30 minutes.

   Refer DTA setup guide(link) for the usage of DTA application.

6.5.3 Firmware Download

NXP provides precompiled firmware for ARM platforms. NXP also can provide firmware as .c file and it can be compiled as .so file with the platform compiler. Firmware resides at location `/system/vendor/lib64/` on the android target system.

Firmware can be updated when NXP releases an updated version. Steps to update are as follows:

1. Compile the firmware to .so file using the file received in .C file format. If firmware is in .so format then this step can be skipped.
2. Push the firmware file to
   a. `/system/vendor/lib64/libsn100u_fw.so` for SN1xx
   b. `/system/vendor/lib64/lib<snxxx>u_fw.so` for SNxxx
   c. `/system/vendor/lib64/libpn557_fw.so` for PN557on target.

3. Reboot the device or disable and enable NFC service. New firmware will be downloaded during the NFC service boot up
4. Firmware file can be downloaded from below location for all for all chip types with folder name as <chiptype> Ex. Sn220

   https://github.com/NXP/nfc-NXPNFCC_FW/tree/master

Note 1: Firmware download can take up around 10 seconds including host delay.
Note 2: It is strongly recommended not to modify the original firmware download logic of Android NFC.
Note 3: It is recommended that Firmware is always upgraded and not downgraded. If firmware version is required to be downgraded, then please consult NXP.

6.6 Enable SecureElement OMAPI AIDL interface to Vendor Services

To enable communication from KeyMint/Weaver HAL to SecureElement OMAPI AIDL service, following settings Required in the build environment

1. Add overlay in the device configuration folders as below (Example is given as per Dragon board)

   Create new folder **device/linaro/dragonboard/overlay/packages/apps/SecureElement/res/value**

   Create new file "config.xml" with following content:

   ```
   <?xml version="1.0" encoding="utf-8"?>
   <resources>
     <!-- To enable vendor stable service, set this to true and
          make sure its vntf manifest entry is also configured. -->
     <bool name="secure_element_vintf_enabled">true</bool>
   </resources>
   ```

   config.xml file should be present in the "device/linaro/dragonboard/overlay/packages/apps/SecureElement/res/value"

2. Add following line in <ANDORID_ROOT>/vendor/nxp/SNxxx/BoardConfigNfc.mk

   PRODUCT_MANIFEST_FILES += packages/apps/SecureElement/secure_element-service.xml

6.7 Strongbox, Weaver & AuthSecret Hal Integration

NXP Secure Element enables tamper-resistant key storage for Android Apps using StrongBox. StrongBox is an implementation of the Keymint HAL that resides in a hardware security module.

Weaver provides secure storage of secret value (device PIN/Password) that may only be read if the corresponding key has been presented.

This section is not applicable for PN557

6.7.1 Weaver Hal Integration

NXP Weaver applet shall be preinstalled on eSE, please contact NXP CAS for further support.

Below steps shall be followed to enable Weaver Hal in Android.

- Download Weaver Hal source from NXP git hub
  - https://github.com/NXPNFCProject/nfcandroid_weaver_hidlimpl
- Integrate Weaver Hal to AOSP Code (br_android_ncihalx_comm_15)
  - cp -rf nfcandroid_weaver_hidlimpl/weaver AOSP/hardware/nxp/weaver
  - Copy below folder if keymint hal is not integrated, please skip if keymint hal is integrated

    - cp –rf nfcandroid_keymint_hidlimpl/keymint/transport/ AOSP/hardware/nxp/weaver
    - Update include path in AOSP/hardware/nxp/weaver/libese_weaver/Android.bp
- Required sepolicy rules for Weaver HAL in link below
    - https://github.com/NXPNFCProject/NXPNFC_Reference/tree/br_android_ncihalx_comm_15/nxp/SNxxx/sepolicy/weaver
- Add Below permission in "AOSP/vendor/nxp/SNxxx/sepolicy/file_context"
    - "(vendor|system/vendor)/bin/hw/android\.hardware\.weaver-service\.nxp u:object_r:hal_weaver_default_exec:s0"
- Add Weaver HAL Service Pkg in "AOSP/vendor/nxp/SNxxx/Device.mk"
    - PRODUCT_PACKAGES += android.hardware.weaver-service.nxp
    - BOARD_SEPOLICY_DIRS += vendor/$(NXP_VENDOR_DIR)/SNxxx/sepolicy/weaver
- Minimal FW logic shall be enabled in NFC Hal(only required for SN110), Please make sure below configs are set
    - Android makefile: -DNXP_NFC_RECOVERY=TRUE
    - Libnfc-nxp config file option

```
# Enable or Disable the minimal FW recovery support.
# This logic will get enabled on early NFC hal boot.
# Disable NFCC RECOVERY support        0x00
# Enable  NFCC RECOVERY support        0x01
NXP_NFCC_RECOVERY_SUPPORT=0x01
```

    - NFC hal shall be configured as early hal, SE policy changes shall be adopted in SE and NFC hal https://github.com/NXPNFCProject/NXPNFC_Reference/tree/br_android_ncihalx_comm_15/nxp/SNxxx/sepolicy
- Weaver VTS test cases to be executed:

Sl No	Module	Location in AOSP	Steps to Execute
1	VtsHalWeaverTargetTest	hardware/interfaces/weaver/vts/	run vts -a arm64-v8a -m VtsHalWeaverTargetTest

6.7.2 Strongbox Hal(Keymint) Integration

Android Keymint Hal supportd Android Hardware backed Keystore. **Keymint both Hal are available in GitHub, but are mutually exclusive.** Only one service should be integrated in system. Also corresponding NXP Keymint applet shall be preinstalled on eSE. Please contact NXP CAS for info on which Hardware backed keystore is supported for specific chip types.

Keymint uses OMAPI Transport layer. Hence ARA rules need to be updated for keymint HAL to access eSE via OMAPI. Please contact NXP CAS for ARA applet and ARA rules support.

- Get Keymint HAL source from below location
  - https://github.com/NXPNFCProject/nfcandroid_keymint_hidlimpl/tree/br_android_ncihalx_comm_15
- cp -rf nfcandroid_keymint_hidlimpl/keymint AOSP/hardware/nxp/JavacardKeymaster
  - Enable compilation of strongbox HAL source by adding android.hardware.security.keymint-service.strongbox in board config file (vendor/nxp/SNxxx/device-nfc.mk)

    PRODUCT_PACKAGES + android.hardware.security.keymint-service.strongbox
  - Required sepolicy changes as below in vendor/nxp/SNxxx/sepolicy/file_contexts

    #StrongBox Keymint HAL

    + /vendor/bin/hw/android\.hardware\.security\.keymint-service\.strongbox u:object_r:hal_keymint_strongbox_exec:s0
  - vendor/nxp/SNxxx/sepolicy/hal_keymint_strongbox.te shall have changes available in below link
    - https://github.com/NXPNFCProject/NXPNFC_Reference/blob/br_android_ncihalx_comm_15/nxp/SNxxx/sepolicy/hal_keymint_strongbox.te
  - vendor/nxp/SNxxx/config.fs shall have changes available in below link & config.fs file should be added as TARGET_FS_CONFIG_GEN (e.g., TARGET_FS_CONFIG_GEN += vendor/nxp/SNxxx/config.fs) in BoardConfigNfc.mk
    - https://github.com/NXPNFCProject/NXPNFC_Reference/blob/br_android_ncihalx_comm_15/nxp/SNxxx/config.fs

- Please make sure below binaries are present on device:
  - SB HAL binary: /vendor/bin/hw/android.hardware.security.keymint-service.strongbox
  - SB HAL init rc: /vendor/etc/init/android.hardware.security.keymint-service.strongbox.rc
  - Manifest: /vendor/etc/vintf/manifest/android.hardware.security.keymint-service.strongbox.xml
  - Manifest: /vendor/etc/vintf/manifest/android.hardware.security.sharedsecret-service.strongbox.xml
  - uuid mapping xml file : vendor/etc/hal_uuid_map_config.xml

6.7.2.1 Changes required for Keymint VTS, CTS and RKPD test case execution:

- Enable Keystore attest key feature
  - PRODUCT_COPY_FILES += \

  frameworks/native/data/etc/android.hardware.keystore.app_attest_key.xml:system/etc/permissions/android.hardware.keystore.app_attest_key.xml

- **For RKP functionality**
  - PRODUCT_PRODUCT_PROPERTIES += remote_provisioning.hostname=remoteprovisioning.googleapis.com
  - PRODUCT_PRODUCT_PROPERTIES += remote_provisioning.enable_rkpd=true
  - PRODUCT_PRODUCT_PROPERTIES += remote_provisioning.strongbox.rkp_only=true

- **Setting ro.vendor.build.security_patch for Keymint 3.0 VTS Test cases**
  - VENDOR_SECURITY_PATCH = $(PLATFORM_SECURITY_PATCH)

- **For testing deleteAllKeys during factory reset**
  - PRODUCT_PRODUCT_PROPERTIES += ro.crypto.metadata_init_delete_all_keys.enabled=true

- **For Device Id Attestation Testcases**
  These changes are just for reference only and properties values might vary based on the OEM device configurations.
  - PRODUCT_COPY_FILES += \
  frameworks/native/data/etc/android.software.device_id_attestation.xml:$(TARGET_COPY_OUT_VENDOR)/etc/permissions/android.software.device_id_attestation.xml
  - PRODUCT_COPY_FILES += \
  frameworks/native/data/etc/handheld_core_hardware.xml:$(TARGET_COPY_OUT_VENDOR)/etc/permissions/handheld_core_hardware.xml

  - PRODUCT_PRODUCT_PROPERTIES += ro.product.device_for_attestation=$(TARGET_PRODUCT)
  - PRODUCT_PRODUCT_PROPERTIES += ro.product.product_for_attestation=unknown
  - PRODUCT_PRODUCT_PROPERTIES += ro.product.manufacturer_for_attestation=unknown
  - PRODUCT_PRODUCT_PROPERTIES += ro.product.vendor.name=unknown
  - PRODUCT_PRODUCT_PROPERTIES += ro.product.name=unknown
  - PRODUCT_MODEL_FOR_ATTESTATION := $(TARGET_PRODUCT)
  - PRODUCT_BRAND_FOR_ATTESTATION := Android

List of Keymint CTS/VTS test cases to be executed:

Sl No	Module	Location in AOSP	Steps to Execute
1	VtsAidlKeyMintTargetTest	hardware/interfaces/security/keymint/aidl/vts/functional	run vts -a arm64-v8a -m VtsAidlKeyMintTargetTest
2	VtsRemotelyProvisionedComponentTests	hardware/interfaces/security/rkp/aidl/vts/functional	run vts -a arm64-v8a -m VtsHalRemotelyProvisionedComponentTargetTest
3	VtsAidlSharedSecretTargetTest	hardware/interfaces/security/sharedsecret/aidl/vts/functional	run vts -a arm64-v8a -m VtsAidlSharedSecretTargetTest
4	CtsKeystoreTestCases	https://source.android.com/docs/compatibility/cts/downloads	run cts -a arm64-v8a -m CtsKeystoreTestCases
5	CtsKeystoreWycheproofTestCases		run cts -a arm64-v8a -m CtsKeystoreWycheproofTestCases
6	CtsKeystorePerformanceTestCases		run cts -a arm64-v8a -m CtsKeystorePerformanceTestCases

Remote Key Provision test cases to be executed.

Sl No	Package/tool	Location in AOSP	Description	Steps to Execute
1	rkp_factory_extraction_tool	system/security/provisioner/	Extract the RKP CSR in factory to share with Google. This shall be done prior to any test.	• adb push rkp_factory_extraction_tool /vendor/bin/ • adb shell /vendor/bin/rkp_factory_extraction_tool

2	rkpdapp.apk	packages/modules/RemoteKeyProvisioning/app/	RKP service	This always runs in background on boot complete.
3	RkpdAppIntegrationTests.apk	packages/modules/RemoteKeyProvisioning/app/tests/e2e	RKP end to end tests	• adb install -t RkpdAppIntegrationTests.apk • adb shell am instrument -w com.android.rkpdapp.e2etest/androidx.test.runner.AndroidJUnitRunner
4	RkpdAppUnitTests.apk	packages/modules/RemoteKeyProvisioning/app/tests/unit	RKP unit test	• adb install -t RkpdAppUnitTests.apk • adb shell am instrument -w com.android.rkpdapp.unittest/androidx.test.runner.AndroidJUnitRunner
5	RkpdAppStressTests.apk	packages/modules/RemoteKeyProvisioning/app/tests/ stress	RKP stress test	• adb install -t RkpdAppStressTests.apk • adb shell am instrument -w com.android.rkpdapp.stress/androidx.test.runner.AndroidJUnitRunner

6.7.3 AuthSecret Hal Integration

NXP IAR applet shall be preinstalled on eSE, please contact NXP CAS for further support.

Below steps shall be followed to enable AuthSecret Hal in Android.

- Download KeyMint Hal source from NXP git hub
  - https://github.com/NXPNFCProject/nfcandroid_keymint_hidlimpl/tree/br_android_ncihalx_comm_15
- Integrate Weaver Hal to AOSP Code (br_android_ncihalx_comm_15)
  - cp -rf nfcandroid_keymint_hidlimpl/ authsecret AOSP/hardware/nxp/authsecret
- Copy below folder if keymint hal is not integrated, please skip if keymint hal is integrated
  - cp –rf nfcandroid_keymint_hidlimpl/transport/ AOSP/hardware/nxp/authsecret
  - Update include path in AOSP/hardware/nxp/authsecret/Android.bp
- Required sepolicy rules for AuthSecret HAL in link below
  - https://github.com/NXPNFCProject/NXPNFC_Reference/tree/br_android_ncihalx_comm_15/nxp/SNxxx/sepolicy/hal_authsecret_defaul.te
  - https://github.com/NXPNFCProject/NXPNFC_Reference/tree/br_android_ncihalx_comm_15/nxp/SNxxx/sepolicy/secure_element.te
- Add Below permission in “AOSP/vendor/nxp/SNxxx/sepolicy/file_context”

  - "/vendor/bin/hw/android\.hardware\.authsecret-service\.nxp u:object_r:hal_authsecret_default_exec:s0"
- Add AuthSecret HAL Service Pkg in "AOSP/vendor/nxp/SNxxx/Device.mk"
  - PRODUCT_PACKAGES += android.hardware.authsecret-service.nxp
- BOARD_SEPOLICY_DIRS += vendor/$(NXP_VENDOR_DIR)/SNxxx/sepolicy
- vendor/nxp/SNxxx/config.fs shall have changes available in below link & config.fs file should be added as TARGET_FS_CONFIG_GEN (e.g., TARGET_FS_CONFIG_GEN += vendor/nxp/SNxxx/config.fs) in BoardConfigNfc.mk
  - https://github.com/NXPNFCProject/NXPNFC_Reference/blob/br_android_ncihalx_comm_15/nxp/SNxxx/config.fs
- Make sure uuid mapping xml file (vendor/etc/hal_uuid_map_config.xml) UUID mapping for AuthSecret HAL service UID.

- AuthSecret VTS test cases to be executed:

Sl No	**Module**	**Location in AOSP**	**Steps to Execute**
1	VtsHalAuthSecretTargetTest	hardware/interfaces/authsecret/aidl/vts	run vts -a arm64-v8a -m VtsHalAuthSecretTargetTest

6.8 Enable ULPDET feature (Optional)

To enable ULPDET feature please add following property to the "libnfc-nxp.conf"

NXP_DEFAULT_ULPDET_MODE=1

Note: This feature is supported only on SN220 or later chipset. This is not applicable for SN1xx & PN557.

6.9 Power Tracker feature (Optional)

1) To enable Power feature please add following property to the "libnfc-nxp.conf"

   NXP_SYSTEM_POWER_TRACE_POLL_DURATION_SEC=30
2) By Default power tracker specific libraries build as part of NFC HAL compilation
3) During full build add command line argument POWER_TRACKER_FEATURE=true to enable power tracker service.

   Example: make TARGET_NXP_NFC_HW=<SNXXX> POWER_TRACKER_FEATURE=true

For more information related to Power tracker integration & test steps please refer below link.

Link:
https://github.com/NXPNFCProject/nfcandroid_nfc_hidlimpl/blob/br_android_ncihalx_comm_15/snxxx/halimpl/power-tracker/README.txt

Note: This feature is supported only on SN220.

6.10 Adding proprietary HALs to device compatibility matrix

Due to the core_hals only restriction (AOSP main branch) in the framework compatibility matrix all the proprietary hal entries added to the Device Framework Compatibility Matrix.

Link for further reference https://source.android.com/docs/core/architecture/vintf/comp-matrices

In the Board or OEM specific make file set below property and its file path

DEVICE_FRAMEWORK_COMPATIBILITY_MATRIX_FILE :=
vendor/nxp/SNxxx/framework_compatibility_matrix.xml

Below are the contents of framework_compability_matrix.xml shall be added/skipped based on the HALs supported by customer.

vendor/nxp/SNxxx/framework_compatibility_matrix.xml

```
<compatibility-matrix version="1.0" type="framework">
    <hal format="aidl" optional="true">
        <name>android.hardware.security.keymint</name>
        <interface>
            <name>IRemotelyProvisionedComponent</name>
            <instance>strongbox</instance>
        </interface>
    </hal>
    <hal format="hidl" optional="true">
        <name>vendor.nxp.nxpnfc</name>
        <version>2.0</version>
        <interface>
            <name>INxpNfc</name>
            <instance>default</instance>
        </interface>
    </hal>
    <hal format="hidl" optional="true">
        <name>vendor.nxp.nxpese</name>
        <version>1.0</version>
        <interface>
            <name>INxpEse</name>
            <instance>default</instance>
        </interface>
    </hal>
    <hal format="aidl" optional="true">
        <name>android.se.omapi</name>
        <version>1</version>
        <interface>
            <name>ISecureElementService</name>
            <instance>default</instance>
        </interface>
    </hal>
    <hal format="aidl" optional="true">
        <name>vendor.nxp.nxpnfc_aidl</name>
        <version>1</version>
        <interface>
            <name>INxpNfc</name>
            <instance>default</instance>
        </interface>
    </hal>
</compatibility-matrix>
```

6.11 Android15 Observe Mode

6.11.1 Ways to enable and use observed mode & polling loop notifications.

1. Default Lx Debug notification/Polling loop notification shall be enabled with Field info notifications, Type A,B,F, timestamp and signal strength etc.
2. If Card Emulation application opts-in observe mode and either
    - It is brought to foreground.
    - Or chosen as default Wallet application (Introduced in A15)
    - Or Chosen in Tap & pay settings.

    Below sequence is seen

    - The Discovery configuration enables Field detect mode.(Disable Listen, reader allowed)
    - When in reader field would receive polling loop notification filters and then NfcService can bind with matching polling-loop-filter registered in its xml file.
    - Service can call disable observe mode and enable normal discovery with listen/transaction enabled further continue transaction.
    - Once the transaction is finished it shall reenable observe mode back.

Methods to enable Observe Mode

- NfcAdapter.setObserveModeEnabled(Boolean enabled) API
- In apps host-apdu-service xml add android:defaultToObserveMode=”true”
- CardEmulation.setShouldDefaultToObserveModeForService(Component service, Boolean enable) API

6.11.2 Default config options to be enabled for Observe mode

1) By default “nfc_observe_mode_supported” is disabled, Please use overlay as below to enable the feature

/overlay/packages/apps/Nfc/res/values/config.xml

```
<resources>
  <bool name="nfc_observe_mode_supported">true</bool>
</resources>
```

For example:
https://cs.android.com/android/platform/superproject/main/+/main:device/google/sunfish/rro_overlays/NfcOverlay/res/values/config.xml;l=26?q=nfc_observe_mode_supported&ss=android%2Fplatform%2Fsuperproject%2Fmain

2) Below config option shall be enabled

⇨ NXP_EXTENDED_FIELD_DETECT_MODE=0x03

⇨ NXP_OBSERVE_MODE_REQ_NOTIFICATION_TYPE=0x02

in libnfc-nxp.conf

6.11.3 Android vendor Logging support

There is a new config “NXPLOG_AVCNCI_LOGLEVEL” in libnfc-nxp.conf to support Android vendor NCI Messages (Command , Response and Notification) support.

6.12 Android one specific

Android one compliant stack is where only vendor partition(HAL source), config files are from NXP remaining layers(Framework, NFC service, JNI and libnfc source) i.e. system partition is default AOSP source. Following section contains list of changes needed for Android-one specific configuration.

This section is not applicable for PN557

6.12.1 Card emulation through Off-host in Android-one platform

To achieve card emulation functionality through off-host(eSE/UICC) on Android one stack

below changes are needed in libnfc-nxp config file which is different from regular config options

Default AOSP implementation only supports below config options related to routing table management

1) DEFAULT_ISODEP_ROUTE(libnfc-nci.conf)
2) DEFAULT_SYS_CODE_ROUTE(libnfc-nxp.conf)
3) DEFAULT_OFFHOST_ROUTE(libnfc-nxp.conf)

	Value

Route	Android One	Regular
eSE	0xC0	0x01
UICC1	0x80	0x02
UICC2	0x81	0x03
eUICC1	0xC1	0x05
eUICC2	0xC2	0x06

Table 6: NFCEE route Ids

Hence the platforms which are willing to use Card emulation functionality through off-host locations shall updateconfig file with values indicated above

6.12.2 To Support eUICC SMB debug over SMB

To use this interface shall use OMAPI terminal reader and also below changes would be needed in libnfc-nxp.conf config options.

1. NXP_NFC_SE_TERMINAL_NUM to “eSE2”(can be configurable in order starting from eSE1, as eSE1 used for T=1 SPI in MW default release)

2. Make NXP_SE_SMB_TERMINAL_TYPE field as below.(as this is debug feature not enabled by default)
   - 01 for eSE APDU (Apdu Pipe ID : 0x19)
   - 02 for eUICC APDU in SN300(Apdu Pipe ID : 0x27)
   - 03 for eUICC APDU in SN220(Apdu Pipe ID : 0x19)

6.12.3 To configure KeyMint HAL for 2.0 configuration

As default MW KM HAL is configured as KM3.0 shall apply below patch to KM HAL repo

```
diff --git a/keymint/KM200/__Android.bp__ b/keymint/KM200/Android.bp
similarity index 100%
rename from keymint/KM200/__Android.bp__
rename to keymint/KM200/Android.bp
diff --git a/keymint/KM200/res/config.fs b/keymint/KM200/res/config.fs
index f1b7da3..52deba7 100755
--- a/keymint/KM200/res/config.fs
+++ b/keymint/KM200/res/config.fs
```

```
@@ -9,10 +9,10 @@ value:2902
 mode: 0755
 user: AID_VENDOR_NXP_STRONGBOX
 group: AID_SYSTEM
-caps: SYS_ADMIN SYS_NICE
+caps: SYS_ADMIN SYS_NICE WAKE_ALARM

 [vendor/bin/hw/android.hardware.weaver@1.0-service.nxp]
 mode: 0755
 user: AID_VENDOR_NXP_WEAVER
 group: AID_SYSTEM
-caps: SYS_ADMIN SYS_NICE
+caps: SYS_ADMIN SYS_NICE WAKE_ALARM
diff --git a/keymint/KM300/Android.bp b/keymint/KM300/__Android.bp__
similarity index 100%
rename from keymint/KM300/Android.bp
rename to keymint/KM300/__Android.bp__
diff --git a/keymint/transport/Android.bp b/keymint/transport/Android.bp
index 8ddacd6..58ce80a 100644
--- a/keymint/transport/Android.bp
+++ b/keymint/transport/Android.bp
@@ -42,9 +42,6 @@ cc_library {
     srcs: [
         "*.cpp",
     ],
-    defaults: [
-        "keymint_use_latest_hal_aidl_ndk_shared",
-    ],
     cflags: [
         "-DOMAPI_TRANSPORT",
         "-DINTERVAL_TIMER",
```

7. Legal information

Table of Contents