MCU-OTA SBL and SFW User Guide



Contents

Chapter 1 Introduction	5
1.1 Acronyms and abbreviations	5
1.2 About MCU SBL and SFW	6
1.3 Features	7
1.4 Supported MCU boards	8
1.5 SBL and SFW organization	9
1.6 Host system requirements	.10
Chapter 2 Quick start	12
2.1 Windows host	. 12
2.1.1 GCC_ARM	12
2.1.2 IAR IDE	. 13
2.1.3 MDK IDE	. 14
2.2 Linux host	. 15
2.3 SFW	17
Chapter 3 Framework	18
3.1 SCons	18
3.1.1 Overview	. 18
3.1.2 SConscript and SConstruct	18
3.1.3 Basic commands	18
3.2 Kconfig	19
3.3 Host tool	. 19
Chapter 4 MCU ISP	21
4.1 About ISP	. 21
4.2 Features	. 21
4.3 Set ISP timeout	21
4.4 MCU Boot Utility usage	. 22
Chapter 5 Security	24
5.1 BootROM secure boot	.24
5.1.1 High Assurance Boot (HAB)	24
5.1.2 LPC55S69 secure boot	25
5.1.3 Encrypted XIP Boot	25
5.1.3.1 Encrypted XIP boot based on BEE	25
5.1.3.2 Encrypted XIP boot based on OTFAD	26
5.1.3.3 Encrypted XIP boot based on PRINCE	26
5.1.4 Image format	26
5.1.5 Tools	. 26
Chapter 6 Firmware	28
	00

6.1 SFW	28
6.2 Operation to set the OTA flag	
6.2.1 Operation for swap mode OTA	
6.2.2 Operation for the remap mode OTA	

Chapter 7 FOTA	
7 1 Design	32
7.1.1 Single image mode of $\Omega T \Delta$	
7.1.2 Swap mode of OTA	
7 1 3 Reman mode of OTA	
7 2 Local FOTA	40
7.2 Lood 1.0 17 timage OTA	42
7 2 2 SD card OTA	45
7 2 3 1 - Disk OTA	40
7 3 Remote FOTA	48
7 3 1 AWS OTA	48
7.3.1.1 AWS OTA Prerequisites	49
7.3.1.2 Prepare the SBL	
7.3.1.3 Prepare the SFW	
7.3.1.4 Prepare image	
7.3.1.5 Upload new image to S3 bucket	
7.3.1.6 Create OTA Job.	
7.3.1.7 Run the application	
7.3.2 Alivun OTA	
7.3.2.1 Create a testing device	
7.3.2.2 Customize device-side SDK	80
7.3.2.3 Set test equipment information	
7.3.2.4 Modify the cur version for testing	
7.3.2.5 Create OTA task	
7.3.2.6 Run the application	
7.4 Secure FOTA	95
7.4.1 Secure boot demonstration for the platform EVKMIMXRTxxxx	
7.4.1.1 Generating Keys and Certificates	
7.4.1.2 SBL image preparation	
7.4.1.3 Application image preparation	100
7.4.1.4 Program OCOTP (eFuse)	
7.4.1.5 Run SBL and application	102
7.4.2 Encrypted XIP boot demonstration for the platform EVKMXRTxxxx	
7.4.2.1 SBL image preparation	103
7.4.2.2 Application image preparation	105
7.4.2.3 Program KEK (eFuse)	105
7.4.2.4 Run SBL and application	106
7.4.2.5 Application OTA image preparation	
7.4.3 Secure boot demonstration for the platform LPC55S69	107
7.4.3.1 Generating Keys and Certificates	107
7.4.3.2 SBL image preparation	108
7.4.3.3 Application image preparation	109
7.4.3.4 Sign and Program encrypted SBL	110
7.4.3.5 Sign and Program application image	113
7.4.4 Secure boot demonstration for the platform EVKMIMXRTxxx	
7.4.4.1 Generating Keys and Certificates	114
7.4.4.2 SBL image preparation	
7.4.4.3 Application image preparation	119
7.4.4.4 Run signed SBL and SFW	121
7.4.4.5 Run encrypted SBL and SFW	122
7.4.4.6 Program OTP (eFuse)	123
7.4.4.7 Application OTA image preparation	

Chapter 8 Known issues	
Chapter 9 Revision history	126

Chapter 1 Introduction

This document provides a complete description of Secure Bootloader (SBL) features, project framework, quick start, and the various software settings. It describes FOTA in detail, including image programming, switch, revert, signature, encryption, and so on. Security is very important and is described based on NXP MCU SoC secure engines. It includes detailed steps to program images via MCU ISP (UART/USB). Other necessary information can be found in the document for convenient understanding and developing.

Secure Firmware (SFW) was created based on FreeRTOS and developed to implement the complete FOTA process along with SBL. SFW supports obtaining the OTA firmware image by U-Disk, SD card in local, or AWS cloud, Aliyun cloud in the remote. Then, SBL checks, authenticates the OTA firmware image, and boots it up in normal.

1.1 Acronyms and abbreviations

The following table lists the acronyms used in this document.

Term	Description
AES	Advanced Encryption Standard
Aliyun	Alibaba cloud
AWS	Amazon Web Services
BEE	Bus Encryption Engine
СААМ	Cryptographic Accelerator and Assurance Module
CRC	Cyclic Redundancy Check
DCP	Data Co-Processor
FOTA	Firmware Over-The-Air
НАВ	High Assurance Boot
LWIP	Lightweight TCP/IP stack
MCU ISP	MCU In-System programming
MQTT	Message Queuing Telemetry Transport
ОСОТР	On-Chip One Time Programmable
ΟΤΑ	Over-The-Air
OTFAD	On-The-Fly AES Decryption
ОТРМК	One-Time Programmable Master Key

Table 1. Acronyms and abbreviations

Table continues on the next page...

Term	Description
RTOS	Real-time operating system
SB	NXP MCU Secure Binary
SBL	Secure Bootloader
SFW	Secure Firmware
SHA	Secure Hash Algorithms
SKB	Secure Kinetis bootloader
SNVS	Secure Non-Volatile Storage
TRNG	True Random Number Generator
XIP	eXecute In Place

Table 1. Acronyms and abbreviations (continued)

1.2 About MCU SBL and SFW

MCU SBL and SFW are C code projects for secure OTA, they support local OTA via UART, USB, or remote OTA via Ethernet, WIFI, and others, and can provide a complete secure trust chain. The Host Tool makes it convenient to program image via UART/USB interface, sign and encrypt image, manage the eFuse and create an SB/SB2 binary.



Figure 2 shows detailed information about the SBL architecture, and the relationship with Firmware and Host Tool. It includes all the possible modules in the project. When building a specific SBL image for one MCU platform, the project can (should) be easily configured based on SoC features.



1.3 Features

- Support NXP MCU platforms (Table 2 lists the platforms by now), uniform code, and architecture for all platforms.
- Complete OTA secure trust chain, support SoC secure engine for signature and encryption
- · Single image or swap images OTA feature
- · Support SoC remap to reduce Flash erase/program
- · Local OTA: UART, USB communication (SBL)
- · Local OTA: SD card, U-Disk (SFW)
- Remote OTA: AWS, Aliyun (SFW)
- · Minimal MCU system resource requirement
- · Support external or internal flash device
- · Support multiple toolchains and developing environment:
 - Linux host: GCC_ARM
 - Windows host: GCC_ARM, IAR, MDK
 - Conveniently create IAR, MDK project by SCons extended command
- · High and easy scalability via Kconfig mechanism
- · Following the OTA process from open-source MCUboot project

1.4 Supported MCU boards

The following table lists the NXP MCU boards supported by SBL and SFW.

Table 2. Supported NXP MCU boards

Board Architecture Boot		Boot	Security		SBL			SFW OTA			
	Device	Signature	Encryption	ISP	Swap	Remap	U-Disk	SD card	AWS	Aliyun	
evkmimxrt1010	CM7	QSPI Flash	•	•	•		•	•			
evkmimxrt1020	CM7	QSPI Flash	•	•	•	•		•	•	•	•
evkbmimxrt1050	CM7	Hyper Flash	•	•	•	•		•	•	•	•
evkmimxrt1060	CM7	QSPI Flash	•	•	•		•	•	•	•	•
evkmimxrt1064	CM7	QSPI Flash	•	•	•		•	•	•	•	•
evkmimxrt1170	CM7+CM4	QSPI Flash	•	•	•		•	•	•	•	•
evkmimxrt500	CM33+F1	Octal Flash	•	•	•		•	•	•		
evkmimxrt600	CM33+HiFi4	Octal Flash	•	•	•		•	•	•		
lpc55s69	CM33+CM33	Internal Flash	•	•	•	•		•	•		

For detailed platform information, refer to the following documents.

- MIMXRT1010 EVK Board Hardware User's Guide
- MIMXRT1020 EVK Board Hardware User's Guide
- MIMXRT1050 EVK Board Hardware User's Guide
- MIMXRT1060 EVK Board Hardware User's Guide
- MIMXRT1064 EVK Board Hardware User's Guide
- MIMXRT1170 EVK Board Hardware User's Guide
- MIMXRT500 EVK Board Hardware User's Guide
- MIMXRT600 EVK Board Hardware User's Guide
- LPC55S69 EVK Board Hardware User's Guide

1.5 SBL and SFW organization

SBL and SFW projects are constructed with source code, SCons tool, Kconfig scripts, Python scripts, Windows executable files and documents. The layer and description of SBL are showed in the Figure 3 and Table 3. The layer and description of SFW are showed in the Figure 4 and Table 4.



Table 3. SBL source code directories

Directory	Description
boot	Source code of MCUboot partly from open source
component	It includes SDK components and board peripheral drivers, for example, flash IAP and UART drivers
doc	Documents of SBL project
include	Header files of SBL project
ріс	Pictures used by README.md
target	All supported platforms: RT1010, RT1020, RT1050, RT1060, RT1064, RT1170, RT500, RT600, LPC55S69
tools	Tools used to build and configure the project
Kconfig	Script file of menuconfig tool
LICENSE	Apache License
README.md	Introduction to SBL project
SW-Content-Register.txt	Used for license check of SBL project



Table 4	SFW	source	code	directories
	01 11	Source	COUC	un ectories

Directory	Description
component	It includes SDK components and board peripheral drivers, for example, flash IAP and UART drivers
doc	Documents of SFW project
firmware	It includes OTA related source code, for example, FreeRTOS, AWS, Aliyun
include	Header files of SFW project
pic	Pictures used by README.md
target	All supported platforms: RT1010, RT1020, RT1050, RT1060, RT1064, RT1170, RT500, RT600, LPC55S69
tools	Tools used to build and configure the project
Kconfig	Script file of menuconfig tool
LICENSE	Apache License
README.md	Introduction to SFW project
SW-Content-Register.txt	Used for license check of SFW project

1.6 Host system requirements

SBL and SFW projects can be developed in both Linux host and Windows host. The system requirements are as below:

- Linux host
 - Git
 - Python 3.6
 - SCons

- GCC_ARM toolchain
- Library: ncurses5-dev
- Windows host
 - Git bash
 - GCC_ARM toolchain
 - or IAR IDE v8.40
 - or MDK IDE v5.30

Chapter 2 Quick start

This chapter introduces the quick start for SBL and SFW projects. Sections 2.1 and 2.2 introduce the quick start for the SBL project, while section 2.3 introduces the quick start for the SFW project. Use the EVKMIMXRT1170 platform as example.

2.1 Windows host

On the Windows host, three toolchains can be selected to build SBL: GCC_ARM, IAR, and MDK.

2.1.1 GCC_ARM

First, obtain the GCC_ARM toolchain from the Arm or MinGW website and install it to the Windows host.

1. Clone SBL project and checkout to v1.1.0, or download the release package

git clone https://github.com/NXPmicro/sbl.git.

- 2. Enter the directory sbl/target/evkmimxrt1170/.
- 3. Double-click the batch file env.bat.
- 4. Configure the evkmimxrt1170 project.

In env.bat, run the scons --menuconfig command. Then the SBL configuration menu is generated.

Configure SBL project according to a specific platform and specific application. After the configuration is completed, save the configuration and exit the menu.

- 5. Build and download the SBL project.
 - a. Set EXEC_PATH as the gcc toolchain install path in the sblprofile.py file for gcc CROSS_TOOL:



For example, in my windows, the gcc toolchain install path is C:\Program Files (x86)\GNU Tools Arm Embedded\9 2019-q4-major\bin. Set the EXEC PATH as below:

EXEC PATH = r'C:\Program Files (x86)\GNU Arm Embedded Toolchain\9 2020-q2-update\bin'

Alternatively, SBL_EXEC_PATH can be added to the Windows environment variable to cover the EXEC_PATH. For SFW, the environment variable is SFW EXEC PATH.

Control Panel Home	System Properties ×		Environment Variables		
💔 Device Manager	Computer Name Hardware Advanced System Protection Remote				
💔 Remote settings	You must be logged on as an Administrator to make most of these changes.		User variables for nxf55126		
System protection 1	Performance		Variable	Value	
Advanced system settings	Visual effects, processor scheduling, memory usage, and virtual memory		ChocolateyLastPathUpdate	132682175566473930	
			MOZ_PLUGIN_PATH	C:_SOFTWARE_INSTALL\FOXIT READER\plugins\	
	Settings		OneDriveCommercial	C:\Users\nxt55126\OneDrive - NXP	
	User Profiles		Path	C:\Users\nxf55126\AppData\Local\Microsoft\WindowsApps:	
	Desktop settings related to your sign-in		TEMP	C:\Users\nxf55126\AppData\Local\Temp	
		GHz	TMP	C:\Users\nxf55126\AppData\Local\Temp	
	Settings				
	Startup and Recovery			New Edit Delete	e
	System startup, system failure, and debugging information				
System startup, system failure, and debugging information Settings.	-	System variables			
	2		Variable	Value	^
	Environment Variables		ARMGCC_DIR	C:\Program Files (x86)\GNU Tools Arm Embedded\9 2019-q4	
	Environment vendores.		ChocolateyInstall	C:\ProgramData\chocolatey	
			ComSpec	C:\WINDOWS\system32\cmd.exe	
	OK Cancel Apply		myCleanUp	C:\windows\System32\Drivers\DriverData	
	computer description:	_	myDestPath	C:\windows\Temp	
New System	Variable	\times	myDisableADIntegration	No	
	4			<u> </u>	
Variable nar	ne: SBL_EXEC_PATH			New Edit Delete	e
	CARE and a File (ACC) CALL Tests And Excluded 0.0 2010 of excluding				
Variable val	ue: C\Program Files (x86)\Giv0 Tools Arm Embedded\9 2019-q4-major\bin				
Browse Di	irectory Browse File	Cancel		OK Cancel	_
browse br	5 OK	cancer			

b. Build the project

In the env.bat file, use the scons command to build the project.

If successfully built, the sbl.bin image is built in the sbl/target/evkmimxrt1170/build directory.

- c. Download the project
 - i. Use a micro USB cable to connect the EVKMIMXRT1170 board to the computer.
 - ii. Set the board to serial download mode.
 - iii. Use the DapLink drag-n-drop function or other tools to download the sbl.bin image to the board.
 - iv. Set the board to XIP mode
 - v. Reset the board.

2.1.2 IAR IDE

For the IAR toolchain, the steps of quick start are as below:

Step 1~ Step 4 are the same as in section 2.1.1

Step 5: Build and download the SBL project.

1. Create the IAR project for the EVKMIMXRT1170 platform.

In the env.bat file, use the scons --ide=iar command to generate the IAR project.

- 2. Enter the directory: sbl/target/evkmimxrt1170/iar
- 3. Double-click the IAR project file: sbl.eww
- 4. Click the Make button to build the project.
- 5. Use a micro USB cable to connect the EVKMIMXRT1170 board to the computer, set the board to serial download mode. To download the project to the board, click the **Download** button. After the image is successfully downloaded into the board, set the board to XIP mode, then reset the board.

2.1.3 MDK IDE

For the MDK toolchain, the steps of quick start are as below:

Step 1~ Step 4 are the same as in section 2.1.1

Step 5: Build and download the SBL project.

1. Create the MDK project for the EVKMIMXRT1170 platform.

In the env.bat file, use the scons --ide=mdk5 command to generate the MDK project.

- 2. Enter the directory: sbl/target/evkmimxrt1170/mdk.
- 3. Double-click the MDK project file: sbl.uvprojx.
- 4. Click the Build button to build the project.
- 5. Use a micro USB cable to connect the EVKMIMXRT1170 board to the computer, set the board to serial download mode. To download the project to the board, click the **Download** button. After the image is successfully downloaded to the board, set the board to XIP mode, then reset the board.

NOTE



It can be solved in the following ways:

• With MDK version 5.30 (or later)

Configure the project for the Assembler Option: armclang (GUN Syntax).

W Options for Target 'sbl'	×
Device Target Output Listing User C/C++ (AC6) Asm Linker Debug Utilities	
Conditional Assembly Control Symbols Define: DEBUG,STARTUP_INITIALIZE_NONCACHEDATA Undefine: Language / Code Generation Assembler Option: armclang (GNU Syntax) Read-Only Position Independent Read-Wite Position Independent	
Inumb Mode No Warnings No Auto Includes	
Include Paths Misc Controls	
Assembler control string -target=arm-arm-none-eabi mcpu=cortex-m7 mfpu=fpv5d16 mfloat-abi=hard -masm=gnu -c gdwarf-3 -Wa,-defsymMICROLIB=1	
OK Cancel Defaults Help	
Figure 9. Assembler Option	

• With an earlier version than MDK 5.30

- 1. Select the option Assemble by using ArmClang V6
- 2. Configure Misc Controls to -masm=auto

Device 1	Target Output Listing User C/C++ (AC6) Asm Linker Debug Utilities	1
- Conditi Def	ional Assembly Control Symbols	
Undel	ine:	
I⊽ Ast	ad-Only Position Independent Split Load and Store Multiple	
∏ Re □ Th □ No	ad-Write Position Independent umb Mode Warnings I No Auto Includes	
Inclu	de	
Mi Contro Assemb	isc masm=auto ols masm=auto	
cont stri	rol -c ng	
	OK Cancel Defaults Help	
Figure 10. Asm Option		

2.2 Linux host

On Linux host, the steps of quick start are as below:

1. Install SCons

For Ubuntu or Debian, use the command:

```
$ sudo apt-get install scons
```

For RPM-based (Red Hat, SUSE, Fedora ...), use the command:

\$ sudo yum install scons

- 2. Install the GCC_ARM toolchain like gcc-arm-none-eabi-9-2019-q4-major.
- 3. Clone the SBL project and checkout to v1.1.0, or download the release package

\$ git clone https://github.com/NXPmicro/sbl.git

4. Switch to the evkmimxrt1170 directory

\$ cd target/evkmimxrt1170

5. Set EXEC PATH as gcc toolchain install path in sblprofile.py file for gcc CROSS TOOL:



Alternatively, SBL_EXEC_PATH can be added to the Linux environment variable to cover the EXEC_PATH. For SFW, the environment variable is SFW EXEC_PATH.

6. Configure the evkmimxrt1170 project

```
$ scons --menuconfig
```

<pre>.config - MCU-SBL RT1170 Configuration MCU-SBL RT1170 Configuration Arrow keys navigate the menu. <enter> selects submenus> (or empty submenus), Highlighted letters are hotkeys. Pressing <y> includes, <n> excludes, <m> modularizes features. Press <esc><esc> to exit, <?> for Help, for Search. Legend: [*] built-in [] excluded <m> module <> module capable</m></esc></esc></m></n></y></enter></pre>
MCU SBL core> MCU SBL Component> Platform Drivers Config>
L <selects <="" exit=""> < Help > < Save > < Load ></selects>
Figure 12 SBL configuration menu

In this menu, configure the SBL project according to the platform and specific bootloader case, such as enabling the single image function or not. After the configuration is completed, save the configuration and exit the menu.

7. Build the image with GCC_ARM toolchain

\$ scons

The sbl.bin image is built in sbl/target/evkmimxrt1170/build directory.

8. Program the image.

Use a micro USB cable to connect the EVKMIMXRT1170 board to the computer, and program *sbl.bin* by DapLink drag-n-drop or other tools. Set the board to XIP boot mode, and reset to start up the SBL.

2.3 SFW

The architecture of the SFW project is similar to the SBL project, so quick start steps are the same as the SBL introduced in section 2.1 and section 2.2 except for the following steps:

1. Clone the SFW project and checkout to v1.1.0, or download the release package

git clone https://github.com/NXPmicro/sfw.git

2. SFW supports two debug modes: SFW project XIP separately or SFW generates the bin file used with SBL. SFW configures which mode to use via scons --menuconfig.

	onfig - MCU-SFW RT1050 Configuration	
	Arrow keys navigate the menu. <enter> selects submenus> (or empty submenus>). Highlighted letters are hotkeys. Pressing <y> includes, <n> excludes, <m> modularizes features. Press <esc><esc><to <?="" exit,=""> for Help, for Search. Legend: [*] built-in [] excluded <m> module <> module capable</m></to></esc></esc></m></n></y></enter>	
	M:U_SFW_core> McU_SFW_component> Platform Drivers Config>	
F	igure 13. SFW configuration menu	

Then select MCU SFW core, in the MCU SFW core menu, if the Enable sfw standalone xip option is selected, the SFW project will XIP separately. If the Enable sfw standalone xip option is not selected, SFW generates the bin file which is used with SBL.

<pre>config - MCU SFW RT1050 Configuration MCU SFW core</pre>	<pre>(Y> includes, <n> excludes, <m> modularizes ple</m></n></pre>
<pre>[1] Enable sfw standalone xip [*] Enable OTA [*] OTA from cloud OTA cloud select (AWS)> AWS Config> [*] OTA from sdcard [*] OTA from u-disk MCU SFW Flash Map> MCU SFW metadata header></pre>	
Figure 14. SFW Debug mode config	

Chapter 3 Framework

This chapter introduces the build framework of SBL and SFW. SBL and SFW projects are built by SCons software construction tool and are configured by the Kconfig file.

3.1 SCons

This section gives the specifics of the SCons software construction tool.

3.1.1 Overview

SCons is an open-source build system written in Python, similar to GNU Make. However, it uses SConstruct and SConscript files instead of usual Makefile files. These files are also Python scripts and can be written using standard Python syntax. Thus, in SConstruct and SConscript files, the Python standard library can be called to perform various complex processing, not limited to the rules set by the Makefile.

SCons and Python tools should be installed before using them. On Windows host, there is no need to install these SCons and Python because the Env configuration tool in SBL comes with them. On Linux host, Python should be installed by default, and SCons can be installed following the command in section 2.1.

3.1.2 SConscript and SConstruct

SCons uses SConscript and SConstruct files to organize the source code structure.

The following three files exist in each SBL and SFW platform directory: sblconfig.py (for SBL) or sfwconfig.py (for SFW), SConstruct, and SConscript, which controls the compilation of the platform. In general, there is only one SConstruct file in one platform, but there are multiple SConscript files.

The SConscript file can control the addition of source code files and can specify the Group of source code files (similar to the concept of Group in IDEs such as MDK/IAR).

SConscript files also exist in most of the source code folders of SBL and SFW projects. These files are "found" by the SConscript files in the specific platform directory to add the source code corresponding to the macros defined in sblconfig.h or sfwconfig.h into the compiler.

3.1.3 Basic commands

This section introduces some basic SCons commands. On Windows host, these commands are used in the env.bat file of a specific platform in the target directory. On Linux host, these commands are used directly in a specific platform directory.

1. scons

Build the project for a specific platform.

If some source files are modified after executing the command, when the scons command is executed again, SCons performs incremental compilation, and only the modified source files are compiled and linked.

 $2. \ {\tt scons} \ {\tt --menuconfig}$

Call Kconfig file to configure the project and generate the *sblconfig.h* file.

3. scons --ide=xxx

Generate IAR or MDK projects for a specific platform.

Use the scons --ide=iar command to generate one IAR project.

Use the scons --ide=mdk5 command to generate one MDK project.

3.2 Kconfig

The SBL project uses the configuration file sblconfig.h generated by the Kconfig file to configure the system, and the SFW project uses sfwconfig.h. The Kconfig file is the source file for various configuration interfaces.

All configuration tools generate the configuration interface by reading the Kconfig file in the current platform directory. This file is the total entry for all configurations. It contains Kconfig files in other directories. The configuration tool reads each Kconfig file, generates a configuration interface for developers to configure the system, and finally generates the configuration file sblconfig.h of the SBL system and sfwconfig.h of the SFW.

When the scons --menuconfig command is executed with the env tool or with Linux host in the specific platform (target/xxx/) directory, the configuration interface of the SBL and SFW systems appears, as shown on Figure 15 and Figure 16.

.config - MCU-SBL RT1170 Configura Arrow keys navigate the menu. features. Press <esc><esc> to</esc></esc>	ion 	
	MCU SBL component> MCU SBL Component> Platform Drivers Config>	
gure 15. SBL menucor	ıfig menu	
.config = MCU-SFW RT1050 Configurat Arrow keys navigate the menu. features. Press <esc><esc> to</esc></esc>	<pre> MCU-SFW RT1050 Configuration</pre>	
	MCU SFW core>	

Figure 16. SFW menuconfig menu

In this menu, there are three submenus to select. For example, to select the MCU SBL core, use the submenu below:

Platform Drivers Config --->

MC0 3BL COTE	- MCU SBL core -
Arrow keys navigate the men features. Press <esc><esc></esc></esc>	J. <enter> selects submenus> (or empty submenus). Highlighted letters are hotkeys. Pressing <y> includes, <n> excludes, <m> modularizes to exit, <? > for Help, for Search. Legend: [*] built-in [] excluded <m> module <> module capable</m></m></n></y></enter>
	<pre>[]] inable ROM to verify sb] (400) Maximum number of flash sectors per image [] Enable single image function MCU SBL Flash Mao></pre>

In this menu, there are some configurable items, press 'y' to include the item, and press 'n' to exclude the item.

After configuring all items, save the configuration and exit the menu. Then the project can be compiled.

3.3 Host tool

NXP provides various host tools to help with the SBL and SFW developing and testing. Here are three basic tools, for more others, visit NXP official website or contact FAE.

1. MCUXpresso Config Tools

MCUXpresso Config Tools is an integrated suite of configuration tools that help guide users from first evaluation to production software development. These configuration tools allow developers to quickly build a custom SDK and leverage pins, clocks, and peripheral tools to generate initialization C code for custom board support. In the SBL target platform, there is an MCUX_Config.mex file which can be opened by MCUXpresso Config Tools and help to generate specific C code for clocks, pins, and so on. For example: target/evkbmimxrt1050/board/MCUX_Config/ MCUX_Config.mex. For more information, refer to the website.

2. Bootloader Host Application (blhost)

The blhost application is a command-line utility used on the host computer to initiate communication and issue commands to the MCU ISP module over the UART or USB connections. The application only sends one command per invocation. The blhost application supports multi-platforms, including Windows, Linux (X86-based), MACOSX, and Linux (Arm-based). For more information, refer to the website.

3. MCU Boot Utility

NXP-MCU Boot Utility is a GUI tool specially designed for NXP MCU secure boot. Its features correspond to the BootROM function in NXP MCU. Currently, it mainly supports i.MXRT series MCU chips, Compared to NXP official security enablement toolset (OpenSSL, CST, sdphost, blhost, elftosb, BD, MfgTool2), NXP-MCU Boot Utility is a real one-stop tool, a tool that includes all the features of NXP's official security enablement toolset, and what is more, it supports full graphical user interface operation. With NXP-MCU Boot Utility, it is easy to get started with NXP MCU secure boot. The main features of NXP-MCU Boot Utility include :

- · Support both UART and USB-HID serial downloader modes
- Support various user application image file formats (elf/axf/srec/hex/bin)
- · Can validate the range and applicability of user application image
- · Support for converting bare image into bootable image
- Support for loading bootable image into external boot devices
- Support common boot device memory operation (Flash Programmer)

For more information about the MCU boot utility, refer to the website.

Chapter 4 MCU ISP

This section descibes the specifics of the MCU ISP.

4.1 About ISP

The MCU ISP provides flash programming utility that operates over a serial connection on the MCUs. It enables quick and easy programming of MCUs. Host-side command line and GUI tools are available to communicate with the SBL device. Users can utilize host tools to upload/download application code and do manufacturing via the MCU ISP.

4.2 Features

- Supports UART and USB peripheral interfaces.
- Supports NXP blhost tool and NXP-MCUBootUtility GUI tool.
- Automatic detection of the active peripheral.
- User-defined timeout for active peripheral detection.
- Autobaud on UART peripheral.
- · Protection of RAM used by the SBL while it is running.
- Programming Serial NOR Flash.

4.3 Set ISP timeout

In SBL menuconfig, when MCU ISP support is enabled, ISP timeout can be set, the default timeout value is 5 seconds. If SBL target does not receive the ISP command from host within the timeout period, then ISP process is bypassed.

🕒 cmd - sconsmenuconfig	- 🗆 X
<pre><1> cmd - scons -</pre>	Search 🔎 🗄 🔻 🔝 🛨 🗎 🚍
.config - MCU-SBL RT1010 Configurati	on
→ MCU SBL Component → mcu isp suppor mcu i Apport kovs pavigate the mout	sp support
submenus). Highlighted let includes, <n> excludes, <m> modu exit, <? > for Help, for Sear</m></n>	<pre>ters are hotkeys. Pressing <y> larizes features. Press <esc><esc> to ch. Legend: [*] built-in []</esc></esc></y></pre>
[*] Enable mcu isp support (5) Set isp timeout by sec	onds
<pre></pre>	Help > < Save > < Load >
kconfia-mconf.exe*1321:272324 « 180206	1641 1/1 [+] NUM PRI 80x25 (11.10) 25V 274952 100% //
kconny inconnect [52],272324 (100200	
18. SBL menuconfig set timeout	

If the default 5 seconds timeout value is not enough, select this option and edit the timeout value.

_			
G	cmd - sconsmenuconfig	- 🗆 ×	
	1> cmd - scons -	Search 🔎 🗄 🔻 🔝 🖛 🔠 🗮	
	config - MCU-SBL RT1010 Configuration		
→	• MCU SBL Component → mcu isp support ———		
	Set isp timeout by s	econds	
	<tab> key to move from the input field to</tab>	the buttons below it.	
	< Ok > < He	lp >	
kco	onfig-mconf.exe*[32]:272324 « 180206[64] 1/1 [+]	NUM PRI 80x25 (7,12) 25V 274952 100% //	
Figure 19. SBL menuco	onfig set timeout		

4.4 MCU Boot Utility usage

The NXP-MCU Boot Utility GUI tool (v3.3 or later) is recommended as the preferred host tool for ISP downloading. The ones who want to use the blhost command-line tool, should contract NXP. For detailed information, see the steps below:

NXP MCU Bo File Edit View	ot Utility v3.3.0 Tools Window Help			
Target Setup	Run Mode	>	Entry	V Unsigned Image Boot
MCU Series:	USB Detection	>	Master	Image I and in a Commence of
MCU Device:	Image Readback	>	tart / Offset: 0x0	Byte Length (For Read/Er
Boot Device:	Flashloader Resident	>		
	eFuse Group	>	Read	Erase
Boot	FlexSPI XIP Region	>		
Device C	IVT Entry Type	>		
gure 20. Set tool run mode				

1. Open MCUBootUtility, set mode to SBL OTA in menu Tools/Run Mode.

2. Power on the SBL target board (take EVKMIMXRT1010 as example), then connect USB cable to J9. If everything is fine, USB vid/pid is detected. Click the **Connect to SBL ISP** button.

Target Setup	Secure Boot Type DEV Unsigned Image Boot V All-In-One Action
MCU Series: i.MXRT ~	
MCU Device: IMAYET1011	Image Generation Sequence Image Loading Sequence eFuse Operation Utility Boot Device Memory
	Start / Offset: 0x0 Byte Length (For Read/Erase): 0x2000 bin/s19/hex: Browse
FLEXSPI NOR ~	Read Erase Write (Auto Erase) Execute From Start
Boot Device Configuration	·
Device Configuration Data (DCD)	
Port Setup	
⊖ UART	
Vendor ID: 0x15A2 ~	
Product ID: 0x0073 ~	
🛛 🗌 🖂 🗌 One Step	
Connect to SBL ISP	
	Save image/data file to
Device Status	View Bootable Image Clear The Screen Clear Screen Browse
^	
	Log
	Clear
	Save
v	00:00.000

3. Can do the read/erase/write ISP operation now. The image format can be bin/hex/s19

Target Setup		Secure Boot T	ype DEV Un	igned Image	Boot		~	All-In-One	Action	N			
MCU Series:	i.MXRT ~												
		Image Generati	on Sequence I	mage Loading	Sequence e	Fuse Op	peration Utility	Boot Device	Memory				
MCU Device:	i.MXRT1011 ~	Start / Offset:	0x100000	Byte Lengt	h (For Read/Er	rase):	0x2000	bin/s19/h	iex: app_0	x60100000.bir	Brows	e	
Boot Device:	FLEXSPI NOR V		Read	E	ase		Write (Auto Era	ase)	Exec	ute From Star	t		
Boot	Device Configuration	0x60100000	c9 54 Oc	a8 e4 8e 9	6 eb 82 e3	e4 63	7e 4a cd 47		т	c~J.G		^	
	2	0x60100010	81 df d1	69 41 eO 6	l fa 31 d2	13 9b	dl af f4 3f		iA.a.1	?			
Device C	onfiguration Data (DCD)	0x60100020	fe 35 97	95 fd c0 a) f7 le 33	df 24	2e df 44 2a	•	5	\$D*			
		0x60100030	11 83 37	58 98 62 2 00 00 0 5	2 1d 49 50	39 1c	4a ed cb f2	•	.7X	· J			
Port Setup		0x60100040	20 20 20 39 c1 9c	00 00 00 0 44 a4 70 1	1 01 00 23	03 G7 26 52	0b 49 ce 66		j#. . D - 1983	 П.Т.ғ			
r on occup	-	0x60100060	-8 e3 53	44 64 70 1 fe a7 a9 2	1 JU JO 40 3 d0 6c ea	c2 9b	25 f6 95 e2	۰ د		«			
004	ART (USB-HID	0x60100070	04 40 28	54 ch ae 0	fh 1c h4	e1 4d	45 19 88 b6		@(T	ME			
Vendor ID:	0x1542	0x60100080	a0 7e 54	4a 00 08 b	4 9c da c3	a2 f2	4e a1 06 67		ĨIJ	. N g			
	ox isne	0x60100090	94 2c 97	93 63 45 1	9 da 47 92	ef 81	1e 77 5a 1d		,cEG	wZ.			
Product ID:	0x0073 ~	0x601000a0	09 af ed	00 a3 e5 O	3 b8 4c ed	40 c3	c3 71 6d 10		L.@)qm.			
		0x601000b0	77 36 39	f1 39 50 a	a 81 41 el	4f c7	ad 40 76 fc	u	69.9PA.C	@v.			
-		0x601000c0	e5 a7 f8	c2 95 bf 6	e 3e 23 aa	c9 56	d8 63 a7 8f		n>#	V.c			
	🗹 One Step	0x601000d0	13 91 f1	40 4d 31 8	3 1d c3 a7	fa 5c	00 14 e3 d9	•	@M1	\			
_		0x601000e0	ad 1b e3	14 50 58 a	2 ec f4 43	81 5b	d3 3b 66 bb	•	PXC.	[.:f.			
	Reset device	0x601000f0	7a f5 37	38 13 8c b	a 7e bd ed	fc de	19 55 7d 55	z	. 78			~	
		00100100	8C 40 DU	0U C3 D7 9	23 87 01	De e4	et 82 21 81		r#	/.		•	
Device Status		View	Bootable Image	C	lear The Scree	n	Save image	e/data file to			Browse		
FlexSPI N	JOR memory ^												
Page Size = 256	Bytes	Log											
Sector Size = 4.0	KB												
Block Size = 64.	0 KB	Executing D:\g read-memory \commonDat	jithub_repo_jay\ 1611661312 819 iFromBootDevic	NXP-MCUBo 2 D:\github_re :e.dat 9	ptUtility\tools\ po_jay\NXP-N	\blhost2 //CUBoo	2_3\win\blhost - otUtility\tools\bl	t 5242000 -u lhost2_3\win	0x15A2,0x00 \vectors	73-j /	Clea	ar 'e	
		00:01.481											
	\vee	00:01:401											

Chapter 5 Security

This section describes the implemented security feature. Secure Bootloader (SBL) is based on the MCUboot project. SBL keeps the MCUboot legacy RSA and ECDSA signatures. It also provides secure boot based on ROM bootloader and encrypted boot (XIP) based on hardware engine. So images can be signed, encrypted, or signed + encrypted.

MCUboot legacy signing method RSA and ECDSA sign the image by computing hash over the image and then signing that hash. Refer to the MCUboot design document for the details. SBL uses RSA-2048 and ECDSA-P256 by default.

SBL can support ROM secure boot and encrypted boot (XIP) on MIMXRT 4-digit platforms (MIMXRTxxxx), MIMXRT 3-digit platforms (MIMXRTxxx), and LPC55S69.

5.1 BootROM secure boot

The Secure Boot provides the guarantee that unauthorized code cannot be executed on a given product. It involves the device's ROM always executing when coming out of reset. The ROM examines the first user executable image resident in the flash memory to determine the authenticity of that code. If the code is authentic, the control is transferred to it. It establishes a chain of trusted code from the ROM to the user boot code. In this case, BootROM verifies SBL and SBL verifies the application image.

5.1.1 High Assurance Boot (HAB)

NXP MIMXRT 4-digit platforms provide the High Assurance Boot (HAB). It is the high-assurance boot feature in the system boot ROM, that detects and prevents the execution of unauthorized software (malware) during the boot sequence.

HAB uses asymmetric cryptography to sign the image. The bootable image can be signed by the CST tool. The tool generates the CSF data in the binary file format that consists of command sequences and signatures based on a given input command sequence file (CSF file).

The OEM uses a utility provided by NXP to generate a private key and corresponding public key pairs. Then the private key is used to encrypt the digest of the image which OEM wants to release. This encryption generates a unique identifier for the image which is called a signature. The certification with the public key is also attached to the image. Before applying the application, the public key is used to decrypt the signature. The OEMs burn the digest (hash) of the public key to the eFuses of MIMXRT chips. Once burned, it cannot be modified. BootRom can verify the public key by this value.

Below is the bootable image format for HAB.



It does not include Flash Configuration Block for a bootable application. As BootROM has configured flash by reading this field of SBL. All MIMXRT platforms support the RSA public key (1024, 2048, 3072 or 4096). MIMXRT1170 also supports ECDSA signature verification using the ECC public key (P256, P384, P521).

5.1.2 LPC55S69 secure boot

LPC55S69 devices support booting of RSA signed images using RSASSA-PKCS1-v1_5 signature verification. The boot code is signed with RSA private keys. The corresponding RSA public keys used for signature verification are contained in the signed image.

LPC55S69 devices support 2048-bit or 4096-bit RSA keys and X.509 V3 certificates.

Image validation is a two-step process.

- 1. Validate and extracts the Image public Key from the x509 certificate embedded in the image.
- 2. Uses Image_key (Public) to validate image signature.

The BootROM API skboot_authenticate is used to verify the authenticity of an image. Before running the application with this IAP API, the PFR region (CFPA and CMPA) should be configured.

PFR resides at the end of the flash region and can be programmed through ROM in ISP mode.

LPC55S69 stores configuration for the boot ROM in Protected Flash Region (PFR).

	0x20 Image length		
	0x24 Image type		
	0x34Load addr		
	Plain Image		
	Trust zone conf. (Optional)		
Figure 24. Signed image format without MCUboot header			

5.1.3 Encrypted XIP Boot

MIMXRT 4-digit series BootROM supports XIP on the Serial NOR flash device directly with On-the-fly decryption feature (using AES) powered by BEE/OTFAD controller.

The PRINCE is used for real-time encrypt/decrypt operation on LPC55S69 on-chip flash contents.

5.1.3.1 Encrypted XIP boot based on BEE

EVKMIMXRT1060/1064/1050/1020 supports XIP with on-the-fly FlexSPI (QSPI) Flash decryption via Bus Encryption Engine (BEE). The BootROM supports two separate encrypted regions using two separate AES Keys. One encrypted region can be used for SBL, another can be used for application. The image can be encrypted by AES-CTR-128 or AES-ECB-128.

Before doing Encrypted XIP, the BootROM must set the BEE controller correctly, the configurable parameters are organized as Protection Region Descriptor Block (PRDB), the entire PRDB is encrypted using AES-CBC-128 mode with the AES KEY and IV in a Key Info Block (KIB). The KIB is encrypted as Encrypted KIB (EKIB) using the AES key provisioned in eFUSE (SW_GP2) or derived from OTPMK (One-Time Programmable Master Key). The BootROM decrypts KIB using AES ECB-128 mode, up to 2 EKIBs are supported, EKIB0 is located at offset 0x400, and KIB1 is located at offset 0x800.

The image key is AES KEY in the key info. In this solution, SW_GP2 is used as KEK to encrypt the key info.

The tool image_enc.exe can be used to encrypt the image on the host. It is a command-line host program that a customer can use to verify the encrypted procedure.

5.1.3.2 Encrypted XIP boot based on OTFAD

EVKMIMXRT1170/1010 and EVKMIMXRTxxx support XIP with on-the-fly FlexSPI(QSPI)Flash decryption via On-the-Fly AES Decryption Module (OTFAD). The OTFAD supports up to 4 separate encrypted regions using separate AES keys.

Before booting Encrypted XIP, the BootROM must set the OTFAD module correctly, the configurable parameters are organized as KeyBlob. A KeyBlob contains encryption keys for OTFAD, and is always encrypted with a KEK. The KEK can be scrambled for each encryption region. The entire KeyBlob is encrypted using AES-CTR-128 mode. KeyBlob is at offset 0x0 in flash.

The KEK is stored in the OTP/EFUSE block. For EVKMIMXRT1170, the KEK can be restored by the PUF, using the PUF key store as part of the Encrypted XIP image.

In this solution, two KeyBlobs are used. One KeyBlob is used for SBL and another is used for application.

5.1.3.3 Encrypted XIP boot based on PRINCE

LPC55S69 supports on-the-fly encryption/decryption to/from internal flash through PRINCE. Data stored in on-chip internal Flash could be encrypted in real time.

LPC55S69 supports 3 regions that allow multiple code images from independent encryption base to co-exist. Each PRINCE region has a secret-key supplied from on-chip SRAM PUF via secret-bus interface (not SW accessible). PRINCE encryption algorithm does not add latency.

PRINCE keys are 128-bit symmetric key and are sourced from on-chip SRAM PUF via an internal hardware interface, without exposing the key on the system bus.

The PUF controller provides secure key storage without storing the key. It is done by using the digital fingerprint of a device derived from SRAM. Instead of storing the key, a key code is generated, which in combination with the digital fingerprint is used to reconstruct PRINCE keys that are routed to the AES engine or for use by software. These key codes are stored in PFR region of flash.

During the startup, the ROM checks if valid key store data structure is present in PFR. If so, the whole key store data structure is loaded into RAM and ROM issues PUF start procedure. It initializes PUF and reconstructs original keys so that each key can be used if needed.

5.1.4 Image format

Below is the final file format. Application image can be signed, encrypted, or signed + encrypted. If the image is encrypted, the key context should be inserted into the image header part for the MIMXRT 4-digit platform. It is at offset 0x100 in the MCUboot header.



5.1.5 Tools

To use the security feature, prepare the following tools:

- CST Tool (Optional) Code Signing Tool, an application running on a build host to allow manufacturers to sign or encrypt the software for their products incorporating NXP processors.
- elftosb.exe v4.0.0 Combined with CST is used to generate an unsigned/signed bootable image.
- image_enc.exe It is a command-line host program used to encrypt image.

 MCUXpresso Secure Provisioning Tool (SPT) – It is a GUI tool made to simplify the generation and provisioning of bootable executables on NXP MCU platforms.

MCUX Secure Provisioning Tool includes cst.exe, elftosb.exe, and image_enc.exe. Download them from the website.

In the folder sbl\target\evkmimxrtxxxx\secure, there are one-stop scripts to generate signed and encrypted image with these tools. For more details, please see section 7.4.

Chapter 6 Firmware

This section gives the details of the operation of Secure Firmware (SFW).

6.1 SFW

Secure Firmware (SFW) is an instance of application, it was created based on FreeRTOS, and developed to implement the complete FOTA process together with SBL. SFW supports obtaining the OTA firmware image by U-Disk, SD card in local or AWS cloud, Aliyun cloud in remote. Then SBL checks, authenticates the OTA firmware image and boots it up in normal.

SFW follows the same framework of SBL, they have the same building environment, configuring process, and compiling commands. Once familiar with SBL, it is not difficult to use SFW.

For both swap and remap mode, SFW provides a function <code>enable_image()</code> to let the users call after writing new image to the flash. Because of the different flag mechanism of these two modes, SFW uses macros to distinguish them.

6.2 Operation to set the OTA flag

This section gives the details of setting the OTA flag.

6.2.1 Operation for swap mode OTA

For the swap mode OTA, the **image_trailer** of the two slots (**the trailer is in the last 32 bytes of two slots**) is used to judge the swap type and control the rollback. Figure 26 shows the state of the flag. Unset is 0xFF, Set is 0x01.



To initialize the OTA process, after writing the new firmware to slot2, the old firmware that receives the new firmware must write the magic (fixed value, 16 bytes) to the end of slot2 to inform the bootloader that the new firmware has been written to the slot2. After writing down the magic value, reset the board.

The bootloader now detects the OTA type, which is test type. Then the bootloader performs the exchange, during the exchange process, the trailer in slot1 becomes the trailer in slot2, and the position of the trailer in slot2 is cleared. The bootloader goes to slot1 to execute the new firmware. If the new firmware operates normally, it writes the <code>image_ok</code> flag to the slot1 trailer to disable the revert. Otherwise, an error occurs in the new firmware, the <code>image_ok</code> flag is not set, then the watchdog resets the board, the bootloader judges the OTA type, now the type is Revert, exchange the two slots, and clear the trailer position of the slot2, now the trailers of the two slots are all unset.

Note: For a board using swap mode OTA, the firmware must contain two writing flag operations. First, the magic part of the flag is written, this operation must be performed after the new firmware is written. The magic address is 0x2FFFF0. The second operation is writing $image_ok$ flag. After the firmware itself runs the whole task period and during the period everything is OK, the firmware must set the flag. The address of $image_ok$ is 0x2FFFE8. The magic value is as on Figure 27.



6.2.2 Operation for the remap mode OTA

For the remap mode OTA, the remap update flag is used to judge the remap type and control the rollback. The flag is in the fixed offset address of the flash, the offset is 0xFFE0, flag structure occupies 32 bytes of space. Figure 28 shows the state of the flag. Unset is 0xFF, set is 0x01, revert is 0x04.



To initialize the OTA process, after writing the new firmware to slot2 or slot1, the old firmware that receives the new firmware must write the magic (fixed value, 16 bytes, same value as the swap mode) to the position of the magic flag to inform the bootloader that the new firmware has been written to slot1 or slot2. After writing the magic value is done, reset the board.

The bootloader reads the remap update flag to get the current firmware position and judge the OTA type, now the type is test type. If the current position is 0x01 (slot1), set the image_ok part of the flag to 0x04 (means revert) and enable the remap function

and run on slot2 physically. If the new firmware operates normally, it clears the *image_ok* and the magic part of the flag to disable rollback. Otherwise, an error occurs in the new firmware, the new firmware does not clear the flag, then the watchdog resets the board, the bootloader judges the OTA type, now the type is Revert, flip the state of the remap function, and clear the *image_ok* and the magic part of the flag.

Note: For a board using swap mode OTA, the firmware must contain two writing flag operations. First, the magic part of the flag is written, this operation must be performed after the new firmware is written. The magic address is 0xFFFF0. The second operation is writing *image_ok* flag. After the firmware itself runs the whole task period and during the period everything is OK, the firmware must clear these two parts of the flag.

Chapter 7 FOTA

SBL is a secondary bootloader designed for the Firmware Over-The-Air (OTA) application. It stores and manages the OTA image upgrade by reading, authenticating, and writing the OTA image to internal/external memory devices.

It provides the following OTA features:

- · Image swap and revert
- · Image remap and revert
- FlashIAP
- · Security
- ISP

7.1 Design

This section is dedicated to the design of the Firmware Over-The-Air (OTA) application.

7.1.1 Single image mode of OTA

The flash layout for the single image mode of OTA is as below:



The workflow for the single image mode of OTA is as below:



The period for the single image mode of OTA is as below:

	• Single	Image Boot			
	Device Reset		Reset		
	Writing (blhost)	Firmware program			
	Execution (Firmware)			Firmware execution	
	Execution (Bootloader)		Bootloader execution Verify		
	Other Tasks				
t Gantt chart					
Figure 31. Update period of bootloader for single image mode					

7.1.2 Swap mode of OTA

The OTA image itself consists of the image header, image data and image trailer. The image header information is shown in the below table.

Table	5.	Image	header	format
10010	• ••	mage	noador	Torritore

Offset	Width (bytes)	Field	Description
0x00	4	magic	Image header tag Fixed value
0x04	4	load_addr	Point to the load address of the application

Table continues on the next page...

Offset	Width (bytes)	Field	Description
0x08	2	header_size	Size of the image header
0x0a	2	reserved	Reserved for future use
0x0c	4	image_size	The size of the image (not including the Image Header Size)
0x10	4	flags	Not used now
0x14	8	image_version	Image version
0x1c	4	reserved	Reserved for future use

Table 5. Image header format (continued)

The image data are the actual image content, it supports raw binary image format.

The image trailer information is shown in the below table.

l able 6. Image traller forma	Table	6.	Image	trailer	format
-------------------------------	-------	----	-------	---------	--------

Offset	Width (bytes)	Field	Description
0x00	1	copy_done	Flag that the swap done
0x01	7	Pad	Reserved
0x08	1	image_ok	Flag that control the OTA state
0x09	7	pad	Reserved
0x10	16	magic	Image trailer tag Fixed value

The flash layout for swap mode of OTA is as below:



- The SBL resides at the start of the Flash memory.
- The Swap area now is equal to slot1 and 2, this area can be reduced to the size of a sector.

The workflow for swap mode of OTA is as below:



The period for swap mode of OTA is as below:


7.1.3 Remap mode of OTA

The OTA image of remap mode contains the image header, image data. The image header information is the same as swap mode, refer to Table 5.

To control the remap state, before the first slot, the remap update flag structure is set to control the update process. The format is as shown in the table below.

Offset	Width (bytes)	Field	Description
0x00	1	Image_position	The current firmware position
0x01	7	Pad	Reserved
0x08	1	image_ok	Flag that control the OTA state
0x09	7	pad	Reserved
0x10	16	magic	Image trailer magic Fixed value

Table 7. Remap flag format

The flash layout for remap mode of OTA is as below:



The workflow for remap mode of OTA is as below:



The period for remap mode of OTA is as below:



7.2 Local FOTA

For all three OTA modes (single, swap, remap), the default configuration of SBL must verify the signature of the image, so after generating the image file by IAR/MDK/GCC, add a header and signature to the image file. The steps below introduce how to make an available application image.

All SBL target can support U-Disk to update the image, and all the target can support SD Card update except EVKMIMXRT1010.

1. Prepare the image

For single image mode:

Select the 'hello world' demo from SDK as an example. First, to make space for adding a header later, change the linker file. The default application offset address is 0x100000, modify the linker to adapt to this address, the IAR linker of EVKMIMXRT1060 in the picture below can be used for reference.

FOTA

20 */ 27 define symbolram_vector_table_si 28 define symbolram_vector_table_of	<pre>isdefinedsymbol(_ram_vector_table_) ? 0x000004400 : 0; set isdefinedsymbol(_ram_vector_table) ? 0x0000037F : 0;</pre>		<pre>% */ % define symbolram_vector_table_size % define symbolram_vector_table_offs</pre>	<pre></pre>
31 define symbol m_interrupts_start 32 define symbol m_interrupts_end	= 0x60002000; = 0x600023FF;	¢ s	of define symbol m_interrupts_start define symbol m_interrupts_end	= 0x60100400; = 0x601007FF;
33 34 define symbol m_text_start	= 0x60002400;	4	4 define symbol m_text_start	= 0x60100800;
35 define symbol m_text_end 36 37 define symbol m interrupts ram star	= 0x607FFFFF; = 0x20000000;		85 define symbol m_text_end 86 87 define symbol m interrupts ram start	= 0x607FFFF; = 0x20000000;
38 define symbol m_interrupts_ram_end 39	= 0x20000000 +ram_vector_table_offset;		8 define symbol m_interrupts_ram_end 89	= 0x200000000 +ram_vector_table_offset_;
40 define symbol m_data_start 41 define symbol m_data_end 41	<pre>- m_interrupts_ram_start +ram_vector_table_size; = 0x2001FFFF;</pre>		<pre>0 define symbol m_data_start 1 define symbol m_data_end 1 1</pre>	<pre>- m_interrupts_ram_start +ram_vector_table_size; = 0x2001FFFF;</pre>
43 define symbol m_data2_start 44 define symbol m_data2_end	= 0x20200000; = 0x202BFFFF;		define symbol m_data2_start define symbol m_data2_end	= 0x20200000; = 0x202BFFFF;

Remove the XIP header information by set XIP_BOOT_HEADER_ENABLE = 0 in iar project option, and then compile the project and generate a binary file named hello world.bin.

The SFW project already included above changes, so build and use the SFW image directly for single image mode.

For swap mode and remap mode:

Double-click the env.bat in the directory of the corresponding target of SFW. Using command then in the configuration menu, uncheck the Enable sfw standalone xip option, and check the OTA from sdcard and OTA from u-disk options.

	NCU SFW core	
Arrow keys navigate the	e menu. <enter> selects submenus> (or empty submenus). Highlighted letters are hotkeys. Pressing <y> includes, <n> e</n></y></enter>	xcludes
<m> modularizes teature</m>	is. Press <esc><tsc> to exit, <? > for Help, for Search. Legend: [*] built-in [] excluded <m> module < > module capable</m></tsc></esc>	
	[*] Frable STW Standalone XIP	
	[] OTA from cloud	
	[*] OTA from sdcard	
	(*) OIA from u-disk MCHI SEW Elseb Man	
	<pre></pre>	

Following commands described in chapter 2 to generate project, build the project and the images are generated .

2. Generate signature key pair and prepare the bootloader

python imgtool.py keygen -k xxxx_priv.pem -t rsa-2048-sign

The xxxx priv.pem is used to sign the application image.

Generate the public key python imgtool.py getpub -k xxxx_priv.pem -o xxxx_pub.pem -t sign.

The xxxx_pub.c file generated by the above command contains the data structure of the public key, which is an array. It should be compiled with the bootloader to verify the signature. Use it to replace the content in sign-rsa2048-pub.c in sbl\component\secure\mcuboot\ directory.

3. Add signature and header to the image

FOTA

Using the imgtool command to generate the useful application image, type the command in the shell to finish the operation. For single image mode, use hello_world.bin generated in step1. For swap and remap modes, to run the test, use the image that SFW project generated.

And use the command below to generate the first signed image:

```
python imgtool.py sign --key xxxx_priv.pem --align 4 --version "1.1" --header-size 0x400 --
pad-header --slot-size 0x100000 --max-sectors 32 hello world1.bin app1.bin
```

Use another image which differs from the first image and repeat the command:

```
python imgtool.py sign --key xxxx_priv.pem --align 4 --version "1.2" --header-size 0x400 --
pad-header --slot-size 0x100000 --max-sectors 32 hello world2.bin app2.bin
```

Now two signed images are generated. More information about the above sign commands is described below.

- xxxx priv.pem: The private key certificate generated in step 2
- --version: The version format can be major.minor.rev
- hello_world.bin, app.bin: The file path can also be added for hello_world1.bin, app1.bin, hello_world2.bin, and app2.bin in the command.

NOTE

For EVKMIMXRT685, to enable the remap function and the single image function (which have reset operations), use J-link to write the shadow register in advance, the specific steps are as follows:

- 1. Remove the JP2 jumper cap.
- 2. Make sure to connect EVKMIMXRT685 with J-link, then use J-link to write the shadow register in advance, the instruction is as follows: w4 0x40130184 0x314000.
- 3. After shadow register is written, make sure that the MCU cannot be powered down during the entire operation, otherwise the previous operation is invalid. This limitation only exists on EVKMIMXRT685 FlexSPI port b. To ensure that the entire operation is not powered down, after the image is downloaded to the flash, reset the MCU by SW3 instead of power-down reset. In addition, if the DAPLink is used to download images, make JP2(PIN1-2) short, then remove the J-Link probe.

NOTE

For the EVKMIMXRT595 / EVKMIMXRT685 U-disk update function, power on the EXT PWR port. Otherwise, the power supply current is not enough, which causes the U-disk update to fail.

NOTE

The default SDIO interface of the latest EVKMIMXRT595 board is eMMC, not SD card. When using the U disk update function with the latest board, rewrite the board first.

7.2.1 Single image OTA

1. Configure the SBL

The single image function only involves verification of the signature of the image, and there is no erase and write operations on the flash during the bootloader stage of SBL. OTA in this mode must be carried out with MCU ISP.

Check the Enable single image function option in the menuconfig interface of Scons to enable single image mode. At the same time, check the Enable mcu isp support option in the menuconfig interface of Scons to enable the MCU ISP function.

Arrow keys navigat <m> modularizes fe</m>	e the menu. 〈Enter atures. Press 〈Esc	> selects submenus > <esc> to exit, <? > +</esc>	-> (or empty su for Help, fo	ıbmenus). Higi or Search. Legend:	lighted letters ar [*] built-in []	e hotkeys. Pressi excluded <m> modu</m>	ng <y> includes, le < > module c</y>	<n> excludes apable</n>
		[] Enable RO (400) Maximum [*] Enable sin MCU SBL F	I to verify sbl number of flas Igle image funct ash Map>	n sectors per image <mark>ion</mark>				

Figure 40. Enable single image

Annual transmitter for the	- mcu isp support	2.1
Arrow keys navigate the mer <m> modularizes features.</m>	nu. <thter> selects submenus> (or empty submenus). Highlighted letters are hotkeys. Pressing </thter>	excludes, le
	[*] Enable mcu isp support	
	(5) Set isp timeout by seconds	
	ACTIVITY A Fritz A Hala X A Faux X A Land X	
	Contraction (Co	

Generate the project, compile, and download the SBL to the target board. Then, connect the UART or USB port to the PC and reset the board.

2. Run the test

UART connect command type: blhost -p COMx, 115200 -- command

USB connect command type: blhost -u <vid>, <pid> -- command

NOTE

When using UART type to run the test, it must directly connect the UART port to PC by using the TTL2USB module, refer to the schematic of EVK board.

On PC side, put the image which is download and the blhost.exe tool into a folder, and open the terminal under this folder, type the command to connect the blhost and the board within 5 s after the board is powered on. blhost -u - get-property 1 0

After connecting successfully, type the below commands to do the flash related operations.

```
Read: blhost -u -- read-memory [address] [size]
Erase: blhost -u -t [ms] -- flash-erase-region [address] [size]
Write: blhost -u -- write-memory 0x60100000 xxx.bin
```

NOTE

If the size is very large, when using erase operation to erase the flash, add **-t [ms]** in the command to add the timeout. In case the timeout is too short, the mcu ISP may return erase failed.

The picture below shows the entire PC-side blhost update image operation process.



To reset the board, write the new image to the flash slot and then type the reset command. Enter the boot process after 5 s.

	A 3.115200 × hello sbl. Bootloader Version 0.0.1 Bootloader chainload address offset: 0x100000 Reset_Handler address offset: 0x100400 Jumping to the first image slot hello world, appl is running.	
	hello sbl. Bootloader Version 0.0.1 Bootloader chainload address offset: 0x100000 Reset_Handler address offset: 0x100400 Jumping to the first image slot hello world, swap success,now app2 is running.	
Figure 43. Single image updating	g log	

7.2.2 SD card OTA

In this mode, the SD card is used to update the image as reference.

NOTE

For EVKMIMXRT1170, to enable the SD card, connect R136 to the REV C EVK board.

1. Prepare the SBL

To disable single image mode and disable MCU ISP support, disable the 'Enable single image function option and the Enable mcu isp support option in the menuconfig interface of Scons.

Compile the SBL project and download it to the target board.

NOTE

For EVKMIMXRT595, EVKMIMXRT685, EVKMIMXRT1170, and EVKMIMXRT1010, when downloading the sbl.bin file, start from 0x400 offset of the flash.

2. Download the first image

To test the SBL for the first time, download the first image to the board to run the test. Use the MCUBootUtility tool to download app1.bin to the first slot of the board, the default location of the slot1 is from flash_offset+0x100000 to flash_offset+0x200000, the whole slot size is 1 MB.

NOTE

Set the MCUBootUtility to master mode, if MCU ISP function is not enabled.

Target Setup	Secure Boot Type	DEV Unsigned Image Boot	~ A	I-In-One Action		
MCU Series: i.MXRT	~	1		2		
MCU Device: : NAVDT11	Image Generation Seque	ince Image Loading Sequen	ce eFuse Operation Utility Bo	ot Device Memory		
Root Devicer	Start / Offset 0x10000	0 Byte Length (For Ke	ad/ write): 0x2000	C:\Users\nxf55067\D	Browse	
FLEXSPIN	JOR V Read	Erase	Write (Auto Erase)	Execute From Start		
Boot Device Confi	guration		3		^	
Device Configuration	Data (DCD)					
,,						
Port Setup						
O UART 🔘 US	8-HID					
Vendor ID: 0x15A2	~					
Product ID: 0x0073	~					
	ne Step					
Reset devic	2					
			Save image/data file	to		
Device Status	View Bootable Image	Clear The Screen			Browse	
Fuse BOOT_CFGx	· · · · · · · · · · · · · · · · · · ·					
IOMUXC_GPR->GPR16 =	Log					
Page Size = 256 Bytes	Executing C:_small_to read-memory 80530739	ols\NXP-MCUBootUtility-2.3.0 92 1024 C:_small_tools\NXP-I)\tools\blhost2_3\win\blhost -t : MCUBootUtility-2.3.0\tools\blho	242000 - u 0x15A2,0x0073 - j st2_3\win\vectors	^ Clear	
Sector Size = 4 KB	\flexspiNorCfg.dat 9				Save	
Block Size = 64 KB	00:00.850					

- 3. Run the test
 - Insert an SD card to the PC, and copy the app2.bin to the SD card.
 - Rename the filename to newapp.bin.
 - Remove the SD card and insert it to the target board.
 - The debug console then prints the updating log.
 - After the SD card downloading finished, when the log 'sys rst...' printed, remove the SD card from the board, otherwise, it starts a new updating.

FOTA

👔 💉 2. 115200 × 🔁	
<pre>hello sbl. Bootloader Version 0.0.1 Primary image: magic=unset, copy_done=0x3, image_ok=0x3 Scratch: magic=unset, copy_done=0x0, image_ok=0x3 Boot source: primary slot Swap type: none Bootloader chainload address offset: 0x100000 Reset_Handler address offset: 0x100400 Jumping to the first image slot sd card ota app task idle ctr 3 sec. Write 0K flag: off = 0x1fffe0 APP1 is running</pre>	
Card inserted. reading new img: 1.2.0 updating finished Start checking image write magic number offset = 0x2fffe0 sys rst hello sbl. Bootloader Version 0.0.1 Primary image: magic=good, copy_done=0x3, image_ok=0x1 Scratch: magic=unset, copy_done=0x0, image_ok=0x3 Boot source: primary slot Swap type: test Bootloader chainload address offset: 0x100000 Reset_Handler address offset: 0x100400 Jumping to the first image slot sd card ota app task idle ctr 3 sec. Write 0K flag: off = 0x1fffe0 This is APP2	
Figure 45. Swap updating log	

After the app2 log is printed, push the Reset button on the board. To confirm that the updating is successful, the app2 log should be printed.

7.2.3 U-Disk OTA

In this mode, the U-Disk is used to update the image as reference.

1. Prepare the SBL

To disable single image mode and disable MCU ISP support, disable the 'Enable single image function option and the Enable mcu isp support option in the menuconfig interface of Scons.

Compile the SBL project and download it to the target board.

2. Download the first image and the remap image flag

To test the SBL for the first time, download the first image to the board to run the test. Use the MCUBOotUtility tool to download appl.bin to the first slot of the board, the default location of the slot1 is from flash_offset+0x100000 to <code>flash offset+0x200000</code>, the whole slot size is 1 MB.

- 3. Run the test
 - · Connect a U-Disk to the PC.

- Copy the app2.bin to the U-Disk,
- Rename the bin file to newapp.bin
- Disconnect the U-Disk from the PC and connect it to the target board (using an otg cable, if the board has two USB ports, connect it to USB1)
- The debug console prints the updating log
- after the image downloading finished, when the log sys rst... printed, remove the U-Disk from the board, otherwise, it starts a new updating.



After the app2 log is printed, push the **Reset** button on the board. To confirm that the updating is successful, the app2 log should be printed.

7.3 Remote FOTA

This section is dedicated to remote FOTA.

7.3.1 AWS OTA

This section walks through the steps how to perform the AWS OTA firmware update of the board using AWS IoT and the EVKMIMXRT1170 platform as an example. The aim is to demonstrate the testing process.

7.3.1.1 AWS OTA Prerequisites

Create an AWS Account

Create an AWS account: https://console.aws.amazon.com/console/home

- · Create an Amazon S3 Bucket to store the update
- 1. Go to the https://console.aws.amazon.com/s3/
- 2. Choose Create bucket.

	Amazon \$3			
	Account snapshot Storage lens provides visibility into storage usage and activity trends. Learn more	View Sto	rage Lens dashboard	
	Buckets (2) Buckets are containers for data stored in S3. Learn more 🖸	C Copy ARN Empty Delete	Create bucket	
	Q. Find buckets by name		< 1 > 💿	
Figure 47. Create bucket	:			

- 3. Type a bucket name.
- 4. Select Enable for Bucket Versioning.

	Bucket Versioning Versioning is a means of keeping multiple variants of an object in the same bucket. You can use versioning to preserve, retrieve, and restore every version of every object stored in your Amazon S3 bucket. With versioning, you can easily recover from both unintended user actions and application failures. Learn more	
	Bucket Versioning Disable Enable 	
Figure 48. Bucket version	oning	

- 5. Other options keep default configurations and choose Create bucket.
- Create an OTA service role
- 1. Sign in to the https://console.aws.amazon.com/iam/
- 2. From the navigation pane, choose Roles.

	Identity and Access Management (IAM)
	Dashboard
	✓ Access management
	User groups
	Users
	Roles
	Policies
	Identity providers
	Account settings
Figure 49. Roles	

- 3. Choose Create role.
- 4. Under Select type of trusted entity, choose AWS Service.

Create role	1	2 3 4
Select type of trusted entity		
AWS service Another AW EC2, Lambda and others Belonging to you	S account u of 3rd party Web identity Cognito or any OpenID provider SA	AML 2.0 federation sur corporate directory
Allows AWS services to perform actions on your behalf. Learn	more	
Figure 50. AWS service		

5. Choose IoT from the list of AWS services.

0									
Cor	mmon use cases								
Allo	Allows EC2 instances to call AWS services on your behalf.								
Lan Allo	Lambda Allows Lambda functions to call AWS services on your behalf.								
Or	Or select a service to view its use cases								
AF	Pl Gateway	CodeBuild	EMR	IoT SiteWise	RDS				
AV	VS Backup	CodeDeploy	EMR Containers	IoT Things Graph	Redshift				
AV	VS Chatbot	CodeGuru	ElastiCache	KMS	Rekognition				
AV	VS Marketplace	CodeStar Notifications	Elastic Beanstalk	Kinesis	RoboMaker				
AV	VS Support	Comprehend	Elastic Container Registry	Lake Formation	S3				
Ап	nplify	Config	Elastic Container Service	Lambda	SMS				
Ap	pStream 2.0	Connect	Elastic Transcoder	Lex	SNS				
Ap	pSync	DMS	ElasticLoadBalancing	License Manager	SWF				
Ap	plication Auto Scaling	Data Lifecycle Manager	EventBridge	MQ	SageMaker				
Ap	plication Discovery	Data Pipeline	Forecast	Machine Learning	Security Hub				
Se	ervice	DataBrew	GameLift	Macie	Service Catalog				
Ba	itch	DataSync	Global Accelerator	Managed Blockchain	Step Functions				
Bra	aket	DeepLens	Glue	MediaConvert	Storage Gateway				
Bu	idgets	Directory Service	Greengrass	Migration Hub	Systems Manager				
Ce	ertificate Manager	DynamoDB	GuardDuty	Network Firewall	Textract				
Ch	hime	EC2	Health Organizational View	OpsWorks	Transfer				
Clo	oudFormation	EC2 - Fleet	Honeycode	Personalize	Trusted Advisor				
Clo	oudHSM	EC2 Auto Scaling	IAM Access Analyzer	Purchase Orders	VPC				
Clo	oudTrail	EC2 Image Builder	Inspector	QLDB	WorkLink				
Cle	oudWatch Alarms	EKS	IoT	RAM	WorkMail				
	oudWatch Application								

6. Under Select the use case, choose IoT.

Select your use case	
Horizon de Calendaria y para estama en la constance de la	
IoT - Device Defender Mitigation Actions Provides AWS IoT Device Defender write access to IoT and related resources for execution of Miti- Provides AWS IoT Device Defender write access to IoT and related resources for execution of Miti- execution of Mitigation access and access and access and access	gation Actions.
Figure 52. IoT use case	

- 7. Choose Next: Permissions.
- 8. Choose Next: Tags.
- 9. Choose Next: Review.
- 10. Enter a role name and description and then choose to Create role.

Create role	
Review	
Provide the required information below and review t	v this role before you create it.
Role name*	OTARde
	Use alphanumeric and '+=, @' characters. Maximum 64 characters.
Role description	Allows IoT to call AWS services on your behalf.
	©
	Maximum 1000 characters. Use alphanumeric and *+=_,β' characters.
Trusted entities	AWS service: lot.amazonaws.com
Policies	👔 👔 AWSIOTLogging 🗷
	AWSIGTRUIPACIONS 17
Permissions boundary	Permissions boundary is not set
No tago wayo addad	
No rags were aubeu.	
	/
^ Required	Cancel Previous Create role
Figure 53. Create role	

· Add OTA update permissions to the OTA service role

1. In the search box on the IAM console page, enter the name of the role, and then choose it from the list.

lder Mar	ntity and Access nagement (IAM)	Create role Delete role			
- A	AWS Account (QOTAR			
Da	roups	Role name 👻	Description	Trusted entities	
Us	sers C	OTARole	Allows IoT to call AWS services on your behalf.	AWS service: iot	
Pol	olicies				
Ide	entity providers				
Ac	count settings				
Cre	redential report				
a	λ Search IAM				
igure 54. Role	search				

2. Choose Attach policies.

	Roles > OTARole Summary	Delete r	ble					
	Role ARN am ans sam: Oteo 07ARole @ Role dascription Attors for to call AVMS services on your behalf. Edit Instance Profile ARNs @ Path @ Creation time 2019-11-05 13.04 CST Maximum CLUARP session duration 1 hour Edit							
	Permissions policies (2 policies apolicit)							
	Vermissions policies (a policies applied) Attach paties Attach paties Attach paties Add Inline polic							
	Policy name 👻	Policy type 👻						
	AWSIoTThingsRegistration	AWS managed policy						
	AWSIOTLogging	AWS managed policy						
	Show 1 more							
Figure 55. Attach policies								

3. In the Search box, enter AmazonFreeRTOSOTAUpdate, select AmazonFreeRTOSOTAUpdate from the list of filtered policies, and choose Attach policy to attach the policy to the service role.

FOTA

A	dd normisgions to OTAPolo		
A			
			0
			Phone 1 result
F	litter policies V Q Amazon FreeRTOSOTAUpdate		anowing Tresur
	Policy name 👻	Туре	Used as
E	2 > 0 AmazonFreeRTOSOTAUpdate	AWS managed	None
		1	
	Cancel	ach policy	
uro 56 Atta	ch AmazonEreeRTOSOTALIndate policy		
uie Ju. Alla	on Amazoni reek rooo rAopuale policy		

· Add the required IAM permissions to the OTA service role

1. Choose Add inline policy.

Roles > OTARale Summary		Delete role
Policy AmazonFreeRTOSOTAUpdate has been attached for the OTARole.		×
Role ARN am raws sam:Orde/OTARde @ Role description Allows tor to call AVVS services on your behalt Edit Instance Profile ARNs @ Path / Creation film 2019-11-05 13.04 CST Maximum CLIAPI session duration 1 hour Edit		
Permissions Trust relationships Tags Access Advisor Revoke sessions		
Permissions policies (4 policies applied)		1
Attach policies	O Add ini	line policy
Policy name 💌	Policy type 💌	
O AWSIOTThingsRegistration	AWS managed policy	×
 NVSIoTLogging 	AWS managed policy	×
	AWS managed policy	×
¹ AWSIoTRuteActions	AWS managed policy	×
Permissions boundary (not set)		
7. Add inline policy		

- 2. Choose the **JSON** tab.
- 3. Copy and paste the following policy document into the text box:

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
               "iam:GetRole",
               "iam:PassRole"
            ],
            "Resource": "arn:aws:iam::<your_account_id>:role/<your_role_name>"
        }
    ]
}
```

FOTA

4. Choose Review policy.

Create policy A policy defines the AWS permissions that you can assign to a user, group, or role. You can create and edit a	policy in the visual editor and using JSON. Learn more
Visual editor JSON	Import managed policy
<pre> ''Version": "2012-10-17", ''Statement": ['''Action": ['''artion": ['''artion": ['''artion": ['''artion": [''''artion": [''''''''''''''''''''''''''''''''</pre>	
	1
	Cancel Review policy
Review policy	

5. Enter a name for the policy, and then choose Create policy.

Create policy	/			1 2
Review policy Before you create this po	licy, provide the required infor	mation and review this policy.		
Nam	Maximum 128 characters. U	se alphanumeric and '*=, $_{a_{a_{a_{a}}}}$ characters.		
Junia	Q Filter Service •	Access level	Resource	Request condition
	Allow (1 of 203 servi	ces) Show remaining 202 Limited: Read, Write	RoleName string like OTARole>	None
* Required			Can	cel Previous Create policy
Figure 59. Create policy				

- Add the required Amazon S3 permissions to the OTA service role
- 1. Choose Add inline policy.

	,							
		Role ARN	arn:aws:ii	am:)ro	ole/OTARol	2		
		Role description	Allows to	T to call AWS service:	es on your t	half. Edit		
	Insta	nce Profile ARNs	43					
		Path	1					
		Creation time	2019-11-0	05 13:04 CST				
	Maximum CLI/API	session duration	1 hour Ed	it.				
Permissions	Trust relationships	Tags Acces	ss Advisor	Revoke sessions	5			
 Permissi 	ions policies (5 policie	s applied)						1
Attach polic	cies							• Add inline policy
Policy	name 👻						Policy type 👻	
🕨 🤨 AW	StoTThingsRegistration						AWS managed policy	×
🕨 🤨 AW	StoTLogging						AWS managed policy	×
🕨 🥫 Ami	azonFreeRTOSOTAUpdate						AWS managed policy	×
▶ 🤨 AW	StoTRuleActions						AWS managed policy	×
 OT/ 	ARolePolicy						Inline policy	×
 Permissi 	ions boundary (not se	t)						

- 2. Choose the **JSON** tab.
- 3. Copy and paste the following policy document into the box:

```
{
  "Version": "2012-10-17",
   "Statement": [
     {
         "Effect": "Allow",
         "Action": [
            "s3:ListBucketVersions",
            "s3:GetObjectVersion",
         "s3:GetObject",
         "s3:PutObject"
         ],
         "Resource": [
            "arn:aws:s3:::<example-bucket>/*",
            "arn:aws:s3:::<example-bucket>"
        ]
      }
   ]
}
```

This policy grants the OTA service role permission to read Amazon S3 objects. Make sure to replace <example-bucket> with the name of the bucket.

- 4. Choose Review policy.
- 5. Enter a name for the policy, and then choose Create policy.

Create policy			1 2
Review policy Before you create this policy, provide the Name* OTABUO Maximum 12	required information and review this policy.		
Q Filter			
Service	Access level	Resource	Request condition
Allow (1	of 203 services) Show remaining 202		
\$3	Limited: Read, Write	Multiple	None
			,
* Required			Cancel Previous Create policy

- Create an OTA User Policy
- 1. Open the https://console.aws.amazon.com/iam/https://console.aws.amazon.com/iam/console.
- 2. In the navigation pane, choose Users.
- 3. Choose Add user.
- 4. Enter a user name, select Programmatic access type, and then choose Next: Permissions.

Add user		1 2 3 4 5
Set user details	nnce with the same access type and nermissions. Learn more	
Use	name* testota	
	• Add another user	
Select AWS access typ Select how these users will a	e cess AWS. Access keys and autogenerated passwords are provided in the last s	step. Learn more
Accer	s type" Programmatic access Enables an access key ID and secret access key for the AWS A other development tools.	API, CLI, SDK, and
	 AWS Management Console access Enables a password that allows users to sign-in to the AWS Man 	nagement Console.
Figure 62. Add user		

- 5. Choose Next: Tags.
- 6. Choose Next: Review
- 7. Choose Create user.
- 8. After adding the user, choose the IAM user from the list.
- 9. Choose Add permissions.

FOTA

	Permissions	Groups	Tags	Security credentials	Access Advisor	
	 Permission 	ions policies (2 policies applied)				
	Add permiss	sions 🔶	•			
Figure 63. Add permissions						

10. Choose Attach existing policies directly, and then choose Create policy.

	Grant permissions			
	Use IAM policies to grant permissions	s. You can assign an existing policy or c	reate a new one.	
	Add user to group	Copy permissions from existing user	Attach existing policies directly	
	Create policy			
Figure 64. Create policy				

11. Choose the JSON tab, and copy and paste the following policy document into the policy editor:

```
{
  "Version": "2012-10-17",
  "Statement": [
      {
         "Effect": "Allow",
         "Action": [
            "s3:ListBucket",
           "s3:ListAllMyBuckets",
           "s3:CreateBucket",
           "s3:PutBucketVersioning",
            "s3:GetBucketLocation",
            "s3:GetObjectVersion",
            "acm:ImportCertificate",
            "acm:ListCertificates",
            "iot:*",
            "iam:ListRoles",
            "freertos:ListHardwarePlatforms",
            "freertos:DescribeHardwarePlatform"
        ],
         "Resource": "*"
     },
      {
         "Effect": "Allow",
         "Action": [
            "s3:GetObject",
            "s3:PutObject"
         ],
         "Resource": "arn:aws:s3:::<example-bucket>/*"
     },
      {
         "Effect": "Allow",
         "Action": "iam:PassRole",
         "Resource": "arn:aws:iam::<your-account-id>:role/<role-name>"
     }
   ]
}
```

Replace <example-bucket> with the name of the Amazon S3 bucket where the OTA update firmware image is stored.

Replace <your-account-id> with the AWS account ID. The AWS account ID can be found in the upper right of the console. When entering the account ID, remove any dashes (-). Replace <role-name> with the name of the created IAM service role.

- 12. Choose Review policy.
- 13. Enter a name for the new OTA user policy, and then choose Create policy.

Create p	olicy					1 2
Review po	licy					
	Name*	OTAUserPolicy				
		Use alphanumeric and '+=,.@' char	acters. Maximum 128 characters.			
	escription					
		Maximum 1000 characters. Use alpha	anumeric and '+=, .@' characters			
	Summary					
		Q, Filter				
		Service 👻	Access level	Resource		Request condition
		Allow (5 of 203 services) She	ow remaining 198			
		Certificate Manager	Full: List Limited: Write	All resources		None
		FreeRTOS	Limited: List, Read	All resources		None
		IAM	Limited: List, Write	Multiple		None
		юТ	Full access	All resources		None
		S3	Limited: List, Read, Write	Multiple		None
* Required					Cance	el Previous Create policy
Figure 65. OTA user policy	y					

- Windows Pre-Requisites
- 1. OpenSSL
 - a. Install OpenSSL: https://slproweb.com/products/Win32OpenSSL.html
 - b. Modify the system environment variable path to add the OpenSSL bin directory

	Edit environment variable	×
	2	New
	C:\OpenSSL\bin	Edit
	C:\MinGW\msys\1.0\MinGW	Browse
	%USERPROFILE%\AppData\Local\Microsoft\WindowsApps %USERPROFILE%\AppData\Roaming\Python\Python38\Scripts	Delete
		Move Up
		Move Down
		Edit text
	ОК	Cancel
Figure 66. OpenSSL config		

Make sure OpenSSL gets assigned to the OpenSSL executable in the command prompt or terminal environment.

- 2. Install the AWS CLI
 - a. Follow the instructions for AWS CLI version 1 bundler installer https://docs.aws.amazon.com/cli/latest/userguide/ install-cliv1.html
 - b. Go to the IAM console https://console.aws.amazon.com/iam/
 - c. In the navigation pane, choose Users.

Identity and Acce Management (IAN	ss 1)
✓ AWS Account (
Dashboard	
Groups	
Users	
Roles	
Policies	
Identity providers	
Account settings	
Credential report	
Q Search IAM	
Figure 67. Users	

- d. Choose the IAM user account.
- e. Select Security credentials.
- f. In the Access keys section, choose Create access key.
- g. To view the new access key pair, choose **Show**. The secret access key cannot be accessed again after closing this dialog box. The credentials look something like this:

Access key ID: AKIAIOSFODNN7EXAMPLE

Secret access key.wJalrXUtnFEMI/K7MDENG/bPxRfiCYEXAMPLEKEY

h. To download the key pair, choose the Download .csv file. Store the keys in a secure location. The secret access key cannot be accessed again after closing this dialog box. Keep the keys confidential to protect the AWS account and never email them. Do not share them outside user's organization, even if an inquiry appears to come from AWS or Amazon.com. No one who legitimately represents Amazon ever asks someone for the secret key.

	Create access key	x
	Success This is the only time that the secret access keys can be viewed or downloaded. You cannot recover them later. However, you can create new access keys at any time.	
	▲ Download.csv file	-
	Access key ID Secret access key	
	Close	
Figure 68. Download .csv file		

- i. After downloading the .csv file, choose **Close**. Once the access key is generated, the key pair is active by default, and the pair can be used right away.
- j. For general use, the AWS configure command is the fastest way to set up the AWS CLI installation

AWS Access Key ID [None]:	
AWS Secret Access Key [None]:	
Default output format [None]: json	
c:\>	

- 3. Creating a Code-Signing Certificate
 - a. In the working directory, use the following text to create a file named cert_config.txt. Replace *test_signer@amazon.com* with user's email address:

```
[ req ]
prompt = no
distinguished_name = my_dn
[ my_dn ]
commonName = test_signer@amazon.com
[ my_exts ]
keyUsage = digitalSignature
extendedKeyUsage = codeSigning
```

b. Using openSSL command line, create an ECDSA code-signing private key:

openssl genpkey -algorithm EC -pkeyopt ec_paramgen_curve:P-256 -pkeyopt ec param enc:named curve -outform PEM -out ecdsasigner.key

c. Create an ECDSA code-signing certificate:

```
openssl req -new -x509 -config cert_config.txt -extensions my_exts
-nodes -days 365 -key ecdsasigner.key -out ecdsasigner.crt
```

d. Import the code-signing certificate, private key, and certificate chain into AWS Certificate Manager:

```
aws acm import-certificate --certificate file://ecdsasigner.crt
--private-key file://ecdsasigner.key
```

NOTE

This command displays ARN for the certificate. Save it locally to use it while creating the OTA update job.



- Grant access to code signing for AWS IoT
- 1. Sign in to the https://console.aws.amazon.com/iam/
- 2. In the navigation pane, choose Policies.

	Identity and Access Management (IAM)
	✓ AWS Account ()
	Dashboard
	Groups
	Users
	Roles
	Policies
	Identity providers
	Account settings
	Credential report
Figure 71. Policies	

- 3. Choose Create Policy.
- 4. On the **JSON** tab, copy and paste the following JSON document into the policy editor. This policy allows the IAM user access to all code-signing operations:

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
               "signer:*"
        ],
            "Resource": "*"
        }
    ]
}
```

- 5. Choose Review policy.
- 6. Enter a policy name and description, and then choose Create policy.

Create p	policy					1 2
Review po	olicy					
	Name*	OTASigningPolicy	rcters Maximum 128 characters			
	Description	Cite apparent the inter-stage chain				
						k
		Maximum 1000 characters. Use alpha	numeric and '+=, @' characters.			
	summary	Q, Filter				
		Service 👻	Access level	Resource	Request condition	
		Allow (1 of 203 services) Sho	ow remaining 202			
		Signer	Full access	All resources	None	
* Resulted						1
requires					Cancel Previous C	reate policy
Figure 72. OTA signing policy						

7. In the navigation pane, choose Users.

- 8. Choose the IAM user account.
- 9. On the Permissions tab, choose Add permissions.
- 10. Choose Attach existing policies directly, and select the checkbox next to the code-signing policy created before.

Use WA process to grant permosants that adapt an extent polyce or Adapt asset of grant permosants that adapt an extent polyce or Capt permissions from extent polyce	Attach existing policies directly		
Create policy			
Fitter policies ~ Q, 07AS			Showing 3 resu
Policy name +		7ype	Used as
OTASigningPolicy		Customer managed	d None
ServiceQuotesFullAccess		AWS managed	Nane
BerviceQuotasReadOntyAccess		AWS managed	None

- 11. Choose Next: Review.
- 12. Choose Add permissions.
- Create an AWS IoT Thing
- 1. Open the AWS IoT console website https://console.aws.amazon.com/iot/

NOTE
When opening the web, first check if the region is the one which the bucket is.

- 2. In the navigation pane, choose Manage -> Things.
- 3. Choose Create.
- 4. On the Creating AWS IoT things page, choose Create a single thing.

	Creating AWS IoT things				
	An IoT thing is a representation and record of your physical device in the cloud. Any physical device needs a thing record in order to work with AWS IoT. Learn more.				
	Register a single AWS IoT thing Create a thing in your registry	Create a single thing			
	Bulk register many AWS IoT things Create things in your registry for a large number of devices already using AWS IoT, or register devices so they are ready to connect to AWS IoT.	Create many things			
	Cancel	Create a single thing			
Figure 74. Create a single thing					

5. On the Create a thing page, in the Name field, enter a name for the thing, such as MyThing. Choose Next.

CREATE A THING Add your device to the thing reg	istry ^{STEP} U3
This step creates an entry in the thing registry and a th Name myThing	iling shadow for your device.
Apply a type to this thing Using a thing type simplifies device management by pr common set of attributes, which describe the identity a Thing Type No type selected	roviding consistent registry data for things that share a type. Types provide things with a and capabilities of your device, and a description.
Add this thing to a group Adding your thing to a group allows you to manage de Thing Group Groups /	vvices remotely using Jobs. Create group Change
Set searchable thing attributes (optional) Enter a value for one or more of these attributes so the Attribute key Provide an attribute key, e.g. Manufacturer Add another Show thing shadow	at you can search for your things in the registry. Value Provide an attribute value, e.g. Acme-Corporation Clear
Cancel	Back Next

6. On the Add a certificate for the thing page, choose Create certificate. It generates an X.509 certificate and key pair.

	CREATE A THREE Add a certificate for your thing	STEP 2/3
	A certificate is used to authenticate your device's connection to AWS IoT.	
	One-click certificate creation (recommended) This will generate a certificate, public key, and private key using AWS IoT's certificate authority.	Create certificate
	Create with CSR Upload your own certificate signing request (CSR) based on a private key you own.	2 Create with CSR
	Use my certificate Register your CA certificate and use your own certificates for one or many devices.	Get started
	Skip certificate and create thing You will need to add a certificate to your thing later before your device can connect to AWS IoT.	Create thing without certificate
Figure 76. Create certificate		

- 7. On the 'Certificate created!' page, download the public and private keys, certificate, and root certificate authority (CA).
- 8. Choose **Download** for the certificate.
- 9. Choose **Download** for the private key.
- Choose Download for the Amazon root CA. A new webpage is displayed. Choose RSA 2048 bit key: Amazon Root CA1. It opens another webpage with the text of the root CA certificate. Copy this text and paste it into a file named Amazon_Root_CA_1.pem.

Most web browsers save downloaded files into a Downloads directory. Copy these files to a different directory when running the sample applications. Choose **Activate** to activate the X.509 certificate, and then choose **Attach a policy**.

	Certificate creat	ed!				
	Download these files and s after you close this page.	ave them in a safe place. Certificates	can be retrieved at any	r time, but the private and public keys cannot be retr	rieved	
	In order to connect a devi	ce, you need to download the follow	ving:			
	A certificate for this thing	c3c4ff2375.cert.pem	Download			
	A public key	c3c4ff2375.public.key	Download			
	A private key	c3c4ff2375.private.key	Download			
	You also need to downloa A root CA for AWS IoTDow Activate	d a root CA for AWS loT: nload				
	Cancel			Done Attach a po	licy	
Figure 77. Files download						

- 11. On the Add a policy for your thing page, choose Register Thing. After registering the thing, create and attach a new policy to the certificate.
- Create an AWS IoT Policy
- 1. Open the AWS IoT console website: https://console.aws.amazon.com/iot/.
- 2. In the left navigation pane, choose Secure, choose Policies, then choose Create.
- 3. On the Create a policy page, in the Name field, enter a name for the policy (for example, MyIotPolicy). In the Action field, enter iot:*. In the Resource ARN field, enter *. Select the Allow checkbox. It allows all clients to connect to AWS IoT. After entering the information for the policy, choose Create.

	Create a policy	
	Create a policy to define a set of authorized actions. You can authorize actions on one or more resources (things, topics, topic filte more about IOT policies go to the AMS IoT Policies documentation page. Name MytoTpolicy	ers). To learn
	Add statements	
	Policy statements define the types of actions that can be performed by a resource.	Advanced mode
	Resource ARN	
	Chart .	
	Allow Deny	
	Add statement	•
		Create
Figure 78. Create IoT policy		

- · Attach an AWS IoT Policy to a Device Certificate
- 1. Open the AWS IoT console website: https://console.aws.amazon.com/iot/.
- 2. In the left navigation pane, choose Secure, and then choose Certificates.
- 3. In the box for the certificate created, choose ... to open a drop-down menu, and then choose Attach policy.

4. In Attach policies to certificate(s), select the checkbox next to the policy created in the previous step, and then choose **Attach**.

	Attach policies to certificate(s)		
	Policies will be attached to the following certificate(s):		
	Choose one or more policies		
	Q Search policies		
	myiol Policy	View	
		1 policy selected Cancel Attach	
Figure 79. Attach IoT po	licy		

- Attach a Certificate to a Thing
- 1. Open the AWS IoT console website: https://console.aws.amazon.com/iot/.
- 2. In the left navigation pane, choose Secure, and then choose Certificates.
- 3. In the box for the certificate created, choose ... to open a drop-down menu, and then choose Attach thing.
- 4. In Attach things to certificate(s), select the checkbox next to the thing registered, and then choose Attach.

	Attach things to certificate(s)
	Things will be attached to the following certificate(s):
	Choose one or more things
	Q Search things
	1 thing selected Cancel Attach
Figure 80. Attach Thing	

5. To verify that the thing is attached, select the box for the certificate.

∰ AWS IоТ	Certificates
Ø Monter Ø Orband	2a540e234673bd148
C ~~ or operations €i Greenyates Conflicates	
Proton Cis di Act Proton	
pia inter	
Settings (D) Learn	
Figure 81. Select certificate to verify	

6. On the Details page for the certificate, in the left navigation pane, choose Things.

	CERTFEATE ACTIME	Artion -
	Decalls Things Policies Things Non-compliance	
Figure 82. Verify attached thir	g	

7. To verify that the policy is attached, on the Details page for the certificate, in the left navigation pane, choose Policies.

	CERTIFICATE			
	ACTIVE		Actions -	
	Details	Policies		
1	Policies Things Non-compliance	*** myloTPolicy		
Figure 83. Verify attached polic	су			

7.3.1.2 Prepare the SBL

In SBL project, enter the sbl/target/evkmimxrt1170 path.

Disable the Enable single image function option and the Enable mcu isp support option in the menuconfig interface of Scons to disable single image mode and disable MCU ISP support.

Compile the SBL project and download it to the target board.

NOTE

- 1. If the new signature key is used, modify the sign-rsa2048-pub.c
- 2. Programming SBL image by drag-drop of DAPLink may erase the whole flash.

7.3.1.3 Prepare the SFW

To prepare SFW config in an SFW project, follow the steps below:

- 1. Generate aws_clientcredential_keys.h file.
 - a. Enter sfw/firmware/aws_ota/tool path.

- b. Using a web browser, open the CertificateConfigurator.html.
- c. Browse to the Certificate and Key files downloaded from the Thing in '**Create an AWS IoT Thing'** part of section 7.3.1.1. Click on Generate and save <code>aws_clientcredential_keys.h</code>.

Certificate Configuration Tool Amazon FreeRTOS Developer Demos
Provide client certificate and private key PEM files downloaded from the AWS IoT Console.
Certificate PEM file: Choose File
Private Key PEM file: Choose Fileate.pem.key
Generate and save aws_clientcredential_keys.h
Figure 84. Generate aws_clientcredential_keys.h

- d. Replace the sfw/firmware/aws_ota/demos/include/aws_clientcredential_keys.h with the file generated in the c) step.
- 2. Modify the aws_ota_codesigner_certificate.h file.
 - a. Open the ecdsaigner.crt file generated in 'Windows Pre-Requisites' part of section 7.3.1.1 using a text editor.
 - b. Opensfw/firmware/aws_ota/demos/include/aws_ota_codesigner_certificate.h file.
 - c. Copy all the content in ecdsaigner.crt and paste to aws_ota_codesigner_certificate.h in the signingcredentialSIGNING CERTIFICATE PEM.

NOTE Be sure to add ' " ' at the begging of a line and ' \n" ' on every line break as below figure. 白 /* * PEM-encoded code signer certificate * Must include the PEM header and footer: * "-----BEGIN CERTIFICATE-----\n" * "...base64 data...\n" * "----END CERTIFICATE-----\n"; */ static const char signingcredentialSIGNING_CERTIFICATE_PEM[] = "----BEGIN CERTIFICATE----\n" "MIIBYTCCAQegAwIBAgIJAKCX9bIhkilFMAoGCCqGSM49BAMCMCMxITAfBgNVBAMM\n" "GEFsZWphbmRyYS5HdXptYW5AbnhwLmNvbTAeFw0xOTEwMjMxNjMzNDJaFw0yMDEw\n" "MjIxNjMzNDJaMCMxITAfBgNVBAMMGEFsZWphbmRyYS5HdXptYW5AbnhwLmNvbTBZ\n" "MBMGByqGSM49AgEGCCqGSM49AwEHA0IABGUghBD51mF1J3wf4LYsQ2VgOaDpg98G\n" "dNC38FWGS7owT4NC5848JumrD8SonnnXpu77Pt7ShuW39hC3Vdi7z1GjJDAiMAsG\n" "AlUdDwQEAwIHgDATBgNVHSUEDDAKBggrBgEFBQcDAzAKBggqhkjOPQQDAgNIADBF\n" "AiEAr0pNz1aMax4arCPNiW9HYFdQTvUGyZdRLcDrUo1/LQoCIH2U2REoZ59V7r6z\n" "CMLfHA+kWq84IjxDUE20gV60RVvC\n" "-----END CERTIFICATE-----\n"; Figure 85. Certificate format

- 3. Enter sfw/target/evkmimxrt1170 path.
- 4. Double-click the batch file env.bat
- 5. Input the scons --menuconfig command to configure the evkmimxrt1170 project
- 6. Select MCU SFW core.

7. Disable Enable sfw standalone xip option, enable OTA, and select AWS OTA cloud.

	<pre>[] Enable sfw standalone xip [*] Enable OTA [*] OTA from cloud OTA Cloud Select (AWS)> AWS Config> [*] OTA from sdcard [*] OTA from u-disk MCU SFW Flash Map> MCU SFW metadata header></pre>	
Figure 86. SFW config		

8. To enter below config menu, select AWS Config.



- 9. Input MQTT DNS name.
 - a. Open the AWS IoT console website https://console.aws.amazon.com/iot/
 - b. In the navigation pane, choose Manage Things and select the previously created Thing.
 - c. In the navigation pane, choose Interact

THING myThing NO TYPE	Actions -
Details Security Thing Groups	This thing already appears to be connected. Connect a device HTTPS
Billing Groups Shadow Interact	Update your Thing Shadow using this Rest API Endpoint. Learn more
Activity	MQTT
Jure 88. Interact of Thing	

d. Select Set MQTT broker DNS name, copy Rest API Endpoint and paste.

	Set MQTT broker DNS name Please enter a string value. Use the <tab> key to move from the input field to the buttons below it.</tab>	
	.iot.us-east-1.amazonaws.com	
	< Ok > < Help >	
Figure 89. MQTT bro	oker DNS name set	

- e. Press Ok
- 10. Input IoT Thing name.
 - a. Select Set IoT Thing name, copy IoT Thing name, and paste.

FOTA

b. Press Ok

11. Select MCU SFW Component -> secure

12. Enable <code>mbedtls</code> and modify <code>mbedtls</code> config file to <code>aws_mbedtls_config.h</code>, and press Ok

	Set mbedtls config file Please enter a string value. Use the <tab> key to move from the input field to the buttons below it. aws_mbedtls_config.h</tab>	
	< Ok > < Help >	
Figure 91. Modify mbedtls confi	g file	

13. Exit and save the configuration.

	Do you wish to save your new configuration? (Press <esc><esc> to continue kernel configuration.)</esc></esc>	
	< Yes > < No >	I
Figure 92. Save config		

7.3.1.4 Prepare image

- 1. Enter the sfw/target/evkmimxrt1170 path, double click the batch file env.bat
- 2. Input command to generate the iar project.

NOTE To generate the keil or gcc project, refer to section 2.

- 3. Enter the sfw/target/evkmimxrt1170/iar path, open sfw.eww project.
- 4. Go to options, select generate additional output, and choose raw binary format.

5. Check the application version in $sfw/firmware/aws_ota/main_enet.c file$

119	#define	APP_VERSION_MAJOR	0
120	#define	APP_VERSION_MINOR	9
121	#define	APP_VERSION_BUILD	2

Figure 94. Application version

- 6. Click the make button to start building the application.
- 7. If the build is successful, sfw.bin is generated in sfw/target/evkmimxrt1170/iar/build/iar/Exe folder. Change its name according to the application version. Move sfw_092.bin to the sbl/component/secure/mcuboot/scripts folder.
- 8. Change APP_VERSION_BUILD to 3 to build a newer image. Rename the new bin file to sfw_093.bin and also move it to sbl/component/secure/mcuboot/scripts folder.

119	#define	APP	VERSION	MAJOR	0
120	#define	APP	VERSION	MINOR	9
121	#define	APP	VERSION	BUILE	3

Figure 95. New version

9. Sign sfw_092.bin and sfw_093.bin images with RSA using below commands. Then sfw092.bin and sfw093.bin are generated.

```
python imgtool.py sign --key sign-rsa2048-priv.pem --align 4 --version "0.9.2" --header-size
0x400 --pad-header --slot-size 0x100000 --max-sectors 32 sfw_092.bin sfw092.bin
python imgtool.py sign --key sign-rsa2048-priv.pem --align 4 --version "0.9.3" --header-size
0x400 --pad-header --slot-size 0x100000 --max-sectors 32 sfw_093.bin sfw093.bin
```

7.3.1.5 Upload new image to S3 bucket

- 1. Use AWS console to open the S3 service https://console.aws.amazon.com/s3
- 2. Select the previously created bucket.
- 3. Click Upload.
- 4. Drag and drop sfw093.bin.
- 5. Click Upload.

7.3.1.6 Create OTA Job

- 1. Open the AWS IoT console website https://console.aws.amazon.com/iot/
- 2. In the navigation pane, choose Manage Jobs
- 3. Select Create job
- 4. Choose Create FreeRTOS OTA update job, then choose Next.
- 5. In step 1, Input Job name, then choose Next.

	OTA job properties Info	
	Job properties	
	Job name DTAUpdateJob Enter a unique name without spaces. Valid characters: a-z, A-Z, 0-9, - (hyphen), and _ (underscore) Description - optional Enter job description	
	► Tags - optional	
Figure 96. Input job name	Cancet Next	

6. In step 2, choose Thing created in previous section, choose MQTT, and Sign a new file for me

	Devices Info This OTA update job will send your file securely over MQTT or HTTP to the FreeRTOS-based things and/or the thing groups that you choose.
	Devices to update
	Choose things and/or thing groups
	Select the protocol for file transfer Select the protocol that your device supports.
	File Info
	Sign and choose your file Code signing ensures that devices only run code published by trusted authors and that the code hasn't been changed or corrupted since it was signed. You have three options for code signing.
	• Sign a new file for me. Choose a previously signed file. Use my custom signed file.
Figure 97. Thing, proto	col, and file select

7. Under Code signing profile, choose Create new profile.

- 8. Enter a name for the code-signing profile.
 - a. Under Device hardware platform, choose Windows Simulator.

Create a code signing profile	
Profile name	
myOTACodeSigning	
Enter a unique name without spaces. Valid characters: a-z, A-Z, 0-9, and _ (underscore)	
Device hardware platform	
Windows Simulator	7

b. Under Code signing certificate, choose Import new code signing certificate, and browse for the certificate files created with AWS CLI, then choose Import.

Import new code sign certificate	Select an existing certificate
Certificates	
Certificate body	ecdsasigner.crt 344 bytes O Uploaded
Certificate private key	ecdsasigner.key 246 bytes O Uploaded
Certificate chain - option	ial
Import	
Figure 99. Select certificate and key	

c. Under Pathname of code signing certificate on device, type the default path /certificates/authcert.pem, then click Create.

	Path name of code signing certificate on device This is the name and location of the certificate that your FreeRTOS device firmware uses to perform OTA image signature verification.			
	/certificates/authcert.pem			
		Cancel	Create	
Figure 100. Pathname of certificate				

9. Under File, choose Select an existing file, then choose Browse S3 to select sfw093.bin file uploaded in S3.

NOTE

Make sure that the region must be the correct one where the bucket is located. Otherwise the uploaded binary cannot be found.

File	
O Upload a new file.	• Select an existing file.
S3 URL	
Q s3:///sfw093.bin	X View 🖸 Browse S3
Format: s3://bucket/prefix/object.	
Figure 101. Image select	

10. Under Pathname of file on device, type the default path /device/updates.

	Path name of file on device This is the name and location where the file will be stored on the FreeRTOS device.	
	/device/updates	
Figure 102. Pathname of file		

11. Under IAM role, choose the role created in previous steps.

Role Choose a role that grants AWS for access to 53, AWS lot jobs, and AWS Code signing resources. Choose on IAM role Figure 103. Select role		IAM role info
Figure 103. Select role		Role Choose a role that grants AWS toT access to 53, AWS foT jobs, and AWS Code signing resources. Choose an IAM role
	Figure 103. Select role	

- 12. Choose Next.
- 13. Under Job run type, choose first item.

	Job run type Choose how to run this job.
	 Your job will complete after deploying to the devices and groups that you chose (snapshot)
	 Your job will continue to deploy to any devices added to the groups that you chose (continuous)
Figure 104. Job run type	

14. Before clicking Create job, run the application.

7.3.1.7 Run the application

1. Use the MCUBootUtility tool to download the sfw092.bin generated previously to the first slot of the board. The default location of slot1 is the flash_offset+0x100000 to flash_offset+0x200000, the whole slot size is 1 MB.
| NXP MCU Boot Utility v2.3.0 | - 🗆 × |
|---|--|
| File Edit View Tools Window He | · |
| Target Setup | Secure Boot Type DEV Unsigned Image Boot ~ All-In-One Action |
| MCU Series: i.MXRT | Image Generation Sequence Image Loading Sequence eFuse Operation Utility Boot Device Memory 2 |
| MCU Device: i.MXRT117x | Start / Offsee Ox100000 Byte Length (For Read/Write): 0x2000 Bin File: C1/User1/mef5067/0 Benease |
| Boot Device: FLEXSPI NOR | |
| | Read Erase Write (Auto Frase) Execute From Start |
| Boot Device Configuration | 3 |
| Device Configuration Data (DCD | |
| | |
| Port Setup | |
| Ventor ID | |
| UKING UKINZ | |
| Product ID: 0x0073 | <u> </u> |
| | |
| | |
| Reset device | × |
| Device Status | View Bootable Image Clear The Screen Save Image/data file to Browse |
| Fuse BOOT_CFGx | |
| FlexRAM memory | Log |
| FleiSPI NOR memory | Executing C1_small_tools/W0P-MCUBootUhility-2.3.0/tools/bilnost2_3/win/bilnost +t 5242000 -u 0x15A2_00073-j Clear |
| Page Size = 256 Bytes
Sector Size = 4 KB | read-memory 8030/392 (104 Crj_small_tools/NAP-MCUBBotUnity-2.3J/tools/bihost2_s/wm/vectors
Villess/NACf9_d49 S |
| Block Size = 64 KB | · · · · · · · · · · · · · · · · · · · |
| | v 000030 |
| | |
| | |
| Figure 105 Download as one has to | Not1 |
| igule 100. Download siw092.bin to | |
| | |

2. After successfully downloading the image, reset the board. The debug console prints the application log as shown:

	hello sti
	boolf toolder Werston (0.0.1 Reapy Type: none
	The image new in PRIMWEY_SLOT vlot
	bootloader chainGod address offset: 0:100000 Insetimater address offset: 0:100000 Impling to the first image slot Nello Stu ¹
	host ini done Dis example to domonstrate how to use U-Disk to implement ota. Hello worldl. Hello worldl.
	Initializing PW This example to demonstrate how to use 50 card to implement ota. 0 49 Thm 'sci Write certificate
	netto surta. Nello surta. Nello surta.
	Hello world. Hello world. Rease insert a card into board.
	Predest prog in a sense to obstrue. 1334 Tur Sey Letting IP address from DHCP 2 17326 Tur Sey 1 PM4 Address: 192.188.8.106 1 1726 Tur Sey DHCP 08
	4 17331 [Ling_thread] [Ihi70] [DBH0] [17320]
	5 1736 ligt thread [IN0 [IN07][1736] Seconds (IV) initialized. 0 1736 ligt thread [IN0 [IN07][1736] Seconds (IV) initialized the demo. Network type for the d7 17347 [ist_thread] [IN070 [M0TT][17347] MOTT Library succ ssnilly initialized. 0 1734/ ligt thread[IN0 dome version 0.5.2]
	1978-044 (United) (Variant) And Linker 1979-044 (United) (Variant) And Linker 1979-04 (United) (Variant) (Vari
	12 2754 [ist_thread][ub0][NPT1][27540] Anonymous metrics (SRX lampung, SRX wersion) will be pr13 27553 [ist_thread][DM00][NPT1[27553] (NDT connection 22026034, (ONECT operation 22026054) MV 27021 [clt_thread][DM00][NPT1][27131] (NDT connection 22026034, (ONECT operation 22026054) MV 27021 [clt_thread][DM00][NPT1][27131] (NDT connection 22026034, Ist_thread][DM00][NPT1][27131] (NDT connection 220260344, Ist_thread][DM00][NDT connection 220260
	12 72840 [Jot_thread] [Oft_AgentInit_internal] Off Task is Ready. 12 72847 [Off AgentTask] [profMayenTask] Called Madler. Current State [Ready] Event [Start] Nev39 72866 [Off Agent Task] [1809][N0T1] [27805] (MDT connec tion 2020F1aB] SBSSTBE operation sched29 27875 [OTA Agent Task] [INF0][N0T1][27875] [MDT connection 2020F1aB, SBSSTBE operation 2020Bello world]. Hollo world2.
	2/ 2019 (DA Apent Task) [U90][U917][2019] (U917][2019) (U917 comection 2020/030, 9855/HEE operation 2020/02 20106 [DA Apent Task] [U90 [U917][2017] [U17][2017][U17][2017][U17][2017][U17][2017][U17][2017][U17][2017][U17][2017][U17][2017][U17][2017][U17][2017][U17][U17][U17][U17][U17][U17][U17][U
	00 /PR(1) /Paperk 199 27 466 [GTA Agent Lask] [10/9] [POTT] [3653] (POTT connection 2026/368) POTT PABLISH operation qu2/9 2866 [GTA Agent Lask] [10/0] [POTT] [3666] (POTT connec Lion 2026/368, PBLISH operation 2026/368 [GTA Agent Lask] [10/0] [POTT] [2665] (POTT connection 2026/368, PBLISH operation 2026/368) POTT connec Lask] [PortPMApertEask] Calcel Amounter, Current States (Poem [Post 2027) CON (POTT) POTT Connection 2026/368) POTT Connection 2026/368 [POTT] [2666] (POTT) [2666/368] (PO
	37 287/3 (01A Agent Taski [provParseS08bbbbbl] parameter not present: afr_ota 39 28771 (01A Agent Taski [provParseS08bbbbbl] parameter not present: foltosoft 39 28797 (01A Agent Taski [provParseS08bbbbbl] parameter not present: files 40 29702 (01A Agent Taski [provParseS08bbbbbl] parameter not present: files the
	122743 [07:Agent task] [pr/praces308mbol] porameter not present: filesize 222802 [07:Agent task] [pr/praces308mbol] porameter not present: filesize
	V 2000 [DVn Agent Lond] [private Lond] [private Long [private Long Long Long Long Long Long Long Long
	47 2889 [Jot thread] Statter Ready Received: 1_Owand: 0_Drocessad: 0_Dropped: 0 48 2885 [JOT Appet Tank] [provint_Close] Context->92/D026954 49 2885 [JOT Appet Tank] [Johand] Apart
	na zaou zana zana zana zana zana zana zana zan
	Hella worldz. 54 25040 [lot_thread] State: MailingforJob Received: 1 Queued: 0 Processed: 0 Dropped: 0
Figure 106. Application	log

3. When running the application, wait until the message of the OTA State Ready appears in the serial terminal as shown:

54 28610 [iot_thread] State: WaitingFor. Hello world1.	Job Received: 1	Queued: 0	Processed: 0	Dropped: 0
Hello World2. 55 29610 [iot_thread] State: WaitingFor. Hello World1 Hello World2.	Job Received: 1	Queued: 0	Processed: 0	Dropped: 0
56 30610 [int thread] State: WaitingFor. Hello world1. Hello world2.	Job Received: 1	Queued: 0	Processed: 0	Dropped: 0
Figure 107. OTA ready log				

4. At this point, the OTA agent is waiting for an OTA job. Go back to the Create OTA job window and click Create job.

	OTA job configuration Info	
	Job run type Choose how to run this job.	
	 Your job will complete after deploying to the devices and groups that you chose (snapshot) Your job will continue to deploy to any devices added to the groups that you chose (continuous) 	
	Job start rollout configuration - optional Specify how quickly devices will be notified when a pending job starts.	
	Job stop configuration - optional These configurations define when to automatically stop the job. The job stops if a percentage of devices fail the deployment after a minimum number have deployed. The job cancels if any of the criteria are met after the job starts.	
	Job run timeout configuration - optional Specify how long the job will run.	
	Cancel Back Create job	
Figure 108. Create job		

- 5. The process starts, and the outputs are as below.
 - a. Start file transfer

70 4047 [01A Agent Task] [prvParseJS000yModel] Extracted parameter [filesize: d75642] 71 40475 [01A Agent Task] [prvParseJS000yModel] Extracted parameter [filesize: d75642] 73 40475 [01A Agent Task] [prvParseJS000yModel] Extracted parameter [filesize: d75642] 97 40475 [01A Agent Task] [prvParseJS000yModel] Extracted parameter [filesize: d75642] 97 40495 [01A Agent Task] [01A-907] 76 4055 [01A Agent Task] [01A-907] 77 4055 [01A Agent Task] [01A-907] 78 4055 [01A Agent Task] [01A-907] 78 4055 [01A Agent Task] [01A-907] 78 4055 [01A Agent Task] [01A-907] 79 4055 [01A Agent Task] [01A-907] 70 4055 [01A Agent Task] [01A
22 41394 (07A Agent Task) [mv/agestbatellock] [docived file block 4, size 1024 23 (408) (07A Agent Task) [mv/agestbatellock] [docived file block 4, size 1024 94 (413) (07A Agent Task) [mv/agestbatellock] [docation; 100 + 404

b. Received the whole file



c. Check file signature

Figure 111. File signature check log

d. Check image version



Figure 112. Image version check log

e. Write update type

2282 158200 [OTA Agent Task] [OTA-NXP] Write update type write update type = 0x3

2257 150992 [OTA Agent Task] [OTA-NXP] CloseFile 2258 150997 [OTA Agent Task] [OTA-NXP] CheckFileSignature

Figure 113. Write update type log

FOTA

f. Write image trailer

 2283 153208 (01A Agent Task) (01A-M2P) Write image trailer

 Write magic number offset = 0xfff0

 Figure 114. Write image trailer log

 Active neuvrimezze, devrice report

g. Active new image, device reset

2284 158218 [OTA Agent Task] [OTA-NXP] ActivateNewImage 2285 158224 [OTA Agent Task] [OTA-NXP] ResetDevice

Figure 115. Active new image log

h. Running new image

hel Boo Rem	lo shi. Itaader Version 0.0.1 aa type: test
The	: image now in SECONDARY_SLOT slot
800 300 100 100 100 100 100 100 100 100 1	<pre>tlaster chainload address offset: 0:100000 ping to the first image slot to style s example to demonstrate how to use U-Disk to implement ota. to world1. 0 example is demonstrate how to use SD card to implement ota. 0 first Sock Write card firster 0 first Sock Write Card Interdet 10 first Sock Wr</pre>
4 1	7175 [iot_thread] [INF0][DEM0][17175] ·······STARTING DEM0······
5 1 6 1 8 5 8 1 9 1	7191 [iot_thread] [IMF0][INTT][17191] SMK successfully initialized. 7292 [iot_thread] [IMF0][IMF0][IMF0]][SMC:Essfully initialized the demo. Network type for the d7 17193 [iot_thread] [IMF0][MQTT][17193] MQTT library succ folly initialized. 7193 [iot_thread] OfA demo version 0.9.3] 7194 [iot_thread] Creating MDTT Client
Figure 116. Run ne	ew image log

i. Self-test

56 27756 [01A Agent Task] [orv07AgentTask] Called handler. Current State [WaitingforJob] Event [Re57 27765 [01A Agent Task] [prvInSelfTestHandler] prvInSelfT estHandler, platform is in self-test.] 58 27774 [01A Agent Task] [01A-W2P] GelPlatformImageState 59 27781 [01A Agent Task] [01A-W2P] dia status = 0x1 Figure 117. Self-test log

j. Write OK flag





6. After OTA success, push the **Reset** button on the board to confirm that the update is successful, the sfw093.bin log must be printed.

hello shi. Bootloader Version 0.0.1 Reamop type: mone	1
The image now in SECONDARY_SLOT slot	
BootTouder chainLand rediress of fist: 0x100000 Reset inarDier address of fist: 0x100000 Jumping to the first image slot inclusion inclusio	
<pre>5 17312 [dot thread] [INFO][INTT][1731] SDK seccessfully initialized. 6 17311 [int_thread] [INFO][INTT][1731] SDK seccessfully initialized the demo. Network type for the d7 17313 [int_thread] [INFO][NOTT][17313] NOTT Library succ 7 17314 [int_thread] [INFO][INTT][1731] SDK seccessfully initialized the demo. Network type for the d7 17313 [int_thread] [INFO][NOTT][17313] NOTT Library succ 7 17314 [int_thread] Connecting to Torker 8 17344 [int_thread] [INFO][NOTT][1732] [INFT Connection. 8 17344 [INFO][NOTT][1734] [INFT Connection. 8 18 27354 [INT_thread] Connecting to thread. 8 18 27354 [INT_thread] Connecting to thread. 8 19 2734 [INFT [INFO][NOTT][17344] [INFT Connection. 8 19 2734 [INFT [INFT]] [INFT Connection. 8 19 2734 [INFT [INFT]] [INFT Connection. 8 19 2744 [INFO][INFT]] [INFT] [INFT] [INFT Connection. 8 19 2744 [INFO][INFT]] [INFT Connection. 8 20 27404 [INFO][INFT]] [INFT] Connection. 8 20 27404 [INFO][INFT]] [INFT Connection. 8 20 27404 [INFO][INFT]] [INFT] [INFT] [INFT] Connection. 8 20 27404 [INFO][INFT]] [INFT] [I</pre>	
Figure 120. New image log	

7.3.2 Aliyun OTA

This section walks through the steps of how to perform the Aliyun OTA firmware update of the board using Aliyun IoT and use the EVKMIMXRT1064 platform as an example to demonstrate the testing process.

7.3.2.1 Create a testing device

1. Open the link: https://iot.console.aliyun.com/. Register an account to log in to the platform and create the Alibaba Cloud account.

C-J Alibaba Cloud	⊗ Inti - English ∨ Homepage Sign Up
ACCELERATE DIGITALIZATION ALIBABA CLOUD SUMMIL LIVE! June 8, 2021 14:00 - 17:30 (GMT+8) Sign Up Now	Sign In Account: Email Password: Forgot Password? Password Sign In Don't have an account? Register Now
	Sign in as RAM User
Figure 121. Account creation interface	

2. After the login is successful, the page of the Alibaba Cloud IOT platform is displayed, click to enter the "Public Instance" interface.

IoT Platform	企业版实例		运行中	2	即将到期 📀	۹	已到期 💿	
Overview	0	· ·	0	•	0		0	
Documentation 🖾 Value-added Services	全部实例	~						
	📦 公共实例			充值				
	ID: 公共实例 状态: • Activated				¥	购买企业版实 企业版实例提供更 高的 SLA 保障。 购买实例	例 丰富的功能,更好的数据器 快速入门	竊, 更

3. After entering the "**Public Instance**", click "**Create Product**", set the parameters as shown in the figure on the "**New Product**" page, customize a product name, and select the first one by default in the category (to test the OTA function).

	← Public Instance	IoT Platform / Devices / Products / Create Product
	Devices ^	← Create Product(Device TSL)
	Products	Create Product Create Product from Device Center
	Devices	* Product Name
	Groups	You must specify a product name
	Jobs	* Category 💿
	CA Certificate	Standard Category Custom Category
	Rules 🗸	Select a standard category View Features
	Maintenance \lor	* Node Type
	Resource Allocation $~~$ $~~$	ed Device
	Link Analytics 🖾	Networking and Data Format
	Link Visual 🗸 🗸	* Network Connection Method
	Documentation and Tools	Ethernet 🗸
		* Data Type 💿
		ICA Standard Data Format (Alink JSON)
		✓ Checksum Type
		V Authentication Mode
		More
		V Product Description
	🗐 Feedback	OK Cancel
Figure 123. Crea	ate Product	

FOTA

IoT Pla	form / Devices / Products / Create Product	Select Category		
~ ~	Create Product(Device TSL)	All categories	Search by category p	ame or scenario
Cn	ate Product Create Product from Device Center			-
* Prod	uct Name	Category Name	The scene	Actions
	must specify a product name	跳灯照明 💿	公共服务	Select
* Cate	му ©	车辆定位卡 ①	公共服务	Select
92 (S)	ndard Category O Custom Category	水湿检测 💿	公共服务	Select
* Nod	Тура	井蓋移位检測 💿	公共服务	Select
1	Directly Connect el Directo Connect inter a Contervary mobiler inter a Contervary and der inter a Contervary a	垃圾满苗检测 💿	公共服务	Select
		地磁检测器 ③	公共服务	Select
Netw	orking and Data Format	红外体征探测器 ①	公共服务	Select
Eth	enet V	红外对射探测器	公共服务	Select
* Data	Type 💿 Standard Data Format (Alink JSDN) 🧹		< 1 2	3 4 46
√ Ch	acksum Type			
V Au	thentication Mode			

Close

Add equipment to the product after it is successfully created.

OK Cancel

Figure 124. Product Type

Products Image: Created a product. Now you can: Devices Image: Created a product. Now you can: Groups Add Device Jobs Add Device CA Certificate Add	
Devices Add Device Groups A device belongs to a product. Iot Platform issues a device a DeviceName that is unique under the product. A device ran consect to Iot Platform directly or through a gateway. Jobs A device belongs to a product. Iot Platform issues a device a DeviceName that is unique under the product. A device ran consect to Iot Platform directly or through a gateway. CAC certificate Add	
Groups Add Denice Jobs A denice belongs to a product. JoT Platform inners a denice a DeniceName that is unique under the product. A denice can connect to JoT Platform directly or through a gateway. CA Certificate Add	
Jobs CA Certificate Add	
CA Certificate Add	
Rules	
Maintenance V Fulteranison or context 131, for a product, services, and events of the product into the TSL. This ficilitate product management and data interaction in the cloud. After you create a TSL for a product, the devices	under this product aut
Resource Allocation Create TSL	
Link Analytics [5] <	
Link Visual V	

4. Select the product that needs to add a device for testing, and click the blue button of "Add Device" to set the device name.

User Guide

← Public Instance	IoT Platform / Devices / Devices					
Devices ^	Devices					
Products	en_test ^	Total Devices	 Activated 	d Devices 🔞	• Online 🗐	
Devices	All	ent Advanced Search				
Groups	en_test 🗸					
Jobs	Aliyun_OTA_test	DeviceName 🗸 En	ter DeviceName	Q Search by De	evice Tag 🗸	
CA Certificate	smart_washing_machine_1		Product	Node Type	State/Ena	ibled 😰
Rules 🗸	TEST1					
Maintenance 🗸					.~0	
Resource Allocation $~~$						
Link Analytics 🖾	<				No data available.	
Link Visual \sim						
Documentation and Tools						
Figure 126. Add device						
Devices						
en test	Total D	evices 🔞	• Activated Devices 🔞	• Onl	line 🕜	
	0		0	0		
Device List B	atch Management Advanced S	Search				
Add Device B	atch Add DeviceName	✓ Enter DeviceName	Add Davisa		×	
DeviceName/A	lias	Product	Add Device		^	
			 Note: You do not nee specified, Alibaba Clo as DeviceName. 	ed to specify DeviceName. I oud will issue a unique iden	f DeviceName is not tifier under the product	
			Products			
			en_test			
			DeviceName			
			ali_test_2			
			Alias 😰			
			Enter an alias.			
					OK Cancel	
Figure 127. Device inform	ation					

After the equipment is added, as shown in the figure below:

Products en_te ProductKey a1DV	rst View							Device	Secret	****** View		
Device Information	Topic List	TSL Data	Device Shadow	Manage Files	Device Log	Online D	Debug (Groups	Task			
Device Information												
Product Name	en_test		ProductKey	a	a1DVFDaRCeT Copy			Region	China (Shanghai)			
Node Type	Devices		DeviceName	a	ali_test_2 Copy				Authentication Mode	Device Secret		
Alias 💿	Edit	Edit		IP Address						Firmware Version	-	
Created At	May 21, 2021	May 21, 2021, 10:38:17		Activated At -						Last Online		
Current Status	Inactive				Real-time Delay 🔘		Test		Device local log report- ing	Disabled		
More Device Informat	ion											
SDK Language					Version						Module Manufacturer	
Module Information												
Tag Information	🖌 Edit											
Device Tag: No results found	ł.											

- 7.3.2.2 Customize device-side SDK
 - Select "Documents and Tools" in the menu bar on the left. To display the interface as shown in the figure below, select "SDK custom" in "Link SDK", set the SDK version information as shown in the figure below, then click "Start Generation" to generate the device side SDK.

Device Connection and Management	5	LinkVisual	4	Edge Computing
IoT Studio		IoT network management service	L D	Authentication of IoT o
The Internet of Things Link SDK				
Equipment Access SDK				
Link SDK For with TCP/IP Protocol Stack of Equipment Provide Access Ali Cloud Internet of	61	Link IoT Edge SDK For Equipment Provide Security Reliable, Low Delay, Low-Cost, Easy to Expand,		Link WAN SDK For Lora Equipment Provi
Things Platform of Ability		Weak Network Rely on the Local Calculation service See Kit		Ability See Kit Node SE

FOTA

	SDK version @ v4.x
	* Device OS
	FreeRTOS V
	* Equipment hardware form
	 single board system MCU + Communication Module
	 Connect to IoT platform protocol ✓ MQT 3.1.1 ✓
	Data encryption
	 TLS-CA
	О т.срск
	O No encryption
	* Device authentication scheme
	Device key 🗸
	Dynamic registration
	Advanced Capabilities
	Thing model OTA
	Pevice shadow Pevice log Device label
	Bootstrap Service Sub-device management Device diagnostics
	Task management
Figure 130. Customize SDK	

2. Unzip the downloaded SDK package, replace the ali_ca_cert.c in \LinkSDK\external with the corresponding file of the tested project. It is consistent with the certificate in the current project, and it does not need to be replaced in this test.

7.3.2.3 Set test equipment information

1. Under the **Device** option of the Alibaba Cloud IOT platform, save the **DeviceName**, **ProductKey**, and **DeviceSecret** information.

	IoT Platform / Devices / ← ali_test_2 Products en_te Product/cr = 210/0	Devices / De Inactive	vice Details						Devic	eSecret	*******	/iew	
	Device Information	Topic List	TSL Data	Device Shadow	Manage Files	Device Log	Online D	Debug	Groups	Task			
	Device Information Product Name	en_test				ProductKey	a	a1DVFDaR	СеТ Сору				
	Node Type	Devices				DeviceName	a	ali_test_2	Сору				
igure '	131. Triple info	ormation											

2. Open the test project \ sfw \target \evkbmimxrt10XX, double-click the script env.bat, using the command scons --menuconfig to select the Alibaba Cloud project, set the triple information as shown below. Copy the product key/ device name/product secret into it.



User Guide





3. In the script, choose MCU SFW Component -> secure -> enable mbedtls, and change the mbedtls config file to ksdk mbedtls config.h



🖪 cmd - scopsmenuconfig — 🗆 🗸	
\square <1> cmd - scops -	=
config - MCU-SEW RT1050 Configuration	-
→ MCU SFW Component → secure	
Set mbedtls config file	
Please enter a string value. Use the <tab> key to move from the input</tab>	
field to the buttons below it.	
kadk whadtle config b	
ksuk_mbedcis_config.n	
< Ok > < Help >	
kconfig-mconf.exe*[32]:33748 « 180206[64] 1/1 [+] NUM PRI 80x25 (27,12) 25V 40108 100%	\mathbb{Z}

Save the setting and use command scons -ide=iar to generate the iar project.

7.3.2.4 Modify the cur_version for testing

- 1. Enter sfw/target/evkmimxrt1064 path, double-click the batch file env.bat
- 2. Input scons -ide=iar command to generate the iar project.

NOTE Refer to section 2 to generate keil or gcc project.

- 3. Enter sfw/target/evkmimxrt1064/iar path, open sfw.eww project.
- 4. Go to options, select generate additional output, and choose raw binary format.



5. Check the current version in sfw/firmware/Aliyun_ota/fota_basic_demo.c file. Set cur_version to 1.0.0.

342	<pre>cur_version = "1.0.0";</pre>	//更改为所需要更新的版本,	如1.1.0
Figure 138. Set the ver	sion		

- 6. Change bin file to sfw_1064_100.bin and click make button to build the project. The bin file will be generated in /sfw/ target/evkmimxrt1064/iar/build/iar/Exe.
- 7. Change the cur_version to "1.4.0", and change bin file name to ${\tt sfw_1064_140.bin}$. Generate it.
- 8. Copy the generated bin files to sbl/component/secure/mcuboot/scripts folder.
- Sign sfw_1064_100.bin and sfw_1064_140.bin images with RSA using the command below. Then 1064_ali_100.bin and 1064_ali_140.bin are generated.

```
python imgtool.py sign --key sign-rsa2048-priv.pem --align 4 --version "1.0.0" --header-size
0x400 --pad-header --slot-size 0x100000 --max-sectors 32 sfw_1064_100.bin 1064_ali_100.bin
python imgtool.py sign --key sign-rsa2048-priv.pem --align 4 --version "1.4.0" --header-size
0x400 --pad-header --slot-size 0x100000 --max-sectors 32 sfw_1064_140.bin 1064_ali_140.bin
```

7.3.2.5 Create OTA task

1. Under the "Maintenance" directory on the left, select "OTA Update".

	← Public Instance	
	Devices	~
	Rules	\sim
	Maintenance	
	Real-time Monitoring	
	Dashboard	
	Online Debug	
	Device Simulation	
	Device Log	
	OTA Update	
	Remote Config	
	Alert Center	
Figure 139. OTA upgrade		

2. Click the "Add Update Package" button to add the upgrade package, enter the upgrade package name, select the corresponding product module, and "Update package version" should correspond to the version of the uploaded bin file, and then click upload to confirm. Now the 1.4.0 version number is used.

	Add Update Package	×
	* Types of Update Packages 💿	
	Full Differential	
	* Update Package Name 💿	
	sfw_1064_140	
	* Product	
	en_test	~
	* Update Package Module	
	default	\vee
	+ Add Module	
	* Update Package Version 💿	
	1.4.0	
	* Signature Algorithm	
	MD5	\vee
	* Select Update Package 💿	
	Re-upload	
	✓ 1064_ali_140.bin (266.21 KB)	×
	* Verify Update Package? 💿	
	● Yes ○ No	
	Update Package Description	
	Please enter upgrade package description	
	0/1	024
	✓ Security Check Service of Update Package	
	OK Canc	el
Figure 140. Upgrade package ir	formation	

7.3.2.6 Run the application

1. Using the MCUBootUtility tool to download the 1064_ali_100.bin generated previously to the first slot of the board. The default location of slot1 is the flash_offset+0x100000 to flash_offset+0x200000, the whole slot size is 1MB.

Target Setup		Secure Boot Type DEV Unsigned Image Boot DEV Unsigned Image Boot All-In-One Action
MCU Series:	i.MXRT ~	1 2
MCU Device:	: MYPT1064 CID	Image Generation Sequence Image Loading Sequence eFuse Operation Utility Boot Device Memory
	LIMARTION SIP	Start / Offset: 0x100000 Byte Length (For Read/Write): 0x2000 Bin File: C:\Users\nxf65135\D Browse
Boot Device:	FLEXSPI NOR ~	Read Erase Write (Auto Erase) 3 Execute From Start
Boot	t Device Configuration	^
Device	Configuration Data (DCD)	
Device	configuration bata (bcb)	
Port Setup		
OU	JART USB-HID	
Vendor ID:	0x15A2 ~	
Product ID:	0x0073	
	One Step	
_		
	Reset device	~ ·
Device Status		View Bootable Image Clear The Screen Save image/data file to Browse
OCOTP->MISC	_CONF0[31:00] = 0x40	
FlexRAM Partio	on =0000 - 128KB ITCM, 256KB OCRAM	Log
FlexSPI	NOR memory	Executing C:\Users\nxf65135\Desktop\tools\NXP-MCUBootUtility-2.3.0\tools\blhost2_3\win\blhost -t 5242000 -u
Page Size = 25	56 Bytes	0x15A2,0x00/3 - J write-memory 188009b/b8 C:\Users\nxtb5135\Uesktop\tools\NXP-MCUBootUtility-2.3.0\gen \user_file\user.dat 9
Block Size = 64	4 KB	v June
	~	00:05:208

2. Prepare the bootloader.

In SBL project, enter sbl/target/evkmimxrt1064 path. Disable the Enable single image function option and the Enable mcu isp support option in the menuconfig interface of Scons to disable single image mode and disable MCU ISP support.

Compile the SBL project and download it to the target board.

NOTE

- a. If the new signature key is used, please also modify the $\mathtt{sign-rsa2048-pub.c}$
- b. Programming SBL image by drag-drop of DAPLink may erase whole flash.
- 3. Plug in the ethernet cable. Run the project and the debug console prints the log as shown below.

The log "The image now in PRIMARY_SLOT slot" and "Getting IP address from DHCP" shows that the image in first slot is booted successfully. The "IPv4 Address:" and "version:1.0.0" shows that the network is connected successfully and Alibaba Cloud get the current device version 1.0.0.



Figure 143. Jump to first image

llo world2 llo world1

4. Verify the upgrade package in Web. Click the **Verify** button, fill in the version number that needs to be upgraded, and select the device to be tested. The upgrade timeout period can be omitted. Then click "**OK**".



FOTA

Verify L	pdate Package	×
Version n	mber to be upgraded	
1.0.	×	~
* Devicet	be verified	
Şelect	device	~
Select De	ce	
Device up	rade time-out (minutes)	
Piedse	nput timeout time (minutes)	
	OK Cance	
Figure 145. Verify the upgrade pac	age information	

5. The debug console shows the OTA progress.

Below is the information about the uploaded package and its version.

[21.888][LK-0309] pub: /ota/device/upgrade/a1X3jdyVAyF/SFW_K_1
<pre>[21.888][LK-0309] pub: /ota/device/upgrade/a1X3jdyVAyF/SFW_K_1</pre> [LK-030A] < 78 22 63 6F 64 65 72 3A 78 22 73 69 7A 65 22 3A 32 37 data': {"size":27 [LK-030A] < 62 61 74 61 22 3A 78 22 73 69 7A 65 22 3A 32 37 data': {"size":27 [LK-030A] < 22 36 30 30 2C 22 64 69 67 65 73 74 53 69 67 6E 2600, "digestign [LK-030A] < 22 3A 22 45 34 45 4F 72 42 60 4C 37 77 55 46 68 ":"E4FOrBmL7AUFK [LK-030A] < 22 3A 22 45 34 45 4F 72 42 60 4C 37 77 55 46 68 ":"E4FOrBmL7AUFK [LK-030A] < 64 48 78 4F 6C 51 31 32 6C 4C 63 7A 6E 46 45 4F dHxOl012LLc2nFE0 [LK-030A] < 46 4C 62 58 5A 66 42 4F 58 76 76 30 55 77 63 6A FLbXZfB0Xvv0Wcj [LK-030A] < 33 35 77 15 2 4A 58 6C 27 60 67 6A 73 50 36 54 83WqRJXL/mgjsP6T [LK-030A] < 33 33 57 71 52 4A 58 6C 27 60 67 6A 73 50 36 54 83WqRJXL/mgjsP6T [LK-030A] < 33 33 57 71 52 4A 58 6C 27 60 67 6A 73 50 36 64 83WqRJXL/mgjsP6T [LK-030A] < 33 32 6A 41 51 7A 50 32 7A 27 F2 F4 55 26 73 66 1VKkZ7kpuDXHmYPf [LK-030A] < 31 56 68 48 7A 37 68 70 75 44 58 48 60 59 59 66 1VKkZ7kpuDXHmYPf [LK-030A] < 58 2F 2F 44 59 53 56 28 76 49 78 63 37 60 6E 32 X//DYSV+v1xC7mn2 [LK-030A] < 71 57 73 35 6A 43 49 69 66 37 41 42 72 33 31 58 qWs5)CI1k7ABr31X [LK-030A] < 71 57 73 35 6A 43 49 69 66 37 41 42 72 33 31 58 qWs5)CI1k7ABr31X [LK-030A] < 61 65 38 76 51 58 61 43 62 56 50 73 72 92 4F 79 ae8VQACbVPsr/0y [LK-030A] < 61 65 38 76 51 58 61 43 62 56 50 73 72 92 4F 79 ae8VQACbVPsr/0y [LK-030A] < 64 68 5A 4B 51 52 62 39 4A 62 78 54 38 72 61 PdhzKQR9bJbXT8ra [LK-030A] < 47 55 43 64 52 37 37 68 57 39 65 455 60 66 50 47 0UCdR77hW9EmRP6 [LK-030A] < 48 54 71 34 56 60 45 68 42 47 72 5A 49 69 40 1JTq4WmEKND7Z1iM [LK-030A] < 64 65 77 79 63 48 50 45 53 77 10 45 73 79 59 46 5A 47470] [LK-030A] < 64 66 77 79 63 48 50 45 63 73 77 30 50 67 60 53 33 34 8 h /VXCAVPSF/0Y [LK-030A] < 64 66 77 79 63 48 50 45 63 73 77 30 50 67 60 53 33 34 8 h /VXCAVPSF/34 [LK-030A] < 64 66 77 79 63 48 50 45 53 77 73 70 45 77 95 94 65 5A 1704WyEENNFATA [LK-030A] < 66 63 64 64 33 53 36 C2 27 77 75 60 66 65 53 1704 140WyEFNSdAN77q
Figure 146. Get the package information
<pre>[LK-030A] < 61 74 75 72 65 30 59 64 59 69 6C 75 6F 77 4E 58 ature=YdYiluowMX [LK-030A] < 25 32 42 70 6C 61 77 52 59 66 64 4C 48 4A 70 47 %20plawRYfdLKJpG [LK-030A] < 74 5A 59 25 33 44 22 2C 22 73 69 67 6E 40 65 74 t2f%30", *signMet [LK-030A] < 68 6F 64 22 3A 22 4D 64 35 22 2C 22 6D 64 35 22 hod":'Md5", "md5" [LK-030A] < 68 6F 64 22 3A 22 4D 64 35 52 2C 22 6D 64 35 22 hod":'Md5", "md5" [LK-030A] < 64 63 35 33 62 38 63 66 36 34 62 61 34 36 31 4 dc53bdcf64ba4614 [LK-030A] < 34 64 22 7D 2C 22 69 64 22 3A 31 36 32 30 38 37 4d"}, "id":162087 [LK-030A] < 35 37 33 35 37 30 38 2C 22 60 65 73 73 61 67 5 5735708, "message [LK-030A] < 22 3A 22 73 75 63 63 65 73 73 22 7D " ":"success"} OTA target firmware version: 1.4.0, size: 272600 Bytes anknown option</pre>
[22.333][LK-0400] > Accept: text/html, application/xhtml+xml, application/xml;q=0.9, */*;q=0.8 [22.333][LK-0400] > Range: bytes=0- [22.333][LK-0400] > Range: bytes=0-

Figure 147. Target firmware version

	download renewal request has been sent successfully [22.888][LK-0400] < HITP/1.1 200 Partial Content [22.888][LK-0400] < Server: Aliyun0SS [22.888][LK-0400] < Date: Thu, 13 May 2021 03:15:46 GMT [22.999][LK-0400] < Content-Type: application/octet-stream [22.909][LK-0400] < Content-Length: 272600
	[22.999][LK-040D] < Connection: keep-alive [22.999][LK-040D] < x-oss-request-id: 609C99E21B27393636E1E89F [22.999][LK-040D] < Content-Range: bytes 0-272599/272600 [22.999][LK-040D] < Accept-Ranges: bytes [22.999][LK-040D] < ETag: "52A5C02AE7AFCDDC53B8CF64BA46144D" [22.999][LK-040D] < ETag: "52A5C02AE7AFCDDC53B8CF64BA46144D" [22.999][LK-040D] < Last-Modified: Thu, 13 May 2021 02:34:39 GMT
	<pre>[22.999][LK-0400] < x-oss-object-type: Nonmat [22.999][LK-0400] < x-oss-hash-crc6decma: 5693646425570967251 [22.999][LK-0400] < content-MD5: UqXAKuevzdxTuM9kukYUTQ== [22.999][LK-0400] < x-oss-server-time: 13 [22.999][LK-0400] < x-oss-server-time: 13 [22.999][LK-0400] <</pre>
	download 5% done, +2048 bytes [23.666][LK-0309] pub: /ota/device/progress/a1X3jdyVAyF/SFW_K_1 [LK-030A] > 7B 22 69 64 22 3A 34 2C 20 22 70 61 72 61 6D 73 {"id":4, "params [LK-030A] > 22 3A 7B 22 73 74 65 70 22 3A 22 35 22 2C 22 64 ":{"step":"5","d [LK-030A] > 65 73 63 22 3A 22 22 7D 7D esc":""}}
	Hello world1. Hello world2. download 10% done, +2048 bytes 124.444jtr.eoxoj pub: /uta/device/progress/aixojdyvAyr/srw_r_1
Figure 148. Download	[LK-030A] > 22 37 78 22 73 74 65 70 22 3A 22 31 30 22 2C 22 ";"step";"10"," [LK-030A] > 64 65 73 63 22 3A 22 22 7D 7D request

Below is shown the download progress finished and the system reset action started.



Below is shown the current package version is 1.4.0. In Alibaba Cloud web, the OTA information shows the verification is done successfully.



Types of Update Packages	Full			Update Package Signa	sture 52a5c02ae7afcddc53b8cf64ba46144d Download	
Signature Algorithm MDS	5			Module Name	default	
Total number of target device	Number of target successes	 Number of target failures 	 Number of Canceled T 	asks		
1	1	0	0			
Batch Management	Device List Update Package I	nformation				
Verify Update Package	Batch Update Please enter the	batch ID Q				
Batch ID	Batch type	Upgrade	e policy	Status 🙄	Created At	Action
ZxCm6rHXhD8qdhz	Verify Update Package	Static Up	odate	 Has complete 	May 21, 2021, 11:10:40	View

7.4 Secure FOTA

This section walks through the steps of how to perform the Secure OTA firmware.

7.4.1 Secure boot demonstration for the platform EVKMIMXRTxxxx

This section describes the steps how to enable secure boot and generate signed image. The demo targeted for the EVKMIMXRT1060 hardware platform is used as an example, although these steps can be applied to other i.MX RT platforms.

7.4.1.1 Generating Keys and Certificates

To enable ROM secure boot, generate keys and certificates. Do the following steps to generate them.

- 1. Retrieve and install the MCUXpresso Secure Provisioning tool
- 2. Run this tool, click the button to switch the processor, select MIMXRT1060. To select a processor from a different family, create a new workspace.

NOTE

: Open MCUXpresso Secure Provisioning tool with administrator mode. Otherwise some important material will not be generated.

NOTE

: For EVKMIMXRT1010 platform, select MIMXRT1015 to generate keys.

FOTA

A Build image A Write image Ø Key	vs Management		· · ·	
Source executable image:				→ Browse
Start address:				XIP: no
Use custom bootable image path:	bootable images\.bin			~ Browse
Use the following keys: Use the following keys: XIP Encryption (BEE User Keys)	Series CLPC555xx © RT10xx RT11xx RT11xx To select processo	Processor MIMXRT1015 MIMXRT1060 MIMXRT1020 MIMXRT1064 MIMXRT1024 MIMXRT1050 or from different family/series: <u>Create Nev</u>	v Workspace	Suild Image

3. Choose Boot Type as Authenticated (HAB).

	Authenticated (HAB)	 Boot Device: flex-spi-nor/IS25L 	PXXXA_IS25WPXXXA V Edit Connection: USB
A Build image A Write image A Key	rs Management		
Source executable image:			V Browse
Start address:			XIP: no
Use custom bootable image path:	bootable_images\.bin		∨ Browse
Use custom DCD (binary):			✓ Browse
Use the following keys:			~
			Build Image

4. In the Keys Management view, click Generated keys, then specify all parameters in this menu.

Processor: MIMXRT1060	Boot Type: Authenticated (HAB) v Boot Device: flex-spi-nor/IS25LPxxxA_IS25WPxxxA v
ABuild image AWrite	image 🔒 Keys Management
	Generate Keys X
	Create new CA O Use existing CA
	Drivate kay
	Certificate Browse
	Key type RSA ~ Key length 2048 ~
	Advanced
	Serial number 12345678 Password phrase test
	Duration [years] 10 Number of keys 4
	Generate Close
Missing SRK keys, at least	one SRK key is needed for selected boot type Add Keys Generate keys Export Keys Import Keys
Log	

5. Click Generate, OpenSSL output is displayed in the progress window.

gnature ok gnature ok e Subject's Distinguished Name is as follows mmonName :T61STRING:'IMG4_1_sha256_2048_65537_v3_usr' rtificate is to be certified until Apr 20 06:08:19 2031 GMT (3650 days)	
gnature ok e Subject's Distinguished Name is as follows mmonName :T61STRING:'IMG4_1_sha256_2048_65537_v3_usr' rificate is to be certified until Anr 20.06:08:18.2031_CMT_(2650_days)	
gnature ok e Subject's Distinguished Name is as follows	
gnature ok	
eck that the request matches the signature	
ing configuration from C:\nxp\MCUX_Provi_v3\bin\tools\cst\ca\openssl.cnf	
is 65537 (0x010001)	

6. You can find generated keys and certificates in the folder "keys" and "crts" in the workspace directory, Copy folder "keys", "crts" and "gen_hab_certs" to the folder sbl/target/evkmimxrt1060/secure.

7.4.1.2 SBL image preparation

To generate a signed bootable SBL image, the steps are as below:

- 1. Run <code>env.bat</code> in target board
- 2. Run scons --menuconfig to enter menu MCU SBL Core to select Enable ROM to verify sbl

FOTA



3. Enter menu MCU SBL Component > secure > selected signing method, select one application signature type, save, and quit menuconfig.

1> cmd - cc	0.05	Sear	ch	0 🖬 🗸 🛛	n - A 🖬
config - MC	U-SBL RT600 Configurati	on	cn ,		
→ MCU_SBL_Co	mponent → secure				
	Selecte	d signing method –			
Us	e the arrow keys to nav	igate this window	or press	the	
no	TKEY OF THE ITEM YOU WI	sn to select follo	wea by the	e KSPACE	
ВА	Ry. Press (Fy Tor addit	10041 1000000000	about thi	s	
	() Select sig	nature type RSA			
	(X) Select sig	nature type ROM us	e		
	() Select sig	nature type ECDSA	P256		
	(splect	A Helm A			
	() arece	v neip v			

- 4. Run scons --ide=iar command to generate IAR project or run scons --ide=mdk5 to generate Keil project.
- 5. Configure the option to generate an image with .src format in IAR project, then build the project. For Keil project, user may generate a srec format image by running:

fromelf.exe --m32combined --output "\$L@L.srec" "#L"

Category:	Factory Settings
General Options Static Analysis Runtime Checking C/C++ Compiler Assembler Output Converter Custom Build Build Actions Linker Debugger Simulator CADI CMSIS DAP GDB Server I-jet J-Link/J-Trace TI Stellaris Nu-Link PE micro ST-LINK	Output Output Output format: Motorola S-records Output file Override default sbl.srec
Third-Party Driver TI MSP-FET	
TLYDS	

7.4.1.3 Application image preparation

To generate the application image, the steps are as below:

1. Open linker file sfw\target\xxxx\board\link\MIMXRTXXXX_flexspi_nor.icf (take IAR linker file as example), and make the following changes in the linker file:

define symbol m_interrupts_start	= BOOT_FLASH_ACT_APP + 0x2000;
define symbol m_interrupts_end	= BOOT_FLASH_ACT_APP + 0x2000 + 0x3FF;
define symbol m_text_start	= BOOT_FLASH_ACT_APP + 0x2000 + 0x400;
define symbol m_text_end	= BOOT_FLASH_CAND_APP - 0x1000;

- 2. Run env.bat in one target board in SFW
- 3. Enter menu MCU SFW core and uncheck menu Enable sfw standalone xip, save and quit menuconfig



- 4. Run scons --ide=iar to generate IAR project or run scons --ide=mdk5 to generate the Keil project for SFW.
- 5. Configure the option to generate an image with .srec format, then build the project.

Category:	Factory Settings
General Options Static Analysis Runtime Checking C/C++ Compiler Assembler Output Converter Custom Build Build Actions Linker Debugger Simulator CAD1 CMSIS DAP GDB Server Light J-Link/J-Trace TI Stellaris Nar-Link PE micro ST-LINK Third Party Driver TI MSP FET TI XDS	Output Generate additional output Output format: Motorola S-records Output file Override default sfw.srec

- NOTE

For the other signing method, generate a binary format application.

7.4.1.4 Program OCOTP (eFuse)

Below is an example of how to program SRK table and enable HAB closed mode.

NOTE

In the Development phase, the device may be under HAB open mode for most use cases.

1. Find and open the script program_ocotp.bat in sbl/target/evkmimxrt1060/secure, then set the correct installation path for tools elftosb, cst, blhost and so on:

```
SET "PATH=C:\nxp\MCUX_Provi_v3\bin\tools\elftosb\win;%PATH%"
SET "PATH=C:\nxp\MCUX_Provi_v3\bin\tools\sdphost\win;%PATH%"
SET "PATH=C:\nxp\MCUX_Provi_v3\bin\tools\blhost\win;%PATH%"
SET "PATH=C:\nxp\MCUX Provi v3\bin\tools\cst\mingw32\bin;%PATH%"
```

2. Open the file SRK fuses.bin in hex mode under the folder gen hab certs.

0 1 2 3 4 5 6 7 8 9 a b c d e f 00000000h: 5D 22 E8 F7 C5 09 46 91 33 00 E0 D3 84 92 3A 29 00000010h: A 65 97 C5 E3 FD D1 46 46 14 C0 DD CA 0B 8D BB

Figure 161. SRK hash value in hex

3. The value located in the efuse file is intended to be burned to the SRK_HASH efuse field on the SoC. This hash value must be burned to the SoC efuses in the following order (the first word to the first fuse row index):

f7e8225d, 914609c5, d3e00033, 293a9284, c59765a8, 46d1fde3, ddc01446, bb8d0bca. Note the data endianness.

If the user program efuses with the script program_ocotp.bat, remember to update the SRK hash value according to user's SRK_fuses.bin in script. The commands used to program efuses are commented out by default. Enable it by hand. For other platform, eFuse OCOTP index of SRK hash table may be different. Refer to the fuse map for the NXP processor used.

```
blhost.exe -u 0x15A2,0x0073 -j -- efuse-program-once 0x18 f7e8225d
blhost.exe -u 0x15A2,0x0073 -j -- efuse-program-once 0x19 914609c5
blhost.exe -u 0x15A2,0x0073 -j -- efuse-program-once 0x1a d3e00033
blhost.exe -u 0x15A2,0x0073 -j -- efuse-program-once 0x1b 293a9284
blhost.exe -u 0x15A2,0x0073 -j -- efuse-program-once 0x1c c59765a8
blhost.exe -u 0x15A2,0x0073 -j -- efuse-program-once 0x1d 46d1fde3
blhost.exe -u 0x15A2,0x0073 -j -- efuse-program-once 0x1e ddc01446
blhost.exe -u 0x15A2,0x0073 -j -- efuse-program-once 0x1e ddc01446
```

 Enable HAB close mode using the following command. In Production phase, enable HAB closed mode and sign the SBL image.

blhost.exe -u 0x15A2,0x0073 -j -- efuse-program-once 0x06 0000002

5. Verify the eFuse value via command efuse-read-once.

blhost.exe -u 0x15A2,0x0073 -j -- efuse-read-once 0x18

 Put switch SW7-4 on the EVKMIMXRT1060 board to enter Serial Downloader mode, then run script program_ocotp.bat.

7.4.1.5 Run SBL and application

To generate signed SBL and application, do the following steps. Then program them to board.

- Copy sbl.srec and sfw.srec to the folder sbl/target/evkmimxrt1060/secure. If you select RSA or ECDSA signing type, use sfw.bin.
- 2. To enter Serial Downloader mode and connect USB OTG and DEBUG USB port, make the EVKMIMXRT1060 board.
- 3. Enter folder secure in scons environment by inputting cd secure.
- 4. Run sign_sbl_app.bat to generate the final signed SBL and application. If you want to generate the application image for the next update, remember to change the parameter version number in the imgtool.py command line.
- 5. Download the signed SBL and application by the script sign_sbl_app.bat or other tools. This script downloads the application image into slot1. If you tested the OTA procedure before, SBL may still try to run application in slot2.

6. Switch (SW7-3 on, SW7-4 off) to normal boot mode, then reset board. You see output in the terminal.



7.4.2 Encrypted XIP boot demonstration for the platform EVKMXRTxxxx

This section describes the steps to enable XIP encrypted authenticated boot. The demo targeted for EVKMIMXRT1060 hardware platform is used as an example.

7.4.2.1 SBL image preparation

To generate signed bootable SBL image, the steps are as below:

- 1. Run ${\tt env.bat}$ in the target board.
- 2. Run scons --menuconfig to enter menu "MCU SBL Core" to select "Enable ROM to verify sbl".

<1> cmd - scons -	Search	+ 🕂 🔍	1 - 🔒 💷	\equiv
.config - MCU-SBL RT1060 Configuration				
→ MCU SBL core				
Arrow keys navigate the menu. <enter> sele submenus). Highlighted letters are he includes, <n> excludes, <m> modularizes fer exit, <? > for Help, for Search. Legen</m></n></enter>	ects submenus otkeys. Pres atures. Pres d: [*] built-	s> (or sing <y> s <esc><es in [] ex</es </esc></y>	empty c> to cluded	
<pre>[*] Enable ROM to verify sbl (400) Maximum number of flash sectors [] Enable single image function MCU SBL Flash Map> MCU SBL metadata header></pre>	per image			
<pre></pre>	< Save >	< Load >		
kconfig-mconf.exe*[32]:19036 « 180206[64] 1/1	[+] NUM PRI 82	2x24 (11,9) 25V	15628 100%	·//

3. Enter menu MCU SBL Component > secure, select Encrypted XIP function, save and quit menuconfig.



4. Run the scons --ide=iar command to generate the IAR project.

5. Configure the project to generate an image with the .srec format, then build the project. For the Keil project, generate the srec format image by running fromelf.exe --m32combined --output "\$L@L.srec".

7.4.2.2 Application image preparation

To generate application image for encryption, the steps are as below:

1. Open the linker file sfw\target\xxxx\board\link\MIMXRTXXXX_flexspi_nor.icf (take the IAR linker file as example), and make the following changes in the linker file:

define symbol m_interrupts_start	= BOOT_FLASH_ACT_APP + 0x2000;
define symbol m_interrupts_end	= BOOT_FLASH_ACT_APP + 0x2000 + 0x3FF;
define symbol m_text_start	= BOOT_FLASH_ACT_APP + 0x2000 + 0x400;
define symbol m_text_end	= BOOT_FLASH_CAND_APP - 0x1000;

- 2. Run env.bat in target platform
- 3. Enter the menu MCU SFW core and uncheck the menu ${\tt Enable}$ sfw standalone xip
- 4. Enter the menu MCU SFW component > secure and check the menu Encrypted XIP function, then save and quit menuconfig.

cmd - sconsmenuconfig		- 🗆 X
<1> cmd - scons -	Search	으 于 🕕 🛨 🚰
.config - MCU-SFW RT1060 Configuration → MCU SFW Component → secure		
Arrow keys navigate the menu. <enter> se submenus). Highlighted letters are includes, <n> excludes, <m> modularizes exit, <? > for Help, for Search. Leg</m></n></enter>	elects submenus hotkeys. Pres features. Pres end: [*] built	s> (or empty ssing <y> ss <esc><esc> to -in [] excluded</esc></esc></y>
<pre>[*] Encrypted XIP function [*] enable mbedtls (aws_mbedtls_config.h) Set mbedtls config.h)</pre>	onfig file	
<pre><select> < Exit > < Help ></select></pre>	< Save >	< Load >
e 165. Enable key context upgrade in SFW		

- 5. Generate the SFW project.
- 6. Change codes to call the function <code>update_key_context()</code> after calling enable_image() if application supports OTA.
- 7. Configure option to generate an image with thesrec format, then build the project.

NOTE For RSA or ECDSA signing method, don't change linker file in step1 and need to generate binary format image.

7.4.2.3 Program KEK (eFuse)

The KEK (Key of Encryption Key) serves as the key for the BEE to unwrap the Key context.

You may not write KEK and keep that efuse value as all 0 s in test phase. Below is an example how to burn KEK to efuse SW_GP2 for EVKMIMXRT1060, add the command below into the file program_ocotp.bat.

```
:: kek=00112233445566778899aabbccddeeff
blhost.exe -u 0x15A2,0x0073 -j -- efuse-program-once 0x29 ccddeeff
blhost.exe -u 0x15A2,0x0073 -j -- efuse-program-once 0x2a 8899aabb
blhost.exe -u 0x15A2,0x0073 -j -- efuse-program-once 0x2b 44556677
blhost.exe -u 0x15A2,0x0073 -j -- efuse-program-once 0x2c 00112233
```

The KEK (Key of Encryption Key) serves as the key for the OTFAD which uses EVKMIMXRT1010 or EVKMIMXRT1170. Follow the order shown below: 0xffeeddcc, 0xbbaa9988, 0x77665544, 0x33221100.

7.4.2.4 Run SBL and application

Do the following steps to generate signed and encrypted SBL and application together.

- 1. Copy sbl.srec and sfw.srec to the folder sbl/target/evkmimxrt1060/secure
- 2. Edit sign enc sbl app.bat and set the KEK and the encrypted region.
- 3. Make the EVKMIMXRT1060 board to enter Serial Downloader mode and connect USB OTG and the DEBUG USB port.
- 4. Run sign enc sbl app.bat to generate final signed and encrypted SBL and application1.
- 5. Download SBL and application by this script or other tools.
- Switch (SW7-3 ON, SW7-4 OFF) to normal boot mode, set SW5-1 ON then reset board. You see the output in the terminal.



7.4.2.5 Application OTA image preparation

To generate the signed and encrypted application for upgrading, do the following steps:

- 1. Generate a project as in section 7.4.2.2
- 2. Build the image and rename it to sfw2.srec or sfw2.bin
- 3. Open script sign enc sfw2.bat, set the KEK and encrypted region.
- 4. Run sign_enc_sfw2.bat, file sfw_2_enc.bin is the final image.

7.4.3 Secure boot demonstration for the platform LPC55S69

This section describes the steps to enable an XIP encrypted authenticated boot. The demo targeted for LPC55S69(revision 1B) hardware platform is used as an example.

7.4.3.1 Generating Keys and Certificates

To enable ROM secure boot, generate keys and certificates. Follow the steps to generate them.

- 1. Install the MCUXpresso Secure Provisioning tool
- 2. Run this tool, click File > New Workspace, then select processor LPC55S69 to create a new workspace.

New Workspa	ice	×
Workspace:	C:\Users\nxp96245\secure_lpc55s69	Browse
Series	Processor	
LPC55Sx	x OLPC55S04 OLPC55S26	
◯ RT10xx	○ LPC55S06 ○ LPC55S28	
⊖ RT11xx	○ LPC55S14 ○ LPC55S66	
⊖ RTxxx	○ LPC55S16	
		Create
		Create
gure 167. Creating a new work	space	

3. Choose **Boot Type** as Encrypted (PRINCE) and Signed.

MCUXpresso Secure Provision	ing - C:\Users\nxp96245\secure_lpc55s69 —	
File Target Tools Help		
Processor: LPC55S69 Boot Typ	Encrypted (PRINCE) and Signed V Boot Device: onchip_memory/Internal Flash V Edit Co	onnection: UART
✓ Build image ✓ Write image	✓ Keys Management	
Source executable image:	C:\LocalData\Work\sbl\target\lpc55s69\iar\build\iar\Exe\sbl.bin	Browse
Start address:	0	XIP: yes
Use custom bootable image p	bootable_images\sbl.sb	Browse
TrustZone pre-configuration Er	nabled (preset) v	Browse
Use the following keys:	OT4: IMG4_1	~
Key source:	eyStore \vee	
User key:		Random
SBKEK: 7	F393C4207EC4864F6663FC54583F5958EA7B9DFDFECDED249DF2304506ABE9C	Random
PRINCE Regions		
		Build Image
168. Selecting Boot Type		

4. In the Keys Management view, click the button Generated keys, then specify all parameters in this menu.

Generate Keys X Certificate chain Chain RoT + IMG X Serial number 3cc30000abababab Key type RSA Key length 2048 X	Build image Write image	 Keys Management 		
Advanced Number of RoT 4 Uuration [years] 10 Generate Close	Certificate chain Chain RoT Serial number 3cc30000a Advanced Number of RoT 4	+ IMG V Key type abababab Key length V Duration [years] 10 Generate	RSA ~ 2048 ~	



5. Click the button Generate, OpenSSL output is displayed in the progress window.

••••••••••••••••••••••••••••	^
e is 65537 (0x010001)	
Signature ok	
subject=CN = ROT4_sha256_2048_65537_v3_ca	
Getting Private key	
Generating RSA private key, 2048 bit long modulus	
+++++	
e is 65537 (0x010001)	
Signature ok	
subject=CN = IMG4_1_sha256_2048_65537_v3_usr	
Getting CA Private Key	
	~
SUCCESS: Generating keys	Close

6. You can find generated keys and certificates in the folder keys and crts in the workspace directory.

7.4.3.2 SBL image preparation

The following steps describe the procedure of creating an SBL image.

- 1. Run ${\tt env.bat}$ in target board.
- 2. Enter the menu MCU SFW Core >, select one application signature type, save and quit menuconfig.
| <pre></pre> | Search | _
₽ ₹ ₹ | □ ×
 |
|---|--|--|---------|
| <pre>→ MCU SBL Component → secure Selected signing me Use the arrow keys to navigate this w hotkey of the item you wish to select BAR>. Press <?> for additional inform () Select signature type (X) Select signature type () Select signature type () Select signature type</pre> | ethod
vindow or pres
followed by
nation about t
RSA
ROM use
ECDSA_P256 | ss the
the <spaci
his</spaci
 | |
| Figure 171. Selecting signing method | | | |

- 3. Run the scons --ide=iar command to generate the IAR project.
- 4. Build binary a image from IDE, it is plain image for unsecure boot.

7.4.3.3 Application image preparation

The following steps describe the procedure for creating signed image.

- 1. Run <code>env.bat</code> in target board under SFW repo
- 2. Enter the menu MCU SFW core and uncheck the menu Enable sfw standalone xip, save and quit menuconfig



3. Generate an IAR project, following the example below:

```
scons --ide=iar
```

- 4. In Project > Options > Output Converter, check Generate additional output and select Raw binary output format
- 5. Build the project.

7.4.3.4 Sign and Program encrypted SBL

This section describes the building and writing of authenticated SBL image by tool MCUXpresso Secure Provisioning Tool (SPT).

- 1. In the Build Image view, select the image position as a Source executable image.
- 2. In Start address tab, input 0.
- 3. Select Enabled(present) in TrustZone pre-configuration.
- 4. For use the following keys, select any key chain, for example ROT1: IMG1_1.
- 5. Input SBKEK or generate one with the **Random** button.

Processor. EPC55569 Boot Type.	encrypted (PKINCE) and signed v boot bevice. Onchip_memory/internal Plash v Edit Con	inection. Use
✓ Build image ▲ Write image ✓	Keys Management	
Source executable image:	C:\LocalData\Work\sbl\target\lpc55s69\iar\build\iar\Exe\sbl.bin	Browse
Start address:	0x0	XIP: yes
Use custom bootable image path	: bootable_images\sbl.sb ~	Browse
TrustZone pre-configuration Enab	led (preset) v	Browse
Use the following keys: ROT	: IMG1_1	\sim
Key source: KeyS	tore 🗸	
User key:		Random
SBKEK: 0F7C	29CA4B5E03E37E77F354335590C7F0AD7700586DA7AD25BE33F185EB1EB0	Random
PRINCE Regions		
	В	uild Image

6. Open the PRINCE configuration and check the configuration. Set the encrypted size of the SBL image in the PRINCE region. If you do not encrypt the image, skip this step. Select Signed in Boot Type.

PRINCE Regi	on Configuration		×
Region:	Region 0	Region 1	Region 2
Erase check	: 🗹 Check erase	Check erase	Check erase
Lock:	Lock	Lock	Lock not supported
Enable:	✓ Enable	Enable	Enable
Base addr:	0x0000#### ~	0x0004#### ~	0x0008#### ~
Start offset	0x###00000 ~	0x###00000 ~	0x###00000 ~
End offset:	0x###0FFFF ~	0x###3FFFF ~	0x###1BFFF ~
Addr range	0x0000000-0x0000FFFF	0x0000000-0x00000000	0x00000000-0x00000000
			OK Cancel
ure 174. Setting encry	ption region		

7. Click **Build image**, output is displayed in the progress window.

C:\Users\nxp96245\secure_lpc55s69\keys\IMG1_1sha256_2048_65537_v3_usr_key.pem" C:\Users\nxp96245\secure_lpc55s69\crts\ROT1_sha256_2048_65537_v3_ca_crt.der" "C:\Users\nxp96245\secure_lpc55s69\crts\ROT2_sha256_2048_65537_v3_ca_crt.der" "C:\Users\nxp96245\secure_lpc55s69\crts\ROT3_sha256_2048_65537_v3_ca_crt.der" "C:\Users\nxp96245\secure_lpc55s69\crts\ROT4_sha256_2048_65537_v3_ca_crt.der" "C:\Users\nxp96245\secure_lpc55s69\crts\ROT1_sha256_2048_65537_v3_ca_crt.der" "C:\Users\nxp96245\secure_lpc55s69\crts\ROT1_sha256_2048_65537_v3_ca_crt.der" "C:\Users\nxp96245\secure_lpc55s69\crts\ROT1_sha256_2048_65537_v3_ca_crt.der" "C:\Users\nxp96245\secure_lpc55s69\crts\IMG1_1_sha256_2048_65537_v3_usr_crt.der" "C:\Users\nxp96245\secure_lpc55s69\crts\LMG1_sha256_bitm"	-R -R -R -S -S	^
RKTH: 122a7fd6fc2527a5741425ffe433e33001045dde783e6a1c59f1afef75eadf72		
elftosh succeeded		
I		~
SUCCESS: Building image		Close

- 8. Make sure that the board is connected and make the processor into ISP mode by pressing the ISP pin during reset stage.
- 9. Click the **USB** tab, set the connection to USB or UART according to selected port and test the connection to the processor

Hotesson. Eresssos boot type. Eletyp		or bevice. Onenip_memory/		
✓ Build image ✓ Write image ✓ Keys M	lanagement			-
Bootable Image to be written	Connection		×	
Use built image		Connection Status		_
Image path: bootable_images\sk	O USB	Feature	Detected value	wse
Additional Input Files	Vendor ID 0x1FC9	Connection	ОК	
Title Path	Product ID 0x0021 V	Mode	ROM BootLoader	
MasterBootImag gen sb\mbi config.	UART	Security	PFR not sealed yet	
SBKEK (.bin) gen_scripts\sbkek.b	COM port COM14	Key-Store	Not enrolled yet	
Prince Configurat gen_sb\prince.json		vill be enrolled by the wr	ite script during first execution	
	Baud Rate 115200	LPC55S69	match	
				Import
		Test Connection	Result: OK	import
		rest connection		Tage
			OK Cancol	
			OK Cancer	:

- 10. Make sure that the Use built image checkbox is selected.
- 11. Click Write Image.



Build image	rite image Keys Management		03
Bootable Image to h	e written		
Use built image			
Image path:	bootable_images\sbl.sb	~	Browse
Additional Input File	s		
Title	Path	Description	
MasterBootImag	gen sb\mbi config.json	Configuration file for generation of Master Boot Image	
SBKEK (.bin)	gen_scripts\sbkek.bin	Key used to sign SB image capsule	
Prince Configurat	. gen_sb\prince.json	PRINCE regions configuration file	
			Impor
		Enable security	rite Image

12. Write result is displayed in the progress window.



7.4.3.5 Sign and Program application image

You can download an SFW image with BootRom in ISP mode or with SBL in ISP mode. Do the following steps to generate a signed SFW image:

- 1. Copy sfw.bin into folder sbl/target/lpc55s69/secure.
- 2. Edit mbi_config.json to set correct keys and certifications path or copy the older keys and crts from MCUXpresso Secure Provisioning lpc55s69 workspace to here.

```
"rootCertificate0File": "./crts/ROT1_sha256_2048_65537_v3_ca_crt.der",
"rootCertificate1File": "./crts/ROT2_sha256_2048_65537_v3_ca_crt.der",
"rootCertificate2File": "./crts/ROT3_sha256_2048_65537_v3_ca_crt.der",
"rootCertificate3File": "./crts/ROT4_sha256_2048_65537_v3_ca_crt.der",
"mainCertPrivateKeyFile": "./keys/IMG1_1_sha256_2048_65537_v3_usr_key.pem",
"chainCertificate0File0": "./crts/ROT1_sha256_2048_65537_v3_usr_crt.der",
"chainCertificate0File1": "./crts/IMG1_1_sha256_2048_65537_v3_usr_crt.der",
```

FOTA

3. Open signed enc sfw.bat and set the correct installation path for tools elftosb and blhost.

```
SET "PATH=C:\nxp\MCUX_Provi_v3\bin\tools\elftosb\win;%PATH%"
SET "PATH=C:\nxp\MCUX Provi v3\bin\tools\blhost\win;%PATH%"
```

- 4. Set correct com port and configure PRINCE region in sign_enc_sfw.bat.
- 5. Enter folder secure in scons environment by inputting cd secure.
- 6. Run sign_enc_sfw.bat to generate final signed application image.
- 7. Press any key to configure the PRINCE region to set encrypted region after generating signed image. You can omit this step and download the signed image.
- 8. Make the LPC55S69 EVK board to enter ISP mode.
- 9. Download image, if the write operation was successful, reset the board.



NOTE

For whole encrypted application image, pad image with 8k bytes aligned.

7.4.4 Secure boot demonstration for the platform EVKMIMXRTxxx

This section describes the steps to enable **XIP** encrypted authenticated boot. The demo targeted for the MIMXRT685S hardware platform is used as an example. This section also applies to platform MIMXRT595S.

7.4.4.1 Generating Keys and Certificates

To enable ROM secure boot, generate keys and certificates. Follow the steps to generate them.

- 1. Install the MCUXpresso Secure Provisioning tool.
- 2. Run this tool, click File > New Workspace, then select processor MIMXRT685 to create a new workspace.

	New Workspace		×
	Workspace: C:\U	Jsers\nxp96245\secure_RT685	Browse
	Series O LPC555xx O RT10xx O RT11xx O RTxxx	Processor MIMXRT533S MIMXRT685S MIMXRT555S MIMXRT595S MIMXRT633S	
			Create
Figure 180. Creating	g a new workspac	Ce	

3. Choose Boot Type as Signed.

Processor: MIMXRT6855 Bo	ot Type: Signed	 I oot Devic 	e: flex-spi-nor/MX25UM5134	5G - Port B \vee Edit Cor	nnection: US
ABuild image AWrite image	🔒 Keys Management				
Source executable image:				~	Browse
Start address:					XIP: no
Use custom bootable image	path: bootable_images\.bin	1		~	Browse
TrustZone pre-configuration	Disabled ~			~	Browse
Use the following keys:					~
Key source:	OTP ~				
User key:					Random
SBKEK:					Random
				В	uild Image
Log					
Status of the operation: SUCC	иннинниннинниннинниннинниннин ESS : New workspace created:	annannannannannannannannan C:\Users\nxp96245\secu	######################################		
		0.00	DICOL		

4. In the Keys Management view, click the button Generated keys, then specify all parameters in this menu.

Processor: N	IMXRT685S Boot Type: Signed	 Boot Device: fle 	x-spi-nor/MX25UM5	1345G - Port B \vee Edit 🛛	Connection: USB
A Build imag	e 🛕 Write image 🔒 Keys Management				
	Generate Keys		×		
	Certificate chain Chain BoT + IMG	Key type RSA	~		
	Serial number 3cc30000abababab	Key length 2048			
		hey length 2010			
	Advanced				
	Number of RoT 4 V Duration [years]	10			
		Generate	lose		
Missing ROT	eys, at least one ROT key is needed for selected boo	ot type	Add Keys Gener	Tate keys Export Keys	Import Keys
I.e.					

5. Click the button Generate. OpenSSL output is displayed in the progress window.

UCCESS: Generating keys	Close
	~
Getting CA Private Key	
ubject=CN = IMG4_1_sha256_2048_65537_v3_usr	
Signature ok	
e is 65537 (0x010001)	
+++++	
Generating RSA private key, 2048 bit long modulus	
Getting Private key	
subject=CN = ROT4_sha256_2048_65537_v3_ca	
Signature ok	
e is 65537 (0x010001)	

6. You can find generated keys and certificates in the folder keys and crts in the workspace directory. Copy folder "keys" and "crts" to folder sbl/target/evkmimxrt600/secure.

7.4.4.2 SBL image preparation

The following steps describe the procedure for creating SBL image.

- 1. Run <code>env.bat</code> which is under folder <code>sbl/target/evkmimxrt600</code>
- $2. \ Run \ \texttt{scons} \ -\texttt{menuconfig} \ to \ \texttt{enter} \ the \ \texttt{menu} \ \texttt{MCU} \ \texttt{SBL} \ \texttt{Core} \ to \ \texttt{select} \ \texttt{Enable} \ \texttt{ROM} \ \texttt{to} \ \texttt{verify} \ \texttt{sbl}.$



3. Enter menu MCU SBL Component > secure > selected signing method, select one application signature type, save, and quit menuconfig.

<pre> Search</pre>		mendeoning					
<pre>> MCU-SBL RT600 Configuration > MCU SBL Component → secure</pre>	📕 <1> cmd	- scons -		Searc	ch	<u> 2</u> 🗄 🔻 [1 - 🛗 🛄
<pre>> MCU SBL Component → secure</pre>	config -	MCU-SBL RT600 Conf	figuration				
Selected signing method Use the arrow keys to navigate this window or press the hotkey of the item you wish to select followed by the <space BAR>. Press <? > for additional information about this () Select signature type RSA (X) Select signature type ROM use () Select signature type ECDSA_P256 () Select signature type ECDSA_P256</space 	MCU SBL	Component → secure					
Selected signing method Use the arrow keys to navigate this window or press the hotkey of the item you wish to select followed by the <space BAR>. Press <? > for additional information about this () Select signature type RSA (X) Select signature type ROM use () Select signature type ECDSA_P256 () Select signature type ECDSA_P256</space 							
Use the arrow keys to navigate this window or press the hotkey of the item you wish to select followed by the <space BAR>. Press <? > for additional information about this () Select signature type RSA (X) Select signature type ROM use () Select signature type ECDSA_P256</space 			Selected signi	ng method -			-
<pre>ose the arrow keys to havigate this window of press the hotkey of the item you wish to select followed by the <space BAR>. Press <?> for additional information about this () Select signature type RSA (X) Select signature type ROM use () Select signature type ECDSA_P256</space </pre>		lise the arrow keys	s to navigate t	his window (or nress	the	
BAR>. Press for additional information about this <pre>() Select signature type RSA (X) Select signature type ROM use () Select signature type ECDSA_P256</pre>		hotkey of the iter	n vou wish to s	elect follo	wed by th	e <space< td=""><td></td></space<>	
<pre>() Select signature type RSA (X) Select signature type ROM use () Select signature type ECDSA_P256 () Select > < Help ></pre>		BAR>. Press fo	or additional i	nformation	about thi	S	
<pre>() Select signature type RSA (X) Select signature type ROM use () Select signature type ECDSA_P256 </pre>							1
(X) Select signature type ROM use () Select signature type ECDSA_P256		<u>()</u> Sel	lect signature	type RSA	_		
() Select signature type ECDSA_P256		(X) <mark>S</mark> el	lect signature	type ROM us	e		
<pre>< Help ></pre>		() Se]	lect signature	type ECDSA_I	P256		
<pre></pre>							
<pre></pre>							
<pre><select> < Help ></select></pre>							
<pre><select> < Help ></select></pre>							
			<select></select>	< Help >			
onfig-mconf.exe*[32]:10444	<i>a</i>		10000			(0.0.4.0)	

4. Run scons --ide=iar command to generate the IAR project or run scons --ide=mdk5 to generate the Keil project.

5. Configure option to generate an image with binary format, then build the project.

Category:	Factory Settin	gs
General Options		
Static Analysis		
Runtime Checking	Output	
C/C++ Compiler		
Assembler		
Output Converter	Generate additional output	
Custom Build		
Build Actions	Output format:	
Linker	Raw binary \vee	
Debugger		
Simulator	Output file	- 11
CADI		
CMSIS DAP		
GDB Server	shi hin	
I-jet	30.011	
J-Link/J-Trace		
TI Stellaris		
Nu-Link		
PE micro		
ST-LINK		
Third-Party Driver		
TI MSP-FET		
TI XDS		

Figure 186. Set output format

7.4.4.3 Application image preparation

To generate plain application image, the steps are as below:

- 1. Run env.bat which is under folder sfw/target/evkmimxrt600.
- 2. Run command scons --menuconfig.
- 3. Enter menu MCU SFW core and uncheck menu Enable sfw standalone xip.

🔄 cmd - sconsmenuconfig					X
<1> cmd - scons -		Search	+ 🛃 🔍	🚺 🔻 🔮	
.config - MCU-SFW RT600 Config	guration				
→ MCU SFW core					_
Arrow keys navigate the me submenus). Highlight includes, <n> excludes, <m exit, <? > for Help, for</m </n>	— MCU SFW cor enu. <enter> s d letters are modularizes or Search. Leg</enter>	e elects submen hotkeys. Pro features. Pro end: [*] buil [:]	us> (or essing <y> ess <esc><es t-in [] e></es </esc></y>	empty sc> to cluded	
[] Enable sfw standal [*] Enable OTA [*] OTA from sdcard [*] OTA from u-disk MCU SFW Flash Map MCU SFW metadata h	.one xip > neader>				
<pre>kselect> < Exit</pre>	> < Help >	< Save >	< Load >		

4. For image, which must be encrypted, enter the menu MCU SFW component > secure and check the menu Encrypted XIP function, then save and quit menuconfig.



- 5. Generate the SFW project.
- 6. Configure the option to generate an image with binary format, then build the project.

7.4.4.4 Run signed SBL and SFW

Do the following steps to generate signed SBL and application, then program them to board.

- 1. Copy sbl.bin and sfw.bin into the folder sbl/target/ evkmimxrt600/secure.
- 2. Make the evkmimxrt600 board to enter ISP mode Downloader modes based on ISP pins (ISP0 on, ISP1 off, ISP2 off).
- 3. Connect a micro USB cable from connector J5 (LINK USB).
- 4. Enter folder secure in scons environment by inputting cd secure.
- 5. Run sign_sbl_app.bat to generate final signed SBL and application. If you want to generate the application image for the next update, remember to change the parameter version number in the imgtool.py command line.
- 6. Download signed SBL and application by the script sign_sbl_app.bat or other tools. This script downloads the application image into slot1. If you test the OTA procedure before, SBL may still try to run application in slot2.
- 7. Program OTP according to section 7.4.4.6
- 8. Switch to normal boot mode, then reset board. You see the output in the terminal.

7.4.4.5 Run encrypted SBL and SFW

Do the following steps to generate signed and encrypted SBL and application together.

- 1. Copy sbl.bin and sfw.bin into folder sbl/target/ evkmimxrt600/secure.
- 2. Edit sign_enc_sbl_app.bat and set the KEK and encrypted region.
- 3. Make the evkmimxrt600 board to enter ISP mode Downloader modes based on ISP pins (ISP0 on, ISP1 off, ISP2 off).
- 4. Connect a micro USB cable from connector J5 (LINK USB).
- 5. Enter folder secure in scons environment by inputting cd secure.
- 6. Run sign enc sbl app.bat to generate final signed and encrypted SBL and application.
- 7. Download SBL and application by this script or other tools.
- 8. Program OTP according to section 7.4.4.6
- 9. Switch to normal boot mode, then reset board. You see the output in the terminal.



7.4.4.6 Program OTP (eFuse)

Below is an example to program SRK table and enable secure boot. In Development phase, user could use shadow registers to test the Secure boot.

1. Find and open script program_ocotp.bat in sbl/target/evkmimxrt600/secure, then set the correct installation path for tool blhost and com port.

SET "PATH=C:\nxp\MCUX_Provi_v3\bin\tools\blhost\win;%PATH%"
SET com port=COMX,115200

2. Find the RKT hash(RKTH) value in the log of generating SBL image.

15. Output the root certificates SHA256 hash (RKTH). Success. RKTH: 8b8123193c27489fe835e104be046187dbc5507c310de41b469e68d5842decc0

Figure 191. RKTH value in hex

 The RKTH value showed in the log is intended to be burned into the OTP fuses. The RKTH value generated by elftosb is in big-endian format. To store the value in OTP correctly, the byte order of the words supplied to efuse-program-once command needs to be byte-swapped to little-endian format. For example, 1923818b, 9f48273c, 04e135e8, 876104be, 7c50c5db, 1be40d31, d5689e46, c0ec2d84.

```
blhost -p COMx,115200 -t 15000 -- efuse-program-once 0x78 1923818b
blhost -p COMx,115200 -t 15000 -- efuse-program-once 0x79 9f48273c
blhost -p COMx,115200 -t 15000 -- efuse-program-once 0x7A 04e135e8
blhost -p COMx,115200 -t 15000 -- efuse-program-once 0x7B 876104be
blhost -p COMx,115200 -t 15000 -- efuse-program-once 0x7C 7c50c5db
blhost -p COMx,115200 -t 15000 -- efuse-program-once 0x7D 1be40d31
blhost -p COMx,115200 -t 15000 -- efuse-program-once 0x7E d5689e46
blhost -p COMx,115200 -t 15000 -- efuse-program-once 0x7F c0ec2d84
```

NOTE -

If user program efuses with script program_ocotp.bat, remember to update the RKTH value according to user's log. The commands used to program efuses are commented out by default. User needs to enable it by hand.

4. User can enable secure boot using the following command:

blhost -p COMx,115200 -t 15000 -- efuse-program-once 0x60 0x900000

5. User can verify the eFuse value via command efuse-read-once

blhost -p COMx,115200 -t 15000 -- efuse-read-once 0x60

6. Enable OTFAD and configure OTP_MASTER_KEY, OTFAD_SEED efuses for image decryption

```
// program OTP MASTER KEY
blhost -p COMx,115200 -t 15000 -- efuse-program-once 0x70 ccddeeff
blhost -p COMx,115200 -t 15000 -- efuse-program-once 0x71 8899aabb
blhost -p COMx,115200 -t 15000 -- efuse-program-once 0x72 44556677
blhost -p COMx,115200 -t 15000 -- efuse-program-once 0x73 00112233
blhost -p COMx,115200 -t 15000 -- efuse-program-once 0x74 0c0d0e0f
blhost -p COMx,115200 -t 15000 -- efuse-program-once 0x75 08090a0b
blhost -p COMx,115200 -t 15000 -- efuse-program-once 0x76 04050607
blhost -p COMx,115200 -t 15000 -- efuse-program-once 0x77 00010203
// program OTFAD SEED
blhost -p COMx,115200 -t 15000 -- efuse-program-once 0x6c 62184d50
blhost -p COMx,115200 -t 15000 -- efuse-program-once 0x6d d5ae8d29
blhost -p COMx,115200 -t 15000 -- efuse-program-once 0x6e bf6af264
blhost -p COMx,115200 -t 15000 -- efuse-program-once 0x6f 3a72eb7f
// enable OTFAD boot
blhost -p COMx,115200 -t 15000 -- efuse-program-once 0x6a 00001000
// config flash settings for MIMXRT685-EVK
blhost -p COMx,115200 -t 15000 -- efuse-program-once 0x61 314000
```

7. Put ISP pins (ISP0 on, ISP1 off, ISP2 off) to make evkmimxrt600 board enter ISP mode Downloader mode, then run the script program ocotp.bat.

7.4.4.7 Application OTA image preparation

Do the following steps to generate signed and encrypted application for upgrading.

- 1. Build the image as in section 7.4.4.3
- 2. Rename it to sfw2.bin and copy it to folder sb1/target/evkmimxrt600/secure.
- 3. Open script sign enc sfw2.bat, set the KEK and encrypted region.
- 4. Run sign_enc_sfw2.bat, file sfw_2_enc.bin is final image.

Chapter 8 Known issues

None

Chapter 9 Revision history

Table 8. Revision history

Revision	Date	Description	
0	26 August 2021	First release to open source	
1.1.0	08 November 2021	 Support RT500, RT600 signature and encryption feature Optimize the revert flow 	

How To Reach Us
Home Page:
nxp.com
Web Support:
nxp.com/support

Limited warranty and liability — Information in this document is provided solely to enable system and software implementers to use NXP products. There are no express or implied copyright licenses granted hereunder to design or fabricate any integrated circuits based on the information in this document. NXP reserves the right to make changes without further notice to any products herein.

NXP makes no warranty, representation, or guarantee regarding the suitability of its products for any particular purpose, nor does NXP assume any liability arising out of the application or use of any product or circuit, and specifically disclaims any and all liability, including without limitation consequential or incidental damages. "Typical" parameters that may be provided in NXP data sheets and/or specifications can and do vary in different applications, and actual performance may vary over time. All operating parameters, including "typicals," must be validated for each customer application by customer's technical experts. NXP does not convey any license under its patent rights nor the rights of others. NXP sells products pursuant to standard terms and conditions of sale, which can be found at the following address: nxp.com/SalesTermsandConditions.

Right to make changes - NXP Semiconductors reserves the right to make changes to information published in this document, including without limitation specifications and product descriptions, at any time and without notice. This document supersedes and replaces all information supplied prior to the publication hereof.

Security — Customer understands that all NXP products may be subject to unidentified or documented vulnerabilities. Customer is responsible for the design and operation of its applications and products throughout their lifecycles to reduce the effect of these vulnerabilities on customer's applications and products. Customer's responsibility also extends to other open and/or proprietary technologies supported by NXP products for use in customer's applications. NXP accepts no liability for any vulnerability. Customer should regularly check security updates from NXP and follow up appropriately. Customer shall select products with security features that best meet rules, regulations, and standards of the intended application and make the ultimate design decisions regarding its products and is solely responsible for compliance with all legal, regulatory, and security related requirements concerning its products, regardless of any information or support that may be provided by NXP. NXP has a Product Security Incident Response Team (PSIRT) (reachable at PSIRT@nxp.com) that manages the investigation, reporting, and solution release to security vulnerabilities of NXP products.

NXP, the NXP logo, NXP SECURE CONNECTIONS FOR A SMARTER WORLD, COOLFLUX,EMBRACE, GREENCHIP, HITAG, ICODE, JCOP, LIFE, VIBES, MIFARE, MIFARE CLASSIC, MIFARE DESFire, MIFARE PLUS, MIFARE FLEX, MANTIS, MIFARE ULTRALIGHT, MIFARE4MOBILE, MIGLO, NTAG, ROADLINK, SMARTLX, SMARTMX, STARPLUG, TOPFET, TRENCHMOS, UCODE, Freescale, the Freescale logo, AltiVec, CodeWarrior, ColdFire, ColdFire+, the Energy Efficient Solutions logo, Kinetis, Layerscape, MagniV, mobileGT, PEG, PowerQUICC, Processor Expert, QorlQ, QorlQ Qonverge, SafeAssure, the SafeAssure logo, StarCore, Symphony, VortiQa, Vybrid, Airfast, BeeKit, BeeStack, CoreNet, Flexis, MXC, Platform in a Package, QUICC Engine, Tower, TurboLink, EdgeScale, EdgeLock, elQ, and Immersive3D are trademarks of NXP B.V. All other product or service names are the property of their respective owners. AMBA, Arm, Arm7, Arm7TDMI, Arm9, Arm11, Artisan, big.LITTLE, Cordio, CoreLink, CoreSight, Cortex, DesignStart, DynamIQ, Jazelle, Keil, Mali, Mbed, Mbed Enabled, NEON, POP, RealView, SecurCore, Socrates, Thumb, TrustZone, ULINK, ULINK2, ULINK-PLUS, ULINKpro, µVision, Versatile are trademarks or registered trademarks of Arm Limited (or its subsidiaries) in the US and/or elsewhere. The related technology may be protected by any or all of patents, copyrights, designs and trade secrets. All rights reserved. Oracle and Java are registered trademarks of Oracle and/or its affiliates. The Power Architecture and Power.org word marks and the Power and Power.org logos and related marks are trademarks and service marks licensed by Power.org. M, M Mobileye and other Mobileye trademarks or logos appearing herein are trademarks of Mobileye Vision Technologies Ltd. in the United States, the EU and/or other jurisdictions.

© NXP B.V. 2021.

All rights reserved.

For more information, please visit: http://www.nxp.com For sales office addresses, please send an email to: salesaddresses@nxp.com

> Date of release: 08 November 2021 Document identifier: MCUOTASBLSFWUG

arm