Document information

<table>
<thead>
<tr>
<th>Information</th>
<th>Content</th>
</tr>
</thead>
<tbody>
<tr>
<td>Keywords</td>
<td>EdgeLock SE05x, Plug &amp; Trust middleware, i.MX 8M</td>
</tr>
<tr>
<td>Abstract</td>
<td>This document explains how to get started with the EdgeLock SE05x Plug &amp; Trust middleware using the EdgeLock SE05x development boards and i.MX 8M board. It provides detailed instructions for connecting the boards, installing the software, running the EdgeLock SE05x Plug &amp; Trust project examples and executing the ssscli tool</td>
</tr>
</tbody>
</table>
Revision history

<table>
<thead>
<tr>
<th>Revision number</th>
<th>Date</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.0</td>
<td>2020-10-21</td>
<td>First document release.</td>
</tr>
<tr>
<td>1.1</td>
<td>2020-12-07</td>
<td>Updated to latest template and fixed broken links.</td>
</tr>
<tr>
<td>1.2</td>
<td>2022-03-28</td>
<td>Add EdgeLock SE050E and EdgeLock A5000 product variants. Update Table 1, Figure 1 and Figure 2. Update SD card flash instructions in Section 4.1. Add note in Section 5 (step 6). Add Section Section 6 Product specific CMake build settings Add Section Section 7 Binding EdgeLock SE05x to a host using Platform SCP. Add Section Section 8 Manage access from multiple Linux processes to the EdgeLock SE05x. Added ssscli installation instructions in Section 9 Updated middleware build instructions in Section 10</td>
</tr>
<tr>
<td>1.3</td>
<td>2022-09-12</td>
<td>Update to EdgeLock SE Plug &amp; Trust Middleware version 04.02.xx. Update Section 6 Product specific CMake build settings. Update Section 7 Binding EdgeLock SE05x to a host using Platform SCP.</td>
</tr>
</tbody>
</table>
1 How to use this document

The Plug & Trust middleware includes a set of project examples that demonstrate the use of EdgeLock SE05x product family in the latest IoT security use cases. This document provides detailed instructions to run project examples for EdgeLock SE05x in i.MX 8M board. The main body of this document should be used in this sequence:

1. Order board samples. Section 2 contains the ordering details of the demo boards required in this document;
2. Prepare the hardware. Section 3 describes how to setup the OM-SE05xARD and i.MX 8M boards.
3. Flash the microSD card image. Section 4 describes how to create a micro-SD card with the Linux image with the pre-installed Plug & Trust middleware.
4. Run project examples. Section 5 describes how to run EdgeLock SE05x included in Plug & Trust middleware.

Supplementary material is provided in the appendices.
2 Hardware required

The EdgeLock SE05x works as an auxiliary security device attached to a host controller, communicating with through an I²C interface. To follow the instructions provided in this document, you need an EdgeLock SE05x development board and a i.MX 8M MCU board, acting as a host controller.

EdgeLock SE05x development boards ordering details

The EdgeLock SE05x and EdgeLock A5000 product support packages are providing development boards for evaluating EdgeLock SE05x and EdgeLock A5000 features. Select the development board of the product you want to evaluate. Table 1 details the ordering details of the EdgeLock SE05x and EdgeLock A5000 development boards.

Table 1. EdgeLock SE05x development boards.

<table>
<thead>
<tr>
<th>Part number</th>
<th>12NC</th>
<th>Description</th>
<th>Picture</th>
</tr>
</thead>
<tbody>
<tr>
<td>OM-SE050ARD-E</td>
<td>9354 332 66598</td>
<td>SE050E Arduino® compatible development kit</td>
<td><img src="image" alt="SE050ARD-E" /></td>
</tr>
<tr>
<td>OM-SE050ARD-F</td>
<td>9354 357 63598</td>
<td>SE050 Arduino® compatible development kit</td>
<td><img src="image" alt="SE050ARD-F" /></td>
</tr>
<tr>
<td>OM-SE050ARD</td>
<td>9353 832 82598</td>
<td>SE050F Arduino® compatible development kit</td>
<td><img src="image" alt="SE050ARD" /></td>
</tr>
<tr>
<td>OM-SE051ARD</td>
<td>9353 991 87598</td>
<td>SE051 Arduino® compatible development kit</td>
<td><img src="image" alt="SE051ARD" /></td>
</tr>
<tr>
<td>OM-A5000ARD</td>
<td>9354 243 19598</td>
<td>A5000 Arduino® compatible development kit</td>
<td><img src="image" alt="A5000ARD" /></td>
</tr>
</tbody>
</table>

Note: The pictures in this guide will show EdgeLock SE05xE, but all boards in Table 1 can be used as well with the same hardware configuration.

i.MX 8M MCU board ordering details

Table 2 details the ordering details for the i.MX 8M board.
## Table 2. i.MX 8M development kit details

<table>
<thead>
<tr>
<th>Part number</th>
<th>12NC</th>
<th>Content</th>
<th>Picture</th>
</tr>
</thead>
<tbody>
<tr>
<td>MCIMX8M-EVKB</td>
<td>935378743598</td>
<td>i.MX 8M evaluation kit</td>
<td><img src="image-url" alt="i.MX 8M evaluation kit" /></td>
</tr>
</tbody>
</table>
3 Boards setup

This section explains how to prepare the OM-SE05xARD boards and i.MX 8M board to run the EdgeLock SE05x Plug & Trust middleware project examples. Follow these steps:

1. Connect the OM-SE05xARD to the i.MX 8M board. The i.MX 8M board does not come with an Arduino connector, so you have to connect the Arduino shield of the OM-SE05xARD board to the J801_I2C connector of the i.MX 8M board using wires. Table 3 details the jumper connection.

<table>
<thead>
<tr>
<th>OM-SE05xARD (# jumper - # pin)</th>
<th>I.MX 8M (# jumper - # pin)</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>J2 - Pin 10</td>
<td>J801 - Pin 1</td>
<td>SCL</td>
</tr>
<tr>
<td>J2 - Pin 7</td>
<td>J801 - Pin 2</td>
<td>Ground</td>
</tr>
<tr>
<td>J2 - Pin 9</td>
<td>J801 - Pin 3</td>
<td>SDA</td>
</tr>
<tr>
<td>J8 - Pin 3</td>
<td>J801 - Pin 5</td>
<td>3.3 Volt supply</td>
</tr>
</tbody>
</table>

Table 3. Wire connection OM-SE05xARD - i.MX 8M

and Figure 1 illustrate the wiring between boards.
2. Make sure the jumper settings in your OM-SE05xARD board are configured as shown in Figure 2:

![Figure 2. Jumper configuration for i.MX 8M board](image)

*Note:* For more information about the jumper settings, refer to AN13539 OM-SE05xARD hardware overview.
4 Software setup

The software setup consists of:

1. Preparing a micro-SD card with the pre-compiled Linux image with the preinstalled Plug & Trust middleware for i.MX 8M board, as described in Section 4.1.
2. Installing the USB to UART Bridge VCOM driver in your laptop, as described in Section 4.2.
3. Installing TeraTerm terminal application, as described in Section 4.3.
4. Booting the i.MX 8M board, as described in Section 4.4.

4.1 Micro-SD card preparation

To prepare the micro-sd card with the pre-compiled Linux image that includes the Plug & Trust middleware, you need to:

1. Download from NXP website the Plug & Trust middleware SD Card Image. This image contains the Plug & Trust middleware pre-installed with already some examples on a bootable SD Card Image. 
   Note: In case the image provided by NXP does not suit your use case, you can find the explanation of how to create a card image using Yocto in the Plug & Trust middleware documentation (simw-top/doc/dev-platforms/platform_imx8_linux.html).
2. Download and install Win32 Disk Imager software. Win32 Disk Imager is a Windows open source program to format SD card images. Instead of Win 32 Disk Imager, you could also use any other software for this operation.
3. Plug your micro-SD card in your laptop.
4. Open Win32 Disk Imager, (1) select from your file system the pre-compiled Linux image you downloaded from the website, and (2) click on the Write button as shown in Figure 3. **Note:** the SD card image might come in a ZIP or BZ2 package. Make sure to unzip the package first with e.g. 7-ZIP before selecting the uncompressed image file in Win32 Disk Imager.

![Win32 Disk Imager](image)

Figure 3. Micro-SD card preparation with Win32 Disk Imager software

4.2 Drivers

To install the i.MX 8M drivers, follow these steps:
1. Plug the power supply and connect the USB cable to your laptop as shown in Figure 4.

Figure 4. Plug the power supply and connect the USB cable to your laptop

2. Download the **USB to UART Bridge VCOM driver** for your processor (either 32 or 64 bits). Install the driver by following the setup wizard until it is finished.

3. Unplug and plug your board.
4. Go to your Device Manager, and check that your board is recognized and assigned to a port number (COMxx). Write down the assigned port number (COMxx) as it is needed in the next steps. Your Device Manager should look like Figure 5. 

**Note:** If you see more than one COM port, use the one denoted as Enhanced COM port.

![Device Manager](image)

**Figure 5.** Check that i.MX 8M board is recognized in Device Manager

4.3 Terminal setup

We need to install a terminal application, for instance TeraTerm, to communicate and view the serial output of the i.MX 8M board from our laptop. To setup TeraTerm application:

1. Download TeraTerm and run the installer.
2. Launch TeraTerm, click the **Serial** option and select from the dropdown list the Enhanced COM port number assigned to your i.MX 8M board (In this case COM11) as shown in Figure 6. Click the OK button to confirm the setup. If you cannot see the
serial port of the board, your i.MX 8M board might not be recognized. In that case, please repeat the driver installation process described in Section 4.2.

**Note:** if you can’t see the serial port in TeraTerm, make sure that the board is plugged in with the USB cable to the PC and restart TeraTerm.

3. Go to Setup → Serial Port and configure the terminal to **115200 Speed, 8 data bits, no parity and 1 stop bit** as shown in Figure 7. Click on **New Setting** to confirm the configuration.

4.4 Booting the i.MX 8M

To boot the i.MX 8M, follow the steps shown in Figure 8 and Figure 9:

1. Insert the micro-SD card with the pre-compiled Linux image into the i.MX 8M card slot.
2. Configure the board switches as follows:
   - SW801 (Boot Device Select Switch): ON, ON, OFF, OFF (from 1-4 bit)
   - SW802 (Boot Mode Select Switch): ON, OFF (from 1-2 bit)
3. Connect the board to the power supply
4. Connect the board to your laptop using a USB cable and make sure that TeraTerm serial port is configured (see Section 4.3).
5. Turn on the power supply switch to boot up the board.

Figure 8. Booting the i.MX 8M
6. During the boot process, the operating system status information will be displayed in TeraTerm as shown in Figure 9. When the process is complete, the user can login with the following credentials:

- Account name: root
- Password: not required

![Figure 9. Sign in in the OS](image)

**Note:** In the case that the precompiled Plug & Trust middleware version does not suit your use case and a different version is needed, it must be downloaded and build into the device. For more information refer to Appendix B.
5 Run preinstalled Plug & Trust middleware test examples

The Plug & Trust middleware comes with several test examples used to verify atomic EdgeLock SE05x security IC features. This section explains how to run the Plug & Trust middleware test example called `se05x_Minimal`.

**Note:** The default build configuration of the Plug & Trust middleware ≤ V04.01.0x generates code for the OM-SE050ARD development board. You need to adapt the CMake settings in case you are using a different EdgeLock secure element development board or a different secure element product IC. The settings are described in Section 6.

1. After booting the board, you will be in the `/usr/local/bin` directory of the file system. To see the list of examples that are available in this directory use the following command: `Send > ls -l /usr/local/bin`

The TeraTerm window should be similar to Figure 10:

![TeraTerm window](image)

**Figure 10.** Go to the Plug & Trust middleware test example directory
2. Execute the `se05x_Minimal` test example. This test example outputs the memory left in EdgeLock SE05x security IC. Send `> se05x_Minimal`. The TeraTerm logs should indicate the available memory in EdgeLock SE05x security IC as can be seen in Figure 11 (in this case, 20820).

![Figure 11. Run se05x_minimal test example](image)
6 Product specific CMake build settings

The NXP Plug & Trust middleware supports the SE05x Secure Elements, the A5000 Secure Authenticator, and the legacy A71CH products.

The EdgeLock Plug & Trust middleware is delivered with CMake files that include the set of directives and instructions describing the project's source files and the build targets. The CMake files are used to select a dedicated EdgeLock product IC and the corresponding IoT applet or Authenticator application.

The SE050 product identification can be obtained as described in AN12436 chapter 1 Product Information. AN12973 describes the same procedure for the SE051 product family.

The following tables show the required PTMN CMake options to build a dedicated product variant. The SSSFTR_SE05X_RSA CMake option is used to optimize the memory footprint for product variants that do not support RSA.

### Table 4. CMake Settings for SE050E product variants

<table>
<thead>
<tr>
<th>Variant</th>
<th>OEF ID</th>
<th>PTMW_Applet</th>
<th>PTMW_FIPS</th>
<th>PTMW_SE05X_Ver</th>
<th>PTMW_SE05X_Auth</th>
<th>PTMW_SCP</th>
<th>SSSFTR_SE05X_RSA</th>
</tr>
</thead>
<tbody>
<tr>
<td>SE050E Dev. Board</td>
<td>A921</td>
<td>SE050_E</td>
<td>None</td>
<td>07_02</td>
<td>any option</td>
<td>None or SCP03_SSS</td>
<td>disabled</td>
</tr>
<tr>
<td>OM-SE050ARD-E</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SE050E2</td>
<td>A921</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

### Table 5. CMake Settings for SE050F product variants

<table>
<thead>
<tr>
<th>Variant</th>
<th>OEF ID</th>
<th>PTMW_Applet</th>
<th>PTMW_FIPS</th>
<th>PTMW_SE05X_Ver</th>
<th>PTMW_SE05X_Auth</th>
<th>PTMW_SCP</th>
<th>SSSFTR_SE05X_RSA</th>
</tr>
</thead>
<tbody>
<tr>
<td>SE050F Dev.Board</td>
<td>A92A</td>
<td>SE05X_C</td>
<td>SE050</td>
<td>03_XX</td>
<td>PlatfSCP03 or UserID_PlatformSCP03 or AESKey_PlatformSCP03 or ECKey_PlatformSCP03</td>
<td>SCP03_SSS</td>
<td>enabled</td>
</tr>
<tr>
<td>OM-SE050ARD-F</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SE050F2</td>
<td>A92A</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

### Table 6. CMake Settings for SE050 Previous Generation product variants

<table>
<thead>
<tr>
<th>Variant</th>
<th>OEF ID</th>
<th>PTMW_Applet</th>
<th>PTMW_FIPS</th>
<th>PTMW_SE05X_Ver</th>
<th>PTMW_SE05X_Auth</th>
<th>PTMW_SCP</th>
<th>SSSFTR_SE05X_RSA</th>
</tr>
</thead>
<tbody>
<tr>
<td>SE050A1</td>
<td>A204</td>
<td>SE05X_A</td>
<td>None</td>
<td>03_XX</td>
<td>any option</td>
<td>None or SCP03_SSS</td>
<td>disabled</td>
</tr>
<tr>
<td>SE050A2</td>
<td>A205</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SE050B1</td>
<td>A202</td>
<td>SE05X_B</td>
<td>None</td>
<td>03_XX</td>
<td>any option</td>
<td>None or SCP03_SSS</td>
<td>enabled</td>
</tr>
<tr>
<td>SE050B2</td>
<td>A203</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
### Table 6. CMake Settings for SE050 Previous Generation product variants...continued

<table>
<thead>
<tr>
<th>Variant</th>
<th>OEF ID</th>
<th>PTMW_Applet</th>
<th>PTMW_FIPS</th>
<th>PTMW_SE05X_Ver</th>
<th>PTMW_SE05X_Auth</th>
<th>PTMW_SCP</th>
<th>SSSFTR_SE05X_RSA</th>
</tr>
</thead>
<tbody>
<tr>
<td>SE050C1</td>
<td>A200</td>
<td>SE05X_C</td>
<td>None</td>
<td>03_XX</td>
<td>any option</td>
<td>None or SCP03_SSS</td>
<td>enabled</td>
</tr>
<tr>
<td>SE050C2</td>
<td>A201</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>OM-SE050ARD</td>
<td>A1F4</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SE050F2</td>
<td>A77E[1]</td>
<td>SE05X_C</td>
<td>SE050</td>
<td>03_XX</td>
<td>PlatfSCP03</td>
<td>SCP03_SSS</td>
<td>enabled</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>or UserID_PlatformSCP03</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>or AESKey_PlatformSCP03</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>or ECKey_PlatformSCP03</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

[1] All SE050F2 with variant A77E have date code in year 2021. All the SE050F2 with date code in the year 2022 have the variant identifier A92A.

### Table 7. CMake Settings for SE051 product variants

<table>
<thead>
<tr>
<th>Variant</th>
<th>OEF ID</th>
<th>PTMW_Applet</th>
<th>PTMW_FIPS</th>
<th>PTMW_SE05X_Ver</th>
<th>PTMW_SE05X_Auth</th>
<th>PTMW_SCP</th>
<th>SSSFTR_SE05X_RSA</th>
</tr>
</thead>
<tbody>
<tr>
<td>SE051A2</td>
<td>A920</td>
<td>SE05X_A</td>
<td>None</td>
<td>07_02</td>
<td>any option</td>
<td>None or SCP03_SSS</td>
<td>disabled</td>
</tr>
<tr>
<td>SE051C2</td>
<td>A8FA</td>
<td>SE05X_C</td>
<td>None</td>
<td>07_02</td>
<td>any option</td>
<td>None or SCP03_SSS</td>
<td>enabled</td>
</tr>
<tr>
<td>SE051W2</td>
<td>A739</td>
<td>SE05X_C</td>
<td>None</td>
<td>07_02</td>
<td>any option</td>
<td>None or SCP03_SSS</td>
<td>enabled</td>
</tr>
<tr>
<td>SE051A3</td>
<td>A565</td>
<td>SE05X_A</td>
<td>None</td>
<td>06_00</td>
<td>any option</td>
<td>None or SCP03_SSS</td>
<td>disabled</td>
</tr>
<tr>
<td>SE051C2</td>
<td>A564</td>
<td>SE05X_C</td>
<td>None</td>
<td>06_00</td>
<td>any option</td>
<td>None or SCP03_SSS</td>
<td>enabled</td>
</tr>
</tbody>
</table>
Table 8. CMake Settings for A5000 product variants

<table>
<thead>
<tr>
<th>Variant</th>
<th>OEF ID</th>
<th>PTMW_Applet</th>
<th>PTMW_FIPS</th>
<th>PTMW_SE05X_Ver</th>
<th>PTMW_SE05X_Auth</th>
<th>PTMW_SCP</th>
<th>SSSFTR_SE05X_RSA</th>
</tr>
</thead>
<tbody>
<tr>
<td>OM-A5000ARD</td>
<td>A736</td>
<td>AUTH</td>
<td>None</td>
<td>07_02</td>
<td>any option</td>
<td>None</td>
<td>disabled</td>
</tr>
<tr>
<td>A5000</td>
<td>A736</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>None or SCP03_SSS</td>
<td></td>
</tr>
</tbody>
</table>

6.1 Example: SE050E CMake build settings

To build the Plug & Trust Middleware to support the SE050E Secure Element applet the following CMake setting needs to be modified before building the middleware according to Table 4:

- Select **SE050_E** for the **CMake option** `PTMW_Applet`.
- Select **None** for the **CMake option** `PTMW_FIPS`.
- Select **07_02** for the **CMake option** `PTMW_SE05X_Ver`.
- Disable the **CMake option** `SSSFTR_SE05X_RSA`.

In this example we use plain communication. Plain communication for the example execution is enabled by selecting the following options:

- Select **None** for the **CMake option** `PTMW_SE05X_Auth`.
- Select **None** for the **CMake option** `PTMW_SCP`.

How to enable Platform SCP is described in How to enable Platform SCP in the CMake-based build system.

Run the following commands to update the CMake settings and rebuild the Plug & Trust middleware:

```
cd ~/se_mw/simw-top_build/imx_native_se050_t1oi2c
cmake -DPTMW_Applet=SE050_E -DPTMW_FIPS=None -DPTMW_SE05X_Ver=07_02 -DSSSFTR_SE05X_RSA=0 -DPTMW_SCP=None -DPTMW_SE05X_Auth=None .
cmake --build .
sudo make install
sudo ldconfig /usr/local/lib/
```
7 Binding EdgeLock SE05x to a host MCU/MPU using Platform SCP

Binding is a process to establish a pairing between the IoT device host MPU/MPU and EdgeLock SE05x, so that only the paired MPU/MPU is able to use the services offered by the corresponding EdgeLock SE05x and vice versa.

A mutually authenticated, encrypted channel will ensure that both parties are indeed communicating with the intended recipients and that local communication is protected against local attacks, including man-in-the-middle attacks aimed at intercepting the communication between the MPU/MPU and the EdgeLock SE05x and physical tampering attacks aimed at replacing the host MPU/MPU or EdgeLock SE05x.

EdgeLock SE05x natively supports Global Platform Secure Channel Protocol 03 (SCP03) for this purpose. PlatformSCP uses SCP03 and can be enabled to be mandatory.

This chapter describes the required steps to enable Platform SCP in the middleware for EdgeLock SE05x.

The following topics are discussed:

- **Section 7.1 Introduction to the Global Platform Secure Channel Protocol 03 (SCP03)**
- **How to configure the product specific default Platform SCP keys** How to configure the EdgeLock SE05x product specific SCP keys in the Plug & Trust middleware
- **How to enable Platform SCP in the CMake-based build system** How to enable Platform SCP in the Plug & Trust middleware

7.1 Introduction to the Global Platform Secure Channel Protocol 03 (SCP03)

The Secure Channel Protocol SCP03 authenticates and protects locally the bidirectional communication between host and EdgeLock SE05x against eavesdropping on the physical I2C interface.

EdgeLock SE05x can be bound to the host by injecting in both the host and EdgeLock SE05x the same unique SCP03 AES key-set and by enabling the Platform SCP feature in the Plug & Trust middleware. The AN12662 Binding a host device to EdgeLock SE05x describes in detail the concept of secure binding.

SCP03 is defined in Global Platform Secure Channel Protocol '03' - Amendment D v1.2 specification.

SCP03 can provide the following three security goals:

- **Mutual authentication (MA)**
  - Mutual authentication is achieved through the process of initiating a Secure Channel and provides assurance to both the host and the EdgeLock SE05x entity that they are communicating with an authenticated entity.

- **Message Integrity**
  - The Command- and Response-MAC are generated by applying the CMAC according to NIST SP 800-38B.

- **Confidentiality**
  - The message data field is encrypted across the entire data field of the command message to be transmitted to the EdgeLock SE05x, and across the response transmitted from the EdgeLock SE05x.
The SCP03 secure channel is set up via the EdgeLock SE05x Java Card OS Manager using the standard ISO7816-4 secure channel APDUs.

The establishment of an SCP03 channel requires three static 128-bit AES keys shared between the two communicating parties: Key-ENC, Key-MAC and Key-DEK. These keys are stored in the Java Card Supplementary Security Domain (SSD) and not in the secure authenticator applet.

Key-ENC and Key-MAC keys are used during the SCP03 channel establishment to generate the session keys. Session Keys are generated to ensure that a different set of keys are used for each Secure Channel Session to prevent replay attacks.

Key-ENC is used to derive the session key S-ENC. The S-ENC key is used for encryption/decryption of the exchanged data. The session keys S-MAC and R-MAC are derived from Key-MAC and used to generate/verify the integrity of the exchanged data (C-APDU and R-APDU).

Key-DEK key is used to encrypt new SCP03 keys in case they get updated.

Table 9. Static SCP03 keys

<table>
<thead>
<tr>
<th>Key</th>
<th>Description</th>
<th>Usage</th>
<th>Key Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>Key-ENC</td>
<td>Static Secure Channel Encryption Key</td>
<td>Generate session key for Decryption/Encryption (AES)</td>
<td>AES 128</td>
</tr>
<tr>
<td>Key-MAC</td>
<td>Static Secure Channel Message Authentication Code Key</td>
<td>Generate session key for Secure Channel authentication and Secure Channel MAC Verification/Generation (AES)</td>
<td>AES 128</td>
</tr>
<tr>
<td>Key-DEK</td>
<td>Data Encryption Key</td>
<td>Sensitive Data Decryption (AES)</td>
<td>AES 128</td>
</tr>
</tbody>
</table>

The session key generation is performed by the Plug & Trust middleware host crypto.

Table 10. SCP03 session keys

<table>
<thead>
<tr>
<th>Key</th>
<th>Description</th>
<th>Usage</th>
<th>Key Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>S-ENC</td>
<td>Session Secure Channel Encryption Key</td>
<td>Used for data confidentiality</td>
<td>AES 128</td>
</tr>
<tr>
<td>S-MAC</td>
<td>Secure Channel Message Authentication Code Key for Command</td>
<td>Used for data and protocol integrity</td>
<td>AES 128</td>
</tr>
<tr>
<td>S-RMAC</td>
<td>Secure Channel Message Authentication Code Key for Response</td>
<td>User for data and protocol integrity</td>
<td>AES 128</td>
</tr>
</tbody>
</table>

Note: For further details please refer to Global Platform Secure Channel Protocol '03' - Amendment D v1.2.
5.2 How to configure the product specific default Platform SCP keys

The default Platform SCP key values are described for the EdgeLock SE05x product variants in AN12436 and for the EdgeLock A5000 variants in AN12973.

For evaluation purpose, the Platform SCP keys can be defined either in the Plug & Trust middleware source code (see Section 7.2.1) or provided as text file (see Section 7.2.2).

7.2.1 Defining the default Platform SCP keys in the Plug & Trust middleware source code

The Plug & Trust middleware header file ex_sss_tp_scp03_keys.h contains the default values of all EdgeLock SE05x, EdgeLock A5000, A5000 and A71CH product variants.
The `ex_sss_tp scp03_keys.h` header file location in the following location:

`~/.se_mw/simw-top/sss/ex/inc/`

---

**Figure 14. Default Platform SCP keys are defined in `ex_sss_tp scp03_keys.h`**

The `fsl_sss_ftr.h.in` file includes options to select one of the predefined default Platform SCP keys. This file is located in: `~/se_mw/simw-top/sss/inc`. Select the desired value of the compilation option by setting exclusively the corresponding C-preprocessor define `SSS_PFSCP_ENABLE_xx` to 1 (enable).

All other values for the same option (represented by C-preprocessor defines `SSS_PFSCP_ENABLE_xx`) must be set to 0.
Figure 15. Select the actual Platform SCP keys in infsl_sss_ftr.h

The Plug & Trust Middleware uses a feature file to select/detect used/enabled features within the middleware stack. The file `fsl_sss_ftr.h` is automatically generated into the used build directory. CMake is overwriting the `fsl_sss_ftr.h` file every time CMake is invoked. CMake is using the SCP key settings of the `fsl_sss_ftr.h.in` file as input to generate the `fsl_sss_ftr.h` file. You do not have to manually edit the `fsl_sss_ftr.h` feature file. Selections from CMake edit cache automatically updates the generated feature file.

**Note:** The Platform SCP key selection in the `fsl_sss_ftr.h.in` CMake input file is persistent.

The location of the generated `fsl_sss_ftr.h` feature header file is: `~/se_mw/simw-top_build/imx_native_se050_t1oi2c`

The following tables contain the Platform SCP key header file define to be set to 1 (enable) for the different secure element and secure authenticator product variants.

**Table 11. Platform SCP key define prefix for SE050E product variants**

<table>
<thead>
<tr>
<th>Variant</th>
<th>OEF ID</th>
<th>Platform SCP key define to be set to ‘1’</th>
</tr>
</thead>
<tbody>
<tr>
<td>SE050E Dev. Board OM-SE050ARD-E</td>
<td>A921</td>
<td>SSS_PFSCP_ENABLE_SE050E_0001A921</td>
</tr>
<tr>
<td>SE050E2</td>
<td>A921</td>
<td>SSS_PFSCP_ENABLE_SE050E_0001A921</td>
</tr>
</tbody>
</table>
### Table 12. Platform SCP key define prefix for SE050F product variants

<table>
<thead>
<tr>
<th>Variant</th>
<th>OEF ID</th>
<th>Platform SCP key define to be set to ‘1’</th>
</tr>
</thead>
<tbody>
<tr>
<td>SE050F Dev.Board</td>
<td>A92A</td>
<td>SSS_PFSCP_ENABLE_SE050F2_0001A92A</td>
</tr>
<tr>
<td>OM-SE050ARD-F</td>
<td></td>
<td></td>
</tr>
<tr>
<td>SE050F2</td>
<td>A92A</td>
<td>SSS_PFSCP_ENABLE_SE050F2_0001A92A</td>
</tr>
</tbody>
</table>

### Table 13. Platform SCP key define prefix for SE050 Previous Generation product variants

<table>
<thead>
<tr>
<th>Variant</th>
<th>OEF ID</th>
<th>Platform SCP key define to be set to ‘1’</th>
</tr>
</thead>
<tbody>
<tr>
<td>SE050A1</td>
<td>A204</td>
<td>SSS_PFSCP_ENABLE_SE050A1</td>
</tr>
<tr>
<td>SE050A2</td>
<td>A205</td>
<td>SSS_PFSCP_ENABLE_SE050A2</td>
</tr>
<tr>
<td>SE050B1</td>
<td>A202</td>
<td>SSS_PFSCP_ENABLE_SE050B1</td>
</tr>
<tr>
<td>SE050B2</td>
<td>A203</td>
<td>SSS_PFSCP_ENABLE_SE050B2</td>
</tr>
<tr>
<td>SE050C1</td>
<td>A200</td>
<td>SSS_PFSCP_ENABLE_SE050C1</td>
</tr>
<tr>
<td>SE050C2</td>
<td>A201</td>
<td>SSS_PFSCP_ENABLE_SE050C2</td>
</tr>
<tr>
<td>SE050 Dev Board</td>
<td>A1F4</td>
<td>SSS_PFSCP_ENABLE_SE050_DEVKIT</td>
</tr>
<tr>
<td>OM-SE050ARD</td>
<td></td>
<td></td>
</tr>
<tr>
<td>SE050F2</td>
<td>A77E[1]</td>
<td>SSS_PFSCP_ENABLE_SE050F2</td>
</tr>
</tbody>
</table>

[1] All SE050F2 with variant A77E have date code in year 2021. All the SE050F2 with date code in the year 2022 have the variant identifier A92A.

### Table 14. Platform SCP key define prefix for SE051 product variants

<table>
<thead>
<tr>
<th>Variant</th>
<th>OEF ID</th>
<th>Platform SCP key define to be set to ‘1’</th>
</tr>
</thead>
<tbody>
<tr>
<td>SE051A2</td>
<td>A920</td>
<td>SSS_PFSCP_ENABLE_SE051A_0001A920</td>
</tr>
<tr>
<td>SE051C2</td>
<td>A8FA</td>
<td>SSS_PFSCP_ENABLE_SE051C_0005A8FA</td>
</tr>
<tr>
<td>SE051W2</td>
<td>A739</td>
<td>SSS_PFSCP_ENABLE_SE051W_0005A739</td>
</tr>
<tr>
<td>SE051A2</td>
<td>A565</td>
<td>SSS_PFSCP_ENABLE_SE051A2</td>
</tr>
<tr>
<td>SE051C2</td>
<td>A564</td>
<td>SSS_PFSCP_ENABLE_SE051C2</td>
</tr>
</tbody>
</table>

### Table 15. Platform SCP key define prefix for A5000 product variants

<table>
<thead>
<tr>
<th>Variant</th>
<th>OEF ID</th>
<th>Platform SCP key define to be set to ‘1’</th>
</tr>
</thead>
<tbody>
<tr>
<td>A5000 Dev. Board</td>
<td>A736</td>
<td>SSS_PFSCP_ENABLE_A5000_0004A736</td>
</tr>
<tr>
<td>OM-A5000ARD</td>
<td></td>
<td></td>
</tr>
<tr>
<td>A5000</td>
<td>A736</td>
<td>SSS_PFSCP_ENABLE_A5000_0004A736</td>
</tr>
</tbody>
</table>

#### 7.2.2 Defining the deault Platfrom SCP keys in a text file

For evaluation purpose the Plug & Trust middleware supports to store the Platform SCP key in a plain text file. For further details see Plug & Trust middleware documentation chapter 11.10 Using own Platform SCP03 keys.

The following Linux commands can be used to create the Platform SCP key text file (se050_Dev_Kit scp_keys.txt):

```bash
The Platform SCP key text file can be stored in any location. In this example the file is stored in: ~/se_mw/simw-top_build/imx_native_se050_t1oi2c/bin
cd ~/se_mw/simw-top_build/imx_native_se050_t1oi2c/bin
```
echo ENC D2DB63E7A0A5AED72A6460C4DFDCAF64 > se050E_scp_keys.txt
echo MAC 738D5B798ED241B0B24768514BFBA95B >> se050E_scp_keys.txt
echo DEK 6702DAC30942B2C85E7F47B42CED4E7F >> se050E_scp_keys.txt

Check the se050E_scp_keys.txt file content:
cat se050E_scp_keys.txt

The Linux environment variable EX_SSS_BOOTSCP03_PATH is used to define the Platform SCP key textfile (filename and location).

export EX_SSS_BOOTSCP03_PATH=/se_mw/simw-top_build/imx_native_se050_t1oi2c/bin/se050E_scp_keys.txt

Figure 16. EdgeLock SE05x Platform SCP plain text key file

Note: In this example the MCIMX8M-EVKB is used for evaluation purpose only. Because different host MCU/MPU platforms are providing different hardware security mechanisms to protect keys it is not in the scope of this document to demonstrate how to store the Platform SCP shared binding keys securely. For commercial deployment the secure storage of Platform SCP keys must be adapted accordingly.

7.3 How to enable Platform SCP in the CMake-based build system

To enable Platform SCP is required to rebuild the SDK with the following CMake options:

- Select SCP03_SSS for the CMake option PTMW_SCP.
- Select PlatfSCP03 for the CMake option PTMW_SE05X_Auth.

Run the following commands to update the CMake settings and rebuild the Plug & Trust middleware:

cd ~/se_mw/simw-top_build/imx_native_se050_t1oi2c
cmake -DPTMW_SE05X_Auth=PlatfSCP03 -DPTMW_SCP=SCP03_SSS .
cmake --build .
sudo make install
sudo ldconfig /usr/local/lib/
Figure 17. SE050E CMake Settings - PlatformSCP enabled

**Note:** In this document the MCIMX8M-EVK is used for evaluation purpose only. Because different host MCU/MPU platforms are providing different hardware security mechanisms to protect keys it is not in the scope of this document to demonstrate how to store the Platform SCP shared binding keys securely. For commercial deployment the secure storage of Platform SCP keys must be adapted accordingly.

In the next step we can verify if we successfully enabled Platform SCP. For this purpose we run again the se05x_minimal example:

```
  cd bin
  ./se05x_Minimal
```

**Figure 18** shows the log output in case the Platform SCP keys are defined in the Plug & Trust middleware source code (see Defining the default Platform SCP keys in the Plug & Trust middleware source code).

The log output for defining the Platform SCP keys via a text file (see Defining the default Platform SCP keys in a text file) is shown in **Figure 19**.
The Plug & Trust Middleware provides the following additional examples to rotate the PlatformSCP Keys and to mandate Platform SCP:

- **SE05X Rotate PlatformSCP Keys example**: Showcases authentication with default Platform SCP keys and the rotation (update) of those keys with user defined keys. The example documentation is available in the EdgeLock SE05x Plug & Trust Middleware documentation (/simw-top/doc/demos/se05x/se05x_RotatePlatformSCP03Keys/Readme.html). The example source code is available at /simw-top/demos/se05x/se05x_RotatePlatformSCP03Keys.

- **SE05X Mandate SCP example**: Showcases how to make Platform SCP authentication mandatory in EdgeLock SE05x. The example documentation is available in the EdgeLock SE05x Plug & Trust Middleware documentation (/simw-top/doc/demos/se05x/se05x_MandatePlatformSCP/Readme.html). The example source code is available at /simw-top/demos/se05x/se05x_MandatePlatformSCP.

- **SE05x AllowWithout PlatformSCP example**: This project demonstrates how to configure SE05X to allow without platform SCP. The example documentation is available in the EdgeLock SE05x Plug & Trust Middleware documentation (~/se_mw/simwtop/doc/demos/se05x/se05x_AllowWithoutPlatformSCP/Readme.html). The example source code is available at ~/se_mw/simw-top/demos/se05x/se05x_AllowWithoutPlatformSCP.
8 Manage access from multiple Linux processes to the EdgeLock SE05x

The Plug & Trust middleware provides the Access Manager to support concurrent access from multiple Linux processes to the EdgeLock SE05x IoT applet. The Access Manager can establish a connection to the EdgeLock SE05x IoT applet either as a plain connection or using Platform SCP.

Client processes are connecting over the JRCPv1 protocol to the Access Manager.

Please refer to the Plug & Trust middleware documentation chapter Access Manager: Manage access from multiple (Linux) processes to an SE05x IoT Applet for more details.
In Section 3 and Section 4 we have prepared the hardware setup and the software setups respectively. To validate that the whole process was done correctly and that your setup is fully operational, we are going to run the `ssscli` tool. This tool can be used to interact with the EdgeLock SE05x security IC without having to write any code.

To start the `ssscli` tool, send the commands shown in Figure 20:

1. Open the connection:
   
   **Send:** `>ssscli connect se05x t1oi2c none`

   **Note:** In case `ssscli` did not get installed automatically in the image, execute these steps to finish the installation first:
   
   ```
   > cd simw-top/pycli/src
   > python3 setup.py develop
   ```

2. Send the reset command:
   
   **Send:** `>ssscli se05x reset`

   **Figure 20. Start the ssscli tool**

   **Note:** If you see the following message: `WARNING:sss.connect:Session already open, close current session first` as shown in Figure 21, it means that you have a session open. To close it, send: (1) `> ssscli disconnect` and then send once again (2) `> ssscli connect se05x t1oi2c none` and later (3) `> ssscli se05x reset`:

   **Figure 21. Close an already opened session**
3. The SE05x ssscli tool supports several operations. To check which commands are supported by the ssscli tool:

(Figure 22) Send: > ssscli --help

![Figure 22. ssscli info](image)

4. Once you are done using the ssscli tool, close the session with the EdgeLock SE05x security IC:

(Figure 23) Send: > ssscli disconnect

![Figure 23. ssscli disconnect](image)

The ssscli tool uses the installed sss library (/usr/local/lib/libssapisw.so) for communication with the secure element. In order to be able to connect with ssscli to a specific interface type (e.g. T1oI2C or JRCP) or secure element type (A71CH or SE05x) the ssslibrary needs to be compiled and installed with this specific interface/secure element type selected in the compilation options. This can be done by recompiling the Plug & Trust middleware as shown in the third step of the Appendix B (Section 7.2).
10 Appendix B: Update Plug & Trust middleware

In this section it will be described the process needed for downloading, compiling and building a different Plug & Trust middleware version from the one preinstalled in the SD card image seen at Section 5.

10.1 Obtain the latest Plug & Trust middleware version

We need to compile the Plug & Trust middleware into the Linux software image we flashed before. Follow these steps:

1. Download latest version of the Plug & Trust middleware using this link.
2. Connect to the board using a terminal application such as TeraTerm and run the `ifconfig` command to determine the IP address of your board as shown in Figure 24. We will use the board IP address to transfer the Plug & Trust middleware using the SCP protocol.

3. Download and install a file transfer software supporting SCP such as WinSCP in case you are using a Windows host machine.
4. **Windows host**: open WinSCP and configure the SCP connection as shown in Figure 25. The Host name corresponds to the IP address of the board obtained previously. Click on the *Login* button to establish the connection.

![Login page of WinSCP](image)

**Figure 25. Connect to the board using SCP with WinSCP**

5. Once the connection is established, you should see on the right pane the board file system and, on the left pane the file system of the host machine. It is recommended to delete the folder named *se05x_mw_vxxx* in order to avoid confusion. Navigate to the folder where you downloaded the Plug & Trust middleware package and then drag
and drop the package to the right pane as shown in Figure 26. A copy of the Plug & Trust middleware file should now be in the `/home/root` folder of the board.

![Figure 26. Copy the middleware to the board](image)

10.2 Build the Plug & Trust middleware

To build the Plug & Trust middleware, open the TeraTerm and follow the steps listed below:
1. Unzip the Plug & Trust middleware file that you have transferred to the board in Section 10.1.
   Send `unzip <middleware_file_name>.zip` as shown in Figure 27.

![Figure 27. Unzip the middleware]
2. A new folder called `simw-top` should have been created in the root directory. You can now create the middleware CMake projects for the board as shown in Figure 28:

   (1) Navigate to the `simw-top/scripts` folder. Send `cd simw-top/scripts/`.
   (2) Send the command `python3 create_cmake_projects.py` and wait until the command is finished executing.

   **Note:** This command may take a few seconds to complete.

![Figure 28. Create CMake projects for the board](image)

3. Finally, compile and install the project as shown in Figure 29:

   (1) Navigate to the folder where the project has been generated. Send `cd ../../simw-top_build/imx_native_se050_t1oi2c`
   (2) Build the project. This might take some time. Send `cmake --build .`

   **Note:** The default build configuration of the Plug & Trust middleware ≤ V04.01.0x generates code for the OM-SE050ARD development board. You need to adapt the CMake settings in case you are using a different EdgeLock secure element.
development board or a different secure element product IC. The settings are described in Section 6.

**Note:** in the default CMake configuration the SCP03 protocol is not active. How to enable Platform SCP is described in Section 7.

(3) Install the project.
Send `sudo make install`

![Figure 29. Build and install the project](image)

(4) Refresh linker cache to let it find the newly installed libraries
Send `sudo ldconfig /usr/local/lib`
11 Legal information

11.1 Definitions

Draft — A draft status on a document indicates that the content is still under internal review and subject to formal approval, which may result in modifications or additions. NXP Semiconductors does not give any representations or warranties as to the accuracy or completeness of information included in a draft version of a document and shall have no liability for the consequences of use of such information.

11.2 Disclaimers

Limited warranty and liability — Information in this document is believed to be accurate and reliable. However, NXP Semiconductors does not give any representations or warranties, expressed or implied, as to the accuracy or completeness of such information and shall have no liability for the consequences of use of such information. NXP Semiconductors takes no responsibility for the content in this document if provided by an information source outside of NXP Semiconductors. In no event shall NXP Semiconductors be liable for any indirect, incidental, punitive, special or consequential damages (including profit loss, lost savings, business interruption, costs related to the removal or replacement of any products or rework charges) whether or not such damages are based on tort (including negligence), warranty, breach of contract or any other legal theory. Notwithstanding any damages that customer might incur for any reason whatsoever, NXP Semiconductors and its affiliates’ total liability to customer for all damages shall be limited to the greater of the amount actually paid by customer for the product or the product returns, and customer’s exclusive remedy for all of the foregoing shall be limited to the entire liability of NXP Semiconductors, its affiliates and their suppliers.

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.

Suitability for use — NXP Semiconductors products are not designed, authorized or warranted to be suitable for use in life support, life-critical or safety-critical systems or equipment, nor in applications where failure or malfunction of an NXP Semiconductors product can reasonably be expected to result in personal injury, death or severe property or environmental damage. NXP Semiconductors and its suppliers accept no liability for inclusion and/or use of NXP Semiconductors products in such equipment or applications and therefore such inclusion and/or use is at the customer’s own risk.

Applications — Applications that are described herein for any of these products are for illustrative purposes only. NXP Semiconductors makes no representation or warranty that such applications will be suitable for the specified use without further testing or modification. Customers are responsible for the design and operation of their applications and products using NXP Semiconductors products, and NXP Semiconductors accepts no liability for any assistance with applications or customer product design. It is customer’s sole responsibility to determine whether the NXP Semiconductors product is suitable and fit for the customer’s applications and products planned, as well as for the planned application and use of customer’s third party customer(s). Customers should provide appropriate design and operating safeguards to minimize the risks associated with their applications and products. NXP Semiconductors does not accept any liability related to any default, damage, costs or problem which is based on any weakness or default in the customer’s applications or products, or the application or use by customer’s third party customer(s). Customer is responsible for doing all necessary testing for the customer’s applications and products using NXP Semiconductors products in order to avoid a default of the applications and the products or of the application or use by customer’s third party customer(s). NXP does not accept any liability in this respect.

Export control — This document as well as the item(s) described herein may be subject to export control regulations. Export might require a prior authorization from competent authorities.

Evaluation products — This product is provided on an “as is” and “with all faults” basis for evaluation purposes only. NXP Semiconductors, its affiliates and their suppliers expressly disclaim all warranties, whether express, implied or statutory, including but not limited to the implied warranties of non-infringement, merchantability and fitness for a particular purpose. The entire risk as to the quality, or arising out of the use or performance, of this product remains with customer. In no event shall NXP Semiconductors, its affiliates or their suppliers be liable to customer for any special, indirect, consequential, punitive or incidental damages (including without limitation damages for loss of business, business interruption, loss of use, loss of data or information, and the like) arising out the use of or inability to use the product, whether or not based on tort (including negligence), strict liability, breach of contract, breach of warranty or any other theory, even if advised of the possibility of such damages. Notwithstanding any damages that customer might incur for any reason whatsoever (including without limitation, all damages referenced above and all direct or general damages), the entire liability of NXP Semiconductors, its affiliates and suppliers and customer’s exclusive remedy for all of the foregoing shall be limited to actual damages incurred by customer based on reasonable reliance up to the greater of the amount actually paid by customer for the product or five dollars (US$5.00). The foregoing limitations, exclusions and disclaimers shall apply to the maximum extent permitted by applicable law, even if any remedy fails of its essential purpose.

Translations — A non-English (translated) version of a document is for reference only. The English version shall prevail in case of any discrepancy between the translated and English versions.

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.

11.3 Trademarks

Notice: All referenced brands, product names, service names and trademarks are the property of their respective owners.
## Tables

<table>
<thead>
<tr>
<th>Tab.</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tab. 1</td>
<td>EdgeLock SE05x development boards.</td>
<td>4</td>
</tr>
<tr>
<td>Tab. 2</td>
<td>i.MX 8M development kit details</td>
<td>5</td>
</tr>
<tr>
<td>Tab. 3</td>
<td>Wire connection OM-SE05xARD - i.MX 8M</td>
<td>6</td>
</tr>
<tr>
<td>Tab. 4</td>
<td>CMake Settings for SE050E product variants</td>
<td>17</td>
</tr>
<tr>
<td>Tab. 5</td>
<td>CMake Settings for SE050F product variants</td>
<td>17</td>
</tr>
<tr>
<td>Tab. 6</td>
<td>CMake Settings for SE050 Previous Generation product variants</td>
<td>17</td>
</tr>
<tr>
<td>Tab. 7</td>
<td>CMake Settings for SE051 product variants</td>
<td>18</td>
</tr>
<tr>
<td>Tab. 8</td>
<td>CMake Settings for A5000 product variants</td>
<td>19</td>
</tr>
<tr>
<td>Tab. 9</td>
<td>Static SCP03 keys</td>
<td>21</td>
</tr>
<tr>
<td>Tab. 10</td>
<td>SCP03 session keys</td>
<td>21</td>
</tr>
<tr>
<td>Tab. 11</td>
<td>Platform SCP key define prefix for SE050E product variants</td>
<td>24</td>
</tr>
<tr>
<td>Tab. 12</td>
<td>Platform SCP key define prefix for SE050F product variants</td>
<td>25</td>
</tr>
<tr>
<td>Tab. 13</td>
<td>Platform SCP key define prefix for SE050 Previous Generation product variants</td>
<td>25</td>
</tr>
<tr>
<td>Tab. 14</td>
<td>Platform SCP key define prefix for SE051 product variants</td>
<td>25</td>
</tr>
<tr>
<td>Tab. 15</td>
<td>Platform SCP key define prefix for A5000 product variants</td>
<td>25</td>
</tr>
</tbody>
</table>
Figures

Fig. 1. Connection between OM-SE05xARD and i.MX 8M boards ........................................ 6
Fig. 2. Jumper configuration for i.MX 8M board .......... 7
Fig. 3. Micro-SD card preparation with Win32 Disk Imager software ........................................... 9
Fig. 4. Plug the power supply and connect the USB cable to your laptop ....................................... 10
Fig. 5. Check that i.MX 8M board is recognized in Device Manager ........................................... 11
Fig. 6. Open a TeraTerm serial connection .............. 12
Fig. 7. Configure TeraTerm serial port connection .... 12
Fig. 8. Booting the i.MX 8M ................................... 13
Fig. 9. Sign in in the OS ....................................... 14
Fig. 10. Go to the Plug & Trust middleware test example directory .......................................... 15
Fig. 11. Run se05x_minimal test example ............... 16
Fig. 12. SPC03 mutual authentication – principle ..... 22
Fig. 13. SPC03 Encryption and MACing principle ...... 22
Fig. 14. Default Platform SCP keys are defined in ex_sss_tp scp03_keys.h ................................... 23
Fig. 15. Select the actual Platform SCP keys in infsl_sss_ftr.h.in ................................................. 24
Fig. 16. EdgeLock SE05xPlatform SCP plain text key file ............................................................... 26
Fig. 17. SE050E CMake Settings - PlatformSCP enabled ............................................................... 27
Fig. 18. Run se05x_minimal example with platformSCP enabled - SCP keys defined in the Plug & Trust middleware source ........ 27
Fig. 19. Run se05x_minimal example with platformSCP enabled - SCP keys defined in a text file ........................................... 28
Fig. 20. Start the ssscli tool .................................. 30
Fig. 21. Close an already opened session .................. 30
Fig. 22. ssscli info ............................................. 31
Fig. 23. ssscli disconnect .................................... 31
Fig. 24. Obtain board IP address ............................ 32
Fig. 25. Connect to the board using SCP with WinSCP ................................................................. 33
Fig. 26. Copy the middleware to the board ............... 34
Fig. 27. Unzip the middleware .................................. 35
Fig. 28. Create CMake projects for the board .......... 36
Fig. 29. Build and install the project .......................... 37
Contents

1 How to use this document ................................................. 3
2 Hardware required ....................................................... 4
3 Boards setup .......................................................... 6
4 Software setup .......................................................... 8
   4.1 Micro-SD card preparation ..................................... 8
   4.2 Drivers ............................................................... 9
   4.3 Terminal setup .................................................... 11
   4.4 Booting the i.MX 8M .................................................. 12
5 Run preinstalled Plug & Trust middleware test examples ................................. 15
6 Product specific CMake build settings .................................. 17
   6.1 Example: SE050E CMake build settings .................. 19
7 Binding EdgeLock SE05x to a host MCU/MPU using Platform SCP ......................... 20
   7.1 Introduction to the Global Platform Secure Channel Protocol 03 (SCP03) ......................... 20
   7.2 How to configure the product specific default Platform SCP keys .................................. 22
   7.2.1 Defining the default Platform SCP keys in the Plug & Trust middleware source code ...... 22
   7.2.2 Defining the default Platform SCP keys in a text file ......................................................... 25
   7.3 How to enable Platform SCP in the CMake-based build system ........................................ 26
8 Manage access from multiple Linux processes to the EdgeLock SE05x ......... 29
9 Appendix A: Using the ssscli tool ........................................ 30
10 Appendix B: Update Plug & Trust middleware ..................................... 32
   10.1 Obtain the latest Plug & Trust middleware version ......................................................... 32
   10.2 Build the Plug & Trust middleware .................................. 34
11 Legal information ...................................................... 38

Please be aware that important notices concerning this document and the product(s) described herein, have been included in section ‘Legal information’. 

© NXP B.V. 2022. 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: 12 September 2022