EdgeLock SE05x Quick start guide with Raspberry PiRev. 1.5 — 3 August 2022Application note565815565815

Document information

Information	Content
Keywords	EdgeLock SE05x, EdgeLock A5000, EdgeLock SE Plug & Trust Middleware
Abstract	This document explains how to get started with the SE05x/A5000 development board and the Raspberry Pi board, as a reference for any other device running a Linux distribution. This guide provides detailed instructions for connecting the boards and running the project examples included in EdgeLock SE Plug & Trust Middleware.



Revision history

Revision history				
Revision number	Date	Description		
1.0	2019-08-30	First document release.		
1.1	2020-02-06	Added OM-SE050RPI adapter board.		
1.2	2020-12-07	Updated to latest template and fixed broken links.		
1.3	2021-01-22	Added EdgeLock SE051, terminal Figure changes and appendix addition to show the ssscli command line interface.		
1.4	2022-03-28	 Add EdgeLock SE050E and EdgeLock A5000 product variants. Update <u>Table 1</u>, Figure 1, Figure 2, Figure 3, Add note (step 2) in <u>Section 3.3</u> Build EdgeLock SE Plug & Trust Middleware examples. Add Section <u>Section 4</u> Product specific CMake build settings. Add Section <u>Section 5</u> Binding EdgeLock SE05x to a host using Platform SCP. Add Section <u>Section 6</u> Manage access from multiple Linux processes to the EdgeLock SE05x. 		
1.5	2022-08-03	Clarify to install python 3 in <u>Section 2.2.2</u> . Update to EdgeLock SE Plug & Trust Middleware version 04.02.xx. Update note (step 2) in <u>Section 3.3</u> Build EdgeLock SE Plug & Trust Middleware examples. Update Section <u>Section 4</u> Product specific CMake build settings. Update Section <u>Section 5</u> Binding EdgeLock SE05x to a host using Platform SCP.		

1 Required hardware

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 Raspberry Pi board, acting as a host controller.

1.1 Required hardware

The following hardware will be used throughout the document:

1. 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. <u>Table 1</u> details the ordering details of the EdgeLock SE05x and EdgeLock A5000 development boards.

Part number	12NC	Description	Picture
<u>OM-SE050ARD-E</u>	9354 332 66598	SE050E Arduino [®] compatible development kit	
OM-SE050ARD-F	9354 357 63598	SE050 Arduino [®] compatible development kit	
OM-SE050ARD	9353 832 82598	SE050F Arduino [®] compatible development kit	
<u>OM-SE051ARD</u>	9353 991 87598	SE051 Arduino [®] compatible development kit	
<u>OM-A5000ARD</u>	9354 243 19598	A5000 Arduino [®] compatible development kit	

Table 1. EdgeLock SE05x development boards.

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

2. OM-SE050RPI adapter board for Raspberry Pi:

Table 2. OM-SE050RPI adapter board details

Part number	12NC	Content	Picture
<u>OM-SE050RPI</u>	935398642598	Raspberry Pi to OM- SE05xARD adapter	

3. Raspberry Pi board:

Table 3. Raspberry Pi

Part number	Content	Picture
Raspberry Pi	Any Raspberry Pi model	

2 Prepare your Raspberry Pi

This section explains how to get your Raspberry Pi ready to execute the EdgeLock SE Plug & Trust Middleware. For that, you need to go through the following steps:

- 1. Hardware setup for Raspberry Pi
- 2. Software setup for Raspberry Pi

2.1 Hardware setup

The hardware setup consists of two steps:

- 1. Configuring the OM-SE05xARD jumpers, as described in Section 2.1.1.
- 2. Connecting the OM-SE05xARD to the Raspberry Pi, as described in <u>Section 2.1.2</u>.

2.1.1 Jumper configuration

Make sure the jumpers in your OM-SE05xARD board are configured as shown in Figure 1:



Note: For more information about the jumper settings, refer to <u>AN13539</u> OM-SE05xARD hardware overview.

2.1.2 Connecting the OM-SE05xARD to the Raspberry Pi

You have two options to connect the Raspberry Pi to the OM-SE05xARD board:

- 1. Using the OM-SE05xRPI adapter board, as described in Section 2.1.2.1
- 2. Using the OM-SE05xARD connected with wires, as described in Section 2.1.2.2

2.1.2.1 Using the OM-SE05xRPI adapter board

The Raspberry Pi and the OM-SE05xARD boards can be directly connected using the OM-SE050RPI adapter board. Follow the steps shown in <u>Figure 2</u>:

- 1. Mount the OM-SE05xARD on top of the OM-SE05xRPI board using the Arduino connectors.
- 2. Mount the two boards on top of the Raspberry Pi using the Raspberry connectors in the OM-SE05xRPI.

The result of it is three boards stacked together, being the OM-SE05xRPI the board in between the Raspberry Pi and OM-SE05xARD.



2.1.2.2 Connecting the OM-SE05xARD with wires

In case you do not have the OM-SE05xRPI adapter board, you can also manually wire the Raspberry Pi to the OM-SE05xARD using the I²C connector, as shown in Figure 3:





•	
OM-SE05xARD (# jumper - # pin)	Raspberry Pi (# jumper - # pin)
J2-P10 (ARD_SCL)	J8-P5 (SCL)
J2-P9 (ARD_SDA)	J8-P3 (SDA)
J8-P7 (GND)	J8-P6 (GND)
J8-P4 (3V3_ARD)	J8-P1 (3V3)

Table 4. OM-SE05xARD wiring to the Raspberry Pi board

2.2 Software setup

The software setup consists of three steps:

- 1. Install your preferred Linux distribution in your device. In this guide the Raspberry Pi board running the Raspbian operating system is used as a reference. Raspbian can be installed as described in <u>Section 2.2.1</u>.
- 2. Install the build tools necessary to build the EdgeLock SE Plug & Trust Middleware and the test project examples. The procedure for the Raspbian operating system is described in <u>Section 2.2.2</u>.
- Enable the I²C interface in your Linux distribution to allow the communication with the security IC of the OM-SE05xARD board. The procedure for the Raspbian operating system is described in <u>Section 2.2.3</u>.

2.2.1 Install Raspbian

Before executing the steps described in this guide, it is necessary to install the Raspbian operating system in the Raspberry Pi. The official <u>Raspberry website</u> recommends two options:

- 1. Using New Out of Box Software (NOOBS), an easy operating system installation manager for the Raspberry Pi. This tool is the easiest and most recommended option, but requires a screen to go through the initial installation process. Installation instructions are provided in the official Raspberry <u>NOOBS</u> webpage.
- 2. Downloading the official Raspbian image from the official Raspberry Pi <u>image</u> <u>repository</u> and then flashing the image in the SD card by following the instructions provided in the <u>official documentation</u>.

The steps described in this guide use the latest Raspbian release at the time of writing (Raspbian 10 Buster).

2.2.2 Install build tools

To build the EdgeLock SE Plug & Trust Middleware middleware and the example projects, it is necessary to have the Python and CMake packages installed in the system along with the libssl library (part of OpenSSL toolkit). CMake GUI packages are also required if you want to use the CMake graphical user interface. You can install the required packages by opening a Terminal window and following the steps as shown in Figure 4:

- 1. You can install all the required packages with a single command by sending:
 - >> sudo apt-get install python3 cmake cmake-curses-gui cmakeqt-gui libssl-dev
- 2. You may be asked to proceed with the installation:

Send >> y



2.2.3 Enable the I²C interface

The Raspberry Pi board communicates with the OM-SE05xARD security IC through the I^2C interface. The I^2C interface is not enabled by default in Raspbian and must be activated before the EdgeLock SE Plug & Trust Middleware test examples can be executed. To enable I^2C , open a Terminal window and follow these steps:

1. Verify if I^2C is active by listing the available I^2C interfaces:

>> ls /sys/bus/i2c/devices/

If the *i2c-x* interface is listed, as shown in Figure 5, then you can skip this section and proceed to <u>Section 3</u>. **Note:** the l^2C interface number might be different.

🗾 pi@192.168.1.151:22 - Bitvise xterm - pi@raspberrypi: ~	-	×
pi@raspberrypi:~ \$ ls /sys/bus/i2c/devices/		^
pi@raspberrypi:~ \$		~
Figure 5. List I ² C interfaces		

2. Open the Raspberry Pi software configuration tool, as shown in Figure 6: >> sudo raspi-config

pi@192.168.1.151:22 - Bitvise xterm - pi@raspberrypi: ~	_	×
pi@raspberrypi:~ \$ ls /sys/bus/i2c/devices/ pi@raspberrypi:~ \$ sudo raspi-config		^

Figure 6. Open the Raspberry Pi software configuration tool

3. Use the up and down arrow keys to select the 5th menu entry (Interfacing Options) and then press Enter, as shown in Figure 7:

pi@192.168.1.151:22 - Bitvise xterm - pi	₽raspberrypi: ~ —	>
spberry Pi Model B Plus I	Rev 1.2	
Rasn	perry Pi Software Configuration Tool (respi-config)	
T maspi	certy if Software configuration foor (raspi config) [
1 Change User Pass	word Change password for the 'pi' user	
2 Network Options	Configure network settings	
3 Boot Options	Configure options for start-up	
4 Localisation Opt:	lons Set up language and regional settings to match your location	
6 Overclock	Configure overclocking for your Pi	
7 Advanced Options	Configure advanced settings	
8 Update	Update this tool to the latest version	
9 About raspi-conf:	g Information about this configuration tool	
	<select> <finish></finish></select>	

4. Use the up and down arrow keys to select the 5th menu option (I²C) and then press Enter, as shown in <u>Figure 8</u>:

Z pi@192.168.1.151:22 - Bitvise xterm - pi@raspberrypi: ~ -	×
	^
Raspberry Pi Software Configuration Tool (raspi-config) P1 Camera Enable/Disable connection to the Raspberry Pi Camera P2 SSH Enable/Disable remote command line access to your Pi using SSH P3 VNC Enable/Disable graphical remote access to your Pi using RealVNC P4 SPI Enable/Disable automatic loading of SPI kernel module P5 12C Enable/Disable automatic loading of 12C kernel module P6 Serial Enable/Disable shell and kernel messages on the serial connection P7 1-Wire Enable/Disable one-wire interface P8 Remote GPIO Enable/Disable remote access to GPIO pins	
<select> <back></back></select>	
	×
Figure 8. Enable I ² C interface	

5. You will be asked to confirm your choice to activate the I²C interface. Use the left and right arrow keys to select the Yes option and then press Enter, as shown in Figure 9:



6. Close the Raspberry Pi software configuration tool. Use the left and right arrow keys to select the Finish option and then press Enter, as shown in <u>Figure 10</u>:

provide reaction in the second	×
Raspberry Pi Model B Plus Rev 1.2	^
Response Response Risoftware Configuration Tool (passiconfig)	
[Kasperry II Software configuration foor (Faspi config)]	
1 Change User Password Change password for the 'pi' user	
3 Boot Options Configure options for start-up	
4 Localisation Options Set up language and regional settings to match your location	
5 Interfacing Options Configure connections to peripherals 6 Overclock Configure overclocking for your Pi	
7 Advanced Options Configure advanced settings	
8 Update Update this tool to the latest version	
9 About Paspi-config information about this configuration cool	
<select> <finish></finish></select>	
	v

7. Verify the correct activation of the I^2C interface, as shown in <u>Figure 11</u>:

>> ls /sys/bus/i2c/devices/ The *i2c-x* interface should now be listed. **Note:** the l^2C interface number might be different.

Z pi@192.168.1.151:22 - Bitvise xterm - pi@raspberrypi: ~	-	×
pi@raspberrypi:~ \$ ls /sys/bus/i2c/devices/ 🛑		^
12c-1 pi@raspberrypi:~ \$		~
Figure 11. List I ² C interfaces		

3 Run EdgeLock SE Plug & Trust Middleware test examples

This section details the steps required from the moment you download EdgeLock SE Plug & Trust Middleware until you are able to run an EdgeLock SE Plug & Trust Middleware test example.

3.1 Download EdgeLock SE Plug & Trust Middleware

The EdgeLock SE Plug & Trust Middleware stack includes several project examples for cloud service onboarding. To prepare the EdgeLock SE Plug & Trust Middleware:

 Download the EdgeLock SE Plug & Trust Middleware from <u>NXP website</u> and place the .zip file in the */home/user* directory of your Raspbian distribution. *Note:The user folder can have different names, in this example the user folder's name is pi*

© NXP B.V. 2022. All rights reserved.

- 2. Open a Terminal window and follow the next steps as shown in Figure 12:
 - a. Move to the user's *home* directory:
 (1) >> cd ~
 - b. Create a folder called se050_middleware: (2) >> mkdir se mw
 - c. Unzip the EdgeLock SE Plug & Trust Middleware in the se050_middleware folder:
 (3) >> unzip SE-PLUG-TRUST-MW.zip -d se mw

Note: The name of the zip file might be different. **Note:** This command may take a few seconds to complete.



- 3. You can verify that the files have been correctly unzipped by following these steps: a. Move to the *simw-top* folder inside the *se_mw* folder:
 - >> cd se_mw/simw-top
 - b. List the content of the simw-top folder:
 - >> ls

The content of the folder should be the same as shown in Figure 13:

Z pi@192.168.1.151:22	-						
pi@raspberrypi:	pi@raspberrypi:~ \$ cd se_mw//simw-top						
pi@raspberrypi:	~/se_mw/simw-top \$ ls						
akm	demos	hostlib	README.First.txt	version_inf	fo.txt		
Android.mk	doc	<pre>nxp_iot_agent</pre>	scripts				
binaries	EULA.pdf	PlugAndTrustMW.pdf	SSS				
CleanSpec.mk	ext	projects	Third_Party_License.pdf				
CMakeLists.txt	<pre>git_commit_info.txt</pre>	pycli	tools				
pi@raspberrypi:	~/se_mw/simw-top \$				~		
Figure 13.	simw-top folder	content					

3.2 Build EdgeLock SE Plug & Trust Middleware

The EdgeLock SE Plug & Trust Middleware uses CMake for building the project examples into your local machine. To build the EdgeLock SE Plug & Trust Middleware middleware, open a Terminal window and follow the next steps as shown in Figure 14:

1. Go to the folder with the unzipped SE050 middleware:

(1) >> cd /home/pi/se_mw/simw-top/scripts

2. Generate the EdgeLock SE Plug & Trust Middleware project examples:

(2) >> python create_cmake_projects.py rpi Note: This command may take a few seconds to complete. The last parameter rpi circumvents auto-detection of the host and enforces to build for Raspberry Pi.



 If the compilation is successful you should (1) see a new *simw-top_build* folder inside the se_mw folder and (2) a new folder inside the simw-top_build folder as shown in Figure 15:



3.3 Build EdgeLock SE Plug & Trust Middleware examples

The EdgeLock SE Plug & Trust Middleware contains several examples used to verify atomic EdgeLock SE05x security IC features. This section explains how to compile the EdgeLock SE Plug & Trust Middleware examples. Open a Terminal window and follow these steps:

1. Move to the folder that contains the examples and the source code of the Raspbian EdgeLock SE05x libraries:

```
>> cd /home/pi/se_mw/simw-top_build/
raspbian_native_se050_t1oi2c
```

Note: The default build configuration of the EdgeLock SE Plug & Trust Middleware
 ≥ V04.02.0x generates code for the OM-SE050ARD-E 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 4 Product specific CMake build settings.
 Open the CMake configuration interface, as shown in Figure 16 to change build
 settings:

```
>> ccmake .
```

Note: You can use the graphical interface by sending *cmake-gui* . instead.

🗾 pi@192.168.1.151:22 - Bitvise xterm - pi@raspberrypi: ~/se_mw/simw-top_build/raspbian_native_se050_t1oi2c — 🛛	×
INFO:main:NodeSet generation code successfully printed	^
### Using Raspberry PI	
<pre>#cmake -DHost=Raspbian -DApplet=SE05X_C -DCMAKE_BUILD_TYPE=Debug -DSCP=SCP03_SSS -DSMCOM=T10I2C -E</pre>	ЭНо
stCrypto=OPENSSL	
The C compiler identification is GNU 8.3.0	
The CXX compiler identification is GNU 8.3.0	
Check for working C compiler: /usr/bin/cc	
Check for working C compiler: /usr/bin/cc works	
Detecting C compiler ABI info	
Detecting C compiler ABI info - done	
Detecting C compile features	
Detecting C compile features - done	
Check for working CXX compiler: /usr/bin/c++	
Check for working CXX compiler: /usr/bin/c++ works	
Detecting CXX compiler ABI info	
Detecting CXX compiler ABI info - done	
Detecting CXX compile features	
Detecting CXX compile features - done	
BUILD_TYPE: Debug	
Found OpenSSL: /usr/lib/arm-linux-gnueabihf/libcrypto.so (found version "1.1.1d")	
Found: /usr/lib/arm-linux-gnueabihf/libssl.so/usr/lib/arm-linux-gnueabihf/libcrypto.so	
CMAKE_CXX_COMPILER_ID = GNU	
CMAKE_SYSTEM_NAME = Linux	
SE05X_Auth - None	
CMake version: 3.13.4	
CMake system name: Linux	
Timestamp is 2020-12-23T15:16:16Z	
Configuring done	
Generating done	
Build files have been written to: /home/pi/se_mw/simw-top_build/raspbian_native_se050_t1oi2c	
pi@raspberrypi:~/se_mw/simw-top/scripts \$ cd /home/pi/se_mw/simw-top_build/raspbian_native_se050_t	t10
i2c/	
pi@raspberrypi:~/se_mw/simw-top_build/raspbian_native_se050_t1oi2c \$ ccmake .	~
Figure 16. Open CMake configuration interface	

3. Review the build configuration and make sure that the *Host* parameter is set to the value *Raspbian*, as shown in Figure 17. Leave the default settings and press *q* to return to the console.

Note: If you want to change the configuration you can use the up and down arrow keys to navigate through the available options and the left and right arrow keys to

CMAKE_BUILD_TYPE	Debug
CMAKE_INSTALL_PREFIX	/usr/local
LIB_ANL	/usr/lib/arm-linux-gnueabihf/libanl.so
NXPInternal	OFF
OPENSSL_ROOT_DIR	075
PAHO_BUILD_DEB_PACKAGE	
PAHO_BUILD_SAMPLES	
PAHO_BUTLD_STATTC	OFF
PAHO ENABLE CPACK	ON
PAHO ENABLE TESTING	OFF
PAHO WITH SSL	ON
PTMW_A71CH_AUTH	None
PTMW_Applet	SE05X_C
PTMW_FIPS	None
PTMW_Host	Raspbian
PTMW_HostCrypto	OPENSSL
PTMW_Log	Default
PTMW_RTOS	Default
PTMW_SBL	None
DTMU SEOSY Auth	Dlatfcrpp2
DTMW_SE05X_AUCH	
PTMW_SECOX_VEN	03_AA T10T2C
PTMW_mbedTLS_ALT	None
SSSFTR SE05X AES	ON
SSSFTR SE05X AuthECKey	ON
SSSFTR_SE05X_AuthSession	ON
SSSFTR_SE05X_CREATE_DELETE_CRY	ON
SSSFTR_SE05X_ECC	ON
SSSFTR_SE05X_KEY_GET	ON
SSSFIR_SE05X_KEY_SE1	ON
SSSFIR_SEUSX_KSA	
SSSETR SW ECC	
SSSETR SW KEY GET	
SSSETR SW KEY SET	ON
SSSETR SW RSA	ON
SSSFTR SW TESTCOUNTERPART	ON
WithAccessMgr UnixSocket	OFF
WithCodeCoverage	OFF
WithExtCustomerTPMCode	OFF
WithNXPNFCRdLib ¶	OFF
WithOPCUA_open62541	0FF
WITHSHAREdLIB	UN
Figure 17 Poview build ac	afiguration
Figure 17. Review build con	Ingulation

change the option value. In case you edit the configuration, press c (configure) and then g (generate) to apply the changes.

4. Build the project examples, as shown in Figure 18: >> cmake --build .

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

🗾 pi@192.168.1.151:22 - Bitvise xterm - pi@raspberrypi: -/se_mw/simw-top_build/raspbian_native_se050_t1oi2c — 🗆 🗙
-incompany (or my/cimy too build/complian action action action to act the build
pl@raspberryp1://se_mm/simw-top_bulld/raspblan_native_seoso_tioizc > cmakebulld .
Scanning dependencies of target smcom
[0%] Building C object nostlib/nostlib/indcommon/CMakeriles/smcom.dir//platform/generic/sm_timer
[18] Building C object nostlib/hostLib/libCommo/CMaKeFileS/smCom.dir/_/platform/linux/i2c_a/.c.o
[18] Building C object nostlib/nostlib/libCommon/CMakeriles/smcom.dir//platform/rsp/se05x_reset.
[18] building C object hostild/hostild/indomon/Chakerile/smcon.dir//tstutil/st_sm_time.c.o
[28] building C object hostilp/hostilp/libCommon/Chakerile/smcon.dir/intra/sm_apdu.c.o
[28] Building C object hostilp/hostilp/libCommon/Chakerile/smcom.dir/intra/sm_errors.c.o
[20] Building C object hostil/hostil/libCommer/CMakeriles/smcom.ut//in/ra/sm_printi.c.0
[58] Building C object hostilb/hostilb/hostilb/incommon/cmakeriles/smcom/horz/phixpcseral_izc
1. 301 Ruilding C shiert hertlih/hertlih/lihCommon/(MakeFiles/cmCom din/cmCom/T1oT2C/mbNynEcoPosto78
[53] building cooject hostilo/hostilo/hostilo/incommon/chakeriles/smcom/dif/smcom/fibiz/pinkkpiserioto/a
[3%] Building C object hostlib/hostlib/libCommon/CMakeFiles/smCom dir/smCom/T1oT2C/nbNynEse Ani c
a building cobject hostild, hostild, hostild, indentified, smean, dir, smean, fibild, physics_pitt
[4%] Building C object hostlib/hostlib/libCommon/(MakeFiles/smCom dir/smCom/smCom c o
4%] Building C object hostlib/hostlib/libCommon//MakeFiles/smCom.dir/smCom/smCom/io12C.c.o
4%] Building C object hostlib/libCommon/(MakeFiles/smCom.dir/infra/nxlog.c.o
5%] Building C object hostlib/libCommon/(MakeFiles/smCom.dir/nxScn/nxScn03.com.c.o
5 1 Linking C static library library a
[5%] Built target smCom
Scanning dependencies of target unity
[5%] Building C object ext/unity/CMakeFiles/unity.dir/unity.c.o
6%] Building C object ext/unity/CMakeFiles/unity.dir/unity fixture.c.o
6%] Building C object ext/unity/CMakeFiles/unity.dir/unity fixture addin.c.o
7% Linking C static library libunity.a
7%] Built target unity
Scanning dependencies of target common_ssl_obj_static
[7%] Building C object ext/paho.mqtt.c/src/CMakeFiles/common_ssl_obj_static.dir/MQTTTime.c.o
[8%] Building C object ext/paho.mqtt.c/src/CMakeFiles/common_ssl_obj_static.dir/MQTTProtocolClient
.c.o
· · · · · · · · · · · · · · · · · · ·

Figure 18. Build project examples

5. Install the projects in the system as shown in Figure 19: >> sudo make install Note: This command may take a few seconds to complete.

pi@192.168.1.151:22 - Bitvise xterm - pi@raspberrypi: ~/se_mw/simw-top_build/raspbian_native_se050_t1oi2c	- 0	×
pi@raspberrypi:~/se_mw/simw-top_build/raspbian_native_se050_t1oi2c \$ sudo make install ৎ		^
5%] Built target smCom		
[7%] Built target unity		
[15%] Built target common_ssl_obj_static		
[23%] Built target common_obj_static		
[24%] Built target paho-mqtt3c-static		
[24%] Built target paho-mqtt3a-static		
[25%] Built target MQTTVersion-static		
[26%] Built target paho-mqtt3as-static		
[27%] Built target paho-mqtt3cs-static		
[29%] Built target a7x_utils		
[31%] Built target se05x		
[39%] Built target SSS_APIs		
[41%] Built target jrcpv1_server		
[50%] Built target sssapisw		
[53%] Built target ex_common		
[54%] Built target ex_symmetric		
[55%] Built target ex_hkdf		
[55%] Built target ex_md		
[56%] Built target ex_hmac		
[56%] Built target ex_ecdh		
[57%] Built target ex_ecc		
[58%] Built target ex_ecdaa		
[59%] Built target ex_attest_ecc		
[60%] Built target ex_attest_mont		
[61%] Built target ex_rsa		
[64%] Built target sss_engine		
[65%] Built target se05x_Minimal		
[65%] Built target se05x_ex_export_se_to_host		
[66%] Built target se05x_ex_import_host_to_se		
[67%] Built target se05x_Personalization		
[68%] Built target se05x_Delete_and_test_provision		
68%] Built target se05x_MandatePlatformSCP		
[69%] Built target se05x_TransportLock		
		~

Figure 19. Install projects in the system

6. Update the cache to include the newly installed libraries as shown in Figure 20:

>> sudo ldconfig /usr/local/lib



Figure 20. Load new installed libraries

3.4 Execute EdgeLock SE Plug & Trust Middleware test example

This section explains how to run the EdgeLock SE Plug & Trust Middleware test example called se05x minimal. The se05x minimal project outputs the memory left in the EdgeLock SE05x security IC. To execute the se05x minimal test example follow these steps:

1. Connect the OM-SE05xARD board to the Raspberry Pi as described in Section 2.1.

- 2. Open a Terminal window and follow the steps as shown in Figure 21:
 - a. Move to the directory containing the examples binaries:
 (1) >> cd /home/pi/se_mw/simw-top_build/
 raspbian native se050 tloi2c/bin/
 - b. Run the se05x_minimal example:
 - (2) >> ./se05x_Minimal
 - (3) You should see the EdgeLock SE05x IC available memory (in this case, 32767)



4 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 <u>AN12436</u> chapter 1 *Product Information*. <u>AN12973</u> describes the same procedure for the SE051 product family.

The following tables show the required PTMW CMake options to build a dedicated product variant. The $\texttt{SSSFTR}_\texttt{SE05X}_\texttt{RSA}$ CMake option is used to optimize the memory footprint for product variants that do not support RSA.

Variant	OEF ID	PTMW_ Applet	PTMW_ FIPS	PTMW_ SE05X_ Ver	PTMW_SE05X_Auth	PTMW_ SCP	SSSFTR_ SE05X_ RSA
SE050E Dev. Board OM-SE050ARD-E	A921	SE050_E	None	07_02	any option	None or	disabled
SE050E2	A921	-				SCP03_ SSS	

Table 5. CMake Settings for SE050E product variants

AN12570 Application note

Variant	OEF	PTMW_	PTMW_	PTMW_	PTMW_SE05X_Auth	PTMW_	SSSFTR_
	ID	Applet	FIPS	SE05X_		SCP	SE05X_
				Ver			RSA
SE050F Dev.Board	A92A	SE05X_C	SE050	03_XX	PlatfSCP03	SCP03_	enabled
OM-SE050ARD-F					or	SSS	
SE050F2	A92A	-			UserID_PlatfSCP03		
					or		
					AESKey_PlatfSCP03		
					or		
					ECKey_PlatfSCP03		

Table 6. CMake Settings for SE050F product variants

Table 7. CMake Settings for SE050 Previous Generation product variants

Variant	OEF	PTMW_	PTMW_	PTMW_	PTMW_SE05X_Auth	PTMW_	SSSFTR_
	ID	Applet	FIPS	SE05X_		SCP	SE05X_
				Ver			RSA
SE050A1	A204	SE05X_A	None	03_XX	any	None	disabled
SE050A2	A205				option	or	
						SCP03_	
						SSS	
SE050B1	A202	SE05X_B	None	03_XX	any	None	enabled
SE050B2	A203				option	or	
						SCP03	
						SSS -	
SE050C1	A200	SE05X_C	None	03_XX	any	None	enabled
SE050C2	A201]			option	or	
SE050 Dev Board	A1F4					SCP03_	
OM-SE050ARD						SSS	
SE050F2	A77E ^[1]	SE05X_C	SE050	03_XX	PlatfSCP03	SCP03_	enabled
					or	SSS	
					UserID_PlatfSCP03		
					or		
					AESKey_PlatfSCP03		
					or		
					ECKey_PlatfSCP03		

[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 8. CMake Settings for SE051 product variants

Variant	OEF ID	PTMW_ Applet	PTMW_ FIPS	PTMW_ SE05X_ Ver	PTMW_SE05X_Auth	PTMW_ SCP	SSSFTR_ SE05X_ RSA
SE051A2	A920	se05x_a	None	07_02	any option	None or SCP03_ SSS	disabled

AN12570 Application note

Variant	OEF ID	PTMW_ Applet	PTMW_ FIPS	PTMW_ SE05X_ Ver	PTMW_SE05X_Auth	PTMW_ SCP	SSSFTR_ SE05X_ RSA
SE051C2	A8FA	SE05X_C	None	07_02	any option	None or SCP03_ SSS	enabled
SE051W2	A739	SE05X_C	None	07_02	any option	None or SCP03_ SSS or SCP03_ SSS	enabled
SE051A2	A565	SE05X_A	None	06_00	any option	None or SCP03_ SSS	disabled
SE051C2	A564	se05x_c	None	06_00	any option	None or SCP03_ SSS	enabled

Table 8. CMake Settings for SE051 product variants...continued

Table 9. CMake Settings for A5000 product variants

Variant	OEF ID	PTMW_ Applet	PTMW_ FIPS	PTMW_ SE05X_ Ver	PTMW_SE05X_Auth	PTMW_ SCP	SSSFTR_ SE05X_ RSA
OM-A5000ARD	A736	AUTH	None	07_02	any	None	disabled
A5000	A736	-			option	or	
						SCP03_ SSS	

4.1 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 5:

- Select SE05X E for the CMake option PTWM Applet.
- Select None for the CMake option PTWM FIPS.
- Select 07 02 for the CMake option PTWM 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 <u>How to enable Platform SCP in the CMake-based build system</u>.

A CMake 3.16.3 - /home/pi/se_mw/simw-top_build/raspbian_native_se050_t1oi2c		Ν	- 🗆 ×
<u>F</u> ile <u>T</u> ools <u>Options</u> <u>H</u> elp		63	
Where is the source code: /home/pi/se_mw/simw-top			Browse Source
Where to build the binaries: //home/pi/se mw/simw-top build/raspbian native se050 t	1oi2c		Browse Build
Search		Grouped Advanced	Add Entry
News	Mahua	divanceu	And Endy
CMAKE BUILD TYPE	Debug		
CMAKE_INSTALL_PREFIX	/usr/local /usr/lib/arm-lipux-gp	ueabibf/libanl.so	
NXPInternal	/usi/iib/arm-imux-gri	deabhil/libalil.so	
OPENSSL_ROOT_DIR PAHO BUILD DEB PACKAGE	-		
PAHO_BUILD_SHARED	✓		
PAHO_BUILD_STATIC	<u>_</u>		
PAHO_ENABLE_TESTING			
PAHO_WITH_SSL PTMW_A71CH_AUTH	✓ None		
PTMW_Applet	SE050_E		
PTMW_HPS PTMW Host	Raspbian		
PTMW_HostCrypto	OPENSSL		
PTMW_Log PTMW_RTOS	Default		
PTMW SBL	None		
PTMW_SE05X_Auth	None		
PTMW SE05X Ver PTMW SMCOM	07 02 T1ol2C		
PTMW_mbedTLS_ALT	None		
SSSFIR_SE05X_AES SSSFTR_SE05X_AuthECKey	V V		
SSSFTR_SE05X_AuthSession	√		
SSSFTR_SE05X_ECC	V		
SSSFTR_SE05X_KEY_GET	V V		
SSSFTR_SE05X_RSA			
SSSFTR_SW_AES SSSFTR_SW_ECC	V		
SSSFTR_SW_KEY_GET	√		
SSSFTR_SW_RSA	V		
SSSFTR_SW_TESTCOUNTERPART WithAccessMgr_UnixSocket	<u>~</u>		
WithCodeCoverage			
			•
Press Configure to update and display new values in n	ed, then press Genera	te to generate selected build t	iles.
<u>C</u> onfigure <u>Generate</u> <u>Open Project</u> Current Generator: Unix Makefiles			

Figure 22. SE050E CMake Settings - Plain communication

Run the following commands to update the CMake settings and rebuild the EdgeLock SE Plug & Trust Middleware:

cd ~/se mw/simw-top build/raspbian native se050 tloi2c

cmake-gui .

Note: You can use the commandline interface by sending *ccmake* . instead (see also Section 3.3).

Update the CMake settings as explained above. Press first the *Configure* button and second the *Generate* button and close the CMake GUI.

```
cmake --build .
sudo make install
sudo ldconfig /usr/local/lib/
```

5 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/MCU and EdgeLock SE05x, so that only the paired MPU/MCU 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/MCU and the EdgeLock SE05x and physical tampering attacks aimed at replacing the host MPU/MCU 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 middlware for EdgeLock SE05x.

The following topics are discussed:

- Section 5.1 Introduction to the Global Platform Secure Channel Protocol 03 (SCP03)
- <u>Section 5.2</u> How to configure the EdgeLock SE05x product specific SCP keys in the EdgeLock SE Plug & Trust Middleware
- Section 5.3 How to enable Platform SCP in the EdgeLock SE Plug & Trust Middleware

5.1 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 EdgeLock SE Plug & Trust Middleware. The <u>AN12662</u> *Binding a host device to EdgeLock SE05x* describes in detail the concept of secure binding.

SCP03 is defined in <u>Global Platform Secure Channel Protocol '03' - Amendment D v1.2</u> 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.

AN12570 Application note

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 10. Static SCP03 keys

Key	Description	Usage	Кеу Туре
Key-ENC	Static Secure Channel Encryption Key	Generate session key for Decryption/ Encryption (AES)	AES 128
Кеу-МАС	Static Secure Channel Message Authentication Code Key	Generate session key for Secure Channel authentication and Secure Channel MAC Verification/Generation (AES)	AES 128
Key-DEK	Data Encryption Key	Sensitive Data Decryption (AES)	AES 128

The session key generation is performed by the EdgeLock SE Plug & Trust Middleware host crypto.

Table 11. SCP03 session keys

Key	Description	Usage	Кеу Туре
S-ENC	Session Secure Channel Encryption Key	Used for data confidentiality	AES 128
S-MAC	Secure Channel Message Authentication Code Key for Command	Used for data and protocol integrity	AES 128
S-RMAC	Secure Channel Message Authentication Code Key for Response	User for data and protocol integrity	AES 128

Note: For further details please refer to <u>Global Platform Secure Channel Protocol '03'</u> - <u>Amendment D v1.2</u>.

EdgeLock SE05x Quick start guide with Raspberry Pi







5.2 How to configure the product specific default Platform SCP keys

The initial Platform SCP key values are described for the SE050 product variants in <u>AN12436</u> and for the SE051 variants in <u>AN12973</u>.

For evaluation purpose, the Platform SCP keys can be defined either in the EdgeLock SE Plug & Trust Middleware source code (see <u>Section 5.2.1</u>) or provided as text file (see <u>Section 5.2.2</u>).

Note: In this example the Raspberry Pi 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.

© NXP B.V. 2022. All rights reserved.

AN12570

5.2.1 Defining the deault Platfrom SCP keys in the EdgeLock SE Plug & Trust Middleware source code

The EdgeLock SE Plug & Trust Middleware header file <code>ex_sss_tp_scp03_keys.h</code> contains the default values of all EdgeLock SE05x, EdgeLock A5000 and A71CH product variants.

The ex_sss_tp_scp03_keys.h header file can be found in the following location: / home/pi/se mw/simw-top/sss/ex/inc/

GNU nano 3.2	ex sss tp_scp03_keys.h
#define SSS_AUTH_KEY_MAC \	
{ 0X4F, 0X10, 0X3F, 0X59, 0XF0, 0X74, 0X31, 0XF4, 0X3E, 0	XEZ, UXEE, UX18, UX34, UXAS, UXZ3, UX34, }
$\{0xD4, 0x76, 0xCE, 0x47, 0xAA, 0x27, 0xB5, 0x4A, 0xB3, 0xAA, 0xB3, 0xB3, 0xAA, 0xB3, 0xAA, 0xB3, 0xAA, 0xB3, 0xAA, 0xB3, 0xB3, 0xB3, 0xAA, 0xB3, 0xB$	xDB. 0xEB. 0xE7. 0x65. 0x6D. 0x67. 0x70. }
#endif // SSS PFSCP ENABLE SE051A 0001A920	
// SSS_PFSCP_ENABLE_SE050E_0001A921	
<pre>#if defined (SSS_PFSCP_ENABLE_SE050E_0001A921) && SSS_PFSCP_ENABLE_SE050E_0001A921)</pre>	NABLE_SE050E_0001A921 == 1
#detine SSS_AUTH_KEY_ENC \	
$\#$ define SSS AUTH KEY MAC \	X04, 0X00, 0XC4, 0XDF, 0XDC, 0XAF, 0X04, j
{ 0x73, 0x8D, 0x5B, 0x79, 0x8E, 0xD2, 0x41, 0xB0, 0xB2, 0x	x47, 0x68, 0x51, 0x4B, 0xFB, 0xA9, 0x5B, }
#define SSS AUTH KEY DEK \	,,,,,,
{ 0x67, 0x02, 0xDA, 0xC3, 0x09, 0x42, 0xB2, 0xC8, 0x5E, 0x	x7F, 0x47, 0xB4, 0x2C, 0xED, 0x4E, 0x7F, }
<pre>#endif // SSS_PFSCP_ENABLE_SE050E_0001A921</pre>	
// CCC DECCD ENABLE CEDEAW 000EA730	
#if defined (SSS PESCP ENABLE SE051W 0000A759	NABLE SE051W 00054730 1
#define SSS AUTH KEY ENC \	NDEE_3203111_0003A733 == 1
{ 0x18, 0xB3, 0xB4, 0xE3, 0x40, 0xC0, 0x80, 0xD9, 0x9B, 0x	xEB, 0xB8, 0xB8, 0x64, 0x4B, 0x8C, 0x52, }
<pre>#define SSS_AUTH_KEY_MAC \</pre>	
{ 0x3D, 0x0C, 0xFA, 0xC8, 0x7B, 0x96, 0x7C, 0x00, 0xE3, 0x	x3B, 0xA4, 0x96, 0x61, 0x38, 0x38, 0xA2, }
#define SSS_AUTH_KEY_DEK \	
{ 0X08, 0X00, 0X83, 0XF9, 0X4E, 0X08, 0XCB, 0X94, 0X73, 07 #endif // SSS DESCD ENABLE SE051W 0005A730	XEC, 0XCI, 0X56, 0X7A, 0X1B, 0XDI, 0X09, }
#end(1 // 555_11501_ENABLE_5E051W_0005A755	
// SSS PFSCP ENABLE A5000 0004A736	
<pre>#if defined (SSS_PFSCP_ENABLE_A5000_0004A736) && SSS_PFSCP_ENA</pre>	ABLE_A5000_0004A736 == 1
<pre>#define SSS_AUTH_KEY_ENC \</pre>	
{ 0xC9, 0x11, 0x85, 0x00, 0xB5, 0xFF, 0xA1, 0x43, 0x3A, 0x	x50, 0x22, 0x6F, 0x48, 0x9A, 0x0A, 0xA5, }
$\begin{cases} 4 \text{ def ute SSS_AUTE_KET_MAC} \\ 4 \text{ av } 20 \text{ av } \text{D2} \text{ av } \text{EE} \text{ av } 28 \text{ av } \text{E7} \text{ av } \text{EE} \text{ av } \text{EB} \text{ av } 15 \text{ av } 30 \text{ av} \end{cases}$	Y68 0YBE 0Y38 0Y1E 0Y61 0YBC 0Y01 }
#define SSS AUTH KEY DEK \	, or
{ 0x61, 0x24, 0xD3, 0x84, 0x02, 0x11, 0x80, 0x60, 0xED, 0x	x91, 0x03, 0x60, 0xFC, 0x5A, 0x42, 0x78, }
#endif // SSS_PFSCP_ENABLE_A5000_0004A736	
Figure 25. Default Platform SCP keys are define	d in the ex sss tp scp03
kowa h hoador filo	
keys.n lieduel lile	

The fsl_sss_ftr.h.in file includes options to select one of the predefined default Platform SCP keys. This file is located in: /home/pi/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.

EdgeLock SE05x Quick start guide with Raspberry Pi

GNU nano 3.2	fsl_sss_ftr.h.in
/* Enable one of these	
* If none is selected, default config would be used */	
<pre>#define SSS_PFSCP_ENABLE_SE050A1 0 #define SSS_PFSCP_ENABLE_SE050A2 0</pre>	
<pre>#define SSS_PFSCP_ENABLE_SE050B1 0 #define SSS_PFSCP_ENABLE_SE050B2 0</pre>	
<pre>#define SSS_PFSCP_ENABLE_SE050C1 0 #define SSS_PFSCP_ENABLE_SE050C2 0</pre>	
<pre>#define SSS_PFSCP_ENABLE_SE050_DEVKIT 0 #define SSS_PFSCP_ENABLE_SE051A2 0</pre>	
#define SSS_PFSCP_ENABLE_SE051C2_0 #define_SSS_PFSCP_ENABLE_SE050F2_0	
#define SSS_PFSCP_ENABLE_SE051L_0005A8FA 0 #define SSS_PFSCP_ENABLE_SE051A_0001A920_0 #define_SSS_PFSCP_ENABLE_SE05E6_0001A021_1	
#define SSS_PFSCP_ENABLE_SE050E_0001A921 1 #define SSS_PFSCP_ENABLE_SE051W_0005A739 0 #define SSS_PFSCP_ENABLE_A5000_0004A736_0	
#define SSS_PFSCP_ENABLE_SE050F2_0001A92A 0	
Figure 26. Select the actual Platform SCP keys in the infsl_ss	s_ftr.h.in file

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 overwritting 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 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 <code>fsl_sss_ftr.h.in</code> CMake input file is persistent.

The location of the generated fsl_sss_ftr.h feature header file is: /home/pi/se_mw/ simw-top build/raspbian native se050 tloi2c

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

Table 12. Platform SCP key define prefix for SE050E product variants					
Variant OEF ID		Platform SCP key define to be set to '1'			
SE050E Dev. Board OM-SE050ARD-E	A921	SSS_PFSCP_ENABLE_SE050E_0001A921			
SE050E2 A921		SSS_PFSCP_ENABLE_SE050E_0001A921			

 Table 12. Platform SCP key define prefix for SE050E product variants

Table 13. Platform SCP key define prefix for SE050F product variants

Variant	OEF ID	Platform SCP key define to be set to '1'		
SE050F Dev.Board	A92A	SSS_PFSCP_ENABLE_SE050F2_0001A92A		
OM-SE050ARD-F				
SE050F2	A92A	SSS_PFSCP_ENABLE_SE050F2_0001A92A		

Table 14. Platform SCP key define prefix for SE050 Previous Generation product variants

Variant	OEF ID	Platform SCP key define to be set to '1'
SE050A1	A204	SSS_PFSCP_ENABLE_SE050A1

© NXP B.V. 2022. All rights reserved.

 Table 14. Platform SCP key define prefix for SE050 Previous Generation product

 variants...continued

Variant	OEF ID	Platform SCP key define to be set to '1'
SE050A2	A205	SSS_PFSCP_ENABLE_SE050A2
SE050B1	A202	SSS_PFSCP_ENABLE_SE050B1
SE050B2	A203	SSS_PFSCP_ENABLE_SE050B2
SE050C1	A200	SSS_PFSCP_ENABLE_SE050C1
SE050C2	A201	SSS_PFSCP_ENABLE_SE050C2
SE050 Dev Board	A1F4	SSS_PFSCP_ENABLE_SE050_DEVKIT
OM-SE050ARD		
SE050F2	A77E ^[1]	SSS_PFSCP_ENABLE_SE050F2

[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.

able 15.	Platform	SCP key	define	prefix for	SE051	product variants
----------	----------	---------	--------	------------	-------	------------------

Variant	OEF ID	Platform SCP key define to be set to '1'
SE051A2	A920	SSS_PFSCP_ENABLE_SE051A_0001A920
SE051C2	A8FA	SSS_PFSCP_ENABLE_SE051C_0005A8FA
SE051W2	A739	SSS_PFSCP_ENABLE_SE051W_0005A739
SE051A2	A565	SSS_PFSCP_ENABLE_SE051A2
SE051C2	A564	SSS_PFSCP_ENABLE_SE051C2

Table 16.	Platform SCP	key define p	orefix for A5000	product variants
-----------	--------------	--------------	------------------	------------------

Variant	OEF ID	Platform SCP key define to be set to '1'
A5000 Dev. Board OM-A5000ARD	A736	SSS_PFSCP_ENABLE_A5000_0004A736
A5000	A736	SSS_PFSCP_ENABLE_A5000_0004A736

5.2.2 Defining the default Platfrom SCP keys in a text file

For evaluation purpose the EdgeLock SE Plug & Trust Middleware supports to store the Platform SCP key in a plain text file. For further details see EdgeLock SE 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):

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/raspbian_native_se050_t1oi2c/bin

cd ~/se_mw/simw-top_build/raspbian_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_BOOT_SCP03_PATH is used to define the Platform SCP key textfile (filename and location).

export EX_SSS_BOOT_SCP03_PATH=~/se_mw/simw-top_build/
raspbian native se050 tloi2c/bin/se050E scp keys.txt

pi@raspberrypi:~/se_mw/simw-top_build/raspbian_native_se050_t1oi2c/bin \$ echo ENC 020863E7A0A5AED72A6460C4DFDCAF64 > se050E_scp_keys.txt
pi@raspberrypi:~/se_mw/simw-top_build/raspbian_native_se050_t1oi2c/bin \$ echo MC 730D57980E024180B24768514BFBA95B >> se050E_scp_keys.txt
pi@raspberrypi:~/se_mw/simw-top_build/raspbian_native_se050_t1oi2c/bin \$ echo MC 730DX67982AE0ESE7F47B42CED4E7F >> se050E_scp_keys.txt
Pi@raspberrypi:~/se_mw/simw-top_build/raspbian_native_se050_t1oi2c/bin \$ echo MC 730DX67982AE0ESE7F47B42CED4E7F >> se050E_scp_keys.txt
EKC 12086376AA5AED72A640caDFDCAF64
MC 730DX639A28CE0EF2F47442CED4E7F
EKC 120863764942CE0E427F
EKC 12086376442CE0H2FF
pi@raspberrypi:~/se_mw/simw-top_build/raspbian_native_se050_t1oi2c/bin \$ export EX_SS5_B00T_SCP03_PATH=~/se_mw/simw-top_build/raspbian_native_se050_tioi2c/bin \$ export EX_SS5_B00T_SCP03_PATH=~/se_mw/simw-top_build/raspbian_se050_tioi2c/

Figure 27. EdgeLock SE05xPlatform SCP plain text key file

Note: The EdgeLock SE Plug & Trust Middleware will first look for the default path /tmp/SE05X/plain_scp.txt, if it is not able to find the file, it will try to use the environment variable EX_SSS_BOOT_SCP03_PATH, and lastly, it will fall back to precompiled keys.

5.3 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.

The following images show the configuration for the SE050E development board OM-SE05ARD-E.

EdgeLock SE05x Quick start guide with Raspberry Pi



```
cd ~/se_mw/simw-top_build/raspbian_native_se050_t1oi2c
```

cmake-gui .

Update the CMake settings as explained above. Press first the *Configure* button and second the *Generate* button and close the CMake GUI.

```
cmake --build .
sudo make install
sudo ldconfig /usr/local/lib/
```

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
```

AN12570 Application note

Figure 29 shows the log output in case the Platform SCP keys are defined in the EdgeLock SE Plug & Trust Middleware source code (see <u>Section 5.2.1</u>).

pi@ra	spberry	pi:~/s	se mw/s	simw	-top b	ouild/	rasi	obian	nati	ve	se05	0 t	t1oi2c \$ cd bin	
pi@ras	spberry	pi:~/s	se mw/s	s imw	-top t	uild/	rasi	bian ⁻	nati	ve	se05(0 ⁻ t		
App	:INF0	:Plug/	AndTrus	st v	04.00	00 20	2110	926 -	-	_		_		
App	:INFO	:Runn i	ing ./s	se05	x_Min	imal								
Арр	:INFO	:If yo	ou want	t to	over.	<pre>ride</pre>	the	seled	tion	, u	se El	NV=	V=EX_SSS_BOOT_SSS_PORT or pass in command line arguments.	
Арр	:INFO	Using	g defau	ult	Platf	SCP03	key	5. You	ı can	use	e ke	ys	s from file using ENV=EX_SSS_BOOT_SCP03_PATH	
SSS	:INF0	:atr (Len=3	5)										
	01 A0	00 00	03	96	04 03	E8	00	FE 02	2	0B (93 E	80	00	
	01 00	00 00	00	64	13 88	0A	00	65 53	3	45 3	30 3!	53	31	
	00 00	00												
Арр	:INFO	:mem=3	32767											
Арр	:INF0	:se05>	(_Min ù	nal	Examp	le Suc	ces	5 !!!.						
Арр	:INFO	:ex_ss	ss Fin	ishe	d								_	
pi@ras	spberry	/pi:~/s	se_mw/s	s imw	/-top_t	ouild/	ras	obian_	nati	ve_s	se050	0_t	_t1oi2c/bin \$	
	-	-	_				-	-			_			

Figure 29. Run se05x_minimal example with platformSCP enabled - SCP keys defined in the EdgeLock SE Plug & Trust Middleware source

The log output for defining the Platform SCP keys via a text file (see <u>Section 5.2.2</u>) is shown in <u>Figure 30</u>.



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 (~/se_mw/simw-top/doc/ demos/se05x/se05x_RotatePlatformSCP03Keys/Readme.html). The example source code is available at ~/se_mw/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 (~/se_mw/simw-top/doc/demos/se05x/se05x_MandatePlatformSCP/Readme.html). The example source code is available at ~/se_mw/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/simw-top/doc/demos/se05x/se05x_AllowWithoutPlatformSCP/ Readme.html). The example source code is available at ~/se_mw/simw-top/ demos/se05x/se05x_AllowWithoutPlatformSCP.

6 Manage access from multiple Linux processes to the EdgeLock SE05x

The EdgeLock SE 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 EdgeLock SE Plug & Trust Middleware documentation chapter Access Manager: Manage access from multiple (Linux) processes to an SE05x IoT Applet for more details.

7 Appendix A: Using the ssscli tool

EdgeLock SE Plug & Trust Middleware also provides the ssscli tool. This tool can be used to interact with the EdgeLock SE05x security IC without having to write any code.

The ssscli is a command line tool that can be used to send commands to EdgeLock SE05x interactively through the command line. For example, you can use the ssscli to create keys and credentials in the EdgeLock SE05x security IC during evaluation, development and testing phases. The ssscli tool is written in Python and supports complex provisioning scripts that can be run in Windows, Linux, OS X and other embedded devices. It can be used to:

- Insert keys and certificates in DER or PEM format into the EdgeLock SE05x.
- Retrieve the public keys and certificates form EdgeLock SE05x and store the key into a DER (<u>Distinguished Encoding Rules</u>) or PEM (<u>Privacy Enhanced Mail</u>) formatted file.
- Create reference-keys and store the key into a DER or PEM formatted file.
- Delete EdgeLock SE05x (erase) keys and certificates inside.
- Generate keys inside the EdgeLock SE05x.
- · Attach policies to objects.
- List all EdgeLock SE05x secure objects.
- Retrieve the EdgeLock SE05x device unique ID.
- Run some basic EdgeLock SE05x operations like sign/verify and encrypt/decrypt operations.

Please refer to the EdgeLock SE Plug & Trust Middleware documentation chapter "9. CLI Tool" for detailed description how to use ssscli tool.

For installing the ssscli tool follow the steps below shown in Figure 31:

- 1. Move to the user directory
 - >> cd /home/pi

3. Ensure libffi-dev is installed:

>> sudo apt-get install libffi-dev Note: In this case, the packages were already installed

<pre>signaspherrypi:-/s muf 5 cd /home/pi 4</pre>	🗾 pi@192.168.1.151:22 - Bitvise xterm - pi@raspberrypi: ~	-		×
<pre>bigensphereppir= 5 sudo apt_get install python3-pip to the set of the se</pre>	pi@raspberrypi:~/se_mw \$ cd /home/pi 🖕 🚹			
<pre>Reading package lists Done Building dependency tree Reading state information Done Sython3-pip is already the newest version (18.1-5+rpt1). The following package were automatically installed and are no longer required: liberiv2-14 libgfortran3 libgmime-2.6-0 libncurses5 uuid-dev Se 'sud apt autoremove' to remove them. upgraded, 0 newly installed, 0 to remove and 0 not upgraded. if@raspherrypi:</pre>	pi@raspberrypi:~ \$ sudo apt-get install python3-pip 💶			
<pre>building dependency tree eading state information Done zython3-pip is already the newest version (18.1-5+rpt1). The following packages were automatically installed and are no longer required: libexiv2-14 libgfortran3 libgmime-2.6-0 libncurses5 uuid-dev se 'sudo apt autoremove' to remove and 0 not upgraded. Sigmaspherrypi:- \$ sudo apt_get install libffi-dev se teading packages lists Done Wilding dependency tree teading packages lists Done Wilding dependency tree teading packages usere automatically installed and are no longer required: libefi-dev is already the newest version (3.2.1-9). The following packages were automatically installed and are no longer required: libefi-dev 14 libgfortran3 libgmime-2.6-0 libncurses5 uuid-dev se 'sudo apt autoremove' to remove them. 9 upgraded, 0 newly installed, 0 to remove and 0 not upgraded. Di@raspberrypi:- \$ Figure 31. Install python and libffi-dev Make sure you have cmake installed and configured for the Raspbian Host as dor Section 3.3. Ensure click, cryptography and func-timeout modules are installed. Figure 32 show now to install these modules, change directory to: > cd /home/pi/se_mw/simw-top/pycli ind run the following command: > pip3 install -r requirements.txt pip13 install -r requirements.txt pip13 install -r requirements.txt pip13 install -r requirements.txt pip13 install -r nequirements.txt pip14:04.115122-Bitwistem-pi@raspberppi-/se_mv/simw-top/pycli hoflowing package lists Done Wilding dependency tree teading state information Done ibexiv2-14 libgfortran3 libgmime-2.6-0 libncurses5 uuid-dev se 'sudo apt autoremove' to remove and 0 not upgraded. ibgraspberrypi:- \$ cd /home/pi/se_mv/simw-top/pycli hording packages were automatically installed and are no longer required: libexiv2-14 libfortran3 libgmime-2.6-0 libncurses5 uuid-dev se 'sudo apt autoremove' to remove them. huggraded, 0 newly installed, 0 to remove and 0 not upgraded. iggraspherrypi:- \$ cd /home/pi/se_mv/simm-top/pycli</pre>	Reading package lists Done			
<pre>Reading state information Done yothon3-pip is already the newest version (18.1-5+rpt1). The following packages were automatically installed and are no longer required: libexiv2-14 libgfortran3 libgmime-2.6-0 libncurses5 uuid-dev ligenapherrypir-\$ sudo aptrget install libffi-dev Reading package lists Done Wilding dependency tree Reading state information Done libffi-dev is already the newest version (3.2.1-9). The following packages were automatically installed and are no longer required: libexiv2-14 libgfortran3 libgmime-2.6-0 libncurses5 uuid-dev ligenspherrypir-\$ sudo aptrget install libffi-dev ligenspherrypir-\$ sudo aptrget install libffi-dev ligenspherrypir > 5 uud-dev ligenspherrypir > 6 uud-dev ligens</pre>	Building dependency tree			
<pre>python3-pip is already the newest version (18.1-5+rpt1). The following packages were automatically installed and are no longer required: libexiv2-14 libgfortran3 libgmime-2.6-0 libncurses5 uuid-dev Jse 'sudo apt autoremove' to remove and 0 not upgraded. Digraspherrypi-5 sudo apt get install libffi-dev Wilding dependency tree teading package lists Done libffi-dev is already the newest version (3.2.1-9). The following packages were automatically installed and are no longer required: libexiv2-14 libgfortran3 libgmime-2.6-0 libncurses5 uuid-dev Jse 'sudo apt autoremove' to remove them. Jugraspherrypi-5 J</pre>	Reading state information Done			
The following packages were automatically installed and are no longer required: libexiv2-14 libgfortran3 libgmime-2.6-0 libncurses5 uuid-dev augraded, 0 newly installed, 0 to remove and 0 not upgraded. bi@raspberrypi: \$ sudo apt-get install libffi-dev audition of the newset version (3.2.1-9). The following packages were automatically installed and are no longer required: libexiv2-14 libgfortran3 libgmime-2.6-0 libncurses5 uuid-dev se 'sudo apt autoremove' to remove them. bi@raspberrypi: \$ Figure 31. Install python and libffi-dev Make sure you have cmake installed and configured for the Raspbian Host as dor Section 3.3. Ensure click, cryptography and func-timeout modules are installed. Figure 32 show low to install these modules, change directory to: > cd /home/pi/se_mw/simw-top/pycli und run the following command: > pigraded, 0 newly installed, 0 to remove and 0 not upgraded. bi@raspberrypi:- \$ sudo apt get installed and configured for the Raspbian Host as dor Section 3.3. Ensure click, cryptography and func-timeout modules are installed. Figure 32 show low to install these modules, change directory to: > cd /home/pi/se_mw/simw-top/pycli und run the following command: > pip3 install -r requirements.txt pigraspberrypi:- \$ sudo apt-get install libffi-dev teading package lists Done bilding dependency tree teading state information Done libffi-dev is already the newest version (3.2.1-9). The following packages were automatically installed and are no longer required: libexiv2-14 libgfortran3 libmme-2.6-0 libncurses5 uuid-dev se 'sudo apt autoremove' to remove and 0 not upgraded. bigraspberrypi:- \$ cd /home/pi/se mw/simw-top/pycli	python3-pip is already the newest version (18.1-5+rpt1).			
<pre>IDEXIV-14 IDEGrotTrans IDEGRIME-2.6-0 IDECLIVESES UNID-GEV 39 yaraded, 0 newly installed, 0 to remove and 0 not upgraded. Di@raspberrypi:- \$ sudo apt-get install libffi-dev Seliding dependency tree Reading state information Done UDEGRIM Dependency tree Seliding aperages were automatically installed and are no longer required: libexiv2-14 libgfortran3 libgmime-2.6-0 libncurses5 unid-dev Se 'sudo apt autoremove' to remove them. upgraded, 0 newly installed, 0 to remove and 0 not upgraded. Di@raspberrypi:- \$ Figure 31. Install python and libffi-dev Make sure you have cmake installed and configured for the Raspbian Host as dor Section 3.3. Ensure click, cryptography and func-timeout modules are installed. Figure 32 show iow to install these modules, change directory to: > cd /home/pi/se_mw/simw-top/pycli ind run the following command: > pip3 install -r requirements.txt pip3 install -r requirements.txt pigureded, 0 newly installed, 0 to remove and 0 not upgraded. id@raspberrypi:- \$ sudo apt-get install libffi-dev wugraded, 0 newly installed, 0 to remove and 0 not upgraded. id@raspberrypi:- \$ upgraded, 0 newly installed, 0 to remove and 0 not upgraded. id@raspberrypi:- \$ upgraded, 0 newly installed, 0 to remove and 0 not upgraded. id@raspberrypi:- \$ upgraded, 0 newly installed, 0 to remove and 0 not upgraded. id@raspberrypi:- \$ upgraded, 0 newly installed, 0 to remove and 0 not upgraded. id@raspberrypi:- \$ upgraded, 0 newly installed, 0 to remove and 0 not upgraded. id@raspberrypi:- \$ upgraded, 0 newly installed, 0 to remove and 0 not upgraded. id@raspberrypi:- \$ upgraded, 0 newly installed, 0 to remove and 0 not upgraded. id@raspberrypi:- \$ upgraded, 0 newly installed, 0 to remove and 0 not upgraded. id@raspberrypi:- \$ upgraded, 0 newly installed, 0 to remove and 0 not upgraded. id@raspberrypi:- \$ upgraded, 0 newly installed, 0 to remove and 0 not upgraded. id@raspberrypi:- \$ upgraded, 0 newly installed, 0 to remove and 0 not</pre>	The following packages were automatically installed and are no longer require	ed :		
<pre>Jse suco apt autoremove to remove and 0 not upgraded. Digrapherypi:- \$ sudo apt-get install libffi-dev set Reading package lists Done Building dependency tree Reading state information Done libffi-dev is already the newest version (3.2.1-9). The following packages were automatically installed and are no longer required: libexiv2-14 libgfortran3 libgmime-2.6-0 libncurses5 uuid-dev Jse 'sudo apt autoremove' to remove them. 9 upgraded, 0 newly installed, 0 to remove and 0 not upgraded. 9 upgraded, 0 newly installed, 0 to remove and 0 not upgraded. 9 upgraded, 0 newly installed, 0 to remove and 0 not upgraded. 9 upgraded, 0 newly installed, 0 to remove and 0 not upgraded. 9 upgraded, 0 newly installed, 0 to remove and 0 not upgraded. 9 upgraded, 0 newly installed, 0 to remove and 0 not upgraded. 9 upgraded, 0 newly installed, 0 to remove and 0 not upgraded. 9 upgraded, 0 newly installed, 0 to remove and 0 not upgraded. 9 upgraded, 0 newly installed, 0 to remove and 0 not upgraded. 9 upgraded, 0 newly install these modules, change directory to: 9 cd /home/pi/se_mw/simw-top/pycli 9 upgraded, 0 newly install -r requirements.txt 9 pip3 install -r requirements.txt 9 pip3 install -r requirements.txt 9 upgraded, 0 newly installed, 0 to remove and 0 not upgraded. 9 upgraded, 0 newly installed, 0 to remove and 0 not upgraded. 9 upgraded, 0 newly installed, 0 to remove and 0 not upgraded. 9 upgraded information Done 9 ubfridev is already the newest version (3.2.1-9). 9 he following packages were automatically installed and are no longer required: 11bexiv2-14 libgfortran3 libgmime-2.6-0 libncursesS uuid-dev 9 upgraded, 0 newly installed, 0 to remove and 0 not upgraded. 10ergraded, 0 newly installed, 0 to remove and 0 not upgraded. 10ergraded, 0 newly installed, 0 to remove and 0 not upgraded. 10ergraded, 0 newly installed, 0 to remove and 0 not upgraded. 10ergraded, 0 newly installed, 0 to remove and 0 not upgraded. 10ergraded, 0 newly installed, 0 to remove and 0 not upgraded. 10ergraded, 0 newly installed, 0 t</pre>	libexiv2-14 libgfortran3 libgmime-2.6-0 libncurses5 uuid-dev			
<pre>b upgraded, 0 mely installed, 0 to remove and 0 not upgraded. iggraspherypiir. S sudo apt-get install libffi-dev) teading package lists Done Building dependency tree Reading state information Done libffi-dev is already the newest version (3.2.1-9). The following packages were automatically installed and are no longer required: libexiv2-14 libgfortran3 libgnime-2.6-0 libncurses5 uuid-dev Jse 'sudo apt autoremove' to remove them. 9 upgraded, 0 newly installed, 0 to remove and 0 not upgraded. 10 upgraded, 0 newly installed, 0 to remove and 0 not upgraded. 10 upgraded, 0 newly installed, 0 to remove and 0 not upgraded. 10 upgraded, 0 newly installed, 0 to remove and 0 not upgraded. 10 upgraded, 0 newly installed, 0 to remove them. 9 upgraded, 0 newly installed, 0 to remove them. 9 upgraded, 0 newly installed, 0 to remove them configured for the Raspbian Host as dor 10 section 3.3. Ensure click, cryptography and func-timeout modules are installed. Figure 32 show 10 wto install these modules, change directory to: 10 cd /home/pi/se_mw/simw-top/pycli 11 nd run the following command: 12 pip192 install -r requirements.txt 13 pip192.108.1151/22-8hvisetem -pi@respberypk-/se_mw/simw-top/pycli 14 upgraded, 0 newly installed, 0 to remove and 0 not upgraded. 14 upgraded, 0 newly installed, 0 to remove and 0 not upgraded. 14 upgraded, 0 newly installed, 0 to remove and 0 not upgraded. 14 upgraded information Done 14 biffi-dev is already the newest version (3.2.1-9). 15 following packages were automatically installed and are no longer required: 15 bisviz-14 libgfortran3 libgmine-2.6-0 libncurses5 uuid-dev 16 se show and a not upgraded. 16 onewly installed to remove and 0 not upgraded. 16 septervpi:</pre>	Use sudo apt autoremove to remove them.			
<pre>indiang package lists Done Wilding dependency tree Reading state information Done libffi-dev is already the newest version (3.2.1-9). The following packages were automatically installed and are no longer required: libexiv2-14 libgfortran3 libgmime-2.6-0 libncurses5 uuid-dev se 'sudo apt autoremove' to remove them.) upgraded, 0 newly installed, 0 to remove and 0 not upgraded. ideraspberrypi:</pre>	ni@nashbannyni:v \$ sudo ant-get install libffi-dev			
<pre>Building dependency tree Reading state information Done libffi-dev is already the newest version (3.2.1-9). The following packages were automatically installed and are no longer required: libexiv2-14 libgfortranal libgmime-2.6-0 libncursesS uuid-dev See 'sudo apt autoremove' to remove them. 9 upgraded, 0 newly installed, 0 to remove and 0 not upgraded. bi@raspberrypi:~ \$ Figure 31. Install python and libffi-dev Make sure you have cmake installed and configured for the Raspbian Host as dor Section 3.3. Ensure click, cryptography and func-timeout modules are installed. Figure 32 show now to install these modules, change directory to: >> cd /home/pi/se_mw/simw-top/pycli und run the following command: >> pip3 install -r requirements.txt pigrased, 0 newly installed, 0 to remove and 0 not upgraded. bi@raspberrypi:~ \$ sudo apt.get install libffi-dev teading package lists Done building dependency tree teading state information Done .ibffi-dev is already the newest version (3.2.1-9). The following package were automatically installed and are no longer required: libexiv2-14 libgfortran3 libgmime-2.6-0 libncursesS uuid-dev Se 'sudo apt autoremove' to remove and 0 not upgraded. i@raspberrypi:~ \$ cd /home/pi/se mw/simw-top/pycli } upgraded, 0 newly installed, 0 to remove and 0 not upgraded. i@raspberrypi:~ \$ cd /home/pi/se mw/simw-top/pycli } }</pre>	Reading nackage lists Done			
<pre>Reading state information Done libffi-dev is already the newest version (3.2.1-9). The following packages were automatically installed and are no longer required: libexiv2-14 libgfortran3 libgmime-2.6-0 libncurses5 uuid-dev Jse 'sudo apt autoremove' to remove them. 9 upgraded, 0 newly installed, 0 to remove and 0 not upgraded. pi@raspberrypi:~ \$ Figure 31. Install python and libffi-dev Make sure you have cmake installed and configured for the Raspbian Host as dor Section 3.3. Ensure click, cryptography and func-timeout modules are installed. Figure 32 show now to install these modules, change directory to: .> cd /home/pi/se_mw/simw-top/pycli ind run the following command: .> pip3 install -r requirements.txt pi@raspberrypi:~ \$udo apt-get install libfi-dev teading package lists Done uilding dependency tree teading package lists Done libfi-dev is already the newest version (3.2.1-9). The following packages were automatically installed and are no longer required: libexiv2-14 libgfortran3 libgmime-2.6-0 libncurses5 uuid-dev tse 'sudo apt autoremove' to remove them. yugraded, 0 newly installed, 0 to remove and 0 not upgraded. i@raspberrypi:~ \$udo apt-get common (3.2.1-9). The following packages were automatically installed and are no longer required: libexiv2-14 libgfortran3 libgmime-2.6-0 libncurses5 uuid-dev tse 'sudo apt autoremove' to remove them. yugraded, 0 newly installed, 0 to remove and 0 not upgraded. i@raspberrypi:~ \$ cd /home/pi/se_mw/simw-top/pycli</pre>	Building dependency tree			
<pre>libffi-dev is already the newest version (3.2.1-9). The following packages were automatically installed and are no longer required: libexiv2-14 libgfortran3 libgmime-2.6-0 libncurses5 uuid-dev Jse 'sudo apt autoremove' to remove them. 0 upgraded, 0 newly installed, 0 to remove and 0 not upgraded. 0 ideraspberrypi:~ \$ Figure 31. Install python and libffi-dev Make sure you have cmake installed and configured for the Raspbian Host as dor Section 3.3. Ensure click, cryptography and func-timeout modules are installed. Figure 32 show now to install these modules, change directory to: >> cd /home/pi/se_mw/simw-top/pycli und run the following command: >> pip13 install -r requirements.txt pip092168.1.51:22-Bivisextem-pi@rspberrypi./se_mw/simw-top/pycli 0 upgraded, 0 newly installed, 0 to remove and 0 not upgraded. 0 upgraded, 0 newly installed, 0 to remove and 0 not upgraded. 0 upgraded, 0 newly installed, 0 to remove and 0 not upgraded. 0 upgraded, 0 newly installed, 0 to remove and 0 not upgraded. 0 upgraded, 0 newly installed, 0 to remove and 0 not upgraded. 0 upgraded, 0 newly installed, 0 to remove and 0 not upgraded. 0 upgraded, 0 newly installed, 0 to remove and 0 not upgraded. 0 upgraded, 0 newly installed, 0 to remove and 0 not upgraded. 0 upgraded, 0 newly installed, 0 to remove and 0 not upgraded. 0 upgraded, 0 newly installed, 0 to remove and 0 not upgraded. 0 upgraded, 0 newly installed, 0 to remove and 0 not upgraded. 0 upgraded, 0 newly installed, 0 to remove and 0 not upgraded. 0 upgraded, 0 newly installed, 0 to remove and 0 not upgraded. 0 upgraded, 0 newly installed, 0 to remove them. 0 upgraded, 0 newly installed, 0 to remove them. 0 upgraded, 0 newly installed, 0 to remove and 0 not upgraded. 0 upgraded, 0 newly installed, 0 to remove and 0 not upgraded. 0 upgraded, 0 newly installed, 0 to remove and 0 not upgraded. 0 upgraded, 0 newly installed, 0 to remove and 0 not upgraded. 0 upgraded, 0 newly installed, 0 to remove and 0 not upgraded. 0 upgraded, 0 newly installed, 0 to remove and 0 not u</pre>	Reading state information Done			
The following packages were automatically installed and are no longer required: libexiv2-14 libgfortran3 libgmime-2.6-0 libncurses5 uuid-dev Jse 'sudo apt autoremove' to remove them. a upgraded, 0 newly installed, 0 to remove and 0 not upgraded. bi@raspberrypi:-\$ Figure 31. Install python and libffi-dev Make sure you have cmake installed and configured for the Raspbian Host as dor Section 3.3. Ensure click, cryptography and func-timeout modules are installed. Figure 32 show how to install these modules, change directory to: > cd /home/pi/se_mw/simw-top/pycli and run the following command: > pip13 install -r requirements.txt pi@192168.1.51:22-Bitviexterm-pi@raspberrypi-/se_mw/simw-top/pycli > upgraded, 0 newly installed, 0 to remove and 0 not upgraded. bidfrage lists Done Bitlifi dev is already the newest version (3.2.1-9). The following package lists Done libfifi-dev is already the newest version (3.2.1-9). The following packages were automatically installed and are no longer required: libexiv2-14 libgfortran3 libgmime-2.6-0 libncurses5 uuid-dev lse 'sudo apt autoremove' to remove them. > upgraded, 0 newly installed, 0 to remove and 0 not upgraded. bidfig dependency tree teading state information Done libfif-dev is already the newest version (3.2.1-9). The following packages were automatically installed and are no longer required: libexiv2-14 libgfortran3 libgmime-2.6-0 libncurses5 uuid-dev lse 'sudo apt autoremove' to remove them. > upgraded, 0 newly installed, 0 to remove and 0 not upgraded. 1@raspberrypi:-\$ cd /home/pi/se_mw/simw-top/pycli	libffi-dev is already the newest version (3.2.1-9).			
<pre>libexiv2-14 libgfortran3 libgmime-2.6-0 libncurses5 uuid-dev Jse 'sudo apt autoremove' to remove them. a upgraded, 0 newly installed, 0 to remove and 0 not upgraded. pi@raspberrypi:~ \$ Figure 31. Install python and libffi-dev Make sure you have cmake installed and configured for the Raspbian Host as dor Section 3.3. Ensure click, cryptography and func-timeout modules are installed. Figure 32 show now to install these modules, change directory to:</pre>	The following packages were automatically installed and are no longer require	ed :		
Jse 'sudo apt autoremove' to remove them. ∂ upgraded, 0 newly installed, 0 to remove and 0 not upgraded. Di@raspberrypi:~ \$ Figure 31. Install python and libffi-dev Make sure you have cmake installed and configured for the Raspbian Host as dor Section 3.3. Ensure click, cryptography and func-timeout modules are installed. Figure 32 shown now to install these modules, change directory to: -> cd /home/pi/se_mw/simw-top/pycli and run the following command: -> pip13 install -r requirements.txt pip192.168.1151:22-Bitvise.stempi@rspberrypi:~/se_mw/simw-top/pycli 	libexiv2-14 libgfortran3 libgmime-2.6-0 libncurses5 uuid-dev			
<pre>a upgraded, 0 newly installed, 0 to remove and 0 not upgraded. bi@raspberrypi:~ \$ Figure 31. Install python and libffi-dev Make sure you have cmake installed and configured for the Raspbian Host as dor Section 3.3. Ensure click, cryptography and func-timeout modules are installed. Figure 32 show now to install these modules, change directory to: > cd /home/pi/se_mw/simw-top/pycli und run the following command: > pip3 install -r requirements.txt pi@192.168.1.151:22-Bitvisextem-pi@rspberrypi:~/se_mw/simw-top/pycli -</pre>	Use 'sudo apt autoremove' to remove them.			
<pre>Figure 31. Install python and libffi-dev Make sure you have cmake installed and configured for the Raspbian Host as dor Section 3.3. Ensure click, cryptography and func-timeout modules are installed. Figure 32 show iow to install these modules, change directory to: >> cd /home/pi/se_mw/simw-top/pycli and run the following command: >> pip3 install -r requirements.txt pip13 install -r requirements.txt pip12.168.1.151:22-Bitvisexterm -pi@raspberrypi ~/se_mw/simw-top/pycli >> upgraded, 0 newly installed, 0 to remove and 0 not upgraded. >>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>></pre>	0 upgraded, 0 newly installed, 0 to remove and 0 not upgraded.			
<pre>Figure 31. Install python and libffi-dev Make sure you have cmake installed and configured for the Raspbian Host as dor Section 3.3. Ensure click, cryptography and func-timeout modules are installed. Figure 32 show now to install these modules, change directory to: > cd /home/pi/se_mw/simw-top/pycli and run the following command: > pip3 install -r requirements.txt pip12.108.1.151:22-Bitvisextem-pi@raspberypi-v/se_mw/simw-top/pycli -</pre>	pi@raspberrypi:~ \$			
<pre>Ensure click, cryptography and func-timeout modules are installed. Figure 32 show now to install these modules, change directory to: >> cd /home/pi/se_mw/simw-top/pycli and run the following command: >> pip3 install -r requirements.txt pip3 install -r requirements.txt pip412.168.115h22-Bitvisexterm-pi@raspberrypi:~/se_mw/simw-top/pycli > upgraded, 0 newly installed, 0 to remove and 0 not upgraded. Di@raspberrypi:~ \$ sudo apt-get install libffi-dev teading package lists Done Building dependency tree teading state information Done libffi-dev is already the newest version (3.2.1-9). The following packages were automatically installed and are no longer required: libexiv2-14 libgfortran3 libgmime-2.6-0 libncurses5 uuid-dev Ise 'sudo apt autoremove' to remove and 0 not upgraded. pi@raspberrypi:~ \$ cd /home/pi/se_mw/simw-top/pycli }</pre>	Make sure you have cmake installed and configured for the Base	abian Uaat a		
<pre>install click, cryptography and func-timeout modules are installed. Figure 52 show now to install these modules, change directory to: >> cd /home/pi/se_mw/simw-top/pycli and run the following command: >> pip3 install -r requirements.txt pip132.168.1.151:22-Bitvise xterm-pi@raspberypi:~/se_mw/simw-top/pycli >> upgraded, 0 newly installed, 0 to remove and 0 not upgraded. >> pi@raspberrypi:~ \$ sudo apt-get install libffi-dev teading package lists Done Building dependency tree teading state information Done libffi-dev is already the newest version (3.2.1-9). The following packages were automatically installed and are no longer required: libexiv2-14 libgfortran3 libgmime-2.6-0 libncurses5 uuid-dev Ise 'sudo apt autoremove' to remove them. >> upgraded, 0 newly installed, 0 to remove and 0 not upgraded. >> upgraded, 0 newly installed, 0 to remove and 0 not upgraded. >> upgraded, 0 newly installed, 0 to remove and 0 not upgraded. >> upgraded, 0 newly installed, 0 to remove and 0 not upgraded. >>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>></pre>	Section 3.3	opian nosta	s uo	ne
<pre>who is to install these modules, change directory to: >> cd /home/pi/se_mw/simw-top/pycli ind run the following command: >> pip3 install -r requirements.txt ===================================</pre>	Section 3.3. Section 3.3.		s uu	ne
<pre>>> cd /home/pi/se_mw/simw-top/pycli and run the following command: >> pip3 install -r requirements.txt pip12.168.1.151:22-Bitvisexterm-pi@raspberrypi~/se_mw/simw-top/pycli</pre>	Section 3.3. Ensure click, cryptography and func-timeout modules are installe	ed. <u>Figure 32</u>	s do	one ow:
<pre>and run the following command: >> pip3 install -r requirements.txt apping installed, 0 to remove and 0 not upgraded. bigmaspherrypi:~ \$ sudo apt-get install libffi-dev teading package lists Done Wilding dependency tree teading state information Done libffi-dev is already the newest version (3.2.1-9). The following packages were automatically installed and are no longer required: libexiv2-14 libgfortran3 libgmime-2.6-0 libncurses5 uuid-dev lse 'sudo apt autoremove' to remove them.) upgraded, 0 newly installed, 0 to remove and 0 not upgraded. bigmaspherrypi:~ \$ cd /home/pi/se_mw/simw-top/pycli } </pre>	Section 3.3. Ensure click, cryptography and func-timeout modules are installen now to install these modules, change directory to:	ed. <u>Figure 32</u>	s do shc	one ow:
<pre>>> pip3 install -r requirements.txt pip3 install -r requirements.txt pip13 installed, 0 to remove and 0 not upgraded. pip13 installed, 0 to remove and 0 not upgraded. pip13 installed, 0 to remove and 0 not upgraded. pip13 installed, 0 to remove and 0 not upgraded. pip13 installed, 0 to remove and 0 not upgraded. pip13 installed, 0 to remove and 0 not upgraded. pip13 installed, 0 to remove and 0 not upgraded. pip13 installed, 0 to remove and 0 not upgraded. pip13 installed, 0 to remove and 0 not upgraded. pip13 installed, 0 to remove and 0 not upgraded. pip13 installed, 0 to remove and 0 not upgraded. pip13 installed, 0 to remove and 0 not upgraded. pip13 installed, 0 to remove and 0 not upgraded. pip13 installed, 0 to remove and 0 not upgraded. pip13 installed, 0 to remove and 0 not upgraded. pip13 installed, 0 to remove and 0 not upgraded. pip13 installed, 0 to remove and 0 not upgraded. pip13 installed, 0 to remove and 0 not upgraded. pip13 installed, 0 to remove and 0 not upgraded. pip13 installed, 0 to remove and 0 not upgraded. pip13 installed, 0 to remove and 0 not upgraded. pip13 installed, 0 to remove and 0 not upgraded. pip13 installed, 0 to remove and 0 not upgraded. pip13 installed, 0 to remove and 0 not upgraded. pip13 installed, 0 to remove and 0 not upgraded. pip13 installed, 0 to remove and 0 not upgraded. pip13 installed, 0 to remove and 0 not upgraded. pip13 installed, 0 to remove and 0 no</pre>	Section 3.3. Ensure click, cryptography and func-timeout modules are installen how to install these modules, change directory to: >> cd /home/pi/se mw/simw-top/pycli	ed. <u>Figure 32</u>	s do shc	ome
<pre>i i i i i i i i i i i i i i i i i i i</pre>	Section 3.3. Ensure click, cryptography and func-timeout modules are installen how to install these modules, change directory to: >> cd /home/pi/se_mw/simw-top/pycli and run the following command:	ed. <u>Figure 32</u>	s do	one
<pre>pi@192.168.1.151:22 - Bivise xterm - pi@raspberrypi: -/se_mw/simw-top/pycli</pre>	Section 3.3. Ensure click, cryptography and func-timeout modules are installenew to install these modules, change directory to: >> cd /home/pi/se_mw/simw-top/pycli and run the following command: >> pip3 install -r requirements.txt	ed. <u>Figure 32</u>	s do	one ow:
<pre>0 upgraded, 0 newly installed, 0 to remove and 0 not upgraded. bi@raspberrypi:~ \$ sudo apt-get install libffi-dev Reading package lists Done Building dependency tree Reading state information Done libffi-dev is already the newest version (3.2.1-9). The following packages were automatically installed and are no longer required: libexiv2-14 libgfortran3 libgmime-2.6-0 libncurses5 uuid-dev Jse 'sudo apt autoremove' to remove them. to upgraded, 0 newly installed, 0 to remove and 0 not upgraded. pi@raspberrypi:~ \$ cd /home/pi/se_mw/simw-top/pycli</pre>	Section 3.3. Ensure click, cryptography and func-timeout modules are installent how to install these modules, change directory to: >> cd /home/pi/se_mw/simw-top/pycli and run the following command: >> pip3 install -r requirements.txt	ed. <u>Figure 32</u>	s do	one ow:
<pre>pi@raspberrypi:~ \$ sudo apt-get install libffi-dev Reading package lists Done Juilding dependency tree Reading state information Done Libffi-dev is already the newest version (3.2.1-9). The following packages were automatically installed and are no longer required: libexiv2-14 libgfortran3 libgmime-2.6-0 libncurses5 uuid-dev Jse 'sudo apt autoremove' to remove them. upgraded, 0 newly installed, 0 to remove and 0 not upgraded. <u>pi@raspberrypi:~ \$ cd /home/pi/se_mw/simw-top/pycli</u> </pre>	Section 3.3. Ensure click, cryptography and func-timeout modules are installen how to install these modules, change directory to: >> cd /home/pi/se_mw/simw-top/pycli and run the following command: >> pip3 install -r requirements.txt pip3 install -r requirements.txt	ed. <u>Figure 32</u>	s do <u>s</u> shc	one ow:
<pre>Reading package lists Done Building dependency tree Reading state information Done libffi-dev is already the newest version (3.2.1-9). The following packages were automatically installed and are no longer required: libexiv2-14 libgfortran3 libgmime-2.6-0 libncurses5 uuid-dev Jse 'sudo apt autoremove' to remove them. upgraded, 0 newly installed, 0 to remove and 0 not upgraded. <u>ni@raspberrypire \$ cd /home/pi/se_mw/simw-top/pycli</u> </pre>	Section 3.3. Ensure click, cryptography and func-timeout modules are installen now to install these modules, change directory to: >> cd /home/pi/se_mw/simw-top/pycli and run the following command: >> pip3 install -r requirements.txt pip3 install -r requirements.txt pip3 install -r requirements.txt	ed. <u>Figure 32</u>	s do <u>s</u> shc	ww ww
Sullarg dependency tree Reading state information Done libffi-dev is already the newest version (3.2.1-9). The following packages were automatically installed and are no longer required: libexiv2-14 libgfortran3 libgmime-2.6-0 libncurses5 uuid-dev Jse 'sudo apt autoremove' to remove them. 9 upgraded, 0 newJy installed, 0 to remove and 0 not upgraded. <u>vi@raspberrypire \$ cd /home/pi/se_mw/simw-top/pycli</u>	<pre>Section 3.3. Ensure click, cryptography and func-timeout modules are installe now to install these modules, change directory to: >> cd /home/pi/se_mw/simw-top/pycli and run the following command: >> pip3 install -r requirements.txt pip3 install -r requirements.txt pip3 install of remove and 0 not upgraded. pi@raspberrypi:~\$ sudo apt-get install libffi-dev</pre>	ed. <u>Figure 32</u>	s do <u>s</u> shc	one ow:
<pre>veading state information bone libffi-dev is already the newest version (3.2.1-9). The following packages were automatically installed and are no longer required: libexiv2-14 libgfortran3 libgmime-2.6-0 libncurses5 uuid-dev Jse 'sudo apt autoremove' to remove them.) upgraded, 0 newly installed, 0 to remove and 0 not upgraded. <u>vi@raspberrypir~ \$ cd /home/pi/se_mw/simw-top/pycli</u> / 4</pre>	<pre>Section 3.3. Ensure click, cryptography and func-timeout modules are installe now to install these modules, change directory to: >> cd /home/pi/se_mw/simw-top/pycli and run the following command: >> pip3 install -r requirements.txt pip192.168.1.151:22-Bitvise xterm - pi@raspberrypi:~/se_mw/simw-top/pycli 0 upgraded, 0 newly installed, 0 to remove and 0 not upgraded. pi@raspberrypi:~ \$ sudo apt-get install libffi-dev Reading package lists Done</pre>	ed. <u>Figure 32</u>	s do shc	ow:
The following packages were automatically installed and are no longer required: The following packages were automatically installed and are no longer required: The following packages were automatically installed and are no longer required: The following packages were automatically installed and are no longer required: The following packages were automatically installed and are no longer required: The following packages were automatically installed and are no longer required: The following packages were automatically installed and are no longer required: The following packages were automatically installed and are no longer required: The following packages were automatically installed and are no longer required: The following packages were automatically installed and are no longer required: The following packages were automatically installed and are no longer required: The following packages were automatically installed and are no longer required: The following packages were automatically installed and are no longer required: The following packages were automatically installed and are no longer required: The following packages were automatically installed and are no longer required: The following packages were automatically installed and are no longer required: The following packages were automatically installed and are no longer required: The following packages were automatically installed and are no longer required: The following packages were automatically installed and are no longer required: The following packages were automatically installed and are no longer required: The following packages were automatically installed and are no longer required: The following packages were automatically installed and are no longer required: The following packages were automatically installed and are no longer required: The following packages were automatically installed and are no longer required: The following packages were automaticalled and are no longer required: The following packages were au	<pre>Section 3.3. Ensure click, cryptography and func-timeout modules are installe now to install these modules, change directory to: >> cd /home/pi/se_mw/simw-top/pycli and run the following command: >> pip3 install -r requirements.txt pip192.168.1.151:22 - Bitvise xterm - pi@raspberrypi: ~/se_mw/simw-top/pycli 0 upgraded, 0 newly installed, 0 to remove and 0 not upgraded. pi@raspberrypi:~ \$ sudo apt-get install libffi-dev Reading package lists Done Building dependency tree</pre>	ed. <u>Figure 32</u>	s do shc	•ne •w:
libexiv2-14 libgfortran3 libgmime-2.6-0 libncurses5 uuid-dev Jse 'sudo apt autoremove' to remove them.) upgraded, 0 newly installed, 0 to remove and 0 not upgraded. <u>i@raspberrypir~ \$ cd /home/pi/se_mw/simw-top/pycli</u>	Section 3.3. Ensure click, cryptography and func-timeout modules are installent how to install these modules, change directory to: >> cd /home/pi/se_mw/simw-top/pycli and run the following command: >> pip3 install -r requirements.txt pip192.168.1.151:22 - Bitvise xterm - pi@raspberrypi: ~/se_mw/simw-top/pycli 0 upgraded, 0 newly installed, 0 to remove and 0 not upgraded. pi@raspberrypi:~ \$ sudo apt-get install libffi-dev Reading package lists Done Building dependency tree Reading state information Done Piblifi dow is a ploady the acuract warriage (2.3.1.0)	ed. <u>Figure 32</u>	<u>s</u> shc	•ne •••••
Je 'sudo apt autoremove' to remove them. a) upgraded, 0 newly installed, 0 to remove and 0 not upgraded. <u>Di@raspberrypir~ \$ cd /home/pi/se_mw/simw-top/pycli</u>	<pre>Section 3.3. Ensure click, cryptography and func-timeout modules are installe now to install these modules, change directory to: >> cd /home/pi/se_mw/simw-top/pycli and run the following command: >> pip3 install -r requirements.txt pip3 install -r requirements.txt pipaspherrypi:~ \$ sudo apt-get install libffi-dev Reading package lists Done Building dependency tree Reading state information Done libffi-dev is already the newest version (3.2.1-9). The following approxemant is the package install preserves in the package is already the newest version (3.2.1-9).</pre>	ed. <u>Figure 32</u>	<u>s do</u>	•ne
0 upgraded, 0 newly installed, 0 to remove and 0 not upgraded. <u>pi@raspberrypi:~ \$ cd /home/pi/se_mw/simw-top/pycli</u>	<pre>Section 3.3. Ensure click, cryptography and func-timeout modules are installe now to install these modules, change directory to: >> cd /home/pi/se_mw/simw-top/pycli and run the following command: >> pip3 install -r requirements.txt pip3 install -r requirements.txt pip492168.1.151:22-Bitvisexterm-pi@raspberrypi:~/se_mw/simw-top/pycli 0 upgraded, 0 newly installed, 0 to remove and 0 not upgraded. pi@raspberrypi:~ \$ sudo apt-get install libffi-dev Reading package lists Done Building dependency tree Reading state information Done libffi-dev is already the newest version (3.2.1-9). The following packages were automatically installed and are no longer require libery2-14 libefortranal libergime-2 6-0 libergrees5 unid-dev</pre>	ed. <u>Figure 32</u>	s do	•ne
pi@raspberrypi:~ \$ cd /home/pi/se_mw/simw-top/pycli	<pre>Section 3.3. Ensure click, cryptography and func-timeout modules are installe how to install these modules, change directory to: >> cd /home/pi/se_mw/simw-top/pycli and run the following command: >> pip3 install -r requirements.txt pip3 install -r requirements.txt pipaseded, 0 newly installed, 0 to remove and 0 not upgraded. pipaseberrypi:~ \$ sudo apt-get install libffi-dev Reading package lists Done Building dependency tree Reading state information Done libffi-dev is already the newest version (3.2.1-9). The following packages were automatically installed and are no longer require libexiv2-14 libgfortran3 libgmime-2.6-0 libncurses5 uuid-dev Use 'sudo apt autoremove' to remove them.</pre>	ed. <u>Figure 32</u>	s do	•ne
	<pre>Section 3.3. Ensure click, cryptography and func-timeout modules are installe how to install these modules, change directory to: >> cd /home/pi/se_mw/simw-top/pycli and run the following command: >> pip3 install -r requirements.txt pip3 install -r requirements.txt pigaded, 0 newly installed, 0 to remove and 0 not upgraded. pigaspberrypi:~\$ sudo apt-get install libffi-dev Reading package lists Done Building dependency tree Reading state information Done libffi-dev is already the newest version (3.2.1-9). The following packages were automatically installed and are no longer require libexiv2-14 libgfortran3 libgmime-2.6-0 libncurses5 uuid-dev Use 'sudo apt autoremove' to remove them. 0 upgraded, 0 newly installed, 0 to remove and 0 not upgraded.</pre>	ed. <u>Figure 32</u>	s do	
oi@raspberrypi:~/se_mw/simw-top/pycli % pip3 install -r requirements.txt <mark>5</mark>	<pre>Section 3.3. Ensure click, cryptography and func-timeout modules are installe now to install these modules, change directory to: >> cd /home/pi/se_mw/simw-top/pycli and run the following command: >> pip3 install -r requirements.txt pip3 install -r requirements.txt pigraded, 0 newly installed, 0 to remove and 0 not upgraded. pi@raspberrypi:~ \$ sudo apt-get install libffi-dev Reading package lists Done Building dependency tree Reading state information Done libffi-dev is already the newest version (3.2.1-9). The following packages were automatically installed and are no longer require libexiv2-14 libgfortran3 libgmime-2.6-0 libncurses5 uuid-dev Use 'sudo apt autoremove' to remove them. 0 upgraded, 0 newly installed, 0 to remove and 0 not upgraded. pi@raspberrypi:~ \$ cd /home/pi/se_mw/simw-top/pycli 4</pre>	ed. Figure 32	<u>s</u> shc	
_ooking in indexes: https://pypi.org/simple, https://www.piwheels.org/simple	Section 3.3. Ensure click, cryptography and func-timeout modules are installent how to install these modules, change directory to: >> cd /home/pi/se_mw/simw-top/pycli and run the following command: >> pip3 install -r requirements.txt pip192.168.1.151:22 - Bitvise xterm - pi@raspberrypi ~/se_mw/simw-top/pycli 0 upgraded, 0 newly installed, 0 to remove and 0 not upgraded. pi@raspberrypi:~ \$ sudo apt-get install libffi-dev Reading package lists Done Building dependency tree Reading state information Done libffi-dev is already the newest version (3.2.1-9). The following packages were automatically installed and are no longer require libexiv2-14 libgfortran3 libgmime-2.6-0 libncurses5 uuid-dev Use 'sudo apt autoremove' to remove them. 0 upgraded, 0 newly installed, 0 to remove and 0 not upgraded. pi@raspberrypi:~ \$ cd /home/pi/se_mw/simw-top/pycli 4	ed. Figure 32	<u>s</u> shc	>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>

Requirement already satisfied: click in /usr/lib/python3/dist-packages (from -r requirements.txt (li ne 1)) (7.0) Requirement already satisfied: cryptography in /usr/lib/python3/dist-packages (from -r requirements. txt (line 2)) (2.6.1) Requirement already satisfied: func-timeout in /home/pi/.local/lib/python3.7/site-packages (from -r requirements.txt (line 3)) (4.3.5) pi@raspberrypi:~/se_mw/simw-top/pycli \$



>> sudo python3 setup.py develop

🗾 pi@192.168.1.151:22 - Bitvise xterm - pi@raspberrypi: ~/se_mw/simw-top/pycli/src	-		×
pi@raspberrypi:~/se_mw/simw-top/pycli \$ cd src			^
pi@raspberrypi:~/se_mw/simw-top/pycli/src \$ sudo python3 setup.py develop 🛑			
/usr/lib/python3.7/distutils/dist.py:274: UserWarning: Unknown distribution option:	'console	· ·	
warnings.warn(msg)			
running develop			
running egg_info			
creating ssscli.egg-info			
writing ssscli.egg-info/PKG-INFO			
writing dependency_links to ssscli.egg-info/dependency_links.txt			
writing entry points to ssscli.egg-info/entry_points.txt			
writing requirements to ssscli.egg-info/requires.txt			
writing top-level names to ssscli.egg-info/top_level.txt			
writing manifest file 'ssscli.egg-info/SOURCES.txt'			
file ssscli.py (for module ssscli) not found			
reading manifest file 'ssscli.egg-info/SOURCES.txt'			
writing manifest file 'ssscli.egg-info/SOURCES.txt'			
running build_ext			
Creating /usr/local/lib/python3.7/dist-packages/ssscli.egg-link (link to .)			
ssscli 2.14.0 is already the active version in easy-install.pth			
Installing ssscli script to /usr/local/bin			

Figure 33. Install ssscli tool

To start the ssscli tool, send the commands shown in Figure 34:

- 1. Move to the user directory:
 - >> cd /home/pi
- 2. Open the connection

>> ssscli connect se05x t1oi2c none

🗾 pi@192.168.39.198:22 - Bitvise xterm - pi@raspberrypi: ~	-	×
pi@raspberrypi:~ \$ cd /home/pi] 1 pi@raspberrypi:~ \$ ssscli connect se050 tloi2c none 2		^
pi@raspberrypi:~ \$		 ~
Figure 34. Start ssscli tool		

The SE05x ssscli tool supports several operations. To check which commands are supported by the ssscli tool (Figure 35):

>> ssscli --help

EdgeLock SE05x Quick start guide with Raspberry Pi

-						
pi@192.168.1.15	:22 - Bitvise xterm - pi@raspberrypi: ~	_		×		
pi@raspberryp	i:~ \$ sssclihelp 🛑			^		
Usage: ssscli [OPTIONS] COMMAND [ARGS]						
Command lin	e interface for SE050					
Options:						
-v,verbo	se Enables verbose mode.					
version	Show the version and exit.					
help	Show this message and exit.					
Commands:						
a/1ch	A/ICH specific commands					
cloud .	(Not Implemented) Cloud Specific utilities.					
connect	Open Session.					
decrypt	Decrypt Operation					
disconnect	Close session.					
encrypt	Encrypt Operation					
erase	Erase ECC/RSA/AES Keys or Certificate (contents)					
generate	Generate ECC/RSA Key pair					
get	Get ECC/RSA/AES Keys or certificates					
policy	Create/Dump Object Policy					
refpem	Create Reference PEM/DER files (For OpenSSL Engine).					
se05x	SE05X specific commands					
set	Set ECC/RSA/AES Keys or certificates					
sign	Sign Operation					
verify	verify Operation					
pi@raspberryp	i:~ \$			~		
Figure 35.	ssscli tool help menu					

Each of these options provides information about the syntax used for each specific command. For instance, the se05x option:

>> ssscli se05x



To read the credentials and secure objects stored in the EdgeLock SE05x, you can send the following command (Figure 37):

>> ssscli se05x readidlist

EdgeLock SE05x Quick start guide with Raspberry Pi

ni@192 168 1 151-22 - Bitvise vte	erm - ni@rasnhern/ni' ~				_	П	×
	enn præraspberrypi.~						
pi@raspberrypi:~ \$ sss	scli se05x read	idlist					^
sss :INFO :atr (Len=	=39)						
00 A0 00 00 0	03 96 04 03	E8 00 FE 02	0B 03 E8 08				
01 00 00 00 0	00 64 00 00	0E 00 69 53	45 30 35 31				
55 30 0B 01 0	00 00 00						
sss :INFO :Newer ver	rsion of Applet	Found					
sss :INFO :Compiled	for 0x30100. G	ot newer 0x4040	9				
sss :WARN :Communica	ation channel is	s Plain.					
sss :WARN :!!!Not re	commended for	production use.	111				
Key-Id: 0Xf000003 B	BINARY		Size(Bits):	3760			
Key-Id: 0Xf000001 B	BINARY		Size(Bits):	3760			
Key-Id: 0Xf000002 N	IIST-P	(Key Pair)	Size(Bits):	256			
Key-Id: 0Xf000000 N	IIST-P	(Key Pair)	Size(Bits):	256			
Key-Id: 0Xf0000012	IST-P	(Key Pair)	Size(Bits):	256			
Key-Id: 0Xf0000020 N	IIST-P	(Public Key)	Size(Bits):	256			
Key-Id: 0X7fff0204 N	IIST-P	(Public Key)	Size(Bits):	256			
Key-Id: 0X7fff0202 N	IIST-P	(Key Pair)	Size(Bits):	256			
Key-Id: 0X7fff0201 N	IIST-P	(Key Pair)	Size(Bits):	256			
Key-Id: 0X7fff0206 B	BINARY		Size(Bits):	144			
pi@raspberrypi:~ \$							~
Figure 37. ssscl	i se05x read	lidlist					

EdgeLock SE05x Quick start guide with Raspberry Pi

8 Legal information

8.1 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.

8.2 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 - without limitation - lost profits, 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' aggregate and cumulative liability towards customer for the products described herein shall be limited in accordance with the Terms and conditions of commercial sale of NXP Semiconductors.

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 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 their 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.

8.3 Trademarks

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

EdgeLock SE05x Quick start guide with Raspberry Pi

Tables

Tab. 1.	EdgeLock SE05x development boards
Tab. 2.	OM-SE050RPI adapter board details4
Tab. 3.	Raspberry Pi 4
Tab. 4.	OM-SE05xARD wiring to the Raspberry Pi
	board6
Tab. 5.	CMake Settings for SE050E product
	variants
Tab. 6.	CMake Settings for SE050F product
	variants19
Tab. 7.	CMake Settings for SE050 Previous
	Generation product variants19
Tab. 8.	CMake Settings for SE051 product variants 19
Tab. 9.	CMake Settings for A5000 product variants 20

Figures

Fig. 1. Fia. 2.	OM-SE05xARD jumper configuration OM-SE05xARD connection to the	5
	Raspberry Pi using the OM-SE05xRPI	
	adapter board	5
Fig 3	OM-SE05xARD wiring to the Raspherry Pi	0
r 19. 0.	board	6
Fig. 4.	Install build tools	7
Fig. 5.	List I2C interfaces	
Fig. 6.	Open the Raspberry Pi software	
	configuration tool	8
Fia. 7.	Enable I2C interface	8
Fia. 8.	Enable I2C interface	9
Fig. 9.	Enable I2C interface	9
Fig. 10.	Close the Raspberry Pi sofware	
U	configuration tool	10
Fig. 11.	List I2C interfaces	10
Fig. 12.	Create se050 middleware folder	11
Fig. 13.	simw-top folder content	11
Fig. 14.	Build EdgeLock SE Plug & Trust	
	Middleware middleware	12
Fig. 15.	EdgeLock SE05x middleware project	
	structure	12
Fig. 16.	Open CMake configuration interface	13
Fig. 17.	Review build configuration	14
Fig. 18.	Build project examples	15
Fig. 19.	Install projects in the system	16
Fig. 20.	Load new installed libraries	17

Tab. 10.	Static SCP03 keys	23
Tab. 11.	SCP03 session keys	23
Tab. 12.	Platform SCP key define prefix for SE050E product variants	26
Tab. 13.	Platform SCP key define prefix for SE050F product variants	26
Tab. 14.	Platform SCP key define prefix for SE050 Previous Generation product variants	26
Tab. 15.	Platform SCP key define prefix for SE051 product variants	27
Tab. 16.	Platform SCP key define prefix for A5000 product variants	27

Fig. 21. Fig. 22.	Run se05x_minimal example SE050E CMake Settings - Plain	18
•	communication	21
Fig. 23.	SPC03 mutual authentication – principle	24
Fig. 24.	SPC03 Encryption and MACing principle	24
Fig. 25.	Default Platform SCP keys are defined in	
•	the ex_sss_tp_scp03_keys.h header file	25
Fig. 26.	Select the actual Platform SCP keys in the	
	infsl_sss_ftr.h.in file	26
Fig. 27.	EdgeLock SE05xPlatform SCP plain text	
	key file	28
Fig. 28.	SE050E CMake Settings - PlatformSCP	
	enabled	29
Fig. 29.	Run se05x_minimal example with	
	platformSCP enabled - SCP keys defined	
	in the EdgeLock SE Plug & Trust	
	Middleware source	30
Fig. 30.	Run se05x_minimal example with	
	platformSCP enabled - SCP keys defined	
	in a text file	30
Fig. 31.	Install python and libffi-dev	32
Fig. 32.	Install required modules	32
Fig. 33.	Install ssscli tool	33
Fig. 34.	Start ssscli tool	33
Fig. 35.	ssscli tool help menu	34
Fig. 36.	ssscli se05x help menu	34
Fig. 37.	ssscli se05x readidlist	35

EdgeLock SE05x Quick start guide with Raspberry Pi

Contents

1	Required hardware	3
1.1	Required hardware	3
2	Prepare your Raspberry Pi	4
2.1	Hardware setup	4
2.1.1	Jumper configuration	4
2.1.2	Connecting the OM-SE05xARD to the	
	Raspberry Pi	5
2.1.2.1	Using the OM-SE05xRPI adapter board	5
2.1.2.2	Connecting the OM-SE05xARD with wires	5
2.2	Software setup	6
2.2.1	Install Raspbian	6
2.2.2	Install build tools	7
2.2.3	Enable the I2C interface	7
3	Run EdgeLock SE Plug & Trust	
	Middleware test examples	10
3.1	Download EdgeLock SE Plug & Trust	
	Middleware	10
3.2	Build EdgeLock SE Plug & Trust	
	Middleware	11
3.3	Build EdgeLock SE Plug & Trust	
	Middleware examples	12
3.4	Execute EdgeLock SE Plug & Trust	
	Middleware test example	17
4	Product specific CMake build settings	. 18
4.1	Example: SE050E CMake build settings	. 20
5	Binding EdgeLock SE05x to a host MCU/	
	MPU using Platform SCP	. 22
5.1	Introduction to the Global Platform Secure	
	Channel Protocol 03 (SCP03)	. 22
5.2	How to configure the product specific	
	default Platform SCP keys	. 24
5.2.1	Defining the deault Platfrom SCP keys in	
	the EdgeLock SE Plug & Trust Middleware	
	source code	25
5.2.2	Defining the default Platfrom SCP keys in a	
	text file	27
5.3	How to enable Platform SCP in the CMake-	
	based build system	28
6	Manage access from multiple Linux	
	processes to the EdgeLock SE05x	31
7	Appendix A: Using the ssscli tool	31
8	Legal information	36

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: 3 August 2022 Document identifier: AN12570 Document number: 565815