

PSA Crypto API Lab

This lab procedure walks through the steps to generate a wrapped key in the Secure Vault High device and use the PSA Crypto API to perform the ECDSA digital signature.

KEY POINTS

- Secure key storage in the Secure Vault High device
- ECDSA sign and verify

1 Prerequisites

For this lab you will need the following:

- EFR32xG21B 2.4G Hz 10 dBm Radio Board (BRD4181C)
- Simplicity Studio v5 with Gecko SDK Suite v3.2.0
- Terminal program to receive UART communication form the WSTK

1.1 Update the SE firmware

Click the **Update to <version>** in **Update to <version>** | **Changelog** link (if it appears) to update the **Secure FW** (version **1.2.8 or above**).

🛐 v5_workspace - Simplicity Studio™		
File Edit Navigate Search Project Run Window Help		
♠ Welcome 🕙 Recent 🎬 Tools 📩 Install 🌞 Preferences		
📑 Debug Adapters 🛛 🗞 🗳 🖉 🗶 🛠 🗘 🔻 🔲 📄 🕀 🕀 🗖		
 ✓ EFR32xG21B 2.4 GHz 10 dBm RB (ID:440030580) ✓ EFR32xG21B 2.4 GHz 10 dBm Radio Board (BRD4181C) 	EFR32xG21B 2.4 GHz 10 df	3m RB, WSTK
EFR32MG21B010F1024IM32 Wireless Starter Kit Mainboard (BRD4001A Rev A01)	OVERVIEW EXAMPLE PROJECTS & DEMOS	DOCUMENTATION
	General Information Connected Via: J-Link Silicon Labs Debug Mode: Onboard Device (MCU) Adapter FW: 1v4p6b1171 Secure FW: 1.2.6 	 Configure ✓ Change Latest e to 1.2.8) Changelog

1.2 Erase the flash

Run the *security erasedevice* command in Simplicity Commander to erase the main flash. This procedure is to make sure there is no security key in NVM3.

commander security erasedevice

1.3 Import the Project to Simplicity IDE

- 1. Go to Launcher Perspective of Simplicity Studio, click the Tools icon to select Simplicity IDE
- 2. Click **OK** to launch the Simplicity IDE

🇱 Tools 🛃 Install 🎄 Preferences	
Si Tools Dialog	×
Simplicity Commander	^
Simplicity IDE	
Device Console	
Network Analyzer	
Fixed Function Configuration	
	~
Add/Remove Too	ols
OK Cancel	

3. Import the *psa_crypto_ecdsa* project from the folder containing the *psa_crypto_ecdsa.sls* (in psa_crypto_api_lab.zip) file

Si Import Project					×
Project Search					
Select a location to scar	n for a project				
Select a project to import	ha ar				
C:\Users\amleung\Dov	wnloads		~	Brows	e
Detected projects:		Droject Turo			
Project Name		Project Type	(de)		_
psa_crypto_ecusa		Simplicity Studie) (.515)		
More Import Options					
? < Back	Next >	Finish		Cance	:I

4. Click Next, Next, and Finish to generate the project

File	1	dit	Navigate	e Searc	h Proj	ect f	Run	Window	Help									
*	•		- 🛛 🕼	1 ·	∛ -	<u>ب</u>	嗣 •	- \$ \$	-	- 🖻 💷	A Welco	me 🔅	9 Recent	Tools	🛃 Install	* Preference	es	
b 1	Pro	ject	Explorer	×												E	🔁 🍸 🖇	
~ 1	ß	psa	_crypto_ec	dsa [GN	U ARM	v10.2.	.1 - 0	Debug] [El	R32M0	G21B010F	1024IM32 -	Gecko	SDK Sui	te: Flex 3.2	.0.0, Micriu	im OS Kernel,	Platform (v3.2.0)]
	>	Ы	Includes															
	>	0	autogen															
	>	6	config															
	>	6	gecko_sdk_	3.2.0														
	>	c	app_init.c															
	>	.h	app_init.h															
	>	c	app_proces	SS.C														
	>	h	app_proces	ss.h														
	>	c	main.c															
		٢	imported_p	project_r	eport.h	tml												
		٥	psa_crypto	_ecdsa.p	intool													
			psa_crypto	_ecdsa.sl	ср													
			psa_crypto	_ecdsa.s	ps													

5. Open the *app_process.c* file to start the lab

2 Coding

The figure below is the expected output on the terminal program.

6 0	J-Link Silicon Labs (440030580) 🛛						
9	No translation	 Line terminator 	CR-LF (DOS, OS/2, MS Windows)	\sim	Buffer size:	100000	Serial 1, 0 bytes out, 626 bytes in.
2	🛿 Serial 0 🏾 🚔 Serial 1 🚔 Admin 🚔 Debug						
	. PSA Crypto API lab + PSA Crypto initialization OK + Checking if there is already a persist + Setting up the key attribute to create + Creating a SEC9265R1 (256-bit) PERSIST + Signing a hash with a SECP256R1 (256-b + Verifying the signature of a hash with	tent key with the c a SECP256R1 pers: TENT WRAPPED key. bit) PERSISTENT WR h a SECP256R1 (256-	given identifier Does not ex istent wrapped key . OK APPED private key OK -bit) public key OK	ist			

Hints:

- The slides in the PSA Crypto.pptx
- The PSA Crypto API document https://docs.silabs.com/mbed-tls/latest/
- The PSA Crypto macros are defined in the crypto_values.h (C:\SiliconLabs\SimplicityStudio\v5\developer\sdks\gecko_sdk_suite\v3.2\util\third_party\crypto\mbedtls\include\psa)
- The source code of the PSA Crypto ECDSA platform example (C:\SiliconLabs\SimplicityStudio\v5\developer\sdks\gecko_sdk_suite\v3.2\app\common\example\psa_crypto_ecdsa)

2.1 PSA Crypto Initialization

Write the code for PSA Crypto initialization in the app_process.c.

```
// Initialize the PSA Crypto
printf(" + PSA Crypto initialization... ");
// Write your code here
```

2.2 Set Up the Key Attributes

Write the code to set up the key attributes in the app_process.c.

```
// Set up the key attribute to create a SECP256R1 persistent wrapped key, key ID is
PERSISTENT_KEY_ID
printf(" + Setting up the key attribute to create a SECP256R1 persistent wrapped key\n");
```

// Write your code here

2.3 Generate a SECP256R1 Persistent Wrapped Key

Write the code to generate a SECP256R1 persistent wrapped key in the app_process.c.

```
// Generate a SECP256R1 persistent wrapped key
printf(" + Creating a SECP256R1 (256-bit) PERSISTENT WRAPPED key... ");
```

// Write your code here

2.4 Sign a Hash with the SECP256R1 Private Key

Write the code to sign a hash in the *app_process.c*.

```
// Sign a hash with the SECP256R1 private key
printf(" + Signing a hash with a SECP256R1 (256-bit) PERSISTENT WRAPPED private key... ");
// Write your code here
```

2.5 Verify the Signature with the SECP256R1 Public Key

Write the code to verify the signature in the *app_process.c*.

```
// Verify the signature with the SECP256R1 public key
printf(" + Verifying the signature of a hash with a SECP256R1 (256-bit) public key.. ");
// Write your code here
```

2.6 Reference solution

The reference solution can be found in the app_process_solution.c (in psa_crypto_api_lab.zip) file.

3 Testing

3.1 Build and Download the Code to the BRD4181C Radio Board

🎋 🕶 🖻 🕶 📓 🖷 🖷 🐨 🕌 🕶	<u>ل</u> ا ج	- 🖓 👻 😓 -	• 🗘 •	📑 💷 🏦 Welco	ome 🕑 R	ece					
Project Explorer 🖾 🛛 🛛 Build	'GNI	J ARM v10.2.1 -	Debug'	for project 'psa_cn	pto ecds	a' L					
✓ ⅔ psa_crypto_ecdsa [GNU ARM	v10.2	2.1 - Debug] [El	FR32		-	-					
> 🔊 Includes											
> 🗁 autogen											
> 🗁 config											
> 🧁 gecko_sdk_3.2.0											
> app_init.c											
> b app init.h											
app_men											
> h app process h											
> a main c											
imported project report h	tml										
nsa crunto ecdea nintool	um										
sa_crypto_ecusa.pintoor											
E psa_crypto_ecusa.sips											
Project Explorer		E 🕏 7 🕴									
✓ I psa_crypto_ecdsa [GNI]	J ARN	1 v10.2.1 - Debug]	[EFR32								
🗸 🗱 Binaries											
> 🕸 psa_crypto_ecds	a.axf -	[arm/le]									
> D psa_crypto_ecds	a.bin -	[unknown/le]									
> O psa_crypto_ecds	a.hex	- [unknown/le]									
> O psa_crypto_ecds	a.c27 .	Now	1								
> D Includes		New		· · · · · · · · · · · · · · · · · · ·	-						
> 🔁 autogen		Open									
> Config		Show In		Alt+Shift+W>							
> GNU ARM v10.2.1 -	D	Open With		>							
> app init.c		Сору		Ctrl+C							
> 🖻 app_init.h		Paste		Ctrl+V							
> <a>le app_process.c	^	Delete		Delete							
> 🖻 app_process.h		Rename		F2							
> 🖻 main.c		Inconst									
imported_project_r	er	Import		,							
psa_crypto_ecdsa.p		Build Project									
sa_crypto_ecusa.si		Refresh		F5	-						
i psa_crypto_ecusa.si	0	Run As		>							
	₹₽.	Debug As		>							
		Compare With		>							
		Replace With		,							
				,							
		Browse Files Her	re Lling Llos	-							
	28 4 18	Elash to Device	Line Her	e							
		Dramarting		Alter Free							
		Properties		Alt+Enter	1						

3.2 First Run

There is no key in the device, the persistent wrapped key is generated and stored in the flash through the NVM3 driver.

🔗 J-Link Silicon Labs (440030580) 🛛									
So translation	 Line terminator: CR-LF (DOS, OS/2, MS Windows) 	Buffer size: 100000 Serial 1, 0 bytes out, 626 bytes in.							
🚡 Serial 0 🚔 Serial 1 🚔 Admin 🚔 Debug									
. PSA Crypto API lab + PSA Crypto initialization OK									
+ Checking if there is already a persistent key with the given identifier Does not exist + Setting up the key attribute to create a SECP256R1 persistent wrapped key + Creating a SECP256R1 (256-bit) PERSISTENT WRAPPED key OK									
+ Signing a hash with a SECP255R1 (256-bit) PERSISTENT WRAPPED private key OK + Verifying the signature of a hash with a SECP256R1 (256-bit) public key OK									

3.3 Consecutive Run

Move the **Power Source Select** switch to the **USB** position to power off the radio board, then back to the **AEM** position to power on the radio board to re-run the program.

The persistent wrapped key is retrieved from the flash for ECDSA operations.

0	🔐 J-Link Silicon Labs (440030580) 💠										
6	No translation	\sim	Line terminator:	CR-LF	(DOS, OS/2, MS Windows)	~	Buffer size:	100000	Serial 1, 0 bytes out, 1441 bytes in.		
e	🖹 Serial 0 🚔 Serial 1 🏯 Admin 🚔 Debug										
	. PSA Crypto API lab + PSA Crypto initialization OK										
ΠL	+ Checking if there is already a persistent key with the given identifier Already exists										
Ш	+ Signing a mash with a SECP256R1 (256-bit) PERSISIENT WRAFFED private key OK + Verifying the signature of a hash with a SECP256R1 (256-bit) public key OK										

3.4 Destroy the Key

There are two ways to destroy the persistent wrapped key in the flash.

2. Simplicity Commander

• Erase the NVM3 area (default size is 40 kB) to destroy the key.

commander device pageerase --range 0xf4000:0xfe000
Erasing range 0x000f4000 - 0x000fe000
DONE