Revisions

|  |  |  |
| --- | --- | --- |
| **Rev** | **Date** | **Description** |
| 1.0 | 2018/09/13 | Initial reversion |
| 2.0 | 2018/10/31 | 1. Modify the parameters of some interfaces, add passphrase and passphrase\_len to the interface parameters to enhance security 2. Add join request interface for applying for Intel® EPID certificate online |
|  |  |  |
|  |  |  |
|  |  |  |

CATALOGUE

[coconut SDK manual for mobile client 3](#_Toc528853796)

[1. Functions provided by the SDK 3](#_Toc528853797)

[2 SDK component release 3](#_Toc528853798)

[3 SDK integration method 4](#_Toc528853799)

[4 SDK API reference 5](#_Toc528853800)

[4.1 LoadPrebuiltFiles method 5](#_Toc528853801)

[4.2 GenerateJoinReq method 6](#_Toc528853802)

[4.3 GeneratePrvkey method 8](#_Toc528853803)

[4.4 MakeAllSigs method 10](#_Toc528853804)

[4.5 SignTransaction method 12](#_Toc528853805)

[APPENDIX 14](#_Toc528853806)

[1. Error code list 14](#_Toc528853807)

[2. naming convention for the signature materials 15](#_Toc528853808)

[3. A simple example for using coconut SDK in iOS 15](#_Toc528853809)

[**Table 1 coconut SDK build&release details** 3](#_Toc528853810)

[**Table 2 naming convention details** 15](#_Toc528853811)

[**figure 1 SDK interfaces structure for android** 4](#_Toc528853815)

[**figure 2 SDK interfaces structure for iOS** 5](#_Toc528853816)

# coconut SDK manual for mobile client

## Functions provided by the SDK

Coconut SDK provide the related functions of the Intel® EPID certificate application and use, including generating join request for applying for EPID certificate online, generating EPID member private key locally, making all signatures responding to every basename in the basename list and signing message specified by the certificate holder. Additionally, the SDK provide the interface to load test data, such as group public key, member private key, basename list, so that the SDK interfaces can be tested very conveniently.

## SDK component release

● For android

The SDK is internally implemented in C language, and the external interfaces are written in java native interface. the SDK component is released in the form of a dynamic link library(.so binary file) conforming to the JNI standard. After the android project loads so file, java can directly call the SDK interface of JNI form.

● For iOS

The internal implementation of the SDK is written in C language, the external interface is written in swift (bridged Objective-C), and the SDK component is released in the form of static swift framework of cocoa Touch Framework. After the framework is imported into the iOS project, swift can directly call the SDK interface.

**Table 1 coconut SDK build&release details**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **OS** | **architecture** | **file type of SDK** | **devices supported** | **file name of SDK** | **language (internal)** | **language (external)** | **compiler** |
| android | armv7/arm64  x86/x86\_64 | .so | iphone | libCoconutMcSDK.so | C | JNI C++ | NDK r17 tools chain |
| iOS | armv7/arm64 | .framework | android5.0 or later | CoconutMcSDK.framework | C | Swift  (bridging OC) | C/OC：Apple LLVM 9.0  swift：swift4.1 |

● For iOS system, corresponding to a so file in each architecture, currently supports four kinds of architecture, divided into four so files, because there are very few android devices with mips architecture, currently do not support mips/mips64 architecture.

● For iOS system, support armv7/arm64 architecture, released in the form of fat binaries, The frameworks with different architecture are packaged together to provide only one .framework file, only provide the generic iOS device release version, no simulator version.

## SDK integration method

● For android

In the java package of the android app, create the java package and the class with the same JNI interface as the java package and the class name, and then declare the native method of the JNI interface prototype released in the class; after the package and the class are created successfully, load the so file corresponds to the current architecture, and then these interfaces can be called in the java code according to the JNI interface prototype of the SDK.

● For iOS

Import the swift .framework released by the SDK into the iOS app project, and then import the framework in your swift code, after that, you can use swift to call these interfaces according to the external interface prototype released by the SDK.

coconut

|

|

mcsdk

| 🡪java package: coconut.mcsdk

|

Signer 🡪java public class: Signer

| 🡪JNI native interfaces

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ |\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

| | | | |

| | | | |

LoadPrebuiltFiles GenerateJoinReq GeneratePrvkey MakeAllSigs SignTransaction

**figure 1 SDK interfaces structure for android**

CoconutMcSDK

| 🡪swift cocoa touch framework, static library

|

CoconutMcSDK-Brigding-Header.h

| 🡪Objective-C bridge

|

Signer 🡪swift cocoa touch class

| 🡪swift interfaces

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_|\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

| | | | |

| | | | |

LoadPrebuiltFiles GenerateJoinReq GeneratePrvkey MakeAllSigs SignTransaction

**figure 2 SDK interfaces structure for iOS**

## SDK API reference

### 4.1 LoadPrebuiltFiles method

The interface is a interface only for test. In order to facilitate the integration test of the SDK, the necessary signature materials, such as the EPID group public key file, the EPID member private key file, and the basename list file are also loaded. In addition, some real-time data is also loaded for the convenience of testing. These data files, including the membership credential file used to generate the EPID member private key, and the transaction data to be signed.

★ All of these pre-loaded files are only used for testing the SDK interface in the initial

stage of integration development. In actual use scenarios, real data generated during the process of applying for an EPID certificate and signing a transaction by the client user must be used.

**Syntax**

|  |
| --- |
| int LoadPrebuiltFiles(  String res\_directory\_path  ) |

**Parameters**

|  |
| --- |
| res\_directory\_path |

the path to store the pre-load files, usually stored in the private directory of the current app.

**Return Value**

If executed successfully, it returns 0 (EPIDA\_OK), otherwise it returns the corresponding error code.

**Remarks**

This interface is only a test interface. It is used only when there is no real data in the initial development stage. The release version in production environment should not use this interface.

**Requirements(android)**

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

Target Platform android

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Minimum supported system android 5.0

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

DLL libCoconutMcSDK.so

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Caller Language java

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

packet/class coconut.mcsdk.Signer

**Requirements(iOS)**

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

Target Platform iOS/iPhone

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Minimum supported system iOS9

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

framework CoconutMcSDK.framework

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Caller Language swift 4.1

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

framework/class CoconutMcSDK/Signer

### 4.2 GenerateJoinReq method

This interface is used to generate a request for online application for EPID certificate. The generated request data combined with other authenticate data is sent to the certificate issuer, and finally membership credential that constitutes the private key is obtained.

**Syntax**

|  |
| --- |
| int GenerateJoinReq(  String res\_directory\_path,  String nonce\_file\_fullname,  String passphrase,  int passphrase\_len,  String joinreq\_file\_fullname  ) |

**Parameters**

|  |
| --- |
| res\_directory\_path |

the file path for storing the signature materials (for example, public key, encrypted private key, basename list, etc.)

|  |
| --- |
| nonce\_file\_fullname |

the file path of the certificate issuer nonce is stored. The nonce data is obtained from the issuer during the process of initiating the online application certificate request, and then stored in the local file, and the full path of the file is saved when the interface is invoked.

|  |
| --- |
| passphrase |

the signature passphrase set when the user applies for a certificate online. The passphrase is used as part of the data to generate an encryption key to encrypt and save privacy data generated during the interface is invoked.

|  |
| --- |
| passphrase\_len |

the length of signature passphrase

|  |
| --- |
| joinreq\_file\_fullname |

the full path(file path + file name) of the file stored the join request generated by invoking the interface

**Return Value**

If executed successfully, it returns 0 (EPIDA\_OK), otherwise it returns the corresponding error code.

**Remarks**

How to obtain the issuer nonce and how to use the join request generated by the interface, please refer to /coconut/certificationRequestStep1 in the <<Coconut RestAPI>> document.

**Requirements(android)**

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

Target Platform android

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Minimum supported system android 5.0

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

DLL libCoconutMcSDK.so

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Caller Language java

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

packet/class coconut.mcsdk.Signer

**Requirements(iOS)**

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

Target Platform iOS/iPhone

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Minimum supported system iOS9

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

framework CoconutMcSDK.framework

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Caller Language swift 4.1

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

framework/class CoconutMcSDK/Signer

### 4.3 GeneratePrvkey method

The interface is used to generate a EPID member private key locally on the mobile client. Part of the data for generating the private key comes from the certificate issuer when applying for the certificate, and another secret part randomly generated by the mobile client. It not only guarantees the authority of certificate issuance, but also ensures that the final generated private key is only known to the owner of the private key, ie the mobile client user.

★ The member private key generated locally by the mobile client has been encrypted and saved by SDK.

**Syntax**

|  |
| --- |
| int GeneratePrvkey(  String res\_directory\_path,  String credential\_file\_fullname,  String passphrase,  String passphrase\_len  ) |

**Parameters**

|  |
| --- |
| res\_directory\_path |

the file path for storing the signature materials (for example, public key, encrypted private key, basename list, etc.)

|  |
| --- |
| credential\_file\_fullname |

the full path(including the path and file name of the file) of the file that stored the membership credential. The file can be deleted after the member private key was generated. The membership credential is obtained from the certificate issuer during applying for certificate. For the specific acquisition interface, see /coconut/certificationRequestStep2 in the <Coconut RestAPI> document.

|  |
| --- |
| passphrase |

the signature passphrase set when the user applies for a certificate online. The passphrase is used as part of the data to generate an encryption key to encrypt and store the member private key generated by invoking the interface.

|  |
| --- |
| passphrase\_len |

the length of signature passphrase

**Return Value**

If executed successfully, it returns 0 (EPIDA\_OK), otherwise it returns the corresponding error code.

**Remarks**

The caller of the interface is responsible for specifying the path saved the member private key.

After the SDK generates the private key, it will be saved to the path with the agreed file name. For the agreed file name of the private key file, refer to the section on signature material file naming conventions in the appendix of this document.

**Requirements(android)**

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

Target Platform android

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Minimum supported system android 5.0

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

DLL libCoconutMcSDK.so

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Caller Language java

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

packet/class coconut.mcsdk.Signer

**Requirements(iOS)**

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

Target Platform iOS/iPhone

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Minimum supported system iOS9

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

framework CoconutMcSDK.framework

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Caller Language swift 4.1

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

framework/class CoconutMcSDK/Signer

### 4.4 MakeAllSigs method

After the user applies for an EPID certificate and generates a private key successfully, the interface generates all signatures for every basename in the basename list according to the signature materials (group public key, member private key, and basename list) . All the signatures are saved in a file whose path specified by the interface caller.

**Syntax**

|  |
| --- |
| int MakeAllSigs(  String res\_directory\_path,  String passphrase,  int passphrase\_len,  String all\_sigs\_file\_fullname,  ) |

**Parameters**

|  |
| --- |
| res\_directory\_path |

the file path for storing the signature materials (for example, public key, encrypted private key, basename list, etc.)

|  |
| --- |
| passphrase |

the signature passphrase set when the user applies for EPID certificate online. The passphrase is used to generate the decrypted key which decrypts the decrypted member private key.

If the passphrase is inconsistent with the signature passphrase set when the user applies for the certificate, generating signature will fail.

|  |
| --- |
| passphrase\_len |

the length of signature passphrase

|  |
| --- |
| all\_sigs\_file\_fullname |

the full path (file path + file name) of the file saved all signatures. Each basename in the basename list generates a signature, and all signatures are saved in this file.

**Return Value**

If executed successfully, it returns 0 (EPIDA\_OK), otherwise it returns the corresponding error code.

**Remarks**

The following is the file format for saving all signatures. In actual use, all individual signatures may need to be extracted from the file according to the file format.

@@SigCount:10@@@@SigNo:1@@XXXXXXXXXXXXXXXXXXXX@@SigNo:2@@XXXXXXXXXXXXXXXXXXXXXXXXXX@@SigNo:3@@..........................@@SigNo:10@@XXXXXXXXXXXXXX

The content in @@SigCount:10@@ indicates the total count of signatures, @@SigNo:1@@ indicates the signature sequence, and the content immediately following is the signature content. The current signature content continues until the next signature sequence identifier starts or the file ends.

**Requirements(android)**

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

Target Platform android

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Minimum supported system android 5.0

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

DLL libCoconutMcSDK.so

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Caller Language java

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

packet/class coconut.mcsdk.Signer

**Requirements(iOS)**

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

Target Platform iOS/iPhone

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Minimum supported system iOS9

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

framework CoconutMc.framework

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Caller Language swift 4.1

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

framework/class CoconutMcSDK/Signer

### 4.5 SignTransaction method

Sign a message, the signature is stored in the file specified by the interface caller.

**Syntax**

|  |
| --- |
| int SignTransaction(  String res\_directory\_path,  String passphrase,  int passphrase\_len,  String transaction\_file\_fullname,  String signature\_file\_fullname  ) |

**Parameters**

|  |
| --- |
| res\_directory\_path |

the path of files required for signing a message, such as public key, member group key, etc.

|  |
| --- |
| passphrase |

the signature passphrase set when the user applies for EPID certificate online. The passphrase is used to generate the decrypted key which decrypts the decrypted member private key.

If the passphrase is inconsistent with the signature passphrase set when the user applies for the certificate, generating signature will fail.

|  |
| --- |
| passphrase\_len |

the length of signature passphrase

|  |
| --- |
| transaction\_file\_fullname |

the full path of file saved the message to be signed

|  |
| --- |
| signature\_file\_fullname |

the full path of file stored the signature

**Return Value**

If executed successfully, it returns 0 (EPIDA\_OK), otherwise it returns the corresponding error code.

**Remarks**

The file format for saving the signature is as follows. The fields can be extracted from the file according the file format, if the individual field is required.

@@Basename:XXXXX@@groupID@@XXXXXXXXXXXXXXXXXXXXXXXXXXX

The XXXXX content in @@Basename:XXXXX@@ indicates the basename used for the signature, and the XXXXX in @@groupID:XXXXX@@ indicates the group ID of public key, and the following is the signature content until the end of the file.

**Requirements(android)**

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

Target Platform android

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Minimum supported system android 5.0

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

DLL libCoconutMcSDK.so

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Caller Language java

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

packet/class coconut.mcsdk.Signer

**Requirements(iOS)**

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

Target Platform iOS/iPhone

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Minimum supported system iOS9

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

framework CoconutMcSDK.framework

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Caller Language swift 4.1

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

framework/class CoconutMcSDK/Signer

## APPENDIX

### Error code list

|  |
| --- |
| typedef enum enCoconutSDKErrCode  {  EPIDA\_ERR = -1,  EPIDA\_OK = 0,  EPIDA\_INVALID\_PARAMETERS = 1,  EPIDA\_NO\_RES\_PATH = 2,  EPIDA\_INVALID\_RES\_PATH\_LEN = 3,  EPIDA\_NO\_CACER\_FILE = 4,  EPIDA\_INVALID\_CACER = 5,  EPIDA\_NO\_PUBKEY\_FILE = 6,  EPIDA\_INVALID\_PUBKEY = 7,  EPIDA\_NO\_PRIVKEY\_FILE = 8,  EPIDA\_INVALID\_PRIVKEY = 9,  EPIDA\_NO\_BSN\_LIST\_FILE = 10,  EPIDA\_INVALID\_BSN\_LIST = 11,  EPIDA\_SINMSG\_FAIL = 12,  EPIDA\_READ\_FILE\_FAIL = 13,  EPIDA\_WRITE\_FILE\_FAIL = 14,  EPIDA\_NO\_BSN\_SIG\_FILE\_PATH = 15,  EPIDA\_NO\_TRANS\_FILE\_PATH = 16,  EPIDA\_NO\_TRANS\_FILE = 17,  EPIDA\_NO\_TRANSC\_SIG\_PATH = 18,  EPIDA\_NO\_CREDENTIAL\_FILE\_PATH = 19,  EPIDA\_NO\_CREDENTIAL\_FILE = 20,  EPIDA\_INVALID\_CREDENTIAL = 21,  EPIDA\_NO\_MEMBERKEY\_FILE\_PATH = 22,  EPIDA\_NO\_MEMBERKEY\_FILE = 23,  EPIDA\_GEN\_MEMBERKEY\_FAIL = 24,  EPIDA\_NO\_SIG\_FILE\_PATH = 25,  EPIDA\_NO\_SIG\_FILE = 26,  EPIDA\_NO\_MSG\_FILE\_PATH = 27,  EPIDA\_NO\_MSG\_FILE = 28,  EPIDA\_NO\_VERIFY\_BASENAME = 29,  EPIDA\_SET\_VERIFIER\_BSN\_FAIL = 30,  EPIDA\_SET\_VERIFIER\_CTX\_FAIL = 31,  EPIDA\_VERIFY\_SIG\_FAIL = 32,  EPIDA\_NO\_NI\_FILE = 33,  EPIDA\_NO\_RANDOM\_FILE = 34,  EPIDA\_NO\_PRIVATEF\_FILE = 35,  EPIDA\_INVALID\_PRIVATEF = 36,  EPIDA\_NO\_JOINREQ\_FILE\_PATH = 37,  EPIDA\_MAKE\_JOINREQ\_FAIL = 38,  EPIDA\_NO\_PASSPHRASE = 39,  EPIDA\_ENCRYPT\_FAIL = 40,  EPIDA\_DECRYPT\_FAIL = 41,  }COCONUT\_ERR\_CODE\_E; |

### naming convention for the signature materials

The SDK uses some fixed file names internally. The caller must use the agreed file name if the file created by the caller.

**Table 2 naming convention details**

|  |  |  |
| --- | --- | --- |
| filename | file format | content |
| epid\_cacert.bin | binary | CA certificate |
| epid\_pubkey.bin | binary | EPID group public key |
| epid\_mprivkey.dat | binary | EPID member private key |
| epid\_basenames.dat | JSON | basename list |

### A simple example for using coconut SDK in iOS

|  |
| --- |
| 1. /////////////////////////////////////////////////////////////////////////////////  2. a simple example for using coconut SDK in iOS  3. ////////////////////////////////////////////////////////////////////////////////   1. import UIKit 2. import CoconutMcSDK 3. import Security 4. class ViewController: UIViewController { 5. override func viewDidLoad() { 6. super.viewDidLoad() 7. // Do any additional setup after loading the view, typically from a nib. 9. let DocumentPath = NSSearchPathForDirectoriesInDomains( 10. FileManager.SearchPathDirectory.documentDirectory, 11. FileManager.SearchPathDomainMask.userDomainMask, true) 12. print("sandbox:document path:", DocumentPath) 14. let applicationPath = NSSearchPathForDirectoriesInDomains( 15. FileManager.SearchPathDirectory.applicationDirectory, 16. FileManager.SearchPathDomainMask.userDomainMask, true) 17. print("sandbox:application path:", applicationPath) 18. } 20. //load the signature materials for signing message as test data，only a interface for testing. 21. @IBAction func LoadPreBuiltFiles(\_ sender: Any) { 22. let DocumentPath = NSSearchPathForDirectoriesInDomains( 23. FileManager.SearchPathDirectory.documentDirectory, 24. FileManager.SearchPathDomainMask.userDomainMask, true) 25. if (0 == Signer.LoadPrebuiltFiles(res\_directory\_path: DocumentPath[0])) 26. { 27. print("Load prebuilt files successfully!") 28. makeAllSigsOut.text = "Load prebuilt files successfully!" 29. } 30. else 31. { 32. print("Failed to load prebuilt files>>>>") 33. makeAllSigsOut.text = "Failed to load prebuilt files>>>" 34. makeAllSigsOut.adjustsFontSizeToFitWidth = true; 35. } 36. } 38. //generate the join request 39. @IBAction func GenJoinReq(\_ sender: Any) { 40. let DocumentPath = NSSearchPathForDirectoriesInDomains( 41. FileManager.SearchPathDirectory.documentDirectory, 42. FileManager.SearchPathDomainMask.userDomainMask, true) 44. let ns\_nonce\_filename = "/epid\_nonce.dat" 45. let ns\_nonce\_file\_fullname = "\(DocumentPath[0])\(ns\_nonce\_filename)" 46. let ns\_joinreq\_file\_filename = "/epid\_joinReq.dat" 47. let ns\_joinreq\_file\_fullname = "\(DocumentPath[0])\(ns\_joinreq\_file\_filename)" 48. let ns\_passphrase = "myPassword" 50. if (0 == Signer.GenerateJoinReq(res\_directory\_path: DocumentPath[0], 51. nonce\_file\_fullname: ns\_nonce\_file\_fullname, 52. passphrase: ns\_passphrase, passphrase\_len: 10, 53. joinreq\_file\_fullname: ns\_joinreq\_file\_fullname)) 54. { 55. print("Generate join request successfully!") 56. makeAllSigsOut.text = "Generate join request successfully!" 57. } 58. else 59. { 60. print("Failed to generate join request>>>>") 61. makeAllSigsOut.text = "Failed to generate join request>>>" 62. makeAllSigsOut.adjustsFontSizeToFitWidth = true; 63. } 64. } 66. //generate the member private key 67. @IBAction func GenPrvKey(\_ sender: Any) { 68. let DocumentPath = NSSearchPathForDirectoriesInDomains( 69. FileManager.SearchPathDirectory.documentDirectory, 70. FileManager.SearchPathDomainMask.userDomainMask, 71. true) 73. let credential\_filename = "/epid\_credential.dat" 74. let credential\_fullname = "\(DocumentPath[0])\(credential\_filename)" 75. let ns\_passphrase = "myPassword" 77. if (0 == Signer.GeneratePrvKey(res\_directory\_path:DocumentPath[0], 78. credential\_file\_fullname:credential\_fullname, 79. passphrase:ns\_passphrase, passphrase\_len: 10)) 80. { 81. print("Generate private key successfully!") 82. makeAllSigsOut.text = "Generate private key successfully!" 83. } 84. else 85. { 86. print("Failed to generate private key>>>>") 87. makeAllSigsOut.text = "Failed to generate private key>>>" 88. makeAllSigsOut.adjustsFontSizeToFitWidth = true; 89. } 90. } 92. //Generate an signature for each basename and save it to the specified file in a certain file format. 93. 98. //All signature materials must be prepared before calling the interface, 94. 99. // otherwise generating a signature will fail. 95. @IBAction func sigmsg(\_ sender: Any) { 96. let DocumentPath = NSSearchPathForDirectoriesInDomains( 97. FileManager.SearchPathDirectory.documentDirectory, 98. FileManager.SearchPathDomainMask.userDomainMask, true) 100. let ns\_passphrase = "myPassword" 101. let sigs\_dat\_filename = "/epida\_sigs.dat" 102. let sigs\_fullname = "\(DocumentPath[0])\(sigs\_dat\_filename)" 104. if (0 == Signer.MakeAllSigs(res\_directory\_path: DocumentPath[0], 105. passphrase: ns\_passphrase, passphrase\_len: 10, 106. all\_sigs\_file\_fullname: sigs\_fullname)) 107. { 108. print("Make all sigs successfully!") 109. makeAllSigsOut.text = "Make all sigs successfully!" 110. } 111. else 112. { 113. print("Failed to make all sigs>>>>") 114. makeAllSigsOut.text = "Failed to make all sigs>>>" 115. makeAllSigsOut.adjustsFontSizeToFitWidth = true; 116. } 117. } 119. //Sign a message 120. @IBAction func SignTransaction(\_ sender: Any) { 121. let documents\_path = NSSearchPathForDirectoriesInDomains( 122. FileManager.SearchPathDirectory.documentDirectory, 123. FileManager.SearchPathDomainMask.userDomainMask, true) 125. let ns\_passphrase = "myPassword" 126. let trans\_file\_name = "/transaction.dat" 127. let trans\_dat\_fullname = "\(documents\_path[0])\(trans\_file\_name)" 129. let trans\_sig\_file\_name = "/trans\_sig.dat" 130. let trans\_sig\_fullname = "\(documents\_path[0])\(trans\_sig\_file\_name)" 132. if (0 == Signer.SignTransaction(res\_directory\_path: documents\_path[0], 133. passphrase: ns\_passphrase, passphrase\_len: 10, 134. transaction\_file\_fullname: trans\_dat\_fullname, 135. signature\_file\_fullname: trans\_sig\_fullname)) 136. { 137. print("sign the transaction successfully!") 138. makeAllSigsOut.text = "sign the transaction successfully!" 139. makeAllSigsOut.isHighlighted = true 140. } 141. else 142. { 143. print("Failed to sign the transaction>>>>") 144. makeAllSigsOut.text = "Failed to sign the transaction>>>>" 145. makeAllSigsOut.isHighlighted = true 146. } 147. } 149. @IBOutlet weak var makeAllSigsOut: UILabel! 151. override func didReceiveMemoryWarning() { 152. super.didReceiveMemoryWarning() 153. // Dispose of any resources that can be recreated. 154. } 155. } |