FCS-SDK-DOC

Prepared by

Ford Motor Company

27-Oct.-2020

Version 1.0.0

Contributor

Team EESE

|  |  |  |  |
| --- | --- | --- | --- |
| **REVISION HISTORY** | | | |
| Version | Date | Name | Comment |
| 1.0 | 27-Oct-2020 | Hongda(hzhu67)/YaoJian | Initial version |
| 1.1 | 11-Aug-2021 | Zhangjia | Add verifyCtl Interface |
| 1.2 | 01-Sep-2021 | ZhangJia/WuJia | Add TEE encrypt Interface  Add TEE decrypt Interface  Add TEE saveFile Interface  Add syncp Encode Interface  Add syncp decode Interface |

|  |  |  |  |
| --- | --- | --- | --- |
| **ABBREVIATE TABLE** | |  |  |
| IVI | In-Vehicle Infotainment | |  |
|  |  | |  |

目录

[1. Introduction 2](#_Toc1434683308)

[2. Defination 3](#_Toc1189632284)

[2.1 FCS 3](#_Toc217863261)

[2.2 FCS-SDK 3](#_Toc390418762)

[3. Permission Control 4](#_Toc1685868360)

[4. API Specifications 5](#_Toc797674251)

[4.1 static ClientManager getInstance(Context context) 5](#_Toc1128091171)

[4.2 void init(FcsConfig config, ConnectorCallback connectorCallback) 5](#_Toc1457334454)

[4.3 void encryptByCloudKey(String content, int keyType, final Callback callback) 5](#_Toc1143919002)

[4.4 void encryptByNewCloudKey(String content, int keyType, final Callback callback) 6](#_Toc1654002783)

[4.5 void decrpByCloudKey(String content, int keyType, final Callback callback) 6](#_Toc422477608)

[4.6 void retrieveCommKeyHash(final Callback callback) 7](#_Toc1019007119)

[4.7 void retrievePlatformId(final Callback callback) 7](#_Toc80751948)

[4.8 void retrieveDeviceId(final Callback callback) 7](#_Toc1114895768)

[4.9 requestIVT 8](#_Toc1200231828)

[4.10 getCtlLog 8](#_Toc1905367772)

[4.11 getCtlSig 8](#_Toc1357918155)

[4.12 encryptByLocalKey 8](#_Toc925469019)

[4.13 decrpByLocalKey 8](#_Toc1890811957)

[4.14 void verifyCtl(String url, Callback callback) 8](#_Toc280678300)

[4.15 synchronized SyncpResult syncpEncode(SyncpEncodeParam param) 8](#_Toc936189664)

[4.16 synchronized SyncpResult syncpDecode(byte[] buffer, int size) 9](#_Toc309046609)

[4.17 void v2iEncryptASync(SecurityRequestBean encryptParm, Callback callback) 9](#_Toc150452243)

[4.18 void v2iDecryptASync(SecurityRequestBean decryptParm, Callback callback) 9](#_Toc1077972048)

[4.19 byte[] v2iEncryptSync(SecurityRequestBean encryptParm) 10](#_Toc2005180177)

[4.20 byte[] v2iDecryptSync(SecurityRequestBean decryptParm) 10](#_Toc405957634)

[4.21 boolean saveFile(byte[] fileSource, String fileAlia) 10](#_Toc1572270717)

[5. Codes 11](#_Toc1357221928)

[5.1 KeyTypeCode-加解密所用的key类型 11](#_Toc50470867)

[5.2 ResultCode-回调返回的结果状态 11](#_Toc1949339324)

[5.3 HTTP Status Code 13](#_Toc1322113655)

1. Introduction

This document describes the interfaces exposed by FCS-SDK for FCS connection.

1. Defination

This document explains the process of application connecting and call function with FCS in IVI Service Layer.

* 1. FCS

In phase4 products , IVISL is named FCS (Ford Connection Service)

* 1. FCS-SDK

FCS-SDK is a Android app connect FCS by aild connection, which exposed API for other apps in IVI device.

1. Permission Control

FCS 服务对客户端的调用进行了***权限控制***，Client App必须使用Desay 统一提供的App 签名文件给app签名。

1. API Specifications

This section documents the APIs which are hosted by FCS-SDK

* 1. static ClientManager getInstance(Context context)

|  |  |  |
| --- | --- | --- |
| **Field** | **Description** | |
| 功能 | 获取 FCS ClientManager实例。 | |
| 参数 | context | Application context |
| 返回 | ClientManager | FCS ClientManager实例 |

* 1. void init(FcsConfig config, ConnectorCallback connectorCallback)

|  |  |  |
| --- | --- | --- |
| **Field** | **Description** | |
| 功能 | 初始化ClientManager, 连接FCS远程service。 | |
| 参数 | config | 配置信息参数： String appID：为调用方从Ford云端申请到的唯一ID标识符。  boolean debugFlag：调试日志开关。  \* int interval：初始化连接未成功的重试时间间隔。  \* int times：初始化连接未成功的重试次数 |
|  | connectorCallback | 用于接收结果的回调，  连接成功回调方法为connected()。  连接失败断开的回调方法为disconnected()。 |

* 1. void encryptByCloudKey(String content, int keyType, final Callback callback)

|  |  |  |
| --- | --- | --- |
| **Field** | **Description** | |
| 功能 | 使用指定的Key来加密指定的内容，并使用对应的Callback来接收结果 | |
| 参数 | content | 待加密的明文消息 |
| keyType | 指定需要哪个key来对内容进行加密，数字码与key的对照关系请见Key Type Code章节。 |
| callback | 用于接收结果的回调，  成功回调方法为onSuccess(int status, in byte[] result)。  result为json格式数据，cipherText为密文。格式为  {"cipherText": "", "deviceId": "" }。  失败的回调方法为onFail(int status, String msg)。 |

* 1. void encryptByNewCloudKey(String content, int keyType, final Callback callback)

|  |  |  |
| --- | --- | --- |
| **Field** | **Description** | |
| 功能 | 请求在IVISL APK里执行指定Key的Key Rotation操作，然后使用云端新生成的Key来加密指定的内容，最后使用IEncrpSvcCallback返回结果 | |
| 参数 | content | 参数 |
|  | keyType | 指定需要哪个key来对内容进行解密，数字码与key的对照关系请见Key Type Code章节。 |

* 1. void decrpByCloudKey(String content, int keyType, final Callback callback)

|  |  |  |
| --- | --- | --- |
| **Field** | **Description** | |
| 功能 | 请求TEE 使用指定Key的来对提供的内容进行解密。 | |
| 参数 | msg | 网络侧返回的需要解密的内容 |
|  | keyType | 指定需要哪个key来对内容进行解密，数字码与key的对照关系请见Key Type Code章节。 |
|  | callback | 用于接收结果的回调，  成功回调方法为onSuccess(int status, in byte[] result)。  result为json格式数据，plainText为明文。格式为  { "plainText": "xx"}。  失败的回调方法为onFail(int status, String msg)。 |

* 1. void retrieveCommKeyHash(final Callback callback)

|  |  |  |
| --- | --- | --- |
| **Field** | **Description** | |
| 功能 | 通过Tee获取communication key 的hash。 | |
| 参数 | callback | 用于接收结果的回调，  成功回调方法为onSuccess(int status, in byte[] result)。result即为communication key 的hash值。  失败的回调方法为onFail(int status, String msg)。 |

* 1. void retrievePlatformId(final Callback callback)

|  |  |  |
| --- | --- | --- |
| **Field** | **Description** | |
| 功能 | 获取用于云端通信的platformId。 | |
| 参数 | callback | 用于接收结果的回调，  成功回调方法为onSuccess(int status, in byte[] result)。result为platformId  。  失败的回调方法为onFail(int status, String msg)。 |
|  |  |

* 1. void retrieveDeviceId(final Callback callback)

|  |  |  |
| --- | --- | --- |
| **Field** | **Description** | |
| 功能 | 获取设备deviceId。 | |
| 参数 | callback | 用于接收结果的回调，  成功回调方法为onSuccess(int status, in byte[] result)。result为deviceId。  失败的回调方法为onFail(int status, String msg)。 |

* 1. requestIVT

void requestIVT(in IEncrpSvcCallback callback ,in String info)

* 1. getCtlLog

void getCtlLog(in IEncrpSvcCallback callback)

* 1. getCtlSig

void getCtlSig(in IEncrpSvcCallback callback)

* 1. encryptByLocalKey

byte[] encryptByLocalKey(in byte[] content, int keyType)

* 1. decrpByLocalKey

byte[] decrpByLocalKey(in byte[] content, int keyType)

* 1. void verifyCtl(String url, Callback callback)

|  |  |  |
| --- | --- | --- |
| **Field** | **Description** | |
| 功能 | 校验url CTL。 | |
| 参数 | url | 待校验url。 |
| callback | 用于接收结果的回调，  成功回调方法为onSuccess(int status, in byte[] result)。result为”success”的。  失败的回调方法为onFail(int status, String msg)。 |

* 1. SyncpReult encode(SyncpEncodeParam param)

|  |  |  |
| --- | --- | --- |
| **Field** | **Description** | |
| 功能 | 使用SyncP Lib加密 | |
| 参数 | param | 需要加密的内容 |
| 返回值 | SyncpResult | Int statusCode：0 成功，-1失败  byte[] result：加密成功后返回的内容  int size：加密成功后内容的长度 |

* 1. SyncpResult decode(byte[] buffer, int size)

|  |  |  |
| --- | --- | --- |
| **Field** | **Description** | |
| 功能 | 使用SyncP Lib解密 | |
| 参数 | buffer | 需要解密的内容 |
| size | 解密内容的长度 |
| 返回值 | SyncpResult | Int statusCode：0 成功，-1失败  byte[] result：解密成功后返回的内容  int size：解密成功后内容的长度 |

* 1. void teeEncryptASync(SecurityRequestBean encryptParm, Callback callback)

|  |  |  |  |
| --- | --- | --- | --- |
| **Field** | **Description** | |  |
| 功能 | 使用TEE加密（结果异步返回） | | |
| 参数 | encryptParam | byte[] mDataSource | 明文 |
| byte[] mIVector | iv向量 |
| int mEncryptType | 加密类型（0 AES256 ECB；  1 AES256 CBC） |
| String mKeyAlia | key名称 |
| callback | | 加密操作结果(Int statusCode:1 成功，-1失败;  byte[] result：密文) |

* 1. void teeDecryptASync(SecurityRequestBean decryptParm, Callback callback)

|  |  |  |  |
| --- | --- | --- | --- |
| **Field** | **Description** | |  |
| 功能 | 使用TEE解密（结果异步返回） | | |
| 参数 | decryptParam | byte[] mDataSource | 密文 |
| byte[] mIVector | iv向量 |
| int mEncryptType | 加密类型（0 AES256 ECB；  1 AES256 CBC） |
| String mKeyAlia | key名称 |
| callback | | 解密操作结果(Int statusCode:1 成功，-1失败;  byte[] result：明文) |

* 1. byte[] teeEncryptSync(SecurityRequestBean encryptParm)

|  |  |  |  |
| --- | --- | --- | --- |
| **Field** | **Description** | |  |
| 功能 | 使用TEE 加密（结果同步返回） | | |
| 参数 | decryptParam | byte[] mDataSource | 明文 |
| byte[] mIVector | iv向量 |
| int mEncryptType | 加密类型（0 AES256 ECB；  1 AES256 CBC） |
| String mKeyAlia | key名称 |
| 返回值 | byte[] | | 密文 |

* 1. byte[] teeDecryptSync(SecurityRequestBean decryptParm)

|  |  |  |  |
| --- | --- | --- | --- |
| **Field** | **Description** | |  |
| 功能 | 使用TEE解密（结果同步返回） | | |
| 参数 | decryptParam | byte[] mDataSource | 密文 |
| byte[] mIVector | iv向量 |
| int mEncryptType | 加密类型（0 AES256 ECB；  1 AES256 CBC） |
| String mKeyAlia | key名称 |
| 返回值 | byte[] | | 明文 |

* 1. boolean teeSaveFile(byte[] fileSource, String fileAlia)

|  |  |  |
| --- | --- | --- |
| **Field** | **Description** | |
| 功能 | 向TEE存储文件或加密key 。 | |
| 参数 | fileSource | 文件字节数组 |
| fileAlia | 文件存储后名称，加解密时传递此参数 |

1. Codes
   1. KeyTypeCode-加解密所用的key类型

|  |  |
| --- | --- |
| **Code** | **Description** |
| **0** | Communication Key |
| **1** | Local Pinning Key (required by EA) |
| **2** | Local TEE AES key (required by Account) |

* 1. ResultCode-回调返回的结果状态

|  |  |
| --- | --- |
| **Code** | **Description** |
| **0** | 成功 |
| **-1** | IVI SL inner error |
| **-100** | TEE 解析onBoardingKey失败 |
| **-101** | TEE储存Communication Key失败 |
| **-102** | TEE储存V2I的AES KEY失败 |
| **-103** | TEE无法使用Coummunication Key解密 |
| **-104** | TEE中丢失对应的Key |
| **-105** | TEE异常 |
| **-200** | Certificate Pinning获取指纹信息失败 |
| **-201** | Certificate Pinning证书校验失败 |
| **-202** | 从云端获取V2I的 AES KEY失败 |
| **-203** | 从云端获取V2I correlation\_id失败 |
| **-204** | 云端执行Key Rotation失败 |
| **-205** | 云端不支持当前vehicle |
| **-206** | 云端数据格式错误 |
| **-207** | 网络服务错误 |
| **-208** | 云端数据解码错误 |
| -209 | 云端服务器http错误 |
| **-300** | AppId错误/不支持 |
| **-301** | 不存在指定的Key Type |
| **-308** | Client 传参解码错误 |
| **-400** | 无法正确获取当前汽车的2码(Vin, ESN) |
| **-501** | AES加解密错误 |
| **-502** | SHA256异常 |
| **-600** | 数据库异常 |
| **-700** | IVT 云端接口发生异常 |

服务器接口范围的业务错误码会透传给App，最新的具体status code请查阅相关接口文档：IVI Service Layer Interface Document – Auth.docx 和IVI Service Layer Interface Document - V2I.docx

Auth api status codes:

|  |  |
| --- | --- |
| **Code** | **Description** |
| **130200** | Request is processed successfully |
| **130400** | Invalid parameter(parameter name list which are invalid) |
| **130401** | Platform Id authentication fail |
| **130402** | Onboarding key not found |
| **130403** | App Id authentication fail |
| **130404** | invalid deviceId |
| **130405** | illegal transaction id. |
| **130406** | sign check fail. |
| **130407** | Request time is invalid. |
| **130500** | Service internal error |
| **130501** | Backend service error |
| **130502** | No data found |
| **130701** | Encryption key is invalid |
| **130702** | Communication key is expired |
| **130703** | Device hasn't been registered. |
| **130704** | Communication key not found |
| **130705** | Device has been registered. |
| **130801** | Device authentication fail |

* 1. HTTP Status Code

云端网络侧返回的代码如下，可从实际情况下考虑是否需要转交给App侧。

|  |  |
| --- | --- |
| **Code** | **Description** |
| **200** | Success (Response body should contain an application status code) |
| **400** | Bad request(Response body should contain an application status code) |
| **401** | Unauthorized – if the Authorization token is invalid or has expired |
| **403** | Forbidden |
| **404** | Unknown API or the API is off |
| **500** | Internal server error |
| **502** | Unexpected error from backend system |
| **503** | The server is currently unable to handle the request due to a temporary overloading or maintenance of the server. |
| **504** | Backend system did not respond within time limit |