福建联迪商用设备有限公司

(BC07□□□)

MPOS 读卡器方案
IOS SDK 开发指南

版　　本：V2.5

发行控制：

2014 年 11 月 27 日 编写　　2014 年 11 月 27 日 实施

编写部门：新兴应用软件研发部　　编写: 方湖东

文件修订履历表

文件修订履历表	文　件名称	MPOS 读卡器方案 IOSSDK 开发指南
	文　件编号	

修订日期	版次	修订记录	修订人	审核
2013-07-06	1.0	版本创建	黄智强	
2014-11-27	2.0	版本更新	方湖东	
2015-03-26	2.1	增加工程设置的说明-all_load	方湖东	
2015-05-11	2.2	增加 3.1 节的 API 的调用流程详细说明 增加请求用户示卡片时时对单音频刷卡头的过程消息回调	方湖东	
2015-06-15	2.3	增加音频类无密码键盘的明文 pin 加密接口	方湖东	
2015-08-21	2.4	增加接口对特定机器配置的约束，如只支持 M15 或者只支持 M36 增加打印机状态的返回类型 增加了手写签名接口	方湖东	
2015-09-22	2.5	增加 M1 卡操作接口 增加打印点阵接口 增加打印 BMP 接口	方湖东	

目录

MPOS 读卡器方案
IOSSDK 接口使用说明

第 1 章 简介

1.1 编写目的

方便开发人员快速使用 SDK，开发 MPOS 下的应用

1.2 SDK 的限制

目前蓝牙通信，SDK 仅支持 iphone 4S 及以上的机型，系统推荐 IOS 6.0 以上的版本，音频支持 iphone 4 及以上机型。

第 2 章 SDK 使用快速入门

2.1 权限

如果走蓝牙通信要保证蓝牙开关已经正常打开，音频通信需要保证应用已经取得麦克风的使用权限。

2.2 导入静态库到工程中

SDK 由 3 个文件组成：

2 个头文件：LandiMPOS.h 定义了所有 SDK 的接口，LDCommon.h 定义了 SDK 使用到枚举、结构类及相当宏。

1 个静态库 libMPOSReader.a

使用前请先导入这 3 个文件，并在工程的 Build Settings 中找到 Other Link Flags 加入 -all_load -lstdc++

2.3 快速入门

LandiMPOS 控制器为 SDK 的基础类，设计为单例模式，应用只要调用[LandiMPOS getInstance]获取该类的实例就可以直接使用所有接口，如

[[LandiMPOS getInstance] startSearchDev:1000

```
searchOneDeviceBlcok:^(LDC_DEVICEBASEINFO *deviceInfo) {
        NSLog(@"发现一台设备，名称为%@",deviceInfo.deviceName);
    } completeBlock:^(NSMutableArray *deviceArray) {
        NSLog(@"搜索设备结束");
    }];
```

2.4 销毁资源

为了更好的用户体验我们的建议是当一次连接上设备后就不再断开设备连接，保持设备通信的长连接，但是在 App 崩溃或退出后记得要关闭设备并销毁相当资源，调用接口为:

```
- (void) closeDevice;
```

第 3 章 SDK 接口说明

3.1 SDK 调用流程指南

3.1.1 设备连接

调用 getInstance 获取 SDK 的实例
调用 startSearchDev 开始搜索设备，通过 searchOneDeviceCB 回调方法，SDK 会通知 app 当前搜索到的设备信息，通过 searchCompleteCB 方法，SDK 通知 app 搜索设备结束
调用 openDevice 方法建立与设备的通信连接，对于蓝牙设备，只要传入设备的 ID 及通道，对于音频设备，直接传入设备的通道即可，ID 可放空，接口成功后，就可以和 MPOS 交互了。

3.1.2 初始化

调用 getDeviceInfo 方法获取终端信息，如果支持 TMS 固件更新的，需要检查一下终端的固件版本号，有更新的话做终端固件远程更新。
根据终端的 SN 号检查绑定信息，及密钥信息，对于 app 有支持远程密钥更新的，请再调用 querryMKeyInfo 获取终端主密钥状态，无主密钥的请自动发起主密钥更新流程，不支持主密钥已经用更新的请略过。
APP 根据 SN 号向后台发起签到，获取到磁道，PIN 及 MAC 密钥后，调用 SDK 的 loadKey 接口下装工作密钥
对于有支持 IC 卡参数下载的 app，请再检查一下后台参数是否更新，有更新的话请调用 AddAid 接口增加 AID 参数，调用 AddPuk 接口增加公钥参数，对于预置参数的请略过。
初始化成功。

3.1.3 终端固件远程更新

仅适用于蓝牙连接的产品如 M35、M36，音频类产品速度过慢暂时不支持如 M15
检查到终端程序有更新，请先将终端程序下载到本地
调用 enterFirmwareUpdateMode 接口启动固件更新模式，成功后调用 updateFirmware 接口传入固件地址，开始更新固件

3.1.4 发起交易

用户在 app 上做完交易金额输入后，开始调用 SDK 完成金融交易
请先调用 waitingCard 接口请求用户刷卡或者插卡、挥卡（要读取什么卡种 APP 可以自由控制），用户如果在超时时间内操作到卡片，该接口会回调告知 APP 当前操作的卡片类型，APP 再根据卡片类型决定以后的交易步骤。

3.1.5 磁条卡交易流程

请调用 getPAN 接口获取磁卡卡号，调用 getTrackData 接口得到磁卡的磁道信息（明文还是密文及加密方式要向联迪确认），这两个接口的顺序不要相反，不然会得不到卡号，调用 inputPin 接口获取磁条卡的密码，打包交易报文，完成后调用 calculateMac 方法（MAC 若是 ECB 算法要向联迪说明）计算交易报文的 MAC，送后台，一笔交易完成

3.1.6 IC 卡交易流程

调用 startPBOC 接口开始一笔 IC 卡交易，传入交易的类型，金额及交易日期（YYMMDD）及时间（HHMMSS），成功后会返回 IC 卡卡号，IC 卡等效 2 磁道数据及序列号等，然后再调用 continuePBOC 接口根据 PBOC 执行结果码，如果是请求联机请将该接口的返回值 PIN 及 dol（即 55 域数据）打包，完成后调用 calculateMac 计算一下报文 MAC，送到后台处理，后台返回的新的 55 域数据及 39 域返回码，请调用 onlineDataProcess 接口送入终端处理，处理成功后再调用 PBOCStop 接口完成 IC 卡交易结果，如果是交易成功，请将 onlineDataProcess 返回的 dol 数据（即 TC）上送到后台保存，无论交易成功与否如果 dol 中包含 DF31 标签，要将此数据做脚本上送处理，如果出现，返回的 39 域为成功，而执行 onlineDataProcess 返回的结果码不为交易批准，请发冲正，这样子才算完成一笔 IC 卡交易

3.1.7 QPBOC 快速支付流程

调用 QpbocPurchase 接口直接开始一笔非接触快速支付，回调返回接口会告知 APP 这笔交易扣款是否成功
调用 QpbocGetRecordNums 得到当前终端内保存了多少笔非接的流水
调用 QpbocGetOneRecord 接口可以把流水的内容一条一条的全取出来展示
调用 QpbocDelOneRecord 接口可以删除设备的流水，只支持从头往后，或者 从尾往头

删除

正常的操作步骤是 QpbocPurchase 交易成功后，调用 QpbocGetRecordNums 接口得到当前终端内有多少笔未上送的脱机交易，调用 QpbocGetOneRecord 读取第一条脱机交易的内容，打包上送到服务器，服务器返回成功后，调用 QpbocDelOneRecord 接口删除第一条流水，再调用 读取第一条脱机交易的内容，再上送，成功后，再删除，直接到所有的流水全部送成功。

3.2 SDK 通用 Block 定义

```
*      @brief 指令执行失败调用的block，所有的失败回调的都是这种类型，包含错误码及原因
typedef void (^onErrorCB)(NSString* errCode,NSString* errInfo)

*      @brief 无参数的回调，一般是执行成功但无任何数据返回的接口
typedef void (^onVoidCB)();

*      @brief 返回NSString值的成功返回，只返回单一数据类型的接口，如给卡片上电返回的ATR
typedef void (^onNSStringCB)(NSString* stringCB)

*      @brief 返回NSData值的成功返回，如获取设备时间
typedef void (^onNSDataCB)(NSData* dateCB)
```

3.3 SDK 基本操作接口

3.3.1 基础接口

3.3.1.1 获取 LandiMPOS 控制器实例

```
*获取LandiMpos的实例，所有的接口都是在这个类实现的
+ (id) getInstance;
```

3.3.1.2 销毁 LandiMPOS 控制器

```
~~+ (id) closeResources;~~//已经废弃，请使用
- (void) closeDevice;
```

3.3.1.3 获取 SDK 当前版本号

```
- (NSString *) getLibVersion;
```

3.3.1.4 SDK 日志控制接口

打开或者关闭库的日志输出

```
- (void) switchLog:(LDE_LOGSTATE)logState;
```

3.3.2 通信接口

3.3.2.1 搜索设备

```
 *    @param      timeout   搜索的超时时间，毫秒为单位
 *    @param      searchOneDeviceCB    搜索到一个设备回调
 *    @param      searchCompleteCB     搜索设备完成回调
- (void) startSearchDev:(NSInteger)timeout
    searchOneDeviceBlcok:(onSearchOneDeviceCB)searchOneDeviceCB
completeBlock:(onSearchCompleteCB)searchCompleteCB;
```

搜索到一个设备的回调定义如下：

```
typedef void (^onSearchOneDeviceCB)(LDC_DEVICEBASEINFO *deviceInfo)
```

LDC_DEVICEBASEINFO 类提供接口读取搜索到设备的名称，ID 及通信通道

搜索设备完成的回调，注意此处 deviceArray 永远返回 nil。

```
typedef void (^onSearchCompleteCB)(NSMutableArray *deviceArray)
```

3.3.2.2 终止搜索设备

```
- (void) stopSearchDev;
```

3.3.2.3 判断当前设备连接状态

```
 Ture－ 已经连接      false－ 未连接设备
- (BOOL)isConnectToDevice;
```

3.3.2.4 打开设备

```
*    @param      identifier      设备Id，搜索设备的时候会返回
 *    @param      channel   设备的通讯通道，搜索设备的时候会返回
```

```
 *    @param      mode     通讯的模式，建议采用主从模式
- (void) openDevice:(NSString *)identifier channel:(LDE_CHANNEL)channel
mode:(LDE_COMMUNICATIONMODE)mode
          successBlock:(onVoidCB)successCB failedBlock:(onErrorCB)failedCB
```

3.3.2.5 关闭设备

```
- (void) closeDevice;
```

3.3.2.6 取消当前操作

取消终端的长等待操作，如请求用户出示卡片，请求用户输入密码，请求用户确认卡号之类的长等待操作，一般指令执行速度很快，取消不会有什么效果。

注意：取消操作设备为通知消息，终端是否要终止当前操作由终端自己判断，该接口回调成功不一定取消成功，要根据实际指令的返回为取消成功的依据

```
- (void) cancelCMD:(onVoidCB)successCB failedBlock:(onErrorCB)failedCB
```

3.4 MPOS 操作接口

3.4.1 终端固件远程升级接口

仅适用于蓝牙连接的产品如 M35 或 M36，M15 不支持

3.4.1.1 启动固件升级模式

```
 -          (void)          enterFirmwareUpdateMode:(onVoidCB)successCB
failedBlock:(onErrorCB)failedCB
```

3.4.1.2 开始升级固件

App 需要先将固件下载到本地再启动升级

```
*    @param      filePath   固件存放的本地路径
- (void) updateFirmware:(NSString*)filePath
              completeBlock:(onVoidCB)downloadCompleteCB
              progressBlock:(onDownloadProcessCB)downloadProgressCB
                 errorBlock:(onDownloadErrCB)downloadErrorCB;
 *    @param          current: 当前流程值
 *    @param          total: 总共的流程值
```

current/ total 即为更新百分比
typedef void (^onDownloadProcessCB)(unsigned int current,unsigned int total)

3.4.2 基础接口

3.4.2.1 获取终端信息

- (void) getDeviceInfo:(onGetDeviceCB)successCB failedBlock:(onErrorCB)failedCB;
回调接口为：
typedef void (^onGetDeviceCB)(LDC_DeviceInfo* deviceInfo)
LDC_DeviceInfo 中包含了大部分的设备基本信息，列举几个比较重要：

```
*    @param    productSN        联迪SN，联迪后标牌8位序列号
*    @param    custormerSN      客户定制SN，客户定制的序列号，位数可变
*    @param    bootSoftVer      设备的BOOT版本
*    @param    ctrlSoftVer      设备的CTRL版本
*    @param    userSoftVer      设备的固件版本号，用做固件升级判断的依据
*    @param    hardwareVer      硬件版本
*    @param    hardwareSN       硬件32字节完整序列号
*    @param    powerLevel       电量级别
```

3.4.2.2 获取终端 RTC 时间

仅支持 M35 或者 M36 ，M15 无 RTC 不支持
- (void) getDateTime:(onNSDataCB)successCB failedBlock:(onErrorCB)failedCB;

3.4.2.3 修改终端 RTC 时间

暂时不提供修改时间的接口

3.4.2.4 屏幕文字显示

仅支持 M35 或者 M36 有 ，M15 无屏幕不支持

```
*    @param    text       需要显示的文本
*    @param    row        显示行号
*    @param    col        显示列号
*    @param    timeout    显示停留时间
*   @param     clearflag      显示前是否先清屏
```

- (void) displayLines:(NSString*)text Row:(NSUInteger)row Col:(NSUInteger)col Timeout:(int)timeout ClearScreen:(LDE_CLEARFLAG)clearflag

successBlock:(onVoidCB)successCB failedBlock:(onErrorCB)failedCB

3.4.2.5 获取终端基本参数

- (void) getTerminalParam:(onLDC_TerminalBaseParaCB)successCB
failedBlock:(onErrorCB)failedCB
回调的参数定义如下：
typedef void (^onLDC_TerminalBaseParaCB)(LDC_TerminalBasePara* terminalPara)

LDC_TerminalBasePara 终端基本参数包括：
```
 *    @param    merchantNO    商户号
 *    @param    merchantName  商户名称
 *    @param    terminalNO    终端号
 *    @param    serialNO      流水号
 *    @param    bathcNO       批次号
```

3.4.2.6 设置终端基本参数

-(void) setTerminalParam:(LDC_TerminalBasePara *)terminalPara
successBlock:(onVoidCB)successCB failedBlock:(onErrorCB)failedCB

3.4.2.7 启动手写签名

仅适用于带电子签名板的终端，终端返回的数据为经过 JBIG 压缩后的数据，解压后为一张 240 * 80 的二值图片，还原图片可以借助联迪提供的 JBIG 库。

```
/**
 *    @brief 启动手写签名
 *
 *    @param    specialCode 特征码
 *    @param    timeOut 超时
 *    @param    successCB    成功回调
 *    @param    failedCB     失败回调
 *
 *    @returnvoid
 */
```

- (void) startUserSign:(NSString *)specialCode timeOut:(Byte)timeOut
successBlock:(onNSStringCB)successCB failedBlock:(onErrorCB)failedCB;

3.4.3 MKSK 密钥相关接口

3.4.3.1 查询终端主密钥状态

- (void) querryMKeyInfo:(onQueryMKeyCB)successCB failedBlock:(onErrorCB)failedCB
成功回调定义为：
typedef void (^onQueryMKeyCB)(LDC_MasterKey* keyData);
LDC_MasterKey 的返回参数为
* @param hasMasterKey 是否已加载主密钥，用来做主密钥远程下载的判断
* @param masterKeyIndex 主密钥索引号，废弃无用

3.4.3.2 导入工作密钥

该接口支持导入 4 种类型的密钥，Mkey，TDK，PIK 及 MAK
* @param keyData 导入的密钥的数据
- (void) loadKey:(LFC_LoadKey *)keyData
successBlock:(onVoidCB)successCB failedBlock:(onErrorCB)failedCB

LFC_LoadKey 类有两个参数，一个是密钥类型，一个为密钥值
@property (nonatomic) LDE_KEYTYPE keyType;
@property (nonatomic, copy) NSString* keyData;
LDE_KEYTYPE 是一个枚举包括如下 4 种类型
* @param KEYTYPE_TRACK 磁道加密密钥
* @param KEYTYPE_PIN PIN加密密钥
* @param KEYTYPE_MAC MAC计算密钥
* @param KEYTYPE_MKEY 主密钥

keyData 密钥值为定长格式，以 HEXString 形式 40 位表示 20 个字节，前 32 位（16 字节）为密钥的数据，后 8 位（4 字节）为密钥的 KCV 校验值
如果密钥为 16 位（8 字节）的 DES 密钥，请置 keyData 的第 17-32 位为 0，KCV 不变。
KCV 校验值为密钥的明文，对 8 个字节的 0x00 做 DES/3DES 加密取结果的前 4 字节

3.4.3.3 计算报文 MAC 值

- (void) calculateMac:(NSString*)macData
successBlock:(onNSDataCB)successCB failedBlock:(onErrorCB)failedCB;
macData 以 HexString 的形式传入如@"1122334455667788990 0"代表计算
0x11 0x22 0x33 0x44 0x 55 0x66 0x77 0x88 0x99 0x00, 10 个字节的数据 MAC 值

3.4.4 交易相关接口

3.4.4.1 请求用户出示卡片

这是交易需要调用的第一个接口，当判断到卡片类型后，再由 APP 控制交易逻辑,如果有操作到单音频刷卡卡的 app，请调用第二个接口，有过程消息返回。

3.4.4.1.1 无过程消息回调接口

```
/**
 *    @brief 等待刷、插卡
 *
 *   @param      text          交易的标题行，一般显示交易类型，如“查询”，“消费”
 *    @param     timeout   等待出示卡片的超时时间，秒为单位，一般设置为60秒以内
 *    @param     cardType      希望读取的卡片类型，3种类型可以自由组合
 *    @param     moneyNum    消费金额，为0时不显示金额
 *
 *    @returnvoid
 */
- (void) waitingCard:(NSString *)text timeOut:(int)timeout
CheckCardTp:(LDE_SUPPORTCARDTYPE)cardType moneyNum:(NSString*)moneyNum
          successBlock:(onLDE_CardTypeCB)successCB failedBlock:(onErrorCB)failedCB
```

LDE_SUPPORTCARDTYPE 为枚举判断包括有 5 个值，代表此次寻卡希望读取的卡片界面类型，如，可只请求读取磁卡，这个时候插卡和挥卡是不会有任何反应的，如果同时支持磁道及 IC 卡，终端将会判断磁道的服务码，如果判断磁卡服务码标明此卡片有芯片，将会强制要求用户插 IC 卡交易。

```
SUPPORTCARDTYPE_MAG          = 0x01,
SUPPORTCARDTYPE_IC           = 0x02,
SUPPORTCARDTYPE_RF           = 0x04,
SUPPORTCARDTYPE_MAG_IC
SUPPORTCARDTYPE_MAG_IC_RF
```

当用户刷卡或者插卡后将回调

```
typedef void (^onLDE_CardTypeCB)(LDE_CardType cardtype);
CARDTYPE_MAGNETIC    = 0x01,//用户刷了磁卡
CARDTYPE_ICC         = 0x02,//用户插入IC卡
CARDTYPE_RF          = 0x04,//用户挥 RF 卡
```

3.4.4.1.2 有过程消息回调接口

该接口针对单音频刷卡（M15）头设计返回过程消息，如刷卡错误，请重刷，检查到卡片有芯片，请插入 IC 卡交易等，请通过过程消息来判断。

```
- (void) waitingCard:(NSString *)text timeOut:(int)timeout
CheckCardTp:(LDE_SUPPORTCARDTYPE)cardType moneyNum:(NSString*)moneyNum
          successBlock:(onLDE_CardTypeCB)successCB
```

```
progressMsg:(onNSStringCB)msg failedBlock:(onErrorCB)failedCB;
```

3.4.4.2 获取卡号

获取当前操作的卡片的卡号支持磁卡及 IC 卡，支持非接卡片

```
*      @param      isCipher      是否加密，已经废弃
- (void) getPAN:(LDE_PANDATATYPE)isCipher successCB:(onNSStringCB)successCB
failedBlock:(onErrorCB)failedCB;
```

3.4.4.3 获取磁道信息

注意，获取完磁道信息，终端将会清除缓存的磁道数据，此后不可再读取卡号，所以一般流程，要先获取完卡号再取磁道信息，否则会报错。

```
*      @brief 获取磁道信息
- (void) getTrackData:(LDE_TRACKTYPE)isEncrypt
successCB:(onQueryTrackCB)successCB failedBlock:(onErrorCB)failedCB;
LDE_TRACKTYPE 为枚举代表要读取的磁道数据格式，有明文和密文两种方式
 *   @param      TRACKTYPE_PLAIN      明文
 *   @param      TRACKTYPE_ENCRYPT    密文
```

3.4.4.4 获取交易密码

仅针对有键盘有屏幕的产品如 M35、M36，不支持 M15

```
*      @param      inputPIN      输入密码的所需信息
- (void) inputPin:(LFC_GETPIN*)inputPIN
        successBlock:(onNSDataCB)successCB failedBlock:(onErrorCB)failedCB;
@property (nonatomic, copy) NSString* panBlock;//交易的卡号，卡号会参与密码的计算
@property (nonatomic, copy) NSString* moneyNum;//交易的金额
@property (nonatomic) OneByeInt timeout;     //读取密码的超时时间
```

3.4.4.5 加密明文密码

仅对音频类无密码键盘的设备有用如 M15，有密码键盘的设备不支持（如 M35、M36 不支持）

```
- (void) encClearPIN:(NSString *)clearPin withPan:(NSString *)pan
successBlock:(onNSStringCB)successCB failedBlock:(onErrorCB)failedCB
@property (nonatomic, copy) NSString* pan;//交易的卡号，卡号会参与密码的计算
@property (nonatomic, copy) NSString* clearPin;//用户输入的明文pin
```

3.4.5 PBOC 接口

3.4.5.1 PBOC 参数管理接口

3.4.5.1.1 导入 IC 卡参数

- (void) AddAid:(NSString *)aid successBlock:(onVoidCB)successCB failedBlock:(onErrorCB)failedCB

3.4.5.1.2 清除 IC 卡参数列表

- (void) clearAids:(onVoidCB)successCB failedBlock:(onErrorCB)failedCB

3.4.5.1.3 导入 IC 卡公钥

- (void) addPubKey:(NSString *)pubKey successBlock:(onVoidCB)successCB failedBlock:(onErrorCB)failedCB

3.4.5.1.4 清除 IC 卡公钥列表

- (void) clearPubKey:(onVoidCB)successCB failedBlock:(onErrorCB)failedCB

3.4.5.2 PBOC 交易接口

3.4.5.2.1 PBOC 开始交易

- (void) startPBOC:(LFC_EMVTradeInfo *)tradeInfo trackInfoSuccess:(onLFEMVProgressCB)trackInfoSuccess failedBlock:(onErrorCB)failedCB
开始 EMV 交易送入数据
@property (nonatomic) LDE_FORCEONLINE flag;//是否强制联机
@property (nonatomic) LDE_TradeType type;//交易类型
@property (nonatomic,copy) NSString* moneyNum;//交易金额
@property (nonatomic,copy) NSString* date;//日期YYMMDD
@property (nonatomic,copy) NSString* time;//时间 HHMMSS

交易流程执行后返回
typedef void (^onLFEMVProgressCB)(LFC_EMVProgress* emvProgress);
LFC_EMVProgress 包含如下信息
@property (nonatomic,copy) NSString *track2data;//IC卡二磁道等效数据
@property (nonatomic,copy) NSString *cardExpired;//IC卡有效期
@property (nonatomic,copy) NSString *panSerialNO;//IC卡序列号
@property (nonatomic,copy) NSString *pan;//IC 卡主账号

3.4.5.2.2 PBOC 继续交易

当调用开始 PBOC 交易成功后可以继续调用该接口继续 PBOC 流程
些指令里边会自己调用 密码输入接口，用户执行 PBOC 交易不用再自己手动调用
- (void) continuePBOC:(LFC_GETPIN *)getPin

successBlock:(onLFEMVStartCB)successCB failedBlock:(onErrorCB)failedCB
送入的 GetPin 内容为
@property (nonatomic, copy) NSString* panBlock;//交易的卡号，卡号会参与密码的计算
@property (nonatomic, copy) NSString* moneyNum;//交易的金额
@property (nonatomic) OneByeInt timeout; //读取密码的超时时间

该接口执行成功后将调用
typedef void (^onLFEMVStartCB)(LFC_EMVResult* emvResult);
@property (nonatomic) LDE_EMVTRADERETCODE result;//PBOC执行结果码
@property (nonatomic,copy) NSData* password;//密码
@property (nonatomic,copy) NSString* dol;//55 域数据，55 域数据前两位为 tag 个数，应用如果不需要，记得自己过滤

result 只可能返回
* @param EMVTRADERETCODE_REJECT 交易拒绝
* @param EMVTRADERETCODE_REQONLINE 请求联机，联机送55域
* @param EMVTRADERETCODE_FALLBACK FALLBACK，已经废弃
* @param EMVTRADERETCODE_EXCEPTION EMV 处理异常
应用在判断 PBOC 执行结果码为请求联机才可以发送报文给交易服务器

3.4.5.2.3 PBOC 联机数据处理

后台返回的数据送了内核再判断交易的最终结果
- (void) onlineDataProcess:(LFC_EMVOnlineData *)onLineData
successBlock:(onEMVTradeResultCB)successCB failedBlock:(onErrorCB)failedCB

LFC_EMVOnlineData 输入两个信息
@property (nonatomic,copy) NSString *responseCode;//服务器处理结果码，同39域的内容如@"00"--交易成功 @"55" –密码错误 @"35" – 余额不足 @"96"服务器故障等
@property (nonatomic,copy) NSString *onlineData;//后台服务器返回的联机结果 55 域数据，一般包括发卡行认证，脚本

执行指令成功后将回调
typedef void (^onEMVTradeResultCB)(LDC_EMVResult* emvResult)
LDC_EMVResult 包含两个信息
@property (nonatomic) LDE_EMVTRADERETCODE result; //PBOC执行结果码
@property (nonatomic, copy) NSString* dol;//TC 及脚本执行结果，前两位为 TAG 个数，应用 不要要记得过滤
result 只可能返回
* @param EMVTRADERETCODE_PERMISION 交易成功，需要上送TC，如果返回的dol里边有DF31标签还要有上送脚本通知
* @param EMVTRADERETCODE_REJECT 交易拒绝
* @param EMVTRADERETCODE_EXCEPTION EMV 处理异常
注意！如果后台执行成功，但此处返回非成功，需要发冲正

3.4.5.2.4 PBOC 交易结束

结束交易，提醒用户拔卡

- (void) PBOCStop:(onVoidCB)successCB failedBlock:(onErrorCB)failedCB

3.4.6 QPBOC 非接触交易接口

仅对设备有非接硬件的设置支持，如带非接天线的 M35 或者 M36，M15 无天线不支持

3.4.6.1 QPBOC 消费

* @param tradeInfo 交易信息元素

- (void) QpbocPurchase:(LFC_EMVTradeInfo *)tradeInfo successBlock:(onEMVTradeResultCB)successCB failedBlock:(onErrorCB)failedCB

LFC_EMVTradeInfo 至少需要输入如下 3 个元素

@property (nonatomic,copy) NSString* moneyNum;//金额

@property (nonatomic,copy) NSString* date;//日期 @"141127"

@property (nonatomic,copy) NSString* time;//时间 @"134733"

交易结果完成后回调判断

typedef void (^onEMVTradeResultCB)(LDC_EMVResult* emvResult)

LDC_EMVResult有两个域：

@property (nonatomic) LDE_EMVTRADERETCODE result;

@property (nonatomic, copy) NSString* dol;//对 QBPOC 交易无意义

LDE_EMVTRADERETCODE 在 QPBOC 交易中只可能返回：

* @param EMVTRADERETCODE_PERMISION 交易批准

* @param EMVTRADERETCODE_REJECT 交易拒绝

* @param EMVTRADERETCODE_REQONLINE 请求联机

* @param EMVTRADERETCODE_QPBOC_FULL QPBOC消费流水已经满

* @param EMVTRADERETCODE_EXCEPTION EMV 处理异常

3.4.6.2 QPBOC 查询流水个数

- (void) QpbocGetRecordNums:(onNSStringCB)successCB failedBlock:(onErrorCB)failedCB

返回的个数请使用[NSString intValue]方法转换成整型

3.4.6.3 QPBOC 获取一条流水内容

- (void) QpbocGetOneRecord:(Byte)recordId

successBlock:(onQPBOCReadRecord)successCB failedBlock:(onErrorCB)failedCB
获取一条流水成功的回调
typedef void (^onQPBOCReadRecord)(LDC_QPOBCReadRecord* qpbocRecord)
LDC_QPOBCReadRecord 包含以下内容
@property (nonatomic,copy) NSData* record;//流水内容
@property (nonatomic,copy) NSString* amount;//消费金额
@property (nonatomic,copy) NSString* pan;//卡号
@property (nonatomic,copy) NSString* panSn;//卡序列号
@property (nonatomic,copy) NSString* time;//时间
@property (nonatomic,copy) NSString* date;//日期

3.4.6.4 QPBOC 删除一条流水

删除流水提供 2 种类型，一种是删除最早的一笔交易，一种是删除最近的一笔交易
- (void) QpbocDelOneRecord:(LDE_QPBOCRECORDTYPE)recordType
successBlock:(onVoidCB)successCB failedBlock:(onErrorCB)failedCB

* @param QPBOC_FIRST_RECORD 第一个记录（最早的）
* @param QPBOC_LAST_RECORD 最后一个记录（最新的）

3.4.6.5 QBPOC 清除所有流水

该接口将删除所有的脱机流水，正常情况下不应该使用，否则会丢交易记录
- (void) QpbocDelAllRecords:(onVoidCB)successCB failedBlock:(onErrorCB)failedCB{

3.4.7 行业卡 APDU 透传指令接口

3.4.7.1 给卡片上电

- (void) powerUpICC:(LDE_ICC_SLOT_TYPE) iccSlot successBlock:
(onNSStringCB)successCB failedBlock:(onErrorCB)failedCB
上电成功将返回 ATR 信息

```
typedef enum {
    IC_SLOT_ICC1        = 0x00,//半埋卡1
    IC_SLOT_PSAM 1      = 0x01,// Psam卡槽1
    IC_SLOT_ICC2        = 0x02,//  半埋卡2
    IC_SLOT_PSAM2       = 0x03,//Psam卡槽2
    IC_SLOT_RF          = 0x04,//RF卡
}LDE_ICC_SLOT_TYPE;
```

3.4.7.2 给卡片下电

- (void) powerDownICC:(LDE_ICC_SLOT_TYPE) iccSlot
successBlock:(onVoidCB)successCB failedBlock:(onErrorCB)failedCB

3.4.7.3 发送 APDU 指令给卡片

- (void) sendApduICC:(LDE_ICC_SLOT_TYPE) iccSlot withApduCmd:(NSString *)apduCmd successBlock:(onNSStringCB)successCB failedBlock:(onErrorCB)failedCB

3.4.8 打印机接口

仅针对有打印机的设备如 M36，M15 或者 M35 不支持

3.4.8.1 获取打印机状态

- (void) getPrinterStatue:(onLDE_PrinterStatusCB)successCB
failedBlock:(onErrorCB)failedCB
获取成功将回调 onLDE_PrinterStatusCB
typedef void (^onLDE_PrinterStatusCB)(LDE_PrinterStatus printerStatues)

LDE_PrinterStatus 定义如下
* @param PRINTERSTATUS_NORMAL 正常
* @param PRINTERSTATUS_NOPAPER 缺纸
* @param PRINTERSTATUS_BUSY 打印机忙
* @param PRINTERSTATUS_NORESPON 打印机无响应
* @param PRINTERSTATUS_ELSEERR 未知错误
* @param PRINTERSTATUS_OUT_OF_BAT 电池电量低
* @param PRINTERSTATUS_NO_BAT 电池不在位

3.4.8.2 简单打印

- (void) printText:(NSString *)text successBlock:(onVoidCB)successCB
failedBlock:(onErrorCB)failedCB

3.4.8.3 多联打印

多联打印功能强大，可以在调用一次 API 接口的情况下完成最多 4 联签购单的打印，每联

打印的内容，以行添加，每行的字体大小，对齐方式都可以独立控制，而且支持打印二维码及电子签名

```
 *    @param        times                  有几联要打印
 *    @param        printContent           打印的文本,每行为LDC_PrintLineStu对象
 *     @param       successCB              成功回调
 *    @param        failedCB               失败回调
- (void) printText:(OneByeInt)times withPrintContent:(NSArray *)printContent
successBlock:(onVoidCB)successCB failedBlock:(onErrorCB)failedCB
LDC_PrintLineStu
@property (nonatomic) LDE_PRINTPOSITION position;
@property (nonatomic) LDE_PRINTALIGN ailg;
@property (nonatomic) LDE_PRINTTYPE type;
@property (nonatomic) LDE_PRINTZOOM zoom;
@property (nonatomic) NSString* text;
```

3.4.8.4 打印点阵图片

打印点阵图片，需要先把图片变成点阵形式，点阵的高度必须是 8 的倍数，,且点阵宽*点阵长/8 必须和打印的点阵数据字节数相等，否则无法打印，受限于蓝牙通讯单包数据量，一次最大仅能打印 1.5K 的数据量

```
/**
 *    @brief 打印点阵图片
 *
 *    @param        offsetDot              打印位置偏移
 *    @param        dotHigh                点阵高度
 *    @param        dotLength              点阵宽度
 *    @param        printContent           打印点阵数据，点阵数据量在1.5K以内
 *     @param       successCB              成功回调
 *     @param       failedCB               失败回调
 *
 *    @returnvoid
 */
- (void) printDot:(int)offsetDot andDotHigh:(int)dotHigh andDotLength:(int)dotLength
andDotData:(NSData *)dotData successBlock:(onVoidCB)successCB
failedBlock:(onErrorCB)failedCB;
```

3.4.8.5 多联 BMP 图片

打印 BMP 图片，传入的是原始的 BMP 文件信息，支持最大到 5 倍的放大，对图片高度没有限制，受限于蓝牙通讯单包数据量，一次最大仅能打印 1.5K 的数据量

```
/**
 *    @brief 打印BMP图片
 *
```

```
 *  @param      offsetDot           打印位置偏移
 *  @param      zoom                图片放大倍数
 *  @param      printContent        BMP图片数据，数据量在1.5K以内
 *  @param      successCB           成功回调
 *  @param      failedCB            失败回调
 *
 *  @returnvoid
 */
- (void) printBmp:(int)offsetDot andZoom:(Byte)zoom andBmpData:(NSData*)bmpData
successBlock:(onVoidCB)successCB failedBlock:(onErrorCB)failedCB;
```

3.4.9 KMS 远程密钥下载接口

待添加…

3.4.9.1 获取终端认证数据

3.4.9.2 认证服务器数据

3.4.9.3 获取密钥下载请求数据

3.4.9.4 下载主密钥

3.4.10 M1 卡操作接口

3.4.10.1 激活卡片

调用该指令前卡片必须在非接感应区内，否则返回失败，可以先调用等待读卡指令waitingCard，在超时时间内放卡返回成功后，继续调用该接口激活。

```
/**
 *  @brief 激活M1卡
 *
 *  @param      successCB   成功回调
 *  @param      failedCB    失败回调
 *
 *  @returnvoid
 */
```

```
- (void) M1Active:(onActiveM1CardCB)successCB
        failedBlock:(onErrorCB)failedCB;
```

回调返回的结构体 LDC_M1CardInfo

```
@property (nonatomic,copy) NSString *ATQ;//卡片ATQ
@property (nonatomic,copy) NSString *M1CardSerial;//卡片内部序列号
@property (nonatomic) LDE_RFCARDTYPE cardType;//卡片类型
```

3.4.10.2 认证块数据

为保证操作的正确性，请遵循认证一块再操作一块的原则操作卡片。

```
/**
 *    @brief M1卡片认证
 *
 *    @param      key             认证密钥为 6个字节 12个BCD字符
 *    @param      blockNo     块号
 *    @param      successCB     成功回调
 *    @param      failedCB      失败回调
 *
 *    @returnvoid
 */
- (void) M1Auth:(NSString *) key
        BlockNo:(Byte)blockNo
    WithKeyType:(LDE_M1KEYTYPE)keyType
        successCB:(onVoidCB)successCB
    failedBlock:(onErrorCB)failedCB;
```

3.4.10.3 读取块数据

读取一个数值据块

```
/**
 *    @brief M1读取块数据
 *
 *    @param      blockNo     块号
 *    @param      successCB     成功回调
 *    @param      failedCB      失败回调
 *
 *    @returnvoid
 */
- (void) M1ReadBlock:(Byte)blockNo
          successCB:(onNSStringCB)successCB
        failedBlock:(onErrorCB)failedCB;
```

3.4.10.4 写入块数据

写入一个数值据块

```
/**
 *    @brief M1写入块数据
 *
 *    @param      blockData   块数据16个字节 32个BCD字符
 *    @param      blockNo     块号
 *    @param      successCB   成功回调
 *    @param      failedCB    失败回调
 *
 *    @returnvoid
 */
- (void) M1WriteBlock:(Byte)blockNo
                BlockData:(NSString *)blockData
                successCB:(onVoidCB)successCB
              failedBlock:(onErrorCB)failedCB;
```

3.4.10.5 增量操作

对数值块增量操作

```
/**
 *    @brief M1增量操作
 *
 *    @param      data        加上的数值
 *    @param      blockNo     块号
 *    @param      successCB   成功回调
 *    @param      failedCB    失败回调
 *
 *    @returnvoid
 */
- (void) M1IncreBlock:(Byte)blockNo
                WithData:(unsigned int)data
                successCB:(onVoidCB)successCB
              failedBlock:(onErrorCB)failedCB;
```

3.4.10.6 减量操作

对数值块减量操作

```
/**
 *    @brief M1减量操作
 *
```

```
 *    @param      data          减去的数值
 *    @param      blockNo       块号
 *    @param      successCB     成功回调
 *    @param      failedCB      失败回调
 *
 *    @returnvoid
 */
- (void) M1DecreBlock:(Byte)blockNo
                 WithData:(unsigned int)data
               successCB:(onVoidCB)successCB
             failedBlock:(onErrorCB)failedCB;
```

3.4.10.7 卡片下电

```
/**
 *    @brief M1下电
 *
 *    @param      successCB     成功回调
 *    @param      failedCB      失败回调
 *
 *    @returnvoid
 */
- (void) M1CardPowerOff:(onVoidCB)successCB
               failedBlock:(onErrorCB)failedCB;
```