龙芯 3A5000 用户手册—基础 SDK 使用指南

卫士通信息产业股份有限公司

二〇二一年十月

目　次

前　言

本程序员手册是 3A5000 的 SDK 接口使用说明，主要适用于服务器密码机终端用户。

本手册详细介绍了 3A5000 的 SDK 接口中使用的宏定义、结构定义和函数描述。

1 范围

本手册详细说明了 3A5000 的 SDK 接口。
本手册适用于 3A5000 终端用户的开发、使用。

2 符号和缩略语

下列缩略语适用于本部分：
ECC 椭圆曲线算法（Elliptic Curve Cryptography）
IPK 内部加密公钥（Internal Public Key）
ISK 内部加密私钥（Internal Private Key）
EPK 外部加密公钥（External Public Key）
KEK 密钥加密密钥（Key Encrypt Key）

3 算法标识和数据结构

3.1 算法标识定义

对称算法标识		
宏描述	预定义值	说明
#define SGD_SM4_ECB	0x00000401	SM4 算法 ECB 加密模式
#define SGD_SM4_CBC	0x00000402	SM4 算法 CBC 加密模式
#define SGD_SM4_MAC	0x00000410	SM4 算法 MAC 加密模式
非对称算法标识		
宏描述		
#define SGD_SM2	0x00020100	椭圆曲线密码算法
杂凑算法标识		
宏描述	预定义值	说明
#define SGD_SM3	0x00000001	SM3 杂凑算法

3.2 设备信息定义

字段名称	数据长度（字节）	含义
IssuerName	40	设备生产厂商名称
DeviceName	16	设备型号
DeviceSerial	16	设备编号，包含：日期（8 字符）、批次号（3 字符）、流水号（5 字符）
DeviceVersion	4	密码设备内部软件的版本号
StandardVersion	4	密码设备支持的接口规范版本号

AsymAlgAbility	8	前 4 字节表示支持的算法,表示方法为非对称算法标识按位或的结果；后 4 字节表示算法的最大模长，表示方法为支持的模长按位或的结果
SymAlgAbility	4	所有支持的对称算法，表示方法为对称算法标识按位或运算结果
HashAlgAbility	4	所有支持的杂凑算法，表示方法为杂凑算法标识按位或运算结果
BufferSize	4	每个文件支持 8k,共 120 个文件

实际数据结构定义：

```
typedef struct DeviceInfo_st{
    unsigned char IssuerName[40];
    unsigned char DeviceName[16];
    unsigned char DeviceSerial[16]
    unsigned int  DeviceVersion;
    unsigned int  StandardVersion;
    unsigned int  AsymAlgAbility[2];
    unsigned int  SymAlgAbility;
    unsigned int  HashAlgAbility;
    unsigned int  BufferSize;
}DEVICEINFO;
```

3.3 ECC 密钥数据结构定义

ECC 密钥结构存储时顺序为从高到低，即密钥存放时从密钥结构数组的最高位开始，最高字节填在最高位，不足位填充数据 0（前面填充 0，例如 256bits 的公钥填充为 0000000000000000000000000000000000000000000000000000000000000000BD04CF7E0995DD05D77699E197E43A0A14ECA589A258BCA49EDC8BBE4AF5ECE6000000000000000000000000000000000000000000000000000000042699F418A143920054BB0A2B07F0C598BBDB3CA3F834152865D8EB5DBA19055）

公钥数据结构定义		
字段名称	数据长度	含义
bits	4	模长
x	ECCref_MAX_LEN	公钥 x 坐标
y	ECCref_MAX_LEN	公钥 y 坐标
私钥数据结构定义		
字段名称	数据长度	含义
bits	4	模长
D	ECCref_MAX_LEN	私钥

实际数据结构定义：

```
#define ECCref_MAX_BITS              512
#define ECCref_MAX_LEN               ((ECCref_MAX_BITS+7) / 8)
typedef struct ECCrefPublicKey_st
{
    unsigned int    bits;
    unsigned char x[ECCref_MAX_LEN];
    unsigned char y[ECCref_MAX_LEN];
} ECCrefPublicKey;

typedef struct ECCrefPrivateKey_st
{
    unsigned int    bits;
    unsigned char D[ECCref_MAX_LEN];
} ECCrefPrivateKey;
```

3.4 ECC 加密数据结构定义

ECC 加密数据结构存储时顺序为从高到低，即密钥存放时从密钥结构数组的最高位开始，最高字节填在最高位，不足位填充数据 0

加密数据结构定义		
字段名称	数据长度	含义
x	ECCref_MAX_LEN	X 分量
y	ECCref_MAX_LEN	Y 分量
M	32	明文的 SM3 杂凑值
L	4	密文数据长度
C	L	密文数据（变长）

实际数据结构定义：

```
typedef struct ECCCipher_st
{
    unsigned char x[ECCref_MAX_LEN];
    unsigned char y[ECCref_MAX_LEN];
    unsigned char M[32];
    unsigned int    L;
    unsigned char C[1];
} ECCCipher;
```

3.5 ECC 签名数据结构定义

ECC 签名数据结构存储时顺序为从高到低，即密钥存放时从密钥结构数组的最高位开始，最高字节填在最高位，不足位填充数据 0

签名数据结构定义		
字段名称	数据长度	含义

r	ECCref_MAX_LEN	签名的 r 部分
s	ECCref_MAX_LEN	签名的 s 部分

实际数据结构定义：

```
typedef struct ECCSignature_st
{
    unsigned char r[ECCref_MAX_LEN];
    unsigned char s[ECCref_MAX_LEN];
} ECCSignature;
```

4 设备接口描述

4.1 设备管理类函数

设备管理类函数包括以下具体函数，各函数返回值见附录 A 错误代码定义：

A. 打开设备：SDF_OpenDevice
B. 关闭设备：SDF_CloseDevice
C. 创建会话：SDF_OpenSession
D. 关闭会话：SDF_CloseSession
E. 获取设备信息：SDF_GetDeviceInfo
F. 产生随机数：SDF_GenerateRandom
G. 获取私钥使用权限：SDF_GetPrivateKeyAccessRight
H. 释放私钥使用权限：SDF_ReleasePrivateKeyAccessRight

4.1.1 打开设备

原型：	int SDF_OpenDevice(void **phDeviceHandle);	
描述：	打开密码设备	
参数：	phDeviceHandle[out]	返回设备句柄
返回值：	0	成功
	非 0	失败，返回错误代码
备注：	phDeviceHandle 由函数初始化并填写内容	

4.1.2 关闭设备

原型：	int SDF_CloseDevice(void *hDeviceHandle);	
描述：	关闭密码设备，并释放相关资源	
参数：	hDeviceHandle[in]	已打开的设备句柄
返回值：	0	成功
	非 0	失败，返回错误代码

4.1.3 创建会话

原型：	int SDF_OpenSession(void *hDeviceHandle, void **phSessionHandle);
描述：	创建与密码设备的会话

参数：	hDeviceHandle[in]	已打开的设备句柄
	phSessionHandle[out]	返回与密码设备建立的新会话句柄
返回值：	0	成功
	非 0	失败，返回错误代码
备注：	本密码设备存储的 Session 数目最多支持 32 个	

4.1.4 关闭会话

原型：	int SDF_CloseSession(void *hSessionHandle);	
描述：	关闭与密码设备已建立的会话，并释放相关资源	
参数：	hSessionHandle [in]	与密码设备已建立的会话句柄
返回值：	0	成功
	非 0	失败，返回错误代码

4.1.5 获取设备信息

原型：	int SDF_GetDeviceInfo (void *hSessionHandle, DEVICEINFO *pstDeviceInfo);	
描述：	获取密码设备能力描述	
参数：	hSessionHandle[in]	与设备建立的会话句柄
	pstDeviceInfo [out]	设备能力描述信息，内容及格式见设备信息定义(其算法等意义跟定义有不同)
返回值：	0	成功
	非 0	失败，返回错误代码

4.1.6 产生随机数

原型：	int SDF_GenerateRandom (void *hSessionHandle, unsigned int uiLength, unsigned char *pucRandom);	
描述：	获取指定长度的随机数	
参数：	hSessionHandle[in]	与设备建立的会话句柄
	uiLength[in]	欲获取的随机数长度,长度不能小于 8 字节
	pucRandom[out]	缓冲区指针，用于存放获取的随机数
返回值：	0	成功
	非 0	失败，返回错误代码

4.1.7 获取私钥使用权限

原型：	int SDF_GetPrivateKeyAccessRight (void *hSessionHandle, unsigned int uiKeyIndex, unsigned char *pucPassword, unsigned int uiPwdLength);	
描述：	获取密码设备内部存储的指定索引私钥的使用权	
参数：	hSessionHandle[in]	与设备建立的会话句柄
	uiKeyIndex[in]	密码设备存储私钥的索引值
	pucPassword[in]	使用私钥权限的标识码
	uiPwdLength[in]	私钥权限标识码长度， 8 字节
返回值：	0	成功
	非 0	失败，返回错误代码
备注：	本密码设备存储的密钥对索引值的的起始索引值为 1，ECC 密钥索引从 1 到 8 每个 Session 在使用私钥前需要获取对应索引的私钥权限。	

4.1.8 释放私钥使用权限

原型：	int SDF_ReleasePrivateKeyAccessRight (void *hSessionHandle, unsigned int uiKeyIndex);	
描述：	释放密码设备存储的指定索引私钥的使用授权	
参数：	hSessionHandle[in]	与设备建立的会话句柄
	uiKeyIndex[in]	密码设备存储私钥索引值
返回值：	0	成功
	非 0	失败，返回错误代码
备注：	本密码设备存储的密钥对索引值的的起始索引值为 1，ECC 密钥索引从 1 到 8	

4.2 密钥管理类函数

密钥管理类函数包括以下具体函数，各函数返回值见附录 A 错误代码定义：（密钥加密密钥索引为 1 到 64,）

A. 导出 ECC 签名公钥：SDF_ExportSignPublicKey_ECC
B. 导出 ECC 加密公钥：SDF_ExportEncPublicKey_ECC
C. 产生 ECC 非对称密钥对并输出：SDF_GenerateKeyPair_ECC
D. 生成会话密钥并用内部 ECC 公钥加密输出：SDF_GenerateKeyWithIPK_ECC
E. 生成会话密钥并用外部 ECC 公钥加密输出：SDF_GenerateKeyWithEPK_ECC
F. 导入会话密钥并用内部 ECC 私钥解密：SDF_ImportKeyWithISK_ECC
G. 生成密钥协商参数并输出：SDF_GenerateAgreementDataWithECC
H. 计算会话密钥：SDF_GenerateKeyWithECC
I. 产生协商参数并计算会话密钥：SDF_GenerateAgreementDataAndKeyWithECC
J. 基于 ECC 算法的数字信封转换：SDF_ExchangeDigitEnvelopeBaseOnECC
K. 生成会话密钥并用密钥加密密钥加密输出：SDF_GenerateKeyWithKEK
L. 导入会话密钥并用密钥加密密钥解密：SDF_ImportKeyWithKEK

M. 导入明文会话密钥：SDF_ImportKey
N. 销毁会话密钥：SDF_DestoryKey

4.2.1 导出 ECC 签名公钥

	int SDF_ExportSignPublicKey_ECC(void *hSessionHandle, unsigned int uiKeyIndex, ECCrefPublicKey *pucPublicKey);	
描述：	导出密码设备内部存储的指定索引位置的签名公钥	
参数：	hSessionHandle[in]	与设备建立的会话句柄
	uiKeyIndex[in]	密码设备存储的 ECC 密钥对索引值
	pucPublicKey[out]	ECC 公钥结构
返回值：	0	成功
	非 0	失败，返回错误代码

4.2.2 导出 ECC 加密公钥

原型：	int SDF_ExportEncPublicKey_ECC(void *hSessionHandle, unsigned int uiKeyIndex, ECCrefPublicKey *pucPublicKey);	
描述：	导出密码设备内部存储的指定索引位置的加密公钥	
参数：	hSessionHandle[in]	与设备建立的会话句柄
	uiKeyIndex[in]	密码设备存储的 ECC 密钥对索引值
	pucPublicKey[out]	ECC 公钥结构
返回值：	0	成功
	非 0	失败，返回错误代码

4.2.3 产生 ECC 密钥对并输出

原型：	int SDF_GenerateKeyPair_ECC(void *hSessionHandle, unsigned int uiAlgID, unsigned int uiKeyBits, ECCrefPublicKey *pucPublicKey, ECCrefPrivateKey *pucPrivateKey);	
描述：	请求密码设备产生指定类型和模长的 ECC 密钥对	
参数：	hSessionHandle[in]	与设备建立的会话句柄
	uiAlgID[in]	指定算法标识
	uiKeyBits [in]	指定密钥长度
	pucPublicKey[out]	ECC 公钥结构
	pucPrivateKey[out]	ECC 私钥结构
返回值：	0	成功

	非 0	失败，返回错误代码

4.2.4 生成会话密钥并用内部 ECC 公钥加密输出

原型：	int SDF_GenerateKeyWithIPK_ECC (void *hSessionHandle, unsigned int uiIPKIndex, unsigned int uiKeyBits, ECCCipher *pucKey, void **phKeyHandle);	
描述：	生成会话密钥并用指定索引的内部 ECC 加密公钥加密输出，同时返回密钥句柄	
参数：	hSessionHandle[in]	与设备建立的会话句柄
	uiIPKIndex[in]	密码设备内部存储公钥的索引值
	uiKeyBits[in]	指定产生的会话密钥长度
	pucKey[out]	缓冲区指针，用于存放返回的密钥密文
	phKeyHandle[out]	返回的密钥句柄
返回值：	0	成功
	非 0	失败，返回错误代码
备注：	返回的密钥不包含 IV 部分。	

4.2.5 生成会话密钥并用外部 ECC 公钥加密输出

原型：	int SDF_GenerateKeyWithEPK_ECC (void *hSessionHandle, unsigned int uiKeyBits, unsigned int uiAlgID, ECCrefPublicKey *pucPublicKey, ECCCipher *pucKey, void **phKeyHandle);	
描述：	生成会话密钥并用外部 ECC 公钥加密输出，同时返回密钥句柄	
参数：	hSessionHandle[in]	与设备建立的会话句柄
	uiKeyBits[in]	指定产生的会话密钥长度
	uiAlgID[in]	外部 ECC 公钥的算法标识
	pucPublicKey[in]	输入的外部 ECC 公钥结构
	pucKey[out]	缓冲区指针，用于存放返回的密钥密文
	phKeyHandle[out]	返回的密钥句柄
返回值：	0	成功
	非 0	失败，返回错误代码
备注：	返回的密钥不包含 IV 部分。	

4.2.6 导入会话密钥并用内部 ECC 私钥解密

原型：	int SDF_ImportKeyWithISK_ECC (void *hSessionHandle, unsigned int uiISKIndex, ECCCipher *pucKey, void **phKeyHandle);	
描述：	导入会话密钥并用内部 ECC 加密私钥解密，同时返回密钥句柄	
参数：	hSessionHandle[in]	与设备建立的会话句柄
	uiISKIndex[in]	密码设备内部存储加密私钥的索引值，对应于加密时的公钥
	pucKey[in]	缓冲区指针，用于存放输入的密钥密文
	phKeyHandle[out]	返回的密钥句柄
返回值：	0	成功
	非 0	失败，返回错误代码

4.2.7 生成密钥协商参数并输出

原型：	int SDF_GenerateAgreementDataWithECC (void *hSessionHandle, unsigned int uiISKIndex, unsigned int uiKeyBits, unsigned char *pucSponsorID, unsigned int uiSponsorIDLength, ECCrefPublicKey *pucSponsorPublicKey, ECCrefPublicKey *pucSponsorTmpPublicKey, void **phAgreementHandle);	
描述：	使用 ECC 密钥协商算法，为计算会话密钥而产生协商参数，同时返回指定索引位置的 ECC 公钥、临时 ECC 密钥对的公钥及协商句柄。	
参数：	hSessionHandle[in]	与设备建立的会话句柄
	uiISKIndex[in]	密码设备内部存储加密私钥的索引值，该私钥用于参与密钥协商
	uiKeyBits[in]	要求协商的密钥长度
	pucSponsorID[in]	参与密钥协商的发起方 ID 值
	uiSponsorIDLength[in]	发起方 ID 长度，小于 64
	pucSponsorPublicKey[out]	返回的发起方 ECC 公钥结构
	pucSponsorTmpPublicKey[out]	返回的发起方临时 ECC 公钥结构
	phAgreementHandle[out]	返回的协商句柄，用于计算协商密钥
返回值：	0	成功
	非 0	失败，返回错误代码
备注：	为协商会话密钥，协商的发起方应首先调用本函数。 如果在具体的应用中，协商双方没有统一分配的 ID，可以将 ID 设定为常量。	

4.2.8 计算会话密钥

原型：	int SDF_GenerateKeyWithECC (void *hSessionHandle, unsigned char *pucResponseID, unsigned int uiResponseIDLength， ECCrefPublicKey *pucResponsePublicKey, ECCrefPublicKey *pucResponseTmpPublicKey, void *hAgreementHandle, void **phKeyHandle);	
描述：	使用 ECC 密钥协商算法，使用自身协商句柄和响应方的协商参数计算会话密钥，同时返回会话密钥句柄。	
参数：	hSessionHandle[in]	与设备建立的会话句柄
	pucResponseID[in]	外部输入的响应方 ID 值
	uiResponseIDLength[in]	外部输入的响应方 ID 长度，小于 64
	pucResponsePublicKey[in]	外部输入的响应方 ECC 公钥结构
	pucResponseTmpPublicKey[in]	外部输入的响应方临时 ECC 公钥结构
	hAgreementHandle[in]	协商句柄，用于计算协商密钥
	phKeyHandle[out]	返回的密钥句柄
返回值：	0	成功
	非 0	失败，返回错误代码
备注：	协商的发起方获得响应方的协商参数后调用本函数，计算会话密钥。 如果在具体的应用中，协商双方没有统一分配的 ID，可以将 ID 设定为常量。	

4.2.9 产生协商参数并计算会话密钥

原型：	int SDF_GenerateAgreementDataAndKeyWithECC (void *hSessionHandle, unsigned int uiISKIndex, unsigned int uiKeyBits, unsigned char *pucResponseID, unsigned int uiResponseIDLength, unsigned char *pucSponsorID, unsigned int uiSponsorIDLength, ECCrefPublicKey *pucSponsorPublicKey, ECCrefPublicKey *pucSponsorTmpPublicKey, ECCrefPublicKey *pucResponsePublicKey, ECCrefPublicKey *pucResponseTmpPublicKey, void **phKeyHandle);	
描述：	使用 ECC 密钥协商算法，产生协商参数并计算会话密钥，同时返回产生的协商参数和和密钥句柄。	
参数：	hSessionHandle[in]	与设备建立的会话句柄
	uiISKIndex[in]	密码设备内部存储加密私钥的索引值，该私钥用于参与密钥协商

	uiKeyBits[in]	协商后要求输出的密钥长度
	pucResponseID[in]	响应方 ID 值
	uiResponseIDLength[in]	响应方 ID 长度，小于 64
	pucSponsorID[in]	发起方 ID 值
	uiSponsorIDLength[in]	发起方 ID 长度，小于 64
	pucSponsorPublicKey[in]	外部输入的发起方 ECC 公钥结构
	pucSponsorTmpPublicKey[in]	外部输入的发起方临时 ECC 公钥结构
	pucResponsePublicKey[out]	返回的响应方 ECC 公钥结构
	pucResponseTmpPublicKey[out]	返回的响应方临时 ECC 公钥结构
	phKeyHandle[out]	返回的密钥句柄
返回值：	0	成功
	非 0	失败，返回错误代码
备注：	本函数由响应方调用。 如果在具体的应用中，协商双方没有统一分配的 ID，可以将 ID 设定为常量	

4.2.10 基于 ECC 算法的数字信封转换

	int SDF_ExchangeDigitEnvelopeBaseOnECC(void *hSessionHandle, unsigned int uiKeyIndex, unsigned int uiAlgID, ECCrefPublicKey *pucPublicKey, ECCCipher *pucEncDataIn, ECCCipher *pucEncDataOut);	
描述：	将由内部加密公钥加密的会话密钥转换为由外部指定的公钥加密，可用于数字信封转换。	
参数：	hSessionHandle[in]	与设备建立的会话句柄
	uiKeyIndex[in]	密码设备存储的 ECC 密钥对索引值
	uiAlgID[in]	外部 ECC 公钥的算法标识
	pucPublicKey [in]	外部 ECC 公钥结构
	pucEncDataIn[in]	缓冲区指针，用于存放输入的会话密钥密文
	pucEncDataOut[out]	缓冲区指针，用于存放输出的会话密钥密文
返回值：	0	成功
	非 0	失败，返回错误代码

4.2.11 生成会话密钥并用密钥加密密钥加密输出

原型：	int SDF_GenerateKeyWithKEK (void *hSessionHandle, unsigned int uiKeyBits, unsigned int uiAlgID, unsigned int uiKEKIndex, unsigned char *pucKey, unsigned int *puiKeyLength, void **phKeyHandle);	
描述：	生成会话密钥并用密钥加密密钥加密输出，同时返回密钥句柄。	
参数：	hSessionHandle[in]	与设备建立的会话句柄
	uiKeyBits[in]	指定产生的会话密钥长度
	uiAlgID[in]	算法标识，指定对称加密算法
	uiKEKIndex[in]	密码设备内部存储密钥加密密钥的索引值
	pucKey[out]	缓冲区指针，用于存放返回的密钥密文
	puiKeyLength[out]	返回的密钥密文长度
	phKeyHandle[out]	返回的密钥句柄
返回值：	0	成功
	非 0	失败，返回错误代码

4.2.12 导入会话密钥并用密钥加密密钥解密

原型：	int SDF_ImportKeyWithKEK (void *hSessionHandle, unsigned int uiAlgID, unsigned int uiKEKIndex, unsigned char *pucKey, unsigned int *puiKeyLength, void **phKeyHandle);	
描述：	导入会话密钥并用密钥加密密钥解密，同时返回会话密钥句柄。	
参数：	hSessionHandle[in]	与设备建立的会话句柄
	uiAlgID[in]	算法标识，指定对称加密算法
	uiKEKIndex[in]	密码设备内部存储密钥加密密钥的索引值
	pucKey[in]	缓冲区指针，用于存放输入的密钥密文
	puiKeyLength[in]	输入的密钥密文长度
	phKeyHandle[out]	返回的密钥句柄
返回值：	0	成功
	非 0	失败，返回错误代码
备注：	加密模式使用 ECB 模式。	

4.2.13 导入明文会话密钥

原型：	int SDF_ImportKey (void *hSessionHandle, unsigned char *pucKey, unsigned int uiKeyLength, void **phKeyHandle);	
描述：	导入明文会话密钥，同时返回密钥句柄	
参数：	hSessionHandle[in]	与设备建立的会话句柄
	pucKey[in]	缓冲区指针，用于存放输入的密钥明文
	puiKeyLength[in]	输入的密钥明文长度
	phKeyHandle[out]	返回的密钥句柄
返回值：	0	成功
	非 0	失败，返回错误代码

4.2.14 销毁会话密钥

原型：	int SDF_DestroyKey (void *hSessionHandle, void *hKeyHandle);	
描述：	销毁会话密钥，并释放为密钥句柄分配的内存等资源。	
参数：	hSessionHandle[in]	与设备建立的会话句柄
	hKeyHandle[in]	输入的密钥句柄
返回值：	0	成功
	非 0	失败，返回错误代码
备注：	在对称算法运算完成后，应调用本函数销毁会话密钥。	

4.3 非对称算法运算类函数

非对称算法运算类函数包括以下具体函数，各函数返回值见附录 A 错误代码定义：

- 外部密钥 ECC 签名：SDF_ExternalSign_ECC
- 外部密钥 ECC 验证：SDF_ExternalVerify_ECC
- 内部密钥 ECC 签名：SDF_InternalSign_ECC
- 内部密钥 ECC 验证：SDF_InternalVerify_ECC
- 外部密钥 ECC 加密：SDF_ExternalEncrytp_ECC
- 外部密钥 ECC 解密：SDF_ExternalDecrypt_ECC

4.3.1 外部密钥 ECC 签名

原型：	int SDF_ExternalSign_ECC(void *hSessionHandle, unsigned int uiAlgID, ECCrefPrivateKey *pucPrivateKey, unsigned char *pucData, unsigned int uiDataLength, ECCSignature *pucSignature);	
描述：	使用外部 ECC 私钥对数据进行签名运算	
参数：	hSessionHandle[in]	与设备建立的会话句柄
	uiAlgID[in]	算法标识，指定使用的 ECC 算法
	pucPrivateKey[in]	外部 ECC 私钥结构
	pucData[in]	缓冲区指针，用于存放外部输入的数据
	uiDataLength[in]	输入的数据长度
	pucSignature[out]	缓冲区指针，用于存放输出的签名值数据
返回值：	0	成功
	非 0	失败，返回错误代码
备注：	对原文的杂凑运算，在函数外部完成。	

4.3.2 外部密钥 ECC 验证

原型：	int SDF_ExternalVerify_ECC(void *hSessionHandle, unsigned int uiAlgID, ECCrefPublicKey *pucPublicKey, unsigned char *pucDataInput, unsigned int uiInputLength, ECCSignature *pucSignature);	
描述：	使用外部 ECC 公钥对 ECC 签名值进行验证运算	
参数：	hSessionHandle[in]	与设备建立的会话句柄
	uiAlgID[in]	算法标识，指定使用的 ECC 算法
	pucPublicKey[in]	外部 ECC 公钥结构
	pucData[in]	缓冲区指针，用于存放外部输入的数据
	uiDataLength[in]	输入的数据长度
	pucSignature[in]	缓冲区指针，用于存放输入的签名值数据
返回值：	0	成功
	非 0	失败，返回错误代码
备注：	对原文的杂凑运算，在函数外部完成。	

4.3.3 内部密钥 ECC 签名

原型：	int SDF_InternalSign_ECC(void *hSessionHandle, unsigned int uiISKIndex, unsigned char *pucData, unsigned int uiDataLength, ECCSignature *pucSignature);	
描述：	使用内部 ECC 私钥对数据进行签名运算	
参数：	hSessionHandle[in]	与设备建立的会话句柄
	uiISKIndex [in]	密码设备内部存储的 ECC 签名私钥的索引值
	pucData[in]	缓冲区指针，用于存放外部输入的数据
	uiDataLength[in]	输入的数据长度
	pucSignature [out]	缓冲区指针，用于存放输出的签名值数据
返回值：	0	成功
	非 0	失败，返回错误代码
备注：	对原文的杂凑运算，在函数外部完成。	

4.3.4 内部密钥 ECC 验证

原型：	int SDF_InternalVerify_ECC(void *hSessionHandle, unsigned int uiISKIndex, unsigned char *pucData, unsigned int uiDataLength, ECCSignature *pucSignature);	
描述：	使用内部 ECC 公钥对 ECC 签名值进行验证运算	
参数：	hSessionHandle[in]	与设备建立的会话句柄
	uiISKIndex [in]	密码设备内部存储的 ECC 签名公钥的索引值
	pucData[in]	缓冲区指针，用于存放外部输入的数据
	uiDataLength[in]	输入的数据长度
	pucSignature[in]	缓冲区指针，用于存放输入的签名值数据
返回值：	0	成功
	非 0	失败，返回错误代码
备注：	对原文的杂凑运算，在函数外部完成。	

4.3.5 外部密钥 ECC 公钥加密

原型：	int SDF_ExternalEncrypt_ECC(void *hSessionHandle, unsigned int uiAlgID, ECCrefPublicKey *pucPublicKey, unsigned char *pucData, unsigned int uiDataLength, ECCCipher *pucEncData);	
描述：	使用外部 ECC 公钥对数据进行加密运算	
参数：	hSessionHandle[in]	与设备建立的会话句柄
	uiAlgID[in]	算法标识，指定使用的 ECC 算法
	pucPublicKey[in]	外部 ECC 公钥结构
	pucData[in]	缓冲区指针，用于存放外部输入的数据
	uiDataLength[in]	输入的数据长度
	pucEncData[out]	缓冲区指针，用于存放输出的数据密文
返回值：	0	成功
	非 0	失败，返回错误代码
备注：	输入的数据长度 uiDataLength 不大于 1024。	

4.3.6 外部密钥 ECC 私钥解密

原型：	int SDF_ExternalDecrypt_ECC(void *hSessionHandle, unsigned int uiAlgID, ECCrefPrivateKey *pucPrivateKey, ECCCipher *pucEncData, unsigned char *pucData, unsigned int *puiDataLength);	
描述：	使用外部 ECC 私钥进行解密运算	
参数：	hSessionHandle[in]	与设备建立的会话句柄
	uiAlgID[in]	算法标识，指定使用的 ECC 算法
	pucPrivateKey[in]	外部 ECC 私钥结构
	pucEncData[in]	缓冲区指针，用于存放输入的数据密文
	pucData[out]	缓冲区指针，用于存放输出的数据明文
	puiDataLength[out]	输出的数据明文长度
返回值：	0	成功
	非 0	失败，返回错误代码

4.4 对称算法运算类函数

对称算法运算类函数包括以下具体函数，各函数返回值见附录 A 错误代码定义：

- 对称加密：SDF_Encrypt(支持算法 SM4)
- 对称解密：SDF_Decrypt(支持算法 SM4)

- 计算 MAC：SDF_CalculateMAC(仅支持 SM4)

4.4.1 对称加密

原型：	int SDF_Encrypt(void *hSessionHandle, void *hKeyHandle, unsigned int uiAlgID, unsigned char *pucIV, unsigned char *pucData, unsigned int uiDataLength, unsigned char *pucEncData, unsigned int *puiEncDataLength);	
描述：	使用指定的密钥句柄和 IV 对数据进行对称加密运算	
参数：	hSessionHandle[in]	与设备建立的会话句柄
	hKeyHandle[in]	指定的密钥句柄
	uiAlgID[in]	算法标识，指定对称加密算法
	pucIV[in|out]	缓冲区指针，用于存放输入和返回的 IV 数据
	pucData[in]	缓冲区指针，用于存放输入的数据明文
	uiDataLength[in]	输入的数据明文长度
	pucEncData[out]	缓冲区指针，用于存放输出的数据密文
	puiEncDataLength[out]	输出的数据密文长度
返回值：	0	成功
	非 0	失败，返回错误代码
备注：	此函数不对数据进行填充处理，输入的数据必须是指定算法分组长度的整数倍，数据输入长度不大于 3968 字节。	

4.4.2 对称解密

原型：	int SDF_Decrypt (void *hSessionHandle, void *hKeyHandle, unsigned int uiAlgID, unsigned char *pucIV, unsigned char *pucEncData, unsigned int uiEncDataLength, unsigned char *pucData, unsigned int *puiDataLength);	
描述：	使用指定的密钥句柄和 IV 对数据进行对称解密运算	
参数：	hSessionHandle[in]	与设备建立的会话句柄
	hKeyHandle[in]	指定的密钥句柄
	uiAlgID[in]	算法标识，指定对称加密算法
	pucIV[in|out]	缓冲区指针，用于存放输入和返回的 IV 数据
	pucEncData[in]	缓冲区指针，用于存放输入的数据密文

	uiEncDataLength[in]	输入的数据密文长度
	pucData[out]	缓冲区指针，用于存放输出的数据明文
	puiDataLength[out]	输出的数据明文长度
返回值：	0	成功
	非 0	失败，返回错误代码
备注：	此函数不对数据进行填充处理，输入的数据必须是指定算法分组长度的整数倍，数据输入长度不大于 3968 字节。	

4.4.3 计算 MAC

原型：	int SDF_CalculateMAC(void *hSessionHandle, void *hKeyHandle, unsigned int uiAlgID, unsigned char *pucIV, unsigned char *pucData, unsigned int uiDataLength, unsigned char *pucMAC, unsigned int *puiMACLength);	
描述：	使用指定的密钥句柄和 IV 对数据进行 MAC 运算	
参数：	hSessionHandle[in]	与设备建立的会话句柄
	hKeyHandle[in]	指定的密钥句柄
	uiAlgID[in]	算法标识，指定 MAC 加密算法
	pucIV[in|out]	缓冲区指针，用于存放输入和返回的 IV 数据
	pucData[in]	缓冲区指针，用于存放输出的数据明文
	uiDataLength[in]	输出的数据明文长度,16 字节整数倍
	pucMAC[out]	缓冲区指针，用于存放输出的 MAC 值
	puiMACLength[out]	输出的 MAC 值长度
返回值：	0	成功
	非 0	失败，返回错误代码
备注：	此函数不对数据进行分包处理，多包数据 MAC 运算由 IV 控制最后的 MAC 值，数据输入长度不大于 3968 字节。	

4.5 杂凑运算类函数

杂凑运算类函数包括以下具体函数，各函数返回值见附录 A 错误代码定义：

- 杂凑运算初始化：SDF_HashInit(支持 SM3)
- 多包杂凑运算：SDF_HashUpdate
- 杂凑运算结束：SDF_HashFinal

4.5.1 杂凑运算初始化

原型：	int SDF_HashInit(void *hSessionHandle, unsigned int uiAlgID ECCrefPublicKey *pucPublicKey, unsigned char *pucID, unsigned int uiIDLength);	
描述：	三步式数据杂凑运算第一步。	
参数：	hSessionHandle[in]	与设备建立的会话句柄
	uiAlgID[in]	指定杂凑算法标识
	pucPublicKey[in]	签名者的 ECC 公钥，产生用于 ECC 签名的杂凑值时有效
	pucID[in]	签名者的 ID 值，产生用于 ECC 签名的杂凑值时有效
	uiIDLength[in]	签名者的 ID 长度
返回值：	0	成功
	非 0	失败，返回错误代码
备注：	如果在具体的应用中，协商双方没有统一分配的 ID，可以将 ID 设定为常量。	

4.5.2 多包杂凑运算

原型：	int SDF_HashUpdate(void *hSessionHandle, unsigned char *pucData, unsigned int uiDataLength);	
描述：	三步式数据杂凑运算第二步，对输入的明文进行杂凑运算	
参数：	hSessionHandle[in]	与设备建立的会话句柄
	pucData[in]	缓冲区指针，用于存放输入的数据明文
	uiDataLength[in]	输入的数据明文长度
返回值：	0	成功
	非 0	失败，返回错误代码

4.5.3 杂凑运算结束

原型：	int SDF_HashFinal(void *hSessionHandle, unsigned char *pucHash, unsigned int *puiHashLength);	
描述：	三步式数据杂凑运算第三步，杂凑运算结束返回杂凑数据并清除中间数据	
参数：	hSessionHandle[in]	与设备建立的会话句柄
	pucHash[out]	缓冲区指针，用于存放输出的杂凑数据
	puiHashLength[out]	返回的杂凑数据长度

返回值：	0	成功
	非 0	失败，返回错误代码

4.6 用户文件操作类函数

用户文件操作类函数包括以下具体函数，各函数返回值见附录 A 错误代码定义：

A. 创建文件：SDF_CreateFile
B. 读取文件：SDF_ReadFile
C. 写文件：SDF_WriteFile
D. 删除文件：SDF_DeleteFile

4.6.1 创建文件

原型：	int SDF_CreateFile(void *hSessionHandle, unsigned char *pucFileName， unsigned int uiNameLen， unsigned int uiFileSize);	
描述：	在密码设备内部创建用于存储用户数据的文件	
参数：	hSessionHandle[in]	与设备建立的会话句柄
	pucFileName[in]	缓冲区指针，用于存放输入的文件名，最大长度 128 字节
	uiNameLen[in]	文件名长度，
	uiFileSize[in]	文件所占存储空间的长度,不超过 8k,文件个数不超过 120.
返回值：	0	成功
	非 0	失败，返回错误代码

4.6.2 读取文件

原型：	int SDF_ReadFile(void *hSessionHandle, unsigned char *pucFileName， unsigned int uiNameLen， unsigned int uiOffset, unsigned int *puiFileLength, unsigned char *pucBuffer);	
描述：	读取在密码设备内部存储用户数据的文件的内容	
参数：	hSessionHandle[in]	与设备建立的会话句柄
	pucFileName[in]	缓冲区指针，用于存放输入的文件名，最大长度 128 字节
	uiNameLen[in]	文件名长度
	uiOffset[in]	指定读取文件时的偏移值
	puiFileLength[in|out]	入参时指定读取文件内容的长度；出参时返回实际读取文件内容的长度
	pucBuffer[out]	缓冲区指针，用于存放读取的文件数据

返回值：	0	成功
	非 0	失败，返回错误代码

4.6.3 写文件

原型：	int SDF_WriteFile(void *hSessionHandle, unsigned char *pucFileName， unsigned int uiNameLen， unsigned int uiOffset, unsigned int uiFileLength, unsigned char *pucBuffer);	
描述：	向密码设备内部存储用户数据的文件中写入内容	
参数：	hSessionHandle[in]	与设备建立的会话句柄
	pucFileName[in]	缓冲区指针，用于存放输入的文件名，最大长度 128 字节
	uiNameLen[in]	文件名长度
	uiOffset[in]	指定写入文件时的偏移值
	uiFileLength[in]	指定写入文件内容的长度
	pucBuffer[in]	缓冲区指针，用于存放输入的写文件数据
返回值：	0	成功
	非 0	失败，返回错误代码

4.6.4 删除文件

原型：	int SDF_DeleteFile(void *hSessionHandle, unsigned char *pucFileName， unsigned int uiNameLen);	
描述：	删除指定文件名的密码设备内部存储用户数据的文件	
参数：	hSessionHandle[in]	与设备建立的会话句柄
	pucFileName[in]	缓冲区指针，用于存放输入的文件名，最大长度 128 字节
	uiNameLen[in]	文件名长度
返回值：	0	成功
	非 0	失败，返回错误代码

附录 A

错误代码定义(BASE+预定义值)

错误代码标识		
宏描述	预定义值	说明
#define SDR_OK	0x0	操作成功

#define SDR_BASE	0x01000000	错误码基础值
#define SDR_UNKNOWERR	SDR_BASE + 0x00000001	未知错误
#define SDR_NOTSUPPORT	SDR_BASE + 0x00000002	不支持的接口调用
#define SDR_COMMFAIL	SDR_BASE + 0x00000003	与设备通信失败
#define SDR_HARDFAIL	SDR_BASE + 0x00000004	运算模块无响应
#define SDR_OPENDEVICE	SDR_BASE + 0x00000005	打开设备失败
#define SDR_OPENSESSION	SDR_BASE + 0x00000006	创建会话失败
#define SDR_PARDENY	SDR_BASE + 0x00000007	无私钥使用权限
#define SDR_KEYNOTEXIST	SDR_BASE + 0x00000008	不存在的密钥调用
#define SDR_ALGNOTSUPPORT	SDR_BASE + 0x00000009	不支持的算法调用
#define SDR_ALGMODNOTSUPPORT	SDR_BASE + 0x0000000A	不支持的算法模式调用
#define SDR_PKOPERR	SDR_BASE + 0x0000000B	公钥运算失败
#define SDR_SKOPERR	SDR_BASE + 0x0000000C	私钥运算失败
#define SDR_SIGNERR	SDR_BASE + 0x0000000D	签名运算失败
#define SDR_VERIFYERR	SDR_BASE + 0x0000000E	验证签名失败
#define SDR_SYMOPERR	SDR_BASE + 0x0000000F	对称算法运算失败
#define SDR_STEPERR	SDR_BASE + 0x00000010	多步运算步骤错误
#define SDR_FILESIZEERR	SDR_BASE + 0x00000011	文件长度超出限制
#define SDR_FILENOEXIST	SDR_BASE + 0x00000012	指定的文件不存在
#define SDR_FILEOFSERR	SDR_BASE + 0x00000013	文件起始位置错误
#define SDR_KEYTYPEERR	SDR_BASE + 0x00000014	密钥类型错误
#define SDR_KEYERR	SDR_BASE + 0x00000015	密钥错误
#define SDR_ENCDATAERR	SDR_BASE + 0x00000016	ECC 加密数据错?
#define SDR_RANDERR	SDR_BASE + 0x00000017	随机数产失败
#define SDR_PRDRERR	SDR_BASE + 0x00000018	私钥使用权限获取失败
#define SDR_MACERR	SDR_BASE + 0x00000019	MAC 运算失败
#define SDR_FILEEXISTS	SDR_BASE + 0x0000001A	指定文件已存在
#define SDR_FILEWERR	SDR_BASE + 0x0000001B	文件写入失败
#define SDR_NOBUFFER	SDR_BASE + 0x0000001C	存储空间不足
#define SDR_INARGERR	SDR_BASE + 0x0000001D	输入参数错误
#define SDR_OUTARDERR	SDR_BASE + 0x0000001E	输出参数错误
#define SDR_HASHINITERR	SDR_BASE + 0x00000024	HASH 初始化失败
#define SDR_HASHUPDATAERR	SDR_BASE + 0x00000025	多包 HASHUPDATE 失败
#define SDR_HASHFINALERR	SDR_BASE + 0x00000026	多包 HASHFINAL 失败
#define SDR_AUTHERR	SDR_BASE + 0x00000027	管理员认证失败
#define SDR_IMPORTSYMKEYERR	SDR_BASE + 0x00000028	导入对称密钥失败

#define SDR_IMPORTASYMKEYERR	SDR_BASE + 0x00000029	导入非对称密钥失败
#define SDR_VERIFYKEYERR	SDR_BASE + 0x00000030	密钥校验失败