智能IC卡及智能密码钥匙

密码应用接口规范

Smart Card and Smart Token

Cryptography Application Interface Specification

国家密码管理局

2010年4月

**目 次**

前 言 IV

1 范围 1

2 规范性引用文件 1

3 术语和定义 1

4 缩略语 2

5 结构模型 2

5.1 层次关系 2

5.2 设备的应用结构 3

6 数据类型定义 4

6.1 算法标识 4

6.1.1 分组密码算法标识 4

6.1.2 非对称密码算法标识 5

6.1.3 密码杂凑算法标识 5

6.2 基本数据类型 5

6.3 常量定义 6

6.4 复合数据类型 6

6.4.1 版本 6

6.4.2 设备信息 6

6.4.3 RSA公钥数据结构 7

6.4.4 RSA私钥数据结构 8

6.4.5 ECC公钥数据结构 8

6.4.6 ECC私钥数据结构 9

6.4.7 ECC密文数据结构 9

6.4.8 ECC签名数据结构 9

6.4.9 分组密码参数 10

6.4.10 文件属性 10

6.4.11 权限类型 11

6.4.12 设备状态 11

7 接口函数 11

7.1 设备管理 11

7.1.1 概述 11

7.1.2 等待设备插拔事件 11

7.1.3 取消等待设备插拔事件 12

7.1.4 枚举设备 12

7.1.5 连接设备 12

7.1.6 断开连接 12

7.1.7 获取设备状态 12

7.1.8 设置设备标签 12

7.1.9 获取设备信息 13

7.1.10 锁定设备 13

7.1.11 解锁设备 13

7.1.12 设备命令传输 13

7.2 访问控制 13

7.2.1 概述 13

7.2.2 修改设备认证密钥 14

7.2.3 设备认证 14

7.2.4 修改PIN 14

7.2.5 获取PIN信息 14

7.2.6 校验PIN 15

7.2.7 解锁PIN 15

7.2.8 清除应用安全状态 15

7.3 应用管理 15

7.3.1 概述 15

7.3.2 创建应用 15

7.3.3 枚举应用 16

7.3.4 删除应用 16

7.3.5 打开应用 16

7.3.6 关闭应用 16

7.4 文件管理 17

7.4.1 概述 17

7.4.2 创建文件 17

7.4.3 删除文件 17

7.4.4 枚举文件 17

7.4.5 获取文件属性 18

7.4.6 读文件 18

7.4.7 写文件 18

7.5 容器管理 18

7.5.1 概述 18

7.5.2 创建容器 19

7.5.3 删除容器 19

7.5.4 打开容器 19

7.5.5 关闭容器 19

7.5.6 枚举容器 19

7.6 密码服务 20

7.6.1 概述 20

7.6.2 生成随机数 21

7.6.3 生成外部RSA密钥对 21

7.6.4 生成RSA签名密钥对 21

7.6.5 导入RSA加密密钥对 21

7.6.6 RSA签名 22

7.6.7 RSA验签 22

7.6.8 RSA生成并导出会话密钥 22

7.6.9 RSA外来公钥运算 23

7.6.10 RSA外来私钥运算 23

7.6.11 生成ECC签名密钥对 23

7.6.12 导入ECC加密密钥对 24

7.6.13 ECC签名 24

7.6.14 ECC验签 24

7.6.15 ECC生成并导出会话密钥 24

7.6.16 ECC外来公钥加密 25

7.6.17 ECC外来私钥解密 25

7.6.18 ECC外来私钥签名 25

7.6.19 ECC外来公钥验签 25

7.6.20 ECC生成密钥协商参数并输出 26

7.6.21 ECC产生协商数据并计算会话密钥 26

7.6.22 ECC计算会话密钥 26

7.6.23 导出公钥 27

7.6.24 导入会话密钥 27

7.6.25 明文导入会话密钥 27

7.6.26 加密初始化 28

7.6.27 单组数据加密 28

7.6.28 多组数据加密 28

7.6.29 结束加密 28

7.6.30 解密初始化 29

7.6.31 单组数据解密 29

7.6.32 多组数据解密 29

7.6.33 结束解密 30

7.6.34 密码杂凑初始化 30

7.6.35 单组数据密码杂凑 30

7.6.36 多组数据密码杂凑 30

7.6.37 结束密码杂凑 31

7.6.38 消息鉴别码运算初始化 31

7.6.39 单组数据消息鉴别码运算 31

7.6.40 多组数据消息鉴别码运算 32

7.6.41 结束消息鉴别码运算 32

7.6.42 关闭密码对象句柄 32

8 设备的安全要求 32

8.1 设备使用阶段 32

8.2 权限管理 32

8.2.1 权限分类 32

8.2.2 权限使用 32

8.2.3 设备认证 33

8.2.4 PIN码安全要求 33

8.3 密钥安全要求 33

8.4 设备抗攻击要求 33

附录A （规范性附录） 错误代码定义和说明 34

**前 言**

本规范涉及的密码算法按照国家密码管理部门的要求使用。

本规范的附录A为规范性附录。

本规范由国家密码管理局提出并归口。

本规范起草单位：北京海泰方圆科技有限公司、北京握奇智能科技有限公司、北京大明五洲科技有限公司、恒宝股份有限公司、深圳市明华澳汉科技股份有限公司、武汉天喻信息产业股份有限公司、北京飞天诚信科技有限公司、华翔腾数码科技有限公司。

本规范主要起草人：石玉平、柳增寿、胡俊义、管延军、项莉、雷继业、胡鹏、赵再兴、段晓毅、刘玉峰、刘伟丰、陈吉、何永福、李高锋、黄东杰、王建承、汪雪林、赵李明。

本规范责任专家：刘平、郭宝安。

智能IC卡及智能密码钥匙密码应用接口规范

1. **范围**

本规范规定了基于PKI密码体制的智能IC卡及智能密码钥匙密码应用接口，描述了密码应用接口的函数、数据类型、参数的定义和设备的安全要求。

本规范适用于智能IC卡及智能密码钥匙产品的研制、使用和检测。

1. **规范性引用文件**

下列文件中的条款通过本规范的引用而成为本规范的条款。凡是注日期的引用文件，其随后所有的修改单（不包括勘误的内容）或修订版均不适用于本规范，然而，鼓励根据本规范达成协议的各方研究是否可使用这些的最新版本。凡是不注日期的引用文件，其最新版本适用于本规范。

GB/T AAAAA 公钥密码基础设施应用技术体系 SM2算法使用规范

1. **术语和定义**

以下术语和定义适用于本规范。

**3.1**

**应用 application**

包括容器、设备认证密钥和文件的一种结构，具备独立的权限管理。

**3.2**

**容器 container**

特指密钥容器，是一个用于存放非对称密钥对和会话密钥的逻辑对象。

**3.3**

**设备 device**

智能IC卡及智能密码钥匙的统称。

**3.4**

**设备认证 device authentication**

智能IC卡或智能密码钥匙对应用程序的认证。

**3.5**

**设备认证密钥 device authentication key**

用于设备认证的密钥。

**3.6**

**设备标签 label**

设备的别名，可以由用户进行设定并存储于设备内部。

**3.7**

**消息鉴别码 message authentication code (MAC)**

消息鉴别算法的输出。

**3.8**

**管理员PIN Administrator PIN**

管理员的密码，为ASCII字符串。

**3.9**

**用户PIN User PIN**

用户的个人密码，为ASCII字符串。

**3.10**

**智能IC卡 smart card**

含CPU的IC卡，这里指能完成密码功能的IC卡。

**3.11**

**智能密码钥匙smart token**

能完成密码功能和安全存储的终端密码产品，一般采用USB接口。

**3.12**

**SM1算法 SM1 algorithm**

一种国家商用密码分组加密算法，分组长度为128比特，密钥长度为128比特。

**3.13**

**SM2算法 SM2 algorithm**

一种国家商用密码非对称密码算法，密钥长度为256比特。

**3.14**

**SM3算法 SM3 algorithm**

一种国家商用密码密码杂凑算法，输出长度为256比特。

**3.15**

**SMS4算法 SMS4 algorithm**

一种国家商用密码分组加密算法，分组长度为128比特，密钥长度为128比特。

**3.16**

**SSF33算法 SSF33 algorithm**

一种国家商用密码分组加密算法，分组长度为128比特，密钥长度为128比特。

1. **缩略语**

下列缩略语适用于本规范：

API 应用编程接口（Application Programming Interface）

PKI 公钥基础设施（Public Key Infrastructure)

PKCS#1 公钥密码使用标准系列规范中的第1部分，定义RSA公开密钥算法加密和签名机制(the Public-Key Cryptography Standard Part 1)

PKCS#5 公钥密码使用标准系列规范中的第5部分，描述一种利用从口令派生出来的安全密钥加密字符串的方法(the Public-Key Cryptography Standard Part 5)

PIN 个人身份识别码（Personal Identification Number)

MAC 消息鉴别码（Message Authentication Code）

ECC 椭圆曲线密码算法（Elliptic Curve Cryptography）

1. **结构模型**
   1. **层次关系**

智能IC卡及智能密码钥匙密码应用接口位于智能IC卡及智能密码钥匙应用程序与设备之间，如图1所示。



图1接口在应用层次关系中的位置

* 1. **设备的应用结构**

一个设备中存在设备认证密钥和多个应用，应用之间相互独立。设备的逻辑结构如图2所示。



图2 设备逻辑结构

应用由管理员PIN、用户PIN、文件和容器组成，可以存在多个文件和多个容器。

每个应用维护各自的与管理员PIN和用户PIN相关的权限状态。

应用的逻辑结构如图3所示。



图3 应用逻辑结构图

容器中存放加密密钥对、签名密钥对和会话密钥。其中加密密钥对用于保护会话密钥，签名密钥对用于数字签名和验证，会话密钥用于数据加解密和MAC运算。其中，签名密钥对由内部产生，加密密钥对由外部产生并安全导入，会话密钥可由内部产生或者由外部产生并安全导入。

1. **数据类型定义**
   1. **算法标识**
      1. **分组密码算法标识**

分组密码算法标识包含密码算法的类型和加密模式。

分组密码算法标识的编码规则为：从低位到高位，第0位到第7位按位表示分组密码算法工作模式，第8位到第31位按位表示分组密码算法类型，分组密码算法的标识如表1所示。

1. 分组密码算法标识表

|  |  |  |
| --- | --- | --- |
| **标签** | **标识符** | **描述** |
| SGD\_SM1\_ECB | 0x00000101 | SM1算法ECB加密模式 |
| SGD\_SM1\_CBC | 0x00000102 | SM1算法CBC加密模式 |
| SGD\_SM1\_CFB | 0x00000104 | SM1算法CFB加密模式 |
| SGD\_SM1\_OFB | 0x00000108 | SM1算法OFB加密模式 |
| SGD\_SM1\_MAC | 0x00000110 | SM1算法MAC运算 |
| SGD\_SSF33\_ECB | 0x00000201 | SSF33算法ECB加密模式 |
| SGD\_SSF33\_CBC | 0x00000202 | SSF33算法CBC加密模式 |
| SGD\_SSF33\_CFB | 0x00000204 | SSF33算法CFB加密模式 |
| SGD\_SSF33\_OFB | 0x00000208 | SSF33算法OFB加密模式 |
| SGD\_SSF33\_MAC | 0x00000210 | SSF33算法MAC运算 |
| SGD\_SMS4\_ECB | 0x00000401 | SMS4算法ECB加密模式 |
| SGD\_SMS4\_CBC | 0x00000402 | SMS4算法CBC加密模式 |
| SGD\_SMS4\_CFB | 0x00000404 | SMS4算法CFB加密模式 |
| SGD\_SMS4\_OFB | 0x00000408 | SMS4算法OFB加密模式 |
| SGD\_SMS4\_MAC | 0x00000410 | SMS4算法MAC运算 |

* + 1. **非对称密码算法标识**

非对称密码算法标识仅定义了密码算法的类型，在使用非对称算法进行数字签名运算时，可将非对称密码算法标识符与密码杂凑算法标识符进行“或”运算后使用，如“RSA with SHA1”可表示为SGD\_RSA | SGD\_SHA1，即0x00010002，“|”表示“或”运算。

非对称密码算法标识的编码规则为：从低位到高位，第0位到第7位为0，第8位到第15位按位表示非对称密码算法的算法协议，如果所表示的非对称算法没有相应的算法协议则为0，第16位到第31位按位表示非对称密码算法类型，非对称密码算法的标识如表2所示。

1. 非对称密码算法标识表

|  |  |  |
| --- | --- | --- |
| **标签** | **标识符** | **描述** |
| SGD\_RSA | 0x00010000 | RSA算法 |
| SGD\_SM2\_1 | 0x00020100 | 椭圆曲线签名算法 |
| SGD\_SM2\_2 | 0x00020200 | 椭圆曲线密钥交换协议 |
| SGD\_SM2\_3 | 0x00020400 | 椭圆曲线加密算法 |

* + 1. **密码杂凑算法标识**

密码杂凑算法标识符可以在进行密码杂凑运算或计算MAC时应用，也可以与非对称密码算法标识符进行“或”运算后使用，表示签名运算前对数据进行密码杂凑运算的算法类型。

密码杂凑算法标识的编码规则为：从低位到高位，第0位到第7位表示密码杂凑算法，第8位到第31位为0，密码杂凑算法的标识如表3所示。

1. 密码杂凑算法标识表

|  |  |  |
| --- | --- | --- |
| **标签** | **标识符** | **描述** |
| SGD\_SM3 | 0x00000001 | SM3密码杂凑算法 |
| SGD\_SHA1 | 0x00000002 | SHA1密码杂凑算法 |
| SGD\_SHA256 | 0x00000004 | SHA256密码杂凑算法 |

* 1. **基本数据类型**

本规范中的字节数组均为高位字节在前（Big-Endian）方式存储和交换。基本数据类型定义如表4所示：

1. 基本数据类型

|  |  |  |
| --- | --- | --- |
| **类型名称** | **描述** | **定义** |
| INT8 | 有符号8位整数 |  |
| INT16 | 有符号16位整数 |  |
| INT32 | 有符号32位整数 |  |
| UINT8 | 无符号8位整数 |  |
| UINT16 | 无符号16位整数 |  |
| UINT32 | 无符号32位整数 |  |
| BOOL | 布尔类型，取值为TRUE或FALSE |  |
| BYTE | 字节类型，无符号8位整数 | typedef UINT8 BYTE |
| CHAR | 字符类型，无符号8位整数 | typedef UINT8 CHAR |
| SHORT | 短整数，有符号16位 | typedef INT16 SHORT |
| USHORT | 无符号16位整数 | typedef UINT16 USHORT |
| LONG | 长整数，有符号32位整数 | typedef INT32 LONG |
| ULONG | 长整数，无符号32位整数 | typedef UINT32 ULONG |
| UINT | 无符号32位整数 | typedef UINT32 UINT |
| WORD | 字类型，无符号16位整数 | typedef UINT16 WORD |
| DWORD | 双字类型，无符号32位整数 | typedef UINT32 DWORD |
| FLAGS | 标志类型，无符号32位整数 | typedef UINT32 FLAGS |
| LPSTR | 8位字符串指针，按照UTF8格式存储及交换 | typedef CHAR \* LPSTR |
| HANDLE | 句柄，指向任意数据对象的起始地址 | typedef void \* HANDLE |
| DEVHANDLE | 设备句柄 | typedef HANDLE DEVHANDLE |
| HAPPLICATION | 应用句柄 | typedef HANDLE HAPPLICATION |
| HCONTAINER | 容器句柄 | typedef HANDLE HCONTAINER |

* 1. **常量定义**

数据常量标识定义了在规范中用到的常量的取值。

数据常量标识的定义如表5所示。

1. 常量定义

|  |  |  |
| --- | --- | --- |
| **常量名** | **取值** | **描述** |
| TRUE | 0x00000001 | 布尔值为真 |
| FALSE | 0x00000000 | 布尔值为假 |
| DEVAPI | \_\_stdcall | \_\_stdcall函数调用方式 |
| ADMIN\_TYPE | 0 | 管理员PIN类型 |
| USER\_TYPE | 1 | 用户PIN类型 |

* 1. **复合数据类型**
     1. **版本**

（1）类型定义

typedef struct Struct\_Version{

BYTE major;

BYTE minor;

}VERSION;

（2）数据项描述参见表6：

1. 版本定义

|  |  |  |  |
| --- | --- | --- | --- |
| **数据项** | **类型** | **意义** | **备注** |
| major | BYTE | 主版本号 | 主版本号和次版本号以“.”分隔，例如 Version 1.0，主版本号为1，次版本号为0；Version 2.10，主版本号为2，次版本号为10。 |
| minor | BYTE | 次版本号 |

* + 1. **设备信息**

（1）类型定义

typedef struct Struct\_DEVINFO{

VERSION Version;

CHAR Manufacturer[64];

CHAR Issuer[64];

CHAR Label[32];

CHAR SerialNumber[32];

VERSION HWVersion;

VERSION FirmwareVersion;

ULONG AlgSymCap;

ULONG AlgAsymCap;

ULONG AlgHashCap;

ULONG DevAuthAlgId;

ULONG TotalSpace;

ULONG FreeSpace;

BYTE Reserved[64];

}DEVINFO,\*PDEVINFO;

（2）数据项描述参见表7：

1. 设备信息描述

|  |  |  |  |
| --- | --- | --- | --- |
| **数据项** | **类型** | **意义** | **备注** |
| Version | VERSION | 版本号 | 数据结构版本号，本结构的版本号为1.0 |
| Manufacturer | CHAR数组 | 设备厂商信息 | 以 ‘\0’为结束符的ASCII字符串 |
| Issuer | CHAR数组 | 发行厂商信息 | 以 ‘\0’为结束符的ASCII字符串 |
| Label | CHAR数组 | 设备标签 | 以 ‘\0’为结束符的ASCII字符串 |
| SerialNumber | CHAR数组 | 序列号 | 以 ‘\0’为结束符的ASCII字符串 |
| HWVersion | VERSION | 设备硬件版本 |  |
| FirmwareVersion | VERSION | 设备本身固件版本 |  |
| AlgSymCap | ULONG | 分组密码算法标识 |  |
| AlgAsymCap | ULONG | 非对称密码算法标识 |  |
| AlgHashCap | ULONG | 密码杂凑算法标识 |  |
| DevAuthAlgId | ULONG | 设备认证使用的分组密码算法标识 |  |
| TotalSpace | ULONG | 设备总空间大小 |  |
| FreeSpace | ULONG | 用户可用空间大小 |  |
| Reserved | BYTE | 保留扩展 |  |

* + 1. **RSA公钥数据结构**

（1）类型定义

typedef struct Struct\_RSAPUBLICKEYBLOB{

ULONG AlgID;

ULONG BitLen;

BYTE Modulus[MAX\_RSA\_MODULUS\_LEN];

BYTE PublicExponent[MAX\_RSA\_EXPONENT\_LEN];

}RSAPUBLICKEYBLOB, \*PRSAPUBLICKEYBLOB;

MAX\_RSA\_MODULUS\_LEN为算法模数的最大长度；

MAX\_RSA\_EXPONENT\_LEN为算法指数的最大长度。

（2）数据项描述参见表8：

1. RSA公钥数据结构

|  |  |  |  |
| --- | --- | --- | --- |
| **数据项** | **类型** | **意义** | **备注** |
| AlgID | ULONG | 算法标识号 |  |
| BitLen | ULONG | 模数的实际位长度 | 必须是8的倍数 |
| Modulus | BYTE数组 | 模数n = p \* q | 实际长度为BitLen/8字节  #define MAX\_RSA\_MODULUS\_LEN 256  #define MAX\_RSA\_EXPONENT\_LEN 4 |
| PublicExponent | BYTE数组 | 公开密钥e | 一般为00010001 |

* + 1. **RSA私钥数据结构**

（1）类型定义

typedef struct Struct\_RSAPRIVATEKEYBLOB{

ULONG AlgID;

ULONG BitLen;

BYTE Modulus[MAX\_RSA\_MODULUS\_LEN];

BYTE PublicExponent[MAX\_RSA\_EXPONENT\_LEN];

BYTE PrivateExponent[MAX\_RSA\_MODULUS\_LEN];

BYTE Prime1[MAX\_RSA\_MODULUS\_LEN/2];

BYTE Prime2[MAX\_RSA\_MODULUS\_LEN/2];

BYTE Prime1Exponent[MAX\_RSA\_MODULUS\_LEN/2];

BYTE Prime2Exponent[MAX\_RSA\_MODULUS\_LEN/2];

BYTE Coefficient[MAX\_RSA\_MODULUS\_LEN/2];

}RSAPRIVATEKEYBLOB, \*PRSAPRIVATEKEYBLOB;

MAX\_RSA\_MODULUS\_LEN为RSA算法模数的最大长度；

（2）数据项描述参见表9：

1. RSA私钥数据结构

|  |  |  |  |
| --- | --- | --- | --- |
| **数据项** | **类型** | **意义** | **备注** |
| AlgID | ULONG | 算法标识号 |  |
| BitLen | ULONG | 模数的实际位长度 | 必须是8的倍数 |
| Modulus | BYTE数组 | 模数n = p \* q | 实际长度为BitLen/8字节 |
| PublicExponent | BYTE数组 | 公开密钥e | 一般为00010001 |
| PrivateExponent | BYTE数组 | 私有密钥d | 实际长度为BitLen/8字节 |
| Prime1 | BYTE数组 | 素数p | 实际长度为BitLen/16字节 |
| Prime2 | BYTE数组 | 素数q | 实际长度为BitLen/16字节 |
| Prime1Exponent | BYTE数组 | d mod (p-1)的值 | 实际长度为BitLen/16字节 |
| Prime2Exponent | BYTE数组 | d mod (q -1)的值 | 实际长度为BitLen/16字节 |
| Coefficient | BYTE数组 | q模p的乘法逆元 | 实际长度为BitLen/16字节 |

* + 1. **ECC公钥数据结构**

（1）类型定义

typedef struct Struct\_ECCPUBLICKEYBLOB{

ULONG BitLen;

BYTE XCoordinate[ECC\_MAX\_XCOORDINATE\_BITS\_LEN/8];

BYTE YCoordinate[ECC\_MAX\_YCOORDINATE\_BITS\_LEN/8];

}ECCPUBLICKEYBLOB, \*PECCPUBLICKEYBLOB;

ECC\_MAX\_XCOORDINATE\_LEN为ECC算法X坐标的最大长度；

ECC\_MAX\_YCOORDINATE\_LEN为ECC算法Y坐标的最大长度。

（2）数据项描述参见表10：

1. ECC公钥数据结构

|  |  |  |  |
| --- | --- | --- | --- |
| **数据项** | **类型** | **意义** | **备注** |
| BitLen | ULONG | 模数的实际位长度 | 必须是8的倍数 |
| XCoordinate | BYTE数组 | 曲线上点的X坐标 | 有限域上的整数  #define ECC\_MAX\_XCOORDINATE\_BITS\_LEN 512 |
| YCoordinate | BYTE数组 | 曲线上点的Y坐标 | 有限域上的整数  #define ECC\_MAX\_YCOORDINATE\_BITS\_LEN 512 |

* + 1. **ECC私钥数据结构**

（1）类型定义

typedef struct Struct\_ECCPRIVATEKEYBLOB{

ULONG BitLen;

BYTE PrivateKey[ECC\_MAX\_MODULUS\_BITS\_LEN/8];

}ECCPRIVATEKEYBLOB, \*PECCPRIVATEKEYBLOB;

ECC\_MAX\_MODULUS\_BITS\_LEN为ECC算法模数的最大长度。

（2）数据项描述参见表11：

1. ECC私钥数据结构

|  |  |  |  |
| --- | --- | --- | --- |
| **数据项** | **类型** | **意义** | **备注** |
| BitLen | ULONG | 模数的实际位长度 | 必须是8的倍数 |
| PrivateKey | BYTE数组 | 私有密钥 | 有限域上的整数  #define ECC\_MAX\_MODULUS\_BITS\_LEN 512 |

* + 1. **ECC密文数据结构**

1. 类型定义

typedef struct Struct\_ECCCIPHERBLOB{

BYTE XCoordinate[ECC\_MAX\_XCOORDINATE\_BITS\_LEN/8];

BYTE YCoordinate[ECC\_MAX\_XCOORDINATE\_BITS\_LEN/8];

BYTE HASH[32];

ULONG CipherLen;

BYTE Cipher[1];

} ECCCIPHERBLOB, \*PECCCIPHERBLOB;

（2）数据项描述参见表12：

1. ECC密文数据结构

|  |  |  |  |
| --- | --- | --- | --- |
| **数据项** | **类型** | **意义** | **备注** |
| XCoordinate | BYTE数组 | 与y组成椭圆曲线上的点（x，y） |  |
| YCoordinate | BYTE数组 | 与x组成椭圆曲线上的点（x，y） |  |
| HASH | BYTE数组 | 明文的杂凑值 |  |
| CipherLen | ULONG | 密文数据长度 |  |
| Cipher | BYTE数组 | 密文数据 | 实际长度为CipherLen |

* + 1. **ECC签名数据结构**

（1）类型定义

typedef struct Struct\_ECCSIGNATUREBLOB{

BYTE r[ECC\_MAX\_XCOORDINATE\_BITS\_LEN/8];

BYTE s[ECC\_MAX\_XCOORDINATE\_BITS\_LEN/8];

} ECCSIGNATUREBLOB, \*PECCSIGNATUREBLOB;

ECC\_MAX\_MODULUS\_BITS\_LEN为ECC算法模数的最大长度；

（2）数据项描述参见表13：

1. ECC签名数据结构

|  |  |  |  |
| --- | --- | --- | --- |
| **数据项** | **类型** | **意义** | **备注** |
| r | BYTE数组 | 签名结果的r部分 |  |
| s | BYTE数组 | 签名结果的s部分 |  |

* + 1. **分组密码参数**

（1）类型定义

typedef struct Struct\_BLOCKCIPHERPARAM{

BYTE IV[MAX\_IV\_LEN];

ULONG IVLen;

ULONG PaddingType;

ULONG FeedBitLen;

} BLOCKCIPHERPARAM, \*PBLOCKCIPHERPARAM;

（2）数据项描述参见表14：

1. 分组密码参数

|  |  |  |  |
| --- | --- | --- | --- |
| **数据项** | **类型** | **意义** | **备注** |
| IV | BYTE数组 | 初始向量，MAX\_IV\_LEN为初始化向量的最大长度  #define MAX\_IV\_LEN 32 |  |
| IVLen | ULONG | 初始向量实际长度（按字节计算） |  |
| PaddingType | ULONG | 填充方式，0表示不填充，1表示按照PKCS#5方式进行填充 |  |
| FeedBitLen | ULONG | 反馈值的位长度（按位计算） | 只针对OFB、CFB模式 |

* + 1. **ECC加密密钥对保护结构**

（1）类型定义

typedef struct SKF\_ENVELOPEDKEYBLOB{

ULONG Version; // 当前版本为 1

ULONG ulSymmAlgID; // 对称算法标识，限定ECB模式

ULONG ulBits; // 加密密钥对的密钥位长度

BYTE cbEncryptedPriKey[64]; // 加密密钥对私钥的密文

ECCPUBLICKEYBLOB PubKey; // 加密密钥对的公钥

ECCCIPHERBLOB ECCCipherBlob; // 用保护公钥加密的对称密钥密文。

}ENVELOPEDKEYBLOB, \*PENVELOPEDKEYBLOB;

（2）数据项描述参见表15：

1. 分组密码参数

|  |  |  |  |
| --- | --- | --- | --- |
| **数据项** | **类型** | **意义** | **备注** |
| Version | ULONG | 版本号，本版本为1 |  |
| ulSymmAlgID | ULONG | 对称算法标识 | 必须为ECB模式 |
| ulBits | ULONG | 加密密钥对的密钥位长 |  |
| cbEncryptedPrivKey | BYTE数组 | 对称算法加密的加密私钥,加密私钥的原文为ECCPRIVATEKEYBLOB结构中的PrivateKey。 | 其有效长度为原文的（ulBits + 7）/8 |
| PubKey | ECCPUBLICKEYBLOB | 加密密钥对的公钥 |  |
| ECCCipherBlob | ECCCIPHERBLOB | 用保护公钥加密过的对称密钥密文 |  |

* + 1. **文件属性**

（1）类型定义

typedef struct Struct\_FILEATTRIBUTE{

CHAR FileName[32];

ULONG FileSize;

ULONG ReadRights;

ULONG WriteRights;

} FILEATTRIBUTE, \*PFILEATTRIBUTE;

（2）数据项描述参见表16：

1. 文件属性

|  |  |  |  |
| --- | --- | --- | --- |
| **数据项** | **类型** | **意义** | **备注** |
| FileName | CHAR数组 | 文件名 | 以‘\0’结束的ASCII字符串，最大长度为32 |
| FileSize | ULONG | 文件大小 | 创建文件时定义的文件大小 |
| ReadRights | ULONG | 读取权限 | 读取文件需要的权限 |
| WriteRights | ULONG | 写入权限 | 写入文件需要的权限 |

* + 1. **权限类型**

权限类型的定义参见表17：

1. 权限类型

|  |  |  |
| --- | --- | --- |
| **权限类型** | **值** | **说明** |
| SECURE\_NEVER\_ACCOUNT | 0x00000000 | 不允许 |
| SECURE\_ADM\_ACCOUNT | 0x00000001 | 管理员权限 |
| SECURE\_USER\_ACCOUNT | 0x00000010 | 用户权限 |
| SECURE\_ANYONE\_ACCOUNT | 0x000000FF | 任何人 |

* + 1. **设备状态**

设备状态的定义参见表18：

1. 设备状态

|  |  |  |
| --- | --- | --- |
| **设备状态** | **值** | **说明** |
| DEV\_ABSENT\_STATE | 0x00000000 | 设备不存在 |
| DEV\_PRESENT\_STATE | 0x00000001 | 设备存在 |
| DEV\_UNKNOW\_STATE | 0x00000002 | 设备状态未知 |

1. **接口函数**
   1. **设备管理**
      1. **概述**

设备管理主要完成设备的插拔事件处理、枚举设备、连接设备、断开连接、获取设备状态、设置设备标签、获取设备信息、锁定设备、解锁设备和设备命令传输等操作。设备管理系列函数如表19所示：

1. 设备管理系列函数

|  |  |
| --- | --- |
| 函数名称 | 功能 |
| SKF\_WaitForDevEvent | 等待设备插拔事件 |
| SKF\_CancelWaitForDevEvent | 取消等待设备插拔事件 |
| SKF\_EnumDev | 枚举设备 |
| SKF\_ConnectDev | 连接设备 |
| SKF\_DisconnectDev | 断开连接 |
| SKF\_GetDevState | 获取设备状态 |
| SKF\_SetLabel | 设置设备标签 |
| SKF\_GetDevInfo | 获取设备信息 |
| SKF\_LockDev | 锁定设备 |
| SKF\_UnlockDev | 解锁设备 |
| SKF\_Transmit | 设备命令传输 |

* + 1. **等待设备插拔事件**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 函数原型 | | ULONG DEVAPI SKF\_WaitForDevEvent(LPSTR szDevName,ULONG \*pulDevNameLen, ULONG \*pulEvent) | | |
| 功能描述 | | 该函数等待设备插入或者拔除事件。szDevName返回发生事件的设备名称。 | | |
| 参数 | | szDevName | [OUT] 发生事件的设备名称。 | |
| pulDevNameLen | [IN/OUT] 输入/输出参数，当输入时表示缓冲区长度，输出时表示设备名称的有效长度,长度包含字符串结束符。 | |
| pulEvent | [OUT]事件类型。1表示插入，2表示拔出。 | |
| 返回值 | SAR\_OK： 成功。  其他： 错误码。 | | |

* + 1. **取消等待设备插拔事件**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 函数原型 | | ULONG DEVAPI SKF\_CancelWaitForDevEvent() | | |
| 功能描述 | | 该函数取消等待设备插入或者拔除事件。 | | |
| 参数 | |  |  | |
| 返回值 | SAR\_OK： 成功。  其他： 错误码。 | | |
| 备注 | 使本进程正在执行的SKF\_WaitForDevEvent函数立即返回。 | | |

* + 1. **枚举设备**

|  |  |  |  |
| --- | --- | --- | --- |
| 函数原型 | | ULONG DEVAPI SKF\_EnumDev(BOOL bPresent, LPSTR szNameList, ULONG \*pulSize) | |
| 功能描述 | | 获得当前系统中的设备列表。 | |
| 参数 | | bPresent | [IN] 为TRUE表示取当前设备状态为存在的设备列表。为FALSE表示取当前驱动支持的设备列表。 |
| szNameList | [OUT] 设备名称列表。如果该参数为NULL，将由pulSize返回所需要的内存空间大小。每个设备的名称以单个‘\0’结束，以双‘\0’表示列表的结束。 |
| pulSize | [IN，OUT] 输入时表示设备名称列表的缓冲区长度，输出时表示szNameList所占用的空间大小。 |
| 返回值 | SAR\_OK： 成功。  其他： 错误码。 | | |

* + 1. **连接设备**

|  |  |  |  |
| --- | --- | --- | --- |
| 函数原型 | | ULONG DEVAPI SKF\_ConnectDev (LPSTR szName, DEVHANDLE \*phDev) | |
| 功能描述 | | 通过设备名称连接设备，返回设备的句柄。 | |
| 参数 | | szName | [IN] 设备名称。 |
| phDev | [OUT] 返回设备操作句柄。 |
| 返回值 | SAR\_OK： 成功。  其他： 错误码。 | | |

* + 1. **断开连接**

|  |  |  |  |
| --- | --- | --- | --- |
| 函数原型 | | ULONG DEVAPI SKF\_DisConnectDev (DEVHANDLE hDev) | |
| 功能描述 | | 断开一个已经连接的设备，并释放句柄。 | |
| 参数 | | hDev | [IN] 连接设备时返回的设备句柄。 |
| 返回值 | SAR\_OK： 成功。  其他： 错误码。 | | |
| 备注 | | 如果该设备已被锁定，函数应首先解锁该设备。断开连接操作并不影响设备的权限状态。 | |

* + 1. **获取设备状态**

|  |  |  |  |
| --- | --- | --- | --- |
| 函数原型 | | ULONG DEVAPI SKF\_GetDevState(LPSTR szDevName*,* ULONG \*pulDevState) | |
| 功能描述 | | 获取设备是否存在的状态。 | |
| 参数 | | szDevName | [IN] 设备名称。 |
| pulDevState | [OUT] 返回设备状态。 |
| 返回值 | SAR\_OK： 成功。  其他： 错误码。 | | |

* + 1. **设置设备标签**

|  |  |  |  |
| --- | --- | --- | --- |
| 函数原型 | | ULONG DEVAPI SKF\_SetLabel (DEVHANDLE hDev, LPSTR szLabel) | |
| 功能描述 | | 设置设备标签。 | |
| 参数 | | hDev | [IN] 连接设备时返回的设备句柄。 |
| szLabel | [IN] 设备标签字符串。该字符串应小于32字节。 |
| 返回值 | SAR\_OK： 成功。  其他： 错误码。 | | |

* + 1. **获取设备信息**

|  |  |  |  |
| --- | --- | --- | --- |
| 函数原型 | | ULONG DEVAPI SKF\_GetDevInfo (DEVHANDLE hDev, DEVINFO \*pDevInfo) | |
| 功能描述 | | 获取设备的一些特征信息，包括设备标签、厂商信息、支持的算法等。 | |
| 参数 | | hDev | [IN] 连接设备时返回的设备句柄。 |
| pDevInfo | [OUT] 返回设备信息。 |
| 返回值 | SAR\_OK： 成功。  其他： 错误码。 | | |

* + 1. **锁定设备**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 函数原型 | | ULONG DEVAPI SKF\_LockDev (DEVHANDLE hDev, ULONG ulTimeOut) | | |
| 功能描述 | | 获得设备的独占使用权。 | | |
| 参数 | | hDev | [IN] 连接设备时返回的设备句柄。 | |
| ulTimeOut | [IN] 超时时间，单位为毫秒。如果为0xFFFFFFFF表示无限等待。 | |
| 返回值 | SAR\_OK： 成功。  其他： 错误码。 | | |

* + 1. **解锁设备**

|  |  |  |  |
| --- | --- | --- | --- |
| 函数原型 | | ULONG DEVAPI SKF\_UnlockDev (DEVHANDLE hDev) | |
| 功能描述 | | 释放对设备的独占使用权。 | |
| 参数 | | hDev | [IN] 连接设备时返回的设备句柄。 |
| 返回值 | SAR\_OK： 成功。  其他： 错误码。 | | |

* + 1. **设备命令传输**

|  |  |  |
| --- | --- | --- |
| 函数原型 | ULONG DEVAPI SKF\_Transmit(DEVHANDLE hDev, BYTE\* pbCommand, ULONG ulCommandLen,BYTE\* pbData, ULONG\* pulDataLen) | |
| 功能描述 | 将命令直接发送给设备，并返回结果。 | |
| 参数 | hDev | [IN] 设备句柄。 |
|  | pbCommand | [IN] 设备命令。 |
|  | ulCommandLen | [IN] 命令长度。 |
|  | pbData | [OUT] 返回结果数据。 |
|  | pulDataLen | [IN，OUT] 输入时表示结果数据缓冲区长度，输出时表示结果数据实际长度。 |
| 返回值 | SAR\_OK： 成功。  其他： 错误码。 | |

* 1. **访问控制**
     1. **概述**

访问控制主要完成设备认证、PIN码管理和安全状态管理等操作。访问控制系列函数如表20所示：

1. 访问控制系列函数

|  |  |
| --- | --- |
| 函数名称 | 功能 |
| SKF\_ChangeDevAuthKey | 修改设备认证密钥 |
| SKF\_DevAuth | 设备认证 |
| SKF\_ChangePIN | 修改PIN |
| SKF\_GetPINInfo | 获得PIN码信息 |
| SKF\_VerifyPIN | 校验PIN |
| SKF\_UnblockPIN | 解锁PIN |
| SKF\_ClearSecueState | 清除应用安全状态 |

* + 1. **修改设备认证密钥**

|  |  |  |  |
| --- | --- | --- | --- |
| 函数原型 | | ULONG DEVAPI SKF\_ChangeDevAuthKey (DEVHANDLE hDev, BYTE \*pbKeyValue， ULONG ulKeyLen) | |
| 功能描述 | | 更改设备认证密钥。 | |
| 参数 | | hDev | [IN] 连接时返回的设备句柄。 |
| pbKeyValue | [IN] 密钥值。 |
| ulKeyLen | [IN] 密钥长度。 |
| 返回值 | SAR\_OK： 成功。  其他： 错误码。 | | |
| 备注 | | 权限要求：设备认证成功后才能使用。 | |

* + 1. **设备认证**

|  |  |  |  |
| --- | --- | --- | --- |
| 函数原型 | | ULONG DEVAPI SKF\_DevAuth (DEVHANDLE hDev, BYTE \*pbAuthData，ULONG ulLen) | |
| 功能描述 | | 设备认证是设备对应用程序的认证。认证过程参见8.2.3。 | |
| 参数 | | hDev | [IN] 连接时返回的设备句柄。 |
| pbAuthData | [IN] 认证数据。 |
| ulLen | [IN] 认证数据的长度。 |
| 返回值 | SAR\_OK： 成功。  其他： 错误码。 | | |

* + 1. **修改PIN**

|  |  |  |  |
| --- | --- | --- | --- |
| 函数原型 | | ULONG DEVAPI SKF\_ChangePIN (HAPPLICATION hApplication, ULONG ulPINType, LPSTR szOldPin, LPSTR szNewPin, ULONG \*pulRetryCount) | |
| 功能描述 | | 调用该函数可以修改Administrator PIN和User PIN的值。  如果原PIN码错误导致验证失败，该函数会返回相应PIN码的剩余重试次数，当剩余次数为0时，表示PIN已经被锁死。 | |
| 参数 | | hApplication | [IN] 应用句柄。 |
| ulPINType | [IN] PIN类型，可为ADMIN\_TYPE或USER\_TYPE。 |
| szOldPin | [IN] 原PIN值。 |
| szNewPin | [IN] 新PIN值。 |
| pulRetryCount | [OUT] 出错后重试次数。 |
| 返回值 | SAR\_OK： 成功。  其他： 错误码。 | | |

* + 1. **获取PIN信息**

|  |  |  |  |
| --- | --- | --- | --- |
| 函数原型 | ULONG DEVAPI SKF\_GetPINInfo(HAPPLICATION hApplication, ULONG ulPINType, ULONG \*pulMaxRetryCount, ULONG \*pulRemainRetryCount, BOOL \*pbDefaultPin) | | |
| 功能描述 | 获取PIN码信息，包括最大重试次数、当前剩余重试次数，以及当前PIN码是否为出厂默认PIN码。 | | |
| 参数 | hApplication | | [IN] 应用句柄。 |
| ulPINType | | [IN] PIN类型。 |
| pulMaxRetryCount | | [OUT] 最大重试次数。 |
| pulRemainRetryCount | | [OUT] 当前剩余重试次数，当为0时表示已锁死。 |
| pbDefaultPin | | [OUT] 是否为出厂默认PIN码。 |
| 返回值 | | SAR\_OK： 成功。  其他： 错误码。 | |

* + 1. **校验PIN**

|  |  |  |
| --- | --- | --- |
| 函数原型 | ULONG DEVAPI SKF\_VerifyPIN (HAPPLICATION hApplication, ULONG ulPINType, LPSTR szPIN, ULONG \*pulRetryCount) | |
| 功能描述 | 校验PIN码。校验成功后，会获得相应的权限，如果PIN码错误，会返回PIN码的重试次数，当重试次数为0时表示PIN码已经锁死。 | |
| 参数 | hApplication | [IN] 应用句柄。 |
| ulPINType | [IN] PIN类型。 |
| szPIN | [IN] PIN值。 |
| pulRetryCount | [OUT] 出错后返回的重试次数。 |
| 返回值 | SAR\_OK： 成功。  其他： 错误码。 | |

* + 1. **解锁PIN**

|  |  |  |  |
| --- | --- | --- | --- |
| 函数原型 | | ULONG DEVAPI SKF\_UnblockPIN (HAPPLICATION hApplication, LPSTR szAdminPIN, LPSTR szNewUserPIN, ULONG \*pulRetryCount) | |
| 功能描述 | | 当用户的PIN码锁死后，通过调用该函数来解锁用户PIN码。  解锁后，用户PIN码被设置成新值，用户PIN码的重试次数也恢复到原值。 | |
| 参数 | | hApplication | [IN] 应用句柄。 |
| szAdminPIN | [IN] 管理员PIN码。 |
| szNewUserPIN | [IN] 新的用户PIN码。 |
| pulRetryCount | [OUT] 管理员PIN码错误时，返回剩余重试次数。 |
| 返回值 | SAR\_OK： 成功。  其他： 错误码。 | | |
| 备注 | | 验证完管理员PIN才能够解锁用户PIN码，如果输入的Administrator PIN不正确或者已经锁死，会调用失败，并返回Administrator PIN的重试次数。 | |

* + 1. **清除应用安全状态**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 函数原型 | | ULONG DEVAPI SKF\_ClearSecureState (HAPPLICATION hApplication) | | |
| 功能描述 | | | 清除应用当前的安全状态。 | |
| 参数 | | | hApplication | [IN] 应用句柄。 |
| 返回值 | SAR\_OK： 成功。  其他： 错误码。 | | | |

* 1. **应用管理**
     1. **概述**

应用管理主要完成应用的创建、枚举、删除、打开、关闭等操作。应用管理系列函数如表21所示：

1. 应用管理系列函数

|  |  |
| --- | --- |
| 函数名称 | 功能 |
| SKF\_CreateApplication | 创建应用 |
| SKF\_EnumApplication | 枚举应用 |
| SKF\_DeleteApplication | 删除应用 |
| SKF\_OpenApplication | 打开应用 |
| SKF\_CloseApplication | 关闭应用 |

* + 1. **创建应用**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 函数原型 | | ULONG DEVAPI SKF\_CreateApplication(DEVHANDLE hDev, LPSTR szAppName, LPSTR szAdminPin, DWORD dwAdminPinRetryCount,LPSTR szUserPin, DWORD dwUserPinRetryCount,DWORD dwCreateFileRights, HAPPLICATION \*phApplication) | | |
| 功能描述 | | 创建一个应用。 | | |
| 参数 | | hDev | [IN] 连接设备时返回的设备句柄。 | |
| szAppName | [IN] 应用名称。 | |
| szAdminPin | [IN] 管理员PIN。 | |
| dwAdminPinRetryCount | [IN] 管理员PIN最大重试次数。 | |
| szUserPin | [IN] 用户PIN。 | |
| dwUserPinRetryCount | [IN] 用户PIN最大重试次数。 | |
| dwCreateFileRights | [IN] 在该应用下创建文件和容器的权限，参见6.4.9权限类型。为各种权限的或值。 | |
| phApplication | [OUT] 应用的句柄。 | |
| 返回值 | SAR\_OK： 成功。  其他： 错误码。 | | |
| 备注 | | 权限要求：需要设备权限。 | | |

* + 1. **枚举应用**

|  |  |  |  |
| --- | --- | --- | --- |
| 函数原型 | | ULONG DEVAPI SKF\_EnumApplication(DEVHANDLE hDev, LPSTR szAppName,ULONG \*pulSize) | |
| 功能描述 | | 枚举设备中存在的所有应用。 | |
| 参数 | | hDev | [IN] 连接设备时返回的设备句柄。 |
| szAppName | [OUT] 返回应用名称列表, 如果该参数为空，将由pulSize返回所需要的内存空间大小。每个应用的名称以单个‘\0’结束，以双‘\0’表示列表的结束。 |
| pulSize | [IN，OUT] 输入时表示应用名称的缓冲区长度，输出时返回szAppName所占用的空间大小。 |
| 返回值 | SAR\_OK： 成功。  其他： 错误码。 | | |

* + 1. **删除应用**

|  |  |  |  |
| --- | --- | --- | --- |
| 函数原型 | | ULONG DEVAPI SKF\_DeleteApplication(DEVHANDLE hDev, LPSTR szAppName) | |
| 功能描述 | | 删除指定的应用。 | |
| 参数 | | hDev | [IN] 连接设备时返回的设备句柄。 |
| szAppName | [IN] 应用名称。 |
| 返回值 | SAR\_OK： 成功。  其他： 错误码。 | | |
| 备注 | | 权限要求：需要设备权限。 | |

* + 1. **打开应用**

|  |  |  |  |
| --- | --- | --- | --- |
| 函数原型 | | ULONG DEVAPI SKF\_OpenApplication(DEVHANDLE hDev, LPSTR szAppName, HAPPLICATION \*phApplication) | |
| 功能描述 | | 打开指定的应用。 | |
| 参数 | | hDev | [IN] 连接设备时返回的设备句柄。 |
| szAppName | [IN] 应用名称。 |
| phApplication | [OUT] 应用的句柄。 |
| 返回值 | SAR\_OK： 成功。  其他： 错误码。 | | |

* + 1. **关闭应用**

|  |  |  |  |
| --- | --- | --- | --- |
| 函数原型 | | ULONG DEVAPI SKF\_CloseApplication(HAPPLICATION hApplication) | |
| 功能描述 | | 关闭应用并释放应用句柄。 | |
| 参数 | | hApplication | [IN]应用句柄。 |
| 返回值 | SAR\_OK： 成功。  其他： 错误码。 | | | |
| 备注 | | 此函数不影响应用安全状态。 | |

* 1. **文件管理**
     1. **概述**

文件管理函数用以满足用户扩展开发的需要，包括创建文件、删除文件、枚举文件、获取文件信息、文件读写等操作。文件管理系列函数如表22所示：

1. 文件管理系列函数

|  |  |
| --- | --- |
| 函数名称 | 功能 |
| SKF\_CreateFile | 创建文件 |
| SKF\_DeleteFile | 删除文件 |
| SKF\_EnumFiles | 枚举文件 |
| SKF\_GetFileInfo | 获取文件信息 |
| SKF\_ReadFile | 读文件 |
| SKF\_WriteFile | 写文件 |

* + 1. **创建文件**

|  |  |  |  |
| --- | --- | --- | --- |
| 函数原型 | | ULONG DEVAPI SKF\_CreateFile (HAPPLICATION hApplication, LPSTR szFileName, ULONG ulFileSize, ULONG ulReadRights，ULONG ulWriteRights) | |
| 功能描述 | | 创建文件时要指定文件的名称，大小，以及文件的读写权限。 | |
| 参数 | | hApplication | [IN] 应用句柄。 |
| szFileName | [IN] 文件名称，长度不得大于32个字节。 |
| ulFileSize | [IN] 文件大小。 |
| ulReadRights | [IN] 文件读权限，参见6.4.9 权限类型。可为各种权限的或值。 |
| ulWriteRights | [IN] 文件写权限，参见6.4.9权限类型。可为各种权限的或值。 |
| 返回值 | SAR\_OK： 成功。  其他： 错误码。 | | | |
| 备注 | | 创建文件需要应用指定的创建文件权限。 | |

* + 1. **删除文件**

|  |  |  |  |
| --- | --- | --- | --- |
| 函数原型 | | ULONG DEVAPI SKF\_DeleteFile (HAPPLICATION hApplication, LPSTR szFileName) | |
| 功能描述 | | 删除指定文件：  文件删除后，文件中写入的所有信息将丢失。  文件在设备中的占用的空间将被释放。 | |
| 参数 | | hApplication | [IN] 要删除文件所在的应用句柄。 |
| szFileName | [IN] 要删除文件的名称。 |
| 返回值 | SAR\_OK： 成功。  其他： 错误码。 | | | |
| 备注 | | 权限要求：删除一个文件应具有对该文件的创建权限。 | |

* + 1. **枚举文件**

|  |  |  |  |
| --- | --- | --- | --- |
| 函数原型 | | ULONG DEVAPI SKF\_EnumFiles (HAPPLICATION hApplication, LPSTR szFileList, ULONG \*pulSize) | |
| 功能描述 | | 枚举一个应用下存在的所有文件。 | |
| 参数 | | hApplication | [IN] 应用句柄。 |
| szFileList | [OUT] 返回文件名称列表，该参数为空，由pulSize返回文件信息所需要的空间大小。每个文件名称以单个‘\0’结束，以双‘\0’表示列表的结束。 |
| pulSize | [IN，OUT] 输入时表示数据缓冲区的大小，输出时表示实际文件名称列表的长度。 |
| 返回值 | SAR\_OK： 成功。  其他： 错误码。 | | | |

* + 1. **获取文件属性**

|  |  |  |  |
| --- | --- | --- | --- |
| 函数原型 | ULONG DEVAPI SKF\_GetFileInfo (HAPPLICATION hApplication, LPSTR szFileName, FILEATTRIBUTE \*pFileInfo) | | |
| 功能描述 | 获取文件信息：  获取应用文件的属性信息，例如文件的大小、权限等。 | | |
| 参数 | hApplication | | [IN] 文件所在应用的句柄。 |
| szFileName | | [IN] 文件名称。 |
| pFileInfo | | [OUT] 文件信息，指向文件属性结构的指针。 |
| 返回值 | | SAR\_OK： 成功。  其他： 错误码。 | | |

* + 1. **读文件**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 函数原型 | | ULONG DEVAPI SKF\_ReadFile (HAPPLICATION hApplication， LPSTR szFileName, ULONG ulOffset, ULONG ulSize, BYTE \* pbOutData, ULONG \*pulOutLen) | | |
| 功能描述 | | | 读取文件内容。 | |
| 参数 | | | hApplication | [IN] 应用句柄。 |
| szFileName | [IN] 文件名。 |
| ulOffset | [IN] 文件读取偏移位置。 |
| ulSize | [IN] 要读取的长度。 |
| pbOutData | [OUT] 返回数据的缓冲区。 |
| pulOutLen | [IN，OUT]输入时表示给出的缓冲区大小；输出时表示实际读取返回的数据大小。 |
| 返回值 | SAR\_OK： 成功。  其他： 错误码。 | | | | |
| 备注 | | | 权限要求：须具备对该文件的读权限。 | |

* + 1. **写文件**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 函数原型 | | ULONG DEVAPI SKF\_WriteFile (HAPPLICATION hApplication, LPSTR szFileName, ULONG ulOffset, BYTE \*pbData, ULONG ulSize) | | |
| 功能描述 | | | 写数据到文件中。 | |
| 参数 | | | hApplication | [IN] 应用句柄。 |
| szFileName | [IN] 文件名。 |
| ulOffset | [IN] 写入文件的偏移量。 |
| pbData | [IN] 写入数据缓冲区。 |
| ulSize | [IN] 写入数据的大小。 |
| 返回值 | SAR\_OK： 成功。  其他： 错误码。 | | | | |
| 备注 | | | 权限要求：须具备该文件的写权限。 | |

* 1. **容器管理**
     1. **概述**

本规范提供的应用管理用于满足各种不同应用的管理，包括创建、删除、枚举、打开和关闭容器的操作。容器管理系列函数如表23所示：

1. 容器管理系列函数

|  |  |
| --- | --- |
| 函数名称 | 功能 |
| SKF\_CreateContainer | 创建容器 |
| SKF\_DeleteContainer | 删除容器 |
| SKF\_EnumContainer | 枚举容器 |
| SKF\_OpenContainer | 打开容器 |
| SKF\_CloseContainer | 关闭容器 |
| SKF\_GetContainerType | 获得容器类型 |
| SKF\_ImportCertificate | 导入数字证书 |
| SKF\_ExportCertificate | 导出数字证书 |

* + 1. **创建容器**

|  |  |  |
| --- | --- | --- |
| 函数原型 | ULONG DEVAPI SKF\_CreateContainer (HAPPLICATION hApplication, LPSTR szContainerName, HCONTAINER \*phContainer) | |
| 功能描述 | 在应用下建立指定名称的容器并返回容器句柄。 | |
| 参数 | hApplication | [IN] 应用句柄。 |
| szContainerName | [IN] ASCII字符串，表示所建立容器的名称，容器名称的最大长度不能超过64字节。 |
| phContainer | [OUT] 返回所建立容器的容器句柄。 |
| 返回值 | SAR\_OK： 成功。  其他： 错误码。 | |
| 备注 | 权限要求：需要用户权限。 | |

* + 1. **删除容器**

|  |  |  |
| --- | --- | --- |
| 函数原型 | ULONG DEVAPI SKF\_DeleteContainer(HAPPLICATION hApplication, LPSTR szContainerName) | |
| 功能描述 | 在应用下删除指定名称的容器并释放容器相关的资源。 | |
| 参数 | hApplication | [IN] 应用句柄。 |
| szContainerName | [IN] 指向删除容器的名称。 |
| 返回值 | SAR\_OK： 成功。  其他： 错误码。 | |
| 备注 | 权限要求：需要用户权限。 | |

* + 1. **打开容器**

|  |  |  |
| --- | --- | --- |
| 函数原型 | ULONG DEVAPI SKF\_OpenContainer(HAPPLICATION hApplication,LPSTR szContainerName,HCONTAINER \*phContainer) | |
| 功能描述 | 获取容器句柄。 | | |
| 参数 | hApplication | [IN] 应用句柄。 | |
| szContainerName | [IN] 容器的名称。 | |
| phContainer | [OUT] 返回所打开容器的句柄。 | |
| 返回值 | SAR\_OK： 成功。  其他： 错误码。 | | |

* + 1. **关闭容器**

|  |  |  |
| --- | --- | --- |
| 函数原型 | ULONG DEVAPI SKF\_CloseContainer(HCONTAINER hContainer) | |
| 功能描述 | 关闭容器句柄，并释放容器句柄相关资源。 | |
| 参数 | hContainer | [IN] 容器句柄。 |
| 返回值 | SAR\_OK： 成功。  其他： 错误码。 | |

* + 1. **枚举容器**

|  |  |  |  |
| --- | --- | --- | --- |
| 函数原型 | | ULONG DEVAPI SKF\_EnumContainer (HAPPLICATION hApplication,LPSTR szContainerName,ULONG \*pulSize) | |
| 功能描述 | 枚举应用下的所有容器并返回容器名称列表。 | | |
| 参数 | hApplication | | [IN] 应用句柄。 |
| szContainerName | | [OUT] 指向容器名称列表缓冲区，如果此参数为NULL时，pulSize表示返回数据所需要缓冲区的长度，如果此参数不为NULL时，返回容器名称列表，每个容器名以单个‘\0’为结束，列表以双‘\0’结束。 |
| pulSize | | [IN，OUT] 输入时表示szContainerName缓冲区的长度，输出时表示容器名称列表的长度。 |
| 返回值 | SAR\_OK： 成功。  其他： 错误码。 | | |

* + 1. **获得容器类型**

|  |  |  |  |
| --- | --- | --- | --- |
| 函数原型 | | ULONG DEVAPI SKF\_GetContianerType(HCONTAINER hContainer, ULONG \*pulContainerType) | |
| 功能描述 | 获取容器的类型 | | |
| 参数 | hContainer | | [IN] 容器句柄。 |
| pulContainerType | | [OUT] 获得的容器类型。指针指向的值为0表示未定、尚未分配类型或者为空容器，为1表示为RSA容器，为2表示为ECC容器。 |
| 返回值 | SAR\_OK： 成功。  其他： 错误码。 | | |

* + 1. **导入数字证书**

|  |  |  |
| --- | --- | --- |
| 函数原型 | ULONG DEVAPI SKF\_ImportCertificate(HCONTAINER hContainer, BOOL bSignFlag, BYTE\* pbCert, ULONG ulCertLen) | |
| 功能描述 | 向容器内导入数字证书。 | |
| 参数 | hContainer | [IN] 容器句柄。 |
| bSignFlag | [IN] TRUE表示签名证书，FALSE表示加密证书。 |
| pbCert | [IN] 指向证书内容缓冲区。 |
| ulCertLen | [IN] 证书长度。 |
| 返回值 | SAR\_OK： 成功。  其他： 错误码。 | |

* + 1. **导出数字证书**

|  |  |  |
| --- | --- | --- |
| 函数原型 | ULONG DEVAPI SKF\_ExportCertificate(HCONTAINER hContainer, BOOL bSignFlag, BYTE\* pbCert, ULONG \*pulCertLen) | |
| 功能描述 | 从容器内导出数字证书。 | |
| 参数 | hContainer | [IN] 容器句柄。 |
| bSignFlag | [IN] TRUE表示签名证书，FALSE表示加密证书。 |
| pbCert | [OUT] 指向证书内容缓冲区，如果此参数为NULL时，pulCertLen表示返回数据所需要缓冲区的长度，如果此参数不为NULL时，返回数字证书内容。 |
| pulCertLen | [IN/OUT] 输入时表示pbCert缓冲区的长度，输出时表示证书内容的长度。 |
| 返回值 | SAR\_OK： 成功。  其他： 错误码。 | |

* 1. **密码服务**
     1. **概述**

密码服务函数提供对称算法运算、非对称算法运算、密码杂凑运算、密钥管理、消息鉴别码计算等功能。密码服务系列函数如表24所示：

1. 密码服务系列函数

|  |  |
| --- | --- |
| 函数名称 | 功能 |
| SKF\_GenRandom | 生成随机数 |
| SKF\_GenExtRSAKey | 生成外部RSA密钥对 |
| SKF\_GenRSAKeyPair | 生成RSA签名密钥对 |
| SKF\_ImportRSAKeyPair | 导入RSA加密密钥对 |
| SKF\_RSASignData | RSA签名 |
| SKF\_RSAVerify | RSA验签 |
| SKF\_RSAExportSessionKey | RSA生成并导出会话密钥 |
| SKF\_ExtRSAPubKeyOperation | RSA外来公钥运算 |
| SKF\_ExtRSAPriKeyOperation | RSA外来私钥运算 |
| SKF\_GenECCKeyPair | 生成ECC签名密钥对 |
| SKF\_ImportECCKeyPair | 导入ECC加密密钥对 |
| SKF\_ECCSignData | ECC签名 |
| SKF\_ECCVerify | ECC验签 |
| SKF\_ECCExportSessionKey | ECC生成并导出会话密钥 |
| SKF\_ExtECCEncrypt | ECC外来公钥加密 |
| SKF\_ExtECCDecrypt | ECC外来私钥解密 |
| SKF\_ExtECCSign | ECC外来私钥签名 |
| SKF\_ExtECCVerify | ECC外来公钥验签 |
| SKF\_GenerateAgreementDataWithECC | ECC生成密钥协商参数并输出 |
| SKF\_GenerateKeyWithECC | ECC计算会话密钥 |
| SKF\_GenerateAgreementDataAndKeyWithECC | ECC产生协商数据并计算会话密钥 |
| SKF\_ExportPublicKey | 导出公钥 |
| SKF\_ImportSessionKey | 导入会话密钥 |
| SKF\_SetSymmKey | 明文导入会话密钥 |
| SKF\_EncryptInit | 加密初始化 |
| SKF\_Encrypt | 单组数据加密 |
| SKF\_EncryptUpdate | 多组数据加密 |
| SKF\_EncryptFinal | 结束加密 |
| SKF\_DecryptInit | 解密初始化 |
| SKF\_Decrypt | 单组数据解密 |
| SKF\_DecryptUpdate | 多组数据解密 |
| SKF\_DecryptFinal | 结束解密 |
| SKF\_DigestInit | 密码杂凑初始化 |
| SKF\_Digest | 单组数据密码杂凑 |
| SKF\_DigestUpdate | 多组数据密码杂凑 |
| SKF\_DigestFinal | 结束密码杂凑 |
| SKF\_MacInit | 消息鉴别码运算初始化 |
| SKF\_Mac | 单组数据消息鉴别码运算 |
| SKF\_MacUpdate | 多组数据消息鉴别码运算 |
| SKF\_MacFinal | 结束消息鉴别码运算 |
| SKF\_CloseHandle | 关闭密码对象句柄 |

* + 1. **生成随机数**

|  |  |  |  |
| --- | --- | --- | --- |
| 函数原型 | | ULONG DEVAPI SKF\_GenRandom (DEVHANDLE hDev, BYTE \*pbRandom,ULONG ulRandomLen) | |
| 功能描述 | 产生指定长度的随机数。 | | |
| 参数 | hDev | | [IN] 设备句柄。 |
| pbRandom | | [OUT]返回的随机数。 |
| ulRandomLen | | [IN] 随机数长度。 |
| 返回值 | SAR\_OK： 成功。  其他： 错误码。 | | | |

* + 1. **生成外部RSA密钥对**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 函数原型 | ULONG DEVAPI SKF\_GenExtRSAKey (DEVHANDLE hDev, ULONG ulBitsLen, RSAPRIVATEKEYBLOB \*pBlob) | | | |
| 功能描述 | 由设备生成RSA密钥对并明文输出。 | | | |
| 参数 | hDev | | [IN]设备句柄。 | |
| ulBitsLen | | [IN] 密钥模长。 | |
| pBlob | | [OUT] 返回的私钥数据结构。 | |
| 返回值 | | SAR\_OK： 成功。  其他： 错误码。 | |
| 备注： | 生成的私钥只用于输出，接口内不做保留和计算。 | | | |

* + 1. **生成RSA签名密钥对**

|  |  |  |
| --- | --- | --- |
| 函数原型 | ULONG DEVAPI SKF\_GenRSAKeyPair (HCONTAINER hContainer, ULONG ulBitsLen  , RSAPUBLICKEYBLOB \*pBlob) | |
| 功能描述 | 生成RSA签名密钥对并输出签名公钥。 | |
| 参数 | hContainer | [IN] 容器句柄。 |
| ulBitsLen | [IN] 密钥模长。 |
| pBlob | [OUT] 返回的RSA公钥数据结构。 |
| 返回值 | SAR\_OK： 成功。  其他： 错误码。 | | |
| 备注 | 权限要求：需要用户权限。 | |

* + 1. **导入RSA加密密钥对**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 函数原型 | ULONG DEVAPI SKF\_ImportRSAKeyPair (  HCONTAINER hContainer, ULONG ulSymAlgId,  BYTE \*pbWrappedKey, ULONG ulWrappedKeyLen,  BYTE \*pbEncryptedData, ULONG ulEncryptedDataLen) | | | |
| 功能描述 | 导入RSA加密公私钥对。 | | | |
| 参数 | hContainer | | [IN] 容器句柄。 | |
| ulSymAlgId | | [IN] 对称算法密钥标识。 | |
| pbWrappedKey | | [IN] 使用该容器内签名公钥保护的对称算法密钥。 | |
| ulWrappedKeyLen | | [IN] 保护的对称算法密钥长度。 | |
| pbEncryptedData | | [IN] 对称算法密钥保护的RSA加密私钥。私钥的格式遵循PKCS #1 v2.1: RSA Cryptography Standard中的私钥格式定义。 | |
| ulEncryptedDataLen | | [IN] 对称算法密钥保护的RSA加密公私钥对长度。 | |
| 返回值 | | SAR\_OK： 成功。  其他： 错误码。 | |
| 备注 | 权限要求：需要用户权限。 | | | |

* + 1. **RSA签名**

|  |  |  |
| --- | --- | --- |
| 函数原型 | ULONG DEVAPI SKF\_RSASignData(HCONTAINER hContainer, BYTE \*pbData, ULONG ulDataLen, BYTE \*pbSignature, ULONG \*pulSignLen) | |
| 功能描述 | 使用hContainer指定容器的签名私钥，对指定数据pbData进行数字签名。签名后的结果存放到pbSignature缓冲区，设置pulSignLen为签名的长度。 | |
| 参数 | hContainer | [IN] 用来签名的私钥所在容器句柄。 |
| pbData | [IN] 被签名的数据。 |
| ulDataLen | [IN] 签名数据长度，应不大于RSA密钥模长-11。 |
| pbSignature | [OUT] 存放签名结果的缓冲区指针，如果值为NULL，用于取得签名结果长度。 |
| pulSignLen | [IN，OUT] 输入时表示签名结果缓冲区大小，输出时表示签名结果长度。 |
| 返回值 | SAR\_OK： 成功。  其他： 错误码。 | | |
| 备注 | 权限要求：需要用户权限。 | |

* + 1. **RSA验签**

|  |  |  |  |
| --- | --- | --- | --- |
| 函数原型 | ULONG DEVAPI SKF\_RSAVerify (DEVHANDLE hDev , RSAPUBLICKEYBLOB\* pRSAPubKeyBlob, BYTE \*pbData, ULONG ulDataLen, BYTE \*pbSignature, ULONG ulSignLen) | | |
| 功能描述 | 验证RSA签名。用pRSAPubKeyBlob内的公钥值对待验签数据进行验签。 | | |
| 参数 | hDev | [IN] 设备句柄。 | |
| pRSAPubKeyBlob | [IN] RSA公钥数据结构。 | |
| pbData | [IN] 待验证签名的数据。 | |
| ulDataLen | [IN] 数据长度，应不大于公钥模长-11。 | |
| pbSignature | [IN] 待验证的签名值。 | |
| ulSignLen | [IN] 签名值长度，必须为公钥模长。 | |
| 返回值 | SAR\_OK： 成功。  其他： 错误码。 | |

* + 1. **RSA生成并导出会话密钥**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 函数原型 | ULONG DEVAPI SKF\_RSAExportSessionKey (HCONTAINER hContainer, ULONG ulAlgId, RSAPUBLICKEYBLOB \*pPubKey, BYTE \*pbData, ULONG \*pulDataLen, HANDLE \*phSessionKey) | | | |
| 功能描述 | 生成会话密钥并用外部RSA公钥加密输出。 | | | |
| 参数 | hContainer | | [IN] 容器句柄。 | |
| ulAlgId | | [IN] 会话密钥算法标识。 | |
| pPubKey | | [IN] 加密会话密钥的RSA公钥数据结构。 | |
| pbData | | [OUT] 导出的加密会话密钥密文，按照PKCS#1v1.5要求封装。 | |
| pulDataLen | | [IN，OUT] 输入时表示会话密钥密文数据缓冲区长度，输出时表示会话密钥密文的实际长度。 | |
| phSessionKey | | [OUT] 导出的密钥句柄。 | |
| 返回值 | | SAR\_OK： 成功。  其他： 错误码。 | |

* + 1. **RSA外来公钥运算**

|  |  |  |
| --- | --- | --- |
| 函数原型 | ULONG DEVAPI SKF\_ExtRSAPubKeyOperation (DEVHANDLE hDev, RSAPUBLICKEYBLOB\* pRSAPubKeyBlob,BYTE\* pbInput, ULONG ulInputLen, BYTE\* pbOutput, ULONG\* pulOutputLen) | |
| 功能描述 | 使用外部传入的RSA公钥对输入数据做公钥运算并输出结果。 | |
| 参数 | hDev | [IN] 设备句柄。 |
| pRSAPubKeyBlob | [IN] RSA公钥数据结构。 |
| pbInput | [IN] 指向待运算的原始数据缓冲区。 |
| ulInputLen | [IN] 待运算原始数据的长度，必须为公钥模长。 |
| pbOutput | [OUT] 指向RSA公钥运算结果缓冲区，如果该参数为NULL，则由pulOutputLen返回运算结果的实际长度。 |
| pulOutputLen | [IN，OUT] 输入时表示pbOutput缓冲区的长度，输出时表示RSA公钥运算结果的实际长度。 |
| 返回值 | SAR\_OK： 成功。  其他： 错误码。 | |

* + 1. **RSA外来私钥运算**

|  |  |  |
| --- | --- | --- |
| 函数原型 | ULONG DEVAPI SKF\_ExtRSAPriKeyOperation (DEVHANDLE hDev, RSAPRIVATEKEYBLOB\* pRSAPriKeyBlob,BYTE\* pbInput, ULONG ulInputLen, BYTE\* pbOutput, ULONG\* pulOutputLen) | |
| 功能描述 | 直接使用外部传入的RSA私钥对输入数据做私钥运算并输出结果。 | |
| 参数 | hDev | [IN] 设备句柄。 |
| pRSAPriKeyBlob | [IN] RSA私钥数据结构。 |
| pbInput | [IN] 指向待运算数据缓冲区。 |
| ulInputLen | [IN] 待运算数据的长度，必须为公钥模长。 |
| pbOutput | [OUT] RSA私钥运算结果，如果该参数为NULL，则由pulOutputLen返回运算结果的实际长度。 |
| pulOutputLen | [IN，OUT] 输入时表示pbOutput缓冲区的长度，输出时表示RSA私钥运算结果的实际长度。 |
| 返回值 | SAR\_OK： 成功。  其他： 错误码。 | |

* + 1. **生成ECC签名密钥对**

|  |  |  |
| --- | --- | --- |
| 函数原型 | ULONG DEVAPI SKF\_GenECCKeyPair (HCONTAINER hContainer, ULONG ulAlgId， ECCPUBLICKEYBLOB \*pBlob) | |
| 功能描述 | 生成ECC签名密钥对并输出签名公钥。 | |
| 参数 | hContainer | [IN] 密钥容器句柄。 |
| ulAlgId | [IN] 算法标识，只支持SGD\_SM2\_1算法。 |
| pBlob | [OUT] 返回ECC公钥数据结构。 |
| 返回值 | SAR\_OK： 成功。  其他： 错误码。 | | |
| 备注 | 权限要求：需要用户权限。 | |

* + 1. **导入ECC加密密钥对**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 函数原型 | ULONG DEVAPI SKF\_ImportECCKeyPair (  HCONTAINER hContainer, PENVELOPEDKEYBLOB pEnvelopedKeyBlob) | | | |
| 功能描述 | 导入ECC公私钥对。 | | | |
| 参数 | hContainer | | [IN] 密钥容器句柄。 | |
| pEnvelopedKeyBlob | | [IN] 受保护的加密密钥对。 | |
| 返回值 | | SAR\_OK： 成功。  其他： 错误码。 | |
| 备注 | 权限要求：需要用户权限。 | | | |

* + 1. **ECC签名**

|  |  |  |
| --- | --- | --- |
| 函数原型 | ULONG DEVAPI SKF\_ECCSignData (HCONTAINER hContainer, BYTE \*pbData, ULONG ulDataLen, PECCSIGNATUREBLOB pSignature) | |
| 功能描述 | ECC数字签名。采用ECC算法和指定私钥hKey，对指定数据pbData进行数字签名。签名后的结果存放到pSignature中。 | |
| 参数 | hContainer | [IN] 密钥容器句柄。 |
| pbData | [IN] 待签名的数据。 |
| ulDataLen | [IN] 待签名数据长度，必须小于密钥模长。 |
| pSignature | [OUT] 签名值。 |
| 返回值 | SAR\_OK： 成功。  其他： 错误码。 | | |
| 备注 | 权限要求：需要用户权限。  输入数据为待签数据的杂凑值。当使用SM2算法时，该输入数据为待签数据经过SM2签名预处理的结果，预处理过程遵循《公钥密码基础设施应用技术体系 SM2算法密码使用规范》。 | |

* + 1. **ECC验签**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 函数原型 | ULONG DEVAPI SKF\_ECCVerify (DEVHANDLE hDev , ECCPUBLICKEYBLOB\* pECCPubKeyBlob, BYTE \*pbData, ULONG ulDataLen, PECCSIGNATUREBLOB pSignature) | | | |
| 功能描述 | 用ECC公钥对数据进行验签。 | | | |
| 参数 | hDev | | [IN] 设备句柄。 | |
| pECCPubKeyBlob | | [IN] ECC公钥数据结构。 | |
| pbData | | [IN] 待验证签名的数据。 | |
| ulDataLen | | [IN] 数据长度。 | |
| pSignature | | [IN] 待验证签名值。 | |
| 返回值 | | SAR\_OK： 成功。  其他： 错误码。 | |
| 备注 | | 输入数据为待签数据的杂凑值。当使用SM2算法时，该输入数据为待签数据经过SM2签名预处理的结果，预处理过程遵循《公钥密码基础设施应用技术体系 SM2算法密码使用规范》。 | |

* + 1. **ECC生成并导出会话密钥**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 函数原型 | ULONG DEVAPI SKF\_ECCExportSessionKey (HCONTAINER hContainer, ULONG ulAlgId, ECCPUBLICKEYBLOB \*pPubKey, PECCCIPHERBLOB pData, HANDLE \*phSessionKey) | | | |
| 功能描述 | 生成会话密钥并用外部公钥加密导出。 | | | |
| 参数 | hContainer | | [IN] 容器句柄。 | |
| ulAlgId | | [IN] 会话密钥算法标识。 | |
| pPubKey | | [IN] 外部输入的公钥结构。 | |
| pData | | [OUT] 会话密钥密文。 | |
| phSessionKey | | [OUT] 会话密钥句柄。 | |
| 返回值 | | SAR\_OK： 成功。  其他： 错误码。 | |

* + 1. **ECC外来公钥加密**

|  |  |  |
| --- | --- | --- |
| 函数原型 | ULONG DEVAPI SKF\_ExtECCEncrypt (DEVHANDLE hDev, ECCPUBLICKEYBLOB\* pECCPubKeyBlob,BYTE\* pbPlainText, ULONG ulPlainTextLen, PECCCIPHERBLOB pCipherText) | |
| 功能描述 | 使用外部传入的ECC公钥对输入数据做加密运算并输出结果。 | |
| 参数 | hDev | [IN] 设备句柄。 |
| pECCPubKeyBlob | [IN] ECC公钥数据结构。 |
| pbPlainText | [IN] 待加密的明文数据。 |
| ulPlainTextLen | [IN] 待加密明文数据的长度。 |
| pCipherText | [OUT] 密文数据。 |
| 返回值 | SAR\_OK： 成功。  其他： 错误码。 | |

* + 1. **ECC外来私钥解密**

|  |  |  |
| --- | --- | --- |
| 函数原型 | ULONG DEVAPI SKF\_ExtECCDecrypt (DEVHANDLE hDev, ECCPRIVATEKEYBLOB\* pECCPriKeyBlob, PECCCIPHERBLOB pCipherText, BYTE\* pbPlainText, ULONG\* pulPlainTextLen) | |
| 功能描述 | 使用外部传入的ECC私钥对输入数据做解密运算并输出结果。 | |
| 参数 | hDev | [IN] 设备句柄。 |
| pECCPriKeyBlob | [IN] ECC私钥数据结构。 |
| pCipherText | [IN] 待解密的密文数据。 |
| pbPlainText | [OUT] 返回明文数据，如果该参数为NULL，则由pulPlainTextLen返回明文数据的实际长度。 |
| pulPlainTextLen | [IN，OUT] 输入时表示pbPlainText缓冲区的长度，输出时表示明文数据的实际长度。 |
| 返回值 | SAR\_OK： 成功。  其他： 错误码。 | |

* + 1. **ECC外来私钥签名**

|  |  |  |
| --- | --- | --- |
| 函数原型 | ULONG DEVAPI SKF\_ExtECCSign (DEVHANDLE hDev, ECCPRIVATEKEYBLOB\* pECCPriKeyBlob, BYTE\* pbData, ULONG ulDataLen, PECCSIGNATUREBLOB pSignature) | |
| 功能描述 | 使用外部传入的ECC私钥对输入数据做签名运算并输出结果。 | |
| 参数 | hDev | [IN] 设备句柄。 |
| pECCPriKeyBlob | [IN] ECC私钥数据结构。 |
| pbData | [IN] 待签名数据。 |
| ulDataLen | [IN] 待签名数据的长度。 |
| pSignature | [OUT]签名值。 |
| 返回值 | SAR\_OK： 成功。  其他： 错误码。 | |
| 备注： | 输入数据为待签数据的杂凑值。当使用SM2算法时，该输入数据为待签数据经过SM2签名预处理的结果，预处理过程遵循《公钥密码基础设施应用技术体系 SM2算法密码使用规范》。 | |

* + 1. **ECC外来公钥验签**

|  |  |  |
| --- | --- | --- |
| 函数原型 | ULONG DEVAPI SKF\_ExtECCVerify (DEVHANDLE hDev, ECCPUBLICKEYBLOB\* pECCPubKeyBlob,BYTE\* pbData, ULONG ulDataLen, PECCSIGNATUREBLOB pSignature) | |
| 功能描述 | 外部使用传入的ECC公钥做签名验证。 | |
| 参数 | hDev | [IN] 设备句柄。 |
| pECCPubKeyBlob | [IN] ECC公钥数据结构。 |
| pbData | [IN] 待验证数据。 |
| ulDataLen | [IN] 待验证数据的长度。 |
| pSignature | [IN] 签名值。 |
| 返回值 | SAR\_OK： 成功。  其他： 错误码。 | |
| 备注： | 输入数据为待签数据的杂凑值。当使用SM2算法时，该输入数据为待签数据经过SM2签名预处理的结果，预处理过程遵循《公钥密码基础设施应用技术体系 SM2算法密码使用规范》。 | |

* + 1. **ECC生成密钥协商参数并输出**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 函数原型 | | ULONG DEVAPI SKF\_GenerateAgreementDataWithECC (HCONTAINER hContainer, ULONG ulAlgId,ECCPUBLICKEYBLOB\* pTempECCPubKeyBlob,BYTE\* pbID, ULONG ulIDLen,HANDLE \*phAgreementHandle) | | |
| 功能描述 | | 使用ECC密钥协商算法，为计算会话密钥而产生协商参数，返回临时ECC密钥对的公钥及协商句柄。 | | |
| 参数 | | hContainer | [IN] 容器句柄。 | |
| ulAlgId | [IN] 会话密钥算法标识。 | |
| pTempECCPubKeyBlob | [OUT] 发起方临时ECC公钥。 | |
| pbID | [IN] 发起方的ID。 | |
| ulIDLen | [IN] 发起方ID的长度，不大于32。 | |
| phAgreementHandle | [OUT] 返回的密钥协商句柄。 | |
| 返回值 | SAR\_OK： 成功。  其他： 错误码。 | | |
| 备注 | | 为协商会话密钥，协商的发起方应首先调用本函数。 | | |  |

* + 1. **ECC产生协商数据并计算会话密钥**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 函数原型： | ULONG DEVAPI SKF\_GenerateAgreementDataAndKeyWithECC(  HANDLE hContainer, ULONG ulAlgId,  ECCPUBLICKEYBLOB\* pSponsorECCPubKeyBlob,  ECCPUBLICKEYBLOB\* pSponsorTempECCPubKeyBlob,  ECCPUBLICKEYBLOB\* pTempECCPubKeyBlob,  BYTE\* pbID, ULONG ulIDLen, BYTE \*pbSponsorID, ULONG ulSponsorIDLen,  HANDLE \*phKeyHandle) | | | |
| 功能描述： | 使用ECC密钥协商算法，产生协商参数并计算会话密钥，输出临时ECC密钥对公钥，并返回产生的密钥句柄。 | | | |
| 参数： | hContainer | | [IN] 容器句柄。 | |
| ulAlgId | | [IN] 会话密钥算法标识。 | |
| pSponsorECCPubKeyBlob | | [IN] 发起方的ECC公钥。 | |
| pSponsorTempECCPubKeyBlob | | [IN] 发起方的临时ECC公钥。 | |
| pTempECCPubKeyBlob | | [OUT] 响应方的临时ECC公钥。 | |
| pbID | | [IN] 响应方的ID。 | |
| ulIDLen | | [IN] 响应方ID的长度，不大于32。 | |
| pbSponsorID | | [IN] 发起方的ID。 | |
| ulSponsorIDLen | | [IN] 发起方ID的长度，不大于32。 | |
| phKeyHandle | | [OUT] 返回的对称算法密钥句柄。 | |
| 返回值 | | SAR\_OK： 成功。  其他： 错误码。 | |
| 备注： | 本函数由响应方调用。 | | | |

* + 1. **ECC计算会话密钥**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 函数原型： | | ULONG DEVAPI SKF\_GenerateKeyWithECC (HANDLE hAgreementHandle,  ECCPUBLICKEYBLOB\* pECCPubKeyBlob,  ECCPUBLICKEYBLOB\* pTempECCPubKeyBlob,  BYTE\* pbID, ULONG ulIDLen, HANDLE \*phKeyHandle) | | |
| 功能描述： | | 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： | | hAgreementHandle | [IN] 密钥协商句柄。 | |
| pECCPubKeyBlob | [IN] 外部输入的响应方ECC公钥。 | |
| pTempECCPubKeyBlob | [IN] 外部输入的响应方临时ECC公钥。 | |
| pbID | [IN] 响应方的ID。 | |
| ulIDLen | [IN] 响应方ID的长度，不大于32。 | |
| phKeyHandle | [OUT] 返回的密钥句柄。 | |
| 返回值 | SAR\_OK： 成功。  其他： 错误码。 | | |
| 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。计算过程遵循《公钥密码基础设施应用技术体系 SM2算法密码使用规范》。 | | |

* + 1. **导出公钥**

|  |  |  |  |
| --- | --- | --- | --- |
| 函数原型 | ULONG DEVAPI SKF\_ExportPublicKey (HCONTAINER hContainer, BOOL bSignFlag， BYTE\* pbBlob, ULONG\* pulBlobLen) | | |
| 功能描述 | 导出容器中的签名公钥或者加密公钥。 | | |
| 参数 | hContainer | [IN] 密钥容器句柄。 | |
| bSignFlag | [IN] TRUE表示导出签名公钥，FALSE表示导出加密公钥。 | |
| pbBlob | [OUT] 指向RSA公钥结构（RSAPUBLICKEYBLOB）或者ECC公钥结构（ECCPUBLICKEYBLOB），如果此参数为NULL时，由pulBlobLen返回pbBlob的长度。 | |
| pulBlobLen | [IN，OUT] 输入时表示pbBlob缓冲区的长度，输出时表示导出公钥结构的大小。 | |
| 返回值 | SAR\_OK： 成功。  其他： 错误码。 | |

* + 1. **导入会话密钥**

|  |  |  |  |
| --- | --- | --- | --- |
| 函数原型 | ULONG DEVAPI SKF\_ImportSessionKey (HCONTAINER hContainer, ULONG ulAlgId,BYTE \*pbWrapedData,ULONG ulWrapedLen，HANDLE \*phKey) | | |
| 功能描述 | 导入会话密钥密文，使用容器中的加密私钥解密得到会话密钥。 | | |
| 参数 | hContainer | [IN] 容器句柄。 | |
| ulAlgId | [IN] 会话密钥算法标识。 | |
| pbWrapedData | [IN] 要导入的会话密钥密文。当容器为ECC类型时，此参数为ECCCIPHERBLOB密文数据，当容器为RSA类型时，此参数为RSA公钥加密后的数据。 | |
| ulWrapedLen | [IN] 会话密钥密文长度。 | |
| phKey | [OUT] 返回会话密钥句柄。 | |
| 返回值 | SAR\_OK： 成功。  其他： 错误码。 | |
| 备注 | 权限要求：需要用户权限。 | | |

* + 1. **明文导入会话密钥**

|  |  |  |
| --- | --- | --- |
| 函数原型 | ULONG DEVAPI SKF\_SetSymmKey (DEVHANDLE hDev, BYTE\* pbKey, ULONG ulAlgID, HANDLE\* phKey) | |
| 功能描述 | 设置明文对称密钥，返回密钥句柄。 | |
| 参数 | hDev | [IN] 设备句柄。 |
| pbKey | [IN] 指向会话密钥值的缓冲区。 |
| ulAlgID | [IN] 会话密钥算法标识。 |
| phKey | [OUT] 返回会话密钥句柄。 |
| 返回值 | SAR\_OK： 成功。  其他： 错误码。 | |

* + 1. **加密初始化**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 函数原型 | | ULONG DEVAPI SKF\_EncryptInit (HANDLE hKey, BLOCKCIPHERPARAM EncryptParam) | | | |
| 功能描述 | | | 数据加密初始化。设置数据加密的算法相关参数。 | | |
| 参数 | | | hKey | [IN] 加密密钥句柄。 | |
| EncryptParam | [IN] 分组密码算法相关参数：初始向量、初始向量长度、填充方法、反馈值的位长度。 | |
| 返回值 | SAR\_OK： 成功。  其他： 错误码。 | | | |

* + 1. **单组数据加密**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 函数原型 | | ULONG DEVAPI SKF\_Encrypt(HANDLE hKey, BYTE \* pbData, ULONG ulDataLen, BYTE \*pbEncryptedData, ULONG \*pulEncryptedLen) | | |
| 功能描述 | | 单一分组数据的加密操作。用指定加密密钥对指定数据进行加密，被加密的数据只包含一个分组，加密后的密文保存到指定的缓冲区中。SKF\_Encrypt只对单个分组数据进行加密，在调用SKF\_Encrypt之前，必须调用SKF\_EncryptInit初始化加密操作。SKF\_Encypt等价于先调用SKF\_EncryptUpdate再调用SKF\_EncryptFinal。 | | |
| 参数 | | hKey | [IN] 加密密钥句柄。 | |
| pbData | [IN] 待加密数据。 | |
| ulDataLen | [IN] 待加密数据长度。 | |
| pbEncryptedData | [OUT] 加密后的数据缓冲区指针，可以为NULL，用于获得加密后数据长度。 | |
| pulEncryptedLen | [IN，OUT] 输入时表示结果数据缓冲区长度，输出时表示结果数据实际长度。 | |
| 返回值 | SAR\_OK： 成功。  其他： 错误码。 | | |

* + 1. **多组数据加密**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 函数原型 | | ULONG DEVAPI SKF\_EncryptUpdate(HANDLE hKey, BYTE \* pbData, ULONG ulDataLen, BYTE \*pbEncryptedData, ULONG \*pulEncryptedLen) | | |
| 功能描述 | | 多个分组数据的加密操作。用指定加密密钥对指定数据进行加密，被加密的数据包含多个分组，加密后的密文保存到指定的缓冲区中。SKF\_EncryptUpdate对多个分组数据进行加密，在调用SKF\_EncryptUpdate之前，必须调用SKF\_EncryptInit初始化加密操作；在调用SKF\_EncryptUpdate之后，必须调用SKF\_EncryptFinal结束加密操作。 | | |
| 参数 | | hKey | [IN] 加密密钥句柄。 | |
| pbData | [IN] 待加密数据。 | |
| ulDataLen | [IN] 待加密数据长度。 | |
| pbEncryptedData | [OUT] 加密后的数据缓冲区指针。 | |
| pulEncryptedLen | [OUT] 返回加密后的数据长度。 | |
| 返回值 | SAR\_OK： 成功。  其他： 错误码。 | | |

* + 1. **结束加密**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 函数原型 | | ULONG DEVAPI SKF\_EncryptFinal (HANDLE hKey, BYTE \*pbEncryptedData, ULONG \*ulEncryptedDataLen ) | | |
| 功能描述 | | 结束多个分组数据的加密，返回剩余加密结果。先调用SKF\_EncryptInit初始化加密操作，再调用SKF\_EncryptUpdate对多个分组数据进行加密，最后调用SKF\_EncryptFinal结束多个分组数据的加密。 | | |
| 参数 | | hKey | [IN] 加密密钥句柄。 | |
| pbEncyptedData | [OUT] 加密结果的缓冲区。 | |
| ulEncyptedDataLen | [OUT] 加密结果的长度。 | |
| 返回值 | SAR\_OK： 成功。  其他： 错误码。 | | |

* + 1. **解密初始化**

|  |  |  |  |
| --- | --- | --- | --- |
| 函数原型 | ULONG DEVAPI SKF\_DecryptInit (HANDLE hKey, BLOCKCIPHERPARAM DecryptParam) | | |
| 功能描述 | 数据解密初始化，设置解密密钥相关参数。调用SKF\_DecryptInit之后，可以调用SKF\_Decrypt对单个分组数据进行解密，也可以多次调用SKF\_DecryptUpdate之后再调用SKF\_DecryptFinal完成对多个分组数据的解密。 | | |
| 参数 | hKey | [IN] 解密密钥句柄。 | |
| DecryptParam | [IN] 分组密码算法相关参数：初始向量、初始向量长度、填充方法、反馈值的位长度。 | |
| 返回值 | SAR\_OK： 成功。  其他： 错误码。 | |

* + 1. **单组数据解密**

|  |  |  |  |
| --- | --- | --- | --- |
| 函数原型 | ULONG DEVAPI SKF\_Decrypt(HANDLE hKey, BYTE \* pbEncryptedData, ULONG ulEncryptedLen, BYTE \* pbData, ULONG \* pulDataLen) | | |
| 功能描述 | 单个分组数据的解密操作。用指定解密密钥对指定数据进行解密，被解密的数据只包含一个分组，解密后的明文保存到指定的缓冲区中。SKF\_Decrypt只对单个分组数据进行解密，在调用SKF\_Decrypt之前，必须调用SKF\_DecryptInit初始化解密操作。SKF\_Decypt等价于先调用SKF\_DecryptUpdate再调用SKF\_DecryptFinal。 | | |
| 参数 | hKey | [IN] 解密密钥句柄。 | |
| pbEncryptedData | [IN] 待解密数据。 | |
| ulEncryptedLen | [IN] 待解密数据长度。 | |
| pbData | [OUT] 指向解密后的数据缓冲区指针，当为NULL时可获得解密后的数据长度。 | |
| pulDataLen | [IN，OUT] 输入时表示结果数据缓冲区长度，输出时表示结果数据实际长度。 | |
| 返回值 | SAR\_OK： 成功。  其他： 错误码。 | |

* + 1. **多组数据解密**

|  |  |  |  |
| --- | --- | --- | --- |
| 函数原型 | ULONG DEVAPI SKF\_DecryptUpdate(HANDLE hKey, BYTE \* pbEncryptedData, ULONG ulEncryptedLen, BYTE \* pbData, ULONG \* pulDataLen) | | |
| 功能描述 | 多个分组数据的解密操作。用指定解密密钥对指定数据进行解密，被解密的数据包含多个分组，解密后的明文保存到指定的缓冲区中。SKF\_DecryptUpdate对多个分组数据进行解密，在调用SKF\_DecryptUpdate之前，必须调用SKF\_DecryptInit初始化解密操作；在调用SKF\_DecryptUpdate之后，必须调用SKF\_DecryptFinal结束解密操作。 | | |
| 参数 | hKey | [IN] 解密密钥句柄。 | |
| pbEncryptedData | [IN] 待解密数据。 | |
| ulEncryptedLen | [IN] 待解密数据长度。 | |
| pbData | [OUT] 指向解密后的数据缓冲区指针。 | |
| pulDataLen | [IN，OUT] 输入时表示结果数据缓冲区长度，输出时表示结果数据实际长度。 | |
| 返回值 | SAR\_OK： 成功。  其他： 错误码。 | |

* + 1. **结束解密**

|  |  |  |  |
| --- | --- | --- | --- |
| 函数原型 | ULONG DEVAPI SKF\_DecryptFinal (HANDLE hKey, BYTE \*pbDecryptedData, ULONG \*pulDecryptedDataLen) | | |
| 功能描述 | | 结束多个分组数据的解密。先调用SKF\_DecryptInit初始化解密操作，再调用SKF\_DecryptUpdate对多个分组数据进行解密，最后调用SKF\_DecryptFinal结束多个分组数据的解密。 | |
| 参数 | | hKey | [IN] 解密密钥句柄。 |
| pbDecryptedData | [OUT] 指向解密结果的缓冲区，如果此参数为NULL时，由pulDecryptedDataLen返回解密结果的长度。 |
| pulDecryptedDataLen | [IN，OUT] 输入时表示pbDecryptedData缓冲区的长度，输出时表示解密结果的长度。 |
| 返回值 | | SAR\_OK： 成功。  其他： 错误码。 | |

* + 1. **密码杂凑初始化**

|  |  |  |
| --- | --- | --- |
| 函数原型 | ULONG DEVAPI SKF\_DigestInit(DEVHANDLE hDev, ULONG ulAlgID, ECCPUBLICKEYBLOB \*pPubKey, unsigned char \*pucID, ULONG ulIDLen, HANDLE \*phHash) | |
| 功能描述 | 初始化密码杂凑计算操作，指定计算密码杂凑的算法。 | |
| 参数 | hDev | [IN] 连接设备时返回的设备句柄。 |
| ulAlgID | [IN] 密码杂凑算法标识。 |
| pPubKey | [IN] 签名者公钥。当alAlgID为SGD\_SM3时有效。 |
| pucID | [IN] 签名者的ID值，当alAlgID为SGD\_SM3时有效。 |
| ulIDLen | [IN] 签名者ID的长度，当alAlgID为SGD\_SM3时有效。 |
| phHash | [OUT] 密码杂凑对象句柄。 |
| 返回值 | SAR\_OK： 成功。  其他： 错误码。 | |
| 备注 | 当ulAlgID为SGD\_SM3且ulIDLen不为0的情况下pPubKey、pucID有效，执行SM2算法签名预处理1操作。计算过程遵循《公钥密码基础设施应用技术体系 SM2算法密码使用规范》。 | |

* + 1. **单组数据密码杂凑**

|  |  |  |
| --- | --- | --- |
| 函数原型 | ULONG DEVAPI SKF\_Digest (HANDLE hHash, BYTE \*pbData, ULONG ulDataLen, BYTE \*pbHashData, ULONG \*pulHashLen) | |
| 功能描述 | 对单一分组的消息进行密码杂凑计算。调用SKF\_Digest之前，必须调用SKF\_DigestInit初始化密码杂凑计算操作。SKF\_Digest等价于多次调用SKF\_DigestUpdate之后再调用SKF\_DigestFinal。 | |
| 参数 | hHash | [IN] 密码杂凑对象句柄。 |
| pbData | [IN] 指向消息数据的缓冲区。 |
| ulDataLen | [IN] 消息数据的长度。 |
| pbHashData | [OUT] 密码杂凑数据缓冲区指针，当此参数为NULL时，由pulHashLen返回密码杂凑结果的长度。 |
| pulHashLen | [IN，OUT] 输入时表示结果数据缓冲区长度，输出时表示结果数据实际长度。 |
| 返回值 | SAR\_OK： 成功。  其他： 错误码。 | |

* + 1. **多组数据密码杂凑**

|  |  |  |
| --- | --- | --- |
| 函数原型 | ULONG DEVAPI SKF\_DigestUpdate (HANDLE hHash, BYTE \*pbData, ULONG ulDataLen) | |
| 功能描述 | 对多个分组的消息进行密码杂凑计算。调用SKF\_DigestUpdate之前，必须调用SKF\_DigestInit初始化密码杂凑计算操作；调用SKF\_DigestUpdate之后，必须调用SKF\_DigestFinal结束密码杂凑计算操作。 | |
| 参数 | hHash | [IN] 密码杂凑对象句柄。 |
| pbData | [IN] 指向消息数据的缓冲区。 |
| ulDataLen | [IN] 消息数据的长度。 |
| 返回值 | SAR\_OK： 成功。  其他： 错误码。 | |

* + 1. **结束密码杂凑**

|  |  |  |
| --- | --- | --- |
| 函数原型 | ULONG DEVAPI SKF\_DigestFinal (HANDLE hHash, BYTE \*pHashData, ULONG \*pulHashLen) | |
| 功能描述 | 结束多个分组消息的密码杂凑计算操作，将密码杂凑结果保存到指定的缓冲区。 | |
| 参数 | hHash | [IN] 密码杂凑对象句柄。 |
| pHashData | [OUT] 返回的密码杂凑结果缓冲区指针，如果此参数NULL时，由pulHashLen返回杂凑结果的长度。 |
| pulHashLen | [IN，OUT] 输入时表示杂凑结果缓冲区的长度，输出时表示密码杂凑结果的长度。 |
| 返回值 | SAR\_OK： 成功。  其他： 错误码。 | |
| 备注 | SKF\_DigestFinal必须用于SKF\_DigestUpdate之后。 | |

* + 1. **消息鉴别码运算初始化**

|  |  |  |
| --- | --- | --- |
| 函数原型 | ULONG DEVAPI SKF\_MacInit (HANDLE hKey, BLOCKCIPHERPARAM\* pMacParam, HANDLE \*phMac) | |
| 功能描述 | 初始化消息鉴别码计算操作，设置计算消息鉴别码的所需参数，并返回消息鉴别码句柄。 | |
| 参数 | hKey | [IN] 计算消息鉴别码的密钥句柄。 |
| pMacParam | [IN] 消息认证计算相关参数，包括初始向量、初始向量长度、填充方法等。 |
| phMac | [OUT] 消息鉴别码对象句柄。 |
| 返回值 | SAR\_OK： 成功。  其他： 错误码。 | |
| 备注 | 消息鉴别码计算采用分组加密算法的CBC模式，将加密结果的最后一块作为计算结果。待计算数据的长度必须是分组加密算法块长的倍数，接口内部不作数据填充。 | |

* + 1. **单组数据消息鉴别码运算**

|  |  |  |
| --- | --- | --- |
| 函数原型 | ULONG DEVAPI SKF\_Mac(HANDLE hMac, BYTE\* pbData, ULONG ulDataLen, BYTE \*pbMacData, ULONG \*pulMacLen) | |
| 功能描述 | SKF\_Mac计算单一分组数据的消息鉴别码。 | |
| 参数 | hMac | [IN] 消息鉴别码句柄。 |
| pbData | [IN] 指向待计算数据的缓冲区。 |
| ulDataLen | [IN] 待计算数据的长度。 |
| pbMacData | [OUT] 指向计算后的Mac结果，如果此参数为NULL时，由pulMacLen返回计算后Mac结果的长度。 |
| pulMacLen | [IN，OUT] 输入时表示pbMacData缓冲区的长度，输出时表示Mac结果的长度。 |
| 返回值 | SAR\_OK： 成功。  其他： 错误码。 | |
| 备注 | 调用SKF\_Mac之前，必须调用SKF\_MacInit初始化消息鉴别码计算操作。SKF\_Mac等价于多次调用SKF\_MacUpdate之后再调用SKF\_MacFinal。 | |

* + 1. **多组数据消息鉴别码运算**

|  |  |  |
| --- | --- | --- |
| 函数原型 | ULONG DEVAPI SKF\_MacUpdate(HANDLE hMac, BYTE \* pbData, ULONG ulDataLen) | |
| 功能描述 | 计算多个分组数据的消息鉴别码。 | |
| 参数 | hMac | [IN] 消息鉴别码句柄。 |
| pbData | [IN] 指向待计算数据的缓冲区。 |
| plDataLen | [IN] 待计算数据的长度。 |
| 返回值 | SAR\_OK： 成功。  其他： 错误码。 | |
| 备注 | 调用SKF\_MacUpdate之前，必须调用SKF\_MacInit初始化消息鉴别码计算操作；调用SKF\_MacUpdate之后，必须调用SKF\_MacFinal结束多个分组数据的消息鉴别码计算操作。 | |

* + 1. **结束消息鉴别码运算**

|  |  |  |
| --- | --- | --- |
| 函数原型 | ULONG DEVAPI SKF\_MacFinal (HANDLE hMac, BYTE \*pbMacData, ULONG \*pulMacDataLen) | |
| 功能描述 | 结束多个分组数据的消息鉴别码计算操作。 | |
| 参数 | hMac | [IN] 消息鉴别码句柄。 |
| pbMacData | [OUT] 指向消息鉴别码的缓冲区，当此参数为NULL时，由pulMacDataLen返回消息鉴别码返回的长度。 |
| pulMacDataLen | [OUT] 调用时表示消息鉴别码缓冲区的最大长度，返回消息鉴别码的长度。 |
| 返回值 | SAR\_OK： 成功。  其他： 错误码。 | |
| 备注 | SKF\_MacFinal必须用于SKF\_MacUpdate之后。 | |

* + 1. **关闭密码对象句柄**

|  |  |  |
| --- | --- | --- |
| 函数原型 | ULONG DEVAPI SKF\_CloseHandle(HANDLE hHandle) | |
| 功能描述 | 关闭会话密钥、密码杂凑对象、消息鉴别码对象、ECC密钥协商等句柄。 | |
| 参数 | hHandle | [IN] 要关闭的对象句柄。 |
| 返回值 | SAR\_OK： 成功。  其他： 错误码。 | |

1. **设备的安全要求**
   1. **设备使用阶段**

设备的使用分成两个阶段：

1）出厂阶段：设备出厂时，预置设备认证密钥，在此阶段除修改设备认证密钥及创建应用操作外，禁止其他操作。

2）应用阶段：已创建了应用的设备进入应用阶段。在此阶段，可进行所有操作。

* 1. **权限管理**
     1. **权限分类**

权限分为设备权限，用户权限和管理员权限。

设备权限：通过设备认证后获得设备权限。

用户权限：用户PIN码验证通过后，获得用户权限，用户权限只作用于其所在的应用。

管理员权限：管理员PIN码验证通过后，获得管理员权限，管理员权限只作用于其所在的应用。

* + 1. **权限使用**

权限的使用遵循以下要求：

1. 设备权限仅用于创建应用、删除应用和修改设备认证密钥；
2. 创建和删除容器需要用户权限；
3. 创建文件的权限在创建应用时指定；
4. 文件的读写权限在创建文件时指定；
5. 容器内私钥的使用需要用户权限；
6. 用户PIN码和管理员PIN码均具有最大重试次数，在创建应用时设定。当验证PIN码错误次数达到最大重试次数后，PIN码即锁死；
7. 用户PIN码的解锁需要管理员权限；
8. 用户PIN码的修改需要用户权限，管理员PIN码的修改需要管理员权限。
   * 1. **设备认证**

必须通过设备认证后才能在设备内创建和删除应用。

设备认证使用分组密码算法和设备认证密钥进行。认证的流程如下：

1)被认证方调用SKF\_GenRandom函数从设备获取8字节随机数RND，并用0x00将其填充至密码算法的分块长度，组成数据块D0；

2)被认证方对D0加密，得到加密结果D1，并调用SKF\_DevAuth()，将D1发送至设备；

3)设备收到D1后，验证D1是否正确。正确则通过设备认证，否则设备认证失败。

* + 1. **PIN码安全要求**

1. PIN码长度不少于6个字节；
2. PIN码在设备和本接口之间的传输过程中应采取保护措施，防止PIN码泄露；
3. PIN码在设备中应安全存储，不可从设备中导出。
   1. **密钥安全要求**

密钥应遵循以下安全要求：

1. 设备内产生的随机数应为真随机数，应符合随机性检测的要求；
2. 设备内产生的会话密钥应使用随机数；
3. 设备内产生非对称密钥使用的素数应满足素性要求；
4. 设备内的密钥应具备有效的密钥保护机制防止解剖、探测和读取；
5. 设备内的密钥应按权限要求使用；
6. 除公钥外的密钥不能以明文形式出现在设备外；
7. 签名私钥必须在设备中产生；
8. 容器内的私钥不能以任何形式导出设备；
9. 删除容器时必须销毁该容器内所有的密钥。
   1. **设备抗攻击要求**

设备应具备抗侧信道、电压、频率、紫外线等攻击的能力。

**附录A  
（规范性附录）  
错误代码定义和说明**

|  |  |  |
| --- | --- | --- |
| 错误代码标识 | | |
| 宏描述 | 预定义值 | 说明 |
| SAR\_OK | 0x00000000 | 成功 |
| SAR\_Fail | 0x0A000001 | 失败 |
| SAR\_UnknownErr | 0x0A000002 | 异常错误 |
| SAR\_NotSupportYetErr | 0x0A000003 | 不支持的服务 |
| SAR\_FileErr | 0x0A000004 | 文件操作错误 |
| SAR\_InvalidHandleErr | 0x0A000005 | 无效的句柄 |
| SAR\_InvalidParamErr | 0x0A000006 | 无效的参数 |
| SAR\_ReadFileErr | 0x0A000007 | 读文件错误 |
| SAR\_WriteFileErr | 0x0A000008 | 写文件错误 |
| SAR\_NameLenErr | 0x0A000009 | 名称长度错误 |
| SAR\_KeyUsageErr | 0x0A00000A | 密钥用途错误 |
| SAR\_ModulusLenErr | 0x0A00000B | 模的长度错误 |
| SAR\_NotInitializeErr | 0x0A00000C | 未初始化 |
| SAR\_ObjErr | 0x0A00000D | 对象错误 |
| SAR\_MemoryErr | 0x0A00000E | 内存错误 |
| SAR\_TimeoutErr | 0x0A00000F | 超时 |
| SAR\_IndataLenErr | 0x0A000010 | 输入数据长度错误 |
| SAR\_IndataErr | 0x0A000011 | 输入数据错误 |
| SAR\_GenRandErr | 0x0A000012 | 生成随机数错误 |
| SAR\_HashObjErr | 0x0A000013 | HASH对象错 |
| SAR\_HashErr | 0x0A000014 | HASH运算错误 |
| SAR\_GenRsaKeyErr | 0x0A000015 | 产生RSA密钥错 |
| SAR\_RsaModulusLenErr | 0x0A000016 | RSA密钥模长错误 |
| SAR\_CspImprtPubKeyErr | 0x0A000017 | CSP服务导入公钥错误 |
| SAR\_RsaEncErr | 0x0A000018 | RSA加密错误 |
| SAR\_RsaDecErr | 0x0A000019 | RSA解密错误 |
| SAR\_HashNotEqualErr | 0x0A00001A | HASH值不相等 |
| SAR\_KeyNotFountErr | 0x0A00001B | 密钥未发现 |
| SAR\_CertNotFountErr | 0x0A00001C | 证书未发现 |
| SAR\_NotExportErr | 0x0A00001D | 对象未导出 |
| SAR\_DecryptPadErr | 0x0A00001E | 解密时做补丁错误 |
| SAR\_MacLenErr | 0x0A00001F | MAC长度错误 |
| SAR\_BUFFER\_TOO\_SMALL | 0x0A000020 | 缓冲区不足 |
| SAR\_KeyInfoTypeErr | 0x0A000021 | 密钥类型错误 |
| sar\_not\_eventerr | 0x0A000022 | 无事件错误 |
| SAR\_DEVICE\_REMOVED | 0x0A000023 | 设备已移除 |
| SAR\_PIN\_INCORRECT | 0x0A000024 | PIN不正确 |
| SAR\_PIN\_LOCKED | 0x0A000025 | PIN被锁死 |
| SAR\_PIN\_INVALID | 0x0A000026 | PIN无效 |
| SAR\_PIN\_LEN\_RANGE | 0x0A000027 | PIN长度错误 |
| SAR\_USER\_ALREADY\_LOGGED\_IN | 0x0A000028 | 用户已经登录 |
| SAR\_USER\_PIN\_NOT\_INITIALIZED | 0x0A000029 | 没有初始化用户口令 |
| SAR\_USER\_TYPE\_INVALID | 0x0A00002A | PIN类型错误 |
| SAR\_APPLICATION\_NAME\_INVALID | 0x0A00002B | 应用名称无效 |
| SAR\_APPLICATION\_EXISTS | 0x0A00002C | 应用已经存在 |
| SAR\_USER\_NOT\_LOGGED\_IN | 0x0A00002D | 用户没有登录 |
| SAR\_APPLICATION\_NOT\_EXISTS | 0x0A00002E | 应用不存在 |
| SAR\_FILE\_ALREADY\_EXIST | 0x0A00002F | 文件已经存在 |
| SAR\_NO\_ROOM | 0x0A000030 | 空间不足 |
| SAR\_FILE\_NOT\_EXIST | 0x0A000031 | 文件不存在 |
| SAR\_REACH\_MAX\_CONTAINER\_COUNT | 0x0A000032 | 已达到最大可管理容器数 |

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