

预植数字证书接口调用规范说明

(版本：30701.07—2023)

中金金融认证中心

2023 年 03 月 30 日

版权声明：本文档的版权属于中金金融认证中心，任何人或组织未经许可，不得擅自修改、拷贝或以其它方式使用本文档中的内容。

目录

第一章 范围.....	1
第二章 规范性引用文件	1
第三章 术语和定义	1
第四章 预植数字证书流程.....	2
1. 单一类型数字证书预植流程	2
2. 复合类型数字证书预植流程	3
3. 导入数字证书流程	5
第五章 预植数字证书接口规范	6
1. GetDllInfo	6
2. Initialize	6
3. WaitKeyEvent	6
4. ClearKey	7
5. GenerateKeyPairs.....	8
6. DoWithRSAPrivateKey	9
7. DoWithSM2PrivateKey4Sign	10
8. ImportSignCert.....	11
9. ImportEncryptCertAndPrivateKey	11
10. VerifyKey.....	12
11. GetSignCertSerialNumber	14
12. GetCert	14
13. Finish（可选实现接口）	15
14. Uninitialize	15

第六章 预植动态库消息结构和发送机制规范（可选）	16
1. 消息发送相关宏定义	16
2. 消息结构	16
3. 范例实现（该实现仅供参考）	17
第七章 加密证书私钥结构	18
1. RSA-1024 加密证书私钥结构	18
2. RSA-2048 加密证书私钥结构	18
3. SM2 加密证书私钥结构	19
第八章 X509 请求公钥、临时公钥及签名结构	19
1. RSA X509 请求公钥、临时公钥及签名结构	19
2. SM2 X509 请求公钥、临时公钥及签名结构	20
第九章 KeyID 密钥属性规则及结构	20
1. KeyID 密钥属性的结构	21

第一章 范围

本规范中，描述了预植工具预植数字证书时，所涉及接口的实现标准。

第二章 规范性引用文件

下列文件对于本文件的应用是必不可少的。凡是注日期的引用文件，仅注日期的版本适用于本文件，凡是不注日期的引用文件，其最新版本适用于本文件。

RFC 3447	PKCS #1: RSA Cryptography Standard
RFC 2986	PKCS #10: Certification Request Syntax Specification
GM/T 0002-2012	SM4 分组密码算法
GM/T 0003-2012 SM2	椭圆曲线公钥密码算法
GM/T 0004-2012 SM3	密码杂凑算法
GM/T 0009-2012 SM2	密码算法使用规范
GM/T 0010-2012 SM2	密码算法加密签名消息语法规范

第三章 术语和定义

数字证书

也称公钥证书，由证书认证机构(CA)签名的包含公开密钥拥有者信息、公开密钥、签发者信息、有效期以及扩展信息的一种数据结构。按用途可分为签名证书、加密证书。

公钥

非对称密码算法中可以公开的密钥。

私钥

非对称密码算法中，只能由拥有者使用的不公开密钥。

会话密钥

在本规范中，会话密钥特指由服务器端产生的对称密钥，用于保护加密证书私钥。

交互密钥

在本规范中，交互密钥特指由申请者产生的非对称密钥对，用于保护会话密钥。

椭圆曲线密码算法

基于有限域上椭圆曲线离散对数问题的非对称密码算法。

SM2 密码算法

一种椭圆曲线密码算法，密钥长度为 256 比特。

SM3 算法

一种杂凑算法，输出长度为 256 比特。

SM4 算法

一种分组密码算法，分组长度为 128 比特，密钥长度为 128 比特。

DES

Data Encryption Standard，是一种使用密钥加密的块密码。

3DES

Triple DES，三重数据加密算法块密码的通称。相当于对每个数据块应用 3 次 DES 加密算法。

ECB

Electronic Code Book Operation Mode，分组密码算法的一种工作模式，其特征是将明文分组直接作为算法的输入，对应的输出作为密文分组。

第四章 预植数字证书流程

1. 单一类型数字证书预植流程

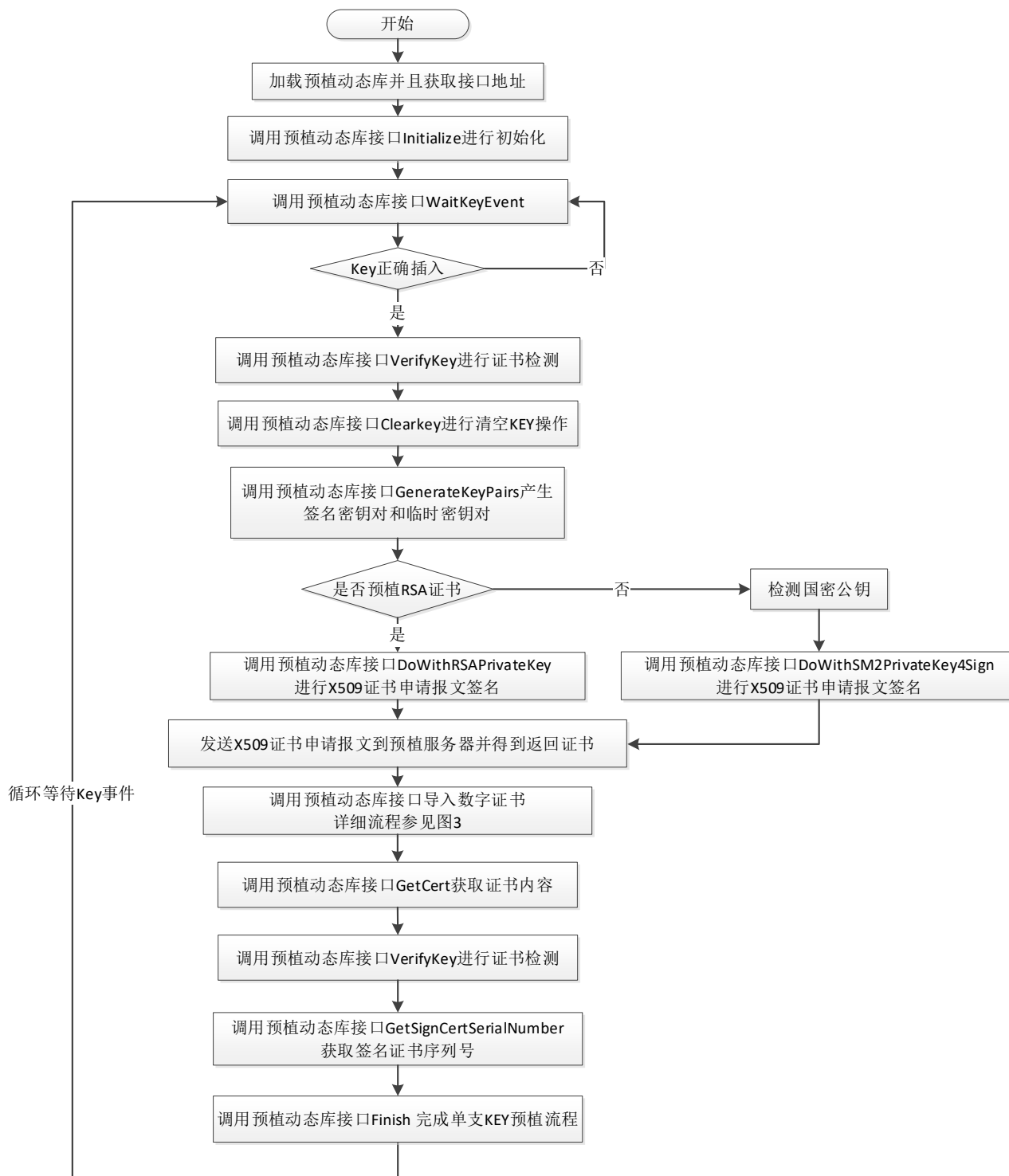


图 1 单一类型数字证书预植流程

2. 复合类型数字证书预植流程

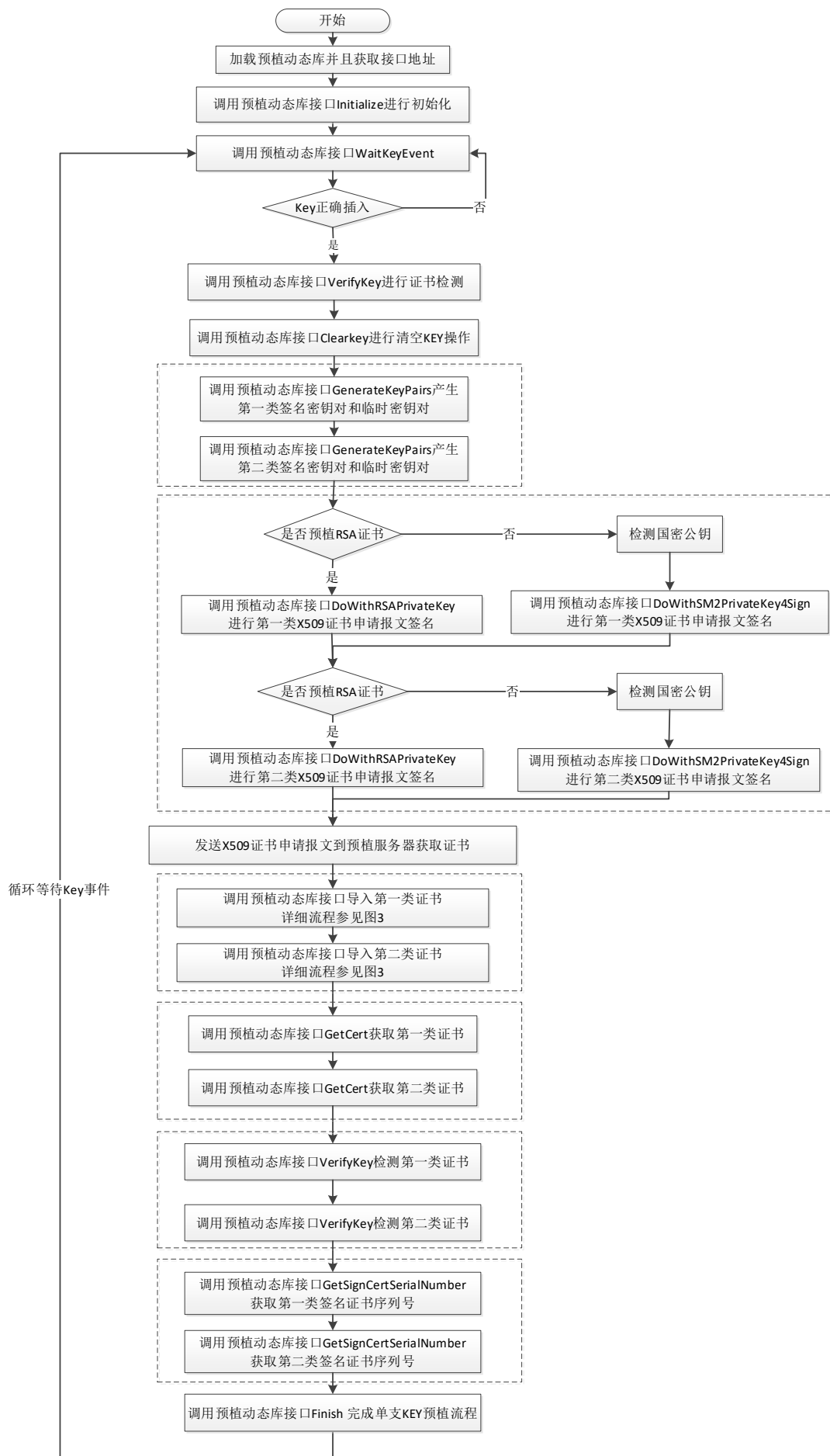


图 2 复合类型数字证书预植流程

3. 导入数字证书流程

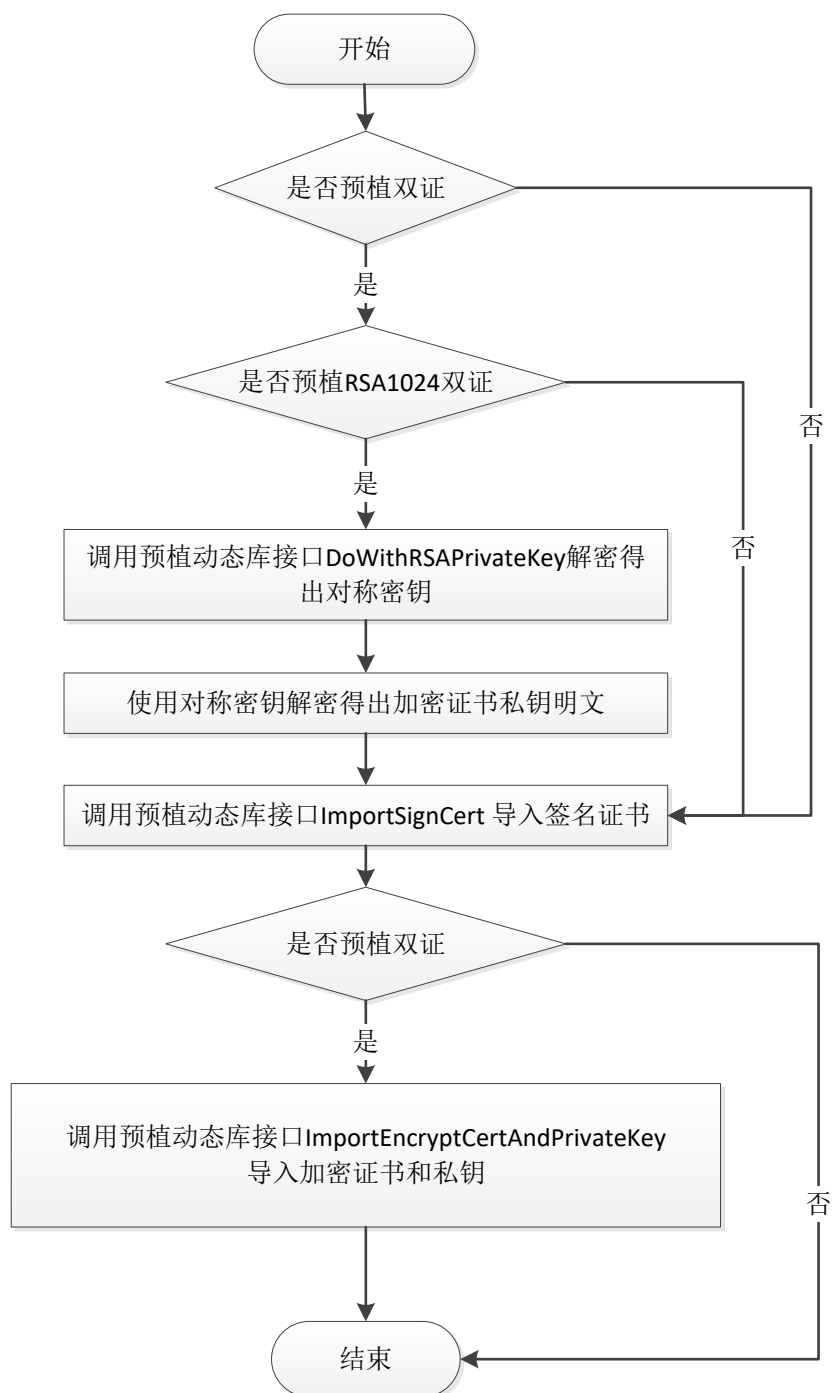


图 3 导入数字证书流程

第五章 预植数字证书接口规范

1. GetDllInfo

```
BOOL GetDllInfo( __out char *pszKeyCompany,  
                 __out char *pszKeyType,  
                 __out char *pszDllVersion)
```

描述：

获取该 DLL 文件对应的相关信息。

参数说明：

pszKeyCompany：Key 商名称(预植工具负责分配内存，长度不超过 256 字节)。

pszKeyType：Key 类型(预植工具负责分配内存，长度不超过 256 字节)。

pszDllVersion：DLL 版本号(预植工具负责分配内存，长度不超过 128 字节)。

备注：因涉及中文字符，特指定传出字符串为 UTF-8 编码，且调用该函数之前不需要调用 Initialize 函数。

2. Initialize

```
BOOL Initialize ( in     DWORD     dwMainThreadId)
```

描述：预植工具加载预植动态库之后调用，用于初始化后续其它预植动态库提供函数所需资源及环境。为最先调用的函数。

参数说明：

dwMainThreadId：主线程的线程 id，供动态库发送消息使用。消息格式和发送机制详见章节 6。

3. WaitKeyEvent

```
long WaitKeyEvent ( __out char *pszKeyId,
```

```

__out long *pUsbPort,
__out char *pszKeyCompany,
__out char *pszKeyType)

```

描述：

监听 USB Key 事件，可以根据返回值确定插入的 Key 是否为某个特定厂商所有，并返回 Key 厂商 的名称和 Key 的类型以及 KeyID 和对应的 USB 端口号。本函数是阻塞式调用，即没有事件发生时该函数不返回。

参数说明：

pszKeyId: KeyID（预植客户端负责分配内存，大小为 64 字节，Key 厂商预植 dll 通过该内存将不带密钥属性的 KeyID 输出，example：CFCAKey0000000000）。

pUsbPort: USB 端口地址。

pszKeyCompany: Key 厂商的名称（预植客户端负责分配内存，长度不超过 256 字节。因涉及中文字符，特指定传出字符串编码格式为 UTF-8）。

pszKeyType: Key 的类型（预植客户端负责分配内存，长度不超过 256 字节。因涉及中文字符，特指定 传出字符串编码格式为 UTF-8）。

返回值说明：

0：插入失败（插入的 Key 不属于此厂商）。

1：插入成功（插入的 Key 属于此厂商）。

2：拔除事件。

4. ClearKey

```

BOOL ClearKey(    in      char*   pszKeyId,
                  in      int      iUsbPort)

```

描述：

强制删除 Key 中所有的密钥对、证书和容器。保证预植进行之前 key 已清空。

参数说明：

pszKeyId: 需要清空数据的 Key 的 KeyID, 该 KeyID 为接口 WaitKeyEvent 返回的 KeyID。

iUsbPort: 需要清空数据的 Key 的 USB 端口号。

5. GenerateKeyPairs

```

BOOL GenerateKeyPairs( __in char* pszKeyId,
                      __in int iUsbPort,
                      __out char* pSignPublicKeyData,
                      __out int* piSignPublicKeyDataSize,
                      __in int tempKeyPairBitLength,
                      __out char* pTempPublicKeyData,
                      __out int* piTempPublicKeyDataSize)

```

描述:

从 Key 中获取 X509 公钥和临时公钥 (AT_KEYEXCHANGE 针对双证)。

参数说明:

pszKeyId: 由接口 WaitKeyEvent 返回的 KeyID 及追加上的 16 字节密钥对属性信息组成的值, 密钥对属性信息用于指定产生密钥对类型、密钥对存放容器索引及密钥对和证书存放的区域。该密钥对属性信息由预植工具指定。密钥对属性信息详见章节 9。

iUsbPort: USB 端口地址。

pSignPublicKeyData: X509 证书公钥 (此处仅返回公钥), 预植客户端负责分配内存, 分配内存大小为 2048 字节。X509 签名公钥结构详见章节 8。

piSignPublicKeyDataSize: X509 证书公钥长度。

tempKeyPairBitLength: 指定产生临时公钥的长度 (现可指定为 1024、2048 和 256(SM2 密钥), 现在使用的临时密钥对只有 RSA1024 (RSA1024 双证和 RSA2048 双证使用) 和 SM2 256 (SM2 证书使用) 两种类型, 且该参数实际意义是预植工具在上层分配存储临时密钥对公钥空间的一个最大长度。因此现在预植工具固定传入 1024 长度。当 Key 为单证时该参数无意义。)

pTempPublicKeyData: X509 临时公钥 AT_KEYEXCHANGE (此处仅返回公钥), 预植客户端

负责分配内存，分配内存大小为 2048 字节（当 Key 为单证时此参数无用，传入为 NULL。当 Key 为双证时，需校验该参数不为 NULL。）X509 临时公钥结构详见章节 8。

piTempPublicKeyDataSize：X509 临时公钥长度。

备注：该接口用于产生签名证书密钥对和临时密钥对，其中临时密钥对用于双证预植流程。预植双证时，临时密钥公钥作为参数之一被设置到 P10 证书申请中，CA 将使用该临时公钥加密加密证书密钥对；双证下载完成后，导入加密证书和加密证书私钥时，该密钥对私钥用于解密加密证书私钥密文。

6. DoWithRSAPrivateKey

```

BOOL DoWithRSAPrivateKey ( __in char* pszKeyId,
                           __in int iUsbPort,
                           __in char* pInputData,
                           __in int iInputDataLen,
                           __in int iPrivateKeyFlag,
                           __out char* pOutputData,
                           __out int* piOutputDataLen)

```

描述：

返回由 Key 中 RSA 加密证书私钥或者签名证书私钥（由参数 iPrivateKeyFlag 指定）解密或者签名(包含填充部分)的数据。

参数说明：

pszKeyId：由接口 WaitKeyEvent 返回的 KeyID 及追加上的 16 字节密钥对属性信息组成的值，密钥对属性信息用于指定产生密钥对类型、密钥对存放容器索引及密钥对和证书存放的区域。该密钥对属性信息由预植工具指定。密钥对属性信息详见章节 9。

iUsbPort：USB 端口地址。

pInputData：传入的源数据（如传入为待签名数据即 iPrivateKeyFlag 为 0，则该数据为哈希后的数据，且已加上哈希头和填充位。此处传入的数据是经过加头和补位的数据。Hash 数据的格式为标准的 pkcs1 补位。Hash 补位数据格式为：0x00 || 0x01 || PS || 0x00 || T）。

iInputDataLen：传入源数据的长度。

iPrivateKeyFlag: 私钥标识 (0 表示签名证书私钥, 1 表示加密证书私钥)。

pOutputData: RSA 解密或者签名后得出的数据 (预植工具负责分配内存, 分配大小为 1024 字节, 如果为 RSA 解密后得到的数据, 传出数据需包含填充位, 预植工具负责去掉填充位)
RSA X509 签名结构请参见章节 8。

piOutputDataLen: RSA 解密或者签名后得出的数据长度。

备注: 预植 RSA1024 双证时, 解密加密证书私钥时需要调用该接口完成会话密钥的解密, 此时需要将 iPrivateKeyFlag 设置为 1。

为保障 Key 的安全性, 该接口用于签名时, 无论签名是否成功, 只能被调用一次。如果需要再次通过该接口进行签名, 则需要首先通过接口 ClearKey 清空 Key, 并再次通过接口 GenerateKeyPairs 重新生成签名密钥。

7. DoWithSM2PrivateKey4Sign

```
BOOL DoWithSM2PrivateKey4Sign( __in char* pszKeyId,
                                __in int iUsbPort,
                                __in char* pInputData,
                                __in int iInputDataLen,
                                __out char* pOutputData,
                                __out int* piOutputDataLen)
```

描述:

返回由 Key 中 SM2 签名证书私钥签名的数据。在构建 SM2 X509 请求数据时则需要该接口来运算得出 X509 请求的签名数据。

参数说明:

pszKeyId: 由接口 WaitKeyEvent 返回的 KeyID 及追加上的 16 字节密钥对属性信息组成的值, 密钥对属性信息用于指定产生密钥对类型、密钥对存放容器索引及密钥对和证书存放的区域。该密钥对属性信息由预植工具指定。密钥对属性信息详见章节 9。

iUsbPort: USB 端口地址。

pInputData: 传入的待签名数据 (传入数据为带 Z 值的 SM3 哈希数据)。

iInputDataLen：待签名数据长度。

pOutputData：传出的签名过后的数据（由预植客户端负责分配内存，分配内存大小为 1024 字节）SM2 X509 签名结构请参见章节 8。

piOutputDataLen：传出的签名过后的数据长度。

备注：为保障 Key 的安全性，无论签名是否成功，该接口只能被调用一次。如果需要再次通过该接口进行签名，则需要首先通过接口 ClearKey 清空 Key，并再次通过接口 GenerateKeyPairs 重新生成签名密钥。

8. ImportSignCert

```

BOOL    ImportSignCert (
                                in    char*    pszKeyId,
                                in    int      iUsbPort,
                                in    char*    pCertData,
                                in    int      iCertDataLen)

```

描述：

将签名证书数据(Base64 编码格式)导入到 keyID 和 USB 端口地址指定的 key 的指定区域，导入过程需要检测证书公钥是否与 Key 中签名密钥对的公钥一致。

参数说明：

pszKeyId：由接口 WaitKeyEvent 返回的 KeyID 及追加上的 16 字节密钥对属性信息组成的值，密钥对属性信息用于指定产生密钥对类型、密钥对存放容器索引及密钥对和证书存放的区域。该密钥对属性信息由预植工具指定。密钥对属性信息详见章节 9。

iUsbPort：USB 端口地址。

pCertData：签名证书数据(经过 Base64 编码)。

iCertDataLen：签名证书数据长度。

9. ImportEncryptCertAndPrivateKey

```

BOOL    ImportEncryptCertAndPrivateKey (
                                in    char*    pszKeyId,

```

```

        in    int        iUsbPort,
        in    char*      pCertData,
        in    int        iCertDataLen,
        in    char*      pPrivateKey,
        in    int        iPrivateKeyLen)

```

描述：

将加密证书数据(Base64 编码格式)及私钥数据导入到 keyID 和 USB 端口地址指定的 key 的指定区域中。

参数说明：

pszKeyld：由接口 WaitKeyEvent 返回的 KeyID 及追加上的 16 字节密钥对属性信息组成的值，密钥对属性信息用于指定产生密钥对类型、密钥对存放容器索引及密钥对和证书存放的区域。该密钥对属性信息由预植工具指定。密钥对属性信息详见章节 9。

iUsbPort：USB 端口地址。

pCertData：加密证书数据(经过 Base64 编码)。

iCertDataLen：加密证书数据长度。

pPrivateKey：加密证书私钥数据（具体结构请参见章节 7）。

iPrivateKeyLen：加密证书私钥长度。

10. VerifyKey

```

long    VerifyKey (        in        char* pszKeyld,
                           in        int iUsbPort)

```

描述：

通过给定的 Keyld 和 USB 端口号，校验指定的 KEY 中的证书是否合法。

参数说明：

pszKeyld：由接口 WaitKeyEvent 返回的 KeyID 及追加上的 16 字节密钥对属性信息组成的值，密钥对属性信息用于指定产生密钥对类型、密钥对存放容器索引及密钥对和证书存放的

区域。该密钥对属性信息由预植工具指定。密钥对属性信息详见章节 9。

iUsbPort: USB 端口地址。

返回值说明：

- ①0x00000001 表示证书 Key 内已有的证书（双证签名加密证书都需要检测）的 CN 的内容是否与本身的 KeyID 是否相符。（相符为 0x00000000，不相符为 0x00000001）。
- ②0x00000010 表示 Key 中证书数目是否正确,单证只有签名证书，双证有签名证书、加密证书。（成功为 0x00000000，失败为 0x00000010）。
- ③0x00000100 表示 Key 中密钥数目是否正确,单证只有签名证书公钥，双证有签名证书公钥、加密 证书公私钥。（成功为 0x00000000，失败为 0x00000100）。
- ④0x00001000 表示用 Key 中签名证书签名、验签是否成功（成功为 0x00000000，失败为 0x00001000）。
- ⑤0x00010000 表示用 Key 中加密证书加密并用其私钥解密是否通过，单证返回成功（成功为 0x00000000， 失败为 0x00010000）。
- ⑥0x00100000 表示 Key 中的指定区域的所有证书的证书用途是否都符合预期（成功为 0x00000000，失败为 0x00100000）。

该函数返回值为上述六个检测项结果“相或”(|)的结果，成功则返回值为 x00000000，检测内容如下：

- 1) 检查 key 内已有的证书的 CN 内容是否与本身的 KeyID 相符，单证只需要检测签名证书，双证需检测签名和加密证书；
- 2) 检查 Key 中证书数目是否正确，单证只有一张签名证书，双证有签名证书和加密证书；
- 3) 检查 key 中密钥对数目是否正确，单证只有签名证书公私钥，双证需要有签名证书公私钥、加 密证书公私钥；
- 4) 检查用 Key 中签名证书签名、验签是否成功，主要是检测签名证书公私钥的完整与正确性；
- 5) 检查双证 Key 中加密证书加密并用私钥解密过程是否通过，主要是检测加密证书公私钥是否完 整、正确、匹配。
- 6) 检查 key 中指定区域的所有证书的证书用途是否都符合预期。

11. GetSignCertSerialNumber

```

BOOL GetSignCertSerialNumber (
    in    char*    pszKeyId,
    in    int      iUsbPort,
    out   char*    pbyCertSerialNumber,
    out   int*     iCertSerialNumberLen)

```

描述：

通过给定的 KeyID 和 Key 端口号，返回签名证书序列号。

参数说明：

pszKeyId：由接口 WaitKeyEvent 返回的 KeyID 及追加上的 16 字节密钥对属性信息组成的值，密钥对属性信息用于指定产生密钥对类型、密钥对存放容器索引及密钥对和证书存放的区域。该密钥对属性信息由预植工具指定。密钥对属性信息详见章节 9。

iUsbPort：USB 端口地址。

pbyCertSerialNumber：证书序列号，预植工具负责分配内存，分配大小为 64 字节（传出证书序列号字节流而非字符串）。

iCertSerialNumberLen:证书序列号长度。

12. GetCert

```

BOOL GetCert ( __in char* pszKeyId,
               __in int iUsbPort,
               __out char* pbySignCert,
               __out int* iSignCertSize,
               __out char* pbyEncryCert,
               __out int* iEncryCertSize)

```

描述：

通过给定的 KeyID 和 Key 端口号，返回对应的签名证书和加密证书。

参数说明：

pszKeyId：由接口 WaitKeyEvent 返回的 KeyID 及追加上的 16 字节密钥对属性信息组成的值，密钥对属性信息用于指定产生密钥对类型、密钥对存放容器索引及密钥对和证书存放的区域。该密钥对属性信息由预植工具指定。密钥对属性信息详见章节 9。

iUsbPort：USB 端口地址。

pbySignCert：签名证书字节流（DER 编码），预植工具分配内存，分配大小为 4096 字节。

iSignCertSize:签名证书长度。

pbyEncryCert：加密证书字节流（DER 编码，如果预植为单证传出值为 NULL），预植工具分配内存，分配大小为 4096 字节。

iEncryCertSize:加密证书长度（如果预植单证，传出值为 0）。

13. Finish（可选实现接口）

```
BOOL Finish (
    in      char*   pszKeyId,
    in      int      iUsbPort)
```

描述：

通过给定的 KeyID 和 Key 端口号，通知预植动态库指定 key 的预植证书流程已经结束。

参数说明：

pszKeyId：通知预植流程结束的 key 的 keyid，该值为接口 WaitKeyEvent 返回的 KeyID。

iUsbPort：USB 端口地址。

14. Uninitialize

```
BOOL Uninitialize (void)
```

描述：

该函数用来释放在 Initialize 调用时所分配的资源

第六章 预植动态库消息结构和发送机制规范（可选）

为记录 Key 在预植中产生的错误信息，以便在 Key 预植过程出错后定位问题，现制定消息结构和发送机制如下：（该功能为可选功能，但为了方便定位及解决问题强烈建议 Key 商实现该功能）

1. 消息发送相关宏定义

①`#define WORK_THREAD_MSG WM_USER+0x101`：该宏作为主消息类型用作在 `PostThreadMessage` 发送线程消息时指明消息类型，以便主线程捕获。

②`#define MSG_CUR_USB_PORT_MSG WM_USER+0x200`：该宏作为子消息类型，主要用作区分该类消息为 DLL 库发出的消息类型。

③`#define WORK_THREAD 0x200`

④`#define MAIN_THREAD 0x100`

⑤`#define MAX_MESSAGE_LEN 512` //单条消息最大长度

2. 消息结构

`struct USER_MSG`

{

`int msg_message;` // 消息类型，对于 DLL 库消息类型为：
`MSG_CUR_USB_PORT_MSG`;

`int msg_origin;` //消息源，对于 DLL 库消息源为：`WORK_THREAD`

`int msg_destin;` //消息目的，对于 DLL 库消息目的为：`MAIN_THREAD`

`int msg_length;` //消息长度，该长度指定了随后的附带消息数据长度，每条消息长度最大为 `MAX_MESSAGE_LEN`

};

注：完整的消息格式如为：消息结构 (`USER_MSG`) + 消息内容(`USER_MSG` 中 `msg_length` 字段指定了该消息内容长度)

3. 范例实现（该实现仅供参考）

```
void SendCurUsbPortMsg(char *pMsg)
```

```
{
```

```
    int len = strlen(pMsg);
```

```
    if (len > MAX_MESSAGE_LEN)
```

```
        return;
```

```
    USER_MSG *p;
```

p = (USER_MSG *) NewMsgMem(sizeof(USER_MSG)+len);//NewMsgMem 函数是用来在堆中分配一块内存来存放消息，该函数由 Key 商自己实现，且此内存由 Key 商来管理，此处仅为范例说明。

```
    if (p!=NULL)
```

```
    {
```

```
        p->msg_destin = MAIN_THREAD;
```

p->msg_message = MSG_CUR_USB_PORT_MSG;//dll 发送的子消息类型，全部是这种，用来显示某一个 usb 口上预植证书操作的当前信息

```
        p->msg_length = len;
```

```
        p->msg_origin = WORK_THREAD;
```

```
        if (pMsg != NULL)
```

```
        {
```

memcpy((char *)((char *)p+sizeof(USER_MSG)),pMsg,len);//往消息结构头后面附带实际需要显示的消息

```
        }
```

::PostThreadMessage(m_nThreadId,WORK_THREAD_MSG,iCurUsbPort,(LONG)p);//发送消息到主线程，其中 m_nThreadId 为主线程 ID 在 Initialize 函数中传入，WORK_THREAD_MSG 为主消息类型，iCurUsbPort 用来指出当前的 usb 口的顺序号，处理时需注意，p 为全部消息内容

```

    }
}

```

第七章 加密证书私钥结构

1. RSA-1024 加密证书私钥结构

RSA-1024 加密证书私钥需以明文形式导入，现以 PKCS#1 私钥结构导入。

2. RSA-2048 加密证书私钥结构

RSA-2048 加密证书私钥需以加密形式导入，私钥结构为自定义的 ASN.1 格式（DER 编码），TAG 值为 0x30，内容如下：

```

EncryptedPrivateKey ::= SEQUENCE {
    Version INTEGER,                                --版本号
    AsymAlgID OBJECT IDENTIFIER,                    --非对称加密算法标识符
    SymAlgID OBJECT IDENTIFIER,                     --对称加密算法标识符
    EncryptedSymKey OCTET STRING,                   --被加密的对称密钥
    EncryptedPrivateKey OCTET STRING                --用对称密钥加密过的私钥
}

```

} 其中： Version：版本号，在本文档中取值为 1。

AsymAlgID：RSA 加密算法标识符，取值为 1.2.840.113549.1.1.1。

SymAlgID：对称加密算法标识符，这里为 3DES ECB 对称加密算法，取值为 1.3.6.1.4.1.4929.1.7。

EncryptedSymKey：被加密的对称密钥，格式请参考 PKCS#1 的 RSA 加密结果。

EncryptedPrivateKey：用对称密钥加密过的私钥，经对称密钥解密后应为 PKCS#1 标准私钥结构，格式请参考 PKCS#1 标准私钥结构。

3. SM2 加密证书私钥结构

SM2 加密证书私钥需以加密形式导入，私钥结构为自定义的 ASN.1 格式（DER 编码），TAG 值为

0x04，内容如下：

OCTET STRING EncryptedPrivateKey

其中： EncryptedPrivateKey：加密私钥密文。

备注：

1、加密私钥密文 EncryptedPrivateKey 存在两种格式：

一种为 C1||C2||C3（国密老的标准），另一种为 C1||C3||C2（国密最新标准）。CSP 与 PKCS#11 需要做出调整，也能够自动兼容上述两种私钥密文格式。

其中 C1(x,y)分别为 32 字节曲线点分量,C2 为加密的数据,C3 为 32 字节 SM3 杂凑值。

2、解密后的 SM2 密钥对为 x||y||d,其中 x, y 是 32 字节的公钥坐标点，d 是 32 字节的私钥。

第八章 X509 请求公钥、临时公钥及签名结构

1. RSA X509 请求公钥、临时公钥及签名结构

RSA X509 公钥结构（RSA X509 临时公钥结构与公钥结构相同）：

PublicKey ::= SEQUENCE {

 n INTEGER, --公钥 n 值

 e INTEGER --公钥 e 值

}

RSA X509 签名结构

RSA 签名结构根据模长返回对应的签名值,即 RSA-1024 返回 128 字节的签名值，RSA-2048 返回 256 字节的签名值（大端）。

2. SM2 X509 请求公钥、临时公钥及签名结构

SM2 X509 公钥结构（SM2 临时公钥结构与公钥结构相同）：

```
PublicKey ::= SEQUENCE {
    x INTEGER,           --公钥 XCoordinate 值
    y INTEGER            --公钥 YCoordinate 值
}
```

备注：

- 1、x 或 y 数值不足 256 位的，在高位补 0 填充至 256 位。
- 2、x 和 y 数值达到 256 位，按照 ASN.1 编码（如果高位为 1，补一个字节 0 以表示正数；反之，不用补 0）
- 3、针对调用 Generatekeypairs 产生的 32 字节的 x 和 y 公钥坐标点，要求第一个字节的值不能为 0。

SM2 X509 签名结构：

```
Signature ::= SEQUENCE {
    r INTEGER,           --签名 r 值
    s INTEGER            --签名 s 值
}
```

备注：

- 1、r 或 s 数值不足 256 位的，在高位补 0 填充至 256 位。
- 2、r 和 s 数值达到 256 位，按照 ASN.1 编码（如果高位为 1，补一个字节 0 以表示正数；反之，不用补 0）。

第九章 KeyID 密钥属性规则及结构

1. KeyID 密钥属性的结构

证书预植过程中 CFCA 预植工具向 key 传入的 keyID 总长度为 32 个字符，其中前 16 个字符为调用 WaitKeyEvent 时返回的 pszKeyId，为从 key 中读取的字符串。该字符串中的第七位仅表示证书的类型为个人证书或是企业证书，不再具有其它含义。后面的 16 个字符为密钥属性信息。针对密钥属性信息，现在使用前 3 个字符表示生成的密钥对的存放位置、密钥类型及密钥区域。第一个字符表示密钥对生成后存放的容器的 Index；第二个字符表示生成证书的类型；第三个字符标识密钥对及证书存放的区域，0 表示密钥及证书存储在普通区域，1 表示密钥及证书存储在业务区域。其余 13 位暂时为保留位。预植工具与预植动态库的交互过程中不会涉及到容器名称的处理，工具只会传送容器的索引值、存放于该容器的密钥对类型及区域编号。预植动态库需要保证根据预植工具传入的容器索引及区域编码对指定区域的容器进行操作返回结果即可。

密钥属性用 16 个字符来表示，如下表：

1	2	3	4-16
Index(0 至 9)	密钥对类型，详见下	密钥对区域	

备注：

1、Index 为存放密钥对和证书的容器 Index，之后使用该 Index 导入证书至指定容器或获取指定容器证书。

2、部分银行需要将业务处理密钥对及证书导入至 Key 中特定区域，因此需要在 Key 中设定不同区域用于存放不同用途的密钥及证书。区域 0 表示密钥及证书存储在普通区域，区域 1 表示密钥及证书存储在业务区域。

3、当前用到前 3 个字符，其他 13 个字符为预留属性默认填 0。

如：pwszKeyIdAndAttributes1=CFCAKey00000000000100000000000000 第 17 位为 0,表示选定 Index 为 0 的容器，第 18 位为 1 表示证书的类型为 RSA1024 单证，第 19 为 0 标示证书存储在 Key 中 A 区域。

pwszKeyIdAndAttributes2=CFCAKey00000000001E1000000000000000 第 17 位为 1,表示选定 Index 为 1 的容器，第 18 位为 E 表示证书的类型为国密双证，第 19 为 1 标示证书存储在 Key 中 B 区域。

密钥对类型信息：

标识	算法名称	密钥长度	证书类型
----	------	------	------

0	RSA	1024	单证
1	RSA	1024	单证
2	RSA	2048	单证
3	RSA	2048	单证
4	SM2	256	单证
5	SM2	256	单证
6	预留	预留	预留
7	预留	预留	预留
8	预留	预留	预留
9	RSA	1024	单证
A	RSA	2048	单证
B	SM2	256	单证
C	RSA	1024	双证
D	RSA	2048	双证
E	SM2	256	双证
F	RSA	1024	双证
G	RSA	2048	双证
H	SM2	256	双证

此后的字母为预留字符，“I”和“O”排除在外。