

CFCA

中金金融认证中心标准

30006.01—2013

RSA 数字证书申请及应用 PKCS#11 接口调用规范

Interface specification of PKCS#11

for RSA certificate enrollment and application

2013-06-01 发布

2013-06-01 实施

中金金融认证中心

发布

目 录

1. 范围 1

2. 规范性引用文件..... 1

3. 术语和定义 1

4. PKCS#11 接口版本 2

5. 容器、证书、密钥区分..... 2

6. RSA 数字证书申请调用 PKCS#11 接口规范 2

 6.1 RSA 数字证书申请流程 2

 6.2 PKCS#11 接口描述..... 3

7. RSA 数字证书导入调用 PKCS#11 接口规范 5

 7.1 RSA 签名证书导入 5

 7.2 RSA 1024 位加密证书及私钥导入流程..... 6

 7.3 RSA 2048/4096 位加密证书及私钥导入流程..... 9

8. RSA 2048/4096 位加密证书对应的私钥结构定义..... 11

9. PIN 码框..... 12

10. 代码签名 12

RSA 数字证书申请及应用 PKCS#11 接口调用规范

1. 范围

本规范中，描述了通过 PKCS#11 接口实现 RSA 数字证书申请及应用时，所涉及接口的实现标准。本规范中未涉及的接口，请参照 PKCS#11 标准。

2. 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。

PKCS#11 v2.20 Cryptographic Token Interface Standard

3. 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。

对称密码算法

加密和解密使用相同密钥的密码算法。

对称密钥

用于对称密码算法的密钥。

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。

会话密钥

在本规范中会话密钥特指，由服务器端产生，用于保护从服务器端返回的加密证书私钥的 3DES 算法对称密钥。

交互密钥

在本规范中交互密钥特指，由客户端产生，用于保护会话密钥的 RSA 非对称密钥对。

4. PKCS#11 接口版本

厂商实现 P11 库时使用的头文件版本，必须使用 CFCA 修订的版本，请联系 CFCA 索取。

5. 容器、证书、密钥区分

CFCA 使用 CKA_LABEL 和 CKA_ID 两个属性字段来区分不同的容器、证书、密钥：

1. CFCA 在产生 PKCS#10 请求时会生成一串 UUID，并以该 UUID 作为容器名称来产生密钥对，并将该容器名称返回给上层应用。
2. 将容器名称作为 CKA_LABEL 属性设置到对应的证书、密钥上。
3. 将容器名称附加“#1”作为 CKA_ID 属性来标识签名证书、密钥；
将容器名称附加“#2”作为 CKA_ID 属性来标识加密证书、密钥。

6. RSA 数字证书申请调用 PKCS#11 接口规范

6.1 RSA 数字证书申请流程

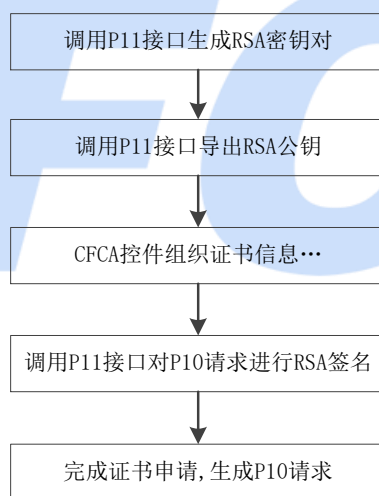


图 1 RSA 数字证书申请流程

RSA 数字证书申请流程如下：

- 1、调用 PKCS#11 接口，生成 RSA 密钥对。若为双证书请求，则需生成两对密钥对。
本步骤中需调用的接口：C_GeneratekeyPair。
- 2、调用 PKCS#11 接口，导出 RSA 公钥。若为双证书请求，应导出两个公钥。
本步骤中需调用的接口：C_GetAttributeValue。
- 3、由 CFCA 控件，组织生成证书信息。
- 4、调用 PKCS#11 接口，对 PKCS#10 请求进行 RSA 签名。
本步骤中需调用的接口：C_SignInit、C_Sign。
- 5、由 CFCA 控件完成证书申请，生成 PKCS#10 请求。

6.2 PKCS#11 接口描述

6.2.1 C_GenerateKeyPair

```
CK_RV C_GenerateKeyPair( CK_SESSION_HANDLE hSession,
                          CK_MECHANISM_PTR pMechanism,
                          CK_ATTRIBUTE_PTR pPublicKeyTemplate,
                          CK_ULONG ulPublicKeyAttributeCount,
                          CK_ATTRIBUTE_PTR pPrivateKeyTemplate,
                          CK_ULONG ulPrivateKeyAttributeCount,
                          CK_OBJECT_HANDLE_PTR phPublicKey,
                          CK_OBJECT_HANDLE_PTR phPrivateKey)
```

描述：产生密钥对。

特殊参数取值说明：pMechanism：机制类型为：CKM_RSA_PKCS_KEY_PAIR_GEN

pPublicKeyTemplate：创建 RSA 密钥对时，使用的公钥模板。

pPrivateKeyTemplate：创建 RSA 密钥对时，使用的私钥模板。

备注：

1、创建 RSA 密钥对时的公钥模板：

```
CKO_OBJECT_CLASS    ulPubkeyClass    = CKO_PUBLIC_KEY;
CK_BBOOL            bTrue            = TRUE;
CK_KEY_TYPE         ulKeyType        = CKK_RSA;
{CKA_CLASS,         &ulPubkeyClass,   sizeof(ulPubkeyClass)},
{CKA_TOKEN,         &bTrue,           sizeof(bTrue)},
{CKA_KEY_TYPE,      &ulKeyType,       sizeof(ulKeyType)},
{CKA_MODIFIABLE,    &bTrue,           sizeof(bTrue)},
{CKA_ID,            byCkID,           strlen((char *)byCkID)},
{CKA_LABEL,         pszContainerName, strlen((char *)pszContainerName)},
{CKA_ENCRYPT,        &bEncryDecry,     sizeof(bEncryDecry)},
{CKA_VERIFY,        &bSignVerify,     sizeof(bSignVerify)},
{CKA_WRAP,          &bTrue,           sizeof(bTrue)},
{CKA_PRIVATE,       &bFalse,          sizeof(bFalse)},
{CKA_MODULUS_BITS,  &ulModulusBits,   sizeof(ulModulusBits)},
{CKA_PUBLIC_EXPONENT, pbyPubKeyExponent, ulPubKeyExponentSize}
```

其中：

CKA_ID：取值为“容器名#1”或“容器名#2”，

例如：9125758C-60F3-45B7-8552-7BD1FDFA2B8F#1

CKA_LABEL：取值为容器名，即 UUID，例如：9125758C-60F3-45B7-8552-7BD1FDFA2B8F

CKA_ENCRYPT：设置密钥用途，需根据签名、加密公钥类型不同，设置为对应的值。

CKA_VERIFY: 设置密钥用途, 需根据签名、加密公钥类型不同, 设置为对应的值。

2、创建 RSA 密钥对时的私钥模板:

```
CKO_OBJECT_CLASS ulPrikeyClass          = CKO_PRIVATE_KEY;
CK_BBOOL          bTrue                  = TRUE;
CK_KEY_TYPE        ulKeyType              = CKK_RSA;
{CKA_CLASS,        &ulPrikeyClass,        sizeof(ulPrikeyClass)},
{CKA_TOKEN,        &bTrue,                sizeof(bTrue)},
{CKA_KEY_TYPE,     &ulKeyType,            sizeof(ulKeyType)},
{CKA_MODIFIABLE,   &bTrue,                sizeof(bTrue)},
{CKA_ID,           byCkID,                strlen((char *)byCkID)},
{CKA_LABEL,        pszContainerName,       strlen((char *)pszContainerName)},
{CKA_SENSITIVE,    &bTrue,                sizeof(bTrue)},
{CKA_SIGN,         &bSignVerify,          sizeof(bSignVerify)},
{CKA_DECRYPT,       &bEncryDecry,          sizeof(bEncryDecry)},
{CKA_UNWRAP,       &bTrue,                sizeof(bTrue)},
{CKA_PRIVATE,      &bTrue,                sizeof(bTrue)},
```

其中:

CKA_ID: 取值为“容器名#1”或“容器名#2”,

例如: 9125758C-60F3-45B7-8552-7BD1FDFA2B8F#1

CKA_LABEL: 取值为容器名, 即 UUID, 例如: 9125758C-60F3-45B7-8552-7BD1FDFA2B8F

CKA_DECRYPT: 设置密钥用途, 需根据签名、加密私钥类型不同, 设置为对应的值。

CKA_SIGN: 设置密钥用途, 需根据签名、加密私钥类型不同, 设置为对应的值。

6.2.2 C_GetAttributeValue

```
CK_RV C_GetAttributeValue(CK_SESSION_HANDLE hSession,
                           CK_OBJECT_HANDLE  hObject,
                           CK_ATTRIBUTE_PTR  pTemplate,
                           CK_ULONG          ulCount)
```

描述: 获取对象属性。

特殊参数取值说明: hObject: 导出公钥时, 该参数为公钥句柄。

pTemplate: 获取参数的模板。

备注:

1、导出公钥的 N 值时, 模板为: {CKA_MODULUS, NULL_PTR, 0}。

2、导出公钥的 E 值时, 模板为: {CKA_PUBLIC_EXPONENT, NULL_PTR, 0}。

6.2.3 C_SignInit

```
CK_RV C_SignInit(CK_SESSION_HANDLE hSession,
                  CK_MECHANISM_PTR pMechanism,
                  CK_OBJECT_HANDLE hKey);
```

描述： 签名初始化。

特殊参数取值说明： pMechanism： 在对 PKCS#10 申请签名时，至少支持：
CKM_SHA1_RSA_PKCS 和 CKM_MD5_RSA_PKCS。

6.2.4 C_Sign

```
CK_RV C_Sign(CK_SESSION_HANDLE hSession,
              CK_BYTE_PTR pData,
              CK_ULONG ulDataLen,
              CK_BYTE_PTR pSignature,
              CK_ULONG_PTR pulSignatureLen)
```

描述： 签名。

特殊参数取值说明： 无。

7. RSA 数字证书导入调用 PKCS#11 接口规范

7.1 RSA 签名证书导入

RSA 签名证书导入时，仅需导入签名公钥证书，只涉及一个 PKCS#11 接口： C_CreateObject。

7.1.1 PKCS#11 接口描述

7.1.1.1 C_CreateObject

```
CK_RV C_CreateObject(CK_SESSION_HANDLE hSession,
                     CK_ATTRIBUTE_PTR pTemplate,
                     CK_ULONG ulCount,
                     CK_OBJECT_HANDLE_PTR phObject)
```

描述： 创建对象。

特殊参数取值说明： pTemplate： 对象模板。

备注：

1、 导入 RSA 公钥证书时，模板取值如下：

```
CK_OBJECT_CLASS      ulCertificateClass      = CKO_CERTIFICATE;
CK_BBOOL              bTrue                  = TRUE;
```

CK_BBOOL	bFalse	= FALSE;
CK_CERTIFICATE_TYPE	ulCertificateType	= CKC_X_509;
{CKA_CLASS,	&ulCertificateClass,	sizeof(ulCertificateClass)},
{CKA_TOKEN,	&bTrue,	sizeof(bTrue)},
{CKA_LABEL,	pszContainerName,	strlen((char *)pszContainerName)},
{CKA_MODIFIABLE,	&bTrue,	sizeof(bTrue)},
{CKA_ID,	byCkID,	strlen((char *)byCkID)},
{CKA_CERTIFICATE_TYPE,	&ulCertificateType,	sizeof(ulCertificateType)},
{CKA_PRIVATE,	&bFalse,	sizeof(bFalse)},
{CKA_VALUE,	pbyCertificate,	ulCertificateSize}

其中：

CKA_LABEL：取值为证书所在的容器名。

CKA_ID：取值为证书对应的密钥对 ID，即“容器名#1”或“容器名#2”。详见章节 5。

7.2 RSA 1024 位加密证书及私钥导入流程

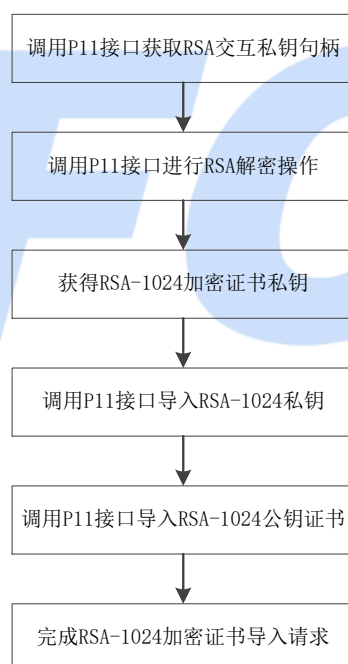


图 2 RSA 1024 位加密证书导入流程

RSA 1024 位加密证书导入过程中，需将 RSA 1024 位公钥证书及对应的私钥同时导入。在本文档中，RSA 1024 位加密证书私钥必须以明文形式导入 KEY 中。导入流程图见图 2，详细说明如下：

1、调用 PKCS#11 接口，根据 CKA_CLASS 和 CKA_ID 查找并获取 RSA 1024 位交互私钥句柄。

本步骤中需调用到的接口：C_FindObjectInit、C_FindObjects、C_FindObjectsFinal。

2、调用 PKCS#11 接口，利用步骤 1 中获取到的交互私钥句柄，进行解密。

本步骤中需调用到的接口：C_DecryptInit、C_Decrypt。

3、由 CFCA 控件获取 RSA 1024 位加密证书私钥明文。

4、调用 PKCS#11 接口，导入 RSA 1024 位加密证书私钥。

本步骤中需调用到的接口：C_CreateObject。

5、调用 PKCS#11 接口，导入 RSA 1024 位加密证书。

本步骤中需调用到的接口：C_CreateObject。

6、由 CFCA 控件完成 RSA 1024 位加密证书导入。

7.2.1 PKCS#11 接口描述

7.2.1.1 C_FindObjectsInit

CK_RV C_FindObjectsInit(CK_SESSION_HANDLE hSession,
CK_ATTRIBUTE_PTR pTemplate,
CK_ULONG ulCount)

描述： 查找对象初始化。

特殊参数取值说明： pTemplate: 查找私钥对象时，通过 CKA_CLASS、CKA_ID 来查找。

7.2.1.2 C_FindObjects

CK_RV C_FindObjects(CK_SESSION_HANDLE hSession,
CK_OBJECT_HANDLE_PTR phObject,
CK_ULONG ulMaxObjectCount,
CK_ULONG_PTR pulObjectCount)

描述： 查找对象。

特殊参数取值说明： 无。

7.2.1.3 C_FindObjectsFinal

CK_RV C_FindObjectsFinal(CK_SESSION_HANDLE hSession)

描述： 结束查找对象。

特殊参数取值说明： 无。

7.2.1.4 C_CreateObject

CK_RV C_CreateObject(CK_SESSION_HANDLE hSession,
CK_ATTRIBUTE_PTR pTemplate,
CK_ULONG ulCount,
CK_OBJECT_HANDLE_PTR phObject)

描述： 创建对象。

特殊参数取值说明： pTemplate: 创建对象时所用的模板，取值由导入对象决定。

备注：

- 1、创建私钥和公钥对象时须删除原私钥、公钥对象（通过 CKA_ID 来查找）。
- 2、在写入公钥时，KEY 需置 CKA_MODULUS_BITS 属性以便以后查看证书密钥长度。
- 3、RSA-1024 私钥以明文形式导入。模板详细信息如下：

CK_OBJECT_CLASS	ulPrikeyClass	= CKO_PRIVATE_KEY;
CK_KEY_TYPE	ulKeyType	= CKK_RSA;
CK_BBOOL	bTrue	= TRUE;
CK_BBOOL	bFalse	= FALSE;
{CKA_CLASS,	&ulPrikeyClass,	sizeof(ulPrikeyClass)},
{CKA_KEY_TYPE,	&ulKeyType,	sizeof(ulKeyType)},
{CKA_TOKEN,	&bTrue,	sizeof(bTrue)},
{CKA_LABEL,	pszContainerName,	strlen((char *)pszContainerName)},
{CKA_ID,	byCkID,	strlen((char *)byCkID)},
{CKA_MODIFIABLE,	&bTrue,	sizeof(bTrue)},
{CKA_SENSITIVE,	&bTrue,	sizeof(bTrue)},
{CKA_UNWRAP,	&bTrue,	sizeof(bTrue)},
{CKA_PRIVATE,	&bTrue,	sizeof(bTrue)},
{CKA_DECRYPT,	&bEncryDecry,	sizeof(bEncryDecry)},
{CKA_SIGN,	&bSignVerify,	sizeof(bSignVerify)},
{CKA_MODULUS,	pbyModulus,	ulModulusSize},
{CKA_PUBLIC_EXPONENT,	pbyPubKeyExponent,	ulPubKeyExponentSize},
{CKA_PRIME_1,	pbyPriKeyP,	ulPriKeyPSize},
{CKA_PRIME_2,	pbyPriKeyQ,	ulPriKeyQSize},
{CKA_PRIVATE_EXPONENT,	pbyPriKeyD,	ulPriKeyDSize},
{CKA_EXPONENT_1,	pbyPriKeyExponentP,	ulPriKeyExponentPSize},
{CKA_EXPONENT_2,	pbyPriKeyExponentQ,	ulPriKeyExponentQSize},
{CKA_COEFFICIENT,	pbyPriKeyCoefficient,	ulPriKeyCoefficientSize}

其中：

CKA_DECRYPT：设置密钥用途，需根据签名、加密私钥类型不同，设置为对应的值。

CKA_SIGN：设置密钥用途，需根据签名、加密私钥类型不同，设置为对应的值。

- 4、在导入私钥时会同时导入对应的公钥对象。通过设置 CKA_TOKEN 为 TRUE，并传入 N、E 值来写入公钥值。其模板信息如下：

CK_OBJECT_CLASS	ulPrikeyClass	= CKO_PUBLIC_KEY;
CK_KEY_TYPE	ulKeyType	= CKK_RSA;
CK_BBOOL	bTrue	= TRUE;
CK_BBOOL	bFalse	= FALSE;

{CKA_CLASS,	&ulPubkeyClass,	sizeof(ulPubkeyClass)},
{CKA_TOKEN,	&bTrue,	sizeof (bTrue)},
{CKA_KEY_TYPE,	&ulKeyType,	sizeof (ulKeyType)},
{CKA_MODIFIABLE,	&bTrue,	sizeof (bTrue)},
{CKA_ID,	byCkID,	strlen((char *)byCkID)},
{CKA_LABEL,	pszContainerName,	strlen((char *)pszContainerName)},
{CKA_ENCRYPT,	&bEncryDecry,	sizeof(bEncryDecry)},
{CKA_VERIFY,	&bSignVerify,	sizeof(bSignVerify)},
{CKA_WRAP,	&bTrue,	sizeof(bTrue)},
{CKA_PRIVATE,	&bFalse,	sizeof (bFalse)},
{CKA_MODULUS,	pbyModulus,	ulModulusSize},
{CKA_PUBLIC_EXPONENT,	pbyPubKeyExponent,	ulPubKeyExponentSize}

其中:

CKA_ENCRYPT: 设置密钥用途, 需根据签名、加密公钥类型不同, 设置为对应的值。

CKA_VERIFY: 设置密钥用途, 需根据签名、加密公钥类型不同, 设置为对应的值。

- 5、在导入证书时, 通过设置 CKA_CLASS 为 CKO_CERTIFICATE, CKA_TOKEN 为 TRUE, 并通过 CKA_VALUE 来写入证书内容。其模板信息如下:

CK_OBJECT_CLASS	ulCertificateClass	= CKO_CERTIFICATE;
CK_BBOOL	bTrue	= TRUE;
CK_BBOOL	bFalse	= FALSE;
CK_CERTIFICATE_TYPE	ulCertificateType	= CKC_X_509;
{CKA_CLASS,	&ulCertificateClass,	sizeof(ulCertificateClass)},
{CKA_TOKEN,	&bTrue,	sizeof (bTrue)},
{CKA_LABEL,	pszContainerName,	strlen((char *)pszContainerName)},
{CKA_MODIFIABLE,	&bTrue,	sizeof (bTrue)},
{CKA_ID,	byCkID,	strlen((char *)byCkID)},
{CKA_CERTIFICATE_TYPE,	&ulCertificateType,	sizeof(ulCertificateType)},
{CKA_PRIVATE,	&bFalse,	sizeof(bFalse)},
{CKA_VALUE,	pbyCertificate,	ulCertificateSize}

7.3 RSA 2048/4096 位加密证书及私钥导入流程

RSA 2048/4096 位加密证书私钥需以密文形式导入, 在导入过程中需将公钥证书及对应的私钥同时导入。导入流程图见图 3, 详细说明如下:

- 1、调用 PKCS#11 接口, 根据 CKA_CLASS、CKA_ID 查找并获取 RSA 交互密钥私钥句柄。

本步骤中需调用到的接口: C_FindObjectInit、C_FindObjects、C_FindObjectsFinal。

- 2、调用 PKCS#11 接口, 导入加密后的 RSA 私钥。

本步骤中需调用到的接口：C_UnwrapKey。

3、调用 PKCS#11 接口，导入 RSA 加密证书。

本步骤中需调用到的接口：C_CreateObject。

4、由 CFCA 控件完成 RSA 加密证书导入请求。

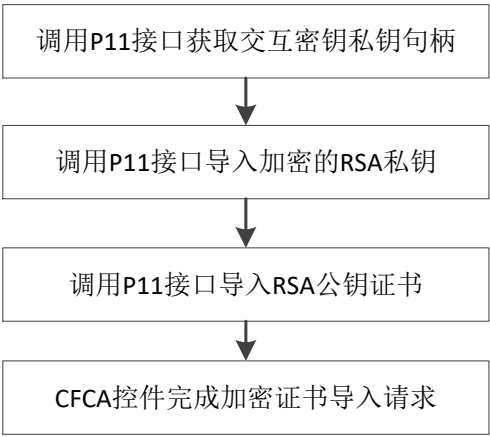


图 3 RSA2048/4096 位加密证书导入流程

7.3.1 PKCS#11 接口描述

7.3.1.1 C_UnwrapKey

CK_RV C_UnwrapKey(CK_SESSION_HANDLE hSession,
CK_MECHANISM_PTR pMechanism,
CK_OBJECT_HANDLE hUnwrappingKey,
CK_BYTE_PTR pWrappedKey,
CK_ULONG ulWrappedKeyLen,
CK_ATTRIBUTE_PTR pTemplate,
CK_ULONG ulAttributeCount,
CK_OBJECT_HANDLE_PTR phKey)

描述：通过 UnwrapKey 导入加密私钥。

特殊参数取值说明：
pMechanism : CKM_RSA_PKCS
hUnwrappingKey : 交互私钥句柄。
pWrappedKey: 使用交互密钥公钥加密的加密证书私钥，
加密证书私钥结构请参照章节 8。
ulWrappedKeyLen: 加密过的私钥长度。
pTemplate: 导入模板。

备注：

- 1、RSA 私钥包含公钥值，创建私钥对象时需同时创建对应的公钥对象（请按照 CFCA 制定的 CKA_LABEL 和 CKA_ID 规则来创建）。

- 2、创建对应的公钥、私钥对象时请先删除原有的交互密钥公钥、私钥对象（因交互密钥对与待导入的加密密钥对的 CKA_LABEL 与 CKA_ID 一样，会造成混淆）。
- 3、在写入公钥时，KEY 需置 CKA_MODULUS_BITS 属性以便以后查看证书密钥长度。
- 4、RSA 2048/4096 位加密证书私钥以密文形式导入，导入私钥时，需指定 CKA_KEY_TYPE 为 CKO_PRIVATE_KEY，且置 CKA_TOKEN 属性为 TRUE，具体私钥值由 KEY 内部解密出后写入，导入模板如下：

```

CK_OBJECT_CLASS  ulPrikeyClass      = CKO_PRIVATE_KEY;
CK_KEY_TYPE      ulKeyType          = CKK_RSA;
CK_BBOOL         bTrue              = TRUE;
CK_BBOOL         bFalse             = FALSE;
{CKA_CLASS,      &ulPrikeyClass,    sizeof(ulPrikeyClass)},
{CKA_KEY_TYPE,   &ulKeyType,        sizeof(ulKeyType)},
{CKA_TOKEN,      &bTrue,            sizeof(bTrue)},
{CKA_LABEL,      pszContainerName,   strlen((char *)pszContainerName)},
{CKA_ID,         byCkID,            strlen((char *)byCkID)},
{CKA_MODIFIABLE, &bTrue,            sizeof(bTrue)},
{CKA_SENSITIVE,  &bTrue,            sizeof(bTrue)},
{CKA_UNWRAP,     &bTrue,            sizeof(bTrue)},
{CKA_PRIVATE,    &bTrue,            sizeof(bTrue)},
{CKA_DECRYPT,     &bTrue,            sizeof(bTrue)},
{CKA_SIGN,       &bFalse,           sizeof(bFalse)}

```

- 5、导入加密证书请参照章节 7.2.1.4。

8. RSA 2048/4096 位加密证书对应的私钥结构定义

RSA 2048/4096 位加密证书对应的私钥需以加密形式导入，私钥结构的自定义 ASN.1 格式（DER 编码）如下：

```

EVPPrivateKey ::= SEQUENCE {
    Version          INTEGER,
    AsymAlgID        OBJECT IDENTIFIER,
    SymAlgID         OBJECT IDENTIFIER,
    EncryptedSymKey   OCTET STRING,
    EncryptedPrivateKey OCTET STRING
}

```

其中：

Version: EVPPrivateKey 结构的版本号，取值为 0x01。

AsymAlgID: 非对称加密算法标识符，取值为：1.2.840.113549.1.1.1。

SymAlgID: 对称算法标识符，本文档中为 3DES ECB 算法标识符，取值为：1.3.6.1.4.1.4929.1.7。

EncryptedSymKey: 经交互密钥公钥加密的对称密钥，详细格式请见 PKCS#1 RSA 加密结果。

EncryptedPrivateKey: 经对称密钥加密的加密证书私钥。

9. PIN 码框

为保证用户数据安全，PIN 码输入框应由 P11 库弹出。

在 C_Login 时，CFCA 会向 P11 库传入默认的 PIN 码：

```
unsigned char byPin[] = {0x01, 0x08, 0x31, 0x32, 0x33, 0x34, 0x35, 0x36, 0x37, 0x38, 0x00};
```

P11 库检测到此 PIN 码，则自动弹出 PIN 码框，要求用户输入 PIN 码。

10. 代码签名

为保证 P11 库文件不被恶意修改，厂商应使用自己的代码签名证书对库文件进行文件签名，CFCA 在调用 P11 库前会首先验证此签名。

厂商需提供以下几项内容：

- 1、P11 库文件名称及其安装路径；例如：/usr/lib/xxx.dylib
- 2、P11 库文件的代码签名文件；

文件内容：P11 库文件的 RSA PKCS#7 分离式签名结果（Base64 编码）。

命名方法：P11 库文件名.sig，例如：xxx.sig

安装路径：与 P11 库文件的安装路径相同。

- 3、验签 P11 库文件代码签名时，所需的证书链。