**常见对称加密算法填充规则，按待加密块大小划分加密算法：**

8字节：DES算法、3DES算法

16字节：AES算法、SM1算法、SM4算法

注：加密算法一般需要按一定长度对齐，叫做块大小，比如8字节，那么对于一段任意的数据，加密前需要对最后一个块填充到8 字节，解密后需要删除掉填充的数据。

1. 如原文数据为(22字节)：

0x10 0x11 0x12 0x11 0x10 0x13 0x10 0x12 0x10 0x11 0x10 0x11 0x10 0x11 0x10 0x11

0xaa 0xbb 0xcc 0xdd 0xee 0x11

待填充字节长度为n=3\*8-22=2

**填充类型：0填充**

数据长度未对齐是使用0x00进行填充，填充后数据如下：

0x10 0x11 0x12 0x11 0x10 0x13 0x10 0x12 0x10 0x11 0x10 0x11 0x10 0x11 0x10 0x11

0xaa 0xbb 0xcc 0xdd 0xee 0x11 0x00 0x00

**填充类型：PKCS#5填充（块长为8）**

数据长度不对齐时使用0xXX填充(0xXX = n)，填充后数据如下：

0x10 0x11 0x12 0x11 0x10 0x13 0x10 0x12 0x10 0x11 0x10 0x11 0x10 0x11 0x10 0x11

0xaa 0xbb 0xcc 0xdd 0xee 0x11 0x02 0x02

**填充类型：PKCS#7填充（块长为1-255）**

数据长度不对齐时使用0xXX填充(0xXX = n)，填充后数据如下：

0x10 0x11 0x12 0x11 0x10 0x13 0x10 0x12 0x10 0x11 0x10 0x11 0x10 0x11 0x10 0x11

0xaa 0xbb 0xcc 0xdd 0xee 0x11 0x02 0x02

1. 如原文数据为(24字节)：

0x10 0x11 0x12 0x11 0x10 0x13 0x10 0x12 0x10 0x11 0x10 0x11 0x10 0x11 0x10 0x11

0xaa 0xbb 0xcc 0xdd 0xee 0x11 0xee 0x11

待填充字节长度为n=3\*8-24=0

**填充类型：0填充**

数据长度对齐时不进行填充

**填充类型：PKCS#5填充（块长为8）**

数据长度对齐时使用0xXX填充(0xXX = 块大小)，填充后数据如下：

0x10 0x11 0x12 0x11 0x10 0x13 0x10 0x12 0x10 0x11 0x10 0x11 0x10 0x11 0x10 0x11

0xaa 0xbb 0xcc 0xdd 0xee 0x11 0xee 0x11 0x08 0x08 0x08 0x08 0x08 0x08 0x08 0x08

**填充类型：PKCS#7填充（块长为1-255）**

数据长度对齐时使用0xXX填充(0xXX = 块大小)，填充后数据如下：

0x10 0x11 0x12 0x11 0x10 0x13 0x10 0x12 0x10 0x11 0x10 0x11 0x10 0x11 0x10 0x11

0xaa 0xbb 0xcc 0xdd 0xee 0x11 0xee 0x11 0x08 0x08 0x08 0x08 0x08 0x08 0x08 0x08

*小结：由于使用PKCS#7和PKCS#5填充时，最后一个字节肯定为填充数据的长度，所以在解密后可以准确删除填充的数据，而使用0填充时，没办法区分真实数据与填充数据，所以只适合以\0结尾的字符串加解密。此外PKCS#7和PKCS#5填充的区别在于分组块的大小不同，可以将PKCS#5填充理解为PKCS#7填充的一个子集。*

*【为啥当为整数倍时，还是要多增加8个0x08？因为解密的需要。*

[*https://www.cnblogs.com/cyds/p/15640763.html*](https://www.cnblogs.com/cyds/p/15640763.html)*】*

**常见非对称算法填充规则： （PKCS#1填充& PSS填充）**

RSA算法和SM2算法（签名和加密无需填充）

**填充类型：PKCS#1填充，以RSA2048为例(d:私钥、e:公钥、n:模数)**

1. RSA\_PKCS#1\_PADDING填充规则：

M：实际原文数据；

PS：填充数据；加密时为非0x00的随机数，签名时00（BT为00）或01（BT为FF），待填数据长度K(256)-3-D，其中PS至少为8字节

BT：填充后数据，公钥操作（加密）为02，私钥操作（签名）为00或01

EM：填充后待加密（签名）原文数据，字节长度为256字节（十六进制），具体数据格式如下：EM = 00 + BT + PS + 00 + M

1. 对数据M进行RSA加密解密，填充规则为RSA\_PKCS#1\_PADDING

加密步骤：

1：C = EM^e mod n

解密步骤：

1：EM = C^d mod n

1. 对数据M进行RSA签名验签，填充规则为RSA\_PKCS#1\_PADDING（PKCS#1签名）

签名步骤：

1：计算 mHash = Hash(M)，

2：构造X509\_SIG结构并DER编码编码结果为X509\_DER，其中包括了摘要算法（Hash）以及摘要结果(mHash)，X509\_DER = Hash函数 + mHash

3: 对2中的结果进行PKCS#1填充，填充后为EM = 00 + BT + PS + 00 + X509\_DER

4：sign = EM^d mod n

验签步骤：

1：EM = sign^e mod n

2: 从X509\_DER中获取摘要算法（Hash）和摘要值（mHash）

3：mHash1 = mHash（M）

4：比较mHash1与mHash，相等则验签通过，不相同则验签未通过。

补充：上述3中所提到的即是P1签名和验签，此外还有P7签名和验签。P7签名P1签名相比一般是在P1签名的基础上按照PKCS#7标准格式将签名值、签名者证书、CRL（证书作废链）、被签名内容信息、签名者信息等内容封装并进行DER编码。具体签名验签如下：

签名步骤：

1：计算 mHash = Hash(M)，

2：构造X509\_SIG结构并DER编码编码结果为X509\_DER，其中包括了摘要算法（Hash）以及摘要结果(mHash)，X509\_DER = Hash函数 + mHash

3: 对2中的结果进行PKCS#1填充，填充后为EM = 00 + BT + PS + 00 + X509\_DER

4：sign = EM^d mod n

5：按PKCS#7标准格式对签名进行编码，得到的结果记为signedData

验签步骤：

1：从signedData中提取签名者证书序列号等信息，按证书序列号获取签名者证书并检验证书的有效性(在签名中提取证书或通过LDAP下载证书)

2：从证书中提取签名者公钥e，计算EM = C^e mod n

3: 从X509\_DER中获取摘要算法（Hash）和摘要值（mHash）

4：mHash1 = mHash（M）

5：比较mHash1与mHash，相等则验签通过，不相同则验签未通过。

*注EB中开头的00是为了防止做加密运算的padding后数据大于模指数N。输入原文数据长度小于（256-11）字节，输出密文数据长度为256字节*

**填充类型：PSS填充，以RSA2048签名验签为例(d:私钥、e:公钥、n:模数)【模数这个词搜不到】**

对数据M进行RSA签名，数据M签名时的PSS填充规则如下：

签名步骤：

1：计算 mHash = Hash(M)，其长度为hashlen

2：计算slen长度的随机值salt，这里slen取hashlen。

3：计算 M' = Hash(0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 || mHash || salt)

4：计算 DB = PS || 0x01 || salt，PS是字节0。PS的长度是nLen - 2\*hashlen - 2 。

5: 计算 MaskedDB = DB^ MGF(M')【MGF掩码生成函数, Mask Generation Function，^应该是指异或运算】【异或符号：**数学符号**⊕；**程序符号**^】

6: 待签名数据为EM = MaskedDB||M' || 0xBC【0xBC终止符 trailer field尾部字段，结束字段】

【It is used in the encoding operation and is an integer. The value MUST be 1, which represents the trailer field with hexadecimal value 0xBC.】

7: C = EM^d mod n

验签步骤：

1：计算EM = C^e mod n

2：计算DB = MaskedDB ^ MGF(M')，获取salt

2：计算 mHash = Hash(M) ，获取mHash

3：计算 M'’= Hash(0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 || mHash || salt)

4：比较M'和M'’，相同验签通过，不同验签未通过。

*注：因RSA算法没有加入乱数，当出现重复性的数据时，攻击者会通过相同的加密密文而猜出原文内容，因此导入填充机制来增强安全性。SM2算法在进行加密和签名时会引入随机数。*

**数字信封：**

数字信封概念：数字信封用于通信双方交换数据。发送方生成一个随机的报文密钥，使用报文密钥对发送内容进行加密(对称算法)，再用接收方的公钥对报文密钥加密，最后将加密的报文密钥和加密的发送内容按PKCS#7标准编码组成一个“数字信封”。发送方还可以为发送内容附加签名。

数字信封制作过程：

1：随机生成报文密钥mk；

2：用接收方公钥对mk加密(按PKCS#1标准)；

3：使用mk对发送内容加密；

4：如果需要签名，则执行如下步骤：

A：对发送内容作数字签名(不作编码)；

B：使用mk为密钥对签名进行加密；

5按PKCS#7格式编码构成数字信封。

解数字信封的过程：

1：使用接收者自已的私钥解出mk；

2：使用mk为密钥解密传送的内容；

3：如果存在签名，则执行：

A：使用mk为密钥解出签名值；

B：验证签名。

数字信封格式见



**摘要算法：**

1. 用途：验证消息完整性
2. 特点：1）输入消息不受限制，但输出消息固定；2）输入消息不同，输出消息一定不同； 3）消息摘要算法不可逆，摘要值无法推出原文
3. 分类：SHA系列（安全散列）、MAC（消息认证码）、MD系列（消息摘要）、SM3

**MD系列**：输出摘要数据长度均为128bit，安全性为MD5>MD4>MD2

**SM3**: 输出摘要数据长度为256bit

**SHA系列**：SHA1摘要数据长度为160bit

SHA224摘要数据长度为224bit

SHA256摘要数据长度为256bit

SHA384摘要数据长度为384bit

SHA512摘要数据长度为512bit

**MAC系列**：MAC算法是在SHA算法和MD算法的基础上增加了秘钥，所以其安全性比SHA算法和MD算法要高，在验证消息是否被篡改的同时实现对信源和信宿的识别，其摘要输出长度与对应的SHA算法和MD算法相同。

实现过程：发送方发送Hk（M）和M至接收方，接收方用M和K算出Hk（M）并比较与收到的Hk（M）是否一致，一致则为通过。

*注：1.MAC算法中的秘钥不同，所计算出的摘要值也不相同，秘钥的要求是16进制且长度是2的倍数*

*2.摘要算法的输入数据需要满足对应算法的分组长度，不满足的需要进行填充，其填充规则与上述涉及的填充规则不同。*

**SKF、SDF、P11接口库：**

SKF库、SDF库和P11库主要的区别在所参考的标准不同，其中SKF库和SDF库参考的是国密标准的相关文档，如SKF库参考《GM-0016-2012-智能密码钥匙密码应用接口规范》和SDF库参考《GM-0018-2012-密码设备应用接口规范》，P11库参考的是国际标准的相关文档如《PKCS#11 V2.20:Cryptographic Token Interface Standard》。

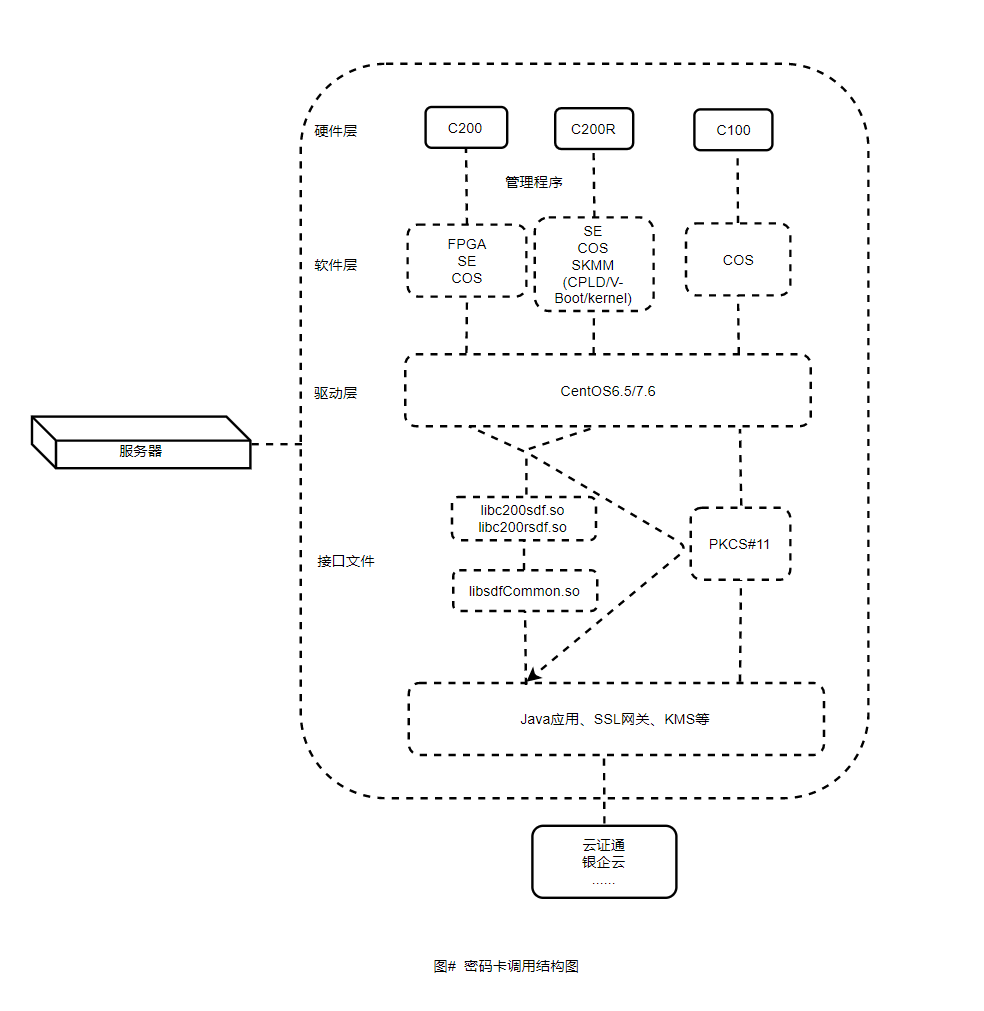
相同点：都可实现相关接口功能在密码钥匙或密码设备内的实现

不同点：按照国密检测的相关要求，就必须使用国密相关文档提供的接口来实现，这样就出现了上述3中接口库

总结：以密码钥匙签章功能来看，在SKF库和P11库中均可实现，但实际使用哪一个库需根据具体场景，如面对国密检测需要时就需要使用SKF库。非国密检测场景时，根据客户需求，若客户的签章控件支持国密接口时可以选择使用SKF签章，若客户的签章控件没有升级，那就只能通过P11库来实现签章功能。

**密码卡设备**

1.密码卡设备调用关系结构图如下：



1. **密码卡简介**：

1）C100型PCIE接口/USB接口密码卡（支持RSA和SM2密钥）

C100型PCIE型接口密码卡产品主要为服务端产品提供安全可靠的密钥管理和数字签名运算服务。*C100型为主、从两级结构的多安全芯片产品（1个主安全模块和若干从安全模块）。*

主要包括：密码卡主机（主/从安全模块）、P11库、设备管理工具，各组成具体功能如下：

密码卡主机（主/从安全模块）：主安全模块主要支持密钥生成、PIN码验证、数字签名和密钥导入等功能。从安全模块主要支持数字签名运算等功能。

P11库：Linux平台，主要为上层应用、网关、KMS等提供功能调用接口。

设备管理工具：Linux平台，具有设备初始化、生成密钥对、导入密钥对、主从密钥同步等密钥管理功能，以及操作员口令解锁功能。

2）C100型SATA接口密码卡（支持RSA和SM2密钥）

C100-SATA加密卡通过SATA接口与服务器主机通信，主要实现：密钥管理、密钥运算和生产管理功能。*C100-SATA加密卡仅采用一个安全模块芯片。*

主要包括：密码卡主机（安全模块）、SDF库、设备管理工具

1）C200型PCI-E密码卡(国密版原型)

C200型PCI-E密码卡可以应用在需要国密算法密码运算和密钥管理等安全功能的、具有PCI-E总线的通信设备、计算机设备、安全保密设备上。

2）C200-HPC型PCI-E密码卡（国密版）

C200-HPC型PCI-E密码卡（标准版）可以应用在需要SM2、SM3和SM4等国密算法密码运算。

3）C200-R型PCI-E密码卡（国际版）

C200-R型PCI-E密码卡具有RSA和ECDSA算法等国际算法密码运算功能。

4）C300-M（慕斯芯片）型PCI-E密码卡（国密+国际）

C300-M型PCI-E密码卡主要应用在需要SM1、SM2、SM3和SM4等国密算法密码运算以及RSA2048、ECDSA、AES等国际算法密码运算。