如果几个应用程序正在使用 Cryptoki，则每个应用程序应当调用 C_Finalize。

12.2.2 使用证书

12.2.2.1 函数说明

1. C_FindObjectsInit 函数

```
CK_DEFINE_FUNCTION(CK_RV, C_FindObjectsInit)(
    CK_SESSION_HANDLE hSession,
    CK_ATTRIBUTE_PTR pTemplate,
    CK_ULONG ulCount);
```

C_FindObjectsInit 按匹配模板，初始化对令牌和会话中的对象的查找。hSession 是会话的句柄；pTemplate 指向确定要匹配的属性值的查找模板；ulCount 是查找模板中的属性数。匹配标准是与模板中所有属性精确的逐字节的匹配。为了找到所有的目标，可将 ulCount 设置为 0。

调用 C_FindObjectsInit 后，应用程序可能一次或多次调用 C_FindObjects 以获得匹配模板的对象的句柄，接着最终调用 C_FindObjectsFinal 来结束活动的查找操作。在一个给定的会话中一次至多有一个查找操作是活动的。

查找只能发现会话能看见的对象。比如，“读/写公共会话”中的对象查找不能发现任何专用对象（即使查找模块中的一个属性指定该查找用于专用对象）。

2. C_FindObjects 函数

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

C_FindObjects 继续查找匹配模板的令牌和会话中的对象，获得对象句柄。hSession 是会话句柄；phObject 指向接收对象句柄列表（数组）的单元；ulMaxObjectCount 是要返回的对象句柄的最大数；pulObjectCount 指向接收实际返回对象句柄数的单元。

如果已没有匹配模板的对象，那么 pulObjectCount 所指单元接收值为 0。调用 C_FindObjects 前必须先调用 C_FindObjectsInit 成功。

3. C_FindObjectsFinal

```
CK_DEFINE_FUNCTION(CK_RV, C_FindObjectsFinal)(
    CK_SESSION_HANDLE hSession);
```

C_FindObjectsFinal 结束查找令牌和会话对象。hSession 是会话的句柄。

4. C_GetAttributeValue 函数

```
CK_DEFINE_FUNCTION(CK_RV, C_GetAttributeValue)(
    CK_SESSION_HANDLE hSession,
    CK_OBJECT_HANDLE hObject,
```

```
CK_ATTRIBUTE_PTR pTemplate,
CK_ULONG ulCount);
```

C_GetAttributeValue 获得对象的一个或多个属性值。hSession 是会话的句柄；hObject 是对象的句柄；pTemplate 指向规定即将获得属性值的模板，并接收属性值；pulCount 是模板中的属性数。

对于属性模板中的每个三元组（类型，值，长度），C_GetAttributeValue 执行以下的算法。

① 如果由于对象敏感或不可获取因而不能返回对象的规定属性(即类型字段规定的属性)的话，那么该三元组中的 ulValueLen 字段被修改，包含的值为−1(即当转换为 CK_LONG 类型时，包含值−1)。

② 否则，如果对象的规定属性无效（对象没有这种属性），那么该三元组中的 ulValueLen 字段被修改，包含的值为−1。

③ 否则，如果 pValue 字段的值为 NULL_PTR，那么 ulValueLen 字段被修改，包含的是对象指定属性的实际长度。

④ 否则，如果 ulValueLen 中规定的长度足够长，能包含对象指定属性的值，那么属性就被复制到 pValue 中的缓冲区里，ulValueLen 字段被修改，包含属性的实际长度。

⑤ 否则，ulValueLen 字段被修改，包含值−1。

如果情况①应用于任何请求的属性，那么调用返回 CKR_ATTRIBUTE_SENSITIVE 值。如果情况②应用于任何请求的属性，那么调用返回 CKR_ATTRIBUTE_TYPE_INVALID 值。如果情况⑤应用于任何请求的属性，那么调用返回 CKR_BUFFER_TOO_SMALL 值。通常，如果这些错误码中的不止一个能用，Cryptoki 可能返回其中的任意一个。只有如果没有一个能适用于请求的属性时才返回 CKR_OK。

12.2.2.2　示例程序

完成证书访问，需要进行如下操作步骤：①初始化环境；②获取槽列表；③打开会话；④用户身份登录；⑤按证书属性查找；⑥登出；⑦关闭会话；⑧清空环境。

1. Cryptoki 通用调用过程代码

```
CK_SLOT_ID            slotList[32];
CK_ULONG              ulSlotCount = 32;
CK_RV                 rv = CKR_OK;
CK_SESSION_HANDLE     hSession = 0;
char                  *passwd=" 111111" ;
int                   passlen = 6;
// 初始化环境
rv = C_Initialize(NULL_PTR);
if (rv != CKR_OK) {   return BAD_INIT_ERROR;    }
// 获取槽列表
rv = C_GetSlotList(TRUE, &slotList[0], &ulSlotCount);
if (rv != CKR_OK) {   return BAD_SLOT_ERROR;   }
// 打开会话
```

```
rv = C_OpenSession(slotList[0],
      CKF_SERIAL_SESSION | CKF_RW_SESSION,
      NULL, NULL, &hSession);
if (rv != CKR_OK) {    return BAD_SESSION_ERROR;     }
// 用户身份登录
rv = C_Login(hSession, CKU_USER, passwd, passlen);
if (rv != CKR_OK) {    return BAD_LOGIN_ERROR;       }
// 具体业务操作
⋮
// 登出
rv = C_Logout(hsession);
// 关闭会话
rv = C_CloseSession(hsession);
// 清空环境
rv = C_Finalize(NULL);
```

2. 查询证书过程代码

```
#define numof(arr)      (sizeof(arr)/sizeof((arr)[0]))
CK_SESSION_HANDLE session = ...; // 假定会话已经打开，并成功登录
int rv, res = -1;
CK_OBJECT_HANDLE obj;
CK_ULONG count;

CK_OBJECT_CLASS cert_search_class = CKO_CERTIFICATE;
CK_ATTRIBUTE cert_search_attrs[] = {
      {CKA_CLASS, &cert_search_class, sizeof(cert_search_class)}   };
// 初始化查找对象操作
rv = C_FindObjectsInit(session, cert_search_attrs, numof(cert_search_attrs));
if (rv != CKR_OK) {      return BAD_FIND_INIT_ERROR;      }
// 循环查找匹配的证书对象
do {
      rv = C_FindObjects(session, &obj, 1, &count);
      // 获得证书 DER 编码值
      res = get_cert_data(session, obj);
      // 如果找到，继续下一个证书对象
} while (rv == CKR_OK && count == 1);
// 完成查找，关闭查找对象
rv = C_FindObjectsFinal(session);
```

3. 获取证书 DER 编码值代码

证书数据就在证书对象的属性里，需要获取证书对象的 CKA_VALUE 属性。

```
int get_cert_data(CK_SESSION_HANDLE session, CK_OBJECT_HANDLE obj)
{
```

```
    char cert_data[2048];
    int cert_len;
    CK_ATTRIBUTE templ;
    int rv;
    templ.type = CKA_VALUE;
    templ.pValue = cert_data;
    templ.ulValueLen = sizeof(cert_data);
    rv = C_GetAttributeValue(session, o, &templ, 1);
    if (rv != CKR_OK) {       return -1;          }
    cert_len = templ.ulValueLen;
    // cert_data 中已经保存了证书的 DER 值
    return 0;
}
```

12.2.3　使用私钥

12.2.3.1　函数说明

1．C_SignInit 函数

签名的第一步是获得私钥对象。通过私钥的标签值，调用 FindObjectsInit、FindObjects 和 FindObjectsFinal，找到私钥对象，接着利用私钥对象调用 C_SignInit 和 C_Sign 完成签名。这两个函数的说明如下。

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

C_SignInit 初始化签名操作。hSession 是会话的句柄；pMechanism 指向签名机制；hKey 是签名密钥的句柄。

调用 C_SignInit 后，可直接调用 C_Sign 对单部分数据进行签名，或者先调用 C_SignUpdate 一次或多次，接着调用 C_SignFinal 给多部分中的数据签名。签名操作是有效的，直到应用程序调用 C_Sign 或 C_SignFinal 真正获得签名。要处理另外的数据（单部分或多部分），应用程序必须再次调用 C_SignInit。

2．C_Sign 函数

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

C_Sign 给单部分中的数据签名。hSession 是会话句柄；pData 指向数据；ulDataLen 是

数据长度；pSignature 指向接收签名的单元；pulSignatureLen 指向包含签名长度的单元。

签名操作必须用 C_SignInit 初始化。调用 C_Sign 总是会结束有效的签名操作，除非它返回 CKR_BUFFER_TOO_SMALL 或这是一次成功的调用（即返回 CKR_OK）来确定包含签名所需的缓冲区的长度。

不能使用 C_Sign 来结束多部分签名操作，C_Sign 必须在不插入 C_SignUpdate 的情况下在 C_SignInit 之后调用。

对于大部分机制，C_Sign 相当于一系列 C_SignUpdate 操作后面跟着 C_SignFinal。

3. C_SignUpdate

```
CK_DEFINE_FUNCTION(CK_RV, C_SignUpdate)(
    CK_SESSION_HANDLE hSession,
    CK_BYTE_PTR pPart,
    CK_ULONG ulPartLen);
```

C_SignUpdate 继续多部分签名操作，处理另一个数据部分。hSession 是会话句柄；pPart 指向数据部分；ulPartLen 是数据部分的长度。

签名操作必须用 C_SignInit 初始化。这一函数可以连续调用若干次。调用 C_SignUpdate 产生错误时终止当前的签名操作。

4. C_SignFinal

```
CK_DEFINE_FUNCTION(CK_RV, C_SignFinal)(
    CK_SESSION_HANDLE hSession,
    CK_BYTE_PTR pSignature,
    CK_ULONG_PTR pulSignatureLen);
```

C_SignFinal 结束多部分签名操作，返回签名值。hSession 是会话句柄；pSignature 指向接收签名的单元；pulSignatureLen 指向包含签名长度的单元。

签名操作必须用 C_SignInit 初始化。调用 C_SignFinal 总是会结束活动的签名操作，除非它返回 CKR_BUFFER_TOO_SMALL 或这是一次成功的调用（即返回 CKR_OK）来确定包含签名所需的缓冲区的长度。

12.2.3.2 示例程序

使用私钥进行签名，需要进行如下操作：①初始化环境；②获取槽列表；③打开会话；④用户身份登录；⑤进行签名操作；⑥登出；⑦关闭会话；⑧清空环境。

除“⑤进行签名操作”外，其他步骤与 12.2.2 节中“Cryptoki 通用调用过程代码”相同。

1. 私钥签名过程

```
// 通过 label 属性获取私钥对象
CK_OBJECT_HANDLE    key;
CK_RV               rv = CKR_OK;
CK_KEY_TYPE         keyType_rsa = CKK_RSA;
CK_OBJECT_CLASS private_key_class = CKO_PRIVATE_KEY;
```