云京电子签章系统方案设计

|  |  |  |
| --- | --- | --- |
| 编写 | 芦蓉 | 2018-12-14 |
| 审核 |  |  |

|  |  |  |  |
| --- | --- | --- | --- |
| 版本 | 变更人 | 变更日期 | 变更项 |
| V1.0 | 芦蓉 | 2018-12-14 | 完成初稿 |
|  |  |  |  |

**目录**

1 引言 1

1.1 背景和目的 1

1.2 范围 1

1.3 术语和缩略语 1

1.4 参考资料 1

2 总体设计 1

2.1 需求规定 1

2.2 运行环境 2

2.3 系统架构设计 2

3 接口设计 3

3.1 用户接口 3

3.2 外部接口 5

3.3 内部模块接口 12

4 系统数据结构设计 14

4.1 逻辑结构设计 14

4.2 数据库设计 35

5 安全性设计 37

5.1 物理和环境安全 37

5.2 网络和通信安全 38

5.3 设备和计算安全 38

5.4 应用和数据安全 38

5.5 密钥管理 38

引言

背景和目的

本文档介绍了云京电子印章系统的设计方案。

本文档编制的目的是：根据国密安全电子印章系统的规范，明确云京电子印章系统的设计思路，并指导云京电子印章系统的实现及测试，最终通过国密资质审核。

范围

本文档适用于云京电子印章系统的开发和测试。

术语和缩略语

* 电子印章系统(electronic seal system)：包含电子印章管理系统和电子签章软件，其中电子印章管理系统包括印章管理员管理、电子印章制作与管理、电子印章验证服务以及安全审计等功能。电子签章软件是使用电子印章对各类电子文件进行电子签章的软件。
* 制章人（electronic stamp maker）：电子印章系统中具有签署和管理电子印章信息权限的管理员。管理员的数字证书可以是单位证书或个人证书，电子印章中的图片和信息必须经制章人的数字证书进行数字签名。
* 签章人(electronic seal signer)：电子印章系统中对电子文件进行签章操作的签章持有用户。

参考资料

[1] GMT0031-2014-安全电子签章密码技术规范

[2] GMT0047-2016-安全电子签章密码检测规范

[3] GMT0009-2012-SM2密码算法使用规范

[4] GMT0029-2014-签名验签服务器技术规范

[5] GMT0014-2012-数字证书认证系统密码协议规范

[6] GMT0015-2012-基于SM2密码算法的数字证书格式规范

[7] 云京电子印章系统需求文档

总体设计

需求规定

主要的功能需求：

* 支持SM2算法的数字签名功能，SM2算法签名数据结构遵循GM/T 0009；
* 电子印章管理系统能正确生成电子印章数据，电子印章数据的数据内容和编码格式满足GM/T 0031；
* 电子印章系统提供电子印章验证功能，包括印章数据格式验证、签名值验证、制章人证书有效性验证、印章有效期验证；
* 电子印章系统能正确生成电子签章数据，数据内容和编码格式满足GM/T 0031；
* 电子印章系统提供电子签章验证功能，包括签章数据格式验证、签名值验证、签章人证书有效性验证、签章人证书列表验证、签章时间有效期验证、签章原文杂凑验证、签章中电子印章有效性验证。

部件需求：

* 系统配用的密码产品应是经国家密码管理主管部门批准使用的产品。

运行环境

该软件运行于linux系统，JAVA运行环境1.8以上，并安装了MySQL数据库。

系统架构设计

2.3.1 云京印章系统

本系统包括2个部分：电子印章管理系统和电子签章系统。



图2-1电子印章系统结构图



图2-2电子印章管理系统结构图



图2-3电子签章系统结构图

2.3.3部署设计

电子签章系统与应用系统就近部署，用于为应用系统所需各类电子文档提供电子印章签章或验章服务，电子签章系统与CA中心数字证书系统相连用于获取证书CRL列表和OCSP服务。

电子印章管理系统可独立部署于印章中心，负责电子印章系统的制章人证书管理和电子印章制作，与CA中心数字证书系统连接用于制章人证书申请和签章人的证书验证；CA中心数字证书系统为电子印章系统提供证书申请、更新、CRL列表下载和OCSP服务。



图2-4电子印章系统部署图

接口设计

用户接口

### 用户管理

1）/sealCenter/config/v1.0/administrator/login

Method: POST

* **Name:** 管理员/审计员登录
* **Header:**

|  |  |  |
| --- | --- | --- |
| **Key** | **Value** | **Describe** |
| Content-Type | application/json |  |

* **Permissions:** [ ]
* **Params:**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Key** | **Type** | **Range** | **Required** | **Describe** |
| account | string |  | true | 账号 |
| password | string | 6-20 | true | 密码，数字、大小写字母、字符两种组合 |
| createTime | datetime |  | false | 创建时间 |

* **InputOnly:** [account, password]
* **OutputOnly:** [ createTime]

Method: DELETE

* **Name:** 退出登录
* **Header:**
* **Permissions:** [IsAdministrator]
* **Params:**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Key** | **Type** | **Range** | **Required** | **Describe** |
| requestTime | datetime |  | true | 请求时间，必须是当前时间 |

* **InputOnly:** [requestTime]
* **OutputOnly:** [ ]

2）/sealCenter/config/v1.0/administrator/password

Method: PUT

* **Name:** Administrator 更新密码
* **Header:**

|  |  |  |
| --- | --- | --- |
| **Key** | **Value** | **Describe** |
| Content-Type | application/json |  |

* **Permissions:** [IsAdministrator]
* **Params:**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Key** | **Type** | **Range** | **Required** | **Describe** |
| password | string | 6-20 | true | 密码，数字、大小写字母、字符两种组合 |
| requestTime | dateTime |  | true | 请求时间，必须是当前时间 |
| account | string | 5-64 | false | 账号 |
| userId | string | uuid4 | false | 用户 id |
| role | enum(Role) |  | false | 用户角色 |
| createTime | datetime |  | false | 创建时间 |
| updateTime | datetime |  | false | 更新时间 |

* **InputOnly:** [password, requestTime]
* **OutputOnly:** [userId, account, role, createTime, updateTime]

3）/sealCenter/config/v1.0/users

Method: POST

* **Name:** 创建用户(管理员、审计员)
* **Header:**

|  |  |  |
| --- | --- | --- |
| **Key** | **Value** | **Describe** |
| Content-Type | application/json |  |

* **Permissions:** [IsAdministrator]，如果是系统初始化，则无权限要求
* **Params:**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Key** | **Type** | **Range** | **Required** | **Describe** |
| account | string | 11 | true | 账号 |
| password | string | 6-20 | true | 密码，数字、大小写字母、字符两种组合 |
| repassword | string | 6-20 | true | 确认密码 |
| role | enum(Role) |  | false | 角色 |
| requestTime | datetime |  | true | 请求时间，必须为当前时间 |
| userId | string | uuid4 | false | 用户 id |
| createTime | datetime |  | false | 创建时间 |
| updateTime | datetime |  | false | 更新时间 |

* **InputOnly:** [password, repassword, role, requestTime]
* **OutputOnly:** [userId, createTime, updateTime]

Method: GET

* **Name:** 列举用户（管理员、审计员）
* **Header:**

|  |  |  |
| --- | --- | --- |
| **Key** | **Value** | **Describe** |

* **Permissions:** [IsAdministrator]
* **Params:**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Key** | **Type** | **Range** | **Required** | **Describe** |
| requestTime | datetime |  | true | 请求时间，必须为当前时间 |

* **InputOnly:** [ requestTime]
* **OutputOnly:** [ account, createTime, role ]
* **OrderByFields:** [createTime, updateTime]
* **DefaultOrderBy:** -createTime

4) /sealCenter/console/v1.0/users/(<userId>)

Method: GET

* **Name:** 获取用户信息
* **Header:**

|  |  |  |
| --- | --- | --- |
| **Key** | **Value** | **Describe** |

* **Permissions:** [IsAdministrator]
* **Params:**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Key** | **Type** | **Range** | **Required** | **Describe** |
| requestTime | datetime |  | true | 请求时间，必须为当前时间 |
| role | enum(Role) |  | false | 角色 |
| userId | string | uuid4 | false | 用户 id |
| account | string |  | false | 用户账号 |
| createTime | datetime |  | false | 创建时间 |
| updateTime | datetime |  | false | 更新时间 |

* **InputOnly:** [requestTime]
* **OutputOnly:** [role, userId, createTime, updateTime]

Method: PUT

* **Name:** 更新控制台用户信息
* **Header:**

|  |  |  |
| --- | --- | --- |
| **Key** | **Value** | **Describe** |
| Content-Type | application/json |  |

* **Permissions:** [IsAdministrator]
* **Params:**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Key** | **Type** | **Range** | | **Required** | **Describe** |
| requestTime | datetime |  | | true | 请求时间，必须为当前时间 |
| password | string | 6-20 | | false | 密码，数字、大小写字母、字符两种组合 |
| role | enum(Role) |  | false | | 角色 |
| userId | string | uuid4 | false | | 用户 id |
| account | string | 5-64 | false | | 用户账号 |
| createTime | datetime |  | false | | 创建时间 |
| updateTime | datetime |  | false | | 更新时间 |

* **InputOnly:** [requestTime, password]
* **OutputOnly:** [role, userId, createTime, updateTime ]

Method: DELETE

* **Name:** 删除用户
* **Header:**

|  |  |  |
| --- | --- | --- |
| **Key** | **Value** | **Describe** |

* **Permissions:** [IsAdministrator]
* **Params:**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Key** | **Type** | **Range** | **Required** | **Describe** |
| requestTime | datetime |  | true | 请求时间，必须为当前时间 |

* **InputOnly:** [requestTime]
* **OutputOnly:** [ ]

### 印章管理

1）/sealCenter/config/v1.0/seals/request

Method: POST

* **Name:** 创建印章申请
* **Header:**

|  |  |  |
| --- | --- | --- |
| **Key** | **Value** | **Describe** |
| Content-Type | application/json |  |

* **Permissions:** [IsAdministrator]
* **Params:**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Key** | **Type** | **Range** | **Required** | **Describe** |
| account | string | 11 | true | 账号 |
| password | string | 6-20 | true | 密码，数字、大小写字母、字符两种组合 |
| repassword | string | 6-20 | true | 确认密码 |
| role | enum(Role) |  | false | 角色 |
| requestTime | datetime |  | true | 请求时间，必须为当前时间 |
| userId | string | uuid4 | false | 用户 id |
| createTime | datetime |  | false | 创建时间 |
| updateTime | datetime |  | false | 更新时间 |

* **InputOnly:** [password, repassword, role, requestTime]
* **OutputOnly:** [userId, createTime, updateTime]

Method: GET

* **Name:** 列举印章申请
* **Header:**

|  |  |  |
| --- | --- | --- |
| **Key** | **Value** | **Describe** |

* **Permissions:** [IsAdministrator]
* **Params:**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Key** | **Type** | **Range** | **Required** | **Describe** |
| requestTime | datetime |  | true | 请求时间，必须为当前时间 |
| id | string | uuid4 | false | 印章申请 id |
| esID | string |  | false | 印章编号（没申请成功的，这项未空） |
| esname | string |  | false | 印章名称 |
| estype | string |  | false | 印章类型（公章） |
| endtime | datetime |  | false | 有效期 |
| status | string |  | false | 印章申请的状态：审核、已审核通过、审核未通过 |

* **InputOnly:** [ requestTime]
* **OutputOnly:** [id, esID, esname, estype, endtime, status]
* **OrderByFields:** [id]

4) /sealCenter/console/v1.0/users/(<userId>)

Method: GET

* **Name:** 获取用户信息
* **Header:**

|  |  |  |
| --- | --- | --- |
| **Key** | **Value** | **Describe** |

* **Permissions:** [IsAdministrator]
* **Params:**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Key** | **Type** | **Range** | **Required** | **Describe** |
| requestTime | datetime |  | true | 请求时间，必须为当前时间 |
| role | enum(Role) |  | false | 角色 |
| userId | string | uuid4 | false | 用户 id |
| account | string |  | false | 用户账号 |
| createTime | datetime |  | false | 创建时间 |
| updateTime | datetime |  | false | 更新时间 |

* **InputOnly:** [requestTime]
* **OutputOnly:** [role, userId, createTime, updateTime]

Method: PUT

* **Name:** 更新控制台用户信息
* **Header:**

|  |  |  |
| --- | --- | --- |
| **Key** | **Value** | **Describe** |
| Content-Type | application/json |  |

* **Permissions:** [IsAdministrator]
* **Params:**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Key** | **Type** | **Range** | | **Required** | **Describe** |
| requestTime | datetime |  | | true | 请求时间，必须为当前时间 |
| password | string | 6-20 | | false | 密码，数字、大小写字母、字符两种组合 |
| role | enum(Role) |  | false | | 角色 |
| userId | string | uuid4 | false | | 用户 id |
| account | string | 5-64 | false | | 用户账号 |
| createTime | datetime |  | false | | 创建时间 |
| updateTime | datetime |  | false | | 更新时间 |

* **InputOnly:** [requestTime, password]
* **OutputOnly:** [role, userId, createTime, updateTime ]

Method: DELETE

* **Name:** 删除用户
* **Header:**

|  |  |  |
| --- | --- | --- |
| **Key** | **Value** | **Describe** |

* **Permissions:** [IsAdministrator]
* **Params:**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Key** | **Type** | **Range** | **Required** | **Describe** |
| requestTime | datetime |  | true | 请求时间，必须为当前时间 |

* **InputOnly:** [requestTime]
* **OutputOnly:** [ ]

### 系统配置接口

1）/sealCenter/config/v1.0/sysconfigs

Method: GET

* **Name:** 列举系统配置项
* **Header:**

|  |  |  |
| --- | --- | --- |
| **Key** | **Value** | **Describe** |
| SToken | uuid4 | token 标识 |
| Signature | hex string | secret 与请求参数的 hmac 值 |

* **Permissions:** [IsAdministrator]
* **Params:**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Key** | **Type** | **Range** | **Required** | **Describe** |
| requestTime | datetime |  | true | 请求时间，必须是当前时间 |

* **InputOnly:** [requestTime]
* **OutputOnly:** [ ]
* **OrderByFields:** [key, createTime, updateTime]
* **DefaultOrderBy:** key

Method: POST

* **Name:** 添加系统配置项
* **Header:**

|  |  |  |
| --- | --- | --- |
| **Key** | **Value** | **Describe** |
| Content-Type | application/json |  |
| SToken | uuid4 | token 标识 |
| Signature | hex string | secret 与请求参数的 hmac 值 |

* **Permissions:** [IsAdministrator]
* **Params:**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Key** | **Type** | **Range** | **Required** | **Describe** |
| requestTime | datetime |  | true | 请求时间，必须为当前时间 |
| key | string | 1-64 | true | 配置项键值 |
| value | string | > 0 | true | 配置项取值 |
| description | string | 0-128 | false | 描述 |
| createTime | datetime |  | false | 创建时间 |
| updateTime | datetime |  | false | 更新时间 |

* **InputOnly:** [requestTime]
* **OutputOnly:** [createTime, updateTime]

2） /sealCenter/config/v1.0/sysconfigs/(?P<key>[0-9a-zA-Z-\_]+)

Method: GET

* **Name:** 获取系统某个配置项
* **Header:**

|  |  |  |
| --- | --- | --- |
| **Key** | **Value** | **Describe** |
| SToken | uuid4 | token 标识 |
| Signature | hex string | secret 与请求参数的 hmac 值 |

* **Permissions:** [IsAdministrator]
* **Params:**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Key** | **Type** | **Range** | **Required** | **Describe** |
| requestTime | datetime |  | true | 请求时间，必须是当前时间 |
| key | string | 1-64 | false | 配置项键值 |
| value | string |  | false | 配置项取值 |
| description | string | 0-128 | false | 描述 |
| createTime | datetime |  | false | 创建时间 |
| updateTime | datetime |  | false | 更新时间 |

* **InputOnly:** [requestTime]
* **OutputOnly:** [key, value, description, createTime, updateTime]

Method: PUT

* **Name:** 修改系统某个配置项
* **Header:**

|  |  |  |
| --- | --- | --- |
| **Key** | **Value** | **Describe** |
| Content-Type | application/json |  |
| SToken | uuid4 | token 标识 |
| Signature | hex string | secret 与请求参数的 hmac 值 |

* **Permissions:** [IsAdministrator]
* **Params:**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Key** | **Type** | **Range** | **Required** | **Describe** |
| requestTime | datetime |  | true | 请求时间，必须为当前时间 |
| key | string | 1-64 | false | 配置项键值 |
| value | string | > 0 | false | 配置项取值 |
| description | string | 0-128 | false | 描述 |
| createTime | datetime |  | false | 创建时间 |
| updateTime | datetime |  | false | 更新时间 |

* **InputOnly:** [requestTime]
* **OutputOnly:** [key, createTime, updateTime]

Method: DELETE

* **Name:** 删除系统某个配置项
* **Header:**

|  |  |  |
| --- | --- | --- |
| **Key** | **Value** | **Describe** |
| SToken | uuid4 | token 标识 |
| Signature | hex string | secret 与请求参数的 hmac 值 |

* **Permissions:** [IsAdministrator]
* **Params:**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Key** | **Type** | **Range** | **Required** | **Describe** |
| requestTime | datetime |  | true | 请求时间，必须为当前时间 |

* **InputOnly:** [requestTime]
* **OutputOnly:** [ ]

3.1.2电子签章系统配置管理界面

。

3.1.3电子签章系统签章生成/验证接口

参考签名服务器规范GM/T 0029-2014的附录A。



图3-1电子签章系统接口

请求的格式：

SVSRequest ::= SEQUENCE {

version Version DEFAULT v1,

reqType ReqType,

request Request,

reqTime GeneralizedTime

}

Version ::= INTEGER {v1(0)}

ReqType ::= INTEGER {

--数据见下表

}

Request ::=CHOICE {

--数据见下表

}

响应的格式：

SVSRespond ::= SEQUENCE {

version Version DEFAULT v1,

reqType ReqType,

respond Respond,

respTime GeneralizedTime

}

Version ::= INTEGER {v1(0)}

ReqType ::= INTEGER {

--数据见下表

}

Request ::=CHOICE {

--数据见下表

}

Respond ::=CHOICE {

--数据见下表

}

reqType协议接口类型定义：

|  |  |  |
| --- | --- | --- |
| 类型字符描述 | 类型值reqType | 说明 |
| sealData | 19 | 单包数字签章 |
| verifysealData | 20 | 单包验证数字签章 |

协议接口功能说明（http POST）：

|  |  |  |
| --- | --- | --- |
| 功能 | 类型  Request/Respond | ASN.1描述（SEQUENCE） |
| 单包数字签章 | 请求 | signMethod INTEGER –使用的签名算法类型  Cert OCTET STRING –签章人对应的签名证书（因为要验证证书，所以不能传证书序列号）  keyIndex INTEGER – 签名者私钥的索引值，如十进制1表示索引值为1的密钥  keyValue OCTET STRING --签名者私钥权限标识码  esID IA5String--电子印章标识，用来选择电子印章  inDataLen INTEGER – 待签的数据原文长度  inData OCTET STRING – 待签的数据原文（各种电子文档格式，将在服务器转换成PDF） |
| 响应 | respValue Integer –0标识成功，非0失败  outDataLen INTEGER – 带签章的数据文档长度  outData OCTET STRING OPTIONAL– 带签章的数据文档(PDF) |
| 单包验证数字签章 | 请求 | type INTEGER – 验证数字签章时使用1证书，2 证书序列号  cert [0] IMPLICT Certificate OPTIONAL – 签名证书，type =1时有效  certSN [1] IMPLICT OCTET STRING OPTIONAL– 签名证书序列号，type =2时有效  inDataLen INTEGER – 待验章的PDF长度  inData OCTET STRING – 待验签章的PDF（服务器从PDF中分离出签章数据） |
| 响应 | respValue Integer –0标识成功，非0失败 |

外部接口

3.2.1 密码卡访问接口

本节内容参考卫士通PCIE密码卡用户手册。

1、关键函数接口：

* [设备访问](file:///F:\reference\wstb\doc\Device.htm)

设备访问是整个安全模块的基础。用户在使用时，首先必须使用该系统获得对硬件设备的访问权限，读取设备的基本信息。之后，才能够继续后续的[安全管道访问](file:///F:\云京电子印章系统\SecPipe.htm)和[密钥访问](file:///F:\云京电子印章系统\KeyManage.htm)操作。

|  |  |
| --- | --- |
| 函数名 | 功能 |
| [SM\_OpenDevice](file:///F:\SM_OpenDevice.htm) | 该函数用来打开安全模块并获取设备访问句柄。  用户在使用[SM\_OpenDevice](file:///F:\云京电子印章系统\SM_OpenDevice.htm)函数打开指定安全模块硬件设备时，通过参数uiDevID与指定的硬件设备相关联。  该函数需要与函数SM\_CloseDevice一一对应使用 |
| [SM\_GetDeviceNum](file:///F:\SM_GetDeviceNum.htm) | 主机中存在的安全模块硬件设备有效数量。 |
| SM\_CloseDevice | 函数用来关闭安全模块 |
| SM\_TestDevice | 用来测试模块平台当前状态。建议在调用SM\_OpenDevice函数后立即调用该函数，以保证模块平台处于正常的工作状态 |
| SM\_GetDeviceInfo | 该函数可用于获取模块平台的安全管道、密钥分组长度、安全存储区、非易失存储区的长度、版本、插槽等方面的信息。 |
| SM\_BuildAuthDev | 函数用来制作用户的认证介质。  输入缺省的PIN，模块在内部产生2对（或1对）公私钥，将公私钥存储在认证介质上，并将对应公钥输出给调用方。这些公私钥在成功调用了SM\_Login命令后作为设备的配用密钥使用 |
| SM\_ChangeUserPin | 更换用户口令 |
| SM\_BackupAuthDev | 备份认证介质上的配用密钥  在备份认证介质时，需要调用方**插上备用的认证介质**，并输入该认证介质的缺省PIN。在成功调用SM\_Login函数登录模块平台以后该函数的调用才有效 |

* [安全管道访问](file:///F:\reference\wstb\doc\SecPipe.htm)

安全管道访问是负责除密钥管理外的安全功能访问操作的子系统。通过访问该子系统，用户可实现数据加解密、数据块摘要、数据签名/验签等安全功能的应用。

对称算法加解密运算函数、数据块摘要函数设计为三段式结构函数。即：

SM\_xxxInit //该函数用于三段式结构运算初始化操作

SM\_xxxUpdate //该函数用于三段式结构运算中间数据运算操作。这里，当参与三段式运算的数据流较大时（大于1M字节），推荐将数据流根据三段式结构运算分组长度的大小，进行整数倍分块后重复调用该函数，以提高设备的处理效率（图示A流程）；而对于数据流较小的运算（小于1M字节），则可直接调用SM\_xxxFinal函数（函数内部将自动调用SM\_xxxUpdate函数，图示B流程）。

SM\_xxxFinal //该函数用于三段式结构运算最后一组数据运算操作

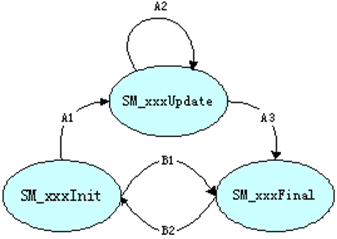


图3-2三段式结构

|  |  |
| --- | --- |
| 函数名 | 功能 |
| SM\_OpenSecPipe | 调用进程在安全模块内部分配资源。多个共享进程（或线程）可以独立地调用该函数，得到各自的操作句柄phPipe。该命令成功返回后，如果某个进程（或线程）不再使用安全管道，它必须调用函数[SM\_CloseSecPipe](file:///F:\云京电子印章系统\SM_CloseSecPipe.htm)命令关闭安全管道，释放phPipe。否则，会造成安全模块内部资源无效占用，参考函数SM\_CloseSecPipe的说明 |
| [SM\_CloseSecPipe](file:///F:\reference\wstb\doc\SM_CloseSecPipe.htm) | 该函数释放安全模块内部对应于hPipe的占用资源。 |
| SM\_EncryptInit | 该函数用参数pstKey所标识的密钥来初始化加密过程。该函数和函数[SM\_EncryptUpdate](file:///F:\云京电子印章系统\SM_EncryptUpdate.htm)、[SM\_EncryptFinal](file:///F:\云京电子印章系统\SM_EncryptFinal.htm)一起构成了加密数据的三段式结构 |
| SM\_EncryptUpdate | 加密中间数据块  明文数据块的字节数，必须为算法分组长度的整数倍。算法的分组长度可通过调用函数[SM\_GetMechanismInfo](file:///F:\云京电子印章系统\SM_GetMechanismInfo.htm)获取。 |
| SM\_EncryptFinal | 加密最后一个数据块  该函数执行后，下一轮加密操作如需更换密钥，则必须重新调用函数SM\_EncryptInit；如继续使用上一轮的密钥，则可直接调用函数SM\_EncryptUpdate。 |
| SM\_Encrypt | 该函数加密一个任意大小的数据块 |
| SM\_DecryptInit | 该函数执行解密初始化操作  pAlgo对称算法类型，具体参数设置方式请参考结构体[SM\_ALGORITHM](file:///F:\云京电子印章系统\SM_ALGORITHM.htm)说明。此处指算法工作模式为CBC |
| SM\_DecryptUpdate | 解密中间数据块 |
| SM\_DecryptFinal | 解密最后一个数据块 |
| SM\_Decrypt | 解密一个任意大小的数据块 |
| SM\_DigestInit | 执行对数据块摘要的初始化操作  当参数pstKey为SM\_NULL时，表示进行Hash运算，pAlgo指出工作模式采用Hash算法 |
| [SM\_DigestUpdate](mk:@MSITStore:D:\API业务小组\项目文档\项目文档\帮助文档-API5.0\api5.0help.chm::/SM_HashUpdate.htm) | 该函数对中间数据块进行摘要运算  数据块的字节数，必须为摘要分组长度的整数倍。 |
| [SM\_DigestFinal](mk:@MSITStore:D:\API业务小组\项目文档\项目文档\帮助文档-API5.0\api5.0help.chm::/SM_HashFinal.htm) | 该函数对最后一个数据块进行摘要运算 |
| SM\_Digest | 该函数执行对数据块摘要的操作 |
| SM\_ECCSignature | 该函数使用ECC私钥对数据进行签名运算 |
| SM\_ECCVerify | 使用ECC公开密钥对输入数据进行验证签名运算 |
| SM\_GenRandom | 从安全模块上获取一定长度随机数 |
| SM\_ECCEncrypt | 该函数使用用户指定ECC公开密钥对输入数据进行ECC加密运算操作。 |
| SM\_ECCDecrypt | 该函数使用用户指定ECC私钥对输入数据进行ECC解密运算操作 |
| SM\_Login | 该函数使用用户认证介质口令登录模块平台  该函数调用成功之前，所有非对称密钥对等敏感数据均处于加密状态，禁止进行私钥的导入\导出以及私钥加解密运算等操作；该函数调用成功后，上述数据即可在模块内部脱密，并允许私钥的导入\导出以及私钥加解密运算等操作 |
| SM\_Logout | 函数用来退出登录状态 |

* [密钥访问](file:///F:\reference\wstb\doc\KeyManage.htm)管理

密钥访问是基于[安全管道访问](file:///F:\云京电子印章系统\SecPipe.htm)子系统，负责设备对称密钥和非对称密钥管理的子系统。通过访问该子系统，用户可实现密钥的创建、导入、导出以及销毁等安全管理应用。

|  |  |
| --- | --- |
| 函数名 | 功能 |
| SM\_ImportKey | 该函数向模块平台内部导入对称商用算法所需的对称密钥WK  参数hKEK是保护(解密)WK的密钥句柄，依据pKEKAlgo的不同，可为非对称密钥句柄、主密钥句柄、对称密钥句柄，或为NULL即WK未被保护。pKEKAlgo指定WK的保护(解密)模式，可以采用对称商用算法ECB模式解密、对称商用算法CBC模式解密、非对称密钥解密三种方式 |
| SM\_ExportKey | 从模块平台内部导出对称商用算法所需的对称密钥 |
| SM\_DestroyKey | 清除模块平台内部维护的对称商用算法所需的密钥WK。  该函数清除指定的模块平台内部维护的对称工作密钥WK |
| SM\_ImportPrivateKey | 该函数向模块平台内部导入非对称商用算法私有密钥。  参数hKEK为保护商用算法类型，密钥类型只能为对称商用算法。  该函数将从外部得到的私有密钥置入模块平台，供加解密过程使用。对于不再使用的密钥应及时调用SM\_DestroyPrivateKey函数，以便释放模块平台内部占用的资源 |
| SM\_ImportPublicKey | 该函数向模块平台内部导入非对称商用算法公开密钥 |
| SM\_DestroyPrivateKey | 该函数清除模块平台内部维护的非对称商用算法私有密钥 |
| SM\_DestroyPublicKey | 该函数清除模块平台内部维护的非对称商用算法公开密钥 |
| SM\_GenerateKey | 该函数在模块平台内部产生对称商用算法所需的对称密钥WK |
| SM\_GenerateKeyPair | 该函数在模块平台内部生成非对称密钥对 |
| SM\_ExportPrivateKey | 该函数从模块平台内部导出非对称商用算法私有密钥  参数hKEK保护商用算法密钥访问句柄, 依据所采用保护商用算法的不同，可为主密钥句柄、对称密钥句柄，或为NULL即不保护。 |
| SM\_ExportPublicKey | 该函数从模块平台内部导出非对称商用算法公开密钥 |
| SM\_UpdateKeyPair | 该函数用于更换认证介质上的配用公私钥。  参数wKeyUse用于指明更换的公私钥用于ECC加解密或者ECC签名验签 |
| SM\_GetCfgKeyHandle | 该函数该用于获取认证介质上的配用密钥，当不再使用密钥句柄时，应当调用SM\_CloseTokKeyHdl关闭句柄。  在成功调用[SM\_Login](file:///F:\云京电子印章系统\SM_Login.htm)函数登录模块平台以后该函数的调用才有效 |
| SM\_CloseTokKeyHdl | 该函数用于关闭认证介质上的配用公私钥句柄 |

2、关键数据结构

* 密钥信息SM\_BLOB\_KEY

typedef struct \_SM\_BLOB\_DATA

{

SM\_UINT uiDataLen; //密钥数据的有效长度，以字节为单位

PSM\_BYTE pbyData; //密钥数据的首地址，该参数必须由用户自行分配缓冲区大小

}SM\_BLOB\_KEY, \*PSM\_BLOB\_KEY;

|  |  |  |
| --- | --- | --- |
| ulDataLen | pbyData | 说明 |
| 密钥句柄（SM\_KEY\_HANDLE）的长度（32位下为4, 64位下为8） | 密钥访问句柄 | 表明此时使用密钥访问句柄 |
| >16 | 密钥有效数据 | 表明此时使用密钥实际有效数据 |

* 密钥属性[SM\_KEY\_ATTRIBUTE](file:///F:\reference\wstb\doc\SM_KEY_ATTRIBUTE.htm)

typedef struct \_SM\_KEY\_ATTRIBUTE {

SM\_UINT uiObjectClass; //密钥对象类型

SM\_KEY\_TYPE KeyType; //密钥类型

SM\_UINT uiKeyLabel; //用户指定的密钥索引标识

SM\_BYTE byStartDate[4]; //用户指定的密钥启用日期

SM\_BYTE byEndDate[4]; //用户指定的密钥结束日期

SM\_UINT uiFlags; //密钥的属性

SM\_VOID\_PTR pParameter; //密钥参数，若该参数是ECC密钥属性访问指针，则应该指向SM\_ECC\_PARAMETER数据结构；否则为SM\_NULL

SM\_UINT uiParameterLen; //密钥参数长度

} SM\_KEY\_ATTRIBUTE, \*PSM\_KEY\_ATTRIBUTE;

|  |  |
| --- | --- |
| uiObjectClass | 说明 |
| SMO\_PUBLIC\_KEY | 0x00000002 |
| SMO\_PRIVATE\_KEY | 0x00000003 |
| SMO\_SECRET\_KEY | 0x00000004 |

|  |  |
| --- | --- |
| KeyType | 说明 |
| SM\_KEY\_ALG\* | 使用ALG\*算法进行运算的对称密钥 |
| SM\_KEY\_ECC\_FP\_PUBLIC | ECC P域的公钥 |
| SM\_KEY\_ECC\_FP\_PRIVATE | ECC P域的私钥 |

|  |  |
| --- | --- |
| uiFlags | 说明 |
| SMKA\_EXTRACTABLE | TRUE 可输出  FALSE 不可输出（缺省值） |
| SMKA\_MODIFIABLE | TRUE 可修改对象属性  FALSE 不可修改对象属性 |
| SMKA\_ENCRYPT | TURE 密钥可用于数据加密  FALSE 密钥不能用于数据加密 |
| SMKA\_DECRYPT | TURE 密钥可用于数据解密  FALSE 密钥不能用于数据解密 |
| SMKA\_SIGN | TURE 密钥可用于签名（包括产生MAC码）  FALSE 密钥不能用于签名（包括产生MAC码） |
| SMKA\_VERIFY | TURE 密钥可用于验证（包括验证MAC码）  FALSE 密钥不能用于验证（包括验证MAC码） |
| SMKA\_WRAP | TURE 密钥可用于保护其他密钥  FALSE 密钥不能用于保护其他密钥 |
| SMKA\_UNWRAP | TURE 密钥可用于恢复其他密钥  FALSE 密钥不能用于恢复其他密钥 |

ECC密钥访问属性SM\_ECC\_PARAMETER

typedef struct \_SM\_ECC\_PARAMETER

{

SM\_UINT uiModulusBits; // ECC模长，以比特为单位

SM\_VOID\_PTR pParameter; //保留参数

SM\_UINT uiParameterLen; //保留参数

} SM\_ECC\_PARAMETER, \*PSM\_ECC\_PARAMETER;

* 算法类型[SM\_ALGORITHM](file:///F:\reference\wstb\doc\SM_ALGORITHM.htm)

typedef struct \_SM\_ALGORITHM{

SM\_ALGORITHM\_TYPE AlgoType; //算法工作模式

SM\_VOID\_PTR pParameter; //算法参数，对应关系见下表

SM\_UINT uiParameterLen; //算法参数长度，为pParameter的字节数

SM\_UINT uiReserve; //算法保留参数

} SM\_ALGORITHM, \*PSM\_ALGORITHM;

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| AlgoType | pParameter | ulParameterLen | ulReserve | 说明 |
| SMM\_ALG34\_ECB | SM\_NULL | 0 | 0 | ECB方式 |
| SMM\_ALG34\_CBC | 指向IV数据块首地址 | SMMA\_ALG34\_IV\_LEN | 0 | CBC方式 |
| SMM\_HASH1\_TZ | SM\_NULL或指向IV数据块首地址 | 0或 SMMA\_HASH1\_TZ\_LEN | 0 | HASH方式 |
| SMM\_ECC\_\* | SM\_NULL | 0 | 算法模长（256） | ECC运算 |

对于MAC和HASH算法，通常参数pParameter取值为SM\_NULL，参数uiParameterLen取值为0；对于部分指定算法要求指定IV数据，此时才在上述两个参数中指定相应的IV数据特性。

3.2.2 数据库接口

本设计采用MySQL数据库。使用 JAVA的Mybatis 连接 MySQL 数据库。

3.2.3 OCSP接口

参考规范GM/T0014-2012（附录C.4）、GB/T 19713-2005。

请求格式：

OCSPRequest ::= SEQUENCE{

tbsRequest TBSRequest,

optionalSignature [0] EXPLICT Signature OPTIONAL

}

TBSRequest ::= SEQUENCE{

version [0] EXPLICIT Version DEFAULT v1, -- 缺省值，则不编码

requestorName [1] EXPLICIT GeneralName OPTIONAL,

requestList SEQUENCE OF Request,

requestExtensions [2] EXPLICT Extensions OPTIONAL

}

Signature ::= SEQUENCE {

signatureAlgorithm AlgorithmIdentifier,

signature BIT STRING,

certs [0] EXPLICT SEQUENCE OF Certificate OPTIONAL

}

Version ::= INTEGER { v1(0)}

Request ::= SEQUENCE {

reqCert CertID,

singleRequestExtensions [0] EXPLICIT Extensions OPTIONAL

}

CertID ::= SEQUENCE {

hashAlgorithm AlgorithmIdentifier,

issuerNameHash OCTET STRING, ---Hash of Issuer’s DN

issuerKeyHash OCTET STRING, ---Hash of Issuer public key

serialNumber CertificateSerialNumber

}

响应格式：

OCSPResponse ::= SEQUENCE {

responseStatus OCSPResponseStatus,

responseBytes [0] EXPLICIT ResponseBytes OPTIONAL

}

OCSPResponseStatus ::= ENUMBERATED{

successful (0),

malformedRequest (1),

internalError (2),

tryLater (3), -- 4没有使用

sigRequired (5),

unauthorized (6)

}

ResponseBytes ::= SEQUENCE {

responseType OBJECT IDENTIFIER,

response OCTET STRING

}

对于基础OCSP，responseType是id-pkix-ocsp-basic，response是BasicOCSPResponse的DER编码。

id-pkix-ocsp OBJECT IDENTIFIER ::= { id-ad-ocsp }

id-pkix-ocsp-basic OBJECT IDENTIFIER ::= { id-pkix-ocsp 1 }

BasicOCSPResponse ::= SEQUENCE {

tbsResponseData ResponseData,

signatureAlgorithm AlgorithmIdentifier,

signature BIT STRING, -- ResponseData的DER编码的签名值

certs [0] EXPLICIT SEQUENCE OF Certificate OPTIONAL

}

ResponseData ::= SEQUENCE {

version [0] EXPLICIT Version DEFAULT v1,

responderID ResponderID,

producedAt GeneralizedTime,

responses SEQUENCE OF SingleResponse,

responseExtensions [1] EXPLICIT Extensions OPTIONAL

}

ResponderID ::= CHOICE {

byName [1] Name,

byKey [2] KeyHash -- 公钥的摘要值(SHA1)

}

SingleResponse ::= SEQUENCE {

certID CertID,

certStatus CertStatus,

thisUpdate GeneralizedTime,

nextUpdate [0] EXPLICIT GeneralizedTime OPTIONAL,

singleExtensions [1] EXPLICIT Extensions OPTIONAL

}

CertStatus ::= CHOICE {

good [0] IMPLICIT NULL,

revoked [1] IMPLICIT RevokedInfo,

unknown [2] IMPLICIT UnknownInfo -- NULL

}

RevokedInfo ::= SEQUENCE {

revocationTime GeneralizedTime,

revocationReason [0] EXPLICIT CRLReason OPTIONAL

}

3.2.4 证书CRL接口

CRL是经CA签名的证书作废列表，用于证书冻结和撤销时对证书有效性状态控制。一般数字证书中都有 CRL分发点地址，提供了HTTP和LDAP方式访问。通过解析证书的扩展项可以获取到CRL地址，然后使用相应方式下载CRL进行验证。

在用LDAP方式下载CRL时，注意LDAP协议名称要大写，不然访问会出错。下载的CRL格式可能是BASE64编码的，需要判断转换成DER编码二进制格式。

CRL分全量CRL和增量CRL，另外还有分段CRL，即同个CA的证书分不同的CRL地址段，主要是为了分流服务器负载。CRL有生效日期和下次更新时间，一般是定时更新，所以CRL并不是即时状态的。因此还有OCSP在线证书状态协议，可以即时的查询证书状态。

* CRL的数据结构

参考规范GM/T 0015

CertificateList ::= SEQUENCE {

tbsCertList TBSCertList,

signatureAlgorithm AlgorithmIdentifier,

signatureValue BIT STRING ---对tbsCertList的ASN.1 DER编码进行签名，结果编成ASN.1

}

TBSCertList ::= SEQUENCE {

version Version OPTIONAL, – 如果使用了extension项，必须存在，且为version 2(即整数1)

signature AlgorithmIdentifier,

issuer Name,

thisUpdate Time, --2049年之前用UTCTime

nextUpdate Time OPTIONAL, ---下次更新不能晚于该时间

revokedCertificates SEQUENCE OF SEQUENCE {

userCertificate CertificateSerialNumber, -- 被撤销的证书序列号

revocationDate Time,

crlEntryExtensions Extensions OPTIONAL} OPTIONAL, --被撤销的原因

crlExtenstions [0] EXPLICIT Extensions OPTIONAL

}

AlgorithmIdentifier ::= SEQUENCE {

algorithm OBJECT IDENTIFIER,

parameters ANY DEFINED BY algorithm OPTIONA --SM2 无参数

}

3.2.5 证书申请/更新接口

参考规范GM/T 0014 、PKCS#10以及相关CA接口。

内部模块接口

|  |  |  |  |
| --- | --- | --- | --- |
| 模块 | 功能 | 描述 | 备注 |
| 密码模块 | SM2算法的数字签名功能 | 输入：SM2密钥索引，H 字节串  输出：SM2Signature | SM2算法的签名数据结构应遵循GM/T 0009  签名数据的预处理  H=SM3(Z||M)，其中Z=SM3(ENTL||ID||a||b||xG||yG||xA||yA)  用户身份ID默认值:  0x31323334353637383132333435363738  密码卡生成密钥对时，只导出公钥  模块接口定义参考JAVA API以及bouncyCastle的crypto包。 |
| SM2算法的数字签名验证功能 | 输入：H 字节串，签名SM2Signature ，公钥SM2PublicKey  输出： “真”表示验证通过 |
| SM3算法的摘要计算功能 | 输入：消息  输出：摘要 |
| ECC签名密钥生成功能 | 输入：无  输出： SM2密钥索引，公钥SM2PublicKey |
| SM4加解密功能 | 输入：数据，SM4的密钥索引  输出：加解密结果 |
|  | 随机数生成 |  |  |
| 证书模块 | 证书申请功能 | 输入：用户信息、签名公钥  输出：签名证书 | PEM编码就是在DER编码基础上进行BASE64编码，然后添加一些头尾信息组成的。  CRL是ASN.1 DER编码。CRL解析后，保存到证书表中  模块接口定义参考JAVA API以及bouncyCastle的crypto包。 |
| 证书验证功能 | 输入：证书(PEM格式)  输出：“真”表示验证通过 |
| 证书解析功能 | 获取证书中包含的公钥、序列号  输入：证书(PEM格式)  输出：公钥或者序列号 |
| CRL解析功能 | 输入：CRL文件  输出：无 |
| OCSP处理功能 | 输入：证书序列号  输出：证书状态 |
| 签章模块 | 签章生成功能 | 输入：签章相关信息  输出：“真”表示验证通过 |  |
| 签章验证功能 | 输入：签章数据(ASN.1 DER编码)  输出： “真”表示验证通过 |
| 印章模块 | 印章生成功能 | 输入：印章相关信息  输出：印章数据(ASN.1 DER编码) |  |
| 印章验证功能 | 输入：印章数据(ASN.1 DER编码)  输出： “真”表示验证通过 |
| ASN.1编解码模块 | BER/DER编码功能 | 输入：Java对象  输出：byte数组（ASN.1格式） | 模块接口定义参考bouncyCastle的asn1包。  编码按照DER的规则，因为DER是BER的子集。  解码时，如果有指定，按照指定来，否则要按照BER。 |
| BER/DER解码功能 | 输入：byte数组（ASN.1格式）  输出：Java对象 |
| Pdf模块 | Pdf生成功能 | 输入：电子文档  输出：pdf文档 |  |
| 盖章功能 | 输入：pdf文档、签章  输出：盖了章的pdf文档 |
| 系统安全模块 | 初始化功能 | 印章管理系统初始化  签章系统初始化 |  |
| 自检功能 | 检查系统的状态 |  |
| 连接配置 | 连接CA、NTP的配置 |  |
| 用户管理 | 用户的注册、PIN更新 |  |
| 证书/密钥管理 | 导入、导出、更新、备份等 |  |
| 印章管理 | 制作、更新、验证等 |  |
| 日志管理 | 查找、归档等 |  |
| 访问控制 | IP白名单设置、登录验证 |  |

系统数据结构设计

逻辑结构设计

4.1.1 密码模块

1、模块结构



2、关键数据结构

* 密钥数据格式（ASN.1）

SM2PrivateKey ::= INTEGER

SM2PublicKey ::= BIT STRING -- 04||X||Y，公钥的x分量和y分量，均为256位

* 签名数据格式（ASN.1）

参考规范GM/T0010

OID = 1.2.156.10197.6.1.4.2.2

SignedData ::= SEQUENCE{

version Version,

digestAlgorithms DigestAlgorithmIdentifiers,

contentInfo SM2Signature,

certificates[0] IMPLICIT ExtendedCertificatesAndCertificates OPTIONAL,

crls[1] IMPLICIT CertificatesRevocationLists OPTIONAL,

signerInfos SignerInfos

}

DigestAlgorithmIdentifiers ::= SET OF DigestAlgorithmIdentifier

SM2Signature ::={

R INTEGER, -- 256位

S INTEGER -- 256位

}

CertificateRevocationLists ::= SET OF CerfiticateRevocationList

ExtendedCertificatesAndCertificates ::=SET OF ExtendedCertificateOrCertificate

ExtendedCertificateOrCertificate ::= CHOICE {

certificate Certificate, -- X.509

extendedCertificate [0] IMPLICIT ExtendedCertificate

}

SignerInfos ::= SET OF SignerInfo

SignerInfo ::= SEQUENCE{

version Version,

issuerAndSerialNumber IssuerAndSerialNumber,

digestAlgorithm DigestAlgorithmIdentifier,

authencateAttributes[0] IMPLICT Attributes OPTIONAL,

digestEncryptionAlgorithm DigestEncryptionAlgorithmIdentifier,

encryptedDigest OCTET STRING

}

* 数字信封格式（ASN.1）

用于为接收者的data、digestedData或signedData做数字信封。加密数据是用数据加密密钥加密的，数据加密密钥是用接收者的公钥加密的

OID = 1.2.156.10197.6.1.4.2.3

EnvelopedData ::= SEQUENCE {

version Version, -- version(1)

recipientInfos RecipientInfos,

encryptedContentInfo EncryptedContentInfo

}

RecipientInfos ::= SET OF RecipientInfo

RecipientInfo ::= SEQUENCE {

version Version,

issuerAndSerialNumber IssuerAndSerialNumber,

keyEncryptionAlgorithm KeyEncryptionAlgorithmIdentifier, --SM2加密算法

encryptedKey OCTET STRING -- SM2Cipher

}

EncryptedContentInfo ::= SEQUENCE {

contentType ContentType,

contentEncryptionAlgorithm ContentEncryptionAlgorithmIdentifier,

encryptedContent[0] IMPLICIT EncryptedContent OPTIONAL,

sharedInfo[1] IMPLICT OCTET STRING OPTIONAL,

sharedInfo[2] IMPLICT OCTET STRING OPTIONAL

}

SM2Cipher ::= SEQUENCE {

XCoordinate INTEGER,

YCoordinate INTEGER,

HASH OCTET STRING SIZE(32), -- SM3对明文数据计算得到的杂凑值

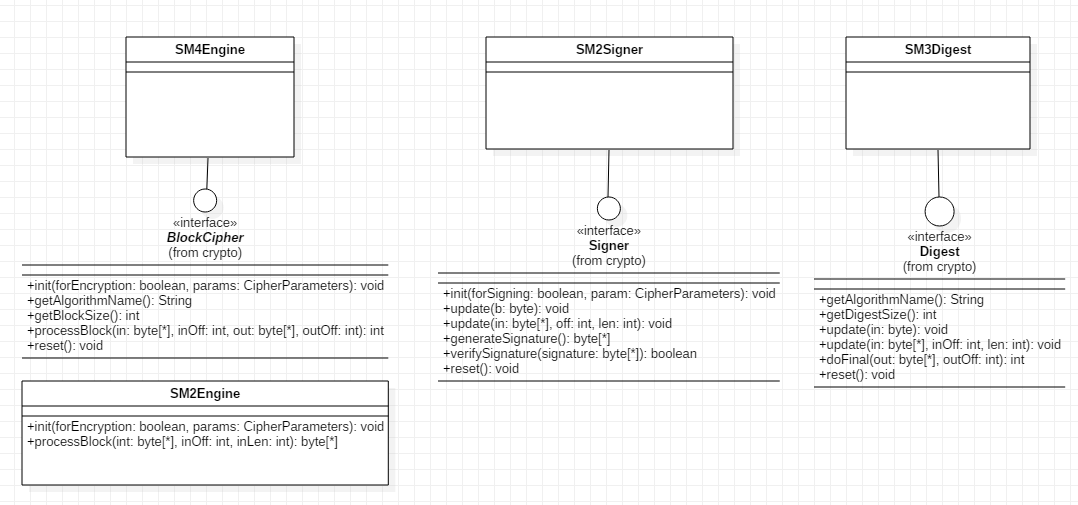
CipherText OCTET STRING SIZE

}

3、关键流程

无

4、类设计



SM4Engine：SM4加解密引擎。输入输出明密文数据采用普通数据格式（非ASN.1）。

SM2engine: SM2 加解密引擎。输入输出明密文数据采用普通数据格式（非ASN.1）。

SM2Signer: SM2签名验证签名引擎。注意：签名数据封装成SM2Signature的格式（ASN.1 DER编码）

SM3Digest: SM3摘要引擎。输出摘要采用普通数据格式（非ASN.1）。

4.1.2 证书模块

1、模块结构



2、关键数据结构

以下数据结构由GM/T0015-2012规范定义。

* 数字证书的基本数据结构（ASN.1）

Certificate ::= SEQUENCE {

tbsCertificate TBSCertificate,

signatureAlgorithm AlgorithmIdentifier,

signatureValue BIT STRING -- ASN.1 DER编码的tbsCertificate作为签名的输入，签名结果SM2Signatue编码成BIT STRING

}

TBSCertificate ::= SEQUENCE {

version [2] EXPLICIT Version DEFAUT v3,

serialNumber CertificateSerialNumber, -- 不大于20个8位字节

signature AlgorithmIdentifier, -- 与signatureAlgorithm项的算法标识符相同

issuer Name,

validity Validity,

subject Name,

subjectPublicKeyInfo SubjectPublicKeyInfo,

issuerUniqueID [1] IMPLICIT UniqueIdentifier OPTIONAL,

subjectUniqueID [2] IMPLICIT UniqueIdentifier OPTIONAL,

extensions [3] EXPLICIT Extensions OPTIONAL

}

Version ::= INTEGER {v1(0), v2(1), v3(2) }

CertificateSerialNumber ::= INTEGER

Validity ::= SEQUENCE {

notBefore Time,

notAfter Time

}

Time ::= CHOICE {

utcTime UTCTime, -- 2049年之前(含2049)时间编码为UTCTime，必须用格林威治标准时间，其格式YYMMDDHHMMSSZ

generalTime GeneralizedTime

}

UniqueIdentifier ::= BIT STRING

SubjectPublicKeyInfo ::= SEQUENCE {

algorithm AlgorithmIdentifier, -- SM2

subjectPublicKey BIT STRING

}

Extensions ::= SEQUENCE SIZE (1..Max) OF Extension

Extension ::= SEQUENCE {

extnID OBJECT IDENTIFIER,

critical BOOLEAN DEFAULT FALSE,

extnValue OCTET STRING

}

AlgorithmIdentifier ::= SEQUENCE {

Algorithm OBJECT IDENTIFIER,

parameters ANY DEFINED BY algorithm OPTIONAL – 依赖于所标识的算法, SM2无参数

}

Name ::= CHOICE { RDNSequence }

RDNSequence ::= SEQUENCE OF RelativeDistinguishedName

RelativeDistinguishedName ::= SET OF AttributeTypeAndValue

AttributeTypeAndValue ::= SEQUENCE {

type AttributeType,

value AttributeValue

}

AttributeType ::= OBJECT IDENTIFIER

AttributeValue ::= ANY DEFINED BY AttributeType -- 通常是DirectoryString类型

DirectoryString ::= CHOICE {

teletexString TeletexString(SIZE(1..MAX)),

printableString PrintableString(SIZE(1..MAX)),

universalString UniversalString(SIZE(1..MAX)),

utf8String UTF8String(SIZE(1..MAX)),

bmpString BMPString(SIZE(1..MAX))

}

* 数字证书扩展域

每个扩展项包括一个对象标识符OID和一个ASN.1结构。OID作为extnID，ASN.1编码结构就是8bit字符串extnValue的值。

id-ce OBJECT IDENTIFIER ::= {joint-iso-ccitt(2) ds(5) 29}

1. 颁发机构密钥标识符用于识别与证书签名私钥对应的公钥。非关键。

id-ce-authorityKeyIdentifier OBJECT IDENTIFIER ::= {id-ce 35}

AuthorityKeyIdentifier ::= SEQUENCE {

keyIdentifier [0] KeyIdentifier OPTIONAL,

authorityCertIssuer [1] GeneralNames OPTIONAL,

authorityCertSerialNumber [2] CertificateSerialNumber OPTIONAL,

}

(WITH COMPONENTS {…, authorityCertIssuer PRESENT, authorityCertSerialNumber PRESENT } | WITH COMPONENTS {…, authorityCertIssuer ABSENT, authorityCertSerialNumber ABSENT })

KeyIdentifier ::= OCTET STRING

除自签证书外，所有证书必须包含本扩展，必须包含keyIdentifier

1. 主体密钥标识符用于区分同一主体的不同密钥。非关键。

id-ce-subjectKeyIdentifier OBJECT IDENTIFIER ::= {id-ce 14}

SubjectKeyIdentifier ::= KeyIdentifier

1. 密钥用法，如果标记为关键，则该证书只用于相应密钥用法位置为”1”的用途

id-ce-keyUsage OBJECT IDENTIFIER ::= {id-ce 15}

KeyUsage ::= BIT STRING{

digitalSignature (0), -- 验证数字签名(签名证书)

nonrepudiation (1), -- 验证用来提供抗依赖服务的数字签名(签名证书)

keyEncipherment (2), -- 加密密钥或者其他安全信息，例如密钥传输

dataEncipherment (3), -- 加密用户数据(加密证书)

keyAgreement (4), 公开密钥协商密钥(加密证书)

keyCertSign (5), -- 验证证书CA签名（只用于CA证书）

cRLSign (6), -- 验证CRL的CA签名

encipherOnly (7), -- 与keyAgreement一起使用(加密证书)

decipherOnly (8) -- 与keyAgreement一起使用(加密证书)

}

所有的CA证书必须包含本扩展，且包含keyCertSign。

1. 扩展密钥用法

id-ce-extKeyUsage OBJECT IDENTIFIER ::= {id-ce 37}

ExtKeyUsageSyntax ::= SEQUENCE SIZE (1..MAX) OF KeyPurposeId

KeyPurposeId ::= OBJECT IDENTIFIER

1. 私有密钥使用期，只用于数字签名密钥。非关键。

id-ce-privateKeyUsagePeriod OBJECT IDENTIFIER ::= {id-ce 16}

PrivateKeyUsagePeriod ::= SEQUENCE {

notBefore [0] GeneralizedTime OPTIONAL,

notAfter [1] GeneralizedTime OPTIONAL

}

1. 证书策略

用户证书中，策略信息条目描述了证书发发所依据的策略及证书的应用目的；CA证书中，策略条目制定了包含这个证书的验证路径的策略集合。

id-ce-certificatePolicies OBJECT IDENTIFIER ::= {id-ce 32}

certificatePolicies ::= SEQUENCE SIZE (1..MAX) OF PolicyInformation

PolicyInformation ::= SEQUENCE {

policyIdentifier CertPolicyId,

policyQualifiers SEQUENCE SIZE (1..MAX) OF PolicyQualifierInfo OPTIONAL

}

CertPolicyId ::= OBJECT IDENTIFIER

PolicyQualifierInfo ::= SEQUENCE {

policyQualifierId PolicyQualifierId,

qualifier ANY DEFINED BY policyQualifierId

}

PolicyQualifierId ::= OBJECT IDENTIFIER (id-qt-cps | id-qt-unotice)

id-qt OBJECT IDENTIFIER ::= {id-pkix 2}

id-qt-cps OBJECT IDENTIFIER ::= {id-qt 1}

id-qt- unotice OBJECT IDENTIFIER ::= {id-qt 2}

1. 策略映射，只用于CA证书。

颁发者CA认为issuerDomainPolicy与主体CA的subjectDomainPolicy等效。

id-ce-policyMappings OBJECT IDENTIFIER ::= {id-ce 33}

PolicyMappingsSyntax ::= SEQUENCE SIZE (1..MAX) OF SEQUENCE {

issuerDomainPolicy CertPolicyId,

subjectDomainPolicy CertPolicyId

}

1. 主体替换名称

把附加身份加到证书的主体上，包括电子邮件地址、DNS名称、IP地址、URI等。

id-ce-subjectAltName OBJECT IDENTIFIER ::= {id-ce 37}

SubjectAltName ::= GeneralNames

GeneralNames ::= SEQUENCE SIZE (1 .. MAX) OF GeneralName

1. 颁发者替换名称

id-ce-issuerAltName OBJECT IDENTIFIER ::= {id-ce 18}

IssuerAltName ::= GeneralNames

1. 主体目录属性，非关键的

id-ce-subjectDirectoryAttributes OBJECT IDENTIFIER ::= {id-ce 9}

SubjectDirectoryAttributes ::= SEQUENCE SIZE (1 .. MAX) OF Attribute

1. 基本限制，CA证书中必须包括，关键的。

id-ce-basicConstraints OBJECT IDENTIFIER ::= {id-ce 19}

BasicConstraintsSyntax ::= SEQUENCE {

CA BOOLEAN DEFAULT FALSE, -- 此公钥证书是否可以用来验证证书签名

pathLenConstrain INTEGER (0..MAX) OPTIONAL –认证路径中最多的CA证书数，0标识只可以向终端实体签发证书。

}

1. 名称限制，CA证书中使用，指定名称空间。

id-ce-nameConstraints OBJECT IDENTIFIER ::= {id-ce 30}

nameConstraintsSyntax ::= SEQUENCE {

permittedSubtrees [0] GeneralSubtrees OPTIONAL,

excludedSubtrees [0] GeneralSubtrees OPTIONAL

}

GeneralSubtrees ::= SEQUENCE SIZE (1 .. MAX) OF GeneralSubtree

GeneralSubtree ::= SEQUENCE {

base GeneralName,

minimum [0] BaseDistance DEFAULT 0,

maximum [0] BaseDistance OPTIONAL

}

1. 策略限制，CA证书中使用，限制路径确认。

id-ce-policyConstraints OBJECT IDENTIFIER ::= {id-ce 36}

PolicyConstraints ::= SEQUENCE {

requireExplicitPolicy [0] SkipCerts OPTIONAL,

inhibitPolicyMapping [0] SkipCerts OPTIONAL

}

SkipCerts ::= INTEGER (0 .. MAX)

1. 证书撤销列表分发点。

id-ce-CRLDistributionPoints OBJECT IDENTIFIER ::= {id-ce 31}

CRLDistributionPointsSyntax::= SEQUENCE (1 .. MAX) OF DistributionPoint

1. 个人身份标识码，非关键。

id-IdentifyCode OBJECT IDENTIFIER ::= {1.2.156.10260.4.1.1}

IdentifyCode ::= CHOICE {

residenterCardNumber [0] PrintableString OPTIONAL,

militaryOfficerCardNumber [0] UTF8String OPTIONAL,

passportNumber [0] PrintableString OPTIONAL,

}

1. 企业工商注册号，非关键。

ID-ICRegistrationNumber OBJECT IDENTIFIER ::= {1.2.156.10260.4.1.3}

ICRegistrationNumber::= PrintableString

1. 企业组织机构代码，非关键。

ID-OrganizationCode OBJECT IDENTIFIER ::= {1.2.156.10260.4.1.4}

OrganizationCode::= PrintableString

1. 企业税号，非关键。

ID-TaxationNumber OBJECT IDENTIFIER ::= {1.2.156.10260.4.1.5}

TaxationNumber::= PrintableString

3、关键流程

* 证书验证流程

对证书的有效性的验证，包括验证证书有效期、验证证书签名有效性、验证是否在CRL中。



* CRL验证流程



* OCSP响应验证

在把OCSP响应视作有效之前，OCSP客户端应确认(参考规范GB/t19713-2005)：

a)收到的响应中所鉴别的证书应该和请求中的证书一致；

b)响应方的签名是有效的；

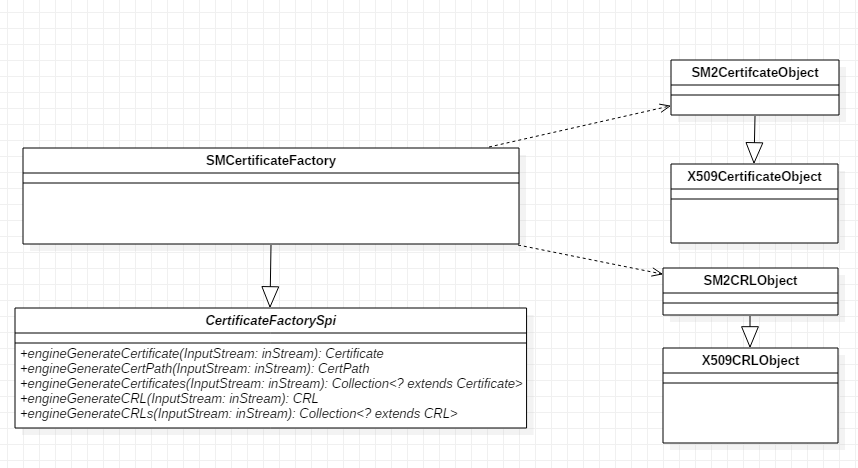
c)响应放的签名者身份和请求的预定接收者保持一致；

d)签名者已经被授权对响应进行签名；

e)指明证书状态的时间(thisUpdate)应为当前最近的时间；

f)如果设置了nextUpdate字段，此时间应晚于客户端当前时间。

4、类设计



4.1.3 签章模块

1、模块结构



2、关键数据结构

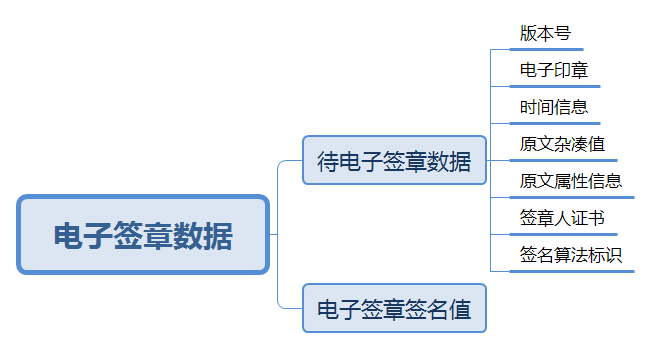


图4-7电子签章数据

电子签章数据的ASN.1定义为：

SES\_Signature::=SEQUENCE{

toSign TBS\_Sign，--待电子签章数据

signature BIT STRING--电子签章中签名值

}

TBS\_Sign::=SEQUENCE{

version INTEGER，--版本信息

Eseal SESeal，--电子印章

timeInfo BIT STRING，--签章时间信息

dataHash BIT STRING，--原文杂凑值

propertyInfo IA5String，--原文数据的属性信息

cert OCTET STRING．--签章人对应的签名证书

signatureAlgorithm OBJECT IDENTIFIER--签名算法标识

}

其中：

version:代表签章数据结构版本号；

eseal:代表生成电子签章使用的电子印章数据；

timeInfo：代表电子签章对应的时间信息，可以是时间戳，也可以是UTCTIME时间；

dataHash:代表待签名原文的杂凑值；

propertyInfo：代表原文数据的属性信息，如文档ID、日期、段落、原文内容的字节数、指示信息、签章保护范围等，此部分受签名保护，propertyInfo的具体含义可白行定义；本系统自定义propertyInfo:==”contents: ALL” ，就是对原文的全部章节进行保护（实现时应考虑该部分的可扩展性）。

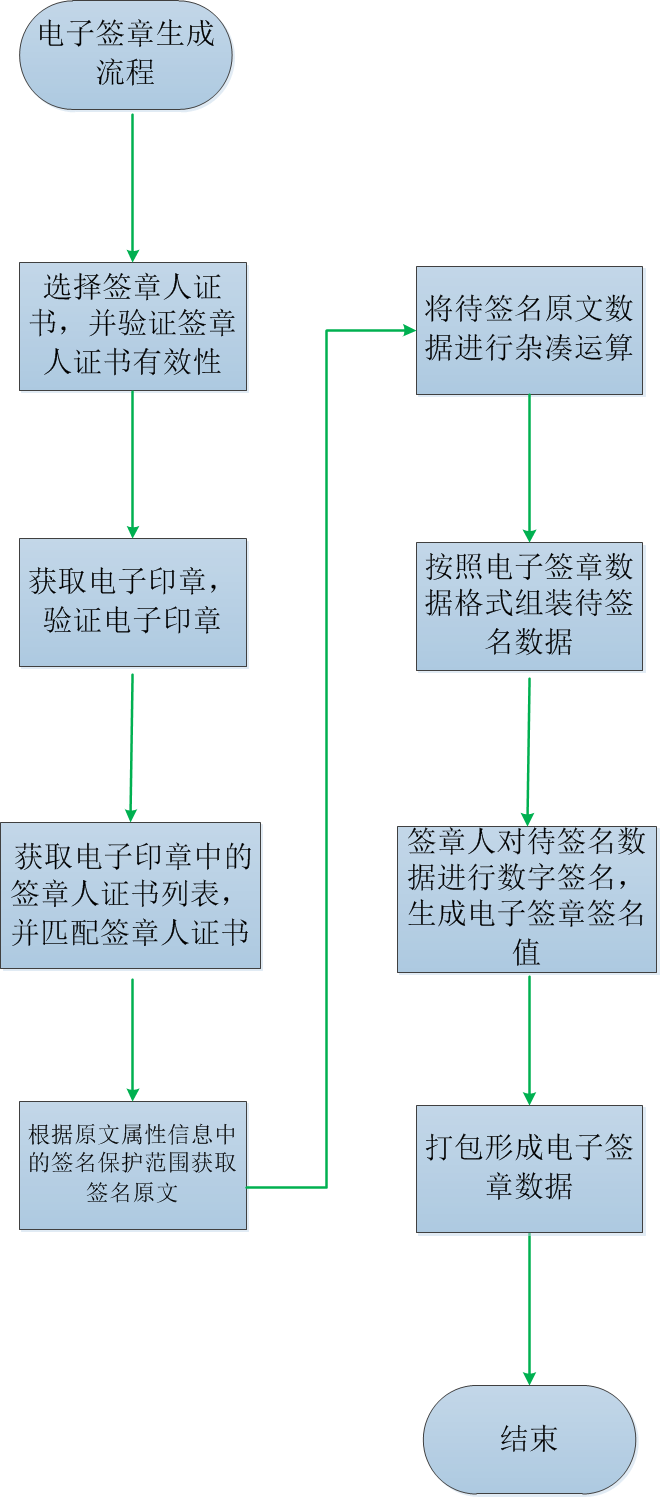
cert：代表执行本次签章操作的签章人数字证书；

signatureAlgorithm：代表签名算法OID，遵循GM/T 0006。例如，使用SM2签名的OID为1.2.156.10197.1.501;

signature：代表签章人对电子签章数据格式中版本号、电子印章、时间信息，原文杂凑值、原文属性信息、证书、签名算法标识组成的待签章数据TBS\_Sign进行数字签名。如果签名算法使用SM2，则遵循GMlT 0009；如果签名算法使用RSA，则遵循PKCS#1(本设计目前只支持SM2)。

原文杂凑值所采用的杂凑算法应与电子签章签名算法保持一致，如果签名算法是SM2，则杂凑算法应采用SM3算法，遵循GM/T 0003；如果签名算法是RSA，则杂凑算法应采用SHA1或SHA256算法，具体参见相应的国际标准。

3、关键流程



电子签章生成流程如下：

a) 选择拟进行电子签章的签章人证书，并验证签章人证书有效性：

验证签章人证书有效性，验证项至少包括：证书信任链验证、证书有效期验证、证书是否被吊销、密钥用法是否正确。如果签章人证书验证失败，返回失败原因并退jLH生成流程。

b) 获取电子印章，按照6.1.2验证印章的合规性和有效性。

c) 获取电子印章中的签章人证书列表，使用步骤a）中的签章人证书逐一进行证书数据二进制比对，确认签章人证书是否在签章人证书列表中。

如果比对失败或证书不在列表当中，返回失败原因并退出生成流程。

如果是因为签章人证书执行更新、重签发等操作而导致证书比对失败，此时需要重新制作印章，并重新进行签章生成流程。

d) 按照propertyInfo信息中的签名保护范围获取待签名原文。

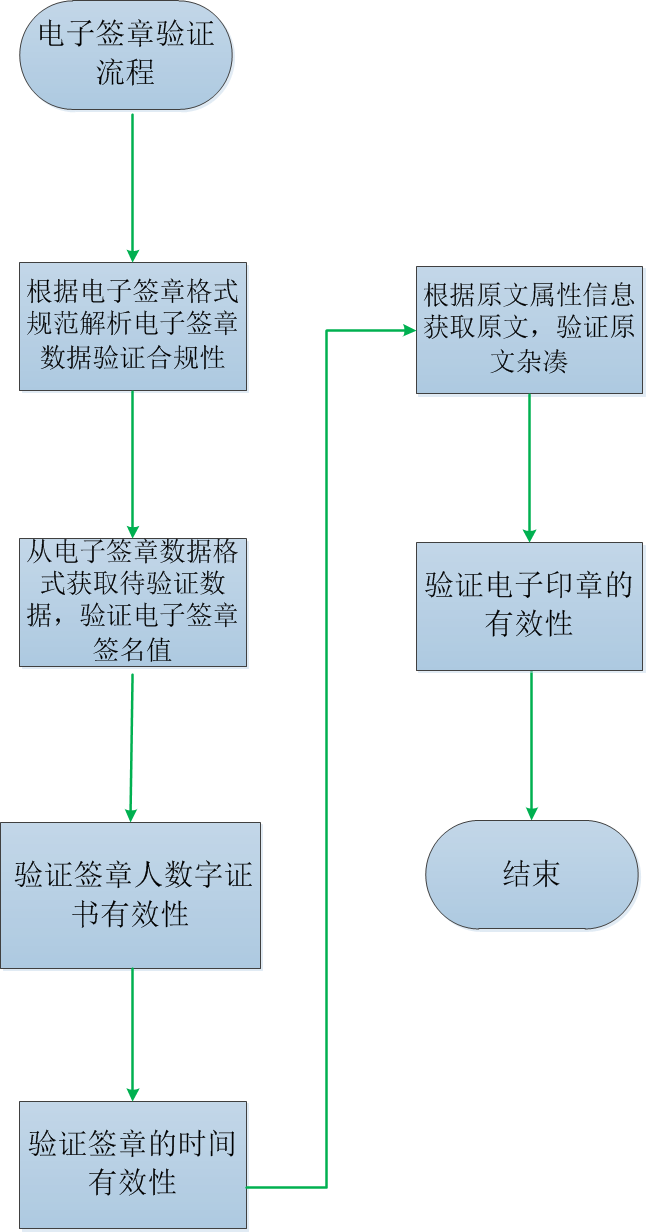
e) 将待签名原文数据进行杂凑运算，形成原文杂凑值。

f) 按照电子签章数据格式组装待签名数据：

待签名数据包括：版本号、电子印章、时间信息、原文杂凑值、原文属性信息、签章人证书、签名算法标识。

g) 签章人对待签名数据进行数字签名，生成电子签章签名值。

h) 按照电子签章数据格式，把以上数据打包形成电子签章数据



电子签章验证流程如下：

a) 验证电子签章数据格式的合规性：

根据电子签章格式规范解析电子签章数据。如果电子签章或电子印章数据格式不合规，则验证失败并退出验证流程。

b) 验证电子签章签名值是否正确：

从电子签章数据格式获取待验证数据，待验证数据包括：版本号、电子印章、时间信息、原文杂凑值、原文属性信息、签章人证书、签名算法标识，验证电子签章签名值的是否正确。

如果签名值验证不正确则验证失败，并将失败原因返回上层应用并退出验证流程。

c) 验证签章人数字证书有效性：

从电子签章数据获得签章人数字证书，验证签章人证书有效性，验证项至少包括：证书信任链验证、证书有效期验证、证书是否被吊销、密钥用法是否正确。

如果是由于证书信任链验证或密钥用法不正确导致的签章人证书有效性验证失败，则返回失败原因并退出验证流程。

如果是由于证书有效期或证书状态已吊销导致的签章人证书有效性验证失败，则还需要进一步结合签章时间进行综合判定。

d) 验证签章的时间有效性：

根据签章人数字证书有效期和电子签章中的时间信息进行比对，判断签章的时间有效性：

1) 如果签章时间处于签章人数字证书有效期内，并且证书有效，则需要继续进一步验证。

2) 如果签章时间不在签章人数字证书有效期内，则签章无效，验证失败，返回失败原因并退出验证流程。

3) 如果签章时间处于签章人数字证书有效期内，但是证书在签章之前已被吊销，则签章视为无效，验证失败，返回失败原因并退m验证流程。

4)如果签章时间处于签章人数字证书有效期内，但是证书在签章之后被吊销，则需要继续进一步验证。

e)验证原文杂凑：

1) 按照propertyInfo信息中的签名保护范围获取待验证原文；

2) 将待验证原文数据进行杂凑运算，形成待验证原文杂凑值；

3) 获取电子签章数据中的原文杂凑值，与待验证原文杂凑值进行二进制比对，如果比对失败，则电子签章验证失败，返回失败原因并退出验证流程。

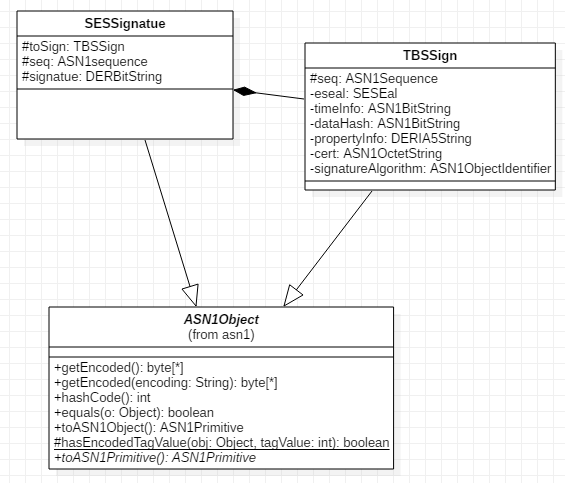
f) 验证电子印章的有效性：

首先，获取电子印章，按照6.1.2验证印章的有效性。再根据电子签章中的时间信息验证签章的有效性。

如果签章时间不处于印章有效期内，则签章无效，验证失败，返回失败原因并退出验证流程。

g) 如果上述各步骤验证均有效，那么电子签章验证结果为有效，可正常退出验证流程。

4、类设计



4.1.4 印章模块

1、模块结构



2、关键数据结构



图1电子印章数据结构图

电子印章数据的ASN.1定义为：

SESeal::=SEQUENCE{

esealInfo SES\_SealInfo，-- 印章信息

signInfo SES\_SignInfo -- 制章人对印章签名的信息

}

SES\_SealInfo::=SEQUENCE{

header SES\_Header，--头信息

esID IA5String，--电子印章标识，电子印章数据的唯一标识编码

property SES\_ESPropertyInfo， --印章属性信息

picture SES\_ESPictrueInfo，--电子印章图片数据

extDatas EXPLICIT ExtensionDatas OPTIONAL --自定义数据

}

其中：

SES\_Header::=SEQUENCE{

ID IA5String．--电子印章数据标识, 固定值“ES”

version INTEGER--电子印章数据版本号标识，如“11”（后续确认）

Vid IA5String--电子印章厂商ID，用于识别不同的软件厂商实现，本系统”yunjingit.com”

}

esID:区分电子印章数据的唯一标识编码，用于查找和索引其他信息。由本系统自行定义，考虑到数量支持和地区支持，设置为；2个字节的制章人编码 | 3个字节的地址码（参考身份证）| 8个字节的序号。

ExtensionDatas::=SEQUENCE SIZE (O..MAX) OF ExtData

ExtData::=SEQUENCE{

extnID OBJECT IDENTIFIER，--自定义扩展字段标识

critical BOOLEAN DEFAULT FALSE，--自定义扩展字段是否关键

extnValue OCTET STRING--自定义扩展字段数据值

}

印章属性信息的ASN.1定义为：

SES\_ESPropertyInfo::=SEQUENCE{

Type INTEGER，--印章类型，如1为单位印章，2为个人印章

Name UTF8String，--印章名称，如个人章、公章、财务章

certList SEQUENCE OF cert，-- 签章人证书列表

createDate UTCTIME，--印章制作日期

validStart UTCTIME，--印章有效起始日期

validEnd UTCTIME--印章有效终止日期

}

印章图片信息的ASN.1定义为：

SES\_ESPictrueInfo::=SEQUENCE{

Type IA5String，--图片类型，如GIF、BMP、JPG

Data OCTET STRING，--图片数据

width INTEGER，--图片显示宽度，单位为毫米(mm)

height INTEGER--图片显示高度，单位为毫米(mm)

}

印章签名信息的ASN.1定义为：

SES\_SignInfo::=SEQUENCE{

cert OCTET STRING．--制章人签名证书

signatureAlgorithm OBJECT IDENTIFIER．--签名算法标识

signData BIT STRING--制章人的签名值

}

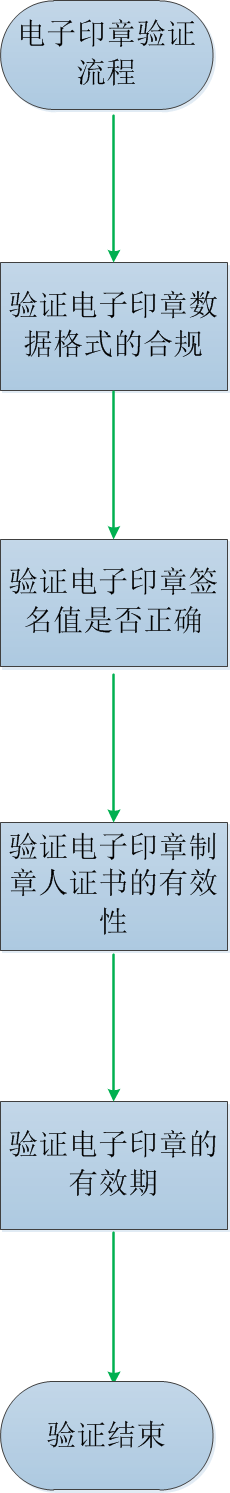
其中：

cert：代表对电子印章数据进行签名的制章人证书；

signatureAlgorithm：代表签名算法OID标识，遵循GM/T 0006。例如，使用SM2签名的OID为1.2.156.10197.1.501;

signData：代表制章人对电子印章格式中印章信息SES\_SealInfo、制章人证书、签名算法标识按SEQUENCE方式组成的信息内容的数字签名。如果签名算法使用SM2，则遵循GM/T 0009；如果签名算法使用RSA，则遵循PKCS#1。

3、关键流程



电子印章验证流程如下：

a) 验证电子印章数据格式的合规性：

按照电子印章格式，解析电子印章，验证是否是符合规范的电子印章格式；

如果电子印章数据格式不合规，则验证失败，返回失败原因并退出验证流程。

b) 验证电子印章签名值是否正确：

根据印章信息数据、制章人证书、签名算法标识验证电子印章签名信息中的签名值是否正确；

如果电子印章签名验证失败，返回失败原因并退出验证流程。

c) 验证电子印章制章人证书的有效性：

验证制章人证书的有效性，验证项至少包括：制章人证书信任链验证、制章人证书有效期验证、

制章人证书是否被吊销、密钥用法是否正确；

知果制章人证书验证失败，返回失败原因并退出验证流程。

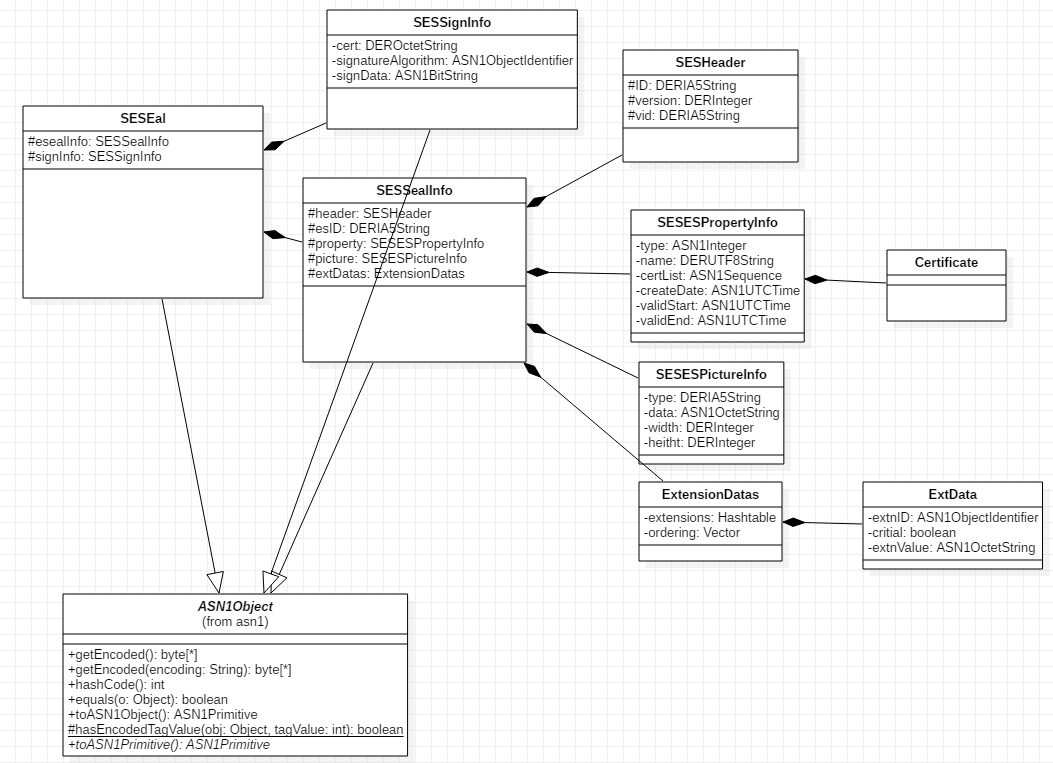
d) 验证电子印章的有效期：

根据印章属性中的印章有效起始日期和有效终止日期，验证电子印章的是否过期；

如果电子印章已过期，则验证失败，返回失败原因并退出验证流程；

e) 如果上述步骤都验证成功，则电子印章验证合规有效，可正常退出验证流程。

4、类设计



类名与印章的ASN.1定义一致。

4.1.5 系统安全模块

1、模块结构



2、关键数据结构

无

3、关键流程

* 印章管理系统启动流程

系统初始化功能主要包括系统配置、生成管理员、审计员等，使设备处于正常工作状态。



* 签章系统启动流程



* 自检

自检包括：算法正确性检测（密码卡）、随机数发生器检测（密码卡）、存储密钥和数据的完整性检测（数据库），以及关键部件的正确性检测等。

数据库设计

* 用户表

|  |  |  |  |
| --- | --- | --- | --- |
| 表名 | User | | |
| 描述 | 用户信息表 | | |
| **字段** | 类型 | 默认值 | 说明 |
| id | INT | 自增1 | 用户ID |
| name | VARCHAR(32) |  | 用户名 |
| password | VARCHAR(32) |  | SHA256(PIN, salt) |
| role | INT |  | 角色类型  支持多个管理员、审计员。 |
| IP | VARCHAR(8) |  | 审计员、管理员这项为空 |
| token | VARCHAR(32) |  | 审计员、管理员这项为空 |
| certificateSN | VARCHAR(20) |  | 审计员、管理员登录USB key对应的证书序列号 |
| info | JSON |  | 应用实体的注册信息（用系统密码卡内置的SM4算法加密） |
| createTime | DATETIME |  |  |
| updateTime | TIMESTAMP |  | 字段为timestamp，这个字段里的时间数据会随其他字段修改的时候自动刷新，所以这个数据类型的字段可以存放这条记录最后被修改的时间 |

* 权限表

|  |  |  |  |
| --- | --- | --- | --- |
| 表名 | Permission | | |
| 描述 | 角色权限表 | | |
| **字段** | 类型 | 默认值 | 说明 |
| id | INT | 自增1 |  |
| role | INT |  | 角色类型（审计员、管理员、签章服务器、应用实体） |
| permission | INT |  | |  |  |  |  | | --- | --- | --- | --- | | 第3字节 | 第2字节 | 第1字节 | 第0字节 | | 1X 系统配置的权限  X1 操作密码卡的权限 | 01 审计日志的权限 | 01 签章服务器权限  10 应用实体权限 | 保留RFU | |
| createTime | DATETIME |  |  |
| updateTime | TIMESTAMP |  |  |

* 证书表

|  |  |  |  |
| --- | --- | --- | --- |
| 表名 | Certificate | | |
| 描述 | 证书信息表 | | |
| **字段** | 类型 | 默认值 | 说明 |
| id | INT | 自增1 |  |
| SN | VARCHAR(20) |  | 证书序列号 |
| keyIndex | INT |  | 密钥索引（密码卡中） |
| keyBackUpIndex | INT |  | 备份的密钥的索引（密码卡中） |
| userID | VARCHAR(32) |  | 证书关联的用户id |
| type | INT |  | 证书类型：制章人、签章人、签章服务器 |
| contents | JSON |  | 证书内容 |
| status | INT |  | CRL或者OCSP查到时填充，默认是正常状态 |
| createTime | DATETIME |  |  |
| revocationDate | DATETIME |  | 被撤销的时间（CRL解析时填充） |

* 印章表

|  |  |  |  |
| --- | --- | --- | --- |
| 表名 | Stamp | | |
| 描述 | 印章信息表 | | |
| **字段** | 类型 | 默认值 | 说明 |
| id | INT | 自增1 |  |
| esID | VARCHAR(100) |  | 印章序列号（印章更新时，使用新的序列号） |
| status | INT |  | 印章的状态：  正常使用07、印章被注销87. |
| content | JSON |  | 印章内容 |
| reqID | INT |  | 对应的印章申请的id |
| createTime | DATETIME |  |  |

* 印章申请表

|  |  |  |  |
| --- | --- | --- | --- |
| 表名 | Stamp | | |
| 描述 | 印章信息表 | | |
| **字段** | 类型 | 默认值 | 说明 |
| id | INT | 自增1 | 申请序列号 |
| status | INT |  | 印章申请的状态：  待审核01、审核通过03、审核未通过80 |
| requestContent | JSON |  | 印章申请时的填充内容，用于在印章更新时生成默认配置。 |
| createTime | DATETIME |  |  |

* 配置表

|  |  |  |  |
| --- | --- | --- | --- |
| 表名 | Config | | |
| 描述 | 配置信息表 | | |
| **字段** | 类型 | 默认值 | 说明 |
| id | INT | 自增1 |  |
| name | VARCHAR(32) |  | 自定义 |
| type | INT |  | CA申请、CRL地址、OCSP服务、签章服务器配置 |
| content | JSON |  | 配置项的内容 |
| status | INT |  | 状态：停用/使用 |
| createTime | DATETIME |  |  |
| updateTime | TIMESTAMP |  |  |

* 摘要表（完整性需求）

|  |  |  |  |
| --- | --- | --- | --- |
| 表名 | Hash | | |
| 描述 | 摘要信息表 | | |
| **字段** | 类型 | 默认值 | 说明 |
| id | INT | 自增1 |  |
| name | VARCHAR(32) |  | 表名 |
| hash | VARCHAR(32) |  | 表内容计算出的hash值（SHA256）。  摘要的计算：相关表每次修改确认后，均立即计算HASH。 |
| createTime | DATETIME |  |  |
| updateTime | TIMESTAMP |  |  |

* 日志表（审计需求）

|  |  |  |  |
| --- | --- | --- | --- |
| 表名 | LogYYMMDD (年月日，也就是每天创建一个新表) | | |
| 描述 | 摘要信息表 | | |
| **字段** | 类型 | 默认值 | 说明 |
| id | INT | 自增1 |  |
| type | INT |  | 日志类型，例如警告 |
| content | JSON |  | 日志内容 |
| createTime | DATETIME |  |  |

上述所有的表均需要备份，除了日志表的备份由审计员触发，其余均由管理员触发。

安全性设计

参考规范GM/T 0054 信息系统密码应用基本要求。

物理和环境安全

本文档不涉及服务器的工作场所的安全性设计。

网络和通信安全

电子签章系统和电子印章系统均配置了RSA证书，采用HTTPS作为网络通信协议，从而保证通信过程中数据的完整性和机密性，并利用HTTPS的双向认证机制对通信双方进行身份认证。通过IP白名单机制对连接到印章系统的签章设备进行限制。

电子签章系统和应用系统采用HTTPS作为网络通信协议，从而保证通信过程中数据的完整性和机密性，通过IP白名单机制对连接到系统的应用进行限制。

电子签章系统和电子印章系统与外部Internet通过防火墙连接，提高网络安全性。

设备和计算安全

管理员、审计员登录需要验证PIN码，且2个月更换。只能通过本地管理，无远程管理功能。

系统重要的数据、资源、日志均保存在数据库中的表中，并备份。通过摘要表，记录这些重要数据的数据指纹，保证数据的完整性。

所有密码算法均由符合国密标准GM/T 0028密码卡完成，流程满足国密标准。

应用和数据安全

IP白名单机制和证书机制对访问的用户身份进行标识和鉴别。

采用HTTPS作为网络通信协议，防止身份鉴别信息，并保证数据传输的机密性和完整性。

只保存PIN码的HASH值，保证存储的机密性。

密钥管理

采用符合国密标准GM/T 0028的密码卡，由密码卡负责密钥的生成、存储、备份、恢复，私钥不出卡，并提供密钥销毁功能。