



# 中华人民共和国劳动和劳动安全行业标准

LD/T 30.4—2009

## 人力资源和社会保障电子认证体系 第4部分:证书应用管理规范

Human resources and social security electronic authentication system—  
Part 4: Management specification of digital certificate application

2009-12-14 发布

2010-03-01 实施

中华人民共和国人力资源和社会保障部 发布

目 次

前言 ..... I

1 范围 ..... 1

2 规范性引用文件 ..... 1

3 术语和定义 ..... 1

4 缩略语 ..... 2

5 证书应用领域 ..... 2

6 证书应用技术体系 ..... 3

    6.1 总体技术框架 ..... 3

    6.2 密码设备 ..... 3

    6.3 基础应用接口 ..... 4

    6.4 高级应用接口 ..... 4

    6.5 证书应用接口技术要求 ..... 14

7 典型证书应用流程 ..... 15

    7.1 数字证书登录认证 ..... 15

    7.2 单向数字签名与验签 ..... 16

    7.3 双向数字签名与验签 ..... 17

    7.4 加密与解密 ..... 18

    7.5 加密签名与解密验签 ..... 18

附录 A (资料性附录) 证书应用场景 ..... 20

附录 B (规范性附录) 高级证书应用接口相关标识 ..... 25

## 前 言

为适应人力资源和社会保障信息化发展要求,满足人力资源和社会保障网络信任体系建设和管理的需要,人力资源和社会保障部组织并制定了 LD/T 30—2009《人力资源和社会保障电子认证体系》。

网络信任体系包括电子认证体系、授权管理体系和责任认定体系,本标准主要描述了人力资源和社会保障电子认证体系相关内容,包括以下五个部分:

- 第 1 部分:框架规范;
- 第 2 部分:电子认证系统技术规范;
- 第 3 部分:证书及证书撤销列表格式规范;
- 第 4 部分:证书应用管理规范;
- 第 5 部分:证书载体规范。

本部分为 LD/T 30—2009 的第 4 部分。

本部分描述了证书应用领域和证书应用技术体系,规范了证书应用接口规范和证书应用接口技术要求,给出了典型证书应用流程和应用场景。

本部分重点引用了国家密码局《公钥密码基础设施应用技术体系》相关规范,并在此基础上,扩展了典型证书应用流程等相关内容,给出了几类常见的人力资源社会保障业务系统的证书应用场景,从满足人力资源社会保障业务需求的角度,对本行业应用系统使用数字证书的接口和流程提出规范和要求。

本部分由中华人民共和国人力资源和社会保障部信息中心提出并归口。

本部分主要起草单位:中华人民共和国人力资源和社会保障部信息中心、上海市人力资源和社会保障局信息中心、北京数字证书认证中心、维豪信息技术有限公司。

本部分主要起草人:赵锡铭、戴瑞敏、贾怀斌、翟燕立、李丽虹、吴问滨、黄勇、吕丽娟、许华光、罗震、张加会、靳朝晖、陆春生、李永亮、宋京燕、杜守国、欧阳晋、林雪焰、李述胜、顾青、宋成。

本部分凡涉及密码相关内容,均按国家有关法规实施。

# 人力资源和社会保障电子认证体系

## 第4部分：证书应用管理规范

### 1 范围

LD/T 30 的本部分规定了数字证书的应用领域,制定了人力资源和社会保障数字证书应用的总体技术框架,规范了人力资源和社会保障应用系统在实现身份认证、数字签名及验签、加密与解密等安全功能时,所采取的应用接口和证书应用处理流程,给出了可供参考的典型应用场景。

本部分适用于指导人力资源和社会保障应用系统实现基于数字证书的安全功能,有助于各级人力资源和社会保障部门采用统一的基于数字证书应用的开发接口。

### 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(包括所有的修改单)适用于本文件。

GB/T 20518—2006 信息安全技术 公钥基础设施 数字证书格式

LD/T 30.5 人力资源和社会保障电子认证体系 第5部分:证书载体规范

公钥密码基础设施应用技术体系 密码设备应用接口规范(国家密码管理局)

公钥密码基础设施应用技术体系 通用密码服务接口规范(国家密码管理局)

公钥密码基础设施应用技术体系 密码设备管理规范(国家密码管理局)

智能 IC 卡及智能密码钥匙密码应用接口规范(国家密码管理局)

信息技术 安全技术 密码术语(国家密码管理局)

### 3 术语和定义

以下术语和定义适用于本部分。

#### 3.1

**证书认证机构** Certification Authority

CA

负责创建和分配证书,受用户信任的权威机构。用户可以选择该机构为其创建密钥。

#### 3.2

**数字证书** digital certificate

由权威认证机构进行数字签名的包含公开密钥拥有者信息、公开密钥、签发者信息、有效期以及一些扩展信息的数字文件。

#### 3.3

**非对称密码算法** asymmetric cryptographic algorithm

使用两种相关变换和非对称密钥对的密码技术,一种是由公开密钥定义的公开变换,另一种是由私有密钥定义的私有变换。两种变换具有以下特性:即使给定公开变换,也不可能通过计算得出私有变换。



## LD/T 30.4—2009

### 3.4

**私有密钥 private key**

**私钥**

在公钥密码体制中,用户密钥对中仅为该用户持有的密钥。

### 3.5

**公开密钥 public key**

**公钥**

在公钥密码体制中,用户密钥对中公布给其他用户的密钥。

### 3.6

**加密 encrypt**

通过密码算法对数据进行变换来产生密文,以便隐藏数据的信息内容。

### 3.7

**解密 decrypt**

与一个可逆的加密过程相对应的反过程。该过程使用适当的密钥,将已加密的文本转换成明文。

### 3.8

**容器 container**

密码设备中用于保存密钥所划分的存储空间的唯一性编号。

### 3.9

**信任 trust**

通常,当一个实体(第一个实体)假设另一个实体(第二个实体)完全按照第一个实体的期望行动时,则称第一个实体“信任”第二个实体。这种“信任”可能只适用于某些特定功能。本框架中“信任”的关键作用是描述鉴别实体和认证机构之间的关系;鉴别实体应确信它能够“信任”认证机构仅创建有效且可靠的证书。

## 4 缩略语

下列缩略语适用于本部分:

API 应用程序接口,简称应用接口(Application Program Interface)

CA 证书认证机构(Certification Authority)

CRL 证书撤销列表(Certificate Revocation List)

CSP 加密服务提供者(Cryptographic Service Provider)

KMC 密钥管理中心(Key Management Center)

LDAP 轻量级目录访问协议(Lightweight Directory Access Protocol)

OID 对象标识符(Object Identifier)

PKCS 公钥密码标准(the Public-Key Cryptography Standard)

RA 证书注册机构(Registration Authority)

## 5 证书应用领域

人力资源和社会保障数字证书的应用领域包括全国性、区域性的各类应用系统,按照业务类别划分为以下三类:

- a) 内部办公类:部、省、市、县/区各级人力资源和社会保障部门各自内部办公系统、档案系统、统计系统、决策系统等。

- b) 人力资源业务类:包括就业管理与服务、失业管理、公务员管理、军转干部管理、专业技术人才管理、人力资源市场、职业资格管理、职业培训、劳动监察等的本地业务管理与服务系统、跨地区业务管理与服务系统、网上公共服务系统、相关纵向联网数据采集系统等。
- c) 社会保险业务类:包括养老、失业、医疗、工伤、生育保险的本地业务管理与服务系统、跨地区业务管理与服务系统、网上公共服务系统、相关纵向联网数据采集系统、基金监管系统等。

## 6 证书应用技术体系

### 6.1 总体技术框架

人力资源和社会保障证书应用技术框架由密码设备、基础应用接口、高级应用接口组成,如图 1 所示。

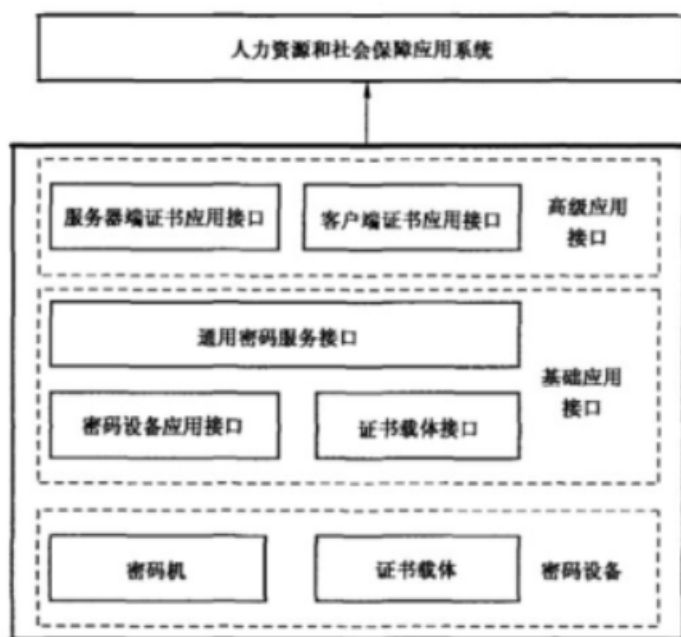


图 1 人力资源和社会保障证书应用总体技术框架

### 6.2 密码设备

密码设备由密码机、证书载体等密码设备组成,通过基础应用接口向上层应用提供基础的、全面的密码服务。

密码机为应用系统的服务器端提供密码服务。

证书载体为应用系统的客户端提供密码服务。

#### 6.2.1 密码机

密码机必须采用国家密码主管部门批准使用的密码设备,具有销售许可资质。

#### 6.2.2 证书载体

证书载体主要指用于存储密钥和数字证书并具有密码运算功能的硬件载体,必须采用国家密码主管部门批准使用的密码设备,具有销售许可资质。

证书载体的各项技术参数应符合 LD/T 30.5。

## LD/T 30.4—2009

### 6.3 基础应用接口

基础应用接口主要包括三类：密码设备应用接口、证书载体接口和通用密码服务接口。

#### 6.3.1 密码设备应用接口

密码设备应用接口应遵循国际标准 PKCS#11 规范和《公钥密码基础设施应用技术体系 密码设备应用接口规范》。

密码设备应用接口应支持 Windows、Linux、Unix 等所有主流操作系统。

#### 6.3.2 证书载体接口

证书载体接口应遵循《智能 IC 卡及智能密码钥匙密码应用接口规范》和 LD/T 30.5。

证书载体接口应支持 Windows 和 Linux 等主流操作系统。

#### 6.3.3 通用密码服务接口

通用密码服务接口位于密码设备应用接口和证书载体接口之上，应遵循《公钥密码基础设施应用技术体系 通用密码服务接口规范》。

### 6.4 高级应用接口

高级应用接口是位于基础应用接口之上，供应用系统直接调用的接口开发包，包括客户端和服务端接口两大部分。

根据应用系统架构和应用环境，应至少支持 C 和 Java 两种开发语言。对于 B/S 和 C/S 架构的应用系统，客户端应提供 ActiveX 控件、DLL 动态连接库或 JAR 开发包，服务器端应提供 JAR 格式的 JAVA 包、DLL 格式的 COM 组件以及 so 文件格式的开发包。

高级应用接口的主要功能包括：身份认证、加密与解密、数字签名与验证等。

#### 6.4.1 客户端应用接口

客户端应用接口的主要功能函数如下：

##### a) 控件初始化 HRSS\_Ocx\_AtivInit()

原型：int HRSS\_Ocx\_AtivInit();

描述：控件初始化，读取客户端证书载体信息及其他初始化工作。

调用一些主要功能接口前必须先调用本接口。

参数：无

返回值：成功

0

失败

错误号，错误号小于 0

##### b) 清除内部存储区 HRSS\_Ocx\_AtivEnd()

原型：int HRSS\_Ocx\_AtivEnd();

描述：进行开发包函数库调用完毕后进行内部存储区的清除。

参数：无

返回值：成功

0

失败

错误号，错误号小于 0

##### c) 获取证书列表 HRSS\_Ocx\_GetUserList()

原型：BSTR HRSS\_Ocx\_GetUserList();

描述：取得当前已安装证书的用户列表。

参数： 无

返回值： BSTR ret 用户列表字符串。

备注： 返回字符串格式：(用户名 1 || 证书 1 所对应的容器名 &&& 用户名 2 || 证书 2 所对应的容器名 &&&...)。

d) 登录 HRSS\_Ocx\_Login ()

原型： BOOL HRSS\_Ocx\_Login(int loginType,BSTR i\_label,BSTR loginPin);

描述： 登录客户端硬件设备。

参数： loginType[IN]:用户登录类型

i\_label [IN]: USBKey 的容器名

loginPin[IN]:用户登录 PIN 码

返回值： 成功 TRUE

失败 FALSE

备注： 调用本接口时,如果 loginPin 输入为“ ”空,则会弹出输入口令对话框;如果不为空,则不会弹出,只返回登录是否成功。

e) 退出客户端硬件登录 HRSS\_Ocx\_Logout ()

原型： BOOL HRSS\_Ocx\_Logout ();

描述： 退出客户端硬件登录。

参数： 无

返回值： 成功 TRUE

失败 FALSE

f) 获取证书信息 HRSS\_Ocx\_GetUserInfo

原型： BSTR HRSS\_Ocx\_GetUserInfo(BSTR Cert,short Type);

描述： 获取证书信息。

参数： BSTR Cert;Base64 编码的证书

int Type;获取信息的类型

Type 不同取值的含义如表 1 所示。

表 1 证书信息类型 TYPE 定义表

| TYPE | 含 义            |
|------|----------------|
| 1    | 证书版本           |
| 2    | 证书序列号          |
| 3    | 证书签名算法标识       |
| 4    | 证书发放者国家名       |
| 5    | 证书发放者组织名       |
| 6    | 证书发放者部门名       |
| 7    | 证书发放者省份名       |
| 8    | 证书发放者通用名       |
| 9    | 证书发放者城市名       |
| 10   | 证书发放者 Email 地址 |
| 11   | 证书有效期起始        |
| 12   | 证书有效期截止        |



描述： 对输入数据进行数字签名的验证。

参数： i\_checkCert [IN]：验证所用的证书。

i\_clearText [IN]：签名的原文。如签名值包含原文，置空即可。

i\_signature [IN]：待验证的签名数据。

i\_algoType[IN]：签名算法。如签名值不符合 PKCS#7 标准，需要输入签名算法。

签名算法取值：32772——sha1RSA

i\_signType[IN]：签名值类型

取值：0——不包含原文(纯签名值)

1——包含原文的符合 PKCS#7 格式的数据

返回值： TRUE 成功

FALSE 失败

备注： 当验证纯签名值时，需要输入签名算法或者通过策略解析获取匹配算法。

#### k) 编码 HRSS\_Ocx\_Base64Encode

原型： BSTR HRSS\_Ocx\_Base64Encode(BSTR inData)

描述： 对输入数据进行编码。

参数： inData [IN]：要编码的数据

返回值： 返回编码值

#### l) 解码 HRSS\_Ocx\_Base64Decode

原型： BSTR HRSS\_Ocx\_Base64Decode(BSTR inData)；

描述： 对输入数据进行解码。

参数： inData [IN]：要解码的数据

返回值： 返回解码后的原文

#### m) 产生随机数 HRSS\_Ocx\_GenRandom

原型： BSTR HRSS\_Ocx\_GenRandom(int len)；

描述： 产生随机数。

生成一定长度的随机数/对称密钥。

参数： len[IN]：指定的长度(字节数)

返回值： 已编码的随机数/对称密钥。

#### n) 加密数据 HRSS\_Ocx\_SymmEncrypt

原型： BSTR HRSS\_Ocx\_SymmEncrypt(long i\_symmAlgo, BSTR i\_symmkey, BSTR i\_indata)；

描述： 使用对称算法加密数据。

用指定产生的对称密钥 KEY 来加密数据。

参数： i\_symmAlgo[IN]：对称算法标识。

例如：CALG\_RC4 = 26625

CALG\_3DES = 26115

CALG\_33 = 9

CALG\_PMC3DES5 = 15

i\_symmKey[IN]： 对称密钥，已编码。

i\_inData[IN]： 输入的原文，二进制数据流编码。

返回值： 已编码后的密文。

#### o) 解密数据 HRSS\_Ocx\_SymmDecrypt

原型: BSTR HRSS\_Ocx\_SymmDecrypt(long i\_symmAlgo, BSTR i\_symmkey, BSTR i\_indata);

描述: 用指定的对称密钥 KEY 来解密密文数据。

参数: i\_symmAlgo[IN]: 对称算法标识。

例如: CALG\_RC4 = 26625  
CALG\_3DES = 26115  
CALG\_33 = 9  
CALG\_PMC3DES5 = 15

i\_symmKey[IN]: 对称密钥, 已编码。

i\_inData[IN]: 输入的密文, 已编码。

返回值: 解密后的原文, 如是二进制数据则为已编码。

p) 数字信封加密 HRSS\_Ocx\_SealEnvelope

原型: BSTR HRSS\_Ocx\_SealEnvelope(BSTR i\_encCert, long i\_symmAlgo, BSTR i\_inData);

描述: 数字信封加密。

内部的处理过程是首先随机生成一对称密钥(由入口参数指定其算法), 然后用此对称密钥加密输入的数据, 最后用公钥加密产生的此对称密钥。此函数的加密输出结果遵循 PKCS#7 的编码标准。

参数: i\_encCert[IN]: 用于加密的数字证书, 已编过码。

i\_symmAlgo[IN]: 信封中所用对称算法标识。取值如下:

例如: CALG\_RC4 = 26625  
CALG\_3DES = 26115  
CALG\_33 = 9  
CALG\_PMC3DES5 = 15

i\_inData[IN]: 输入原文信息, 如是二进制数据则已编过码。

返回值: 已编码的密文结果。

q) 解析数字信封 HRSS\_Ocx\_OpenEnvelope

原型: BSTR HRSS\_Ocx\_OpenEnvelope(BSTR i\_label, BSTR i\_inData);

描述: 解析数字信封。

私钥标签在配置文件中定义, 运行时弹出输入框提示用户输入私钥保护口令。

参数: i\_label[IN]: 输入容器名;

i\_inData[IN]: 输入已编码的信封数据。

返回值: 原文信息, 如是二进制数据则为已编码。

r) 哈希 HRSS\_Ocx\_HashData

原型: BSTR HRSS\_Ocx\_HashData(int hashAlgo, BSTR inData);

描述: 对输入数据进行哈希运算。

参数: hashAlgo[IN]: 哈希算法标识

例如: ALGO\_SHA1 32772  
ALGO\_PMCHASH 32773

inData[IN]: 输入数据信息, 如是二进制数据则已编码。

返回值: 编码后的哈希值。

s) 用多证书批量生成数字信封 HRSS\_Ocx\_SealEnvelopeEx

原型: BSTR HRSS\_Ocx\_SealEnvelopeEx (BSTR i\_encCert, long i\_certsizes, long i\_symmAlgo, BSTR i\_inData);

描述: 用多证书批量生成数字信封。

此函数的加密输出结果遵循 PKCS#7 的编码标准。

参数: i\_encCert[IN]: 用于加密的数字证书,已编过码,多个证书间以“|”隔开。

i\_certsizes[IN]: 保留参数

i\_symmAlgo[IN]: 信封中所用对称算法标识。

例如: CALG\_RC4 = 26625

CALG\_3DES = 26115

CALG\_33 = 9

CALG\_PMC3DES5 = 15

i\_inData[IN]: 输入原文信息,如是二进制数据则为已编码。

返回值: 输出的已编码的密文结果。

#### t) 从本地读取文件内容 HRSS\_Ocx\_ReadFromFile

原型: BSTR HRSS\_Ocx\_ReadFromFile(BSTR fileName);

描述: 从本地读取文件内容。

参数: fileName[IN]: 要读取的文件全路径名。如果为“ ”空,则弹出文件选择对话框。

返回值: 输出已编码过的文件数据信息。

#### u) 写本地文件内容 HRSS\_Ocx\_WriteToFile

原型: BOOL HRSS\_Ocx\_WriteToFile(BSTR fileName, BSTR writeData);

描述: 将数据写入到本地文件。

参数: fileName[IN]: 要保存的文件名称

writeData[IN]: 要写入文件的数据,已编码。

返回值: TRUE 成功

FALSE 失败

### 6.4.2 服务器端应用接口

服务器端应用接口根据服务器应用环境,包括两种不同形态的接口。

COM 组件接口:应用服务器为 ASP 或 ASP.NET。

Java 组件接口:应用服务器为 JAVA 环境。

#### 6.4.2.1 COM 组件接口

COM 组件接口包括以下接口函数:

##### a) 生成随机数 HRSS\_Svr\_pkiGenerateRandom

原型: int HRSS\_Svr\_pkiGenerateRandom(int i\_length, PKI\_PDATA o\_outData);

描述: 生成一定长度的随机数/对称密钥。

参数: i\_length[IN]: 指定的长度(字节数)。

o\_outData[OUT]: 生成的随机数(可作为对称密钥使用)。

返回值: 成功 0

失败 错误代码

##### b) 构造数字信封 HRSS\_Svr\_pkiSealEnvelope



原型: int HRSS\_Svr\_pkiSealEnvelope(PKI\_DATA i\_encCert,UINT i\_symmAlgo,PKI\_DATA i\_inData,PKI\_PDATA o\_outData);

描述: 构造数字信封。

参数: i\_encCert[IN]:用于加密的数字证书。

i\_symmAlgo[IN]:信封中所用对称算法标识。

例如:ALGO\_3DES = 26115

ALGO\_RC4 = 26625

ALGO\_SSF33 = 9

i\_inData[IN]:输入原文。

o\_outData[OUT]:输出的密文结果。

返回值: 成功 0

失败 错误代码

c) 解析数字信封 HRSS\_Svr\_pkiOpenEnvelope

原型: int HRSS\_Svr\_pkiOpenEnvelope (BYTE \* i\_decKeyLabel, BYTE \* i\_passwd,PKI\_DATA i\_inData,PKI\_PDATA o\_outData);

描述: 解析数字信封。

参数: i\_decKeyLabel[IN]:解密私钥的标签。

i\_passwd[IN]:解密私钥的保护口令。

i\_inData[IN]:输入的信封数据。

o\_outData[OUT]:输出的原文。

返回值: 成功 0

失败 错误代码

d) 数字签名 HRSS\_Svr\_pkiSignData

原型: int HRSS\_Svr\_pkiSignData(BYTE \* i\_keyLabel,BYTE \* i\_keyPasswd,ALG\_ID digestAlgo,PKI\_DATA i\_inData,PKI\_PDATA o\_outData);

描述: 对输入数据进行数字签名,签名值包含原文。

参数: i\_keyLabel[IN]:签名私钥的标签。

i\_keyPasswd[IN]:签名私钥的保护口令。

digestAlgo [IN]:摘要算法。

CALG\_SHA1 32772

i\_inData[IN]:输入的待签名数据

o\_outData[OUT]:输出的签名值(符合 PKCS#7 标准,签名值包含原文)。

返回值: 成功 0

失败 错误代码

e) 数字签名的验证 HRSS\_Svr\_pkiVerifyData

原型: int HRSS\_Svr\_pkiVerifyData(PKI\_DATA i\_checkCert,PKI\_DATA i\_signature);

描述: 对输入数据进行数字签名的验证,签名值包含原文,符合 PKCS#7 规范,与 pkiSignData 对应。

参数: i\_checkCert [IN]:验证所用的证书。

i\_signature [IN]:待验证的签名值,签名值符合 PKCS#7 标准。

返回值: 成功 0

失败 错误代码

## f) 不包含原文的数字签名 HRSS\_Svr\_pkiGetSignature

原型: int HRSS\_Svr\_pkiGetSignature(BYTE \* i\_keyLabel, BYTE \* i\_keyPasswd, ALG\_ID digestAlgo, PKI\_DATA i\_inData, PKI\_PDATA o\_outData);

描述: 对输入数据进行数字签名, 签名值不包含原文。

参数: i\_keyLabel[IN]: 签名私钥的标签。

i\_keyPasswd[IN]: 签名私钥的保护口令。

digestAlgo [IN]: 摘要算法。CALG\_SHA1 32772

i\_inData[IN]: 输入的待签名数据。

o\_outData[OUT]: 输出的签名值, 签名值不包含原文, 纯签名。

返回值: 成功 0

失败 错误代码

## g) 验证不包含原文的数字签名 HRSS\_Svr\_pkiVerifySignature

原型: int HRSS\_Svr\_pkiVerifySignature (ALG\_ID digestAlgo, PKI\_DATA i\_checkCert, PKI\_DATA i\_clearText, PKI\_DATA i\_signature);

描述: 对输入数据进行数字签名的验证, 签名值不包含原文, 纯签名, 与 pkiGetSignature 对应。

参数: digestAlgo [IN]: 签名算法, 取值如: CALG\_SHA1

i\_checkCert [IN]: 验证所用的证书。

i\_clearText [IN]: 签名前的原文。

i\_signature [IN]: 待验证的签名值, 签名值不包含原文。

返回值: 成功 0

失败 错误代码

## h) 解析 CRL 信息 HRSS\_Svr\_pkiGetCRLInfo

原型: int HRSS\_Svr\_pkiGetCRLInfo (PKI\_DATA i\_inCRL, PKI\_DATA \* o\_crlInfo);

描述: 获取 CRL 的详细信息。

参数: i\_inCRL[IN]: 输入的 CRL。

o\_crlInfo[OUT]: 获得的 CRL 信息。

返回值: 成功 0

失败 错误代码

## i) 摘要运算 HRSS\_Svr\_pkiHashData

原型: int HRSS\_Svr\_pkiHashData(UINT i\_hashAlgo, PKI\_DATA i\_inData, PKI\_PDATA o\_outData);

描述: 对输入数据进行摘要运算。

参数: i\_hashAlgo[IN]: 摘要算法标识。ALGO\_SHA1 32772

i\_inData[IN]: 输入数据。

o\_outData[OUT]: 输出数据。

返回值: 成功 0

失败 错误代码

## j) 根据 OID 解析证书信息 HRSS\_Svr\_getBasicCertInfoByOid

原型: int HRSS\_Svr\_pkiGetUserInfoByOid (PKI\_DATA i\_inCert, PKI\_DATA i\_Oid, PKI\_DATA \* cert\_info);

描述: 根据 OID 获取证书私有扩展项信息。

参数: i\_inCert[IN]:待解析的证书,已编码。  
 i\_Type[IN]:私有扩展对象 ID,例如“1.6.16.21.88.1”。  
 cert\_info[OUT]:证书 OID 对应的值。

返回值: 成功 0  
 失败 错误代码

k) 解析证书基本信息 HRSS\_Svr\_pkiGetCertInfoEx

原型: int HRSS\_Svr\_pkiGetCertInfoEx(PKI\_DATA i\_inCert,int i\_Type,PKI\_DATA \* cert\_info);

描述: 获取证书信息。

参数: i\_inCert[IN]:待解析的证书,已编码;  
 i\_Type[IN]:获取信息的类型;  
 Type 不同的取值含义如表 1 所示。  
 cert\_info[OUT]:获得的证书信息。

返回值: 成功 0  
 失败 错误代码

l) BASE64 编码 HRSS\_Svr\_pkiBase64Encode

原型: int HRSS\_Svr\_pkiBase64Encode(PKI\_DATA i\_inData,PKI\_PDATA o\_outData);

描述: 对二进制数据进行 BASE64 编码。

参数: i\_inData[in]:要编码的二进制数据;  
 o\_outData[out]:编码后的 BASE64 数据。

返回值: 成功 0  
 失败 错误代码

m) BASE64 解码 HRSS\_Svr\_pkiBase64Decode

原型: int HRSS\_Svr\_pkiBase64Decode(PKI\_DATA i\_inData,PKI\_PDATA o\_outData);

描述: 对 BASE64 数据解码,生成解码后的二进制数据。

参数: i\_inData[in]:要解码的 BASE64 数据;  
 o\_outData[out]:解码后的二进制数据。

返回值: 成功 0  
 失败 错误代码

## 6.4.2.2 Java 组件接口

Java 组件接口包括以下接口函数:

a) 初始化对象 HRSS\_Svr\_release

原型: boolean HRSS\_Svr\_release();

描述: 获得一个对象实例,初始化对象。

函数库调用完毕后,进行内部存储区的清除工作。

参数: 无

返回值: 成功 TRUE  
 失败 FALSE

b) 生成随机数 HRSS\_Svr\_AdvGenRandom

原型: String HRSS\_Svr\_AdvGenRandom(int len);

描述: 输出已编码的随机数/对称密钥。

参数: len[IN]: 指定的长度(字节数)。

返回值: 随机数 (BASE64 编码, 随机数的长度 len)

c) 验证证书有效性 HRSS\_Svr\_AdvCheckCert

原型: boolean HRSS\_Svr\_AdvCheckCert(String i\_inCert, int i\_Type);

描述: 验证证书的有效性。

参数: i\_inCert[IN]: 待验证的证书, 已编码;

i\_Type[IN]: 验证方式: 2——CRL

返回值: 成功 TRUE

失败 FALSE

d) 验证签名 HRSS\_Svr\_AdvVerifySign

原型: boolean HRSS\_Svr\_AdvVerifySign(String i\_checkCert, byte[] i\_clearText, String i\_signature, int i\_algoType, int i\_signType);

描述: 对输入数据进行数字签名的验证。

参数: i\_checkCert [IN]: 验证所用的证书。

i\_clearText [IN]: 签名的原文。如签名值包含原文, 请置为空即可。

i\_signature [IN]: 待验证的签名数据。

i\_algoType[IN]: 签名算法。如签名值不符合 PKCS#7 标准, 需要输入签名算法。取值如: 32772——sha1RSA

i\_signType[IN]: 签名值类型

取值: 0——不包含原文的纯签名

1——包含原文的符合 PKCS#7 格式的数据

返回值: 成功 TRUE

失败 FALSE

备注: 当验证纯签名值时, 需要输入签名算法或者通过策略解析器获取匹配算法; 当验证 PKCS#7 格式的签名内容时, 由于本身自带了签名算法, 因此没有必要再次调用相应的策略解析器。

e) 数字信封加密 HRSS\_Svr\_AdvSealEnvelope

原型: String HRSS\_Svr\_AdvSealEnvelope (String i\_encCert, int i\_symmAlgo, byte[] i\_inData);

描述: 数字信封加密。

内部的处理过程是首先随机生成一对称密钥(由入口参数指定其算法或者通过 PA 确定), 然后用此对称密钥加密输入的数据, 最后用公钥加密产生的此对称密钥。此函数的加密输出结果遵循 PKCS#7 的编码标准(Enveloped-Data)。

参数: i\_encCert[IN]: 用于加密的数字证书, 已编过码。

i\_symmAlgo[IN]: 信封中所用对称算法标识。

i\_inData[IN]: 输入原文信息。

返回值: 已编码的密文信息。

f) 数字信封解密 HRSS\_Svr\_AdvOpenEnvelope

原型: byte[] HRSS\_Svr\_AdvOpenEnvelope (String lable, String keyPasswd,

String i\_inData);

描述: 对数字信封进行解密工作,即拆封工作。

参数: lable [IN]: 容器名,如: LAB\_USERCERT;

keyPasswd[IN]: 解密私钥的保护口令;

i\_inData[IN]: 输入已编码的信封数据。

返回值: 输出原文信息。

#### g) 数字签名 HRSS\_Svr\_AdvSignData

原型: String HRSS \_ Svr \_ AdvSignData (String lable, String keyPasswd, byte[] i\_inData, int i\_algoType, int i\_signType);

描述: 对输入数据进行数字签名。

参数: lable [IN]: 容器名 如: LAB\_USERCERT。

keyPasswd[IN]: 签名私钥的保护口令;

i\_inData[IN]: 输入的待签名数据;

i\_algoType[IN]: 签名算法,取值如: 32772——sha1RSA

sha1RSA 说明签名采用的摘要算法和签名算法

i\_signType[IN]: 签名值类型

取值: 0——不包含原文的纯签名

1——包含原文的符合 PKCS#7 格式的数据

返回值: 已编码的签名数据。

#### h) 取得证书基本信息 HRSS\_Svr\_AdvGetCertInfo

原型: String HRSS\_Svr\_AdvGetCertInfo(String i\_inCert, int i\_Type);

描述: 获取证书信息

参数: i\_inCert [IN]: 待解析的证书,已编码;

i\_Type [IN]: 获取信息的类型。

Type 不同的取值含义如表 1 所示。

返回值: 获取的证书信息。

## 6.5 证书应用接口技术要求

### 6.5.1 密码算法

密码设备和密码服务接口应提供符合国家密码主管部门规定的密码算法和接口规范,并根据国家密码主管部门批准的算法及时调整,以适应国家最新技术标准要求。

a) 对称密码算法: SSF-33、SM1 等。

b) 公钥密码算法: RSA、SM2 等。

c) HASH 算法: SHA-1、SHA256、SM3 等。

### 6.5.2 证书应用接口运行环境

密码服务接口应支持在以下环境中可靠运行:

a) 客户端支持 Windows2000 及以上的所有 Windows 系列版本。

b) 服务器端支持 Windows2000 及以上 Windows 系列版本、Linux、Unix 等所有主流操作系统。

c) 服务器端支持硬件密码机或加密卡,客户端证书载体支持 USBKey、智能 IC 卡等设备。

d) 应用系统支持 C/S 和 B/S 的系统结构。

## 7 典型证书应用流程

人力资源和社会保障证书应用中典型证书应用流程包括以下几类：

- a) 数字证书登录认证；
- b) 单向数字签名与验签；
- c) 双向数字签名与验签；
- d) 加密与解密；
- e) 加密签名与解密验签。

### 7.1 数字证书登录认证

基于数字证书的登录认证是指用户通过数字证书方式安全登录人力资源和社会保障业务系统的过程。

在应用过程中,用户应在业务系统中进行注册,同时需要将业务系统中的注册用户与用户数字证书进行关联。用户使用数字证书登录前,应安装数字证书客户端软件,准备好证书运行环境。

数字证书登录认证流程如图 2 所示。



图 2 数字证书登录认证流程图

数字证书登录认证流程：

- a) 用户将证书载体(USBKey)插入计算机,浏览证书登录首页。
- b) 服务器端处理程序产生一个随机数,通过密码设备签名,并将设备证书、随机数及其电子签名值下载到客户端。登录页面获取本机注册的数字证书,展现给用户一个证书登录框。
- c) 用户选择数字证书,输入证书保护密码(即 USBKey 的用户口令),点击“登录”按钮。此时,客户端程序(如 JavaScript)应调用证书高级应用接口(ActiveX 控件)对证书密码进行校验。以 ActiveX 控件为例,分别调用 HRSS\_Ocx\_AtvInit()和 HRSS\_Ocx\_Login()函数。
- d) 校验密码通过后,调用证书高级应用接口(ActiveX 控件)对服务器端产生的数字签名进行验证。以 ActiveX 控件为例,调用 HRSS\_Ocx\_VerifySign()函数。
- e) 验证签名通过后,使用证书载体(USBKey)对随机数进行签名。以 ActiveX 控件为例,调用 HRSS\_Ocx\_SignData()函数。

- f) 签名成功后,按照约定的数据格式将数据提交给服务器。数据格式中至少应包括用户证书和证书载体对随机数的签名。
- g) 服务器接收到客户端提交的登录请求后,应调用证书应用高级接口验证客户端证书及其签名的有效性,包括以下几个方面:
  - 1) 用户证书是否在有效期内。
  - 2) 用户证书是否为服务器端配置的 CA 证书所签发。
  - 3) 用户证书是否已被注销。
  - 4) 验证用户证书对随机数签名的有效性。以 JAVA 组件为例,分别调用 HRSS\_Svr\_HRSS\_Svr\_Certificate\_IsValid() 和 AdvVerifySign () 函数。
- h) 服务器调用证书应用高级接口,解析用户证书信息,获取证书的实体唯一标识。以 JAVA 组件为例,调用 HRSS\_Svr\_AdvGetUserInfoByOid() 函数。
- i) 服务器程序根据证书的实体唯一标识,在系统中找到对应的用户进行相应授权。

## 7.2 单向数字签名与验签

单向数字签名与验签是指客户端对业务数据进行数字签名,服务器验证后进行业务处理的过程,从而实现客户端向服务器端提交数据的完整性和用户业务操作的不可抵赖性。

单向数字签名和验签流程如图 3 所示。



图 3 单向数字签名与验签流程图

单向数字证书签名与验签流程:

- a) 用户使用数字证书登录系统,选择操作菜单,根据提示填写表单数据,提交业务数据。
- b) 客户端组合业务数据,形成原始数据。
- c) 客户端调用证书高级应用接口(ActiveX 控件),使用用户数字证书对原始数据进行数字签名,数字签名时应采用签名密钥,验证时采用签名证书。以 ActiveX 控件为例,分别调用 HRSS\_Ocx\_AtivInit()、HRSS\_Ocx\_Login()、HRSS\_Ocx\_SignedData() 和 HRSS\_Ocx\_GetCert() 函数。
- d) 按照约定格式,将原始数据和数字签名提交到服务器端。

- e) 服务器端接收到客户端的数据后,应验证客户端对业务数据的签名。以 JAVA 组件为例,调用 HRSS\_Svr\_AdvVerifySign()函数。
- f) 验证签名通过后,服务端程序处理业务数据,并保存业务数据和签名。

### 7.3 双向数字签名与验签

双向数字签名与验签是指客户端和服务端双方分别对业务数据进行数字签名和验证,从而实现客户端和服务端双向数据传输的数据完整性和双方业务操作的不可抵赖性。

双向数字签名和验签流程如图 4 所示。



图 4 双向数字签名与验签流程图

双向的数字签名和验签的流程:

- a) 用户使用数字证书登录系统,选择操作菜单,根据提示填写表单数据,提交业务数据。
- b) 客户端组合业务数据,形成原始数据。
- c) 客户端调用证书高级应用接口(ActiveX 控件),使用用户数字证书对原始数据进行数字签名,数字签名时应采用签名密钥,验证时采用签名证书。以 ActiveX 控件为例,分别调用 HRSS\_Ocx\_AtivInit()、HRSS\_Ocx\_Login()、HRSS\_Ocx\_SignedData()和 HRSS\_Ocx\_GetCert()函数。
- d) 按照约定格式,将原始数据和数字签名提交到服务器端。
- e) 验证签名通过后,服务端程序处理业务数据,并保存业务数据和签名。
- f) 服务端程序处理成功后,使用设备证书对应的私钥对业务操作的结果进行数字签名。以 JAVA 组件为例,调用 HRSS\_Svr\_AdvSignData()函数。
- g) 服务端程序保存原始数据、客户端签名、服务器端的数字签名,作为操作证据安全存储在服务器端,返回回执到客户端供用户下载。
- h) 用户下载回执文件并保存。



#### 7.4 加密与解密

加密与解密是指业务数据的加密与解密,实现业务数据的保密性。人力资源和社会保障业务系统中数据加密与解密的方式包括:

- a) 使用标准 SSL 协议对业务数据加密与解密。
- b) 使用数字信封对业务数据进行加密与解密。

业务系统服务器端不需要保存密文,可选择标准 SSL 协议进行加密和解密。服务器端需要保存密文,应通过高级证书应用接口实现基于数字信封方式的加密与解密,如图 5 所示。

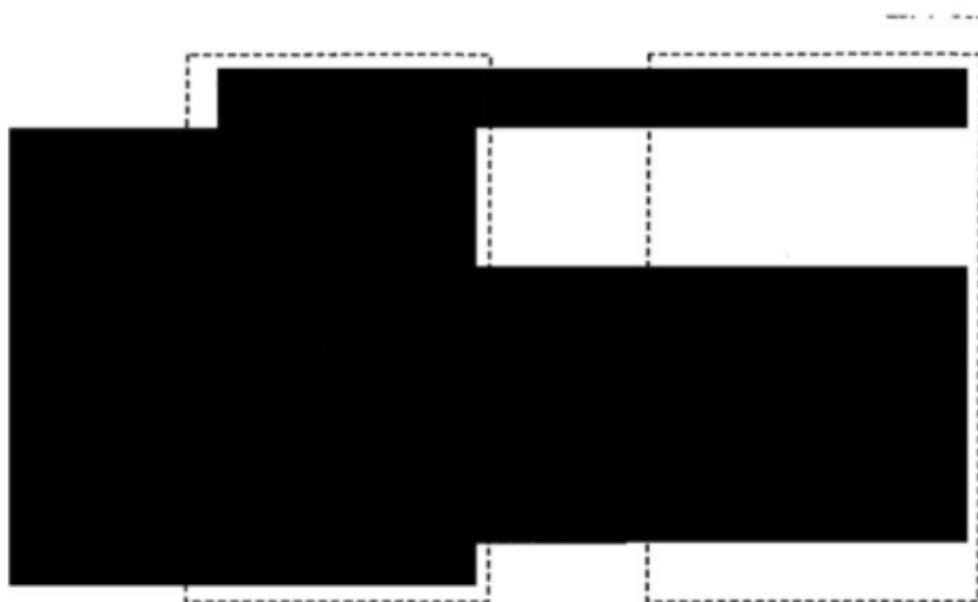


图 5 加密与解密流程图

加密与解密的流程:

- a) 用户使用数字证书登录系统后,选择操作菜单,根据提示填写表单数据,提交业务数据。
- b) 客户端组合业务数据,形成原始数据。
- c) 客户端调用高级证书应用接口(控件),使用服务器端设备证书对原始数据进行加密,形成密文数字信封数据。以 ActiveX 控件为例,调用 HRSS\_Ocx\_SealEnvelope() 函数。
- d) 按照约定格式,将密文数字信封数据提交到服务器端。
- e) 服务端程序接收到密文数字信封的业务数据后,使用服务器设备证书对应的密码设备解密,形成明文格式的业务数据。以 JAVA 组件为例,调用 HRSS\_Svr\_AdvOpenEnvelope() 函数。
- f) 服务端程序将密文数字信封数据解密后,进行相应的业务处理,是否保存密文数字信封数据根据应用需求而定。

#### 7.5 加密签名与解密验签

加密签名与解密验签是指在客户端对业务数据进行加密和签名,服务器端在业务处理前对密文进行解密和验证签名的过程,从而实现数据保密性、完整性和操作的不可否认性。

加密签名与解密验签流程如图 6 所示。

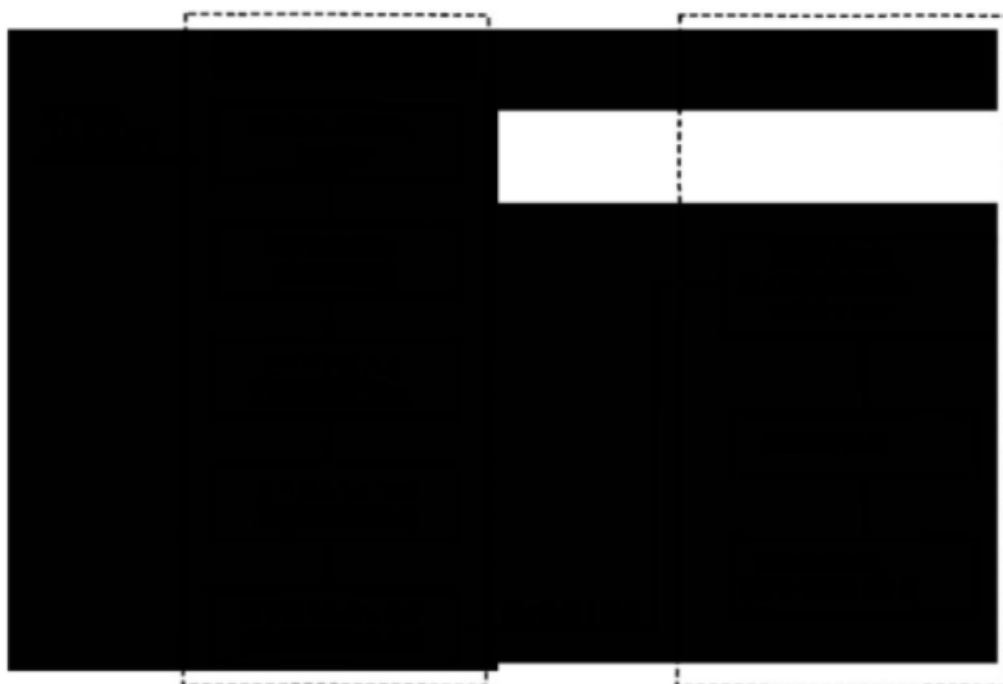


图 6 加密签名与解密验签流程图

加密签名与解密验签的流程：

- a) 用户使用数字证书登录系统后,选择操作,根据提示填写表单数据,提交业务数据。
- b) 客户端组合业务数据,形成原始数据。
- c) 客户端调用高级证书应用接口(控件),使用用户证书对原始数据进行数字签名,数字签名时应采用发送方的签名密钥,验证时采用发送方签名证书。以 ActiveX 控件为例,分别调用 HRSS\_Ocx\_AtvInit()、HRSS\_Ocx\_Login()、HRSS\_Ocx\_SignedData()和 HRSS\_Ocx\_GetCert()函数。
- d) 客户端调用高级证书应用接口(控件),使用设备证书对原始数据进行加密,形成密文数据。以 ActiveX 控件为例,调用 HRSS\_Ocx\_SealEnvelope()函数。
- e) 按照约定格式,将带签名的密文数字信封数据提交到服务器端。
- f) 服务端程序接收到带签名的密文数字信封数据后,使用服务器端设备证书对应的密码设备解密。以 JAVA 组件为例,调用 HRSS\_Svr\_AdvOpenEnvelope()函数。
- g) 服务端程序将密文数字信封数据解密后,使用用户证书验证数字签名,进行相应的业务处理。以 JAVA 组件为例,调用 HRSS\_Svr\_AdvVerifySign()函数。
- h) 保存原始业务数据和客户端签名,作为操作证据安全存储在服务器端。

附录 A  
(资料性附录)  
证书应用场景

### A.1 证书应用集成部署模型

随着政策环境和业务需求的变化,人力资源和社会保障业务系统会随之变化,因此本部分并不针对具体的业务系统进行详细分类和需求分析,而是从证书应用功能角度分析证书应用的安全需求。

证书应用软件分为客户端和服务端两个模块,实现身份认证、数字签名验签、加密解密等基本功能。典型的证书应用集成部署模型如图 A.1 所示。

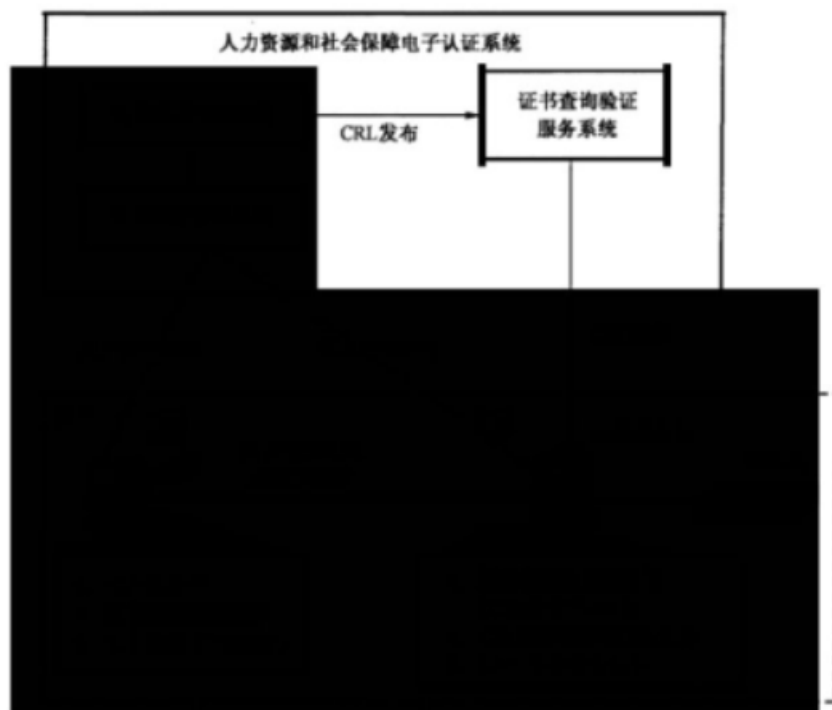


图 A.1 证书应用集成部署模型

根据图 A.1 所示,基于证书的应用系统涉及三个方面:人力资源保障电子认证系统、应用服务器和客户端认证类软件及证书载体等。

人力资源和社会保障电子认证系统负责为用户签发用户数字证书,为应用服务器签发设备证书,并通过证书查验服务系统向应用服务器同步 CRL 文件。

应用服务器端需要部署的内容包括：

- 服务器认证软件包:进行加密解密、数字签名验签的软件接口。软件包为 C 语言的动态库文件或 Java 语言的 Jar 包。
- CRL 同步程序:定时从证书查验服务系统下载 CRL 文件的服务程序。
- CRL 文件:CA 系统发布的黑名单文件。
- CA 证书:根 CA 证书和二级 CA 证书。
- 设备证书:代表服务器身份的数字证书,私钥由密码机产生。
- 密码机:存储服务器设备证书对应密钥的密码设备。

客户端用户需要安装证书应用软件客户端控件及证书载体的驱动程序。在日常登录和使用时应将证书载体(USBKey)插入客户端的电脑中。

客户端和服务端通过双方的数字证书进行双向身份认证,在数据传输过程中,双方可通过数字证书进行签名、验证、加密、解密等安全操作,实现数据完整性、操作的不可抵赖性和数据保密性。

## A.2 证书应用示例

### A.2.1 数字证书初始化绑定

用户首先应按照“证书申请流程”完成能够代表自己身份的数字证书申请,以及业务系统的用户注册。在登录业务系统前,首先应完成数字证书的绑定,即将数字证书的唯一标识与该用户在系统中的账户(或用户名)进行绑定,实现数字证书与系统用户及其权限的一一对应。证书绑定成功后,应及时修改数字证书的 PIN 码。

数字证书绑定的操作流程如图 A.2 所示。

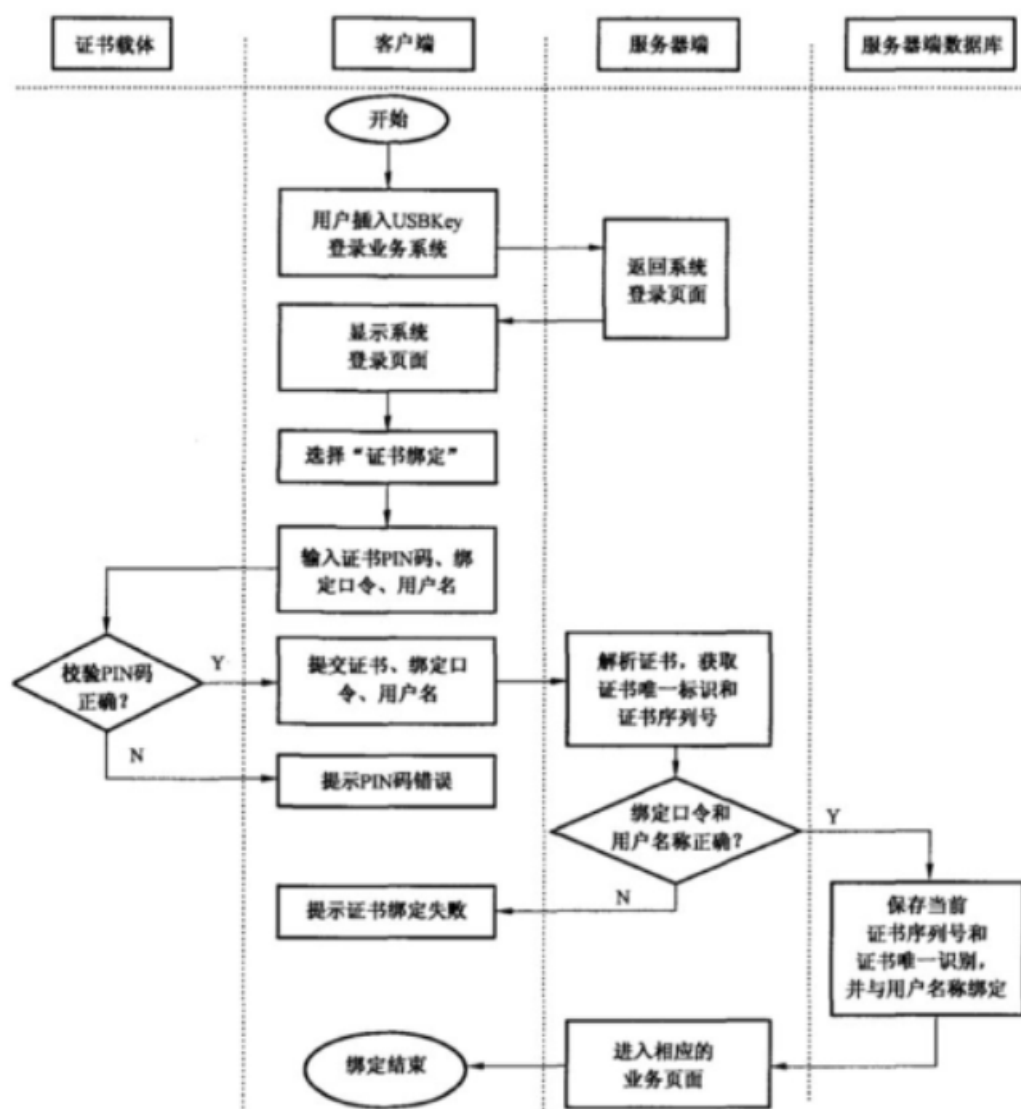


图 A.2 数字证书绑定流程图

A.2.2 数字证书身份认证

证书用户完成证书绑定后,即可使用数字证书登录业务系统,证书登录认证处理流程如下:

- a) 用户将证书载体(USBKey)插入计算机后,输入业务系统 URL 地址,访问系统。
- b) 服务器端程序调用认证软件包产生一个随机数,并保存在会话(Session)中。
- c) 服务器将随机数传递给客户端。
- d) 用户输入证书 PIN 码,由证书载体(USBKey)验证证书 PIN 码的正确性。
- e) 验证证书 PIN 成功后,使用 USBKey 对随机数签名。
- f) 客户端将用户证书和签名值提交到服务器端。
- g) 服务器端程序接收到上述数据后,验证用户证书和签名值的有效性。其中,验证证书的有效性包括:证书有效期、信任源以及是否被吊销。
- h) 解析证书中的唯一标识,根据唯一标识获取用户权限,显示用户权限对应的操作页面。

数字证书身份认证处理流程如图 A.3 所示。

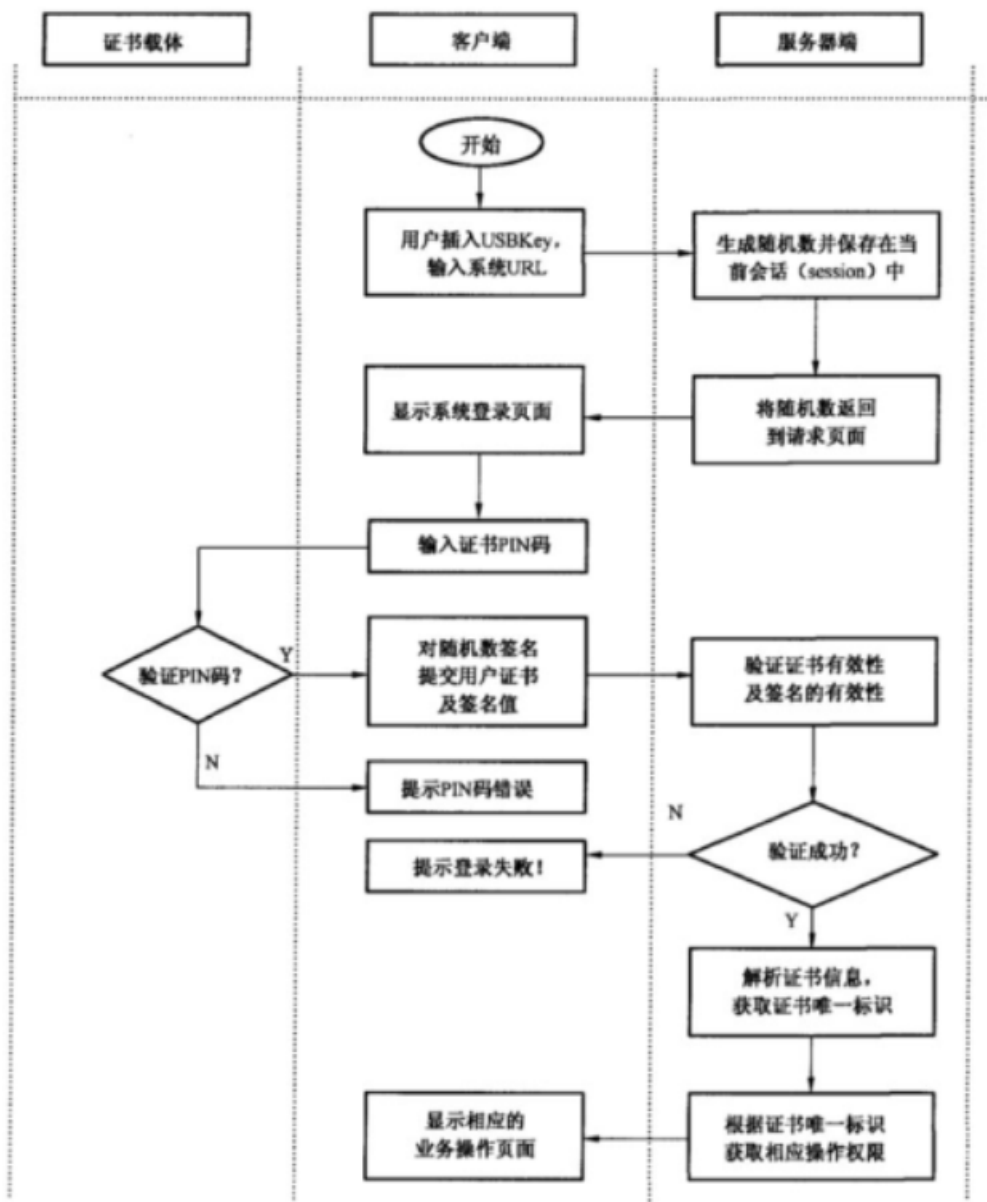


图 A.3 数字证书身份认证流程图

### A.2.3 签名验签示例 1:金保工程异地业务系统

以金保工程异地业务系统为例,介绍在业务系统中如何利用数字证书实现身份认证和对传输的重要业务数据的签名保护,从而实现数据完整性和不可抵赖性。

异地业务系统主要是实现异地社会保险关系转移信息交换的业务系统,异地社会保险关系转移的主要业务操作流程如下:

- a) A 地的参保人员首先登录到本地的异地转移系统,申请保险关系转移业务,并向异地系统上传申请接收函信息。
- b) B 地异地系统查询下载接收函信息,进行办理转出操作,并向异地系统上传分险种的转移单。
- c) A 地异地系统查询转移单,并下载办理转入,完成后上传转移单,操作完毕。

以 A 地办理转入业务申请为例,签名验签处理流程如下:

- a) 用户将证书载体(USBKey)插入计算机,登录到异地业务系统。
- b) 编辑、输入、上传接收函文件。
- c) 使用 USBKey 对接收函文件进行签名。
- d) 提交接收函文件、签名及签名证书。
- e) 服务器端程序验证签名有效性,对接收函验证签名。
- f) 验证成功后,保存当前的接收函,业务处理结束。

异地业务系统证书签名验签业务处理流程如图 A.4 所示。

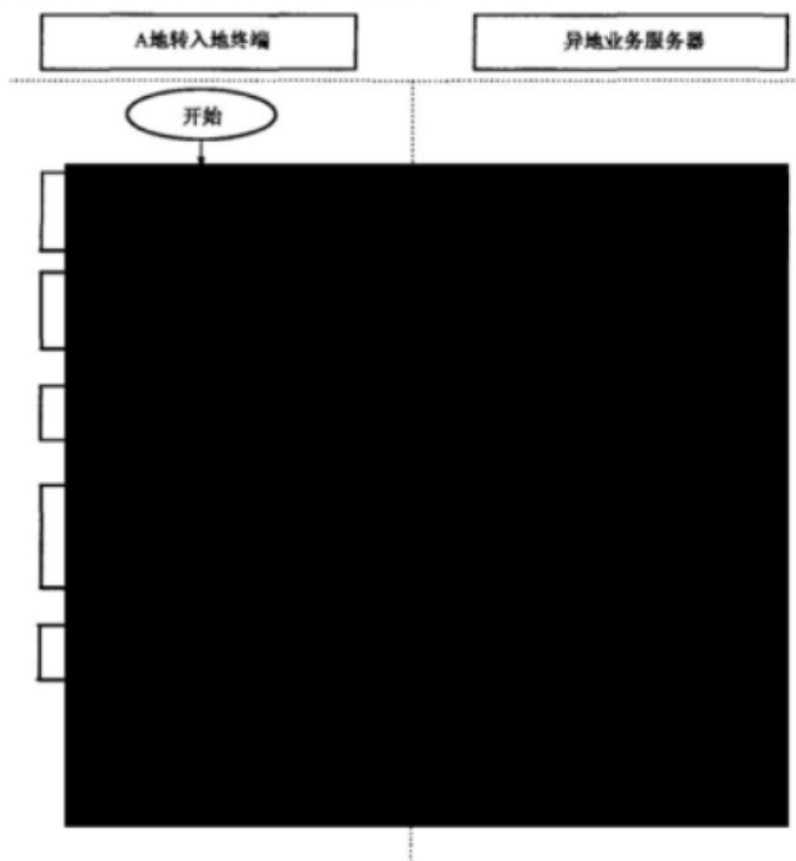


图 A.4 金保工程异地业务系统证书签名验签流程图

### A.2.4 签名验签示例 2:金保工程传输总线系统

金保工程传输总线系统通过建立数据总线通道,利用数字签名技术实现部门间数据的安全上传和下发,主要应用在金保工程联网软件系统数据报表的上传和下发。

金保工程传输总线系统中数字证书签名验签处理流程如下：

- a) 下级应用系统(如:地市联网软件系统)需要上传数据报表到上级部门时,操作联网软件的业务人员将数据上传到本地总线服务器上。
- b) 本地总线服务器定时轮询目录中需要上传的报表文件。当检测到上报文件后,向目的地总线服务器发出服务请求。
- c) 目的地总线服务器接收到请求后,产生随机数,使用目的地总线服务器的设备证书对随机数签名,并将随机数和签名值返回给本地总线服务器。
- d) 本地总线服务器验证目的地总线服务器的证书和签名后,将随机数和需传输的业务数据进行组合,使用本地总线服务器的设备证书对组合数据进行签名,将业务数据和签名值发送到目的地总线服务器。
- e) 目的地总线服务器验证本地总线服务器的设备证书及签名后,保存本地总线服务器传输的业务数据,并返回成功信息。
- f) 至此,从本地总线服务器到目的地总线服务器数据文件传输结束。

传输总线系统证书签名验签业务处理流程如图 A. 5 所示。

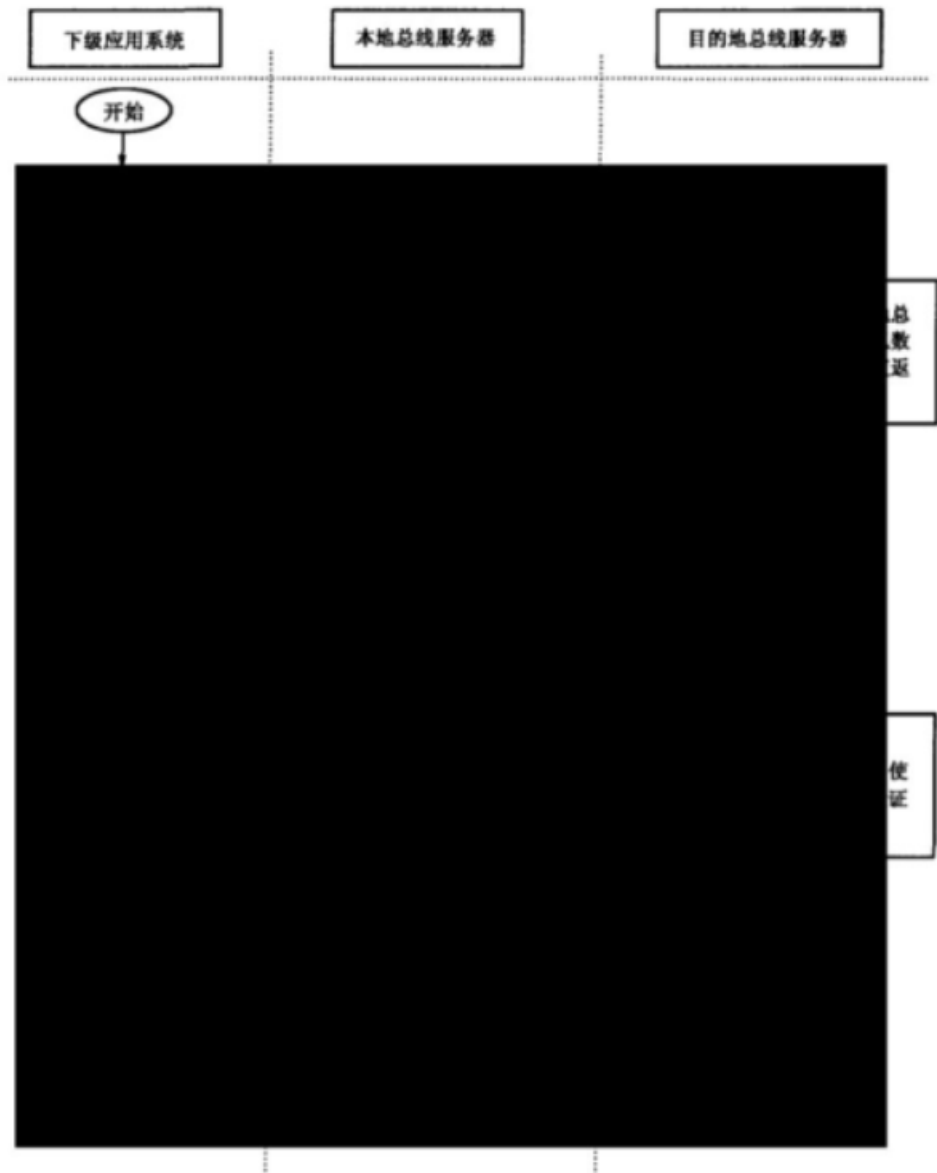


图 A. 5 金保工程传输总线系统证书签名验签流程图

## 附 录 B

## (规范性附录)

## 高级证书应用接口相关标识

|                                 |       |                 |
|---------------------------------|-------|-----------------|
| # define CREATE_CERTCHAIN_ERROR | 13000 | //无法构成证书链       |
| # define CERT_PERIOD_OVER       | 13001 | //证书已过期         |
| # define CERT_IN_CRL            | 13002 | //证书在 CRL 列表中   |
| # define LOCAL_CRL_LOSE         | 13003 | //本地找不到 CRL 列表  |
| # define VERIFY_CERTSIGN_ERROR  | 13004 | //证书签名不通过       |
| # define ABERROR                | 20000 | //异常未知错误        |
| # define ADVINIT_ERROR          | 20001 | //初始化错误         |
| # define ADVEND_ERROR           | 20002 | //释放错误          |
| # define SET_HARDWARE_ERROR     | 20003 | //设置硬件介质错误      |
| # define USER_LOGIN_ERROR       | 20004 | //用户登录失败        |
| # define CHANGE_PIN_ERROR       | 20005 | //修改 PIN 码失败    |
| # define USER_LOGOUT_ERROR      | 20006 | //用户登录退出失败      |
| # define BASE64_ENCODE_ERROR    | 20007 | //BASE64 加密错误   |
| # define BASE64_DECODE_ERROR    | 20008 | //BASE64 解密错误   |
| # define GET_CERT_ERROR         | 20009 | //获取证书错误        |
| # define CHECK_CERT_ERROR       | 20010 | //验证证书错误        |
| # define GET_CERTINFO_ERROR     | 20011 | //获取证书详细信息错误    |
| # define SEAL_ENVELOPE_ERROR    | 20012 | //生成数字信封错误      |
| # define FSEAL_ENVELOPE_ERROR   | 20013 | //对文件作数字信封处理错误  |
| # define SEAL_ENVELOPEEX_ERROR  | 20014 | //批量生成数字信封错误    |
| # define FSEAL_ENVELOPEEX_ERROR | 20015 | //对文件批量生成数字信封错误 |
| # define OPEN_ENVELOPE_ERROR    | 20016 | //拆解数字信封错误      |
| # define FOPEN_ENVELOPE_ERROR   | 20017 | //从文件拆解数字信封错误   |
| # define SIGN_DATA_ERROR        | 20018 | //数字签名错误        |
| # define FSIGN_DATA_ERROR       | 20019 | //对文件数字签名错误     |
| # define SIGN_DATAEX_ERROR      | 20020 | //数字签名错误        |
| # define FSIGN_DATAEX_ERROR     | 20021 | //对文件数字签名错误     |
| # define VERIFY_SIGN_ERROR      | 20022 | //数字验签错误        |
| # define FVERIFY_SIGN_ERROR     | 20023 | //从文件数字验签错误     |
| # define HASH_DATA_ERROR        | 20024 | //哈希运算失败        |
| # define GEN_RANDOM_ERROR       | 20025 | //生成随机数错误       |



|                               |       |                         |
|-------------------------------|-------|-------------------------|
| # define SYMM_ENCRYPT_ERROR   | 20026 | //对称加密错误                |
| # define FSYMM_ENCRYPT_ERROR  | 20027 | //对文件作对称加密错误            |
| # define SYMM_DECRYPT_ERROR   | 20028 | //对称解密错误                |
| # define FSYMM_DECRYPT_ERROR  | 20029 | //对文件作对称解密错误            |
|                               |       |                         |
| # define GETCERT_LDAP_ERROR   | 20030 | //从 LDAP 服务器查询证书错误      |
| # define GET_CRL_ERROR        | 20031 | //从 LDAP 服务器获取 CRL 列表错误 |
|                               |       |                         |
| # define READ_FILE_ERROR      | 20032 | //读取本地文件错误              |
| # define WRITE_FILE_ERROR     | 20033 | //写文件错误                 |
| # define MFC_INIT_ERROR       | 20034 | //MFC 初始化错误             |
| # define MAXFILESIZE_ERROR    | 20035 | //读取文件超过最大值             |
| # define GETALGO_FROMPA_ERROR | 20041 | //通过 PA 获取算法标识失败        |
| # define OPEN_FILE_ERROR      | 20042 | //打开配置文件失败              |
| # define CONFIG_FILE_EMPTY    | 20043 | //配置文件内容为空              |
| # define PARAM_KEY_NOFOUND    | 20044 | //匹配键值失败                |
| # define GET_PARAMVALUE_ERROR | 20045 | //从配置文件获取信息失败           |
|                               |       |                         |
| # define INDATA_IS_NULL       | 20501 | //输入数据为空                |
| # define INCERT_IS_NULL       | 20502 | //输入证书信息为空              |
| # define CLEARTEXT_IS_NULL    | 20503 | //原文信息为空                |
| # define SIGNDATA_IS_NULL     | 20504 | //签名数据信息为空              |
| # define SYMMKEY_IS_NULL      | 20505 | //对称密钥数据为空              |
| # define CERTNUM_IS_ZERO      | 20506 | //证书个数为零                |
| # define FILEPATH_IS_NULL     | 20507 | //文件路径为空                |
| # define SEARCHVALUE_IS_NULL  | 20508 | //搜索字符串为空               |
| # define FILE_IS_NULL         | 20509 | //文件为空                  |
| # define WRITEDATA_IS_NULL    | 20510 | //待写的数据为空               |
| # define INFILE_IS_NULL       | 20512 | //输入文件路径为空              |
| # define OUTFILE_IS_NULL      | 20513 | //输出文件路径为空              |
| # define CLEARFILE_IS_NULL    | 20514 | //原文文件路径为空              |
| # define SIGNFILE_IS_NULL     | 20515 | //签名数据文件路径为空            |
| # define LOGINPIN_IS_NULL     | 20516 | //登录 PIN 为空             |
| # define KEYLABEL_IS_NULL     | 20517 | //私钥标签为空                |
| # define OLDPASSWD_IS_NULL    | 20518 | //旧 PIN 码为空             |
| # define NEWPASSWD_IS_NULL    | 20519 | //新 PIN 码为空             |
| # define OWNERCERT_IS_NULL    | 20520 | //自己证书为空                |
| # define OTHERCERT_IS_NULL    | 20521 | //对方证书为空                |
| # define PARAMKEY_IS_NULL     | 20522 | //查找键名为空                |
| # define GET_CERT_TYPE_ERROR  | 20601 | //证书类型错误                |
| # define UNKOWN_ERROR         | 20700 | //未知错误                  |

中华人民共和国劳动和劳动安全  
行 业 标 准  
人力资源和社会保障电子认证体系  
第 4 部分:证书应用管理规范

LD/T 30.4—2009

\*

中国标准出版社出版发行  
北京复兴门外三里河北街 16 号  
邮政编码:100045

网址 [www.spc.net.cn](http://www.spc.net.cn)

电话:68523946 68517548

中国标准出版社秦皇岛印刷厂印刷  
各地新华书店经销

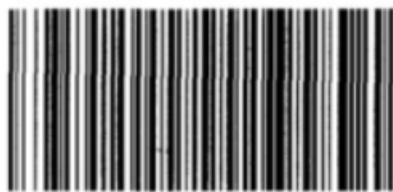
\*

开本 880×1230 1/16 印张 2 字数 56 千字  
2010 年 2 月第一版 2010 年 2 月第一次印刷

\*

书号:155066·2-20299

如有印装差错 由本社发行中心调换  
版权专有 侵权必究  
举报电话:(010)68533533



LD/T 30.4-2009

[www.bzxz.net](http://www.bzxz.net)

免费标准下载网