ICS 35.040
L 80

GM

中华人民共和国密码行业标准

GM/T 0067—2019

基于数字证书的身份鉴别接口规范

Interface specifications of authentication based on digital certificate

2019-07-12 发布　　2019-07-12 实施

国家密码管理局　　发布

目　　次

前　言

本标准以GB/T 15843.3—2016《信息技术　安全技术　实体鉴别　第3部分：采用数字签名技术的机制》为依据，规范了基于数字证书的身份鉴别密码应用接口。

请注意本文件的某些内容可能涉及专利。本文件的发布机构不承担识别这些专利的责任。

本标准由密码行业标准化技术委员会提出并归口。

本标准所使用的密码算法遵从国家密码管理主管部门公布的相关密码算法。

本标准主要起草单位：格尔软件股份有限公司、上海市数字证书认证中心有限公司、山东得安计算机技术有限公司、北京海泰方圆科技有限公司、成都卫士通信息产业股份有限公司、北京数字证书认证中心有限公司、国民技术股份有限公司、长春吉大正元信息技术股份有限公司。

本标准主要起草人：郑强、谭武征、韩玮、马洪富、蒋红宇、罗俊、傅大鹏、付月朋、赵丽丽。

基于数字证书的身份鉴别接口规范

1 范围

本标准规定了公钥密码基础设施体系上层应用中基于数字证书的身份鉴别接口。

本标准适用于公钥密码基础设施体系上层应用中身份鉴别服务的开发、证书应用支撑平台身份鉴别系统的研制及检测，也可用于指导应用系统规范化地使用证书进行身份鉴别。

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 15843.1—2017 信息技术 安全技术 实体鉴别 第1部分：概述

GB/T 15843.3—2016 信息技术 安全技术 实体鉴别 第3部分：采用数字签名技术的机制

3 术语和定义

下列术语和定义适用于本文件。

3.1

证书认证系统 certificate authentication system

对数字证书的签发、发布、更新、撤销等数字证书全生命周期进行管理的系统。

3.2

证书撤消列表 certificate revocation list；CRL

由证书认证机构签发并发布的被撤销证书的列表。

3.3

证书验证 certificate validation

按照验证策略确认证书有效性和真实性的过程。

3.4

数字证书 digital certificate

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。按类别可分为个人证书、机构证书和设备证书，按用途可分为签名证书和加密证书。

3.5

用户凭证 identity token

能够表明用户身份的一段特定数据，用户通过向另一方提交此数据来表明自己的身份，此数据具有不可抵赖性和可验证性。

3.6

双向验证 mutual verify

向双方实体提供对方身份保证的实体鉴别。

3.7

公钥基础设施　public key infrastructure;PKI

基于公钥密码技术实施的具有普适性的基础设施,可用于提供机密性、完整性、真实性及抗抵赖性等安全服务。

3.8

SM2 算法　SM2 algorithm

一种椭圆曲线公钥密码算法,其密钥长度为 256 比特。

3.9

SM3 算法　SM3 algorithm

一种密码杂凑算法,其输出为 256 比特。

4　缩略语

下列缩略语适用于本文件。

CA　证书认证机构(certificate authority)

CN　通用名(common name)

CRL　证书撤销列表(certificate revocation list)

DN　可识别名(distinguished name)

LDAP　轻量级目录访问协议(lightweight directory access protocol)

OID　对象标识符(object identifier)

PKI　公钥密码基础设施(public key infrastructure)

5　实现方式

5.1　概述

身份鉴别实现方式包括代理身份鉴别模式和调用模式,身份鉴别 T 与应用 B 是相互信任的整体,这两种模式下使用的身份鉴别机制遵循 GB/T 15843.3—2016。

5.2　代理身份鉴别模式

在这种模式下,由代理身份鉴别服务 T 对用户 A 的身份进行鉴别,然后把鉴别的结果传递给应用 B,这种身份鉴别模式称为代理身份模式,一般采用消息方式来实现。

鉴别协议在用户 A 和代理身份鉴别服务 T 间进行,这种模式下,用户 A 启动过程并由代理身份鉴别服务 T 对它进行鉴别,唯一性和时效性是通过产生并检验时间戳或序号(见 GB/T 15843.1—2017 的附录 B)来控制的。

代理模式下的单向身份鉴别机制如图 1 所示。

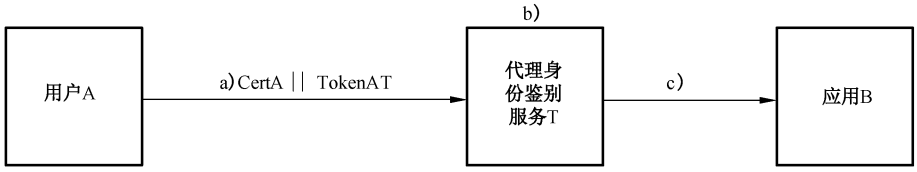

图 1　代理模式下单向身份鉴别机制

a) 用户 A 发送证书和 TokenAT(TokenAT 为用户 A 向代理身份鉴别服务 T 发的凭证,其形式参见 GB/T 15843.3—2016 中的 5.2.2)给代理身份鉴别服务 T;

b) 代理身份鉴别服务 T 在接收到含有 TokenAT 的消息时,执行下列步骤:

1) 验证 A 的证书有效性,包括有效期、是否可信机构颁发、证书状态,以及证书密钥用法验证等;

2) 验证 TokenAT。

c) 代理身份鉴别服务 T 将验证通过的 A 的身份传递给应用 B。

代理身份鉴别模式下的双向身份鉴别机制如图 2 所示。

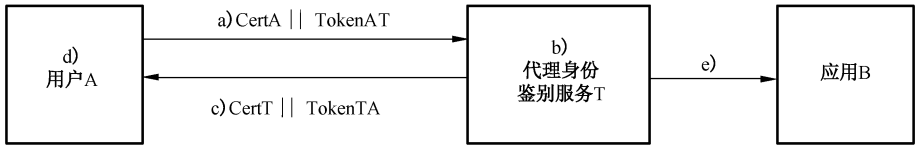

图 2 代理模式下双向身份鉴别机制

a) 用户 A 发送证书和 TokenAT(TokenAT 的形式参见 GB/T 15843.3 中的 5.2.2)给代理身份鉴别服务 T;

b) 代理身份鉴别服务 T 在接收到含有 TokenAT 的消息时,执行下列步骤:

1) 验证 A 的证书有效性,包括有效期,是否可信机构颁发、证书状态,以及证书密钥用法验证;

2) 验证 TokenAT。

c) 代理身份鉴别服务 T 向 A 发送 T 的证书和 TokenTA(TokenTA 的形式参见 GB 15843.3—2016 中的 5.3.2);

d) 在接收到含有 TokenTA 的消息时,用户 A 执行下列步骤:

1) 验证 T 的证书有效性,包括有效期,是否可信机构颁发,是否在黑名单内,以及证书密钥用法验证;

2) 验证 TokenTA。

e) 代理身份鉴别服务 T 将验证通过的 A 的身份传递给应用 B。

5.3 调用模式

对于应用获取到用户身份后,主动调用身份鉴别服务的对外服务接口进行身份鉴别以获取身份鉴别结果的模式,称为调用模式,一般采用接口函数实现。

这种模式中,应用 B 启动验证过程并对用户 A 进行鉴别,通过产生并检验随机数 R_B(见 GB/T 15843.1—2017 的附录 B)来控制鉴别协议的唯一性和时效性。验证机制如图 3 所示:

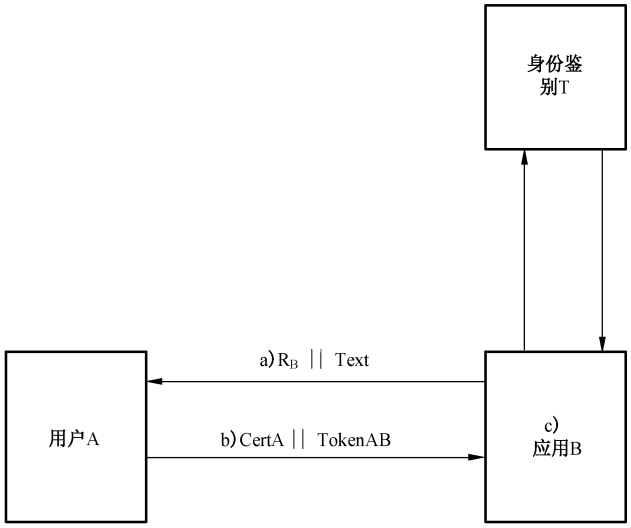

图 3　调用模式下的身份鉴别机制

a)　应用 B 向 A 发送随机数 R_B，并可选地发送一个文本字段 Text；

b)　A 向 B 发送 A 的证书和 TokenAB(TokenAB 的形式参见 GB/T 15843.3—2016 的 5.3.2)；

c)　接收到 A 的证书和 TokenAB 后，应用 B 通过调用身份鉴别 T 来对 A 的身份进行验证。

6　算法标识与数据结构

6.1　算法标识定义

6.1.1　数据类型定义

对于数据的类型定义如表 1 所示。

表 1　数据类型定义

数据类型	描述	定义示例
SGD_HANDLE	会话句柄(指针类型)	typedef void * SGD_HANDLE
SGD_CHAR	8 位字符	typedef char SGD_CHAR
SGD_BYTE	8 位无符号字符	typedef unsigned char SGD_BYTE
SGD_INT32	32 字节有符号整型	typedef int SGD_INT32
SGD_UINT32	32 字节无符号整型	typedef unsigned int SGD_UINT32

6.1.2　常量定义

对于常量的定义如表 2 所示。

表 2　常量定义

宏描述	预定义值	说明
# define SGD_TRUE	0x00000001	布尔值为真
# define SGD_FALSE	0x00000000	布尔值为假

6.1.3　全局参数定义

对于全局参数的定义如表 3 所示。

表 3　全局参数定义

非对称算法标识		
宏描述	预定义值	说明
# define SGD_SM2_1	0x00020200	SM2 签名算法
# define SGD_SM2_2	0x00020400	SM2 密钥交换协议
# define SGD_SM2_3	0x00020800	SM2 加密算法
# define SGD_ECC_n	0x00080000 至 0x80000000	为其他非对称算法预留
杂凑算法标识		
宏描述	预定义值	说明
# define SGD_SM3	0x00000001	SM3 密码杂凑算法
证书解析标识		
宏描述	预定义值	说明
# define SGD_GET_CERT_VERSION	0x00000001	证书版本
# define SGD_GET_CERT_SERIAL	0x00000002	证书序列号
# define SGD_GET_CERT_ISSUER	0x00000005	证书颁发者信息
# define SGD_GET_CERT_ISSUER_CN	0x00000021	证书颁发者 CN
# define SGD_GET_CERT_ISSUER_O	0x00000022	证书颁发者 O
# define SGD_GET_CERT_ISSUER_OU	0x00000023	证书颁发者 OU
# define SGD_GET_CERT_VALID_TIME	0x00000006	证书有效期
# define SGD_GET_CERT_SUBJECT	0x00000007	证书拥有者信息
# define SGD_GET_CERT_SUBJECT_CN	0x00000031	证书拥有者信息 CN
# define SGD_GET_CERT_SUBJECT_O	0x00000032	证书拥有者信息 O
# define SGD_GET_CERT_SUBJECT_OU	0x00000033	证书拥有者信息 OU
# define SGD_GET_CERT_SUBJECT_EMAIL	0x00000034	证书拥有者信息 EMAIL
# define SGD_GET_CERT_DER_EXTENSIONS	0x00000009	证书扩展项信息
验证模式标识		
宏描述	预定义值	说明
# define SGD_CRL_VERIFY	0x00000001	CRL 验证模式
# define SGD_OCSP_VERIFY	0x00000002	OCSP 验证模式

6.2 数据结构定义和说明

对于用户凭证的验证结果如表4所示。

表4 用户凭证验证结果

字段名称	数据长度(字节)	含义
result	4	验证结果
signTime	4	身份鉴别时间,采用的是通用时间类型
resultSign	256	签名信息

实际数据结构定义:

```
typedef struct SIF_ITOKEN_VERIFYED_ {
    SGD_INT32 result;
    SGD_INT32 signTime;
    SGD_BYTE resultSign[256];
}SIF_ITOKEN_VERIFYED;
```

7 接口定义及函数

7.1 身份鉴别接口在公钥密码基础设施应用技术体系框架中的位置

本标准为公钥密码基础设施体系上层应用使用身份鉴别服务制定了统一的接口标准。身份鉴别在公钥密码基础设施应用技术体系框架中的位置如图4所示。

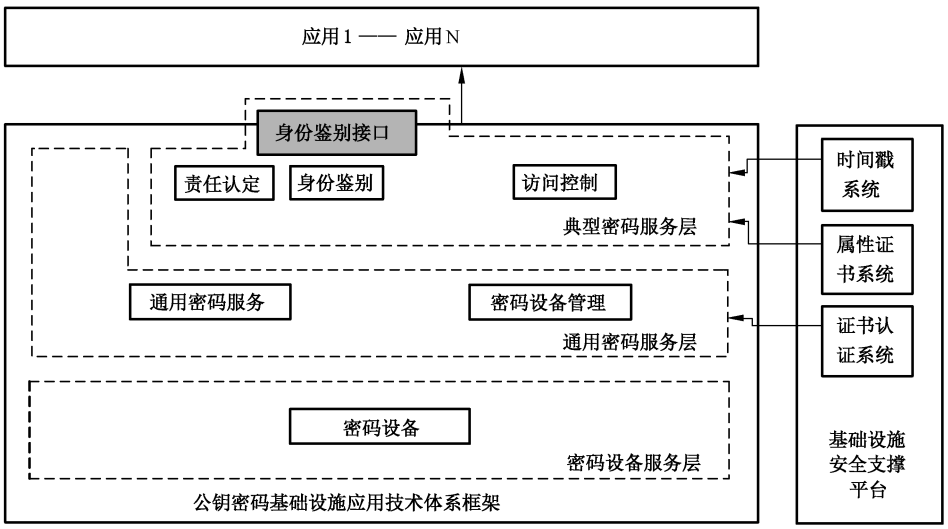

图4 身份鉴别在公钥密码基础设施应用技术体系框架中的位置

7.2 身份鉴别接口逻辑结构

7.2.1 概述

身份鉴别接口的逻辑结构如图5所示。

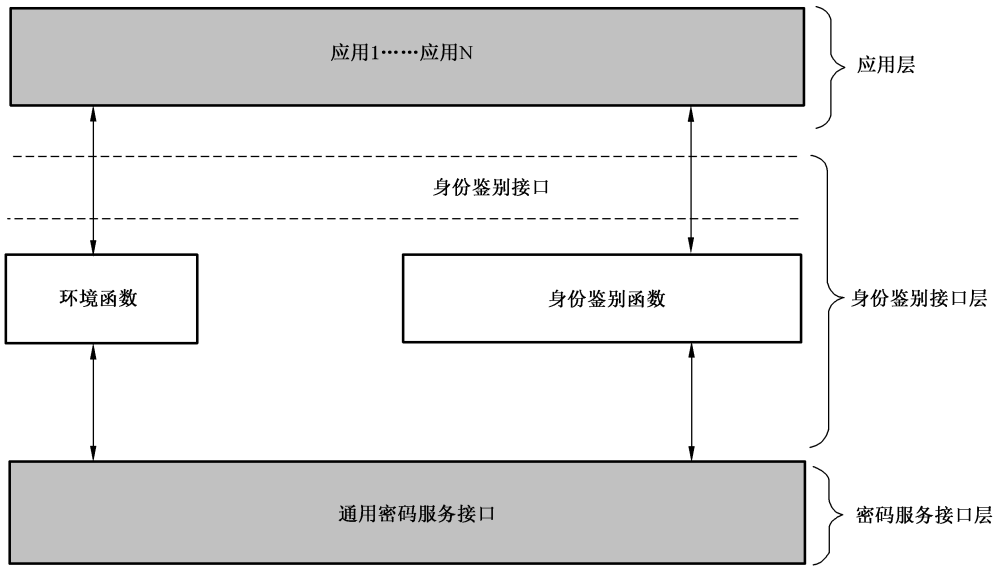

图5 身份鉴别接口体系结构

身份鉴别接口规范所依托的身份鉴别服务模块位于应用系统和密码服务接口之间，通过本接口为应用系统提供身份鉴别服务。身份鉴别模块所需要的密码运算通过密码服务接口规范调用密码服务实现。

身份鉴别接口在逻辑上分为两部分，分别为：环境函数和身份鉴别函数。

7.2.2 环境函数

环境函数负责创建和管理安全程序空间，负责创建和管理安全程序空间中所需的各种资源、信号，并确保安全程序空间在应用程序运行期间不会被非法访问，造成信息泄漏。环境函数负责完成与身份鉴别服务的安全连接，确保后续的安全操作是在安全、可信的程序空间中进行。

应用程序在使用身份鉴别接口时，要首先调用初始化环境函数(SIF_Initialize)创建和初始化安全的应用程序空间，完成与身份鉴别服务的连接和初始化工作。在中止应用程序之前，应调用清除环境函数(SIF_Finalize)，中止与身份鉴别服务的连接，销毁所创建的安全程序空间，防止由于内存残留所带来的安全风险。

7.2.3 身份鉴别函数

身份鉴别函数实现用户信息的获取以及用户身份的验证(主要手段通过证书验证和分析证书撤消列表)。应用程序通过调用身份鉴别函数来实现基于数字证书的身份鉴别。

7.3 消息定义

7.3.1 消息格式定义

消息包括消息头和消息体两个部分，使用XML定义为：

```
〈? xml version="1.0" encoding="UTF-8"?〉
〈msg〉
    〈msg_head〉
      ……
    〈/msg_head〉
    〈msg_body〉
      ……
    〈/msg_body〉
〈/msg〉
```

其中标签 msg_head 标识的段落为消息头，msg_body 标识的段落为消息体。

7.3.2 消息头格式定义

消息头定义方式如下：

```
〈msg_head〉
        〈msg_type〉0〈/msg_type〉
        〈msg_id〉0100〈/msg_id〉
        〈version〉1〈/version〉
〈/msg_head〉
```

其中每个字段定义如表 5 所示：

表 5 消息头字段内容定义

字段名称	字段意义	字段值
msg_type	消息的类型	0——消息请求 1——正确结果返回 2——错误结果返回
msg_id	消息 ID	0001——初始化 0100——获取用户身份信息 1000——生成用户凭证信息 1001——验证用户凭证信息
Version	消息版本号	

7.3.3 获取身份接口消息定义

a) 用户身份获取请求

```
〈? xml version="1.0" encoding="UTF-8"?〉
〈msg〉
    〈msg_head〉
        〈msg_type〉0〈/msg_type〉
        〈msg_id〉0100〈/msg_id〉
        〈version〉1〈/version〉
    〈/msg_head〉
    〈msg_body〉
```

```
        〈connectid〉连接标识〈/connectid 〉
    〈/msg_body〉
〈/msg〉
```

b） 用户身份获取响应

```
〈? xml version="1.0" encoding="UTF-8"?〉
〈msg〉
    〈msg_head〉
        〈msg_type〉1or2〈/msg_type〉
        〈msg_id〉0100〈/msg_id〉
        〈version〉1〈/version〉
    〈/msg_head〉
    〈msg_body〉
        〈connectid〉连接标识〈/connectid 〉
        〈userinfo〉身份信息〈/userinfo〉
        〈error_no〉错误代码〈/error_no〉
    〈/msg_body〉
〈/msg〉
```

7.3.4 用户凭证生成消息

a） 初始化请求

```
〈? xml version="1.0" encoding="UTF-8"?〉
〈msg〉
    〈msg_head〉
        〈msg_type〉0〈/msg_type〉
        〈msg_id〉0001〈/msg_id〉
        〈version〉1〈/version〉
    〈/msg_head〉
    〈msg_body/〉
    〈/msg_body〉
〈/msg〉
```

b） 初始化响应

```
〈? xml version="1.0" encoding="UTF-8"?〉
〈msg〉
    〈msg_head〉
        〈msg_type〉1or2〈/msg_type〉
        〈msg_id〉0001〈/msg_id〉
        〈version〉1〈/version〉
    〈/msg_head〉
    〈msg_body〉
        〈error_no〉错误代码〈/error_no〉
    〈/msg_body〉
〈/msg〉
```

c） 用户凭证生成请求

```
〈? xml version="1.0" encoding="UTF-8"?〉
〈msg〉
    〈msg_head〉
        〈msg_type〉0〈/msg_type〉
        〈msg_id〉1000〈/msg_id〉
        〈version〉1〈/version〉
    〈/msg_head〉
    〈msg_body〉
        〈userseed〉随机信息(Base64 编码)〈/userseed〉
        〈cert〉生成用户凭证用的证书(Base64 编码)〈/cert〉
    〈/msg_body〉
〈/msg〉
```

d) 用户凭证生成响应

```
〈? xml version="1.0" encoding="UTF-8"?〉
〈msg〉
    〈msg_head〉
        〈msg_type〉1or2〈/msg_type〉
        〈msg_id〉1000〈/msg_id〉
        〈version〉1〈/version〉
    〈/msg_head〉
    〈msg_body〉
        〈usertoken〉生成的用户凭证(Base64 编码)〈/usertoken〉
        〈error_no〉错误代码〈/error_no〉
    〈/msg_body〉
〈/msg〉
```

7.3.5 用户凭证验证消息

a) 用户凭证验证请求

```
〈? xml version="1.0" encoding="UTF-8"?〉
〈msg〉
    〈msg_head〉
        〈msg_type〉0〈/msg_type〉
        〈msg_id〉1001〈/msg_id〉
        〈version〉1〈/version〉
    〈/msg_head〉
    〈msg_body〉
    〈usertoken〉生成的用户凭证(Base64 编码)〈/usertoken〉
    〈randmess〉生成用户凭证所用的随机信息(Base64 编码)〈/randmess〉
    〈cert〉生成用户凭证用的证书(Base64 编码)〈/cert〉
    〈/msg_body〉
〈/msg〉
```

b) 用户凭证验证响应

```
〈? xml version="1.0" encoding="UTF-8"?〉
```

```
〈msg〉
    〈msg_head〉
        〈msg_type〉1or2〈/msg_type〉
        〈msg_id〉1001〈/msg_id〉
        〈version〉1〈/version〉
    〈/msg_head〉
    〈msg_body〉
        〈result〉验证结果〈 /result〉
        〈resultsign〉身份鉴别服务对“随机信息 + 验证结果”的签名(Base64 编码)〈/resultsign〉
        〈error_no〉错误代码〈/error_no〉
    〈/msg_body〉
〈/msg〉
```

7.4 函数接口定义

7.4.1 概述

接口函数包括以下具体函数,各函数返回值参见附录 A 错误代码定义:

a) 初始化:SIF_Initialize
b) 终止:SIF_Finalize
c) 获取接口版本:SIF_GetVersion
d) 产生用户凭证需要的随机信息:SIF_GenRandom
e) 产生用户凭证:SIF_GenUserToken
f) 验证用户凭证:SIF_VerifyUserToken
g) 确认验证结果的真实性:SIF_VerifyResult
h) 获取用户身份:SIF_GetUserInfo

7.4.2 初始化函数

原型:	SGD_INT32 SIF_Initialize(SGD _CHAR * pucIpAddr, SGD_INT iPort,SGD_VOID * phHandle);	
描述:	初始化身份鉴别服务,创建身份鉴别服务句柄	
参数:	pucIpAddr [in]	身份鉴别服务器地址,可以为 NULL,表示不连接远程服务
	iPort [in]	身份鉴别服务器端口
	phHandle [out]	函数成功后返回服务句柄,失败为 NULL
返回值:	0	成功
	非 0	失败,返回错误代码

注:此函数在所有其他函数前调用。

当仅仅使用本地证书生成用户凭证时,即声称者调用时可以不连接身份鉴别服务。

7.4.3 终止函数

原型:	SGD_INT32 SIF_Finalize(SGD_HANDLE hHandle);	
描述:	清除应用程序空间,终止身份鉴别服务	
参数:	hHandle [in]	初始化得到的服务句柄
返回值:	0	成功
	非 0	失败,返回错误代码

注：此函数在程序结束时调用。

7.4.4 获取接口版本信息

SGD_INT32 SIF_GetVersion(SGD_HANDLE hHandle,
SGD_UINT32 * puiVersion,
SGD_UINT32 * puiProvider);

原型：		
描述：	获取接口版本信息	
参数：	hHandle [in]	初始化得到的服务句柄
	pVersion [out]	接口版本号
	pProvider[out]	接口提供厂商标识
返回值：	0	成功
	非 0	失败，返回错误代码

7.4.5 产生用户凭证需要的随机信息

SGD_INT32 SIF_GenRandom(SGD_HANDLE hHandle,
SGD_INT32 uiRandLen,
SGD_BYTE * pbRand);

原型：		
描述：	产生指定长度的随机信息，作为生成用户凭证的输入	
参数：	hHandle [in]	初始化得到的服务句柄
	uiRandLen [in]	随机数长度
	pbRand [out]	随机信息
返回值：	0	成功
	非 0	失败，返回错误代码

注：为了保证随机信息的时变性和效率，建议随机信息长度为 16～64 字节。

7.4.6 产生用户凭证

SGD_INT32 SIF_GenUserToken (SGD_HANDLE hHandle,
SGD_BYTE * pbRandMess,
SGD_INT32 uiRandMessLen,
SGD_BYTE * pbCertificate,
SGD_INT32 uiCertificateLen,
SGD_BYTE * pbIdentityToken,
SGD_INT32 * puiIdentityTokenlen);

原型：		
描述：	使用输入数字证书对应的私钥对随机信息签名产生用户凭证	
参数：	hHandle [in]	初始化得到的服务句柄
	pbRandMess [in]	随机信息。该信息可以是 SIF_GenRandom 生成，也可以是时间戳信息或者其他可靠的时变数据
	uiRandMessLen [in]	随机信息的长度
	pbCertificate[in]	Base64 编码的证书
	uiCertificateLen[in]	Base64 编码证书的长度
	pbIdentityToken [out]	产生的用户凭证，格式为 Base64 编码
	puiIdentityTokenlen [in,out]	输入时表示存放用户凭证数据缓冲区的长度，输出时表示用户凭证数据的长度
返回值：	0	成功
	非 0	失败，返回错误代码

7.4.7 验证用户凭证

原型：
```
SGD_INT32 SIF_VerifyUserToken (SGD_HANDLE hHandle,
SGD_BYTE * pbRandMess,
SGD_INT32 uiRandMessLen,
SGD_BYTE * pbIdentityToken,
SGD_INT32 uiIdentityTokenlen,
SGD_BYTE * pbCertificate,
SGD_INT32 uiCertificateLen,
SGD_INT32 iVerifymode,
SIF_ITOKEN_VERIFYED * pstResult);
```

描述：验证用户凭证的有效性，包括验证对应数字证书的有效性

参数：

hHandle [in]	初始化得到的服务句柄
pbRandMess [in]	随机信息
uiRandMessLen [in]	随机信息的长度
pbIdentityToken [in]	产生的用户凭证，数据为 Base64 编码
uiIdentityTokenlen [in]	Base64 编码格式用户凭证数据的长度
pbCertificate [in]	Base64 编码的证书
uiCertificateLen [in]	Base64 编码的证书长度
iVerifymode [in]	验证模式，值参见全局参数定义中的验证模式标识定义
pstResult [out]	凭证验证结果

返回值：

0	成功
非 0	失败，返回错误代码

7.4.8 确认验证结果的真实性

原型：
```
SGD_INT32 SIF_ VerifyResult (SGD_HANDLE hHandle,
SIF_ITOKEN_VERIFYED * pstResult,
SGD_BYTE * pbCertificate,
SGD_INT32 uiCertificateLen);
```

描述：凭证结果的验证，确认是否为指定鉴别服务验证

参数：

hHandle [in]	初始化得到的服务句柄
pstResult [in]	凭证验证结果
pbCertificate [in]	Base64 编码的证书
uiCertificateLen [in]	Base64 编码的证书长度

返回值：

0	成功
非 0	失败，返回错误代码

7.4.9 获取用户身份

原型：
```
SGD_INT32 SIF_GetUserInfo(SGD_CHAR * pcIpAddr,
SGD_INT32 iPort,
SGD_INT32 iConnect,
SGD_CHAR * pbUserInfo,
SGD_INT32 * puiUserInfoLen);
```

描述：	获取声称者身份	
参数：	pcIpAddr [in]	身份鉴别服务器地址
	iPort [in]	身份鉴别服务器端口
	iConnect[in]	声称者连接标识
	pbUserInfo [out]	函数成功后返回声称者证书的 DN 项内容
	puiUserInfoLen [in,out]	输入时表示存放声称者证书 DN 项内容的缓冲区长度，输出时表示声称者证书的 DN 项长度
返回值：	0	成功
	非 0	失败，返回错误代码

注：只在代理模式的实现中使用。

附 录 A
（规范性附录）
错误代码定义和说明

错误代码定义和说明如表 A.1 所示。

表 A.1 错误代码定义和说明

错误代码标识		
宏描述	预定义值	说明
#define SIR_OK	0X00000000	成功
#define SIR_UnknownErr	0X05000001	异常错误
#define SIR_NotInitialize	0X05000002	未初始化
#define SIR_NotConnection	0X05000003	网络连接未成功
#define SIR_MemoryErr	0X05000004	内存不足或内存出错
#define SIR_TimeoutErr	0X05000005	超时错误
#define SIR_VersionErr	0X05000006	版本错误
#define SIR_NotSupport	0X05000007	服务不支持
#define SIR_TrustCertNotFound	0X05000100	信任证书未找到
#define SIR_CRLNotFound	0X05000101	无法获取 CRL
#define SIR_OCSPConnectErr	0X05000002	连接 OCSP 出错
#define SIR_TokenVerifyErr	0X05000200	验证凭证出错
#define SIR_CertInvalid	0X05000201	证书有效期错误
#define SIR_CertNotTrust	0X05000202	证书不受信任
#define SIR_CertInCRL	0X05000203	证书已经作废
#define SIR_NoUserInfo	0X05000300	无法获取指定信息

附 录 B
（资料性附录）
身份鉴别应用流程示例

B.1 使用证书和浏览器访问远程 B/S 应用

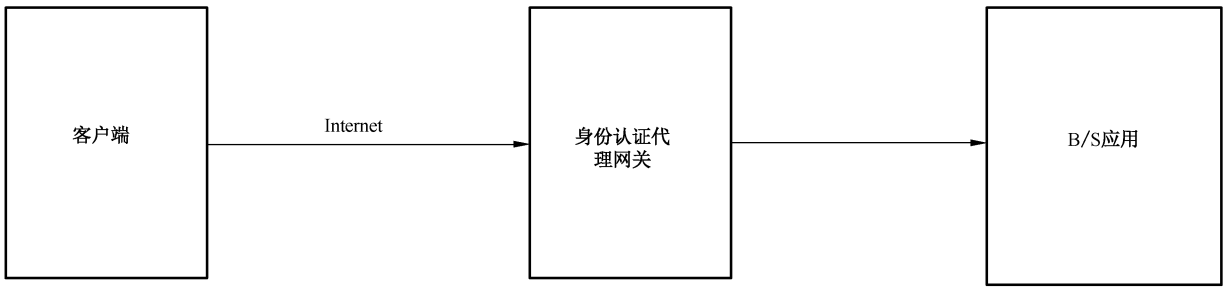

图 B.1 证书和浏览器访问远程 B/S 应用

图 B.1 的场景中，用户远程利用证书和浏览器登录身份鉴别代理网关，通过身份鉴别后，安全访问后端的 B/S 应用，其流程见图 B.2。

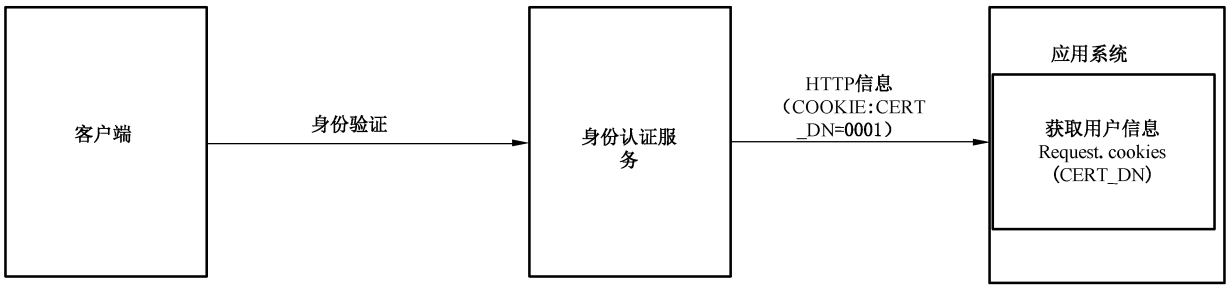

图 B.2 证书和浏览器访问远程 B/S 应用流程图

在此模式中应用仅需要通过脚本从 HTTP 头信息中获取用户信息就可以确认用户，如本示例在 ASP 应用中可以采用 Request.cookies(CERT_DN)获取到用户证书的 DN 信息。

B.2 使用证书客户端访问远程应用

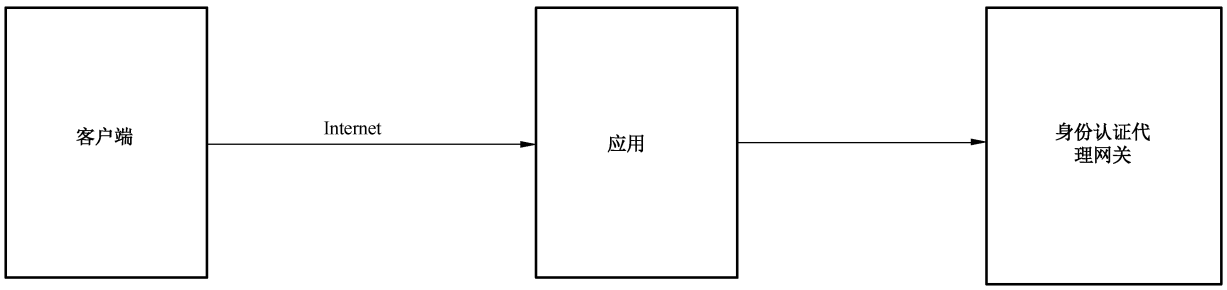

图 B.3 证书客户端访问远程应用

图 B.3 的场景中，用户端安装有专用的应用客户端，它读取用户端的证书凭证，发送到服务端，服务端调用身份鉴别服务提供者提供的身份鉴别接口进行身份鉴别，其流程如图 B.4 所示。

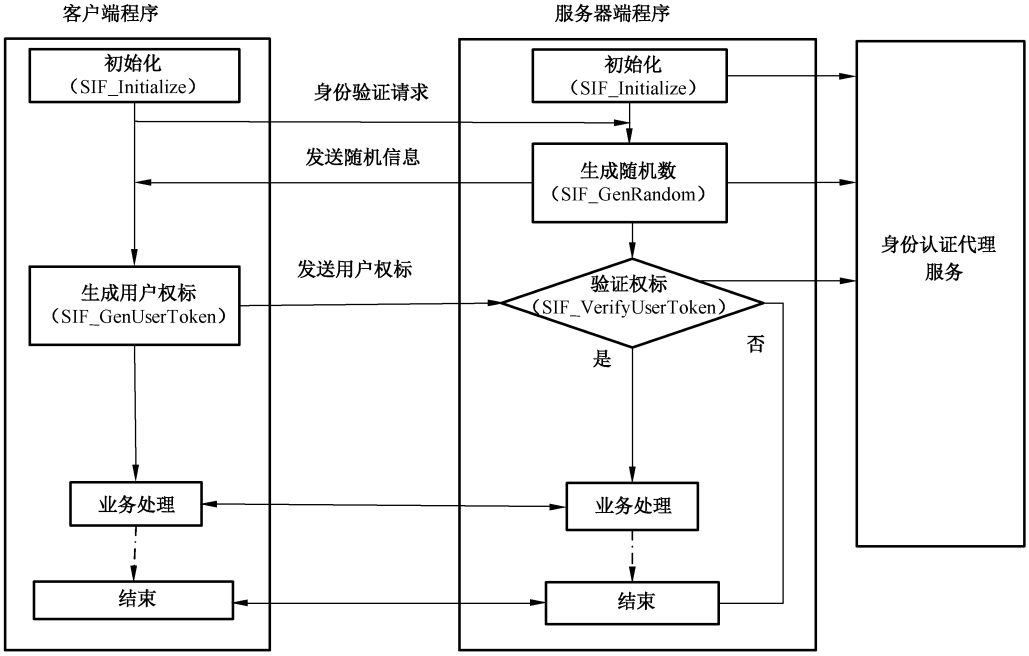

图 B.4　证书客户端访问远程应用流程图

本示例采用调用模式，在该模式中，整个鉴别流程都是由应用系统发起并控制的，由应用系统调用身份鉴别服务提供的各项服务。

参 考 文 献

[1] GB/T 20518—2018 信息安全技术 公钥基础设施 数字证书格式

[2] GB/T 19713—2005 信息安全技术 公钥基础设施在线证书状态协议

[3] GB/T 16264.8—2005 信息技术 开放系统互连 目录 公钥和属性框架

[4] GB/T 17900—1999 网络代理服务器的安全技术要求

[5] GB/T33560—2017 信息安全技术 密码应用标识规范

[6] RFC 2247 Using Domains in LDAP/X.500 Distinguished NamesJanuary 1998

[7] RFC 2252 Lightweight Directory Access Protocol (v3):Attribute Syntax DefinitionsDecember 1997

[8] RFC 2560 X.509 Internet Public Key Infrastructure Online Certificate Status Protocol—OCSPJune 1999

[9] ITU-T X.509 2005 Information technology—Open Systems Interconnection—The Directory:Public-key andattribute certificate frameworks