ICS 35.100.70
L 79

中华人民共和国国家标准

GB/T 16264.8—2005/ISO/IEC 9594-8:2001
代替 GB/T 16264.8—1996

信息技术　开放系统互连　目录
第8部分:公钥和属性证书框架

Information technology—Open Systems Interconnection—The Directory—Part 8:Public-key and attribute certificate frameworks

(ISO/IEC 9594-8:2001,IDT)

2005-05-25 发布　　2005-12-01 实施

中华人民共和国国家质量监督检验检疫总局
中国国家标准化管理委员会　发布

国家图书馆专用

目　　次

国家图书馆专用

国家图书馆专用

前　言

GB/T 16264《信息技术　开放系统互连　目录》分为十个部分：

第1部分：概念、模型和服务的概述

第2部分：模型

第3部分：抽象服务定义

第4部分：分布式操作规程

第5部分：协议规范

第6部分：选择属性类型

第7部分：选择客体类

第8部分：公钥和属性证书框架

第9部分：重复（尚未制定）

第10部分：用于目录行政管理的系统管理用法（尚未制定）

本部分为GB/T 16264的第8部分，等同采用ISO/IEC 9594-8:2001《信息技术　开放系统互连　目录　第8部分：公钥和属性证书框架》。

本部分代替GB/T 16264.8—1996《信息技术　开放系统互连　目录　第8部分：鉴别框架》。本部分与GB/T 16264.8—1996相比，主要变化如下：

——本部分描述了一套作为所有安全服务基础的框架，并规定了在鉴别及其他服务方面的安全要求。本部分还特别规定了以下三种框架：

- 公钥证书框架；
- 属性证书框架；
- 鉴别服务框架。

——定义各种应用使用该鉴别信息执行鉴别的三种方法，并描述如何通过鉴别来支持其他安全服务。

本部分的附录A、附录C、附录D、附录E、附录G和附录H为资料性附录，附录B和附录F为规范性附录。

本部分由中华人民共和国信息产业部提出。

本部分由全国信息安全标准化技术委员会归口。

本部分主要起草单位：中国电子技术标准化研究所。

本部分主要起草人：吴志刚、赵菁华、王颜尊、黄家英、郑洪仁、李丹、高能。

国家图书馆专用

引　言

GB/T 16264 的本部分连同其他几部分一起，用于提供目录服务的信息处理系统的互连。所有这样的系统连同它们所拥有的目录信息，可以看作一个整体，称为“目录”。目录中收录的信息在总体上称为目录信息库(DIB)，它可用于简化诸如 OSI 应用实体、人、终端，以及分布列表等客体之间的通信。

目录在开放系统互连中起着极其重要的作用，其目的是允许在互连标准之下使用最少的技术协定，完成下列各类信息处理系统的互连：

- 来自不同厂家的信息处理系统；
- 处在不同机构的信息处理系统；
- 具有不同复杂程度的信息处理系统；
- 不同年代的信息处理系统。

许多应用都有保护信息的通信免受威胁的安全要求。实际上，所有的安全服务都依赖于通信各方的身份被可靠地认知，即，鉴别。

本部分定义了一个公钥证书框架。这个框架包括了用于描述证书本身和撤销发布证书不再被信任的通知的数据对象规范。本部分中定义的公钥证书框架虽然定义了一些公钥基础设施(PKI)的关键组件，但却不是 PKI 的全部组件。本部分提供了用于建立所有的 PKI 及其规范的基础。

同样的，本部分定义了属性证书的框架。这个框架包括了用于描述证书本身和撤销发布证书不再被信任的通知的数据对象规范。本部分中定义的属性证书框架虽然定义了一些特权管理基础设施(PMI)的关键组件，但却不是 PMI 的全部组件。本部分提供了用于建立所有的 PMI 及其规范的基础。

本部分还定义了目录中的 PKI 和 PMI 对象的持有者信息及存储值和现有值之间的比较。

本部分定义了用于目录向其用户提供鉴别服务的框架。

本部分提供了能被其他标准制定组织和行业论坛定义的行业的基础框架。在这些框架中，许多特性定义为可选的，可以在特定环境中通过描述委托使用。此版为标准的第四版，是在第三版基础上的技术性的修订和增强，但它并不替代第三版。目前实现时仍可使用第三版。然而，在某些方面本部分不支持第三版(即，所报告的缺陷不再予以解决)。推荐尽快执行第四版。

本部分凡涉及密码算法相关内容，按国家有关法规实施。

本部分中所引用的 MD5、SHA-1、RSA、DES、DH 和 DSA 密码算法为举例性说明，具体使用时均须采用国家商用密码管理委员会批准的相应算法。

国家图书馆藏

信息技术　开放系统互连　目录
第8部分:公钥和属性证书框架

第一篇　综　　述

1　范围

本部分描述了一套作为所有安全服务基础的框架,并规定了在鉴别及其他服务方面的安全要求。本部分特别规定了以下三种框架:

- 公钥证书框架;
- 属性证书框架;
- 鉴别服务框架。

本部分中的公钥证书框架包含了公钥基础设施(PKI)信息对象(如公钥证书和证书撤销列表(CRL)等)的定义。属性证书框架包含了特权管理基础设施(PMI)信息对象(如属性证书和属性撤销列表(ACRL)等)的定义。该部分还提供了用于发布证书、管理证书、使用证书以及撤销证书的框架。在规定的证书类型格式和撤销列表模式格式中都包括了扩展机制。本部分同时还分别包括这两种格式一套标准的扩展项,这些扩展项在PKI和PMI的应用中是普遍实用的。本部分包括了模式构件(如对象类、属性类型和用于在目录中存储PKI对象和PMI对象的匹配规则)。超出这些框架的其他PKI和PMI要素(如密钥和证书管理协议、操作协议、附加证书和CRL扩展)将由其他标准机构(如ISO TC68,IETF等)制定。

本部分定义的鉴别模式具有普遍性,并可应用于不同类型的应用程序和环境中。

对目录使用公钥证书和属性证书,本部分还规定了目录使用这两种证书的使用框架。目录使用公钥技术(如证书)实现强鉴别,签名操作和/或加密操作,以及签名数据和/或加密的数据在目录中存储。目录利用属性证书能够实现基于规则的访问控制。本部分只规定框架方面的内容,但有关目录使用这些框架的完整规定、目录所提供的相关服务及其构件在目录系列标准中进行规定。

本部分还涉及鉴别服务框架方面的如下内容:

- 具体说明了目录拥有的鉴别信息的格式;
- 描述如何从目录中获得鉴别信息;
- 说明如何在目录中构成和存放鉴别信息的假设;
- 定义各种应用使用该鉴别信息执行鉴别的三种方法,并描述如何通过鉴别来支持其他安全服务。

本部分描述了两级鉴别:使用口令作为自称身份验证的弱鉴别;包括使用密码技术形成凭证的强鉴别。弱鉴别只提供一些有限的保护,以避免非授权的访问,只有强鉴别才可用作提供安全服务的基础。本部分不准备为鉴别建立一个通用框架,但对于那些技术已经成熟的应用来说本部分可能是通用的,因为这些技术对它们已经足够了。

在一个已定义的安全策略上下文中仅能提供鉴别(和其他安全服务)。因标准提供的服务而受限制的用户安全策略,由一个应用的用户自己来定义。

由使用本鉴别框架定义的应用标准来指定必须执行的协议交换,以便根据从目录中获取的鉴别信息来完成鉴别。应用从目录中获取凭证的协议称作目录访问协议(DAP),由ITU-T X.519|ISO/IEC 9594-5规定。

国家图书馆专用

2 规范性引用文件

下列文件中的条款通过 GB/T 16264 的本部分的引用而成为本部分的条款。凡是注日期的引用文件，其随后所有的修改单(不包括勘误的内容)或修订版均不适用于本部分，然而，鼓励根据本部分达成协议的各方研究是否可使用这些文件的最新版本。凡是不注日期的引用文件，其最新版本适用于本部分。

2.1 等同标准

ITU-T X.411(1999) | ISO/IEC 10021-4 :1999 信息技术 报文处理系统(MHS) 报文传送系统:抽象服务定义和规程

ITU-T X.500(2001) | ISO/IEC 9594-1:2001 信息技术 开放系统互连 目录:概念、模型和服务的概述

ITU-T X.501(2001) | ISO/IEC 9594-2:2001 信息技术 开放系统互连 目录:模型

ITU-T X.511(2001) | ISO/IEC 9594-3:2001 信息技术 开放系统互连 目录:抽象服务定义

ITU-T X.518(2001) | ISO/IEC 9594-4:2001 信息技术 开放系统互连 目录:分布式操作规程

ITU-T X.519(2001) | ISO/IEC 9594-5:2001 信息技术 开放系统互连 目录:协议规范

ITU-T X.520(2001) | ISO/IEC 9594-6:2001 信息技术 开放系统互连 目录:选择的属性类型

ITU-T X.521(2001) | ISO/IEC 9594-7:2001 信息技术 开放系统互连 目录:选择的客体类别

ITU-T X.525(2001) | ISO/IEC 9594-9:2001 信息技术 开放系统互连 目录:重复

ITU-T X.530(2001) | ISO/IEC 9594-10:2001 信息技术 开放系统互连 目录:用于目录行政管理的系统管理的用法

CCITT X.660(1992) | ISO/IEC 9834-1:1993 信息技术 开放系统互连 OSI 登记机构的操作规程:一般规程

ITU-T X.680(1997) | ISO/IEC 8824-1:1998 信息技术 抽象语法记法 1(ASN.1):基本记法规范

ITU-T X.681(1997) | ISO/IEC 8824-2:1998 信息技术 抽象语法记法 1(ASN.1):客体信息规范

ITU-T X.682(1997) | ISO/IEC 8824-3:1998 信息技术 抽象语法记法 1(ASN.1):约束规范

ITU-T X.683(1997) | ISO/IEC 8824-4:1998 信息技术 抽象语法记法 1(ASN.1):ASN.1 规范参数化

ITU-T X.690(1997) | ISO/IEC 8825-1:1998 信息技术 ASN.1 编码规则:基本编码规则(BER)的规范，正规编码规则(CER)和可辨别编码规则(DER)

ITU-T X.812(1995) | ISO/IEC 10181-3:1996 信息技术 开放系统互连:开放式系统安全框架:访问控制框架

ITU-T X.813(1996) | ISO/IEC 10181-4:1996 信息技术 开放系统互连:开放式系统安全框架:认可框架

ITU-T X.880(1994) | ISO/IEC 13712-1:1995 信息技术 远程操作:概念、模型和记法

ITU-T X.881(1994) | ISO/IEC 13712-2:1995 信息技术 远程操作:OSI 实现-远程操作服务元素(ROSE)服务定义

2.2 技术内容等效的标准

CCITT X.800(1991) CCITT 应用的开放式系统互连的安全体系结构

GB/T 9387.2—1995 信息处理系统 开放式系统互连 基本参考模型 第 2 部分:安全体系结构 (idt ISO/IEC 7498-2:1989)

国家图书馆专用

3 术语和定义

下列术语和定义适用于本部分：

3.1 OSI 参考模型安全体系结构定义

下列术语在 GB/T 9387.2—1995 中定义：

a) 非对称(加密) asymmetric(encipherment)；
b) 鉴别交换 authentication exchange；
c) 鉴别信息 authentication information；
d) 机密性 confidentiality；
e) 凭证 credentials；
f) 密码学 cryptography；
g) 数据原发鉴别 data origin authentication；
h) 解密 decipherment；
i) 加密 encipherment；
j) 密钥 key；
k) 口令 password；
l) 对等实体鉴别 peer-entity authentication；
m) 对称(加密) symmetric(encipherment)。

3.2 目录模型定义

下列术语在 GB/T 16264.2 中定义：

a) 属性 attribute；
b) 目录信息库 Directory Information Base；
c) 目录信息树 Directory Information Tree；
d) 目录系统代理 Directory system Agent；
e) 目录用户代理 Directory user Agent；
f) 可辨别名 distinguished name；
g) 项 entry；
h) 客体 object；
i) 根 root。

3.3 定义

本部分定义下列术语：

3.3.1

属性证书 Attribute certificate

属性授权机构进行数字签名的数据结构，把持有者的身份信息与一些属性值绑定。

3.3.2

属性授权机构(AA) Attribute Authority(AA)

通过发布属性证书来分配特权的证书认证机构。

3.3.3

属性授权机构撤销列表(AARL) Attribute Authority Revocation List (AARL)

一种包含发布给属性授权机构的证书索引的撤销列表，发布机构认为这些证书已不再有效。

3.3.4

属性证书撤销列表(ACRL) Attribute Certificate Revocation List (ACRL)

标识由发布机构已发布的、不再有效的属性证书的索引表。

国家图书馆专用

3.3.5

鉴别令牌　authentication token(token)

在强鉴别交换期间传送的一种信息,可用于鉴别其发送者。

3.3.6

机构　Authority

负责证书发布的实体。本部分中定义了两种类型:发布公钥证书的证书认证机构和发布属性证书的属性授权机构。

3.3.7

机构证书　authority certificate

发布给机构(例如证书认证机构或者属性授权机构)的证书。

3.3.8

基础 CRL　base CRL

一种 CRL,用于产生增量 CRL 的基础。

3.3.9

CA 证书　CA certificate

由一个 CA 颁发给另一个 CA 的证书。

3.3.10

证书策略　certificate policy

命名的一组规则,指出证书对具有公共安全要求的特定团体和/或应用的适用范围。例如,一个特定的证书策略表明,用于确认电子数据交换贸易证书的适用范围是价格在某一预定范围内的交易。

3.3.11

证书撤销列表(CRL)　Certificate Revocation List (CRL)

一个已标识的列表,它指定了一套证书发布者认为无效的证书。除了普通 CRL 外,还定义了一些特殊的 CRL 类型用于覆盖特殊领域的 CRL。

3.3.12

证书用户　certificate user

需要确切地知道另一实体的公钥的某一实体。

3.3.13

证书序列号　certificate serial number

为每个证书分配的唯一整数值,在 CA 颁发的证书范围内,此整数值与该 CA 所颁发的证书相关联一一对应。

3.3.14

证书使用系统　certificate using system

证书用户使用的、本部分定义的那些功能实现。

3.3.15

证书确认　certificate validation

确认证书在给定时间有效的过程,可能包含一个证书认证路径的构造和处理,确保该路径上的所有证书在给定时间有效(即证书没有被撤销或者过期)。

3.3.16

证书认证机构 (CA)　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。

3.3.17

证书认证机构撤销列表(CARL)　Certification Authority Revocation List (CARL)

一种撤销列表,它包含一系列发布给证书认证机构的公钥证书,证书发布者认为这些证书不再

国家图书馆专用

有效。

3.3.18

证书认证路径　certification path

一个 DIT 中对象证书的有序序列，通过处理该有序序列及其起始对象的公钥可以获得该路径的末端对象的公钥。

3.3.19

CRL 分布点　CRL distribution point

一个 CRL 目录项或其他 CRL 分发源；由 CRL 分布点分发的 CRL 可以包括仅对某 CA 所发证书全集某个子集的撤销条目，或者可以包括有多个 CA 的撤销条目。

3.3.20

密码体制　cryptographic system ；cryptosystem

从明文到密文和从密文到明文的变换规则汇总，待使用的特定变换由密钥来选定。通常用一种数学算法来定义这些变换。

3.3.21

数据机密性 ，数据保密性　date confidentiality

保护数据免受未授权泄露的服务。数据保密性服务由鉴别框架支持。它可用来防止数据被截取。

3.3.22

委托　delegation

持有特定权限的实体将特定权限移交给另一个实体。

3.3.23

委托路径　delegation path

一个有序的证书序列，将该序列与权限声称者标识的鉴别共同确认权限声称者特定权限的真实性。

3.3.24

增量 CRL　Δ-CRL；delta-CRL；delta CRL；dCRL

部分撤销列表，在可参考的基础 CRL 发布以后，这些证书更改了其撤销状态。

3.3.25

终端实体　end entity

不以签署证书为目的而使用其私钥的证书主体或者是依赖(证书)方。

3.3.26

终端实体属性证书撤销列表(EARL)　End-entity Attribute Certificate Revocation List (EARL)

撤销列表，它包含一系列向持有者发布的属性证书，持有者不是 AA，并且对证书发布者来说这些属性证书不再有效。

3.3.27

终端实体公钥证书撤销列表(EPRL)　End-entity Public-key Certificate Revocation List (EPRL)

撤销列表，它包含发布给非 CA 的主体的公钥证书列表，证书发布者认为这些公钥证书不再有效。

3.3.28

环境变量　environmental variables

与授权决策所需策略相关的信息，它们不包括在静态结构中，但特定权限的验证者可通过本地途径来获得(例如，当天或者当前的账目结余)。

3.3.29

完全 CRL　full CRL

包含在给定范围内全部已被撤销的证书列表。

国家图书馆专用

3.3.30

散列函数,哈希函数　hash function

将值从一个大的(可能很大)定义域映射到一个较小值域的(数学)函数。“好的”散列函数是把该函数应用到大的定义域中的若干值的(大)集合的结果可以均匀地(和随机地)被分布在该范围上。

3.3.31

持有者　holder

由源授权机构直接授权的或由其他属性授权机构间接授权的实体。

3.3.32

间接 CRL(iCRL)　indirect CRL (iCRL)

一个撤销列表,它至少包含不是发布此 CRL 的其他机构发布的证书撤销信息。

3.3.33

密钥协定　key agreement

在线协商密钥值的一种方法,该方法无需传送密钥甚至是加密形式的密钥,例如,迪菲-赫尔曼(Diffie-Hellman)技术(关于密钥协定机制的更多信息参见 GB/T 17901.1)。

3.3.34

对象方法　object method

一个行为,它可在资源上调用(例如,文件系统可以读写和执行对象方法)。

3.3.35

单向函数　one-way function

易计算的(数学)函数 f,通常对于值域中的值 y 来说,很难计算出满足函数 $f(x)=y$ 的在定义域中的自变量 x。可能也存在少数值 y,计算出相应的 x 并不困难。

3.3.36

策略映射　policy mapping

当某个域中的一个 CA 认证另一个域中的一个 CA 时,在第二个域中的特定证书政策可能被第一个域中的证书认证机构认为等价(但不必在各方面均相同)于第一个域中认可的特定证书政策。

3.3.37

私有密钥,私钥　private key

(在公钥密码体制中)用户密钥对中仅为该用户所知的密钥。

3.3.38

特定权限　privilege

由权威机构分派给实体的属性或特性。

3.3.39

特定权限声明者　privilege asserter

使用属性证书或者公钥证书来声明其特定权限的权限持有者。

3.3.40

特定权限管理基础设施(PMI)　Privilege Management Infrastructure (PMI)

支持授权服务的综合基础设施,与公钥基础设施有着密切的联系。

3.3.41

特定权限策略　privilege policy

一种策略,它描述了特权验证者为具有资格的特权声明者提供/执行敏感服务的条件。特权策略与服务相连的属性相关,也和与特权声明者相连的属性相关。

3.3.42

特定权限验证者　privilege verifier

依据特定权限策略验证证书的实体。

国家图书馆专用

3.3.43

公开密钥,公钥 public-key

(在公钥密码体制中)用户密钥对中公布给公众的密钥。

3.3.44

公钥证书 public-key certificate

用户的公钥连同其他信息,并由发布该证书的证书认证机构的私钥进行加密使其不可伪造。

3.3.45

公钥基础设施(PKI) Public-Key Infrastructure (PKI)

支持公钥管理体制的基础设施,提供鉴别、加密、完整性和不可否认性服务。

3.3.46

依赖(证书)方 relying party

依赖证书中的数据来做决定的用户或代理。

3.3.47

角色分配证书 role assignment certificate

一个证书,它包含角色属性,为证书对象/持有者分配一个或多个角色。

3.3.48

角色说明证书 role specification certificate

为角色分配特定权限的证书。

3.3.49

敏感性 sensitivity

表明价值或重要性的资源特性。

3.3.50

弱鉴别 simple authentication

使用口令设置的简单方法进行的鉴别。

3.3.51

安全策略 security ploicy

由管理使用和提供安全服务和设施的安全机构所制定的一组规则。

3.3.52

证书源授权机构(SOA) Source of Authority (SOA)

为资源的特定权限验证者所信任的,位于顶层的分配特定权限的属性授权机构。

3.3.53

强鉴别 strong authentication

使用由密码技术生成的凭证进行的鉴别。

3.3.54

信任 trust

通常,当一个实体假设另一个实体完全按照前者的期望行动时,则称前者"信任"后者。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权威机构之间的关系;鉴别实体应确信它能够"信任"权威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:

AA 属性授权机构

AARL 属性授权机构撤销列表

ACRL 属性证书撤销列表

国家图书馆专用

CA　证书认证机构
CARL　证书认证机构撤销列表
CRL　证书撤销列表
Δ-CRL　增量 CRL
DIB　目录信息库
DIT　目录信息树
DSA　目录系统代理
DUA　目录用户代理
EARL　终端实体属性证书撤销列表
EPRL　终端实体公钥证书撤销列表
ICRL　间接证书撤销列表
PKCS　公钥密码系统
PKI　公钥基础设施
PMI　特定权限管理基础设施
SOA　证书源授权机构

5　约定

本部分目录用 v1 表示本规范的第一版本，v2 表示本规范的第二版本，v3 表示本规范的第三版本，用粗体 Helvetica 字体来表示 ASN.1 记法。当在常规文本中引用 ASN.1 类型和值时，它们同常规文本的差别在于用粗体 Helvetica 字体表示它们。当规定过程语义时典型引用的规程名称同常规文本的差别在于用粗体 Times 表示规程名称。访问控制采用斜体 Times 表示。

如果一个清单中的条款加以编号(与使用“-”或字母的不同)，则这些条款应视为一个规程中的若干步骤。

本部分目录中所使用的符号在表 1 中定义。

表 1　符号

符　号	涵　义
Xp	用户 X 的公钥
Xs	用户 X 的私钥
Xp[I]	用 X 的公钥，对信息 I 进行加密
Xs[I]	用 X 的私钥，对信息 I 进行加密
X{I}	由用户 X 对信息 I 签名，它包含信息 I 和附加加密摘要
CA(X)	用户 X 的证书认证机构
$Ca^n(X)$	(这里，n>1)：CA(CA(..n 次..(X)))
X1《X2》	由证书认证机构 X1 颁发的用户 X2 的证书
X1《X2》X2《X3》	一个(任意长度的)证书链，其中每一项都是一个证书，并且其证书认证机构产生下一个证书。上式等价于下一个证书 X1《Xn+1》。例如：A《B》B《C》提供与 A《C》相同的能力，即给定 Ap，可以从中找到 Cp。
X1p・X1《X2》	一个证书(或证书链)的拆封操作，以便从中获得一个公钥。这是一个中缀操作符，其左操作数为一个证书认证机构的公钥，右操作数则为该证书认证机构颁发的一个证书。输出结果为用户的公钥，它们的证书为右操作数。例如：Ap・A《B》B《C》指出一个操作，该操作使用 A 的公钥，从 B 的证书中获得 B 的公钥 Bp，然后再通过 Bp 来解封 C 的证书。操作的最终结果即为 C 的公钥 Cp
A→B	以 CA(A)《CA^2(A)》开始，以 CA(B)《B》结束的证书链所构成的 A 到 B 的认证路径
注：在该表中出现的符号 X、X1、X2 等等代替用户名，而出现的符号 I 则代替任意信息。	

图表图书馆

6 框架概要

本部分规定了获得和验证实体公钥的框架,以实现解密该实体加密的信息及验证该实体的数字签名。该框架包括证书认证机构(CA)公钥证书的发布和证书用户对该证书的确认。确认包括:

- 在证书用户和证书主体之间建立可信任的证书路径;
- 验证路径中每个证书的数字签名;
- 确认路径中所有的证书有效(即在指定的时间内它们没有过期或没被撤销)。

本部分规定了获得和信任实体特权属性的框架,以决定它们能否被授权访问某个特定资源。该框架包括属性授权机构(AA)证书的发布和特权验证者对证书的确认。确认包括:

- 当证书特权与特权策略比较时,该特权是有效的;
- 如果需要时,应建立证书的可信任委托路径;
- 验证路径途中每个证书的数字签名;
- 确保每个发布者被授予特权代理权;
- 验证证书未过期,或未被发布者撤销。

尽管 PKI 和 PMI 是两个独立的基础设施,并且可以彼此独立建立,但它们相互关联。本部分建议在属性证书中用标示符(pointers)将属性证书的持有者和发布者与相应的公钥证书对应起来。属性证书发布者和持有者的鉴别(即确保声明特权和发布特权的实体的正确性)是通过 PKI 鉴别身份的常规过程完成的。鉴别过程在属性证书框架中不予重复。

6.1 数字签名

PKI 和 PMI 都将数字签名作为一种机制使用,通过该机制发布证书的机构将其绑定在证书中。在 PKI 中,公钥证书上发布 CA 的数字签名证明公钥和证书主体间的绑定。在 PMI 中,发布 AA 的数字签名证明属性(特权)和证书持有者之间的绑定。本条描述了数字签名的通常做法。本部分的第二篇和第三篇专门讨论了在 PKI 和 PMI 中数字签名的使用。

本章并不打算为数字签名规定一种通用的标准,但要规定在 PKI、PMI 及目录中用于签名令牌的方法。

通常是将加密的信息摘要附加在信息(info)的后面来实现对该信息签名。信息的摘要则用一个单向散列函数产生,而加密则是用签名者的私钥来执行的(见图 1)。因此:

$$X\{Info\} = Info, Xs[h,(info)]$$

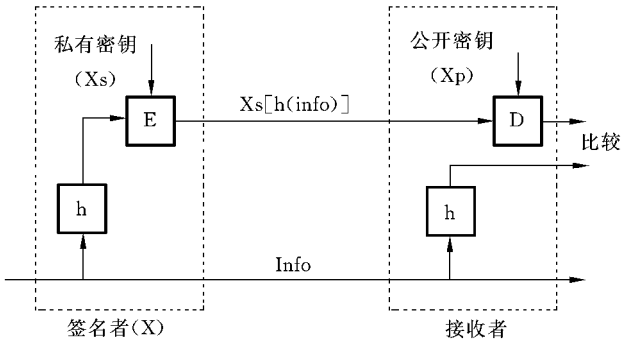


图 1 数字签名

注 1:使用私钥进行加密可以保证签名不被伪造。而散列函数的单向特性则可保证为了拥有相同的散列结果(及签名)而生成的假信息是不能被替代。

已签名的信息的接收方可通过以下方法来验证签名:

- 对信息使用单向散列函数验证;
- 将该结果与通过使用签名者的公钥对签名进行解密的结果作比较。

本部分并不强制在签名时使用单一的单向散列函数。而力图使该框架能够适应任何散列函数，并且支持将来因密码技术、数学技术或计算能力等的更新而带来的方法的改变。但是将要鉴别的两个用户应支持相同的散列函数以确保正确地执行鉴别。因此，在一组相关应用的上下文中，选择一种单一的函数可以充分扩大用户间进行安全鉴别和通信的范围。

被签名的信息包括标识用来计算数字签名的散列算法和加密算法。

对数据项的加密可用下面 ASN.1 来描述：

ENCRYPTED {**ToBeEnciphered** } : : = **BIT STRING** (**CONSTRAINED BY** {

——*必须是把加密过程应用到 BER 编码的八位位组值的结果*——

——*该值为*—— ToBeEnciphered })

位串的值由构成 **ToBeEnciphered** 类型的值的完整编码(使用 ISO/IEC 8825-1:1998 的 ASN.1 基本编码规则)的八位位组及其加密规程来产生。

注 2：加密规程要求约定使用的算法，包括算法的任何参数、任何必需的密钥、初始值，以及填充指令。加密规程规定数据的发送者和接收者之间的同步方法，其中可能包括要发送的信息。

注 3：当输入一个八位位组串时，需要执行加密规程，并产生一个新的八位位组串作为其结果。

注 4：关于数据的发送者和接收者使用的加密算法及其参数的安全协定的机制超出了本部分的范围。

通过加密缩短或者散列变换某数据项来形成该数据项的签名，并且可以利用下列 ASN.1 来描述：

HASH {**ToBeHashed** } : : = **SEQUENCE** {
algorithmIdentifier **AlgorithmIdentifier**,
hashValue **BIT STRING** (**CONSTRAINED BY** {
——*必须是对*-ToBeHashed*的值的 DER 编码八位位组进行散列变换的结果。* }

ENCRYPTED－HASH {**ToBeSigned** } : : = **BIT STRING** (**CONSTRAINED BY** {
—— *必须是把散列过程应用到 DER 编码的八位位组的结果* ——
—— ToBeSigned*的值* — *然后把加密过程应用到这些八位位组*—— })

SIGNATURE {**ToBeSigned** } : : = **SEQUENCE** {
algorithmIdentifier **AlgorithmIdentifier**,
encrypted **ENCRYPTED-HASH** {**ToBeSigned** }}

注 5：加密规程可以是符合注 2 列出的协定，以及散列的八位位组是否直接加密的协定，还可以是仅在使用 ASN.1 基本编码规则进一步将散列的八位位组编码成位串后的协定。

在只要求签名必须附带一个数据类型的情况下，下面 ASN.1 可用来定义由对给定的数据类型应用签名而产生的新的数据类型。

SIGNED {**ToBeSigned** } : : = **SEQUENCE** {
toBeSigned **ToBeSigned**,
COMPONENTS OF **SIGNATURE** {**ToBeSigned** }}

为了能在分布式环境中使 **SIGNED** 和 **SIGNATURE** 类型生效，要求提供一种可辨别的编码。**SIGNED** 和 **SIGNATURE** 数据值的可辨别编码应通过使用 ISO/IEC 8825-1:1998 中定义的基本编码规则以及下列限制来获得：

a) 使用定长格式编码，并使编码的八位位组数最少；

b) 对于串类型，不使用结构化形式编码；

c) 如果一个类型的值是其默认值，则将值省略；

d) Set 类型的成分以其标记值递增次序编码；

国家图书馆专用

e) Set-of 类型的成分以其八位位组值递增次序编码；

f) 如果布尔(Boolean)类型的值为真，则其内容应置为“FF”；

g) 如果一个位串(Bit String)值编码的最后一个八位位组中存在不用的位，则这个位均置为 0；

h) 实数类型的编码不应使用八进制十进制和十六进制，其二进制调节因子为 0；

i) UTC 时间的编码应遵循 ISO/IEC 8825-1:1998 规定；

j) Generalized 时间的编码应遵循 ISO/IEC 8825-1:1998 规定。

生成可辨别编码需要充分理解要编码数据的抽象语法。可以使用本目录对那些包含未知协议扩充或未知属性语法的数据进行签名或检查这些数据的签名。本目录应遵循下列规则：

- 当目录不全知道所接收信息的抽象语法并希望以后对其进行签名时，应保留这些信息的编码。
- 当对要发送的数据进行签名时，签名目录对于那些了解抽象语法的数据采用可辨别编码进行发送，而对于其他的数据则采用其保留的编码进行发送并应对实际发送的编码进行签名；
- 当检验收到数据的签名时，目录应针对收到的实际数据的签名进行检验，而不是把收到的数据转换成可辨别编码。

第二篇 公钥证书框架

这里定义的公钥证书框架是供具有鉴别、完整性、机密性和抗抵赖需求的应用程序使用的。

一个公钥与一个实体的绑定是通过由一个管理机构数字签名的数据结构来实现，该数据结构称为公钥证书。本部分定义了公钥证书的格式，包括一种可扩展性机制和一组特定的证书扩展。不管什么理由，如果管理机构撤销了一个已发布的公钥证书，用户必须能知道这种撤销已经发生，这样，他们就可以避免使用一个不可信赖的证书。本部分定义了撤销列表的格式，包括一个可扩展性机制和一组撤销列表扩展。除了是证书和撤销列表，其他实体也可以定义附加扩展，以便它们更适用于某些特定环境。

任何需要使用公钥证书的系统在应用程序使用证书之前，需要验证该证书的有效性。本部分也定义了实施验证的过程和步骤，包括验证证书本身的完整性、证书的撤销状态以及针对所设定用途的有效性。

本目录使用公钥证书提供的安全服务包括：

- 两个或多个目录组件相互之间的强鉴别；
- 目录操作的鉴别、完整性和机密性；
- 存储数据的完整性和鉴别。

7 公钥和公钥证书

为使一个用户能够信任另一用户的公钥，例如为了鉴别那个用户的身份，应从一个可信任的源获得其公钥。这种源称为证书认证机构(CA)，它通过发布一个公钥证书来证明一个公钥的可信性，该公钥证书将公钥与持有相应私钥的实体绑定到一起。CA 为确保一个实体确实拥有相应私钥所使用的规程以及公钥证书发布相关的其他规程已超出了本部分的范围。证书及其本章后续规定的格式具有以下的特性：

- 任何能够获得和使用证书认证机构的公钥的用户都可以重获证书认证机构所认证的公钥的信任性；
- 除了证书认证机构，没有其他机构能够修改证书而不被检测出来(证书是不可伪造的)。

由于证书是不可伪造的，所以可以通过将其放置在目录中来发布，而不需要以后特意去保护它们。

注 1：尽管在 DIT 中使用唯一性名称来明确定义 CA，但这并不意味着 CA 组织和 DIT 之间有任何联系。

证书认证机构通过对信息集合的签名(见 6.1)来生成用户证书，信息集合包括可辨别的用户名、公钥以及一个可选的包含用户附加信息的唯一性标识符(unique identifier)。唯一性标识符内容的确切格式这里未进行规定，而留给证书认证机构 (CA)去定义。唯一性标识符可以是诸如对象标识符、证书、

日期或是验证可辨别用户名的有效性的证书的其他形式。具体地说，如果一个用户证书的可辨别名为A，唯一性标识符为 UA，并且该证书是由名为 CA 和唯一性标识符为 UCA 的证书认证机构生成的，则用户证书具有下列的形式：

$$CA\langle\langle A\rangle\rangle = CA\{V, SN, AI, CA, UCA, A, UA, Ap, T^A\}$$

这里 V 为证书版本；SN 为证书序列号；AI 为用来签署证书的算法标识符；UCA 为 CA 的可选的唯一性标识符；UA 为用户 A 的可选的唯一性标识符；T^A 表示证书的有效期，由两个日期组成，两者之间时间即是证书的有效期。证书有效期是一个时间区间，在这个时间区间里，CA 必须保证维护该证书的状态信息，也就是说，发布有关撤销的信息数据。由于计时差异，T^A 会在不小于 24 小时的周期内变化，所以要求系统以格林威治时间为基准时间。任何知道 CAp 的用户可用证书上的签名来验证证书的有效性。下列符合 ASN.1 的数据类型可用来表示证书：

```
Certificate                 ::=   SIGNED {SEQUENCE {
    version                 [0]   Version DEFAULT v1,
    serialNumber                  CertificateSerialNumber,
    signature                     AlgorithmIdentifier,
    issuer                        Name,
    validity                      Validity,
    subject                       Name,
    subjectPublicKeyInfo          SubjectPublicKeyInfo,
    issuerUniqueIdentifier  [1]   IMPLICIT UniqueIdentifier OPTIONAL,
                                  --如果存在，版本必须为 v2 或 v3
    subjectUniqueIdentifier [2]   IMPLICIT UniqueIdentifier OPTIONAL,
                                  --如果存在，版本必须为 v2 或 v3
    extensions              [3]   Extensions OPTIONAL
                                  --如果存在，版本必须为 v3    }}

    Version                 ::=   INTEGER {v1(0),v2(1),v3(2) }
    CertificateSerialNumber ::=   INTEGER
    AlgorithmIdentifier     ::=   SEQUENCE {
    Algorithm       ALGORITHM. &id ({SupportedAlgorithms }),
    parameters      ALGORITHM. &Type ({SupportedAlgorithms }{@algorithm }) OPTIONAL }
    --下列信息对象集的定义是可延期的，
    --为了协议执行的一致性声明。集合需要指定由
    --AlgorithmIdentifier 组成的 parameters 的表来约束。
    --SupportedAlgorithms    ALGORITHM    ::=    {... }
    Validity    ::=         SEQUENCE {
```

国家图书馆专用

```
    notBefore      Time,
    notAfter       Time }
SubjectPublicKeyInfo          ::=    SEQUENCE {
    algorithm                        AlgorithmIdentifier,
    subjectPublicKey                 BIT STRING }
Time    ::=    CHOICE {
    utcTime                          UTCTime,
    generalizedTime                  GeneralizedTime }
Extensions    ::=    SEQUENCE OF Extension
Extension     ::=    SEQUENCE {
    extnId                           EXTENSION.&id ({ExtensionSet}),
    critical                         BOOLEAN DEFAULT FALSE,
    extnValue                        OCTET STRING
      --包含一个类型为 &ExtnType 的值的 DER 编码
      --用于由 extnId 所标识的扩展对象
ExtensionSet      EXTENSION      ::=          {...}
```

在将 **Time** 的值用于任何比较操作之前，(例如在搜索中，time 作为匹配规则的一部分)且 **Time** 的语法已选为 **UTCTime** 类型，两位数年域的值将按下列方式合理地转化成一个四位数年域的值，如：

——如果两位数的值是 00 到 49，包括边界，则这个值加上 2000。

——如果两位数的值是 50 到 99，包括边界，则这个值加上 1900。

注 2：在不知道 time 值选择 UTCTime 还是 GeneralizedTime 类型时，使用 GeneralizedTime 可以防止算法执行的相互交叉影响。在描述成员组时，目录标准中定义的证书的某些特定的域负责说明何时可以使用 GeneralizedTime。使用 UTCTime 来表示日期的年域值不应超过 2049。

version 是被编码证书的版本号。如果证书中存在 **extensions** 组件，版本为 v3。如果有 **issuerUniqueIdentifier** 或 **subjectUniqueIdentifier** 组件，版本号将为 v2 或 v3。

如果在扩展中出现未知的元素，并且此扩展没有被标记为关键性的，则按照 ISO/IEC 9594-5. 中 7.5.2.2所描述的扩展性规则，忽略这些未知的元素。

serialNumber 是由 CA 分配给每一个证书的一个整型数。对于 CA 所发布的每一个证书，证书的 **SerialNumber** 的值必须具有唯一性(也就是说，发布者的名字和序列号能识别一个唯一证书)。

signature 包括 CA 签发证书时所使用的算法标识符和散列函数(例如：MD5 With RSA Encryption，SHA-1 With RSA Encryption，id-DSA-with-SHA1 等等)。

issuer 标识签发并发布该证书的实体。

validity 是一个时间区间，在该时间区间里，CA 必须保证维护证书的状态信息。

subject 标识主体公钥域中与公钥相关的实体。

subjectPublicKeyInfo 用来传送可认证的公钥，并标识公钥实例所涉及的算法(例如，RSAEncryption，DHpublicnumber，id-DSA 等)。

国家图书馆专用

issuerUniqueIdentifier 用来在名字重复时唯一地标识发布者。

subjectUniqueIdentifier 用来在名字重复时唯一地标识主体。

注 3：当命名管理机构将一个可辨别名分配给不同的用户时，CA 可使用唯一标识符以避免重复。然而，当多个 CA 为同一用户提供证书时，建议这些 CA 唯一性标识符的分配协调作为该用户注册过程的一部分。

extensions 域允许在证书结构中增加新的域，而不必修改 ASN.1 中对证书结构的定义。扩展域由扩展标识符、关键性标记和一个与所标识的扩展域相关的 ASN.1 类型数据值的编码组成。各扩展域在 **SEQUENCE** 中的排序非常重要，扩展域的规范应该包括重要性排序规则。当一个处理证书的执行过程不能识别扩展域时，如果关键性标记为 **FALSE**，则忽略此扩展域。而如果关键性标记为 **TRUE**，则未被识别的扩展域将使得证书结构无效，也就是说，在证书中，一个不能被识别的关键的证书扩展域将导致使用证书验证签名失败。当一个证书使用的执行过程能识别并处理一个扩展域时，则不管关键性标记取什么值都会处理该扩展域。注：任何标记为非关键的扩展域将在能处理扩展域的证书使用系统与不能识别扩展域的证书使用系统之间引发行为的不一致性，后者将忽略这个扩展域。

如果有未知的元素出现在扩展域中，并且扩展域未被标记为关键性的，按照 ISO/IEC9594-5 的 7.5.2.2中所记录的扩展性规则，将忽略这些未知元素。

对扩展域，CA 可以有三种选择：

a) 它可以排除使用证书中的扩展域；

b) 它可以包含证书的扩展域，但将其标记置为非关键性的；

c) 它可以包含证书的扩展域，并将其标记置为关键性的。

一个有效的工作引擎对扩展域将采取两种可能的行为：

a) 它忽略掉扩展域并接受该证书(其他事件完全一样处理)；

b) 它能处理扩展域，并根据处理过程中扩展域的内容和条件来作出接受或拒绝的决定(比如，路径处理变量的当前值)。

某些扩展域只能标记为关键性的。此时，一个能理解扩展域的有效的工作引擎会处理该扩展域，并根据扩展域的内容(至少部分地)来决定接受/拒绝该证书，而不能理解扩展域的有效引擎将会拒绝该证书。

某些扩展域只能标记为非关键性的。此时，一个能理解扩展域的有效的工作引擎会处理该扩展域，并根据扩展域的内容(至少部分地)来决定接受/拒绝该证书，而不能理解扩展域的有效引擎将接受该证书(除非非扩展域的其他因素导致它被拒绝)。

某些扩展域可以标记为关键性或非关键性的。此时，一个能理解扩展域的有效的引擎会处理该扩展域，并根据扩展域的内容(至少部分地)来决定接受/拒绝该证书，而不管其关键性标记是什么。当扩展域标为非关键性时，不能理解扩展域的有效引擎将接受该证书(除非非扩展域的其他因素导致它被拒绝)，当扩展域标为关键性时，则拒绝该证书。

当 CA 考虑在证书中包含扩展域时，它应该能期待该扩展域在任何情况下都能表现其意图。如果扩展域的内容对确定证书的任何信任度是先决性的，则 CA 标记该扩展域为关键性的。在具体实现中这将会使任何不处理扩展项的有效引擎拒绝该证书(可能限制了能验证证书的应用程序集)。CA 可以标记某些扩展项为非关键性的，以使不能处理扩展域的有效应用程序取得向后的兼容性。当不能处理扩展域的应用程序的向后兼容性和互操作性比 CA 执行扩展域的性能更重要时，则将这些可选的关键扩展项设为非关键性的。当验证者的证书处理应用程序升级到能处理扩展域的过度期内，CA 往往将可选的关键扩展域设为非关键。

如有必要，可以在 ITU-T 推荐/国际标准中定义特定的扩展域，也可由其他组织来定义。标识扩展域的对象标识符应根据 ISO/IEC9834-1 来定义。证书的标准扩展域在本部分的第八章中进行了定义。

以下的对象类用来定义特定的扩展域。

国家图书馆专用

```
EXTENSION ::= CLASS {
  &id                    OBJECTIDENTIFIERUNIQUE,
  &ExtnType }
WITH SYNTAX {
  SYNTAX                 &ExtnType
  IDENTIFIED BY          &id }
```

有两种基本的公钥证书类型,终端实体证书和 CA 证书。

终端实体证书是由 CA 发布给一个主体的证书,而该主体不是任何其他公钥证书的发布者。

CA 证书是由 CA 发布给一个主体的证书,而且此主体本身也是 CA,因此,它也可以发布公钥证书。CA 证书本身可以分为以下类型:

- 自发布证书——证书的发布者和主体是同一个 CA。一个 CA 可以使用自发布的证书,例如,在密钥通过翻转操作期间内提供从旧密钥到新密钥的信任性。
- 自签名证书——这是自发布证书的一种特殊情形,由 CA 使用其私钥来对证书进行签名,而相应公钥在证书中认证。例如,CA 可以使用自签名的证书,对外公布其公钥或有关操作的其他信息。
- 交叉证书——证书的发布者和主体是不同的 CA。交叉认证结构用于这两种情形,CA 发布证书给其他 CA,这种机制可以用来认可主体 CA 的存在性(如,在一个严格的分层结构中),或者认可主体 CA 的存在性(如,一种分布式信任模式)。

每个用户的目录项(如 A,它正参与强鉴别),都包含 A 的一个或多个证书。这个证书是由 A 的证书认证机构(CA)产生的,证书认证机构是 DIT 中的一个实体。A 的证书认证机构(并不一定唯一),可以表示为 CA(A),如果十分了解 A,也可简单的表示成 CA。所以任何一个知道 CA 公钥的用户都可以确认 A 的公钥。这样确认公钥的过程是递归的。

如果用户 A 试图获得用户 B 的公钥,则只要获得用户 B 的 CA 的公钥就可以了。为了使 A 能获得 B 的 CA 公钥,每一个证书认证机构的目录项 X,都包含许多证书。这些证书分为两类:第一类,X 的一些前向证书,这些证书是由其他的 CA 产生的。第二类,是 X 自身生成的返向证书,这些证书用来认证其他证书认证机构的公钥。有了这些证书,用户能够构建从一点到另一点的证书路径。

为了使某一特定的用户获得或验证另一用户的公钥,需要一系列的证书(或称证书列表)。这一证书列表称为证书路径。列表中的每一项都是下一项 CA 的证书。从 A 到 B 的证书路径(表示为 A→B):

- 第一项是由 A 的 CA 产生的一个证书,即由 CA 颁发给某一实体 X1 的证书 CA(A)《X1》;
- 后继的证书为 Xi《Xi+1》;
- 最后一项是 B 的证书。

证书路径在逻辑上形成了两个希望相互验证的用户间目录信息树(DIT)中不可破坏的“信任点链”。用户 A 和 B 为获取证书路径 A→B 和 B→A 的具体方法多种多样。但其中一个简便方法就是形成一个 CA 的层次结构,当然此结构可以与 DIT 的层次结构完全相同,也可部分相同。这样做的好处是拥有这些 CA 层次结构的用户之间只要利用目录就可以建立一个证书路径,而不需要任何先决性信息。为达到这点,每个 CA 可以保存一个(正向)证书和一个反向证书,反向证书主要对应它的上级 CA。

一个用户可以从一个或多个 CA 获取一个或多个证书,每个证书上都有发布它的证书认证机构(CA)的名字。下列 ASN.1 数据类型能用来表示证书和证书路径:

```
Certificates            ::=              SEQUENCE {
userCertificate         Certificate,
certificationPath       Certpath OPTIONAL }
CertificationPath       ::=              SEQUENCE {
```

userCertificate **Certificate**,
theCACertificates **SEQUENCE OF CertificatePair OPTIONAL** }

另外,下列 ASN.1 数据类型可用来表示前向的证书路径。此组件包含能回溯指向始发者的证书路径。

CertPath ::= **SEQUENCE OF CrossCertificates**
CrossCertificates ::= **SET OF Certificate**

在证书路径中每个证书应该具有唯一性,没有证书可以在 CertificationPath 的 theCACertificates 部件的取值中或在 CertPath 的 CrossCertificates 部件的 Certificate 中出现多次。

7.1 密钥对的生成

一个整体安全管理策略的执行者将定义密钥对的生命周期,但这已超出了本框架的范畴。然而,对于整体安全来说,最重要的是只有拥有者才知道所有的私钥。

密钥数据对用户来说是不易记忆的,因此应该采用一种方便传输且合适的方法来存储这些数据。一种比较有效的方法是采用"智能卡"。该卡中保存有用户的私钥和公钥(可选)、用户的证书、以及证书认证机构的公钥的一个拷贝。使用这种卡必须有附加的安全措施,例如,至少应使用一个 PIN(个人标识号),通过要求用户持有并知道如何访问这个卡来增加系统安全性。但存储这种数据的精确方法不在本部分的考虑范围。

有三种产生用户密钥对的方法:

a) 用户自己生成密钥对。这种方法的优点是用户的私钥不会暴露给任何其他实体,但这种方法要求用户有一定级别的能力。

b) 密钥对由第三方生成。第三方应保证以一种物理安全的方式将私钥发放给用户,然后它主动地销毁与生成密钥对有关的所有信息,以及密钥信息本身。必须采用适当的物理安全手段以保证第三方以及数据操作不被篡改。

c) 密钥对由 CA 生成。这是 b)的一种特殊情况。

注:证书认证机构对用户来讲要表现出其可信任的功能性,并且要有保障物理安全的措施。采用这种方法的一个好处就是不用为证书而向 CA 安全传输数据。

所使用的密码系统应对密钥的生成采取某些特殊(技术)限制。

7.2 公钥证书的创建

一个公钥证书关联着公钥和它所描述的用户可辨别名。因此:

a) 证书认证机构在为一个用户创建证书之前,必须使用户的身份标识符满足相关条件;

b) 证书认证机构应保证不会以相同的名字向两个不同用户颁发证书。

向证书认证机构传送信息而不损害其安全性是很重要的,因此应采取适当的物理安全手段:

a) 如果 CA 向一个用户颁发了其公钥已被篡改的证书,这是对安全的严重破坏;

b) 如果采用第 7.1 b)或 7.1 c)中所描述的密钥对生成方法,则用户的私钥必须以安全的方式传送给用户;

c) 如果采用第 7.1 a)或 7.1 b)中所描述的密钥对生成方法,则用户可以使用不同的方式(在线或离线)将其公钥以某种安全的方法传递给 CA。在线方式可以提供一些附加的灵活性以便在用户和 CA 之间执行远程操作。

公钥证书是可公开获得的信息,不需要采取什么特殊的安全手段将其传递给目录。由于证书是由离线的证书认证机构代表用户产生的,用户得到的是证书的一个拷贝(副本),这时用户只需将此信息存入其目录项中,以备今后访问该目录。另一种方法是 CA 可以为用户暂存这个证书,在这种情况下,该代理人应被赋予一定的访问接入的权利。

7.3 证书有效性

发布证书(公钥或属性)的证书认证机构也有责任标识它所发布的证书的有效性。通常,证书随后

有可能被撤销或毁除。撤销和撤销通知可以直接由发布证书的证书认证机构来实施，或者通过发布证书的权威机构授权给其他证书认证机构间接适时地实施。可以通过公开发表它们实际工作的声明，或所发布的证书本身，或其他明确的手段，要求发布证书的权威机构声明，是否：

- 证书不能被撤销；或者
- 证书可以由相同的证书发布证书认证机构直接撤销；或者
- 证书发布证书认证机构授权另一证书认证机构执行撤销工作。

证书认证机构撤销证书的同时必须声明，可以通过一些类似的机制完成声明，使用这种(些)机制可使各依赖方用来获取证书认证机构发布的证书的撤销状态信息。本部分定义证书撤销列表(CRL)机制，但是不排除其他可选机制的使用。可信各方对所有涉及到的证书都核对相应的撤销状态信息，可考虑第 10 章所描述的路径处理过程和在第 16 章所描述的委托路径处理过程，来使证书生效。

证书，包括公钥证书以及属性证书，应该有一个生命周期与之相关联，生命周期一完证书就过期。为了提供服务的连续性，证书认证机构应确保替代证书的可获得性，这个证书用来代替已过期和即将过期的证书。撤销通知日期是证书第一次在 CRL 上出现的日期/时间，而不管它是一个基础 CRL 还是一个增量 CRL。在 CRL 中，撤销通知日期是包含在 thisUpdate 域中的一个值。它是 CA 撤销该证书的实际日期/时间，它与证书在 CRL 上第一次出现的时间可能不同。在 CRL 中，撤销日期是在 revocationDate 组件中所包含的值。

有两种相应做法：

- 可以这样来设计证书的有效性，即每个证书在上一个证书的有效期期满时生效，或者允许在时间上重叠。后者可以使 CA 不必在许多证书同时期满时再安装和分发大量的证书。
- 过期的证书通常会从目录中删除掉。出于安全策略的原因，如果还提供数据的抗抵赖性服务，则可由 CA 将旧的证书保留一定的时间。

证书可以在有其效期满之前撤销。例如，如果假定用户的私钥已被泄露，或用户已不再由证书认证机构认证，或假定证书认证机构证书已不安全，就可以撤销证书。用户证书或证书认证机构证书的撤销将由证书认证机构公布，如果合适，将发放一个替代的新证书。此后证书认证机构可以通过某些离线过程通知证书持有者该证书已撤销。

发布并随后撤销证书的证书认证机构：

a) 对于证书认证机构所发布的所有证书类型，应保留有关其撤销事件的审计记录(例如发布给终端实体和其他证书认证机构的公钥证书、属性证书)；

b) 向使用 CRL 的各依赖方提供撤销状态信息，通过在线证书状态协议或某些其他机制来提供撤销状态信息的发布；

c) 如果使用 CRL，即使撤销证书列表是空的，也要维护和发布 CRL。

各依赖方可以使用多种机制来查找证书认证机构提供的撤销状态信息。例如，证书本身有一个指针向依赖方直接指出提供的撤销信息的位置。撤销列表中有一个指针向依赖方重定向一个不同的撤销列表位置。依赖方可在一个仓库(例如一个目录)内查找撤销信息，或通过此部分范围外的其他方式查找撤销信息(例如本地配置)。

证书认证机构的撤销列表影响着目录项，目录项的维护是此目录和其用户的责任，并且其行为要与安全策略相一致。例如，用户可以用一个新的证书代替旧的证书来修改它的对象项。此后将用新的证书来认证目录中的用户。

如果撤销列表在目录中公布，在目录项内将存在下列类型属性：

- 证书撤销列表；
- 证书认证机构撤销列表；
- 增量撤销列表；
- 属性证书撤销列表；

国家图书馆借用

● 属性权威机构撤销列表。

```
CertificateList          ::=          SIGNED {SEQUENCE {
  version                VersionOPTIONAL,
                                  ——如果引用,必须是版本 2
  signature              AlgorithmIdentifier,
  issuer                 Name,
  thisUpdate             Time,
  nextUpdate             TimeOPTIONAL,
  revokedCertificates    SEQUENE OF SEQUENCE {
  serialNumber           CertificateSerialNumber,
  revocationDate         Time,
crlEntryExtensions       Extensions OPTIONAL }OPTIONAL,
  clrExtensions          [0]     Extensions OPTIONAL }}
```

version 是已编码的撤销列表版本号。如果在撤销列表中出现被标为关键的扩展项组件,则版本号应是 v2。如果在撤销列表中没有出现被标为关键的扩展组件,则版本号可以缺省也可以标为 v2。

signature 包含证书认证机构用来签署撤销列表所使用的算法标识符。

issuer 标识已签署和发布撤销列表的实体。

thisUpdate 是一个日期/时间,在此时刻发布撤销列表。

nextUpdate 如果出现,则表明一个日期/时间,在此时刻将发布这个序列中的下一个撤销列表。下一张撤销列表也能在指定日期之前发布,但一旦超过指定的时间它就不能再发布。

revokedCertificates 标识已被撤销的证书。可以通过其序列号来识别撤销证书。如果这个 CRL 所涵盖的证书都没有被撤销,强烈推荐 CRL 应忽略掉 **revokedCertificates** 参数,而不是在用空的 **SEQUENCE**包含 **revokedCertificates**。

crlExtensions 如果出现,则至少包含一个 CRL 扩展。

注 1:对整个证书列表进行核对是一个本地事件。除非由发布证书认证机构指定特定的次序规则(例如:在某个证书认证机构策略中),此列表不应假定具有特定的次序。

注 2:如果数据的不可抵赖性服务取决于证书认证机构所提供的密钥,则此服务应确保证书认证机构的所有相关密钥(撤销或到期)和时间戳撤销列表都已存档,并由当前证书认证机构认证。

注 3:如果在 CertificateList 中所包含的任何扩展都被定义为关键的,CertificateList 则将出现版本号子项。如果没有包含标为关键的扩展,则可以缺省版本号子项。如果 version 缺省,则在其 CRL 的 revokedCertificates 序列核查中,允许仅支持版本 1CRL 来使用 CRL,它将不会使用到扩展。支持版本 2(或更高版本的) CRL 执行,在缺省版本时,如果在处理中能更早决定非关键的扩展在 CRL 中引用,则能优化其处理。

注 4:当处理证书撤销列表的执行不能识别在 crlEntryExtensions 中的关键扩展时,将假定在最低限度内所认证的证书已撤销,并且不再有效,并将执行有关本地策略所指示的已撤销证书的附加行为。当执行过程不识别在 crlExtensions 中的关键扩展时,将假定所认证的证书已撤销,并且不再有效。然而在后一种情况中,因为列表可能不完整,不能假定撤销证书是有效的证书。对这种情况,本地策略将指示所能采取的行为。在任何情况下,本地策略可以指示本部分以外的更强的行为。

注 5:如果一个扩展影响列表的处理(例如,必须扫描多重 CRL 来检验整个撤销证书列表,或一个能代表一定范围的证书),那么将规定在 crlExtensions 扩展域为关键的,而不管该扩展域在 CRL 中处在什么位置。在一个实体的 crlEntryExtensions 域中所规定的扩展,将放置在该实体中,并且将只影响该实体特定的证书。

注 6:CRL 标准扩展在本部分第 8 章中定义。

8 公钥证书和 CRL 扩展

除非另有规定,本章所定义的证书扩展用于公钥证书。在第 15 章定义了属性证书所使用的扩展。

本章中所定义的CRL扩展可以用于CRL和CARL,也适用于第17章中所定义的ACRL和AARLs。

本章规定以下领域的扩展:

a) 密钥和策略信息:这些证书和CRL扩展传送着有关所涉及的密钥的附加信息,包括主体的密钥标识符和颁发者密钥、假定的或受限的密钥用法的指示器和证书策略的指示器。

b) 主体属性和颁发者属性:这些证书和CRL扩展支持证书主体、证书颁发者或CRL颁发者的各种名字形式的可替换名字。这些扩展也可传送关于证书主体的附加属性信息,以帮助证书用户确信证书主体是一个特定的个人或实体。

c) 认证路径限制:这些证书扩展允许在CA证书中包括限制规范,例如当包括多个证书策略时,由一个CA颁发给另一个CA证书,以便于自动化处理证书路径。当对于某一环境中的不同应用而言策略可不同时,或者当发生与外部环境互操作时,则多个证书策略出现了。这些限制可以限制由CA主体所颁发的证书类型,或者在认证路径上后续可以出现的证书类型。

d) 基本的CRL扩展:这些CRL扩展允许CRL包括撤销原因的指示,提供一个证书的临时暂停和包括CRL颁布序列号,而该序列号允许证书用户在来自某一个CRL颁发者的序列中检测到丢失的CRL。

e) CRL分布点和Δ-CRL:这些证书和CRL扩展允许把来自某一个CA的完整的撤销信息集合分割到若干独立的CRL,并且允许把来自多个CA的撤销信息合并到某一个CRL。这些扩展还支持自早先的CRL颁发以来仅指示变化的部分CRL的使用。

含在证书或CRL中的任何扩展是颁发该证书或CRL的证书认证机构的选项。

在证书或CRL中,扩展被标志为关键的或非关键的。如果扩展被标志为关键的,并且证书使用系统不能识别出该扩展字段类型,或者不执行该扩展的语义,则那个系统应认为该证书无效。如果扩展被标识为非关键的,则不能识别或执行那个扩展类型的证书使用系统可以不检查该扩展而处理证书的其余部分。在本目录标准中的扩展类型定义指出该扩展是否总是关键的、是否总是非关键的、或者是否由证书或CRL颁发者决定其关键性。要求某些扩展总是非关键性的原因是为了允许证书执行如不需要使用这种扩展,以省略对它们支持,而不会危及所有证书认证机构互操作的能力。

注:证书使用系统可以要求某些非关键扩展出现在证书中,以便该证书被认为是可接受的。包括这种扩展的需求可以被证书用户的本地策略规则所蕴含,或者可以是通过包括证书策略扩展(该扩展标志为关键的)中的特定认证策略向证书使用系统所指出的CA策略规则。

对于本部分定义的所有证书扩展、CRL扩展和CRL项扩展,在任何证书、CRL或CRL项中,任何一种扩展类型均不应存在一个以上的实例。

8.1 策略处理

8.1.1 证书策略

此框架包含三种类型的实体:证书用户、证书认证机构和证书主体(或终端实体)。每个实体都按照对应于其他两类实体的义务操作,反过来,也享受它们所提供的有限权利。在证书策略中定义了这些义务和权利。证书策略是一个文档(通常用文本书写)。通过唯一标识符可将它引用到终端实体和证书用户所信任的终端实体上,此唯一标识符可能包含在证书认证机构发布的证书策略扩展中。一张证书可与一个或多个策略联合发布。由策略机构定义策略和分配标识符。由策略机构管理的策略集称为策略域。所有的证书都是按照某种策略发布的,即使此策略既没有在任何地方记录,也没有在证书中引用。本部分不规定证书策略的形式或内容。

通过引入机构公钥并将其作为一个信任点使用,或依赖有相关策略标识符的证书,证书用户可在证书策略下约定其义务。通过发布含有相关策略标识符的证书,机构能在该策略下约定其义务。通过请求和接受含有相关策略标识符的证书并使用相应的私钥,终端实体在该策略下约定其义务。如果不使用证书策略扩展,应通过其他方式来完成所要求的约定。

实体与策略一致的简单声明一般不确保此框架中其他实体的要求。它们需要一些理由相信其他方

进行一个可靠的策略执行。然而,如果在策略中明确规定,证书用户可以接受证书认证机构的保证——终端实体同意在某策略下其职责的约定,没有直接使用终端来证实这一点。证书策略的这一方面超出了本部分的范围。

一个证书认证机构可限制其证书的使用,以便于控制它发布证书所承担的风险。例如,它可以限制证书用户的范围、用户使用证书的目的,以及危害的类型和程度。证书策略中应定义这些事情。

附加的信息,可帮助受作用的实体理解策略,它以策略限定词的形式包含在证书策略扩展中。

8.1.2 交叉认证

一个证书认证机构可以是其他证书认证机构发布证书的主体。在这种情况中,证书称为交叉证书,作为此证书主体的证书认证机构,称为主体证书认证机构,并且,发出交叉证书的证书认证机构,被称为一个中介证书认证机构(见图 2)。交叉证书和终端实体证书都包含一个证书策略扩展。

主体证书认证机构、中介证书认证机构和证书用户所共有的权利和义务,由交叉证书中所标识的证书策略定义,依据该策略,主体证书认证机构可以充当或代表一个终端实体。同时,证书主体、主体证书认证机构和中介证书认证机构所共有的权利和义务,由终端实体证书中所标识的证书策略定义。依据该策略,中介证书认证机构可以充当或代表一个证书用户。

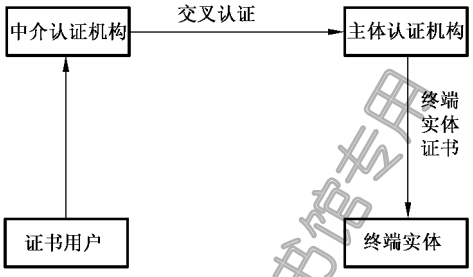


图 2 交叉认证

在某证书路径内的所有证书都共用某策略集,则认为该证书路径在该策略集下有效。

反过来,一个中介证书认证机构也可以是其他证书认证机构发布的证书主体,这样就可以创建超过两个证书长度的证书路径。由于当证书路径变长时,信任度将降低,因此需要控制来确保证书用户将拒绝带有不合理、低关联信任级别的终端实体证书。这一部分是证书路径处理过程的功能。

除了上面所描述情况,还应考虑两种特殊的情况:

a) 证书认证机构不使用证书策略扩展向证书用户传送它的策略请求;或

b) 证书用户或中介证书认证机构将控制策略的工作委托给路径中的下一个证书认证机构。

在第一种情况中,证书不包含证书策略扩展。其结果是,其下的路径有效的策略集是空的。但是,尽管如此,路径是有效的。证书用户仍必须确保它们正在使用的路径与路径中的证书认证机构策略一致。

在第二种情况中,证书用户或中介证书认证机构应该在 *initial-policy-set* 字段或交叉证书中包括特殊的值 *any-policy*。如果证书包含特殊值 *any-policy*,就不应包含其他证书策略标识符。标识符 *any-policy* 与策略限定词没有任何联系。

证书用户能确保通过设置 *initial-explicit-policy* 值来传送其所有合乎标准的义务。这样,只有使用标准证书策略扩展作为其完成约定的证书认证机构,才在路径中是合理的,同时证书用户没有附加义务。因为当证书认证机构担当或代表一个证书用户时,它们也负有义务,它们能确保所有的义务通过在交叉证书中设置 requireExplicitPolicy 可以传送其所有合乎标准的义务。

8.1.3 策略映射

某些证书路径可以跨越两个策略域。交叉证书按照这些权利和义务发布,而这些权利和义务,在本质上,与所有或部分权利和义务是等价的;按照所有或部分权利和义务,主体证书认证机构发布证书给终端实体,即使在策略证书认证机构下,两个证书认证机构使用了不同的唯一标识符用于这些实际等价

的策略中。在这种情况中,中介证书认证机构可以在交叉证书中包含一个策略映射。在策略映射扩展中,中介证书认证机构向证书用户保证,即使在认证路径中随后的实体是不同的策略域中操作的,证书用户仍可以继续享有以前的授权,并且它应继续履行应尽的义务。中介证书认证机构对于某些策略的子集应包含一个或多个映射,在这些策略下,中介证书认证机构发布交叉证书,而对于其他策略,中介证书认证机构不包含映射。如果按照一个或多个证书策略的规定,主体证书认证机构的操作与中介证书认证机构的操作是同样的(也就是说使用相同的唯一标识符),那么这些标识符应不包含在策略映射扩展之中,但包含在证书策略扩展中。

策略映射可以将证书中甚至是证书路径中所有的策略标识符转换为证书用户所公认的等价策略的标识符。

策略不映射到,或不从特殊值 *any-policy* 映射。

证书用户可以决定:除了自身的策略域外,其他策略域所签发的证书不被信任,即使可信任的中介证书认证机构能决定其策略实质与其自身等同。通过将 *initial-policy-mapping-inhibit* 设置到路径确认程序中可确认这一过程。另外,中介证书认证机构可代表其证书用户做类似的决定。为了确保证书用户正确的执行这个请求,它可以在策略约束扩展中设置 **inhibitPolicyMapping**。

8.1.4 认证路径处理

证书用户面临两种策略选择:

a) 能要求认证路径有效,其下的策略集中至少有一条策略由用户预定;或

b) 能请求路径确认模块报告策略集——证书路径是有效的。

在证书用户知道用于特定用途的策略集的情况下,第一种策略可能更为合适,否则,第二种策略可能更为合适。

在第一种情况中,证书路径确认过程表明路径是无效的,只有在 *initial-policy-set* 中指定的一个或多个策略,它才是有效的。然后它将返回 *initial-policy-set* 的子集。在第二种情况中,认证路径确认过程表明在 *initial-policy-set* 下路径是无效的,但在另一个不相交的集合——*authorities-constrained-policy-set* 中是有效的;然后证书用户必须决定其证书的特定使用是否与证书策略的某一个或多个策略一致,在这些策略下路径是有效的。通过将 *initial-policy-set* 设置到 *any-policy* 中,如果路径在任意(没有指定)的策略下都是有效的,证书用户可以引发一个程序来返回一个有效结果。

8.1.5 自发布证书

一个证书认证机构可以给自己发布证书,有三种情形:

a) 一个方便的方法将其公钥编码后传送给证书用户,并由其证书用户储存;

b) 验证除证书和 CRL 签名外的密钥用法(如时间戳);

c) 替换其已到期的证书。

这种类型的证书称为自发布证书,在证书中所引用的发布者和主体名是相同的。为了确认路径,类型 a)的自发布证书使用包含在证书中的公钥来验证,如果它们在路径中出现,将忽略它们。类型 b)的自发布证书仅作为证书路径中的末端证书出现,并且作为末端证书来处理。类型 c)的自发布的证书(也作为自发布中介证书)作为可以中介证书在路径中出现。在替换失效的密钥时,一个很好的实现方法是,交叉证书在使用密钥前请求它的替换公钥,而证书认证机构将请求发布这些带内交叉证书。然而,如果在路径中没有出现自发布证书,它们将作为中介证书而处理,以下情况除外:为了处理 basicConstraints 扩展的 pathLenConstraint 组件、与 *policy-mapping-inhibit-pending* 相关的 *skip-certificates* 值和 *explicit-policy-pending* 标识符,它们将不改变路径长度。

8.2 密钥和策略信息扩展

8.2.1 要求

下列要求与密钥和策略信息有关:

a) CA 密钥对的更新可以以有规则的时间间隔或在特殊情况下发生。需要有一个证书字段来运

送用于验证证书签名的公钥的标识符。证书使用系统可以在查找正确的 CA 证书以确认证书颁发者的公钥时使用这种标识符。

b) 通常,证书主体有不同的公钥,相应地,不同的用途有不同的证书。例如,数字签名和加密密钥协定。需要有一个证书字段帮助证书用户为特定用途的给定主体选择正确的证书,或允许 CA 限定已认证的密钥只可用于特定的用途。

c) 主体密钥对的更新可以在规则的时间间隔或在特殊情况下发生。需要有一个证书字段来运送区别同一主体用于不同时间点的不同公钥的标识符。证书使用系统可以在查找正确的证书时使用这种标识符。

d) 已认证的公钥的私钥一般用于公钥有效性的不同时期。使用数字签名密钥时,签名私钥的使用期一般比验证公钥的时间短。证书的有效期指可以使用公钥的时期。此时期不必与私钥使用期相同。在私钥泄露情况下,如果签名验证者知道私钥的合法使用期,则可以限制暴露期。因此,要求能够在证书中指示私钥使用期。

e) 由于证书可以应用于多证书策略的环境,因此,需要作出在证书中包括证书策略信息的规定。

f) 在一个组织对另一个组织交叉认证时,有时需要商定两个组织中哪些策略可认为是等价的。CA 证书需要证书的颁发者指示其证书策略之一与主体 CA 域内的另一个证书策略等价,这称之为策略映射。

g) 使用本目录标准定义的证书的加密或数字签名系统的用户能预先确定另一个用户所支持的算法。

8.2.2 公钥证书和 CRL 扩展域

定义了下列扩展域:

a) *机构密钥标识符*;

b) *主体密钥标识符*;

c) *密钥使用*;

d) *私钥使用期*;

e) *证书策略*;

f) *策略映射*。

除机构密钥标识符还可用作 CRL 扩展以外,这些扩展字段应只能作为证书扩展使用。除非另有说明,这些扩展可用于 CA 证书和端实体证书。

8.2.2.1 机构密钥标识符域

此字段既可用作证书扩展亦可用作 CRL 扩展。此字段标识用来验证在证书或 CRL 上签名的公钥。它能辨别同一 CA 使用的不同密钥(例如,在密钥更新发生时)。此字段定义如下:

```
authorityKeyIdentifier EXTENSION ::= {
  SYNTAX              AuthorityKeyIdentifier
  IDENTIFIED BY       id-ce-authorityKeyIdentifier }

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 })
```

国家图书馆专用

KeyIdentifier ::= OCTET STRING

此密钥可以通过 keyIdentifier 中的显式密钥标识符来标识，也可以通过此密钥的证书标识（给出 authorityCertIssur 成分中的证书颁发者以及 authorityCertSerialNumber 成分中的证书序列号）来标识，或者可以通过显式密钥标识符和此密钥的证书标识来标识。如果使用两种标识形式，那么，证书或 CRL 的颁发者应保证它们是一致的。对颁发包含扩展的证书或 CRL 的机构的各个密钥标识符而言，某一个密钥标识符均应是唯一的。不要求支持此扩展的实现能够处理 authorityCertIssuer 成分中的各种名字形式。（关于 CeneralName 类型的细节见 12.3.2.1）

证书认证机构应这样分配证书序列号，使得每对（发布者，证书序列号）能唯一地标识单个证书。

此扩展总是非关键的。

8.2.2.2 主体密钥标识符扩展

此字段标识了被认证的公钥。它能够区分同一主体使用的不同密钥（例如，当密钥更新发生时）。此字段定义如下：

subjectKeyIdentifier EXTENSION ::= {
SYNTAX SubjectKeyIdentifier
IDENTIFIED BY id-ce-subjectKeyIdentifier }

SubjectKeyIdentifier ::= KeyIdentifier

对使用密钥标识符的主体的各个密钥标识符而言，某一个密钥标识符均应是唯一的。此扩展总是非关键的。

8.2.2.3 密钥用法扩展

此字段指示已认证的公钥用于何种用途，该字段定义如下：

keyUsage EXTENSION ::= {
SYNTAX KeyUsage
IDENTIFIEDBY id-ce-keyUsage }

KeyUsage ::= BITSTRING {
digitalSignature (0),
nonRepudiation (1),
keyEncipherment (2),
dataEncipherment (3),
keyAgreement (4),
keyCertSign (5),
cRLSign (6),
encipherOnly (7),
decipherOnly (8) }

KeyUsage 类型中的若干比特如下：

a) digitalSignature：验证下列 b)，f)或 g)所标识的用途之外的数字签名；

b) nonRepudiation：验证用来提供抗抵赖服务的数字签名，这种服务防止签名实体假拒绝某种动作（不包括如 f)或 g)中的证书或 CRL 签名）。

c) keyEncipherment：加密密钥或其他安全信息，例如用于密钥传输。

d) dataEncipherment：加密用户数据，但不包括上面 c)中的密钥或其他安全信息。

e) keyAgreement：用作公钥协商密钥。

f) keyCertSign：验证证书的 CA 签名。

国家图书馆专用

g) cRLSign:验证 CRL 的 CA 签名。

h) encipherOnly:当本比特与已设置的 keyAgreement 比特一起使用时,公钥协商密钥仅用于加密数据(本比特与已设置的其他密钥用法比特一起使用的含义未定义)。

i) decipherOnly:当本比特与已设置的 keyAgreement 比特一起使用时,公钥协商密钥仅用于解密数据(本比特与已设置的其他密钥用法比特一起使用的含义未定义)。

keyCertSign 只用于 CA 证书。如果 KeyUsage 被置为 keyCertSign 和基本限制扩展存在于同一证书之中,那么,此扩展 CA 成分的值应被置为 TRUE。CA 还可使用 keyUsag 中定义的其他密钥用法比特,例如,提供鉴别和在线管理事务完整性的 digitalSignature。

此扩展可以是关键的或非关键的,由证书颁发者选择。

如果此扩展标记为关键的,那末,该证书应只用于相应密钥用法比特置为"1"的用途。

如果此扩展标记为非关键的,那么,它指明此密钥的预期的用途或多种用途,并可用于查找具有多密钥/证书的实体的正确密钥证书。它是一个咨询字段,并不是指此密钥的用法限于指定的用途。置为"0"的比特指明此密钥不是预期的这一用途。如果所有比特均为"0",它指明此密钥预期用于所列用途之外的某种用途。

8.2.2.4 扩展密钥用途扩展

此字段指明已验证的公钥可以用于一种或多种用途,除了密钥用法扩展字段指明的基本用途之外的或替代基本用途的用途。此字段定义如下:

```
extKeyUsage EXTENSION ::= {
    SYNTAX                  SEQUENCE SIZE (1..MAX) OF KeyPurposeId
    IDENTIFIED BY           id-ce-extKeyUsage }

KeyPurposeId ::= OBJECT IDENTIFIER
```

密钥的用途可由有此需要的任何组织定义。应按照 ISO/IEC 9834-1 分配用来标识密钥用途的客体标识符。

此扩展可以是关键的,或非关键的,由证书颁发者选择。

如果此扩展标记为关键的,那么,此证书应只用于所指示的用途之一。

如果此扩展标记为非关键的,那么,它指明此密钥的预期用途或一些用途,并可用于查找多密钥/证书的实体的正确密钥/证书。它是一个咨询字段,并不是指证书认证机构将此密钥的用法限于所指示的用途。(然而,使用的应用可以要求所指明的用途,以便证书被此应用接受。)

如果证书包含关键的密钥用途字段和关键的扩展密钥字段,那么,两个字段应独立地处理,并且证书应只用于与两个字段一致的用途。如果没有与两个字段一致的用途,那么,此证书不能用于任何用途。

8.2.2.5 私钥使用期扩展

此字段指明与已验证的公钥相对应的私钥的使用期。它只能用于数字签名密钥。此字段定义如下:

```
privateKeyUsagePeriod EXTENSION ::= {
    SYNTAX                  PrivateKeyUsagePeriod
    IDENTIFIEDBY            id-ce-privateKeyUsagePeriod }

PrivateKeyUsagePeriod ::= SEQUENCE {
    notBefore       [0]     GeneralizedTime OPTIONAL,
    notAfter        [1]     GeneralizedTime OPTIONAL }
    ( WITHCOMPONENTS        {...,notBefore PRESENT }|
```

国家图书馆专用

WITHCOMPONENTS {..., notAfter PRESENT})

notBefore 成分指明私钥可能用于签名的最早日期和时间。如果没有 notBefore 成分，那么不提供有关私钥有效使用期何时开始的信息。notAfter 成分指明私钥可以用于签名的最迟日期和时间。如果没有 notAfter 成分，那么，不提供有关私钥有效使用期何时结束的信息。

此扩展总是非关键的。

注 1：私钥有效使用期可以与证书有效性周期指明的已验证的公钥有效性不同。就数字签名密钥而言，签名的私钥使用期一般比验证公钥的时间短。

注 2：数字签名的验证者想要检查直到验证时刻此密钥是否未被撤销，例如，由于密钥泄露，那么，在验证时，对公钥而言的有效证方应仍存在。在公钥的证书期满之后，签名验证者不能依赖 CRL 所通知的协议。

8.2.2.6 证书策略扩展

此字段列出了由颁发的 CA 所认可的证书策略，这些策略适用于证书以及关于这些证书策略的任选的限定符信息。证书策略表用来决定一条认证路径的有效性，如在 12.4.3 中描述的。可选的限定词没有在处理过程的认证路径中用到，但是为证书提供的相关限定词，用于应用过程的输出，决定一条有效路径是否适合特定事务处理。一般，不同的证书策略性与使用已认证的密钥的不同应用有关。在终端实体证书中，扩展的存在表示有效证书的证书策略。存在于由一个 CA 到另一个 CA 发布的证书中的扩展，指出了包含该证书的证书路径的证书策略可以是有效的。该字段定义如下：

```
certificatePolicies EXTENSION ::= {
    SYNTAX              CertificatePoliciesSyntax
    IDENTIFIED BY       id-ce-certificatePolicies }

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

PolicyInformation ::= SEQUENCE {
    policyIdentifier            CertPolicyId,
    policyQualifiers            SEQUENCE SIZE (1..MAX) OF
                                PolicyQualifierInfoOPTIONAL }

CertPolicyId ::= OBJECT IDENTIFIER

PolicyQualifierInfo ::= SEQUENCE {
    policyQualifierId           CERT-POLICY-QUALIFIER.&id
                                ({SupportedPolicyQualifiers}),
    qualifier                   CERT-POLICY-QUALIFIER.&Qualifier
                                ({SupportedPolicyQualifiers}{@policyQualifierId})
                                OPTIONAL }

SupportedPolicyQualifiers CERT-POLICY-QUALIFIER ::= {... }
```

PolicyInformation 类型的值标识并传送一个证书策略的限定符信息。成分 PolicyIdentifier 包含证书策略的标识符，成分 PolicyQualifiers 包含这一元素的策略限定符的值。

此扩展可以是关键的或非关键的，由证书颁发者选择。

如果此扩展标记为关键的，它表示证书应只用于此用途，并符合所指明的证书策略之一所包含的规则。特定策略的规划可以要求证书使用系统用特定的方法处理此限定符的值。

如果此扩展标记为非关键的，则此扩展的使用不必将证书的使用限于所列的策略。然而，证书用户

国家图书馆专用

为了使用此证书可以要求提供特定的策略(见第 10 章)。可以处理或不理睬策略限定符,由证书用户选择。

证书策略和证书策略限定符类型可由有此需要的任何组织定义。应按照 CCITT Rec. x. 660\ISO/IEC 9834-1 分配用来标识证书策略和证书策略限定符类型的客体标识符。一个 CA 可以使用 anyPolicy 标识符声明任何策略,使所有可能的策略信任一张证书。因为不论是在应用程序还是环境中,都需要对这个特定值进行认证,所以将对象标识符在本部分中进行了分配。本部分中,无对象标识符将分配用于特定的证书策略,此分配是定义证书策略实体的责任。

anyPolicy **OBJECT IDENTIFIER** ::= {2 5 29 32 0}

标识符 **anyPolicy** 没有任何联系的策略限定词。

下面的 ASN.1 对象类用来定义证书策略限定词类型:

```
CERT-POLICY-QUALIFIER ::= CLASS {
    &id             OBJECT IDENTIFIER UNIQUE,
    &Qualifier      OPTIONAL }
WITH SYNTAX {
    POLICY-QUALIFIER-ID             &id
    [QUALIFIER-TYPE     &Qualifier] }
```

策略限定符类型的定义应包括:

——可能值的语义的语句,和

——指示限定符的标识符是否可以没有伴随值而出现在证书策略扩展中,以及如果是这样,也包含隐含的语义。

注:可以将限定符规定为具有任何 ASN.1 类型。当预料到此限定符主要由没有 ASN.1 解码功能的应用使用时,推荐规定类型 OCTET STRING。因此,ASN.1 OCTET STRING 值能够按照策略元素定义组织所规定惯例运送限定符的编码值。

8.2.2.7 策略映射扩展

此字段应只用于 CA 证书,(它允许证书颁发者为包含这一证书的认证路径的用户指明颁发者证书策略之一可被认为与用于主体 CA 域的不同的证书策略等价。)此字段定义如下:

```
policyMappings EXTENSION ::= {
    SYNTAX                  PolicyMappingsSyntax
    IDENTIFIED BY           id-ce-policyMappings }

PolicyMappingsSyntax ::= SEQUENCE SIZE (1..MAX) OF SEQUENCE {
    issuerDomainPolicy      CertPolicyId,
    subjectDomainPolicy     CertPolicyId }
```

issuerDomainPolicy 成分指明在颁发的 CA 域内能够认可的证书策略并且可被认为与 **subjectDomainPolicy** 成分中所指明的在主体 CA 域能够认可的证书策略等价。

策略不会被映射到或来自特殊的值 **anyPolicy**。

证书发布者可以将该扩展选择为关键或非关键。此处推荐为关键,否则一个证书用户就不能正确解释发布的 CA 的规定。

注 1:政策映射的一个例子如下:美国政府可有一个称之为加拿大贸易的政策,加拿大政府可有一个称之为美国贸易的政策。当两个政策可有区别地被标识并被定义时,两国政府之间可有个协定:就相关的用途,在两个政策所隐含的规则之内,允许认证路径延伸过境。

注 2:政策映射意味着作出有关决策时会耗费显著的管理开销和涉及相当大的劳动和委任人员。一般而言,最好的办法是同意使用比应用政策映射更广的全球的公共政策。在上述例子中,美国、加拿大和墨西哥同意一项

国家图书馆专用

公共政策,用于北美贸易那将是最好的。

注 3:预计政策映射实际上只能用于政策声明非常简单的有限环境。

8.3 主体和颁发者信息扩展

8.3.1 要求

下列要求与证书主体和证书颁发者属性有关:

a) 证书必须是应用可用的,这些应用是采用各种名称形式的,包括 Internet 电子邮件名称,Internet 域名,X.400 始发者/接收者地址和 EDI 一方的名称。因此,必须能够使各种名称形式的多个名称与证书主体或证书颁发者或 CRL 颁发者安全地联系起来。

b) 证书用户可能需要安全地知道有关主体的某些标识的信息,以确信主体的确是预期的人或事。例如,可以要求诸如邮政地址,在公司中的地位或图片图像这样的信息。这种信息可以方便地表示成目录属性,但是,这些属性不是可辨别名(称)的必要部分。因此,为运送超过可辨别名(称)中的目录属性的附加目录属性而需要证书字段。

8.3.2 证书和 CRL 扩展字段

定义下列扩展字段:

a) 主体可替换名称;

b) 颁发者可替换名称;

c) 主体目录属性。

这些字段(除颁发者可替换名称外)应只用作证书扩展,而颁发者可替换名称还可用作 CRL 扩展。作为证书扩展,它们可以存在于 CA 证书或端实体证书之中。

8.3.2.1 主体可替换名称扩展

此字段包含一个或多个可替换名称(使用的各种名称形式的任一个),以供通过 CA 与认证的公钥所连接的实体使用。此字段定义如下:

```
subjectAltName EXTENSION ::= {
    SYNTAX          GeneralNames
    IDENTIFIEDBY    id-ce-subjectAltName }

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

GeneralName ::= CHOICE {
    otherName                   [0]     INSTANCE OF OTHER-NAME,
    rfc822Name                  [1]     IA5String,
    dNSName                     [2]     IA5String,
    x400Address                 [3]     ORAddress,
    directoryName               [4]     Name,
    ediPartyName                [5]     EDIPartyName,
    uniformResourceIdentifier   [6]     IA5String,
    iPAddress                   [7]     OCTET STRING,
    registeredID                [8]     OBJECT IDENTIFIER }

OTHER-NAME ::= TYPE-IDENTIFIER

EDIPartyName ::= SEQUENCE {
    nameAssigner                [0]     DirectoryString {ub-name }OPTIONAL,
```

国家图书馆专用

partyName [1] **DirectoryString** {**ub-name** }}

GeneralName 类型中可替换的值是下列各种形式的名称：

——otherName 是按照 OTHER-NAME 信息客体类别实例定义的任一种形式的名称；

——rfc822Name 是按照 Internet RFC822 定义的 Internet 电子邮件地址；

——dNSName 是按照 RFC 1035 定义的 Internet 域名；

——x400Address 是按照 ISO/IEC 10021-4 定义的 O/R 地址；

——directoryName 是按照 ISO/IEC 9594-2 定义的目录名称；

——ediPartyName 是通信的电子数据交换双方之间商定的形式名称；nameAssigner 成分标识分配 partyName 中唯一名称值的机构；

——uniformResourceIdentifier 是按照 Internet RFC1630 定义的用于 WWW 的 UniformR[A]esourceIdentifier；

——iPAddress 是按照 Internet RFC791 定义的用二进制串表示的 Internet Protocol 地址；

——registeredID 是按照 ISO/IEC 9834-1 分配的任何所登记客体的标识符。

对 GenersalName 类型中使用的每个名称形式，应有一个名称登记系统，以保证所使用的任何名称能向证书颁发者和证书使用者无二义地标识一个实体。

此扩展可以是关键的或非关键的，由证书颁发者选择。不要求支持此扩展的实现能处理所有名称形式。如果此扩展标记为关键的，那么，至少应能识别和处理存在的名称形式之一，否则，应认为此证书无效。除先前的限制以外，允许证书使用系统不理睬具有不能识别的或不被支持的名称形式的任何名称。倘若，证书的主体字段包含无二义地标识主体的目录名称，推荐将此字段标记为非关键的。

注 1：TYPE-IDENTIFIER 类别的使用在 ISO/IEC 8824-2 的附录 A 和附录 C 中描述。

注 2：如果存在此扩展并标记为关键的，证书的 subject 字段可以包含空名称（例如，相应可辨别名称的一系列“0”），在此情况下，主体只能用此扩展中的名称或一些扩展名称来标识。

8.3.2.2 颁发者可替换名称扩展

此字段包含一个或多个可替换名称（使用各种名称形式的任一个）以供证书或 CRL 颁发者使用。此字段定义如下：

issuerAltNameEXTENSION ::= {

SYNTAX **GeneralNames**

IDENTIFIED BY **id-ce-issuerAltName** }

此扩展可以是关键的或非关键的，由证书或 CRL 颁发者选择。不要求支持此扩展的实现能处理所有名称形式。如果此扩展标记为关键的，那么至少应能识别和处理存在的名称形式之一，否则，应认为此证书无效。除先前的限制以外，允许证书使用系统不理睬具有不能识别的或不支持的名称形式的任何名称。倘若，证书或 CRL 的颁发者字段包含无二义地标识颁发机构的目录名称，推荐将此字段标记为非关键的。

注：如果存在此扩展，并标记为关键的，证书或 CRL 的 issuer 字段可以包含空名称（例如，相应可辨别名称的一系列“0”），在此情况下，颁发者只能用此扩展中的名称或一些名称来标识。

8.3.2.3 主体目录属性扩展

此字段为证书主体运送证书主体希望的任何目录属性值。此字段定义如下：

subjectDirectoryAttributes EXTENSION ::= {

SYNTAX **AttributesSyntax**

IDENTIFIED BY **id-ce-subjectDirectoryAttributes** }

AttributesSyntax ::= **SEQUENCE SIZE** (1..**MAX**) **OF Attribute**

该扩展总是非关键的。

如果此扩展存在于一个公钥证书中,则在15章中将提出定义的一些扩展。

8.4 认证路径限制扩展

8.4.1 要求

对认证路径处理:

a) 端实体证书要与CA证书相区别,以防止端实体在未授权时将自己建成CA。CA要有可能限制由认证的主体CA产生的后继链的长度,例如,限制只是一个证书,或只是两个证书。

b) CA要能够规定一些限制,这些限制允许证书用户通过颁发证书给不合适的名称空间的主体来检查在认证路径的若干不太受信任的CA有没有违背它们的信任。(即,这些CA从具有证书用户启动的公钥的CA的认证路径进一步下行。)遵守这些限制需要由证书用户可自动检索。

c) 认证路径处理需要能用自动的、自包含的模块实现。允许受信任的硬件或软件模块执行认证路径处理功能是必要的。

d) 在无需依赖与本地用户实时交互的情况下,实现认证路径处理应是可能的。

e) 在无需依赖使用受信任的本地策略描述信息数据库的情况下,实现认证路径处理应是可能的。(需要某些受信任的本地信息—至少是初始公钥—用于认证路径处理,但是这些信息数量应该尽量少。)

f) 认证路径需要在认可的多证书策略的环境中操作。CA需要能够限定它信任的其他域的那些CA,以及用于何种用途。需要支持贯穿多策略域的链接。

g) 要求在信任的模型中有完全的灵活性。在考虑会有与多个企业互连的需要时,对单个组织是唯一的严格的体系结构模型是不够的。在选择认证路径中第一个受信任的CA时要求灵活性。特别是,要求认证路径在公钥用户系统的本地安全域内启动是可能的。

h) 应不需要使用证书中的名称来限制命名结构,即,所考虑的目录名称结构自然用于一些组织或一些地理区域应不需要调整,而适应证书认证机构的要求。

i) 就像ISO/IEC 9594-8以前版本的规定,证书扩展字段需要与不限制的认证路径进入系统后相兼容。

j) CA需要能够禁止使用策略映射和要求在认证路径中后继证书内存在显式证书策略标识符。

注:在任何证书使用系统中,认证路径的处理要求合适的保证级别。本目录标准定义了可用于要求遵守特定保证声明的实现的函数。例如,保证要求要说明认证路径处理必须防止破坏这种处理(例如,软件篡改或数据修改)。保证级别应与事务风险适配。例如

——对用于确认大量资金转移的公钥,可要求在合适的密码模块内部处理,反之,

——对家庭银行营业平衡查询,用软件处理可能是合适的。

因此,认证路径处理功能应适合于按硬件密码模块或密码令牌而选择来实现。

k) CA需要能够阻止在认证路径中的特殊的任一价值策略被认为是合法的策略。

8.4.2 证书扩展字段

定义下列扩展字段:

a) *基础约束*:

b) *名称约束*:

c) *策略约束*:

d) *约束所有策略*:

这些扩展字段应只用作证书扩展项。名称限制和策略限制应只用于CA证书;基本限制也可用于端实体证书。在附录L中给出这些扩展使用的例子。

8.4.2.1 基本限制扩展

在已认证的公钥用来检验证书签名时,此字段指示此主体是否可以起CA作用。如果是,还应规定

认证路径长度限制。此字段定义如下：

```
basicConstraints EXTENSION ::= {
    SYNTAX              BasicConstraintsSyntax
    IDENTIFIED BY       id-ce-basicConstraints }

BasicConstraintsSyntax ::= SEQUENCE {
    cA                  BOOLEAN DEFAULT FALSE,
    pathLenConstraint   INTEGER (0..MAX) OPTIONAL }
```

cA 字段标示此公钥证书是否可用来检验证书签名。

pathLenConslraint 字段应仅在 CA 置为真时存在。它给出此证书之后认证路径中最多的 CA 证书数目。O 值指示此密钥对只可以向终端实体颁发证书，而不可以颁发下级 CA 证书。如果在认证路径的任何证书中未出现 pathLenConstraint 字段，则对认证路径的允许长度没有限制。

此扩展由证书颁发者选择是必须项或是可选项。强烈推荐将它标记为必须项，否则，未被授权为 CA 的实体可以颁发证书，同时证书使用系统会在不知情的情况下使用这样的证书。

如果此扩展存在，并标记为必须项的，那么：

如果 CA 字段的值置为假(FALSE)，那么其公钥应不能用来验证证书签名；如果 CA 字段的值置为真(TRUE)并且 pathLen Constraint 存在，那么，证书使用系统应检查被处理的认证路径是否与 pathLenConstraint 的值一致。

注 1：如果此扩展不存在或标记为可选项的并且未被证书使用系统认可，该证书被系统视为终端用户证书，并且不能用来验证证书签名。

注 2：为限制一证书主体只是一个端实体，即，不是 CA，颁发者可以在扩展中只包含一个空 SEQUENCE 值的扩展字段。

8.4.2.2 名称限制扩展

此字段应只用于 CA 证书，它指示一个名称空间，认证路径中后续证书中的所有主体名称必须位于此空间。此字段定义如下：

```
nameConstraints EXTENSION ::= {
    SYNTAX              NameConstraintsSyntax
    IDENTIFIED BY       id-ce-nameConstraints }

NameConstraintsSyntax ::= SEQUENCE {
    permittedSubtrees   [0]   GeneralSubtrees OPTIONAL,
    excludedSubtrees    [1]   GeneralSubtrees OPTIONAL }

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

GeneralSubtree ::= SEQUENCE {
    base                          GeneralName,
    minimum             [0]       BaseDistance DEFAULT 0,
    maximum             [1]       BaseDistance OPTIONAL }

BaseDistance ::= INTEGER (0..MAX)
```

如果存在 permittedSubtrees 和 excludedSubtrees 字段，则他们每个都规定一个或多个命名子树，每个由此子树的根的名称或以处于其子树内的任意节点名称来定义。子树范围是一个由上界和/或下

国家图书馆专用

界限定的区域。如果 permittedSubtrees 存在,由主体 CA 和认证路径中下级 CA 颁发的所有证书中,只有那些在子树中具有与 permittedSubtrees 字段规定主体名称相同的证书才是可接受的。如果 excludedSubtrees 存在,由主体 CA 或认证路径中后继的 CA 颁发的所有证书中,同 excludedSubtrees 规定主体名称相同的任何证书都是不可接受的。如果 PermittedSutrees 和 excluded Subtrees 都存在并且名称空间重叠,则优选采用排斥声明(exclusion statement)。

通过 GeneralName 字段定义的命名格式,需要那些具有良好定义的分层结构的名称形式用于这些字段,Directory Name 名称形式满足这种要求;使用这些命名格式命名的子树对应于 DIT 子树。在应用中不需要检查和识别所有可能的命名格式。如果此扩展标记为必选项,并且证书使用中不能识别用于 base 字段的命名格式,应同遇到未识别的必选项扩展那样来处理此证书。如果此扩展标记为可选项,并且证书使用中不能识别用于 base 字段的命名格式,那么,可以不理睬此子树规范。当证书主体具有同一名称形式的多个名称时(在 directoryName 名称形式情况下,包括证书主体字段中的名称,如果非"0"),那么,对与此名称形式的名称限制一致性而言应测试所有这些名称。

注:当对与命名各式限制的一致性测试证书主体名称时,即使扩展中标示为可选项也应予以处理。

Minimum 字段规定了子树内这一区域的上边界。最后的命名形式在规定的级别之上的所有名称不包含在此区域内。等于"0"(默认)的 minimum 值对应于此基部(base),即,子树的顶节点。例如,如果 minimum 置为"1",那么,命名子树不包含根节点而只包含下级节点。

Maximum 字段规定了子树内这一区域的下边界。最后的命名形式在规定的级别之下所有名称不包含在此区域内。最大值"0"对应于此基部(base),即,子树的顶。不存在 maximum 成分指示不应把下限值施加到子树内的此区域上。例如,如果 maximum 置为"1",那么,命名子树不包含除子树根节点及其直接下级外的所有节点。

此扩展由证书颁发者选择为必选项或可选项。强烈建议将它标记为必选项,否则,证书用户不能检验认证路径中的后续证书是否位于 CA 签发的所期望的命名域中。

如果此扩展存在,并标记为必选项,那么,证书用户系统应检验所处理的认证路径与此扩展中的值是否一致。

8.4.2.3 策略限制扩展

此字段规定可以要求显式的证书策略标识或禁止对认证路径其余部分的策略映射的限制。此字段定义如下:

```
policyConstraints EXTENSION ::= {
    SYNTAX                  PolicyConstraintsSyntax
    IDENTIFIED BY           id-ce-policyConstraints }

PolicyConstraintsSyntax ::= SEQUENCE {
    requireExplicitPolicy   [0]   SkipCerts OPTIONAL,
    inhibitPolicyMapping    [1]   SkipCerts OPTIONAL }

SkipCerts ::= INTEGER (0..MAX)
```

如果 requireExplicitPolicy 字段存在,并且证书路径包含一个由指定 CA 签发的证书,所有在此路径中的证书都有必要在证书扩展项中包含合适的策略标识符。合适的策略标识符是有用户在证书策略中定义的标示符,或声明通过策略映射与其等价的策略的标识符。指定的 CA 指包含此扩展信息的证书认证机构(如果 requireExplicitpolicy 的为"0")或是认证路径中后续证书认证机构 CA(由非"0"值指示的)。

如果 inhibitploicyMapping 字段存在,它表明在认证路径中从所指定的 CA 开始直到认证路径结束为止的所有证书中,不允许策略映射。指定的 CA 指包含此扩展信息的证书认证机构(如果 inhibitPoli-

cyMapping 的值为“0”)或是认证路径中后续证书认证机构 CA(由非“0”值指示的)。

SkipCerts 类型的值表示在某一限制成为有效之前应在认证路径中需要跳过的证书的个数。

此扩展由证书颁发者选择是必选项还是可选项。强烈建议它标记为必选项,否则证书用户可能不能正确地解释证书认证机构 CA 设定的规则。

8.4.2.4 限制所有策略扩展

该扩展指定了一个限制,它指出了任何策略,对于从指定 CA 开始的认证路径中的所有证书的证书策略,都认为不是显式匹配。指定的 CA 要么是包含这个扩展的证书的主体 CA(如果 **inhitanyPolicy** 值为 0),要么是认证路径(由非 0 值指定)中后继证书认证机构 CA。

inhibitAnyPolicy	**EXTENSION ::= {**
SYNTAX	**SkipCerts**
IDENTIFIED BY	**id-ce-inhibitAnyPolicy }**

此扩展由证书颁发者选择必选项还是可选项。建议它标记为必选项,否则证书用户可能不能正确地解释证书认证机构 CA 设定的规则。

8.5 基本 CRL 扩展

8.5.1 要求

下列要求与 CRL 有关:

a) 证书用户要能够跟踪由 CRL 颁发者或 CRL 分布点颁发的所有 CRL(见 8.6)并要能够按顺序检测丢失的 CRL。因此要求 CRL 含有顺序号。

b) 一些 CRL 用户可能根据项撤销的原因不同而希望对此撤销进行不同的响应。因此对 CRL 项要有指明撤销原因的要求。

c) 对 CA 要能够暂时中止证书有效性的要求,随后要么撤销要么恢复此证书。这种动作的可能原因包括:

——当撤销的请求未予授权以及决定它是否有效的信息不充分时,想要减少错误撤销的责任;

——其他事务需要,例如在审计或调查期间暂时停止使用一个实体的证书。

d) 对每个撤销的证书,CRL 包含 CA 撤销该证书的日期。知道当显式的或潜在的密钥泄密威胁系统安全时的进一步信息可能对证书用户是非常有价值的。撤销日期不足以解决某些争议,因为在证书有效期间颁发的所有签名必须被认为无效(假设最坏情况)。然而,即使在产生签名以后用来签名此报文的密钥被泄密,签名的文件被认为有效对用户来说可能是重要的。为了协助解决这种问题,CRL 项可以包括指示已知或怀疑私钥被泄密的第二个日期。

e) 证书的使用者应该能够从 CRL 本身确定附加的信息,包括被列表覆盖的证书范围,撤销通知的顺序,一个 CRL 序列数唯一存在于那个 CRL 流中。

f) 颁布者需要动态改变 CRL 分割,以及具有分割改变时,并指导证书用户到相关的 CRL 的新位置查询的能力。

g) 更新一个给定的基于 CRL 的 DeltaCRL 可能是有效的。证书的使用者需要从一个给定的 CRL 中来确定一个 DeltaCRL 是否可用,它们在那里,以及发布下一个 DeltaCRL 的确切时间。

8.5.2 CRL 和 CRL 项扩展字段

定义下列扩展字段:

a) *CRL 证书号*;

b) *撤消原因*;

c) *保留指令代码*;

d) *无效日期*。

e) *CRL 范围*;

f) *状态介绍*；

g) *CRL 流标识符*；

h) *顺序表*；

i) *增量信息*。

CRL 数,CRL 范围,状态介绍,CRL 流标识符,顺序表和 delta 信息仅仅用于 CRL 扩展域,其他的域仅用作 CRL 实体扩展域。

8.5.2.1 CRL 号扩展

此 CRL 扩展字段为一给定的 CRL 颁发者通过一给定的 CA 目录属性或 CRL 分布点为每份 CRL 所颁发的单调递增的顺序号。它使得 CRL 用户能检测到正在处理的 CRL 之前颁发的一些 CRL 是否还可查看和处理。此字段定义如下：

```
cRLNumber EXTENSION ::= {
  SYNTAX              CRLNumber
  IDENTIFIED BY       id-ce-cRLNumber }
CRLNumber ::= INTEGER (0..MAX)
```

这个扩展是非关键的。

8.5.2.2 原因代码扩展

此 CRL 项扩展字段标识证书撤销的原因。可由根据本地策略决定对所通告的撤销如何反应的应用使用此原因代码。此字段定义如下：

```
reasonCode EXTENSION ::= {
    SYNTAX              CRLReason
    IDENTIFIED BY       id-ce-reasonCode }

CRLReason ::= ENUMERATED {
    unspecified             (0),
    keyCompromise           (1),
    cACompromise            (2),
    affiliationChanged      (3),
    superseded              (4),
    cessationOfOperation    (5),
    certificateHold         (6),
    removeFromCRL           (8),
    privilegeWithdrawn      (9),
    aACompromise            (10) }
```

下列原因代码值指示证书撤销的原因：

a) keyCompromise 用于撤销端实体证书;它表明证书中主体的密钥或其他方面确认或被怀疑已经被泄密；

b) cACompromise 用于撤销 CA 证书;它表明证书中主体或其他方面确认或被怀疑已经被泄密；

c) affiliationChanged 表明证书中的主体名称或其他信息已被修改,但是私钥仍被认为安全；

d) superseded 指示证书已被更新,但是私钥仍然安全；

e) cessationOFOperation 指示需要此证书中由证书认证机构签发的用途无效,但是其私钥仍被认为安全；

f) privilegeWithdraw 表明了证书(公钥或者属性证书)被撤销,原因是因为证书中的一个特权已经被撤销。

国家图书馆专用

g) aACompromise 表明了在属性证书中已被验证的 AA 方面的泄密，成为了众所周知的或者被怀疑。

可以通过颁发具有 certificateHold 原因代码项的 CRL 并将证书放置在 hold 中。certificate hold 通知可以包括任选的保留指令代码以向证书用户运送附加的信息（见 8.5.2.3）。一旦签发保留器，可用下列三种方法之一来处理：

a) 无须进一步动作，它可保持在 CRL，使用户拒绝在保留期间处理；或，

b) 它可用同一证书的（最终）撤销所取代，在此情况中，原因应是撤销的标准原因之一，撤销日期应是此证书放置在保留器上的日期，并且不应出现任选的指令代码扩展字段；或

c) 它可以显式地释放并且从 CRL 取消此项。

RemoveFromCRL 原因代码仅以 Δ-CRL（见 8.6）方式使用，来表明由于证书期满或者保留发布的原因，存在的 CRL 项应被删除。带有这个原因代码的项应该用在 Δ-CRL 中，对于它相应的基本 CRL 或者任何后来的（delta 或者整个范围）CRL 包含带有原因代码 **certificateHold** 的同一证书的项。

这个扩展是可选项。

8.5.2.3 保留指令代码扩展

此 CRL 项扩展字段提供包含已登记的指令标识符，以指示遇到被保留的证书时要采取的动作。它只适用于具有 certificateHold 原因代码的项。此字段定义如下：

```
holdInstructionCode EXTENSION ::= {
    SYNTAX              HoldInstruction
    IDENTIFIED BY       id-ce-instructionCode }
HoldInstruction ::= OBJECT IDENTIFIER
```

此扩展总是可选项。在本目录标准中未定义标准的保留指令代码。

注：保留指令的例子可能是“请与 CA 联系”或“重新获得用户令牌”。

8.5.2.4 无效日期扩展

此 CRL 项扩展字段指明已知或怀疑私钥曾被泄密的日期，换句话说就是证书应被认为无效的日期。此日期可以先于 CRL 项中撤销的日期，撤销日期是 CA 处理此撤销的日期。此字段定义如下：

```
invalidityDate EXTENSION ::= {
    SYNTAX              GeneralizedTime
    IDENTIFIED BY       id-ce-invalidityDate }
```

这个扩展是非关键的。

注 1：该域中的日期，就它本身来说，对于不可抵赖的目的是不够的。例如：这个日期可能是私钥的持有者建议的一个日期，私钥的持有者为了否认一个合法产生的签名，可能欺骗性地声称在过去的某一个时间密钥被泄密。

注 2：当证书认证机构在 CRL 中第一次邮寄一个撤销的时候，无效的日期可能早于早期 CRL 的发布日期。撤销日期不应该比早期 CRL 的发布日期早。

8.5.2.5 CRL 范围扩展

CRL 范围在 CRL 中用以下的 CRL 扩展来表明。为了阻止一个 CRL 受到不支持范围扩展的应用的代替攻击，如果存在范围扩展的话，必须标识为关键的。

可用于各种 CRL 类型范围声明的扩展包括：

a) 提供被单一授权颁布的证书的撤销信息的单一 CRL；

b) 提供被多个授权颁布的证书的撤销信息的间接 CRL；

c) 更新以前颁布的撤销信息的 Δ-CRL；

d) 间接的 Δ-CRL，它提供撤销信息。撤销信息更新了由单一授权或多个授权颁布的多个基本 CRL。

国家图书馆专用

```
csrlcope EXTENSION ::= {
    SYNTAX              CRLcopeSyntax
    IDENTIFIED BY       id-ce-cRLcope }
cRLcopeSynta ::=        SEQUENCE SIZE (1..MAX) OF PerAuthorityScope
PerAuthorityScope ::= SEQUENCE {
    authorityName               [0]    GeneralNameOPTIONAL,
    distributionPoint           [1]    DistributionPointNameOPTIONAL,
    onlyContains                [2]    OnlyCertificateTypesOPTIONAL,
    onlySomeReasons             [4]    ReasonFlagsOPTIONAL,
    serialNumberRange           [5]    NumberRangeOPTIONAL,
    subjectKeyIdRange           [6]    NumberRangeOPTIONAL,
    nameSubtrees                [7]    GeneralNamesOPTIONAL,
    baseRevocationInfo          [9]    BaseRevocationInfoOPTIONAL
    }
OnlyCertificateTypes            ::= BIT STRING {
    user        (0),
    authority   (1),
    attribute   (2)}
NumberRange ::= SEQUENCE {
    startingNumber  [0]  INTEGER OPTIONAL,
    endingNumber    [1]  INTEGER OPTIONAL,
    modulus              INTEGER OPTIONAL }

BaseRevocationInfo ::= SEQUENCE {
    cRLtreamIdentifier  [0]  CRLtreamIdentifier  OPTIONAL,
cRLNumber               [1]  CRLNumber,
baseThisUpdate          [2]  GeneralizedTime }
```

如果 CRL 间接的为多个证书认证机构组织提供了包含撤销状态信息的 CRL，扩展将包括多个 **PerAuthorityScope** 结构。对于每个证书认证机构及其撤销信息，包含一个或多个结构。同发布该 CRL 的证书认证机构相关的 **PerAuthorityScope** 的每一个实例将包含 **authorityName** 组件。

如果 CRL 是一个 Δ-CRL，它提供由单一证书认证机构颁布的基于基本 CRL 的增量撤销状态信息，扩展将包括多个 **PerAuthotityScope** 结构，对于那些由 Δ-CRL 提供更新得到的基本 CRL，每一个都有一个 **PerAuthorityScope** 结构。尽管有 **PerAuthorityScope** 结构的多实例，如果存在的话，**authorityName** 组件的值将和所有实例的值相同。

如果 CRL 是一个间接的 Δ-CRL，它为多个基本 CRL 提供增量撤销状态信息，多个基本 CRL 由多种证书认证机构颁布，扩展将包括多 **PerAuthotityScope** 结构，对于那些由 Δ-CRL 提供更新得到的基本 CRL，每一个都有多个 **PerAuthorityScope** 结构。同授权相关的，不同于颁布的间接 Δ-CRL 的 **PerAuthorityScope** 的每一个实例将包括 **authorityName** 组件。

对于扩展中出现的每一 **PerAuthorityScope** 实例，该域用法如下。应该注意的是，在间接 CRL 和 Δ-CRL的情形下，每一个 **PerAuthorityScope** 实例将包括这些域的不同组合和不同值。

如果存在 **AuthorityName** 域，则确定了签发这份提供撤销信息的证书的证书认证机构。如果 **authorityName** 省略，它默认为 CRL 发布者名称。

国家图书馆专用

如果存在 **DistributionPoint** 域，其用法同 **issuingDistributionPoint** 扩展中所描述的一样。

如果存在 **OnlyContains** 域，它指出了 CRL 包含撤销状态信息所关联的证书类型。如果该域为空，则 CRL 包含所有证书类型的信息。

如果存在 **OnlySomeReasons** 域，其用法同 **issuingDistributionPoint** 扩展中所描述的一样。

如果存在 **SerialNumberRange** 元素，其用法如下。当一个模数值出现时，在检查是否存在于有效范围内之前，序列号用给定的模数值减少。因此，如果带有一个(减少的)序列号的证书符合下列条件，则被认为在 CRL 范围内：

a) 大于或者等于 **startingNumber**，小于 **endingNumber**，同时出现；或者，

b) 当 **endingNumber** 不出现的时候，大于或者等于 **startingNumber**；或者，

c) 当 startingNumber 不出现的时候，小于 endingNumber。

SubjectkeyidRange 元素，如果出现，解释同 **serialNumberRange**，除非所用的数值是证书 **subjectKeyidentifier** 扩展中的值。(忽略标识、长度和未用的位组的)**BIT STRING** 的 DER 编码被认为是一个 **INTEGER** 的 DER 编码值。如果 **BIT STRING** 的字节 0 设置，那么应该预先考虑一个附加的 0 位组，以确保结果编码代表一个正的 **INTEGER**。例如：

030201f7(代表字节 0-6 设置)

映射到

020200f7(也就是十进制 247)

如果存在 **namesubtrees** 域，使用 **nameconstraints** 扩展中指定的命名格式规定。

如果存在 **baseRevocationinfo** 域，则表明 CRL 是一个 Δ-CRL，相对于被 **perAuthorityscope** 结构覆盖的证书来说。认证一个 CRL 使一个 Δ-CRL 的 **crlScope** 扩展的用法不同于以下方式中 **deltaCRLIdentifer** 扩展的用法。对于 **crlScope** 的情形，**baseRevocationinfo** 组件中的信息，及时指出从更新包含此增量 CRL 信息的根 CRL 的节点信息。尽管这是通过应用一个 CRL 来完成的，所引用的 CRL 可能是或者不是可应用范围中的。但 **deltaCRLIdentifier** 扩展参考了颁布的 CRL，它对于可应用范围来说是完整的。然而，在一个包含 **crlScope** 扩展的 Δ-CRL 中提供的更新信息是根据对于应用范围是完整的撤销信息更新的，不考虑在 **baseRevocationinfo** 中引用的 CRL 是作为同一范围中的完整的来颁布的。次结构提供了比 **deltaCRLIndicator** 扩展更多的适应性，尽管用户可以构建完全的 CRL，以及构建是基于时间而不是基于基本 CRL 的保证，它对于可应用的范围来说是完整的。在这些情形下，Δ-CRL 在一个给定范围的特定上，为证书提供撤销状态的及时更新。但是，在 **deltaCRLIndicator** 的情形下，必须是这样的一个 CRL，它对于所颁布和引用的范围来说是完整的。在 **crlScope** 的情形下，该时间点引用颁布的 CRL，对于范围来说，CRL 可以是或者不是完整的。

依赖于证书认证机构定制的策略，多个 Δ-CRL 可以在一个新的基本 CRL 发布之前发布。Δ-CRL 包含 **crlScope** 扩展用来定义它们的创建点的，不一定需要在 **BaseRevocationInfo** 域中应用最近颁布的基本 CRL 的 **cRLNumber**。但是，在 Δ-CRL 的 **baseRevocationinfo** 域中应用的 **cRLNumber** 应该小于或者等于最近颁布该 CRL 对于可应用范围内的 CRL 的 **cRLNumber**。

注意 **issuingdistributionpoint** 扩展和 **crlScope** 扩展相互之间可能是矛盾的，也不需要同时使用。但是，如果一个 CRL 同时包含 **issuingDistributionpoint** 扩展和 **crlScope** 扩展，那么当且仅当证书满足两个扩展的标准，才能在 CRL 的这个范围内。如果 CRL 不包含 **issuingDistributionpoint** 扩展，也不包含 **crlScope** 扩展，则范围是整个授权的范围，那么 CRL 可用于这个证书认证机构所涵盖的所有证书。

当一个证书使用系统用包含 **crlScope** 扩展的 CRL 来检查证书状态的时候，它应该检查位于 CRL 所认可范围内(如 **crlScope** 扩展所定义的)的证书和原因代码，如下：

a) 证书使用系统应该检查位于这个范围内的证书，该范围由 **serialNumberRange**、**subjectkeyIdRange** 和 **nameSubtrees** 范围的交集指定，对于相关的 **perAuthorityScope** 结构来说，它和 **distributionPoint**、**onlyContains** 是一致的，如果它们存在的话。

b) 如果 CRL 在 **crlScope** 扩展中包括一个 **onlySomeReasons** 组件，证书使用系统必须检查被 CRL 覆盖的原因代码，这对于应用来说是足够的。如果包含 **onlySomeReasons** 组件，可能需要附加的 CRL 解决。注意如果一个 CRL 同时包含 **crlScope** 扩展和 **issuingDistributionPoint** 扩展，同时包含一个 **onlySomeReasons** 组件，则仅仅包含在两个扩展 **onlySomeReasons** 组件中的原因代码被 CRL 覆盖。

8.5.2.6 状态介绍扩展

状态介绍扩展在 CRL 结构中作为向证书用户传送有关撤销通知信息的一种方式来使用。同样地，它也可以出现在自身不包含证书撤销通知的 CRL 结构中。包含扩展的 CRL 结构不能被证书用户或作为撤销通知来源的依赖实体使用，而只是一种确保使用了合适的撤销信息的工具。

扩展提供两种基本功能：

a) 扩展提供了发布可信任“CRL 表”的机制，包括所有决定其是否含有足够的所需撤销信息中的辅助依赖实体的相关信息。例如，授权可以定期以比较高的重发布频率（与其他 CRL 重发布频率相比较）发布一个新的，已认证的 CRL 表。该表包括每个引用 CRL 的最近更新时间/日期。获得该表的证书用户能快速决定高速缓存的 CRL 副本是否仍然是最新的。这能消除许多无用的 CRL 查找。而且，通过使用这种机制，证书用户可以获知证书认证机构常规更新 CRL 的周期，从而提高 CRL 系统的时间性。

b) 扩展还提供了一个机制来重定向从初级域（例如，CRL 分布点扩展的一个特指，或发布授权的目录项）到不同域撤销信息的依赖实体。这种特性使得证书认证机构能修改所使用的 CRL 分区模式而不影响现有证书或证书用户。为能实现这个功能，证书认证机构应包括每个新位置和其 CRL 涵盖范围，以在使用时能确切找到该位置。依赖实体将证书重要性与声明范围进行比较，跟随对于确认证书功能相关的撤销信息的合适的新位置指针。

该扩展是自扩展的，并且在将来，基于撤销模式的其他非 CRL 也会通过使用该扩展而被引用。

```
StatusReferrals EXTENSION ::= {
    SYNTAX              StatusReferrals
    IDENTIFIED BY       id-ce-statusReferrals }
StatusReferrals ::= SEQUENCE SIZE (1..MAX) OF StatusReferral
StatusReferral ::= CHOICE {
    cRLReferral         [0]  CRLReferral,
    otherReferral       [1]  INSTANCE OF OTHER-REFERRAL }
CRLReferral ::= SEQUENCE {
    issuer              [0]  GeneralName OPTIONAL,
    location            [1]  GeneralName OPTIONAL,
    deltaRefInfo        [2]  DeltaRefInfo OPTIONAL,
    CRLcope                  CRLScopeSyntax,
    lastUpdate          [3]  GeneralizedTime OPTIONAL,
    lastChangedCRL      [4]  GeneralizedTime OPTIONAL }
DeltaRefInfo::=SEQUENCE {
    deltaLocation       GeneralName,
    lastDelta           GeneralizedTime OPTIONAL }
OTHER-REFERRAL ::= TYPE-IDENTIFIER
```

issuer 域认证标识 CRL 的实体；默认值是签发此 CRL 的证书认证机构名称。

location 域提供查询所指向的位置，默认值是签发此 CRL 的证书认证机构名称。

deltaRefInfo 域提供了一种在可以获得增量撤消信息和任选增量撤消信息的位置二选一的功能。

cRLScope 域提供能在被引用位置找到的 CRL 涵盖范围。

lastUpdate 域的值即为在最近发布的 CRL 中的 **thisUpdate** 域的值。

lastChangedCRL 在最近发布的修改了的 CRL 中的 **thisUpdate** 域的值。

OTHER-REFERRAL 提供了可以在将来提供的基于撤销模式的其他非 CRL 可扩展性。

该扩展总是被标记为必选项，以确保包含了该扩展的 CRL 不会被证书使用系统作为证书撤消状态信息源使用。

如果该扩展存在，并且得到证书使用系统的公认，则该系统不会使用 CRL 作为撤销状态信息资源。系统要么使用包含在该扩展中的信息，要么使用本部分范围以外的其他方式来查找适当的撤销状态信息。

如果扩展存在，但没有得到证书使用系统的公认，则该系统将不使用 CRL 作为撤销状态信息资源。系统将使用本部分范围以外的其他方式来查找适当的撤销信息。

8.5.2.7 CRL 流标识符扩展

CRL 流标识符域用于认证唯一的 CRL 编号中的范围。

```
CRLStreamIdentifier EXTENSION        ::= {
    SYNTAX                  CRLStreamIdentifier
    IDENTIFIED BY           id-ce-CRLStreamIdentifier }
CRLStreamIdentifier ::= INTEGER (0..MAX)
```

该扩展常作为可选项使用。

每个证书认证机构的该扩展的每个值必须是唯一的。不管 CRL 是何种类型，CRL 流标识符与 CRL 编号服务组合成一个唯一的由任意给定证书认证机构发布的 CRL 标识符。

8.5.2.8 顺序表扩展

顺序表扩展指出了 CRL 中 **revokedCertificates** 域中撤销证书列表的排列顺序是证书编号或撤销时间排列的增长序列。定义如下：

```
orderedList EXTENSION    ::= {
    SYNTAX              OrderedListSyntax
    IDENTIFIED BY       id-ce-orderedList }
OrderedListSyntax ::= ENUMERATED {
ascSerialNum        (0),
ascRevDate          (1) }
```

该扩展总是非关键的。

a) **ascSerialNum** 表示 CRL 中撤销证书的顺序是基于表中证书序列号排序，序列号是自每个项的 **serisNumber** 组件值获取；

b) **ascRevDate** 表示 CRL 中撤销证书的顺序是基于表中证书的撤消时间排序，时间是自每个项的 **revocationDate** 组件值获取。

如果 **orderedList** 不存在，则不提供关于顺序的信息，如果存在，撤销证书列表的 **orderedList** 在 CRL 中。

8.5.2.9 增量信息扩展

CRL 扩展在 CRL 中而不是在 Δ-CRL 中使用，且用于向信赖方表明某个增量 CRL 对包含该扩展的 CRL 仍然有效。该扩展提供了能够找到相关 Δ-CRL 的位置和选择发布下一个 Δ-CRL 的时间。

```
deltaInfo EXTENSION    ::=          {
SYNTAX                  DeltaInformation
IDENTIFIED BY           id-ce-deltaInfo }
DeltaInformation        ::=          SEQUENCE {
```

国家图书馆专用

deltaLocation **GeneralName,**

nextDelta **GeneralizedTime OPTIONAL }**

该扩展总是非关键的。

8.6 CRL 分布点和 Δ-CRL 扩展

8.6.1 需求

因为撤销列表有可能会越变越大,而且难以处理,所以,需要有描述局部 CRL 的能力,两种不同的处理 CRL 的执行类型,需要不同的解决方法。

第一种执行类型是在单独的工作站中,可能是在附加的加密硬件中。这些实现可能没有可信的存储能力,或者即使有,也只有有限的可信存储能力。因此,必须检查整个 CRL 来确定该 CRL 是否有效,然后查看证书是否有效。如果 CRL 很长,这种处理将是非常冗长的。这就需要对 CRL 分区来解决这个问题。

第二种执行类型是运用在处理大容量信息的高性能服务器上,例如,事务处理服务器。在这种环境中 CRL 通常作为后台任务处理,CRL 有效后,CRL 内容将局部地存储在加速审查的表示法中,例如,用一位来表明某张证书是否已被撤销。这种表示法在可信任存储中使用。对于大量的证书认证机构,这种服务器类型通常需要最新的 CRL。由于它已有一张先前撤销证书列表,因此只需要检索一张新近的撤销证书列表就可以了。该表被称为 Δ-CRL,与完整的 CRL 相比,要小一些,而且只需要检索和处理较少的资源。

以下需求与 CRL 分布点和 Δ-CRL 相关:

a) 为了控制 CRL 的大小,需要为一个授权发布的所有证书集合的各个子集分配不同的 CRL。这可以通过联合每张证书和 CRL 分布点来实现,分布点可以是
——一个目录项,如果它被撤销,其 CRL 属性将包含此证书的撤销项,或,
——一个位置,如电子邮件地址或可获得应用 CRL 的因特网的 URI。

b) 由于性能原因,当确认多证书时,例如证书路径,有必要降低需要审查的 CRL 数目。这可以通过使用一个 CRL 发布者签名和发布包含从多个证书认证机构撤销的 CRL 在内的 CRL 来实现。

c) 需要一个单独的 CRL 来记录已撤销的证书认证机构证书和已撤销的终端实体证书。这有助于已撤销的证书认证机构证书的 CRL 的证书路径的处理变得很短(通常为空)。为此目的,**authorityRevocationList** 和 **certificateRevocationList** 属性已指定。为保障分离的安全性,CRL 中需要一个指针来指明它是哪张表。否则,无法察觉到列表的非法置换。

d) 与包括所有常规绑定终端(当没有私钥误用的重大风险时)的状态相比,在存在潜在的泄密状态下(当有私钥误用的重大风险时),需要提供不同的 CRL。

e) 需要对局部 CRL(Δ-CRL)做规定。局部 CRL 只包含从基本 CRL 发布以来已被撤销的证书实体。

f) 对于 deltaCRL,当列表更新后,需要表明更新的日期/时间。

g) 需要在证书中表明什么地方可找到最新的 CRL。(例如,最近 delta)

8.6.2 CRL 分布点和 Δ-CRL 扩展域

定义了以下扩展域:

a) *CRL 分布点;*

b) *发布的分布点;*

c) *证书发布者;*

d) *Delta CRL 指针;*

e) *基本更新;*

f) *最新 CRL。*

国家图书馆样本

CRL 分布点和最新 CRL 仅作为证书扩展使用。发布的分布点,deltaCRL 指针和基本更新仅作为 CRL 扩展使用。证书发布者仅作为 CRL 项扩展使用。

8.6.2.1 CRL 分布点扩展

CRL 分布点扩展仅作为证书扩展使用,它可用于证书认证机构证书,终端实体公钥证书,以及属性证书中。该域指定了 CRL 分布点或表明将参考哪一个证书用户以确定证书是否已被撤销。证书用户能从可用分布点获得一个 CRL,或者它可以从证书认证机构目录项获得当前完整的 CRL。

该域定义如下:

```
cRLDistributionPoints EXTENSION ::= {
    SYNTAX                  CRLDistPointsSyntax
    IDENTIFIED BY           id-ce-cRLDistributionPoints }

CRLDistPointsSyntax ::= SEQUENCE SIZE (1..MAX) OF DistributionPoint

DistributionPoint ::= SEQUENCE {
    distributionPoint       [0]     DistributionPointName OPTIONAL,
    reasons                 [1]     ReasonFlags OPTIONAL,
    cRLIssuer               [2]     GeneralNames OPTIONAL }

DistributionPointName ::= CHOICE {
    fullName                                [0]     GeneralNames,
    nameRelativeToCRLIssuer                 [1]     RelativeDistinguishedName }

ReasonFlags ::= BIT STRING {
    unused                  (0),
    keyCompromise           (1),
    cACompromise            (2),
    affiliationChanged      (3),
    superseded              (4),
    cessationOfOperation    (5),
    certificateHold         (6),
    privilegeWithdrawn      (7),
    aACompromise            (8) }
```

distributionPoint 组件标识能够获得 CRL 的位置。如果此组件缺省,分布点名称默认为 CRL 颁发者的名称。

当使用 fullName 替代名称,或应用默认时,分布点名称可以有多种名称形式。同一名称(至少用其名称形式之一表示的)应存在于颁发 CRL 的分布点扩展的 distributionPoint 字段中。不要求证书使用系统能处理所有名称形式。它可以只处理分布点提供的诸多名称形式中的一种。如果不能处理某一分布点的任何名称形式,倘若能从另一个源,例如另一个分布点或 CA 目录项,获得必要的撤销信息,证书使用系统仍可使用此证书。

如果 CRL 分布点被赋于一个直接从属于 CRL 颁发者的目录名称,则只能使用 nameRelativeToCRLIssuer 组件。在此情况中,name Relative ToCRL Lssuer 组件运送与 CRL 颁发者目录名称有关的相对可辨别名(称)。

reasons 组件指明由此 CRL 所包含的撤销原因。如果没有 reasons 组件,相应的 CRL 分布点分发

国家图书馆专用

包含此证书(如果此证书已被撤销)的项的 CRL,而不管撤销原因。否则,reasons 值指明相应的 CRL 分布点所包含的那些撤销原因。

cRLIssuer 组件标识颁发和签署 CRL 的机构。如果没有此组件,CRL 颁发者的名称默认为证书颁发者的名称。

此扩展可以是关键的或非关键的,由证书颁发者选择,建议该扩展为非关键。

如果该扩展标记为关键,没有第一次检索和核对取自一个包含重要原因代码的指定分布点的 CRL 的证书的情况下,证书使用系统将不使用该证书。在分布点为所有撤销原因代码和由 CA(包括作为关键扩展的 **cRLDistributionPoint**)发布的所有证书分配 CRL 信息的域中,CA 不需要在 CA 项发布一个完整的 CRL。

如果此扩展标记为非关键的,而且证书使用系统未能识别此扩展字段类型,那么,只有在下列情况中,此系统只能使用此证书:

a) 它能从 CA 获得一份完整 CRL 并检查它(通过在 CRL 中设有颁发分布点扩展字段来指示最近的 CRL 是完整的);

b) 根据本地策略不要求撤销检查;或

c) 用其他手段完成撤销检查。

注 1:具有一个以上的 CRL 颁发者对一个证书颁发布 CRL 是可能的。这些 CRL 颁发者和颁发 CA 的协调是 CA 策略的一个方面。

注 2:每个原因代码的意义在本部分的 8.5.2.2 中的原因代码字段中定义。

8.6.2.2 颁发分布点扩展

此 CRL 扩展字段标识了这个特定 CRL 的 CRL 分布点,并标识了此 CRL 是否仅限于对终端实体证书、对 CA 证书,或对一组受限的原因的撤销。CRL 由 CRL 颁发者的密钥来签名——CRL 分布点没有它们自己的密钥对。然而,对通过目录所分布的 CRL 来说,此 CRL 存储在 CRL 分布点的项中,它可以不是 CRL 颁发者的目录项。如果该字段缺省,CRL 应包含所有由 CRL 颁发者发布的已撤销但未到期的证书。

该域定义如下:

```
issuingDistributionPoint EXTENSION ::= {
    SYNTAX              IssuingDistPointSyntax
    IDENTIFIED BY       id-ce-issuingDistributionPoint }
IssuingDistPointSyntax ::= SEQUENCE {
    distributionPoint                [0]  DistributionPointName OPTIONAL,
    onlyContainsUserCerts            [1]  BOOLEAN DEFAULT FALSE,
    onlyContainsAuthorityCerts       [2]  BOOLEAN DEFAULT FALSE,
    onlySomeReasons                  [3]  ReasonFlags OPTIONAL,
    indirectCRL                      [4]  BOOLEAN DEFAULT FALSE,
    onlyContainsAttributeCerts       [5]  BOOLEAN DEFAULT FALSE }
```

distributionPoint 组件包含用一种或多种名称形式表示的分布点名称。如果此字段缺省,此 CRL 应包含由 CRL 颁发者所颁发的所有已撤销的未期满的证书的各个项。当证书出现在 CRL 中,在它到期后,它将从随后的 CRL 中删除。

如果 **onlyContainsUserCerts** 为真,CRL 仅包含被撤销的终端实体证书。如果 **onlyContainsAuthorityCerts** 为真,CRL 仅包含被撤销的证书认证机构证书。如果 **onlySomeReasons** 为真,CRL 仅包含已识别一条或多条原因的撤销,否则 CRL 包含所有原因的撤销。

如果 **indirectCRL** 为真,那么此 CRL 可以包含来自证书认证机构而不是 CRL 颁发者的撤销通知。每一项负责的管理机构是按这一项中的证书颁发者的 CRL 项扩展。或按照 8.6.3.2 中描述的默认规

国家图书馆专用

则而指明的。在这种 CRL 中,保证 CRL 完整是此 CRL 颁发者的责任。因此,完整的 CRL 包含从标识其证书中 CRL 颁发者的所有管理机构来的所有撤销项,并与 **onlyContainsUserCerts** , **onlyContainsCACerts** , **onlySomeReasons** 标示符一致。

如果 **onlyContainsAttributeCerts** 为真,CRL 仅包含属性证书的撤销。

对通过目录分布的 CRL,适用以下的有关属性使用的规则。具有 onlyContainsCACerts 集合的 CRL 应通过有关的分布点的 authorityRevocationList 属性进行分布,或如果没有标识分布点则通过 CRL 颁发者项的 authorityRevocationList 属性来分布。否则,此 CRL 应通过有关的分布点的 certificateRevocationList 属性进行分布,或如果没有标识分布点,则通过授权项的 certificateRevocationList 属性来分布。

此扩展字段总是关键的。没有理解此扩展的证书用户不能假设此 CRL 包含了所标识的管理机构撤销的完整证书列表。不包含关键扩展的 CRL 必须包含对颁发管理机构的所有当前 CRL 项,包括对所有已撤销用户证书和管理机构证书的项。

注 1: CA 向 CRL 颁发者传递撤销信息的方法超出了本部分的范围。

注 2: 如果 CA 从自己的目录项(即,不是从单独命名的 CRL 分布点)颁发具有 onlyConstainsUserCerts 或 onlyConstainsAuthorityCerts 集合的 CRL,则 CA 应保证由此 CRL 包括的所有证书包含 basicConstraints 扩展。

8.6.2.3 证书颁发者扩展

此 CRL 项扩展标识与间接 CRL(即,在其分布点扩展中有 indirectCRL 标示符的 CRL)有关的证书颁发者。如果此扩展不出现在间接 CRL 的第一项中,证书颁发者默认为 CRL 颁发者。在间接 CRL 的后续项中,如果没有此扩展,此项的证书颁发者与先前项的证书颁发者相同。此字段定义如下:

certificateIssuer EXTENSION ::= {
    SYNTAX **GeneralNames**
    IDENTIFIED BY **id-ce-certificateIssuer** }

此扩展总是关键的。如果实现忽略此扩展,该实现不能正确地将 CRL 项与证书相对应。

8.6.2.4 DeltaCRL 指针扩展

deltaCRL 指针域作为一个对已引用的基本 CRL 提供更新的 deltaCRL(Δ-CRL)来确定 CRL。已引用的基本 CRL 是明确作为一个在给定范围内是完整的 CRL 而发布的 CRL。包含 deltaCRL 指针扩展的 CRL 包含了对相同范围内证书撤销状态的更新。该范围不需要包括所有撤销原因或所有由一个 CA 发布的证书,特别在该 CRL 是 CRL 分布点的情况下。然而,对于应用范围来说,在 Δ-CRL 发布的时候,包含 deltaCRL 指针扩展的 CRL,加上在 **BaseCRLNumber** 组件中引用的 CRL,相当于一个完整的 CRL。

该字段的定义如下:

deltaCRLIndicatorEXTENSION::={
    SYNTAX **BaseCRLNumber**
    IDENTIFIED BY **id-ce-deltaCRLIndicator** }

BaseCRLNumber ::= **CRLNumber**

BaseCRLNumber 的值标识了基本 CRL 的 CRL 数目,基本 CRL 作为 Δ-CRL 产生的基础。所引用的 CRL 是应用范围完整的 CRL。

该扩展通常是关键的。如果一个证书用户不理解 Δ-CRL,它将不会使用包含该扩展的 CRL,因为 CRL 可能不是用户期待的那么完整。

8.6.2.5 基本更新扩展

基本更新域在 Δ-CRL 中使用,并用来标识日期/时间,在这个日期/时间之后该 delta 对撤销状态提供更新。该扩展只用于包含 **deltaCRLIndicator** 扩展的 Δ-CRL 中。包含 **crlScope** 扩展的 Δ-CRL 不需要此扩展,**crlScope** 扩展的 **baseThisUpdate** 域也可以用于相同的用途。

baseUpdateTime EXTENSION ::= {
SYNTAX **GeneralizedTime**
IDENTIFIED BY **id-ce-baseUpdateTime** }

这个扩展通常是非关键。

8.6.2.6 最新的 CRL 扩展

最新 CRL 扩展只被作为证书扩展使用，或在发给证书认证机构和用户的证书中使用。该域标识了 CRL，对 CRL 来说证书用户应包含最新的撤销信息（例如：最新的 Δ-CRL）。

该域定义如下：

freshestCRLEXTENSION ::= {
SYNTAX **CRLDistPointsSyntax**
IDENTIFIEDBY **id-ce-freshestCRL** }

根据证书发布者的选择，这个扩展可能是关键的，也可能是非关键的。如果最新的 CRL 扩展是关键的，那么证书使用系统不使用没有进行第一次撤销和核对的最新 CRL 的证书。如果扩展被标记为不关键的，证书使用系统能使用本地方法来决定是否需要检查最新的 CRL。

9 Δ-CRL 与基础的关系

Δ-CRL 包含 **deltaCRLIndicator** 或 **crlScope** 扩展，用来表明由该 Δ-CRL 更新的基本撤销信息。

如果 **deltaCRLIndicator** 在 Δ-CRL 中存在，更新的基本撤销信息就是在扩展中引用的基本 CRL。由 **deltaCRLIndicator** 扩展引用的基本 CRL，应该是完全为其范围发布的 CRL（例如，不是 Δ-CRL 本身）。

如果 **crlScope** 扩展存在，并且包含 **baseRevocationInfo** 组件，则用来引用更新的基本撤销信息，这就涉及到来自 Δ-CRL 的及时特殊点来提供更新。**baseRevocationInfo** 组件引用了一个在或者不在其范围内完全发布的 CRL（例如，引用的 CRL 可能只作为 Δ-CRL 发布）。然而，包含 **baseRevocationInfo** 组件的 Δ-CRL 更新了撤销信息，该信息在引用的 CRL 发布时，对于引用的 CRL 的范围是完整的。证书使用者可以为 CRL 申请 Δ-CRL，其中 CRL 在给定范围内是完整的，并且在包含了 **baseRevocationInfo** 组件的 Δ-CRL 中的引用的 CRL 被发布的同时或之后发布。

由于可能存在不一致信息，CRL 不会同时包含 **deltaCRLIndicator** 扩展和带有 **baseRevocationInfo** 组件的 **crlScope** 扩展。只有在 **crlScope** 扩展中不存在 **baseRevocationInfo** 组件时，CRL 可能同时包含 **deltaCRLIndicator** 扩展和 **crlScope** 扩展。

Δ-CRL 也可能是一个间接的 CRL，在 CRL 中它可能包含与一个或多个证书认证机构发布的基本 CRL 有关的、更新的撤销信息。同间接 Δ-CRL 一样，**crlScope** 扩展也将作为标识 CRL 的一种方法。**crlScope** 扩展会为每个基本 CRL 包含一个 **PerAuthorityScope** 组件实例，同时间接 Δ-CRL 为每个基本 CRL 提供了更新信息。

将 Δ-CRL 应用到引用的基本撤销信息时，必须准确地反映撤销的当前状态。

——带有撤销原因 **certificateHold** 的一个证书撤销通知，可能出现在 Δ-CRL 中，也可能出现于在给定范围内是完整的 CRL 上。原因代码将指出证书的临时撤销，证书没有进一步决定是永久的废除证书还是作为一个没有废除的证书恢复它。

- 如果由于 CRL 上（Δ-CRL 或给定范围内的完整的 CRL），列出了带有撤销原因 **certificateHold** 的撤销证书，并且 CRL 的 **cRLNumber** 是 n，随后发布了所有权，则证书必须包含在发布了所有权后发布的所有 Δ-CRL 中，在这些 Δ-CRL 中，引用的基本 CRL 的 **crlScope** 必须小于或等于 n。使用的扩展指出该 CRL 是一个 Δ-CRL，被引用的基本 CRL 的 CRL 数目是 **deltaCRLIndicator** 扩展的 **BaseCRLNumber** 的组件值或 **cRLScope** 扩展的 **BaseRevocationInfo** 的组件 **CRLNumber** 元素。除非由于 Δ-CRL 中隐藏的撤销原因，使得证书再次被撤销

（在这种情况下，证书必须为了再次的撤销列出适当的理由），否则必须列出带有撤销原因 **removeFromCRL** 的证书。

- 如果证书不从所有权中删除，但被永久废除，那么它必须在所有后来的 Δ-CRL 上列出，被引用的基本 CRL 的 **cRLNumber** 要小于第一次出现永久撤销通知的 CRL（Δ-CRL 或给定范围内的完整的 CRL）的 **cRLNumber**。使用的扩展指出该 CRL 是一个 Δ-CRL，被引用的基本 CRL 的 CRL 数目是 **deltaCRLIndicator** 扩展的 **BaseCRLNumber** 的组件值或 **cRLScope** 扩展的 **BaseRevocationInfo** 组件的 **cRLNumber** 元素。

——证书的撤销通知可以第一次出现在 Δ-CRL 上，并且在可使用范围内的完整的下一个 CRL 发布前，证书的有效期可能终止。在这种情况下，撤销通知必须包含在所有后来的 Δ-CRL 中，直到撤销通知被包含在至少一个已发布的 CRL 上，CRL 在那个证书的范围内是完整的。

给定范围内完整的 CRL，可以在当前用下列任何一种方法进行局部构造：

——在该范围内重新检索当前 Δ-CRL，并且同已发布的该范围内完整的 CRL 以及大于或等于在 Δ-CRL 中引用的基本 CRL 的 **cRLNumber** 的 **cRLNumber** 相联合；或者，

——在该范围内重新检索当前 Δ-CRL，并且同局部构造的该范围内完整的 CRL 以及由 Δ-CRL 构造的 CRL 的联合，其中 Δ-CRL 的 **cRLNumber** 大于或等于在当前 Δ-CRL 中引用的基本 CRL 的 **cRLNumber**。

10 证书认证路径处理过程

证书认证路径的处理是在需要使用远程端实体的公钥的系统中进行，例如，验证由远程实体生成的数字签名的系统。证书策略，基本限制，名称限制和策略限制扩展是设计成能便于证书认证路径逻辑的自动化、自包含实现。

下面是确认证书认证路径用的规程概要。一种实现在功能上应等价于此规程产生的外部行为。由一给定的输入驱动正确的输出的特定实现所使用的算法尚未标准化。

10.1 路径处理的输入

证书认证路径处理过程的输入是：

a) 包含证书认证路径的一组证书；

注：一个证书认证路径中的每个证书是唯一的。包含两次或多次相同证书的路径不是有效的证书认证路径。

b) 受信任的公钥或密钥标识符（如果此密钥存储于证书认证路径处理模块的内部），用来验证证书认证路径中的第一份证书；

c) 包含一个或多个证书策略标识符的、指示这些策略的任一个策略对证书用户是可接受的、用于证书认证路径处理的 *initial-policy-ste*；此输入也可取特别的 *any-policy* 值；

d) *initial-explicit-policy* 标示符值，它指示可接受的策略标识符是否需要显式地出现在此路径中的所有证书的证书策略扩展字段中；

e) *initial-policy-mapping-inhibit* 标示符值，它指示策略映射在证书认证路径中是否是禁止的；

f) *initial-inhibit-policy* 标示符值，它指示特别的 *any-policy* 值如果存在于证书策略扩展中，是否考虑在一个约束集中的任一特别的证书策略值是否是匹配的；

g) 当前日期/时间（如果在证书认证路径处理模块的内部是不可使用的）。

c），d），e）和 f）的值将依靠使用者申请联合的策略需求，联合需要使用被认证的末端实体的公钥。

注：由于这些是路径确认过程的单输入，证书用户可能限制它设置的从任何给定可信公钥到给定证书策略集的信任。仅当 initial-policy-set 输入包含证书使用者所信任的公钥的策略，通过确保给定的公钥是处理的唯一的输入，证书用户才可以得到这个限制。同时，处理得到另一个输入是证书认证路径自身，该控制可以通过事务的基础来实施。

国家图书馆专用

10.2 路径处理的输出

过程的输出是：

a) 证书认证路径确认的成功或失败的指示；

b) 如果确认失败，就是一个指出失败原因的诊断代码；

c) 证书认证机构约束的策略集，以及与此集相联系的限定词，根据这些限定词，证书认证路径是有效的，或者特殊值 *any-policy*；

d) 用户约束的策略集，由 *authorities-constrained-policy-set* 和 *initial-policy-set* 的交集形成；

e) *explicit-policy-indicator*，指出证书用户或路径中的证书认证机构是否要求路径上的每个证书中的可接受策略是可认证的；

f) 处理证书认证路径中出现的任何策略映射的细节。

注：如果确认成功，此证书使用系统仍可按照证书的策略限制或者其他信息而选择不使用此证书。

10.3 路径处理的变量

本过程使用下列一组状态变量：

a) *authorities-constrained-policy-set*：证书认证路径中证书的策略标识符和限定词的表格（行表示策略，它们的限定词和映射历史，列表示证书认证路径中证书）；

b) *permitted-subtrees*：定义子树的子树规范集，证书路径中随后的证书的所有主体名必须在这个范围中，或采用特殊值 *unbounded* ；

c) *excluded-subtrees*：（可能为空）的定义子树的子树规范（每一个规范由子树基本名称和最大，最小等级指针组成）集，在证书路径中随后的证书里的无主体名中；

d) *explicit-policy-indicator*：指出合理的策略在路径中的每个证书中是否必须是可明确认证的；

e) *pathdepth*：等于或大于已经完成处理的证书认证路径中的证书数目的整数；

f) *policy-mapping-inhibit-indicator*：指出是否禁止策略映射；

g) *inhibit-any-policy-indicator*：指出对于任一特别证书策略是否考虑特殊值 any-policy 的匹配；

h) *pending-constraints*：显式策略和/或禁止策略映射约束的细节，它们已经被定义，但还没有生效。有三个一位指针，命名为 *explicit-policy-pending*、*policy-mapping-inhibit-pending* 和 *inhibit-any-policy-pending*，对于每一位指针，有称作 *skip-certificates* 的整数，它们给出了在约束生效之前，需要越过的证书数。

10.4 初始化步骤

该过程包括一个初始化步骤，随后是一序列的证书处理步骤。初始化步骤包含：

a) *authorities-constrained-policy-set* 表格的第 0 行的第零 0 列和第 1 列中写入 *any-policy*；

b) *permitted-subtrees* 变量初始化为 *unbounded*；

c) *excluded-subtrees* 变量初始化为空集；

d) *explicit-policy-indicator* 初始化为 *initial-explicit-policy* 值；

e) *path-depth* 初始化为 1；

f) *policy-mapping-inhibit-indicator* 初始化为 *initial-policy-mapping-inhibit* 值；

g) *inhibit-any-policy-indicator* 初始化为 *initial-inhibit-policy* 值；

h) 将三个 *pending-constains* 标示符初始化为复位。

10.5 证书处理

从使用输入的受信任的公钥签名的证书开始，依次处理每个证书。最近的证书将被认为是最后的证书；任何其他证书被认为是中间证书。

10.5.1 基本证书检查

下列检验适用于证书：

a) 检验签名验证，日期有效，证书主体和证书颁发者名称链的正确，以及证书未被撤销。

电子图书馆专用

b) 对中间证书，如果基本限制扩展字段存在于证书之中，检验 CA 组件存在，并置为真。如 pathLenConstraint 成分存在，检验当前的证书认证路径没有违背此限制。

c) 如果证书策略扩展不存在，那么通过从 *authorities-constrained-policy-set* 表格删除所有行，设置 *authorities-constrained-policy-set* 为空。

d) 如果证书策略扩展存在，对于每个策略 P，除了 anyPolicy 扩展外，*authorities-constrained-policy-set*[0，*path-depth*]中的值不是 *any-policy*，那么用证书中的策略设置 *authorities-constrained-policy-set* 的交集为 *authorities-constrained-policy-set*。为了做到这步，首先从扩展中把策略资格者加入到 *authorities-constrained-policy-set* 表格，对于扩展中的每个策略标识符的值，查找 *authorities-constrained-policy-set* 表格中所有行的位置，*authorities-constrained-policy-set* 表格的[*path-depth*]列实体包含它在扩展中相同的值，附上从扩展到表中策略标识符的策略资格者，然后删除[path-depth]列不包含扩展中的值的所有行。

e) 如果证书策略扩展存在，并且不包含 anyPolicy 的值，或者，如果 *inhabit-any-policy-indicator* 位被设置，那么，删除[*path-depth*]列中包含 *any-Policy* 的列，同时，删除[*path-depth*]列中不包含证书策略扩展的值的行。

f) 如果证书策略扩展存在，并且包含 anyPolicy 值，同时，*inhabit-any-policy-indicator* 位没有设置，则降策略限制与 anyPolicy 联合，置入 *authorities-constrained-policy-set* 表中那些[*path-depth*]列项中包含 *anyPolicy* 或者包含证书策略扩展中没有的值的行。

g) 如果证书不是中介自发布证书，检查主体名是否在由 *permitted-subtrees* 的值产生的命名空间内，并且该主体名不在 *excluded-subtrees* 的值产生的命名空间内。

10.5.2 处理中介证书

对中间证书，随后执行下列限制记录动作以便为下一个证书的处理建立状态变量：

a) 如果带有 permittedSubtrees 组件的 nameConstraints 扩展在证书中存在，设 *permitted-subtrees* 状态变量为它的先前值的交集，值在证书扩展中指出。

b) 如果带有 excludedSubtrees 组件的 nameConstraints 扩展在证书中存在，设 *excluded-subtrees* 状态变量为它的先前值的并集，值在证书扩展中指出。

c) 如果设置了 *policy-mapping-inhibit-indicator*：

——对于在扩展中认证的每一个映射，通过查找 *authorities-constrained-policy-set* 表格中[*path-depth*]列项等于扩展中发布者域中策略值的所有行，并删除，来处理策略映射扩展。

d) 如果没有设置 *policy-mapping-inhibit-indicator*：

——对扩展中的每个被认证的映射，通过查找 *authorities-constrained-policy-set* 表格中所有行的位置，*authorities-constrained-policy-set* 表格的[*path-depth*]列实体等于扩展中发布者范围的策略值，写入相同行的[*path-depth*＋1]列实体中来自扩展的主体范围的策略值。如果扩展绘制了一个发布范围的策略给超过一个的主体范围策略，那么受影响的行必须被复制，新的实体添加到每一行。如果 *authorities-constrained-policy-set*[0，*path-depth*]中的值是 *any-policy*，那么写入[path-depth]列中来自策略计划扩展的每个发布者范围的策略标识符，复制必需的行，如果标识符存在，保留的标识符，写入相同行的[*path-depth*＋1]列实体中来自扩展的主体范围的策略值。

——如果设置了 *policy-mapping-inhibit-pending* 指针，并且证书不是自发布的，减少相应的 *skip-certificates* 值，如果该值变为 0，设置 *policy-mapping-inhibit-indicator*。

——如果在证书中存在 *inhibitPolicyMapping* 约束，执行下面的操作。对于值为 0 的 SkipCerts，设置 *policy-mapping-inhibit-indicator*。对于任何其他 SkipCerts 值，设置 *policy-mapping-inhibit-pending* 指针，设置相应的 *skip-certificates* 值给较少的 SkipCerts 值和

先前的 *skip-certificates* 值(如果已经设置 *policy-mapping-inhibit-pending* 指针)。

e) 对于步骤 c)或 d)中没更改的任一行,以上(这种情况下的每一行,证书中没有映射扩展存在),写入行的[*path-depth*+1]列中来自[*path-depth*]列的策略标识符。

f) 如果没有设置 *inhibit-any-policy-indicator*

——如果设置了 *inhibit-any-policy-pending* 标识符并且证书不是自发布的,减少相应的 *skip-certificates* 值,如果该值变为 0,设置 *inhibit-any-policy-indicator*。

——如果在证书中存在 inhibitAnyPolicy 约束,执行下面的操作。对于值为 0 的 SkipCerts,设置 *inhibit-any-policy-indicator*。对于任何其他 SkipCerts 值,设置 *inhibit-any-policy-pending* 指针,设置相应的 *skip-certificates* 值给较少的 SkipCerts 值和先前的 *skip-certificates* 值(如果已经设置 *inhibit-any-policy-pending* 指针)。

g) 增加[*path-depth*]。

10.5.3 外在策略标识符处理

对于所有证书,执行以下行为:

a) 如果没有设置 *explicit-policy-indicator*:

——如果设置 *explicit-policy-pending* 指针,并且,证书不是自发布的中介证书,减少相应的 *skip-certificates* 值,如果该值变为 0,设置 *explicit-policy-indicator*。

——如果 requireExplicitPolicy 约束存在,执行下面的操作。对于值为 0 的 SkipCerts,设置 *explicit-policy-indicator*。对于任何其他 SkipCerts 值,设置 *explicit-policy-pending* 指针,设置相应的 *skip-certificates* 值给较少的 SkipCerts 值和先前的 *skip-certificates* 值(如果已经设置 *explicit-policy-pending* 指针)。

——如果 requireExplicitPolicy 组件存在,证书认证路径包含由推荐的 CA 发布的证书,对于路径中所包含的所有证书以及证书策略扩展和合理的策略标识符来说,证书认证路径是必需的。一个合理的策略标识符是证书路径中的用户要求的证书策略标识符,已宣布的策略标识符等同于它通过了策略映射或者 *any-policy*。推荐 CA 是包含该扩展的证书的任一发布者 CA(如果需要的 requireExplicitPolicy 是 0),或证书认证路径中的后来证书的主体 CA(作为一个非 0 值指出)。

10.5.4 最终处理

对于终端证书,执行以下行为:

a) 如果设置了 *explicit-policy-indicator*,检查 *authorities-constrained-policy-set* 表格不为空。如果以上任一个检查失败了,那么程序将终止,返回一个失败指令、一个适当的原因代码、*explicit-policy-indicator* 和在 *user-constrained-policy-set* 及 *authorities-constrained-policy-set* 表格中的 null 值。

b) 如果在终端证书中上述的检查没有失败,那么 *user-constrained-policy-set* 应该是通过 *authorities-constrained-policy-set* 和 *the initial-policy-set* 的交集计算出来的。如果 *authorities-constrained-policy-set*[0, *path-depth*]是 *any-policy*,那么,*authorities-constrained-policy-set* 是 *any-policy*. 否则, 对于表中的每一行,*authorities-constrained-policy-set* 的值是不包括标识符任何策略的最左面的单元。那么程序将终止,返回一个与 *explicit-policy-indicator*、*authorities-constrained-policy-set* 表和 *user-constrained-policy-set* 一起的成功指令。如果授权约束集合是 null,在授权约束策略下,路径是有效的,但是对于用户来说,没有一个是合理的。

11 PKI 目录模式

该条款定义了在目录中用来描绘 PKI 信息的目录模式基础。它包括相应的对象类,属性和匹配规则的属性值的规范。

11.1 PKI 目录对象类和命名形式

该部分包括目录中用来描述 PKI 对象的对象类的定义。

11.1.1 PKI 用户对象类

PKI 用户对象类用来定义公钥证书的主体的对象项。

```
pkiUser   OBJECT-CLASS                ::= {
    SUBCLASS OF                       {top }
    KIND                              auxiliary
    MAY CONTAIN                       {userCertificate }
    ID                                id-oc-pkiUser }
```

11.1.2 PKICA 对象类

PKICA 对象类用来定义担当证书认证机构的对象项。

```
pkiCA                                 OBJECT-CLASS    ::= {
    SUBCLASS OF                       {top }
    KIND                              auxiliary
    MAY CONTAIN                       {cACertificate |
                                      certificateRevocationList |
                                      authorityRevocationList |
                                      crossCertificatePair }
    ID                                id-oc-pkiCA }
```

11.1.3 CRL 分布点对象类和命名形式

CRL 分布点对象类用来定义担当 CRL 分布点的对象项。

```
cRLDistributionPoint                  OBJECT-CLASS    ::= {
    SUBCLASSOF                        top }
    KIND                              structural
    MUST CONTAIN                      {commonName }
    MAY CONTAIN                       {certificateRevocationLis t|
                                      authorityRevocationList |
                                      deltaRevocationList }
    ID                                id-oc-cRLDistributionPoint }
```

CRL 分布点命名形式指定对象类 cRLDistributionPoint 项是如何命名的。

```
cRLDistPtNameForm                 NAME-FORM ::= {
    NAMES                             cRLDistributionPoint
    WITH ATTRIBUTES                   {commonName }
    ID                                id-nf-cRLDistPtNameForm }
```

11.1.4 DeltaCRL 对象类

deltaCRL 对象类用来定义持有 delta 撤销列表(例如,CAs,AAs 等)的对象项。

```
deltaCRL                              OBJECT-CLASS    ::= {
    SUBCLASS OF                       {top }
    KIND                              auxiliary
    MAY CONTAIN                       {deltaRevocationList }
    ID                                id-oc-deltaCRL }
```

11.1.5 证书策略和 CPS 对象类

CPCPS 对象类用来定义包含证书策略和/或认证实践信息的对象项。

国家图书馆专用

```
cpCps                                   OBJECT-CLASS    ::= {
    SUBCLASSOF                          {top }
    KIND                                auxiliary
    MAY CONTAIN                         {certificatePolicy |
                                        certificationPracticeStmt }
    ID                                  id-oc-cpCps }
```

11.1.6 PKI 证书路径对象类

PKIcert 路径对象类用来定义包含 PKI 路径的对象项。它一般联合结构化的对象类 pkiCA 的项来使用。

```
pkiCertPath                             OBJECT-CLASS    ::= {
    SUBCLASS OF                         {top }
    KIND                                auxiliary
    MAY CONTAIN                         {pkiPath }
    ID                                  id-oc-pkiCertPath }
```

11.2 PKI 目录属性

这一部分包括在目录中的储存 PKI 信息元素的目录属性定义。

11.2.1 用户证书属性

一个用户可以从一个或者更多的 CA 机构获得一个或者更多的公钥证书。userCertificate 属性类型包含用户从一个或者更多的 CA 获得的公钥证书。

```
userCertificate                         ATTRIBUTE    ::=  {
    WITH SYNTAX                         Certificate
    EQUALITY MATCHING RULE              certificateExactMatch
    ID                                  id-at-userCertificate }
```

11.2.2 CA 证书属性

CA 目录项的 CACertificate 属性将用来储存自发布的和在与 CA 处在相同领域中由 CAs 颁发给该 CA 的证书。在 V3 证书的格式里，这些证书会包括一个 CA 值设置为 TRUE 的 basicConstraints 扩展。该领域的定义完全是一个局部策略的问题。

```
cACertificate                           ATTRIBUTE    ::=  {
    WITH SYNTAX                         Certificate
    EQUALITY MATCHING RULE              certificateExactMatch
    ID                                  id-at-cAcertificate }
```

11.2.3 交叉证书对的属性

CA 目录项的 crossCertificatePair 属性的 issuedToThisCA 元素，用于存储除自发布给 CA 的证书之外的所有证书。同样，CA 目录项的 crossCertificatePair 属性的 issuedByThisCA 元素可以包含由这个 CA 发布给其他 CAs 的证书的一个子集。如果一个 CA 给另一个 CA 发送证书，并且主体 CA 在层次等级上并不低于发送 CA，那么发送 CA 应该在自身目录项的 crossCertificatePair 属性的 issuedByThisCA 元素中放入证书。当 issuedToThisCA 元素和 issuedByThisCA 元素处在一个单一的属性值中，一个证书中的发布名称将会与其他证书中的主体名相匹配，反之亦然，一个证书中的主体公钥将有可能验证其他证书上的数字签名，反之亦然。

当一个 reverse 原理存在，向前原理的值和后退原理的值不需要被存储在同一属性值中；换句话说，它们可以被存储在单一属性值或者是两个属性值中。

在 V3 证书语法格式中，这些将包括一个 CA 值设置为 TRUE 的 basicConstraints 扩展。

```
crossCertificatePair                    ATTRIBUTE      ::=      {
```

国家图书馆专用

```
        WITH SYNTAX                     CertificatePair
        EQUALITY MATCHING RULE          certificatePairExactMatch
        ID                              id-at-crossCertificatePair }
    CertificatePair         ::=  SEQUENCE {
    issuedToThisCA          [0]  Certificate OPTIONAL,
    issuedToThisCA          [1]  Certificate OPTIONAL
                                 ---至少应该存在一对--- }
    (WITH COMPONENTS {..., issuedToThisCA PRESENT} |
    WITH COMPONENTS {..., issuedByThisCA PRESENT})
```

11.2.4 证书撤销列表属性

下面的属性包含一系列撤销证书。

```
certificateRevocationList               ATTRIBUTE   ::=  {
    WITH SYNTAX                         CertificateList
    EQUALITY MATCHING RULE              certificateListExactMatch
    ID                                  id-at-certificateRevocationList }
```

11.2.5 授权撤销列表属性

下面的属性包含一系列撤销的授权证书。

```
authorityRevocationList                 ATTRIBUTE   ::=  {
    WITH SYNTAX                         CertificateList
    EQUALITY MATCHING RULE              certificateListExactMatch
    ID                                  id-at-authorityRevocationList }
```

11.2.6 Delta 撤销列表属性

下面的属性类型被定义为目录项中的一个 Δ-CRL：

```
deltaRevocationList             ATTRIBUTE ::= {
    WITH SYNTAX                     CertificateList
    EQUALITY MATCHING RULE          certificateListExactMatch
    ID                              id-at-deltaRevocationList }
```

11.2.7 支持的算法属性

当与使用目录说明中定义的证书的远程端点实体通讯的时候，将为所使用的算法定义一个目录属性，下面的 ASN.1 定义了这个(多值)属性：

```
supportedAlgorithms             ATTRIBUTE ::= {
    WITH SYNTAX                     SupportedAlgorithm
    EQUALITY MATCHING RULE          algorithmIdentifierMatch
    ID                              id-at-supportedAlgorithms }

SupportedAlgorithm ::= SEQUENCE {
    algorithmIdentifier             AlgorithmIdentifier,
    intendedUsage                   [0]  KeyUsage OPTIONAL,
    intendedCertificatePolicies     [1]  CertificatePoliciesSyntax OPTIONAL }
```

多值属性的每一个值将有一个完全不同的 algorithmIdentifier 值。intendedUsage 组件的值提供了关于算法使用用法的说明(见 8.2.2.3)。intendedCertificatePolicies 组件的值确定了证书策略，并且，能够通过它所认证的算法使用限定的证书策略。

11.2.8 证书实际声明属性

certificationPracticeStmt 属性用来存储关于一个授权的认证实际声明的信息。

```
certificationPracticeStmt        ATTRIBUTE    ::=    {
    WITH SYNTAX                  InfoSyntax
    ID                           id-at-certificationPracticeStmt }

InfoSyntax                       ::=    CHOICE {
    content                      DirectoryString {ub-content },
    pointer                      SEQUENCE {
    name                         GeneralNames,
    hash                         HASH {HashedPolicyInfo }OPTIONAL }}
POLICY          ::=      TYPE-IDENTIFIER

HashedPolicyInfo                 ::=      POLICY. &Type ( {Policies })
Policies POLICY ::= {... }——执行者定义——
```

如果 content 存在，就包括证书认证机构认证实际声明的完整内容。

如果 pointer 存在，name 组件涉及到一个或者更多的域，这些域是授权认证实际说明的副本能够被定位的地方。如果 hash 组件存在，它包含一个证书实际声明的内容的 HASH，可在参考位置发现这个声明。这个 hash 能够用来实现一个参考文档的完整检查。

11.2.9 证书策略属性

certificatePolicy 属性用来储存关于证书策略的信息。

```
certificatePolicy                ATTRIBUTE    ::=    {
WITHSYNTAX                       PolicySyntax
ID    id-at-certificatePolicy }

PolicySyntax::=                  SEQUENCE {
    policyIdentifier             PolicyID,
    policySyntax                 InfoSyntax
}

    PolicyID        ::=      CertPolicyId
```

policyIdentifier 组件包括为特殊证书策略登记的对象标识符。

如果 content 存在，就包括证书策略的完整内容。

如果 pointer 存在，name 组件涉及到一个或者更多的域，这些域是证书策略的副本能够被定位的域。如果 hash 组件存在，它就包含证书策略内容的 HASH，证书策略在参考位置应该能够被发现。这个 hash 能够用来完成参考文档的完整检查。

11.2.10 PKI 路径属性

PKI 路径属性用来储存证书认证路径，每一个都由交叉证书序列组成。

```
pkiPath                          ATTRIBUTE    ::= {
    WITH SYNTAX                  PkiPath
    ID                           id-at-pkiPath }
PkiPath                          ::=      SEQUENCE OF CrossCertificates
```

这些属性能够储存在 CA 目录项中，并且将包含一些从一个 CA 到其他 CAs 的证书路径。如果使

国家图书馆专用

用这个属性,那么能够使频繁使用证书认证路径的交叉证书得到更有效的恢复。同样对于所使用的属性没有特定的要求,储存在属性中值的集合将无法向任何给定 CA 表示完整的向前证书路径集合。

11.3 PKI 目录匹配规则

目录说明定义了类型 Certificate,CertificatePair,CertificateList,CertificatePolicy 和 SupportedAlgorithm 相应属性所使用的匹配规则。这一规定也定义了匹配规则,来促进证书,或详细特性来自持有多重证书的多值属性的 CRLs,或 CRLs 的选择。

11.3.1 证书准确匹配

证书准确匹配规则对所提供的值与类型 Certificate 的属性值的相等性进行比较。它唯一地选择单份证书。

```
certificateExactMatch MATCHING-RULE ::= {
    SYNTAX              CertificateExactAssertion
    ID                  id-mr-certificateExactMatch }
CertificateExactAssertion ::= SEQUENCE {
    serialNumber        CertificateSerialNumber,
    issuer              Name }
```

如果此属性值中的此成分匹配了所提供值中的那个成分,则匹配规则返回 TRUE。

11.3.2 证书匹配

证书匹配规则将所提供的值与类型 certificate 的属性值进行比较。根据不同特征,它选择一份或多份证书。

```
certificateMatch MATCHING-RULE ::= {
    SYNTAX              CertificateAssertion
    ID                  id-mr-certificateMatch }
CertificateAssertion ::= SEQUENCE {
    serialNumber            [0]  CertificateSerialNumber     OPTIONAL,
    Issuer                  [1]  Name                        OPTIONAL,
    subjectKeyIdentifier    [2]  SubjectKeyIdentifier        OPTIONAL,
    authorityKeyIdentifier  [3]  AuthorityKeyIdentifier      OPTIONAL,
    certificateValid        [4]  Time                        OPTIONAL,
    privateKeyValid         [5]  GeneralizedTime             OPTIONAL,
    subjectPublicKeyAlgID   [6]  OBJECT IDENTIFIER           OPTIONAL,
    keyUsage                [7]  KeyUsage                    OPTIONAL,
    subjectAltName          [8]  AltNameType                 OPTIONAL,
    policy                  [9]  CertPolicySet               OPTIONAL,
    pathToName              [10] Name                        OPTIONAL,
    subject                 [11] Name                        OPTIONAL,
    nameConstraints         [12] NameConstraintsSyntax       OPTIONAL
}
AltNameType ::= CHOICE {
    builtinNameForm     ENUMERATED {
                        rfc822Name          (1),
                        dNSName             (2),
                        x400Address         (3),
                        directoryName       (4),
```

国家图书馆专用

```
                ediPartyName                (5),
                uniformResourceIdentifier   (6),
                iPAddress                   (7),
                registeredId                (8) },
    otherNameForm       OBJECT IDENTIFIER }
CertPolicySet ::= SEQUENCE SIZE (1..MAX) OF CertPolicyId
```

如果所提供值中存在的所有元素匹配了此属性值的相应元素，则匹配规则返回 TRUE，具体如下：

Serialnumber 匹配：若此属性值中的这个元素的值等于所提供值中那个元素的值；

issuer 匹配：若此属性值中的这个元素的值等于所提供值中的那个元素的值；

SubjecKeyIdentifier 匹配：若所存储属性值的这个元素的值等于所提供值中的那个元素的值；如果所存储的属性值未包含主体密钥标识符扩展，则没有匹配；

AuthorityKeyIdentifier 匹配：若在所存储属性值的这个元素的值等于所提供值中的那个元素的值；如果所存储的属性值未包含机构密钥标识符扩展项，或者媒体提供的值中所有元素在所存储的属性值中不存在，则没有匹配；

CertificateValid 匹配：若所提供的值处于所存储属性值的有效期之内；

PrivatekeyValid 匹配：若所提供的值处于由所存储属性值的私钥使用期扩展所指示的期间内，或者如果所存储的属性值中没有私钥使用期扩展项；

SubjectPublickeyAlgID 匹配：若它等于所存储的属性值中的 subjectPublickeyInformation 元素的 algorithmIdentifier 的 algorithm 元素；

KeyUsage 匹配：若在所提供的值中设置的所有比特也在所存储属性值密钥用法扩展中设置，或者在所存储的属性值中没有密钥用法扩展；

SubjectAltName 匹配：若所存储的属性值包含具有与所提供值所指示的同一名称类型的 Alt Name 元素的主体替换名称扩展；

Policy 匹配：若在存储属性值的证书策略扩展中至少出现 CertPolicySet 的一个成员，或者所提供的或所存储的证书在策略组中包含有特殊的 anyPolicy 的值，则匹配。如果在所存储属性值中没有证书策略扩展，则没有匹配。

PathToName 匹配：除非证书具有名称限制扩展，而此扩展禁止向所提供名称值构建的证书认证路径。

Subject 匹配：若属性中元素的值与所提供的值相等。

nameConstraints 匹配：若存储属性值中的主体名称是在给定值的允许子树组件值提供的名称空间里，但不在由给定值中的拒绝子树值提供的名称空间里。

11.3.3 证书对准确匹配

证书对准确匹配规则对所提供的值与类型 CertificatePair 的属性值的相等性进行比较。它唯一地选择单个交叉证书对。

```
CertificatePairExactMatch MATCHING-RULE          ::=  {
    SYNTAX          CertificatePairExactAssertion
    ID              id-mr-certificatePairExactMatch }
CertificatePairExactAssertion ::= SEQUENCE {
    issuedToThisCAAssertion   [0] CertificateExactAssertion OPTIONAL,
    issuedByThisCAAssertion   [1] CertificateExactAssertion OPTIONAL }
    (WITH COMPONENTS        {..., issuedToThisCAAssertion PRESENT }|
    WITH COMPONENTS         {..., issuedByThisCAAssertion PRESENT })
```

如果在所提供值的 issuedToThisCAAssertion 和 issuedByThisCAAssertion 成分中存在的成分分

别与所存储属性值中的 issuedToThisCA 和 issuedByThisCA 成分的相应成分匹配，匹配规则返回 TRUE。

11.3.4 证书对匹配

证书对匹配规则把存在值与类型 CertificatePair 的属性值相比较。它选择一个或者更多的在不同特征基础上的交叉证书对，这些特征要么是 issuedToThisCA 或者是 issuedByThisCA 的证书对。

```
certificatePairMatch MATCHING-RULE ::= {
    SYNTAX          CertificatePairAssertion
    ID              id-mr-certificatePairMatch }
CertificatePairAssertion    ::=    SEQUENCE {
    issuedToThisCAAssertion     [0] CertificateAssertion OPTIONAL,
    issuedByThisCAAssertion     [1] CertificateAssertion OPTIONAL }
    (WITH COMPONENTS            {..., issuedToThisCAAssertion PRESENT } |
    WITH COMPONENTS             {..., issuedByThisCAAssertion PRESENT })
```

如果在存储属性值中的 issuedToThisCAAssertion 和 issuedByThisCAAssertion 成分中存在的所有成分分别与所有存储属性值中的 issuedToThisCA 和 issuedByThisCA 成分的相应成分匹配，则匹配规则返回 TRUE。

11.3.5 证书列表准确匹配

证书列表准确匹配规则对所提供的值与类型 certificateList 的属性值的相等性进行比较。它唯一地选择单一 CRL。

```
CertificateListExactMatch MATCHING-RULE    ::=    {
    SYNTAX      CertificateListExactAssertion
    ID          id-mr-certificateListExactMatch }
CertificateListExactAssertion ::= SEQUENCE {
    issuer              Name,
    thisUpdate          Time,
    distributionPoint   DistributionPointName OPTIONAL }
```

如果在存储属性值的这些成分匹配了所提供值中的那些成分，则此规则返回 TRUE。如果 distributionPoint 成分存在，那么它必须至少用一种名称形式进行匹配。

11.3.6 证书列表匹配

证书列表匹配规则将所提供的值与类型 CertificateList 的属性值进行比较。根据不同的特征，它选择一份或多份 CRL。

```
certificateListMatch MATCHING-RULE ::= {
    SYNTAX      CertificateListAssertion
    ID          id-mr-certificateListMatch }
CertificateListAssertion ::= SEQUENCE {
    issuer                  Name                    OPTIONAL,
    minCRLNumber    [0]     CRLNumber               OPTIONAL,
    maxCRLNumber    [1]     CRLNumber               OPTIONAL,
    reasonFlags             ReasonFlags             OPTIONAL,
    dateAndTime             Time                    OPTIONAL,
    distributionPoint   [2] DistributionPointName   OPTIONAL,
    authorityKeyIdentifier [3] AuthorityKeyIdentifier OPTIONAL }
```

如果所有的存在值中的组件与相应的储存属性值组件相匹配，那么匹配规则就返回 TRUE，属性

国家图书馆专用

值如下：

issuer 匹配，如果属性值的这个成分的值等于所提供值的那个成分的值；

minCRLNumber 匹配，如果其值小于或等于所存储属性值的 CRL 号扩展中的值；如果所存储属性值没有包含 CRL 号扩展，则没有匹配；

maxCRLNumber 匹配，如果其值大于或等于所存储属性值的 CRL 号扩展中的值；如果所存储属性值没有包含 CRL 号扩展，则没有匹配；

reasonFlags 匹配，如果在所提供的值中设置的任一比特也在颁发的所存储属性值的分布点扩展的 only SomeReasons 成分中设置；如果所存储属性值不包含颁发的分布点扩展中的 reasonFlags 那么也有一个匹配，或者如果存储属性值不包含颁发的分布点扩展；

注：虽然和 reasonFlags 特殊值上的一个 CRL 相匹配，CRL 可能不包含任何带有原因码的任何撤销记录。

dateAndTime 匹配，如果此值等于或大于所存储的属性值的 ThisUpdate 成分中的值，并且小于所存储的属性值的 nextUpdate 成分中的值；如果所存储的属性值不包含 nextUpdate 成分，则没有匹配；

distributionPoint 匹配，如果所存储的属性值包含颁发的分布点扩展，并且所提供的值中的此成分的值等于在此扩展中对应的至少一个名称形式的值；

authorityKeyIdentifier 匹配，如果在储存属性值中组件值等于在其中存在的值；如果储藏属性值不包含认证密钥认证扩展，或者如果并非所有在存在值中的组件是存在于储藏属性值中，那么就没有匹配。

11.3.7 算法标识符匹配

算法标识匹配规则对所提供的值与类型 SupportedAlgorithm 的属性值的相等性进行比较：

algorithmIdentifierMatch MATCHING-RULE ::= {
SYNTAX AlgorithmIdentifier
ID id-mr-algorithmIdentifierMatch }

如果所提供的值等于所存储的属性值的 algorithmIdentifier 成分，则此规则返回 TRUE。

11.3.8 策略匹配

策略匹配规则把等于存在值和类型 CertificatePolicy 的属性值或者类型 PrivPolicy 属性值相比较。

policyMatch MATCHING-RULE ::= {
SYNTAX PolicyID
ID id-mr-policyMatch }

如果存在值等于储藏属性值的 policyIdentifier 组件，这个规则返回 TRUE。

11.3.9 PKI 路径匹配

PkiPathMatch 匹配规则把同等的一个存在值与一个类型 pkiPath 属性值相比较。证书使用系统可以使用这个匹配规则来选择路径，路径由一个可以信任的 CA 发布的证书开始，以发布给那些颁发终端实体的有效证书的 CA 机构的证书结束。

```
pkiPathMatch MATCHING-RULE ::= {
    SYNTAX              PkiPathMatchSyntax
    ID                  id-mr-pkiPathMatch }
PkiPathMatchSyntax      ::= SEQUENCE {
    firstIssuer         Name,
    lastSubject         Name }
```

如果在 firstIssuer 组件中的存在值与相应的在储存值的 SEQUENCE 中的第一个证书的 issuer 域的规则相匹配，并且在 LastSubject 组件中的存在值与相应的存储值的 SEQUENCE 中的最后一个证书主体域的规则相匹配，那么匹配规则就返回 TRUE。如果任何一个匹配没有完成，那就返回 FALSE。

国家图书馆专用

第三篇　属性证书框架

这里定义的属性证书框架提供了一个基础，在这个基础上能够创建特权管理基础设施（PMI）。这个基础设施能够支持像访问控制这样的应用。

特权到一个实体的绑定是由一个证书认证机构通过被称作属性证书的数字签名的数据结构提供的，或者通过为该目的包含了明确定义的扩展的公钥证书来提供。同时还定义了属性证书的格式，包括一个可扩展的机制以及一套特定证书扩展。属性证书的撤销功能可以有，也可以没有。例如，在某些环境中，属性证书的有效期可能非常短（可以短到几分钟），因此就没有执行撤销操作的必要。反之，如果由于某些原因，证书认证机构撤销了一个以前发布的属性证书，则用户就需要获悉已经发生的撤销，而不再使用一个不可信赖的证书。撤销列表是一个能够用来通告撤销用户的模型。撤销列表的格式在本部分中的第二部分定义，包括一个可扩展机制以及一套撤销列表扩展。同时还定义了附加扩展。在证书和撤销列表两种情况中，其他团体也定义了对它们的特定环境有用的附加扩展。

使用属性证书的系统，应该在应用程序使用证书之前验证证书的有效性。执行有效性检查的程序也在这里进行了定义，包括验证证书本身的完整性、撤销状态，以及对于特定用途的有效性。

本架构也包括了大量的只适合某些环境的可选元素。虽然模型定义得很完备，但本架构也能够用在不是所定义模型的所有组件都被使用的环境中。例如，有些环境中，属性证书的撤销不是必需的。特权的委托及角色的使用也是这个框架的一方面，尽管还没有得到普遍的应用，但是它们仍然被包括在本部分中，因此确实有这方面需求的环境也可以得到支持。

目录使用属性证书来提供对目录信息的基于角色的访问控制。

12　属性证书

公钥证书主要用来提供身份服务，在它的基础上可以建立其他的安全服务，例如：数据的完整性、实体认证、机密性和授权等。为了给持有者绑定一个特权属性，本部分提供了两个不同的机制。

公钥证书与实体认证服务一起使用，如果特权通过发布 CA 的方法和主体相联系，则公钥证书能够直接提供授权服务。公钥证书可以包含一个 subjectDirectoryAttributes 扩展，扩展中包括与公钥证书的主体相联系的特权。这个机制适合于下列情况：发布公钥证书的证书认证机构（CA）也是委托特权（AA）的证书认证机构并且特权的有效期与公钥证书的有效期相符合。终端实体不能作为 AA。如果定义在本部分第 15 章中的任何一个扩展包括在公钥证书中，则那些扩展也会同样使用在公钥证书的 subjectDirectoryAttributes 扩展中分配的所有特权。

在更普遍的情况下，实体特权将有一个和公钥证书的有效期不一致的生命期。特权的生命期通常更短。特权分配认证机构往往不同于相同实体的公钥证书发布认证机构，并且不同的特权可以由不同的属性机构（AA）分配。另外，还可以在一个临时环境中分配特权，并且特权特征的“开启/关闭”可以异步于公钥证书的生命期和/或异步于不同 AA 发布的实体特权。由 AA 发布的属性证书的使用，提供了一个灵活的特权管理基础设施（PMI），可以独立于 PKI 建立和管理它。同时，在 PKI 和 PMI 之间又有着一定的联系，凭这个联系，PKI 可以用来认证属性证书中发布者和持有者的身份。

12.1　属性证书结构

属性证书与主体公钥证书是两个独立的结构。一个主体可以有多个与其公钥证书相关联的属性证书。没有必要由同一个证书认证机构为用户创建公钥证书和属性证书（可以是多个），实际上，职能的分离通常是以其他方式指明。在由不同的证书认证机构负责发布公钥证书和属性证书的情况下，由 CA 发布的公钥证书和由 AA 发布的属性证书将使用不同的私钥进行签名。在一个单独的实体既是发布公钥证书的 CA，又是发布属性证书的 AA 的情况下，强烈推荐使用不同的密钥对属性证书和公钥证书进行签名。发布认证的机构和接收证书的实体之间的交换不属于本部分的范围。

属性证书定义如下：

```
AttributeCertificate ::= SIGNED{AttributeCertificateInfo}
AttributeCertificateInfo ::= SEQUENCE
    {
    Version                         AttCertVersion -version is v2,
    Holder                          Holder,
    Issuer                          AttCertIssuer,
    Signature                       AlgorithmIdentifier,
    serialNumber                    CertificateSerialNumber,
    attrCertValidityPeriod          AttCertValidityPeriod,
    attributes                      SEQUENCE OF Attribute,
    issuerUniqueID                  UniqueIdentifier OPTIONAL,
    extensions                      Extensions OPTIONAL
    }
AttCertVersion ::= INTEGER{v2(1)}
Holder      ::=     SEQUENCE
    {
    baseCertificateID               [0] IssuerSerial        OPTIONAL,
      ——持有者公钥证书的发布者及序列号
    entityName                      [1] GeneralNames        OPTIONAL,
      ——实体或角色名
    objectDigestInfo                [2] ObjectDigestInfo    OPTIONAL
      ——如果存在,则版本号必须为 v2
      ——baseCertificateID,entityName 或 objectDigestInfo 必须至少存在一个——}
    publicKeyCert                       (1),
    otherObjectTypes                    (2)},
    otherObjectTypeID                   OBJECT IDENTIFIER OPTIONAL,
    digestAlgorithm                     AlgorithmIdentifier,
    objectDigest                        BIT STRING }

AttCertIssuer ::=       [0] SEQUENCE {
issuerName              GeneralNames OPTIONAL,
baseCertificateID       [0] IssuerSerial OPTIONAL,
objectDigestInfo        [1] ObjectDigestInfo OPTIONAL }
——至少存在一个成分
(WITH COMPONENTS{...,issuerName PRESENT}|
WITH COMPONENTS{...,baseCertificate IDPRESENT}|
WITH COMPONENTS{...,objectDigestInfo PRESENT})

IssuerSerial ::= SEQUENCE{
    Issuer          GeneralNames,
    Serial          CertificateSerialNumber,
    issuerUID       UniqueIdentifier OPTIONAL }
```

国家图书馆专用

```
AttCertValidityPeriod ::= SEQUENCE {
    notBeforeTime      GeneralizedTime,
    notAfterTime       GeneralizedTime }
```

版本号(version number)用来区分属性证书的不同版本。如果持有者(holder)包括 objectDigestInfo,或者发布者(issuer)包括 baseCertificateID 或者 objectDigestInfo,则版本号必须为 v2。

持有者(holder)域用来传递属性证书持有者的身份。

如果 baseCertificateID 存在,则用该属性证书声称特权时,可以用它认证该持有者身份的公钥证书。

如果 entityName 存在,可以用它认证持有者的一个或多个名称。如果 holder 中只有 entityName,则当用该属性证书声称特权时,任何一个将这些名称之一作为其主体的公钥证书可以用来认证该持有者的身份。如果 baseCertificateID 和 entityName 同时存在,则只能够使用由 baseCertificateID 指定的证书。在这种情况下,entityName 仅仅作为一个帮助特权检验者查找已经过认证的公钥证书的工具。

注 1:单独使用 GeneralNames 认证持有者的风险是 GeneralNames 只指向持有者的名称。另外,单独使用 GeneralNames 也往往不足以认证给其发布了特权的持有者的身份。无论如何,在发布特定的公钥证书时,使用发布者名称和特定公钥证书序列号,能够使属性证书的发布者信赖由 CA 完成的认证处理。另外,特别是当持有者是一个角色而非单独实体时,在命名属性证书持有者时使用 GeneralNames 的一些选项(如 IP Address)是不合适的。单独使用 GeneralNames 作为持有者标识符的另一个问题是,结构内部的多种名称格式没有严格的注册认证机构或名称分配过程。

如果存在,ObjectDigestInfo 直接用来认证持有者的身份,包括一个可执行的持有者(例如一个 Java 程序)。通过相应信息摘要和 objectDigest 内容的比较,实现持有者的认证,这个信息摘要由特权检验者使用 ObjectDigestInfo 中认证的相同算法创建。如果两者相同,则为了实现用该属性证书声称特权,需要对持有者进行认证。

——PublicKey 将在信息摘要中包括实体公钥的 Hash 值时指明。对一个公钥进行散列计算可能不会唯一认证一个证书(也就是说,相同的密钥值可以出现在多个证书中)。为了将属性证书链接到一个公钥上,必须以存在于公钥证书中的公钥表示为基础,计算这些散列值。特别是,散列算法的输入会是密钥 SubjectPublicKeyInfo 表示的 DER 编码,这包括 AlgorithmIdentifier 和 BIT STRING。注意,如果用作输入到 hash 函数中的公钥值是从公钥证书中析取出来的,则上面所说的对一个公钥进行 HASH,可能会唯一认证一个证书(例如,如果数字签名算法的参数是继承得来的),然而,这样又不能为 HASH 提供充分的输入。在这个环境中对 Hash 的正确输入主要包括继承得来的参数值,这样就可以不同于公钥证书中的 SubjectPublicKeyInfo 表示。

——PublicKeyCert 将在公钥证书被 Hash 时指明,这个 hash 遍及了公钥证书的全部 DER 编码,包括签名位。

——OtherObjectTypes 将在除了公钥或公钥证书以外的对象(如软件对象等)被 hash 时指明。可以选择性地提供对象类型的身份。被 hash 的部分对象可以通过明确的、规定的类型标识符决定,或者是如果不支持该标识符,则由使用了该对象的环境决定。

issuer 域传递由 AA 发布的证书的身份。

——如果存在 issuerName,则用来认证发布者的一个或多个名称。

——如果存在 baseCertificateID,则通过涉及到的特定的公钥证书来认证发布者,这个证书的主体就是发布者。

——如果存在 objectDigestInfo,则通过为发布者提供认证信息的散列值来认证发布者。

signature 用来认证对属性证书进行数字签名的加密算法。

serialNumber 是用于在其发布者范围内唯一认证属性证书的序列号。

AttrCertValidityPeriod 表示属性证书的有效期，以 GeneralizedTime 格式表示。

attributes 域包含了同正在被认证的持有者相关的属性（如：特权）。

注 2：在属性描述符属性证书情况下，属性的序列可以为空。

IssuerUniqueID 可以用于在发布者组件不充分的环境中，认证属性证书的发布者。

extentions 域允许在属性证书中添加新域。

这里所描述的属性证书框架主要提供了属性证书中的特权模型。不过，同前面提到的定义一样，在本节中的证书扩展也可以放在使用了 subjectDirectoryAttributes 扩展的公钥证书中。

12.2 属性证书路径

同公钥证书一样，有时也需要传递一个属性证书路径（例如：在一个应用协议内部声称特权）。下面的 ASN.1 数据类型可以用来描述属性证书路径：

```
AttributeCertificationPath ::= SEQUENCE {
    attributeCertificate      AttributeCertificate,
    acPath                    SEQUENCE OF ACPathData OPTIONAL }
ACPathData ::= SEQUENCE {
    certificate               [0]  Certificate OPTIONAL,
    attributeCertificate      [1]  AttributeCertificate OPTIONAL }
```

13 属性权威、SOA 和证书认证机构的关系

属性权威（AA）和证书认证机构（CA）在逻辑上是完全独立的（许多情况下，在物理上也是独立的）。“身份”的创建和维护可以（并且常常应该）从 PMI 中分离出来，因此包括 CA 的完整 PKI，可以在创建 PMI 之前存在并运行。虽然 CA 是其域中的身份的源授权机构，但本身并非特权的源授权机构，因此 CA 没必要自己成为 AA；同时逻辑上，也没必要负责决定由什么样的实体行使 AA 的功能（例如，通过包含在它们身份证书中的标志等）。

源授权机构（SOA）是受特权检验者信任的，最终负责分配特权集合的实体。一种资源可以通过信任特定功能（例如，读特权和写特权）的 SOA，来限制 SOA 权威。SOA 本身就是一个 AA，因为 SOA 为其他实体发布了证书，并在证书中为这些实体分配特权。另外 SOA 同 PKI 中的‘根 CA’或‘信任点’也很类似，在这样的 SOA 中，特权检验者信任 SOA 所签名的证书。在一些环境中，CAs 需要牢牢控制具有 SOAs 功能的实体，本架构就提供了支持这些需求的机制。在其他环境中，不需要这种控制，因此决定在这些环境中行使 SOAs 功能的实体不属于本部分的讨论范围。

由于该架构非常灵活，所以能够满足多种类型环境的需要。

a) 在许多环境中，所有的特权可以直接由单个 AA，即 SOA 分配给个体（独立的实体）。

b) 其他环境可能需要对可选的角色特征的支持，籍此为个体发布证书，在证书中为该实体分配了各种角色。同时还隐含了为个体分配与角色相关的特权。角色特权可以自身或通过其他方式（例如本地配置）分配到一个被发布给角色自身的属性证书中。

c) 该框架的另一个可选特性就是对特权委托的支持。一旦委托完成，则 SOA 将特权分配给一个实体，并允许该实体行使 AA 的功能以及进一步委托特权。委托可以持续在几个中介 AA 中进行，直到最后分配给一个不能再做进一步特权委托的终端实体。中介 AA 对它们所委托的特权，可以是特权声称者，也可以不是。

d) 在一些环境中，同一个物理实体既可以行使 AA，也可以行使 CA 的功能。当特权在公钥证书的 subjectDirectoryAttributes 扩展内部传递时，会经常出现同一物理实体充当双重逻辑角色的情况。在其他环境中，不同的物理实体分别作为 CA 和 AA，在这种情况下，应该使用属性证书而不是公钥证书分配特权。

当属性证书指向其发布者和持有者的公钥证书时，PKI 用于认证持有者（特权声称者）及检验发布

者的数字签名。

13.1 属性证书中的特权

实体可以通过两种方式获得特权：

——通过属性证书的创建(也许完全在于它自身的主动,或第三方的请求),AA 可以单方面地为实体分配特权。创建好的属性证书首先存储在一个可公共访问的仓库中,随后由一个或多个特权检验者操作,以做出授权决定。所有这些能在无实体知识或实体外在行为的情况下发生。

——作为选择,一个实体可以请求某些 AA 的特权。一旦创建,属性证书就能(仅仅)返回给请求实体,当请求访问某些受保护的资源时,该证书就能为请求实体提供。

注意在所有的程序中,AA 必须履行其职责以确保特权确实分配给了该实体。它可以包含某些带外(out-of-band)机制,类似于由 CA 绑定的实体/密钥对的认证。

基于 PMI 的属性证书适合于以下任何一个环境：

——一个不同的实体为持有者分配特殊特权,其他的实体为这个持有者分配公钥证书；

——来自不同证书认证机构的大量特权属性分配给持有者；

——特权生命期不同于持有者的公钥证书有效期(一般特权的生命期短一些),或；

——特权仅在特定的时间段内有效,该时间段与用户的公钥或其他特权的有效期不一致。

13.2 公钥证书中的特权

在某些环境中,特权通过 CA 的实际操作与主体关联。这样的特权可以直接放在公钥证书中(因此重复使用差不多已经建立好的基础设施要胜于发布属性证书)。在这种情况下,特权包含在公钥证书的 subjectDirectoryAttributes 扩展中。

这种机制适合于以下任何一种环境：

——同一个物理实体可以同时行使 CA 和 AA 的功能；

——特权的生命期与包含在证书中的公钥一致；

——不允许特权委托；

——允许委托,但对于任何委托,证书中的所有特权(在 subjectDirectoryAttributes 扩展中的)必须具有相同的委托参数,同时所有与委托相关的扩展要平等地应用到证书中的所有特权中。

国家图书馆专用

14 PMI 模型

14.1 一般模型

一般特权管理模型由三个实体组成:对象,特权声称者和特权检验者。

对象可以作为受保护的资源,例如在一个访问控制应用中,这个受保护的资源作为对象引用。对象类型具有可以被调用的方法(例如,对象可能是具有“接受登录”对象方法的防火墙,或是文件系统中具有可读、写和执行对象方法的文件)。该模型中的另一个对象类型可以是不可抵赖应用中被签名的对象。

特权声称者是持有特殊特权,并为特殊的使用环境声称其特权的实体。

特权检验者是决定所声称的特权对给定的使用环境是否充分的实体。

特权检验者所作出的通过/失败决定取决于以下四个方面：

—— 特权声称者的特权；

—— 适当的特权策略；

—— 如果与环境有关的话,当前的环境变量；

—— 如果与对象方法相关的话,对象方法的敏感性。

特权持有者的特权反映了证书发布者对该持有者的信任程度,特权持有者也会坚持那些不受技术手段辖制的策略。这个特权封装在特权持有者的属性证书(或其公钥证书的 subjectDirectoryAttributes 扩展)中,它可以在调用请求中,提交给特权检验者,或以其他方式分配,例如通过目录。特权

的编码可以通过包含 AttributeType 和 SET OF AttributeValue 的属性结构来实现。用于指定特权的一些属性类型可能只需要非常简单的语法，例如一个单独的 INTEGER 或 OCTET STRING。其他的一些属性可能会有比较复杂的语法。在附录 D 中给出了实例。

特权策略指出了特权的程度，对于给定对象方法的敏感性或使用环境，该程度被认为是充分的。为了保证完整性和真实性，必须保护特权策略。存在许多可能的传输策略。一个极端的情况就是策略根本没有被传达，但经过了简单定义，并曾经只被保留在特权检验者自己的环境中。另一种极端情况就是某些策略是“通用的”，并且是系统中的每个实体都知道的，同时可以被传达给系统中的每个实体。这两种情况之间有许多变换的情况。本部分定义了在目录中存储特权策略信息的计划组件。

特权策略指出了接受给定特权集的阈值。即，特权策略精确地定义了特权检验者对于特权声称者的访问（所请求的对象，资源，应用等），所提供的特权集是充分的。

本部分中特权策略的定义语法是非标准化的。附录 D 中包含了用于此目的的语法实例，当然，仅仅是一些例子。包括明文在内的任何语法都可以用于此目的。不考虑用于定义特权策略的语法，特权策略的每一个实例必须得到唯一地标识。对象标识符就是用于此目的的。

PrivilegePolicy ::= OBJECT IDENTIFIER

如果有关，环境变量获取一些策略的特征，这些特征对特权检验者通过一些本地方式做出通过/失败决定（例如时刻或当前帐目余额等）是必须的。环境变量的表示完全是一个本地事件。

如果有关，对象方法敏感性可以反映文档或即将被处理的请求的属性，像资金转帐的货币值等就意味着授权，或文件内容的机密性。对象方法敏感性可以在一个相关的安全标签或对象方法持有的属性证书中明确编码，或者可以隐式地封装在相关的数据对象的结构及内容中。对象方法敏感性能用许多方式进行编码。例如，在 EDIFACT 交换区，它可以在与文档相关的 X.411 标签中，编码在 PMI 范围外，或在特权检验者的应用中硬编码。它可以选择性地在 PMI 内部或属性证书中编码。另外，在一些环境中，没有使用对象方法敏感性。

在特权检验者和任何特殊的 AA 间没有任何必然的绑定关系。正像特权声称者可以通过许多不同的 AAs 发布属性证书一样，特权检验者可以接收由众多 AAs 发布的证书访问一种特殊资源，这些不同的 AA 不必在层次上彼此关联。

属性证书框架可以为各种目的，管理不同类型的特权。本部分中所使用的一些术语，例如特权声称者，特权检验者等，独立于特殊的应用或用法。

14.1.1 访问控制环境中的 PMI

关于访问控制，标准框架（ISO/IEC10181-3|ITU-TRec. X.812）定义了专门用于访问控制应用的一套相关术语集。另外，这里还提供了用于本部分中的一般术语到访问控制框架中的专用术语的映射，以阐明这个框架模型与规范之间的联系。

本部分中的特权声称者将起到访问控制框架中“发起人”角色的作用。

本部分中的特权检验者将起到访问控制框架中“访问控制决定函数（ADF）”角色的作用。

本部分中用于声称特权的对象方法同访问控制框架中定义的“目标”相对应。

本部分中的环境变量同访问控制框架中的“环境信息”相对应。

本部分中讨论的特权策略包括访问控制框架中定义的“访问控制策略”和“访问控制策略规则”。

本模型允许 PMI 完全无缝地覆盖在一个受保护的现有资源网络上。特别的，该模型将特权检验者作为到敏感对象方法的网关，准许或拒绝该对象方法的撤销请求，使对象受到保护，而对对象本身没有或者只有一点影响。特权检验者将屏蔽全部请求，只有经过适当授权的请求才可以传递到适当的对象方法。

14.1.2 不可抵赖环境中的 PMI

关于不可抵赖，标准框架（ISO/IEC10181-4|ITU-TRec. X.813）定义了专门用于不可抵赖的一套相关术语集。另外，这里还提供了用于本部分中的通用术语到不可抵赖框架中的专用术语的映射，以阐明

这个框架模型与规范之间的联系。

本部分中的特权声称者将起到不可抵赖框架中“证据主体”或“发起人”的角色。

本部分中的特权检验者将起到不可抵赖框架中“证据用户”或“接收者”的角色。

本部分中用于声称特权的对象方法同不可抵赖框架中定义的“目标”相对应。

本部分中的环境变量同不可抵赖框架中的“证据生成及检验的日期和时间”相对应。

本部分中讨论的特权策略包括不可抵赖框架中的“不可抵赖安全策略”。

14.2 控制模型

控制模型阐明了如何在对敏感对象方法的访问上运用控制。模型由五部分组成:特权声称者、特权检验者、对象方法、特权策略和环境变量(见图 3)。特权声称者拥有特权;对象方法拥有敏感性。这里所描述的技术可以使特权检验者通过特权声称者控制对象方法的访问权限,从而与特权策略达成一致。特权和敏感性可以是多值参数。

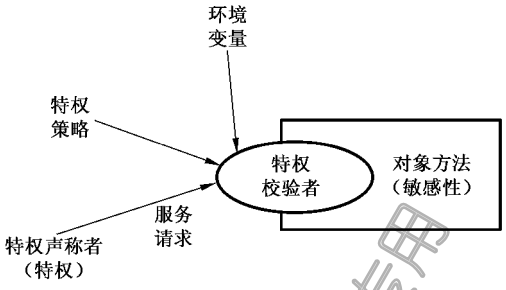


图 3 控制模型

特权声称者可以是由公钥证书所认证的实体,或者是由其磁盘映像摘要所认证的可执行对象等。

14.3 委托模型

某些环境中可能需要委托特权。然而,这只是框架的可选特征,并非所有的环境都需要该特征。委托模型由四部分组成:特权检验者,SOA,其他 AA 以及特权声称者(见图 4)

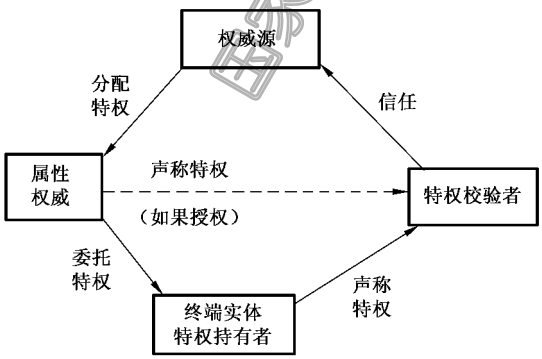


图 4 委托模型

对于不使用委托的环境,SOA 是为特权持有者分配特权的最初的证书发布者。然而,在 SOA 允许特权持有者行使 AA 功能,以及通过包含相同特权(例如它的一个子集)的证书的发布对其他实体进行进一步委托的情况下,SOA 能够在实施的委托上强加限制(例如限制路径长度,限制被实施的委托的名称空间等)。每一个中介 AA 可以在它发布给进一步特权持有者的证书中,通过这些持有者行使 AA 的功能,完成进一步的授权委托。委托的普遍约束就是 AA 不能委托超过其持有范围的特权。委托者也可以进一步限制下属 AA 的能力。

当使用委托时,特权检验者信任由 SOA 将部分或全部特权委托给持有者,其中部分持有者可以进一步将部分或全部特权委托给其他持有者。

特权检验者信任 SOA 作为给定资源特权集的证书认证机构。如果特权声称者的证书不是由 SOA 发布的,则特权检验者必须查找出证书的委托路径,该路径在特权声称者到 SOA 发布的一个证书之

间。委托路径的确认必须包括确认每个 AA 有足够的特权,以及每个 AA 在适当的时候,被授权委托这些特权。

对于用属性证书传送特权的情况,委托路径不同于用于确认包含在委托过程中的实体公钥证书的证书确认路径。然而,由公钥证书确认过程提供的确认质量必须与被保护的对象方法的敏感性相当。

委托路径或者完全由属性证书组成,或者完全由公钥证书组成。如果得到授权,在属性证书中包含其特权的委托者可以只通过后来属性证书的发布进行委托。同样,如果被授权,在公钥证书中包含其特权的委托者可以只通过后来公钥证书的发布进行委托。只有 AA 可以委托特权,终端实体则不能。

14.4 角色模型

角色为个体提供了一种间接分配特权的手段。给个体发布角色分配证书,这些证书可以通过包括在证书中的角色属性为个体分配一个或多个角色。通过角色说明证书将一定的特权分配给角色名,比通过属性证书将特权分配给单独的特权持有者好。间接的程度能够使诸如分配给一个角色的特权得到更新,同时不会与给个体分配角色的证书发生冲突。角色分配证书可以是属性证书,也可以是公钥证书;但角色说明证书只能是属性证书,不能是公钥证书。如果没有使用角色说明证书,则角色的特权分配还可以通过其他方式完成(例如,可以由特权检验者进行本地配置)。

以下几条都是可能的:

—— 任何一个 AA 都可以定义任意数目的角色;

—— 角色本身及其成员可以由不同的 AA 分别定义和管理;

—— 角色的所有成员可以像其他特权一样被委托,并且;

—— 角色及其全体成员可以分配到任何合适的生命期。

如果角色分配证书是属性证书,则角色(role)属性包括在属性证书的属性(attributes)中。如果角色分配证书是公钥证书,则角色(role)属性包括在 subjectDirectoryAttributes 扩展中。后一种情况中,包含在公钥证书中的附加特权是直接分配给证书主体的特权,而不是分配给角色的特权。

因此,特权声称者可以向特权检验者提交一个角色分配证书以证明只有该特权声称者才有特殊的角色(例如,“管理者”或“购买者”)。特权检验者可以预先知道,或是通过其他途径发现为做出通过/失败的授权决定,与所声明的角色相关的特权。角色说明证书可以用于此目的。

特权检验者必须理解为该角色所指定的特权。对这个角色的特权分配可以在角色说明证书的 PMI 内部完成,也可以在 PMI 外部完成(如,本地配置)。如果在角色说明证书内声明角色特权,则本部分提供了为特权声称者将证书同相应角色分配证书链接的机制。角色说明证书不能被委托给任何实体。角色分配证书的发布者可以独立于角色说明证书的发布者,并且完全可以分别对两者进行管理(期满、撤销等)。相同的证书(属性证书或公钥证书)可以是角色分配证书,也可以包括对相同个体其他特权的直接分配。然而,角色说明证书必须是单独的证书。

注:授权框架内部角色的使用可能会增加路径处理的复杂性,因为这样的功能从本质上定义了不同的且必须遵循的委托路径。角色分配证书的委托路径可以包括不同的 AA,并且可以独立于发布角色说明证书的 AA。

14.4.1 角色属性

特权属性类型规范通常是应用问题,不属于本部分的讨论范围。但有一个唯一的例外,就是本部分定义了给声称者分配角色的属性。角色属性值的规范不是本部分的讨论范围。

```
role     ATTRIBUTE     ::= {
    WITH SYNTAX                    RoleSyntax
    ID                             id-at-role }

RoleSyntax                ::=      SEQUENCE {
    roleAuthority                  [0]  GeneralNames  OPTIONAL,
    roleName                       [1]  GeneralName }
```

特权属性用于填充角色分配证书的 attribute 域。如果角色分配证书是公钥证书,则属性用于填充公钥证书的 subjectDirectoryAttributes 扩展。

如果存在,roleAuthority 用来认证负责发布角色说明证书的公认认证机构。

如果 roleAuthority 存在,至少 roleAuthority 的一个名称必须存在于角色说明证书的 issure 域中,同时特权检验者使用角色说明证书,则它们决定分配给角色的特权。如果特权检验者在决定分配给角色的特权时所使用的方式不同于角色说明证书,则由该组件中所命名的认证机构分配特权的保证机制在本部分讨论范围之外。

如果 roleAuthority 不存在,则可靠证书认证机构的身份必须由其他方式决定。角色分配证书中的 roleSpecCertIdentifier 扩展是实现这种绑定的一种途径,在这种情况下,角色说明证书用于将特权分配给角色。

RoleName 用于认证包含该属性的,分配给角色分配证书持有者的角色。如果特权检验者使用角色说明证书决定分配给该角色的特权,则该角色名也必须出现在角色说明证书中的 holder 域中。

15 特权管理证书扩展

为便于特权管理,可将以下证书扩展包含在证书内。根据扩展本身的定义,以下提供了一些包含扩展的证书类型规则。

除 SOA 标识符扩展以外,如果公钥证书已将特权分配给它的主体(如,subjectDirectoryAttributes 扩展存在),那么可能包括公钥证书中可能包含的所有扩展。如果公钥证书中存在所有的扩展,那么扩展可适用于 subjectDirectoryAttributes 扩展中的所有特权。

用于发布属性证书(ACRL 和 AARL)撤销通知的撤销列表,可以包含所有 CRL 或 CRL 入口的扩展,本部分的第二节定义了 CRL 和 CRL 入口扩展。

下列条目指定了在以下领域中的扩展:

a) *基本特权管理*:证书扩展可以传递与特权声明相关的信息;

b) *特权撤销*:证书扩展可以传递与撤销状态位置信息相关的信息;

c) *权威源*:对于给定的资源来说,证书扩展可通过验证者与特权分配的可信源相关;

d) *角色*:证书扩展可以传递与有关角色说明证书位置相关的信息;

e) *授权*:证书扩展可以约束所分配特权的后续授权设置。

15.1 基本特权管理扩展

15.1.1 需求

下列需求与基本特权管理相关:

a) 发布者要能约束特权声明期;

b) 发布者要能把属性证书设为特定的服务器/服务;

c) 发布者要能传输用证书显示特权声明者和/或特权检验程序的信息;

d) 发布者要能约束特权策略,所分配的特权可与特权策略一起使用。

15.1.2 基本特权管理扩展域

定义了下面的扩展域:

a) *时间规范*;

b) *目标信息*;

c) *用户通知*;

d) *合理的特权策略*。

15.1.2.1 时间规范扩展

AA 可以使用时间规范扩展来约束特定的时间周期,在此期间由特权持有者来声明特权,这种特权是在包含扩展的证书中分配的。如,AA 可发布一个分配特权的证书,它只能在周一至周五的上午 9:00

到下午 5:00 之间进行声明。授权情况下,管理员可在外出度假期间将签名权力授给下属。

域定义如下:

```
timeSpecification EXTENSION      ::=      {
    SYNTAX              TimeSpecification
    IDENTIFIED BY       id-ce-timeSpecification }
```

AA(包括 SOA)发布给特权声明者实体和其他 AA 和终端实体的数字证书或公钥证书中可以包含这种扩展。但这种扩展不应该包含在以下证书中:包含 SOA 标识符扩展的证书,或发布给不能充当特权声明者 AA 的证书。

如果发布给 AA 实体的证书中存在这种扩展,那么它只能用于证书中所包含的特权的实体声明。这并不影响 AA 发布证书的时间周期。

由于这种扩展对证书有效期进行了改进,因此应当将这种扩展标记为关键的。(也就是说,通过包含这种扩展,发布者明确定义了特权分配在规定时间外无效。)

如果扩展存在,但特权验证者并不能识别它,则必须拒绝证书。

15.1.2.1.1 时间规范匹配

时间规范匹配规则将比较存在值与 AttributeCertificate 类型值的对等性。

```
timeSpecificationMatch MATCHING-RULE      ::= {
    SYNTAX                  TimeSpecification
    ID                      id-mr-timeSpecMatch }
```

如果存储值中包含 timeSpecification 扩展,并且现有值中的组件与相应存储值中的组件匹配,则匹配规则返回 TRUE。

15.1.2.2 目标信息扩展

目标信息扩展可将属性证书设定为特定的服务器/服务。包含该扩展的属性证书仅能用于特定的服务器/服务。

域定义如下:

```
targetingInformation    EXTENSION      ::= {
    SYNTAX                  SEQUENCE SIZE (1..MAX) OF Targets
    IDENTIFIED BY           id-ce-targetInformation }

Targets ::=                 SEQUENCE SIZE (1..MAX) OF Target

Target                      ::=             CHOICE {
    targetName              [0]     GeneralName,
    targetGroup             [1]     GeneralName,
    targetCert              [2]     TargetCert }

TargetCert                  ::=   SEQUENCE {
    targetCertificate       IssuerSerial,
    targetName              GeneralName OPTIONAL,
    certDigestInfo          ObjectDigestInfo OPTIONAL }
```

如果 targetName 组件存在,则提供目标服务器/服务的名称,并将包含属性的证书设定为目标。

如果 targetGroup 组件存在,则提供目标组的名称,并将包含属性的证书设定为目标。TargetGroup 中目标成员个数的确定不在本部分中进行讨论。

如果 targetCert 组件存在,则通过查阅其证书来识别目标服务器/服务。

国家图书馆专用

AA(包括 SOA)发布给特权声明者实体和其他 AA 和终端实体的属性证书中可以包含这种扩展。但这种扩展不应该包含在以下证书中:公钥证书或发布给不能充当特权声明者 AA 的证书。

如果发布给 AA 实体的属性证书中存在这种扩展,则它仅适用于证书中所包含特权的实体声明。这并不影响 AA 发布证书的时间周期。

扩展是关键的。

如果扩展存在,但特权验证者并不在指定之列,则将拒绝属性证书。

如果扩展不存在,则不将属性证书设定为目标,并且它可被任何服务器接受。

15.1.2.3 用户通知扩展

持有者声明特权时,用户通知扩展使 AA 包含一个显示给持有者的通知;使用包含扩展的属性证书时,则显示给特权验证者。

域定义如下:

userNotice EXTENSION ::= {
SYNTAX SEQUENCE SIZE (1..MAX) OF UserNotice
IDENTIFIED BY id-ce-userNotice }

AA(包括 SOA)发布给特权声明者实体和其他 AA 和终端实体的数字证书或公钥证书中可以包含这种扩展。但这种扩展不应该包含在以下证书中:包含 SOA 标识符扩展的证书,或发布给不能充当特权声明者 AA 的证书。

如果发布给 AA 实体的证书中存在这种扩展,则它仅适用于证书中所包含特权的实体声明。这并不影响 AA 发布证书的时间周期。

在证书发布者看来,扩展可以是关键的或非关键的。

如果扩展被标识为关键的,则必须在每次声明时将用户通知显示给特权验证者。如果特权声明者将属性证书提供给特权验证者(如,特权验证者从存储库内不直接更新),则必须将用户通知显示给特权声明者。

如果扩展被标识为非关键的,则特权验证者可以对证书中所声明的特权进行授权,而不论用户通知是显示给特权声明者和/或特权验证者。

15.1.2.4 合理的特权策略扩展

合理的特权策略域用于约束指定特权的声明,它可与指定特权策略集一起使用。

域定义如下:

acceptablePrivilegePolicies EXTENSION ::= {
SYNTAX AcceptablePrivilegePoliciesSyntax
IDENTIFIED BY id-ce-acceptablePrivilegePolicies }

AcceptablePrivilegePoliciesSyntax ::= SEQUENCE SIZE (1..MAX) OF PrivilegePolicy

AA(包括 SOA)发布给特权声明者实体和其他 AA 和终端实体的属性证书或公钥证书中可以包含这种扩展。如果公钥证书中包含这种扩展,则对于 subjectDirectoryAttribute 扩展中包含的特权来说,它作为特权声明者仅与主体的能力相关。

如果存在,则扩展被标识为关键的。

如果扩展存在,且特权验证者能识别它,则验证者必须确定所要比较的特权策略是扩展中被标识的。

如果扩展存在,但特权验证者并不能识别它,则证书被拒绝。

15.2 特权撤销扩展

15.2.1 需求

以下是与属性证书撤销相关的需求:

国家图书馆专用

a) 为控制 CRL 的大小,需要向不同的 CRL 分配由同一 AA 发布的所有证书集合的子集;

b) 属性证书的发布者应该在属性证书中指出非撤销信息可用于该证书 。

15.2.2 特权撤销扩展域

扩展域定义如下:

a) CRL 分布点;

b) 非撤销信息。

15.2.2.1 CRL 分布点扩展

在本部分的第 2 篇中定义了公钥证书中使用的 CRL 分布点扩展。这个域可以包含在属性证书中,也可以存在于发布给 AA(包括 SOA)以及发布给终端实体的证书中。

如果证书中存在 CRL 分布点扩展域,则特权验证者将会使用公钥证书第 2 篇中所描述的方式来处理扩展。

15.2.2.2 非撤销信息扩展

在某些环境下(例如,所发布的证书有效期非常短)不需要撤销证书。AA 可以用这些扩展来指示:没有为该属性证书提供撤销状态信息。域定义如下:

```
noRevAvail EXTENSION ::=        {
    SYNTAX                      NULL
    IDENTIFIED BY               id-ce-noRevAvail }
```

该扩展可能存在于由 AA(包含 SOA)发布给终端实体的属性证书中。发布给 AA 的公钥证书和属性证书中不包含这种扩展。

扩展是非关键的。

如果扩展存在于属性证书内,则特权验证者不需要寻找撤销状态信息。

15.3 授权扩展源

15.3.1 需求

以下是关于授权源的需求:

a) 在某些环境中,需要 CA 加强对可充当 SOA 的实体的控制;

b) 需要可靠的 SOA 为特权属性制定一些有效的语法定义和控制规则。

15.3.2 SOA 扩展域

定义了以下扩展域:

a) *SOA 标识符;*

b) *属性描述符。*

15.3.2.1 SOA 标识符扩展

为便于特权管理,SOA 标识符扩展应当指出证书主体可以充当 SOA。如,证书主体可以定义分配特权的属性,为该属性发布属性描述符证书,并使用与验证通过的公钥相应的私钥来发布证书,这些证书将特权分配给持有者。后续证书可以是属性证书或公钥证书,且拥有包含特权的 subjectDirectory-Attributes 扩展。

在某些环境下,不需要这个扩展,但可用其他机制来决定可以充当 SOA 的实体。仅仅在要求 CA 加强管理可充当 SOA 的实体的环境下才需要这个扩展。

域定义如下:

```
sOAIdentifier EXTENSION      ::= {
    SYNTAX                              NULL
    IDENTIFIED BY                       id-ce-sOAIdentifier }
```

如果证书内不存在扩展,则主体/持有者充当 SOA 的权力必须用其他方式来确定。

域仅仅存在于发布给 SOA 的公钥证书中。它不包含在发布给其他 AA 或终端实体特权持有者的

国家图书馆专用

属性证书或者公钥证书中。

这个扩展是非关键的。

15.3.2.2 属性描述符扩展

特权验证者需要特权属性和掌管后续特权授权的控制规则的定义来确定授权是否正确完成了。可将这些定义和规则以本部分外的多种方式提供给特权验证者(如,可以是特权验证者的本地配置)。

扩展提供了一种机制,可以由 SOA 来生成特权属性的定义和特权验证者的相关有效控制规则。包括这个扩展的属性证书叫做属性描述符证书,它是属性证书的一种类型。尽管在语法上与属性描述符证书 AttributeCertificate 相同:

——在它的 attributes 域内包含一个空的 SEQUENCE;

——是一个自发布的证书(如,发布者和持有者是同一个实体);

——包括属性描述符扩展。

域定义如下:

```
attributeDescriptor EXTENSION      ::= {
    SYNTAX                                  AttributeDescriptorSyntax
    IDENTIFIED BY                           {id-ce-attributeDescriptor }}
AttributeDescriptorSyntax      ::= SEQUENCE {
    identifier                              AttributeIdentifier,
    attributeSyntax                         OCTET STRING (SIZE(1..MAX)),
    name                                    [0]  AttributeName      OPTIONAL,
    description                             [1]  AttributeDescription      OPTIONAL,
    dominationRule                          PrivilegePolicyIdentifier }
AttributeIdentifier      ::= ATTRIBUTE.&id ({AttributeIDs})
AttributeIDs ATTRIBUTE      ::= {...}
AttributeName      ::= UTF8String(SIZE(1..MAX))
AttributeDescription      ::= UTF8String(SIZE(1..MAX))
PrivilegePolicyIdentifier    ::=          SEQUENCE {
    privilegePolicy                     PrivilegePolicy,
    privPolSyntax                       InfoSyntax }
```

identifier 组件的值 attributeDescriptor 扩展是一个标识其属性类型的对象标识符。

attributeSyntax 组件包括属性语法的 ASN.1 定义。这种 ASN.1 定义对于在 ISO/IEC 9594-2 中定义的匹配规则操作属性的信息组件是特定的。

name 组件可以包含验证属性的用户友好名。

description 组件可以包含属性的用户友好描述符。

对于属性来说,由于授权"次于"授权者所持有的相应特权,因此 dominationRule 组件指明了它的含义。privilegePolicy 组件能用对象标识符对包含规则的特权策略实例进行鉴别。privPolSyntax 组件包含了特权标识符本身或一个位置指针。如果包含了指针,则特权策略的任意散列允许在已引用的特权策略上进行完整性检测。

这个扩展可能存在于属性标识符证书上。除自发布的 SOA 证书外,扩展不会存在于公钥证书或属性证书上。

扩展是非关键的。

在创建/定义相应的属性类型时,SOA 创建的属性描述符证书是一种方式,通过这种方式,可以理解并执行基础设施中的所有授权"下放"通用约束。在当前目录下,包含扩展的属性证书会存储在 SOA 目录项的 attributeDescriptorCertificate 属性中。

15.3.2.2.1 属性描述符匹配

属性描述符匹配规则比较与 AttributeCertificate 类型的属性值的等同性。

```
attDescriptor MATCHING-RULE      ::= {
    SYNTAX                  AttributeDescriptorSyntax
    ID                      id-mr-attDescriptorMatch }
```

如果存储值包含 attributeDescriptor 扩展，且当前值中存在的组件与存储值的相应组件匹配，则匹配规则返回 TRUE。

15.4 角色扩展

15.4.1 需求

以下需求与角色有关：

——如果证书是一个角色分配证书，则特权验证者要能查找到相关角色的规范证书，该证书包含分配给角色本身的特定特权。

15.4.2 角色扩展域

定义了以下扩展域：

——*角色说明证书标识符*

15.4.2.1 角色说明证书标识符扩展

AA 可将扩展作为一个指针来使用，该指针包含角色特权分配的角色说明证书，它可以存在于角色说明证书中（例如，包含角色属性的证书）。

处理角色分配证书时，特权验证者要获得该角色的特权集，从而决定验证通过或失败。如果在角色说明证书中将特权分配给角色，则该域可用于查找证书。

域定义如下：

```
roleSpecCertIdentifier EXTENSION ::=
    {
    SYNTAX                  RoleSpecCertIdentifierSyntax
    IDENTIFIED BY           {id-ce-roleSpecCertIdentifier }
}
RoleSpecCertIdentifierSyntax ::= SEQUENCE SIZE (1..MAX) OF RoleSpecCertIdentifier
RoleSpecCertIdentifier      ::= SEQUENCE {
    roleName                        [0]  GeneralName,
    roleCertIssuer                  [1]  GeneralName,
    roleCertSerialNumber            [2]  CertificateSerialNumber    OPTIONAL,
    roleCertLocator                 [3]  GeneralNames               OPTIONAL
    }
```

roleName 用于标识角色，该名称与被扩展引用的角色说明证书持有者组件中的名称相同。roleCertIssuer 标识发布角色说明证书的 AA。如果 roleCertSerialNumber 存在，则它包含了角色说明证书的序列号。注意，如果分配给角色本身的特权改变了，则会将新的角色说明证书发布给角色。所有包含该扩展的证书（包括 roleCertSerialNumber 组件）将被引用了新序列号的证书所替代。尽管在某种特定的环境下才需要该操作，但在许多其他情况下，它又不是必需的。特别是，缺少组件时，在不影响角色指派证书的情况下，它将自动更新分配给角色的特权。

如果 roleCertLocator 存在，则它包含用于查找角色说明证书的信息。

这些扩展可能存在于角色分配证书中，它是由 AA（包括 SOA）发布给其他 AA 或终端实体特权持有者的属性证书或公钥证书。包含 SOA 标识符扩展的证书中不存在该扩展。

如果存在，特权验证者可将这种扩展用于查找角色说明证书。

国家图书馆专用

如果这个扩展不存在：

a) 可用其他方式来查找角色说明证书；或者

b) 使用除角色说明证书之外的机制来给角色分配特权（例如，角色特权在特权标识符的本地配置）。

此扩展总是非关键的。

15.4.2.1.1 角色说明证书 ID 匹配

角色说明证书标识符匹配规则比较当前值与 **AttributeCertificate** 属性值是否相等。

```
roleSpecCertIdMatch MATCHING-RULE      ::= {
    SYNTAX                  RoleSpecCertIdentifierSyntax
    ID                      id-mr-roleSpecCertIdMatch }
```

如果存储值包含 roleSpecCertIdentifier 扩展值或当前值的组件与存储值的组件相匹配，则匹配规则返回 TRUE。

15.5 授权扩展

15.5.1 需求

以下需求与特权的授权相关：

a) 为防止终端实体在没有授权的情况下将自己设立为 AA，需要将终端实体的特权证书与 AA 证书区分开来。同时 AA 还要约束后续委托路径的长度。

b) AA 需要指定适当的名称空间，该空间包括特权的授权。特权验证者需要对这些约束进行检查。

c) AA 在声明特权授权时，需要指定一个合理的证书策略，委托路径中的特权声明者用于验证自身。

d) 特权标识符需要为发布者查找相应的属性证书，并且发布者确信有足够特权来代表当前的证书特权。

15.5.2 授权扩展域

定义如下：

a) *基本属性约束*；

b) *授权名称约束*；

c) *合理的证书策略*；

d) *权威属性标识符*。

15.5.2.1 基本属性约束扩展

该域指出是否允许证书中所分配特权的后续授权包含这个扩展。如果允许，则可以指定委托路径的长度。

域定义如下：

```
basicAttConstraints EXTENSION ::=
{
    SYNTAX                      BasicAttConstraintsSyntax
    IDENTIFIED BY               {id-ce-basicAttConstraints }
}
BasicAttConstraintsSyntax ::= SEQUENCE
    {
    authority                   BOOLEAN DEFAULT FALSE,
    pathLenConstraint           INTEGER (0..MAX) OPTIONAL
}
```

authority 组件显示持有者是否可被授予更多的权限。如果 authority 为 TRUE,则根据相关约束,持有者可以是 AA 或授予它更多的权限。如果 authority 为 FALSE,则持有者就是一个终端实体,并且不能授予它更多的权限。

只有当 authority 为真时,pathLenConstraint 组件才是有意义的。它提供了最大数目的 AA 证书,这些证书符合委托路径中的证书。0 值表示这个证书的主体可以发布在终端实体的证书里,而不能发布在 AA 的证书里。如果 pathLenConstraint 域存在于委托路径的所有证书里,则委托路径的长度就没有限制。注意该约束对起始路径里的下一个证书有效。这种约束控制着包含该约束的 AA 证书和终端实体证书之间的数目。因此路径的总长度可以超过 2 个证书约束的值。它包括两个终端证书和两个终端间的 AA 证书,这两个终端受到扩展值的约束。

该扩展可以存在于 AA(包括 SOA)发布给其他 AA 或终端实体的属性证书或公钥证书中。包含 SOA 标识符扩展的证书中不包括该扩展。

如果该扩展存在于属性证书里,且 authority 为 TRUE,则授权给持有者发布后续的属性证书,将所包含的特权授予其他实体,但不能是公钥证书。

如果该扩展存在于公钥证书中,且 basicConstraints 扩展指出该主体也是一个 CA,则授权给主体发布后续公钥证书,这些证书将特权授予其他实体,但不是属性证书。如果包括路径长度约束,则主体只能在扩展指定的或 basicConstraints 扩展指定的约束交集中进行授权。如果该扩展存在于公钥证书中,但不存在于 basicConstraints 扩展中,或者显示主体是一个终端实体,则不授予该主体特权。

对证书发布者来说,这个扩展可以是关键的或非关键的,推荐将它标识为关键的,否则未授权为 AA 的持有者就可以发布证书,且特权验证者有可能使用该证书。

如果该扩展存在且被标识为关键的,则:

——如果 authority 的值没有被设置为真,则授权属性不能用于更多的授权;

——如果 authority 值被设置为真,且 pathLenConstraint 约束存在,则特权验证者应当检查处理的委托路径是否与 pathLenConstraint 一致。

如果扩展存在,但未被标识为关键的,且特权验证者也不能识别它,则授权属性用于进一步的授权时,系统将使用其他方式来作决定。

如果该扩展不存在,或扩展存在但 SEQUENCE 值为空,则持有者只能充当终端实体,而不能充当属性证书,且属性证书中不包括委托特权。

15.5.2.1.1 基本属性约束匹配

基本属性约束匹配规则比较当前值与 AttributeCertificate 类型的属性值是否相等。

basicAttConstraintsMatch MATCHING-RULE ::= {
SYNTAX BasicAttConstraintsSyntax
ID id-mr-basicAttConstraintsMatch }

如果存储的值包含 basicAttConstraints 扩展,或者当前存在的组件值与存储值相匹配,则匹配规则返回 TRUE。

15.5.2.2 授权名约束扩展

授权名约束域指定了一个名称空间,其中需要查找委托路径下后续证书中所有持有者的名称。

域定义如下:

delegatedNameConstraints EXTENSION ::= {
SYNTAX NameConstraintsSyntax
IDENTIFIED BY id-ce-delegatedNameConstraints }

对于公钥证书来说,扩展是以与 nameConstraints 扩展相同的方式来进行处理的。如果 permittedSubtrees 存在,则持有者 AA 和委托路径中后续 AA 所发布的属性证书之中,只有那些子树中带有持有者名称的属性证书才是合理的。如果 excludedSubtrees 存在,则由持有者 AA 或委托路径中的后续

国家图书馆藏书

AA 发布的属性证书是不合理的，该证书拥有子树中的持有者名称。如果 permittedSubtrees 和 excludedSubtrees 都存在，且名称空间重叠，则外部声明优先。

该扩展可以存在于 AA(包括 SOA)发布给其他 AA 的属性证书或公钥证书中。该扩展不应当存在于终端实体发布的证书或包含 SOA 标识符扩展的证书中。

如果该扩展存在于公钥证书中，且 nameConstraints 扩展也存在，则主体只能在扩展中指定的约束和 nameConstraints 扩展指定约束的交集中进行授权。

对于属性证书发布者来说，扩展可以是关键的，也可以是非关键的。这里推荐将它标识为关键的，否则属性证书用户可以不检查发布 AA 的名称空间中委托路径下的后续属性证书。

15.5.2.2.1 授权名称约束匹配

授权名称约束匹配规则比较当前值与 AttributeCertificate 类型的属性值是否相等。

```
delegatedNameConstraintsMatch MATCHING-RULE      ::= {
    SYNTAX              NameConstraintsSyntax
    ID                  id-mr-delegatedNameConstraintsMatch }
```

如果存储值包含 attributeNameConstraints 扩展，且当前值中存在的组件与相应存储值的组件相匹配，则匹配规则返回 TRUE。

15.5.2.3 合理的证书策略扩展

在采用属性证书的授权中，使用了合理的证书策略域来控制证书策略，这种策略发布了委托路径中后续持有者的公钥证书。通过在域中列举一系列的策略，AA 需要委托路径中的后续发布者仅将所包含的特权授予在一种或多种列举证书策略下发布的公钥证书持有者。这里所列举的策略并不是发布属性证书的策略，而是必须发布的后续持有者合理公钥证书下的策略。

域定义如下：

```
acceptableCertPolicies EXTENSION ::= {
    SYNTAX                  AcceptableCertPoliciesSyntax
    IDENTIFIED BY           id-ce-acceptableCertPolicies }
AcceptableCertPoliciesSyntax ::= SEQUENCE SIZE (1..MAX) OF CertPolicyid
CertPolicyid ::= OBJECT IDENTIFIER
```

该扩展可能只存在于由 AA(包括 SOA)发布给其他 AA 的属性证书里。该扩展不应被包含于终端实体的属性证书或任何公钥证书里。在使用公钥证书授权的情况下，certificatePolicies 和其他相关扩展可以提供同样的功能。

如果存在，则扩展应当被标识为关键的。

如果扩展存在并且特权验证者也能识别它，则验证者必须保证所有委托路径中的后续特权声明者都用公钥证书，在一个或多个列举的证书策略下进行了认证。

如果扩展存在，但特权验证者不能识别它，则必须拒绝该证书。

15.5.2.3.1 合理的证书策略匹配

合理的证书策略匹配规则将当前值与 AttributeCertificate 属性值进行比较，看它们是否相等。

```
acceptableCertPoliciesMatch MATCHING-RULE      ::= {
    SYNTAX                  AcceptableCertPoliciesSyntax
    ID                      id-mr-acceptableCertPoliciesMatch }
```

如果存储值包含 acceptableCertPolicies 扩展，或者当前值中出现的组件和存储值中相应的组件匹配，则匹配规则返回 TRUE。

15.5.2.4 授权属性标识符扩展

在特权委托中，委托特权的 AA 自身至少必须具有同样的特权和权力来委托特权。将特权委托给其他 AA 或终端实体的 AA 可将扩展放置在它所发布的 AA 或终端实体的证书中。扩展是一个指向证

书的后台指针,其中将相应的特权分配给了包含扩展的证书发布者。特权验证者可用该扩展来确保所发布的 AA 已经有了足够的特权来委托包含扩展的证书持有者。

域定义如下:

```
authorityAttributeIdentifier EXTENSION      ::=
    {
    SYNTAX                  AuthorityAttributeIdentifierSyntax
    IDENTIFIED BY           {id-ce-authorityAttributeIdentifier }
    }
AuthorityAttributeIdentifierSyntax          ::=  SEQUENCE SIZE (1..MAX) OF AuthAttId
AuthAttId                   ::=             IssuerSerial
```

包含该扩展的证书可将多种特权委托给证书持有者。如果对发布证书的 AA 的特权分配在多个证书中完成,则扩展将包括多个指针。

该扩展可能存在于 AA 发布给其他 AA 或终端实体特权持有者的属性证书或公钥证书中。该扩展不应包含在由 SOA 发布的证书或包含 SOA 标识符扩展的公钥证书里。

该扩展总是非关键的。

15.5.2.4.1 AA 标识符匹配

授权属性标识符匹配规则将当前值与 AttributeCertificate 属性值进行比较,看它们是否相等:

```
authAttIdMatch MATCHING-RULE      ::= {
    SYNTAX                  AuthorityAttirbuteIdentifierSyntax
    ID                      id-mr-authAttIdMatch }
```

如果存储值包含 authorityAttributeIdentifier 扩展,并且当前值中出现的组件与存储值的相应组件匹配,则匹配规则返回 TRUE。

16 特权路径处理过程

特权路径处理由特权验证者来执行。属性证书的路径处理规则与公钥证书的路径处理规则相似。

路径处理的其他组件还包括证书签名的验证、证书有效期的确认等,在此没有进行进一步阐述。

对于由单一证书组成的特权路径(也就是说,SOA 直接将特权分配给特权声明者),只需要 16.1 中描述的基本过程,除非已将特权分配给了角色。在这种情况下,如果不用指定的角色特权来配置特权验证者,则可能需要获取为角色分配特定特权的角色说明证书,见 16.2 中的描述。如果特权声明者的特权是由中间 AA 授予的,则需要 16.3 中描述的委托路径过程。这些过程不是连续执行的。角色处理过程和授权处理过程是在确定所声明特权在基本过程的使用环境中是否足够之前完成的。

16.1 基本处理过程

必须验证路径中每一个证书的签名。确认签名和公钥证书的过程在这里不再重复。特权验证者必须使用第 10 项中的过程,验证路径中每一个实体的身份。注意检查属性证书上的签名必然包括检查所引用公钥证书的有效期。在使用属性证书分配特权的时候,路径处理引擎必须在决定特权声明者的属性证书的最终有效期过程中考虑 PMI 和 PKI 的原理。一旦这个有效期得到了确认,根据相关的特权策略以及证书使用环境相关信息的比较,可以使用包含在证书中的特权。

所使用的环境必须确定特权持有者是否真正想声明在该环境中所使用的特权。用可信 SOA 证书链来进行确定,从本质上来说是不够的。必须明确说明和验证使用证书的特权持有者。但是,确保特权持有者已经对这种特权声明进行了充分论证的机制不在本部分讨论范围之内。例如,如果特权持有者签发了对证书的一个引用,这种特权声明必须是可验证的,从而说明他们在该环境中使用证书。

对于路径中不包含 noRevAvail 扩展的属性证书,特权验证者必须确保属性证书没有被撤销。

特权验证者必须确保所声明的特权对于所谓的“评估时间”是有效的,“评估时间”可在任何时间进

行,也就是说,当前正在检查的时间,或者过去的任何时间。在访问控制服务环境中,检查总是针对当前的时间。但是,在不可抵赖环境中,可以为过去或者现在的时间进行检查。确认证书时,特权验证者必须确保评估时间落在路径中所有证书的有效期内。并且,如果路径中的任一证书包含 timeSpecification 扩展,则对可声明特权的时间约束在评估时间内必须允许特权声明有效。

如果 targetingInformation 扩展出现在声明特权的证书中,则特权验证者必须检查它所验证的服务器/服务是否包含在目标列表中。

如果证书是一个角色分配证书,则需要用 16.2 中描述的处理过程来确保特权得到了验证。如果特权授给了实体,但不是由特权验证者信任的 SOA 直接分配的,则需要用 16.3 中描述的处理过程来确保授权行为是正确的。

特权验证者必须确定所声明的特权对于所使用的环境是否是足够的。特权策略建立了作出这个确定的规则,还包括了需要考虑的任何环境变量的规范。所声明的特权,包括 16.2 中角色过程的结果,16.3 中的授权过程,以及任何相关的环境变量(例如,一天的时间或者当前的帐目结算),与特权策略进行比较,以此来决定它们对于所使用的环境是否足够。如果 acceptablePrivilegePolicies 扩展存在,则仅当与特权验证者相比较的特权策略包含在扩展中时,特权声明才能成功。

如果比较成功,则特权验证者可以得到任何相关的用户通知。

16.2 角色处理过程

如果声明的证书是角色分配证书,则特权验证者必须获得分配给该角色的指定特权。得到特权声明者的角色名包含在证书的角色属性中。如果不用指定角色的特权来配置,特权验证者可能需要查找分配特权给该角色的角色说明证书的位置。角色属性和 roleSpecCertIdentifier 扩展中的信息可用来查找证书的位置。

分配给角色的特权都隐含地分配给了特权声明者,因此也就包含在所声明的特权中,它们通过与 16.1 中基本过程中的特权策略进行比较,来决定所声明的特权对于所使用的环境是否足够。

16.3 授权处理过程

如果所声明的特权是由中间 AA 授予给特权声明者的,则特权验证者必须保证路径是有效的委托路径,它是确保通过下列操作而实现的:

——授权给在委托路径上发布证书的每个 AA 可进行这些操作;

——对于强加在其上的路径和名称约束来说,委托路径中的每个证书都是有效的;

——委托路径中的每个实体都得到了公钥证书的验证,该公钥证书对于任何强加的策略约束来说是有效的;

——AA 的授权特权不会比它所持有的特权高。

在委托路径确认之前,特权验证者必须获得如下信息。它们中的任何一个都可以由特权声明者提供,或者由特权声明者从其他的源获得,例如目录。服务属性可以以结构或者其他方式提供给特权验证者。

——在公共验证密钥中建立信任,用来确认可信的 SOA 签名。可以通过带外方式或通过 CA 发布给 SOA 的公钥证书来建立信任,在这个 CA 中特权验证者已经建立了信任。这样的证书包含 sOAIdentifier 扩展。

——特权声明者的特权,在其属性证书,或公钥证书的主体目录属性扩展中进行编码。

——证书的委托路径,从特权声明者到可信 SOA。

——所声明特权的控制规则,可从 SOA 发布的属性描述符中获得或通过带外方式获得,该 SOA 负责所讨论的属性。

——特权策略,可从目录或带外方式获得。

——环境变量,包括当前的日期/时间,当前的帐目结算等。

实现在功能上应与该过程所产生的外部行为相当,然而,用来从给定输入获得正确输出的特定实现

的算法并没有被标准化。

16.3.1 验证控制规则的完整性

控制规则与授予的特权相关。获得控制规则的语法和方法没有被标准化,但可以验证恢复控制规则的完整性。负责授予属性的 SOA 发布的属性描述符证书可能包含控制规则的 HASH。特权验证者可以在控制规则的副本上再次生成 HASH,并将这两个 HASH 进行比较,如果它们相同,那么验证者拥有精确的控制规则。

16.3.2 建立有效的委托路径

特权验证者必须找到委托路径,并获得路径中每一个实体的证书。委托路径从直接特权声明者延伸到 SOA。委托路径中的每一个中间证书必须包含 basicAttConstraints 扩展,其授权组件设置为 TRUE。每个证书的发布者应该与证书的持有者/主体相同,在委托路径中它们是相邻的。authorityAttributeIdentifier 扩展用于查找委托路径中临近主体的相应证书。路径中从实体到其直接特权声明者(包括在内)的证书数目不能比实体的 basicAttConstraints 扩展中 pathLenConstraint 的数值大 2 以上。这是因为 pathLenConstraint 限制了这两个端点(即,包含约束的证书和终端实体证书)之间的中间证书数,所以最大的长度是约束值加上终端证书。

如果 delegatedNameConstraints 扩展出现在委托路径中的任何一个证书中,那么约束的处理方法和第 10 章中证书路径处理过程中的 nameConstraints 扩展的处理方法相同。

如果 acceptableCertPolicies 扩展在委托路径中的任何一个证书中出现,特权验证者必须确保委托路径中的每一个后续实体的验证是用公钥证书来完成的,该公钥证书至少包含一个合理的策略。

16.3.3 验证特权授权

授权者不能授予比自身所拥有特权更高的特权。属性描述符属性中的控制规则提供了这样的规则:所授予属性的一个给定值小于另一个值。

对于委托路径中的每一个证书(包括直接特权声明者的证书)来说,特权验证者必须保证授权者有权授予这个特权,并且所授予的特权不大于它自身拥有的特权。

对于每一个证书来说,特权验证者必须将授权者所授予的特权与它自身所拥有的特权进行比较,它符合特权控制规则。授权者所拥有的特权是从委托路径中临近的证书获得的,见 16.2 所述。两个特权的比较是按 16.3.1 中的控制规则进行的。

16.3.4 通过/失败确定

假设建立了一条有效的委托路径,则直接特权声明者的特权将被作为输入来与 16.1 中所讨论的特权策略进行比较,以此确定直接的特权声明者对于使用的环境是否有足够的权限。

17 PMI 目录模式

本章定义了用于描述目录中 PMI 信息的目录模式元素。它包括相关对象类、属性和属性值匹配规则的规范。

17.1 PMI 目录对象类

定义了目录中描述 PMI 对象的对象类。

17.1.1 PMI 用户对象类

PMI 用户对象类在定义对象入口中使用,这个对象可以是属性证书的持有者。

```
pmiUser OBJECT-CLASS ::= {
    ---- 一个 PMI 用户(例如,一个 "holder")
    SUBCLASS OF          {top}
    KIND                 auxiliary
    MAY CONTAIN          {attributeCertificateAttribute}
    ID                   id-oc-pmiUser}
```

17.1.2 **PMI AA 对象类**

PMI AA 对象类用来定义作为属性权威的对象入口。

```
pmiAA OBJECT-CLASS ::= {
        ——一个 PMI AA
    SUBCLASS OF     {top}
    KIND            auxiliary
    MAY CONTAIN     {aACertificate |
                     attributeCertificateRevocationList |
                     attributeAuthorityRevocationList}
  ID                id-oc-pmiAA }
```

17.1.3 **PMI SOA 对象类**

PMI SOA 对象类用来定义作为权威源的对象入口。要注意的是，如果通过发布包括 sOAldentifie 扩展的公钥证书，授予对象 SOA 权限，那么描述该对象的目录入口也应该包含 pkiCA 对象类。

```
pmiSOA OBJECT-CLASS ::= {      —— 一个 PMI Surce of Authority
    SUBCLASS OF         {top}
    KIND                auxiliary
    MAY CONTAIN         {attributeCertificateRevocationList |
                         attributeAuthorityRevocationList |
                         attributeDescriptorCertificate}
  ID                    id-oc-pmiSOA }
```

17.1.4 **属性证书 CRL 分布点对象类**

属性证书 CRL 分布点对象类用于定义对象的入口，对象中包含属性证书和/或撤销列表分段。在说明入口时，这个辅助类将与 crlDistributionPoint 结构对象类进行组合。由于在这个类中 certificateRevocationList 和 authorityRevocationList 属性是可选的，因此它可能会创建入口，可能仅包括属性权威撤销列表或多种类型的撤销列表的入口，这些都取决于具体的需求。

```
attCertCRLDistributionPt    OBJECT-CLASS ::= {
    SUBCLASS OF         {top}
    KIND                auxiliary
    MAY CONTAIN         {attributeCertificateRevocationList |
                         attributeAuthorityRevocationList}
  ID                    id-oc-attCertCRLDistributionPts}
```

17.1.5 **PMI 委托路径对象类**

PMI 委托路径对象类用来为包含委托路径的对象定义入口。它通常与结构化对象类 pmiAA 的入口一起使用。

```
pmiDelegationPath       OBJECT-CLASS        ::= {
    SUBCLASS OF         {top}
    KIND                auxiliary
    MAY CONTAIN         {delegationPath}
    ID                  id-oc-pmiDelegationPath}
```

17.1.6 **特权策略对象类**

特权策略对象类用于定义对象的入口，该对象包含特权策略信息：

```
privilegePolicy         OBJECT-CLASS    ::= {
```


国家图书馆专用

```
    SUBCLASS OF          {top }
    KIND                 auxiliary
    MAY CONTAIN          {privPolicy }
    ID                   id-oc-privilegePolicy }
```

17.2 PMI 目录属性

定义了目录属性,它用于在目录入口中存储 PMI 数据。

17.2.1 属性证书的属性

以下属性包含属性证书,该证书发布给特定持有者,并且存储在此持有者的目录入口中。

```
attributeCertificateAttribute ATTRIBUTE      ::= {
    WITH SYNTAX                              AttributeCertificate
    EQUALITY MATCHING RULE                   attributeCertificateExactMatch
    ID                                       id-at-attributeCertificate }
```

17.2.2 AA 证书属性

下面的属性包含发布给 AA 的属性证书,并且它被保存在此持有者 AA 的目录入口中。

```
aACertificate          ATTRIBUTE      ::=      {
    WITH SYNTAX                               AttributeCertificate
    EQUALITY MATCHING RULE                    attributeCertificateExactMatch
    ID                                        id-at-aACertificate }
```

17.2.3 属性描述符证书属性

下面的属性包含由 SOA 发布的属性证书,它包含 attributeDescriptor 扩展。这些属性证书包含了有效的语法和特殊属性控制规则的详细内容,并存储在发布的 SOA 目录入口中。

```
attributeDescriptorCertificate     ATTRIBUTE      ::= {
    WITH SYNTAX                                   AttributeCertificate
    EQUALITY MATCHING RULE                        attributeCertificateExactMatch
    ID                                 id-at-attributeDescriptorCertificate }
```

17.2.4 属性证书撤销列表属性

下面的属性包含了属性证书撤销列表。这些列表可以存储在发布权威的目录入口或其他目录入口(例如,一个分布点)。

```
attributeCertificateRevocationList          ATTRIBUTE ::= {
    WITH SYNTAX                             CertificateList
    EQUALITY MATCHING RULE                  certificateListExactMatch
    ID                                      id-at-attributeCertificateRevocationList }
```

17.2.5 AA 证书撤销列表属性

下面的属性包含了发布给 AA 的属性证书撤销列表。这些列表能存储在发布权威的目录入口或其他目录入口(如,一个分布点)。

```
attributeAuthorityRevocationList            ATTRIBUTE      ::= {
    WITH SYNTAX                             CertificateList
    EQUALITY MATCHING RULE                  certificateListExactMatch
    ID                                      id-at-attributeAuthorityRevocationList }
```

17.2.6 委托路径属性

委托路径属性包含委托路径,每个路径由一系列属性证书组成。

```
delegationPath              ATTRIBUTE    ::= {
    WITH SYNTAX             AttCertPath
```

国家图书馆专用

ID **id-at-delegationPath** }

AttCertPath ::= **SEQUENCE OF AttributeCertificate**

这个属性存储在 AA 目录入口,并且包含从这个 AA 到其他 AA 的委托路径。如果使用了这个属性,它能更有效的恢复授权属性证书,这些授权属性证书形成了常用的委托路径。如果对这个属性的使用没有特殊的要求,则存储在属性里的值的集合不一定描述了任何给定 AA 的完整委托路径。

17.2.7 特权策略属性

特权策略属性包含特权策略的信息。

privPolicy **ATTRIBUTE** ::= {
WITH SYNTAX **PolicySyntax**
ID **id-at-privPolicy** }

Policydentifier 组件包括特殊特权策略注册的对象标识符。

如果 content 出现,则包含特权策略的完整内容。

如果 pointer 出现,则 name 组件引用了一个或多个位置,可通过此位置找到特权策略的副本。

如果 hash 组件存在,则它包含特殊策略内容的 HASH,并能在引用位置找到。这个 hash 可用来执行所使用文档的完整性检查。

17.3 PMI 普通目录匹配规则

定义了 PMI 目录属性的匹配规则。

17.3.1 属性证书精确匹配

属性证书精确匹配规则比较当前值与 AttributeCertificate 类型的属性值是否相等。

attributeCertificateExactMatch **MATCHING-RULE** ::= {
SYNTAX **AttributeCertificateExactAssertion**
ID **id-mr-attributeCertificateExactMatch** }
AttributeCertificateExactAssertion ::= **SEQUENCE** {
serialNumber **CertificateSerialNumber,OPTIONAL,**
issuer **IssuerSerial** }

如果属性值中的组件与当前值的相应组件匹配,则匹配规则返回 TRUE。

17.3.2 属性证书匹配

属性证书匹配规则将当前值和 AttributeCertificate 属性值进行比较。匹配规则允许比 certificateExactMatch 更复杂的匹配。

attributeCertificateMatch **MATCHING-RULE** ::= {
SYNTAX **AttributeCertificateAssertion**
ID **id-mr-attributeCertificateMatch** }
AttributeCertificateAssertion ::= **SEQUENCE** {
holder [0] **CHOICE** {
baseCertificateID [0] **IssuerSerial,**
holdertName [1] **GeneralNames** }**OPTIONAL,**
issuer [1] **GeneralNames OPTIONAL,**
attCertValidity [2] **GeneralizedTime OPTIONAL,**
attType [3] **SET OF AttributeType OPTIONAL** }

—至少存在一个组件

如果当前值中的所有组件和属性值中的相应组件都匹配,则匹配规则返回 TRUE,如下:

—— 如果 baseCertificateID 与存储属性值的 IssuerSerial 组件相等,则它匹配;

国家图书馆专用

—— 如果存储属性值包含与当前值中名称类型相同的名称扩展，则 holderName 匹配；

—— 存储属性值包含与当前值中相同名称类型的名称组件，则 issuer 匹配；

—— 如果 attCertValidity 落在存储属性值指定的有效期内，则它匹配；

—— 对于当前值中的每个 attType 来说，存储值的 attributes 组件中都存在一个该类型的属性。

17.3.3 持有者发布者匹配

属性证书持有者发布者匹配规则将持有者的当前值和/或当前值的发布者组件与 AttributeCertificate 类型的属性值进行比较，看其是否相等。

```
holderIssuerMatch MATCHING-RULE      ::= {
    SYNTAX                  HolderIssuerAssertion
    ID                      id-mr-holderIssuerMatch }
HolderIssuerAssertion       ::=         SEQUENCE {
    holder                  [0]     Holder          OPTIONAL,
    issuer                  [1]     AttCertIssuer   OPTIONAL }
```

如果当前值中出现的所有组件都匹配属性值中的相应组件，则匹配规则返回 TRUE。

17.3.4 委托路径匹配

delegationPathMatch 匹配规则将当前值和 delegationPath 类型的属性值进行比较，看其是否相等。特权验证者可用这个匹配规则选择一条以 SOA 发布的证书为起点，以发布给 AA 的证书为终点的路径，该 AA 发布给终端实体持有者的证书是有效的。

```
delegationPathMatch MATCHING-RULE      ::= {
    SYNTAX                  DelMatchSyntax
    ID                      id-mr-delegationPathMatch }

DelMatchSyntax              ::=     SEQUENCE {
    firstIssuer             AttCertIssuer,
    lastHolder              Holder }
```

如果 firstIssuer 组件中的当前值与存储值中 SEQUENCE 中的第一个证书发布者域的相应元素匹配，并且 lastHolder 组件中的当前值与存储值中 SEQUENCE 中的最后证书的相应持有者域元素匹配，则匹配规则返回 TRUE。如果两个中有一个不匹配，则匹配规则返回 FALSE。

国家图书馆专用

第四篇　公钥目录的使用和属性证书框架

目录使用公钥证书框架作为许多安全服务的基础，包括强鉴别、目录操作保护、以及存储数据保护。目录使用证书属性框架作为基于规则访问控制模式的基础。此处定义了不同的目录安全服务的公钥证书框架和属性证书框架的元素间的联系。由目录提供的特殊安全服务在这套完整的目录规范中有详细的说明。

18　目录鉴别

目录支持通过 DUA 访问目录的用户鉴别，以及目录系统对用户和其他 DSA 的鉴别。根据环境的不同，可以使用弱鉴别或强鉴别。以下描述了目录中弱鉴别或强鉴别的规程。

18.1　弱鉴别规程

弱鉴别的目的是提供本地授权，该授权是建立在用户可辨别名、双方同意的（可选）口令、以及在某个单一区域中双方都能理解的口令的使用和处理方法之上。弱鉴别一般只用于本地的对等实体，即一个 DUA 和一个 DSA 之间、或一个 DSA 与另一个 DSA 之间的鉴别。通常可采用以下几种方法实现弱鉴别：

a) 以明文(无保护)的方式将用户的可辨别名和(可选的)口令传送给接收方考察;

b) 将用户的可辨别名、口令,以及一个随机数和/或时间戳一起通过使用单向函数进行保护,并传送;

c) 将 b)中描述的保护信息连同一个随机数和/或时间戳一起通过使用单向函数进行保护,并传送。

注 1:不要求一定使用不同的单向函数。

注 2:用于保护口令的过程可能需要对本部分进行扩展。

如果口令没有被保护,则只提供了最低限度的安全保护以防止未授权的访问,不应将其看作安全服务的基础。对用户的可辨别名和口令的保护则提供更高级别的安全。用于保护机制的算法通常是极易实现的非加密单向函数。

图 5 给出了进行弱鉴别的一般过程。

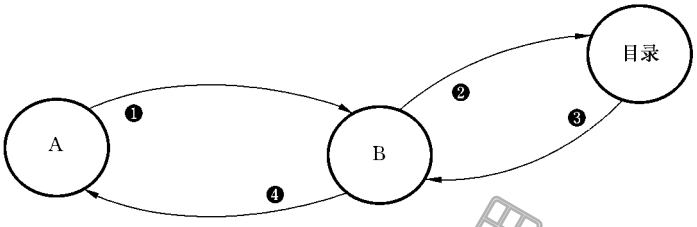


图 5 无保护的弱鉴别规程

弱鉴别规程一般包括以下几个步骤:

a) 发方用户 A 将其可辨别名和口令发送给收方用户 B;

b) B 将 A 声称的可辨别名和口令发送给目录,然后目录将该口令与作为 A 目录项中的 User-Password 属性保存的口令(采用目录的比较运算)进行比较;

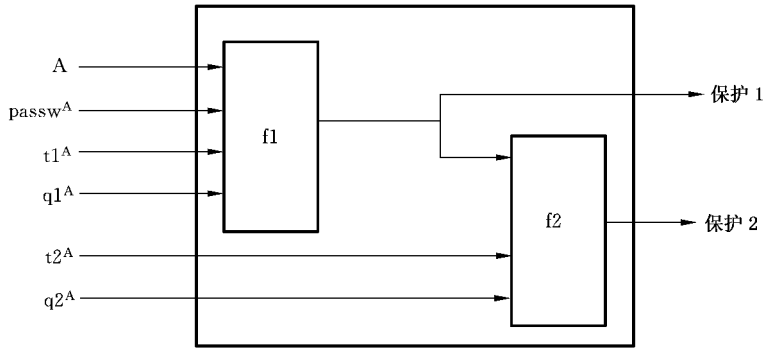
c) 目录对凭证有效性向 B 进行确认(或否认);

d) 鉴别的成功(或失败)可以传送给 A。

弱鉴别最基本的形式只包含步骤 a),并且在 B 检查完可辨别名和口令之后也可包含步骤 d)。

18.1.1 有保护标识信息的生成

图 6 给出了可以生成有保护标识信息的两种方法。其中,f1 和 f2 为单向函数(它们可以相同,也可以不同),且时间戳和随机数为可选项,并服从双边协定。



A 用户的可辨别名

t^A 时间戳

$Passw^A$ A 的口令

q^A 随机数,可选用计数器

图 6 有保护的弱鉴别

国家图书馆专用

18.1.2 有保护的弱鉴别规程

图 7 给出了有保护的弱鉴别规程。

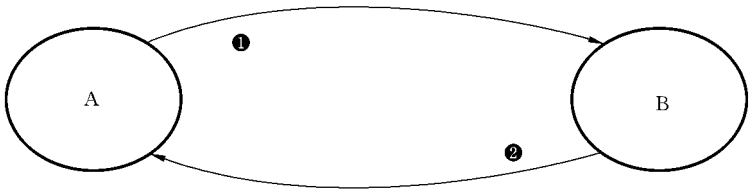


图 7 有保护的弱鉴别规程

有保护的弱鉴别过程包括以下步骤(最初只使用 f1):

a) 发方用户 A,向用户 B 发送其有保护的标识信息(鉴别符 1)。通过使用图 2 中的单向函数 f1 获得保护,其中时间戳和/或随机数(当使用时)用于使重放最少和隐藏口令。

A 口令的保护形式如下:

$$保护\ 1 = f1(t1^A, q1^A, A, passwA)$$

传送给 B 的信息的形式如下:

$$鉴别符\ 1 = t1^A, q1^A, A, 保护\ 1。$$

b) B 通过(使用由 A 提供的可辨别名、可选的时间戳和/或随机数连同 A 口令的本地副本)生成(形式为保护 1 的)A 口令的本地保护副本来验证由 A 提供的有保护的标识信息。B 比较所声称的标识信息(保护 1)和本地生成的值是否相等。

c) B 将核对保护标识信息的结果(证实或否认)返回给 A。

使用 f1 和 f2 可对所描述的鉴别过程进行修改以提供更强的保护,其主要区别如下:

a) A 将其附加保护的标识信息(鉴别符 2)发送给 B。通过使用图 2 中的单向函数 f2 来获得附加保护。其附加保护形式如下:

$$保护\ 2 = f2(t2^A, q2^A, 保护\ 1)$$

传送给 B 的信息形式如下:

$$鉴别符\ 2 = t1^A, t2^A, q1^A, q2^A, A, 保护\ 2$$

为了比较用户 B 生成用户 A 的附加保护口令的本地值,并比较该值和保护 2 的值是否相等。(类似于 6.2 的步骤 2))。

b) B 将核对保护标识信息的结果(证实或否认)返回给 A。

注:在这些条款中定义的规程都是利用 A 或 B 来描述的。对于目录(由 ITU-T X.511|ISO/IEC 9594-3 和 ITU-T X.518|ISO/IEC 9594-4 规定)来说,A 可以是与一个 DSA(B)相绑定的 DUA;另一情况,A 可以是与另一个 DSA(B)相绑定的 DSA。

18.1.3 用户口令属性类型

用户口令属性类型包含一个客体的口令。该用户口令的属性值则为由该客体指定的一个字符串。

```
userPassword ATTRIBUTE    ::=    {
    WITH SYNTAX                    OCTET STRING (SIZE (0..ub-user-password))
    EQUALITY MATCHING RULE         octetStringMatch
    ID                             id-at-userPassword }
```

18.2 强鉴别

这部分描述用于 DUA 和 DSA 之间,以及 DSA 对之间的鉴别。此过程使用了标准中定义的公钥证书框架。另外,此过程使用了目录,将其作为用于执行鉴别的必需公钥信息的存储库。在协议规范中定义了目录协议中的相关参数。用于定义强认证的过程也可用于除目录以外的应用,该目录也使用了这个存储库。对于过程中目录的使用来说,术语"用户"可参考 DUA 或 DSA。

本目录规范中采用的强鉴别方法是利用一簇密码体制的特性,即通常所知的公钥密码体制(PKCS)来实现的。这些密码体制,也称非对称密码体制,包含一对密钥,其中一个为用户私有的,而另一个则为公开的;它不同于传统密码体制中使用的单一密钥。在附录C中简要介绍这些非对称密码系统以及在鉴别过程中有用的特性。目前可用于本鉴别框架PKCS应该具备这样的特性,即密钥对中的两个密钥都可用于加密,如果公钥用于加密则私钥用于解密,如果私钥用于加密则公钥用于解密;换句话说,它们必须满足Xp·Xs=Xs·Xp,这里,Xp/Xs为用户X的公钥/私钥的加密/解密函数。

注:PKCS的替换类型(即不要求具有可置换特性,并能在不对本目录规范作大的修改的情况下得到支持)可能是今后的扩展。

该鉴别框架并不强制使用某个特定的密码体制。本框架应适用于任何合适的公开密码体制,并能支持今后对密码学、数学技术、或可计算能力方面发展所带来的所用方法的改变。然而两个想要相互鉴别的用户必须支持相同的密码算法,才能正确地执行鉴别。因此,在一组相关应用的上下文中,选择一个单一的算法,可以最大可能地增加用户间相互鉴别和安全通信的能力。

鉴别取决于每个具有唯一可辨别名的用户。可辨别名的分配由命名机构负责。每一个用户都应相信命名机构不会发出重复的可辨别名。

每个用户都可用其所拥有的私钥来标识。另一个用户则可根据其通信伙伴是否拥有这个私钥来确定他是否确实为(授权)用户。这种确证的有效性取决于只有用户才拥有该私钥。

一个用户若要确定其通信对方是否拥有其私钥,他自己就必须拥有对方的公钥。用户的公钥的值可以直接从目录的用户项中获得,但要验证其正确性却更困难一些。有许多可能的方法来验证用户的公钥。第8章描述了通过引用目录来验证用户公钥的操作过程。该过程只在请求鉴别的用户之间的目录中存在一条不间断的信任链的情况下进行。这样的链可通过标识一个公共信任点来构造。该公共信任点应通过一条不间断的信任链与每个用户相链接。

18.2.1 从目录中取得公钥证书

证书作为属性存放在目录入口中,这些属性分为三种类型:UserCertificate,CACertificate和CrossCertificatePair。目录可以识别这些属性。和其他属性一样,这些属性可用相同的协议来进行操作。这些类型的定义见3.3;这些属性类型的描述在11.2中定义。

通常,在用户能够相互鉴别之前,目录应提供完整的鉴别,并返回证书认证路径。但在实际操作过程中,对某个特定的鉴别实例来讲,通过如下途径可以减少从目录中获得的信息量:

a) 如果两个想要鉴别的用户具有同一个证书认证机构,那么证书认证路径将变得毫无价值,而且用户可以相互直接打开彼此的证书;

b) 如果用户的CA是按层次安排的,那么一个用户可以存储用户与DIT根之间的所有证书认证机构的公钥、证书和反向证书。作为一种典型的情况,可能使用户涉及到三个或四个证书认证机构的公钥和证书的用户。该用户只需要获得来自公共信任点的证书认证路径。

c) 如果一个用户频繁地与被某个特定的其他CA认证的用户通信,则该用户只须从目录中获得该用户的证书,从而取得从本地到那个CA的证书认证路径,并从这个CA返回这条证书认证路径。

d) 证书认证机构可以通过双边协定彼此进行交叉认证,从而可以缩短证书认证路径。

e) 如果两个用户以前曾经相互通信过,并且彼此已取得对方的证书,则他们无需目录资源就能相互鉴别。

不论是哪一种情况,用户从证书认证路径中取得彼此的证书后,应检查收到的证书的有效性。

18.2.1.1 实例

图8给出了假设的DIT段的举例,这里,若干CA形成了层次结构。除了知道CA的信息外,还假定每一个用户都知道其证书认证机构的公钥,以及他自己的公钥和私钥。

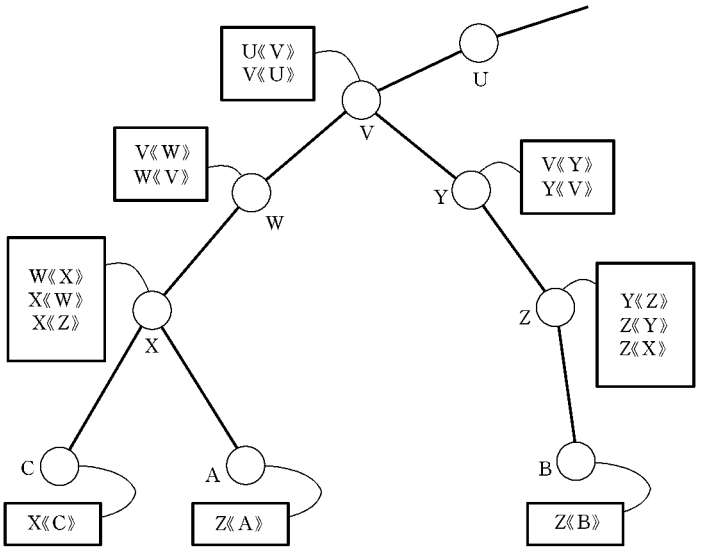


图 8　CA 的层次结构一个假设举例

如果用户的 CA 是按层次结构安排的，则 A 可以从目录中得到下列证书，以建立到 B 的证书认证路径：

X《W》，W《V》，V《Y》，Y《Z》，Z《B》

当 A 已经得到这些证书时，则可以按次序打开这个证书认证路径，进而得出 A（包括 Ap）的证书的内容：

Bp＝Xp・X《W》W《V》V《Y》Y《Z》Z《B》

一般情况下，A 还应从目录中得到下列证书，以建立从 B 到 A 的反向证书认证路径：

Z《Y》，Y《V》，V《W》，W《X》，X《A》

当 B 从 A 收到这些证书时，则可以按次序打开这个反向证书认证路径，进而得出 B（包括 Ap）的证书的内容：

Ap＝Zp・Z《Y》Y《V》V《W》W《X》X《A》

可对 18.2.1 使用优化：

a）　例如，A 和 C 都知道 X 的公钥 Xp，因此，A 只须从目录中直接得到 C 的证书。需打开的证书认证路径可以缩减为：

Cp＝Xp・X《C》

并且，需打开的反向证书认证路径也可以缩减为：

Ap＝Xp・X《A》

b）　假定 A 知道 W《X》、Wp、V《W》、Vp、U《V》、Up 等等，则从目录中获取的用于构造证书认证路径的信息可以减少为：

V《Y》、Y《Z》、Z《B》

并且，从目录中获取的用于构造反向证书认证路径的信息为：

Z《Y》、Y《V》

c）　假定 A 与由 Z 认证的用户通信频繁，则他除了已知道上面 b）中的各个公钥外，还能知道 V《Y》，Y《V》，Y《Z》，和 Z《Y》。因此，为与 B 通信，A 只须从目录中获得 Z《B》即可。

d）　假定进行频繁通信的用户都由 X 和 Z 认证，那么在目录中，X 的目录项应持有 X《Z》，反之亦然，Z 的目录项应持有 Z《X》（见图 8）。如果 A 想要鉴别 B，A 只需获得：

X《Z》、Z《B》

以构造证书认证路径；同样，只需获得：

Z《X》

即可构造反向证书认证路径。

e) 假定用户 A 和 C 以前曾经通信过，并且相互间已知彼此的证书，那么他们可以直接使用彼此的公钥；即：

Cp＝Xp・X《C》

Ap＝Xp・X《A》

在更一般的情况下，证书认证机构并不按层次结构相互联系。在图 9 中给出了一个假想的举例，在这个举例中，假定由 U 认证的用户 D 想要鉴别由 W 认证的用户 E。用户 D 的目录项持有证书 U《D》，用户 E 的目录持有证书 W《E》。

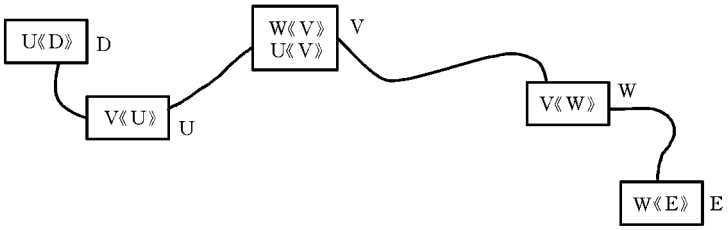


图 9 非层次结构的认证路径—举例

假设 V 为一个 CA，证书认证机构 U 和 W 曾通过 V 按照可信任的途径相互交换过彼此的公钥。其操作结果是生成了证书 U《V》、V《U》、W《V》和 V《W》，并已存入目录中。假定 U《V》、W《V》存在于 V 的目录项中，V《U》存在于 U 的目录项中，而 V《W》存在于 W 的目录项中。

用户 D 必须找出到用户 E 的证书认证路径。有几种方法可供使用。其中之一就是将用户和 CA 看作结点，而证书则为有向曲线图上的弧。在这里，D 应在曲线图上执行一次搜索以找到一条从 U 到 E 的路径，例如，U《V》、V《W》、W《E》。当找到这条路径以后，其反向路径亦可据此构造出来，即 W《V》、V《U》、U《D》。

18.2.2 强鉴别规程

鉴别的基本方法在前面已有论述，通过出示所拥有的私钥来证实其身份。然而，这样可能会出现许多采用这种方法的鉴别规程。通常，使用何种恰当的规程取决于特定的应用环境，以满足该应用的安全策略要求。本章描述了三种在一定范围内很有用的特定的鉴别规程。

注：本目录标准并不规定实现规程的细节。但是，可以设想一些附加的标准（它们可以是专用的，也可以是通用的方式）来达此目的。

这三种规程包括许多不同鉴别信息的交换，并为它们的参与者提供不同类型的保证。特别地，

a) 单向鉴别在 18.2.2.1 中描述包含从一个用户（A）到另一个目标用户（B）的信息的单次传送，并建立：

A 的身份，和实际由 A 产生的鉴别令牌；

B 的身份，和实际发送给 B 的鉴别令牌；

正被传送的鉴别令牌的完整性和“始发性”（即，没有被发送两次或多次的特性）。

后者也可为伴随传送的任何附加数据一起建立。

b) 双向鉴别（在 18.2.2.2 中描述）包含其他从 B 到 A 的回答。它建立：

在回答中实际由 B 产生的并发送给 A 的鉴别令牌；

在回答中发送的鉴别令牌的完整性和始发性；

（可选）令牌的部分的相互保密性。

c) 三向鉴别（在 18.2.2.3 中描述），另外还包含从 A 到 B 的进一步传送。它建立与双向鉴别相同的特性，但不必联系时间戳检查来做这件事。

在进行强鉴别的任何一种情况下，A 应该获得 B 的公钥，并在进行任何信息交换之前返回从 B 到

A 的证书认证路径。这可能包含 18.2 中描述的对目录的访问。在下面的规程的描述中不再提到这种对目录的访问。

只有当本地环境使用了同步时钟，或者，如果时钟是通过双边协定的逻辑同步，那么，时间戳的检查将在下面的章条中提到。建议使用国际协调时。

对于下述任何一种鉴别规程来说，假设 A 实体已检查了证书认证路径中的所有证书的有效性。

18.2.2.1 单向鉴别

见图 10，包括以下步骤：

1) A 产生 r^A，一个不重复的数，它用于检测重放攻击并防止伪造签名。

2) A 向 B 发送下面消息：

$$B \rightarrow A,\ A\{t^A,\ r^A,\ B\}$$

这里，t^A 是一个时间戳。t^A 由一个或两个日期组成：令牌的生成时间（可选）和期满日期。另外，如果“sgnData”的数据原发鉴别由数字签名者提供，则发送：

$$B \rightarrow A,\ A\{t^A,\ r^A,\ B,\ sgnData\}$$

如果被传送的信息随后会被用在私钥运送的情况下，（该信息一般称作“encData”）则发送：

$$B \rightarrow A,\ A\{t^A,\ r^A,\ B,\ sgnData,\ Bp[encData]\}$$

使用“encData”作为私钥意味着应对其进行仔细选择，例如，在令牌的“sgnData”字段中指定的密钥对于任何密码体制都是健壮密钥。

3) B 执行下列动作：

a) 获得从 B→A 的 Ap，并检查 A 的证书没有过期；

b) 核实签名，以保证被签名信息的完整性；

c) 检查 B 自己是否就是目标接收者；

d) 检查时间戳 t^A 是否为“当前”；

e) 可选检查 r^A 是否已被重放。例如，这可以通过在 r^A 中包含一个序号，并由本地实现来检查其值是否唯一。

r^A 只在由 t^A 指示的期满日期内有效。r^A 总与一个序号一起使用，该序号指示 A 在有效时间 t^A 范围内不能重复该令牌，因此，不需要检查 r^A 本身的值。

在任何一种情况下，B 都可以与时间戳 t^A 一起，并在 t^A 的许可时间范围内与令牌的散列部分一起存储其序号。

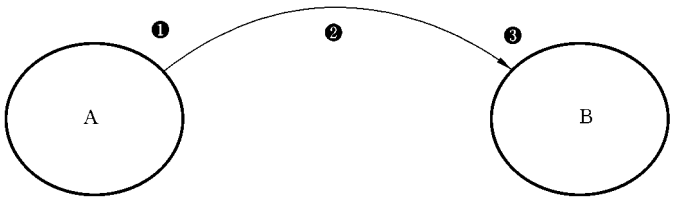


图 10 单向认证

18.2.2.2 双向鉴别

见图 11 的描述，包括以下步骤：

1) 同 18.2.2.1；

2) 同 18.2.2.1；

3) 同 18.2.2.1；

4) B 产生一个非重复的数 r^B，其使用目的与 r^A 类似。

5) B 向 A 发送以下鉴别令牌：

$B\{t^B,\ r^B,\ A,\ r^A\}$ 这里，t^B 是一个与 t^A 定义相同的时间戳；

如果要使用数字签名提供“sgnData”的数据原发鉴别，则发送：

国家图书馆专用

$B\{t^B, r^B, A, r^A, sgnData\}$

如果被传送的信息随后会被用在私钥运送的情况下,(该信息一般称作“encData”)则发送:

$B\{t^B, r^B, A, r^A, sgnData, Ap[encData]\}$

使用“encData”作为私钥意味着应对其进行仔细选择,例如,在令牌的“sgnData”字段中指定的密钥对于任何密码体制都是强密钥。

6) A 执行下列动作:

a) 核实签名,以保证被签名信息的完整性;

b) 检查 A 自己是否就是目标接收者;

c) 检查时间戳 t^B是否为“当前”;

d) (可选)检查 r^B是否已被重放。(见 18.2.2.1 步骤 3 的 d)。

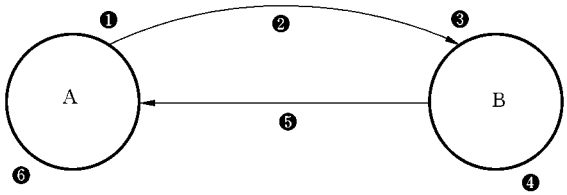


图 11 双向鉴别

18.2.2.3 三向鉴别

见图 12 的描述,包括以下步骤:

1) 同 18.2.2.2。

2) 同 18.2.2.2。时间戳 t^A可以是 0。

3) 除不需检查时间戳以外,同 18.2.2.2。

4) 同 18.2.2.2。

5) 同 18.2.2.2。时间戳 t^B 可以是 0。

6) 除不需检查时间戳以外,同 18.2.2.2。

7) A 检查接收的 r^A 是否与所发送的 r^A 相等。

8) A 向 B 发送下面鉴别令牌:

$A\{r^B, B\}$

9) B 执行以下动作:

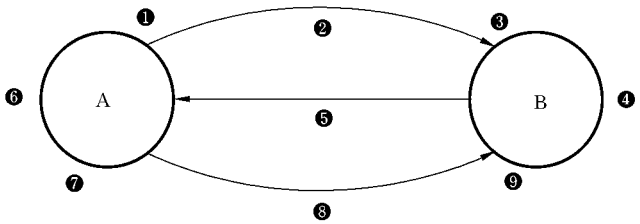


图 12 三向鉴别

a) 检查签名,以保证被签名信息的完整性;

b) 检查接收的 r^B 是否与 B 所发送的 r^B 相等。

19 访问控制

多种不同的管理授权环境下的目录控制对 DIB 部分的访问。目录环境下访问控制模式的定义包括以下方法:

● 指定访问控制信息;

国家图书馆专用

- 实施由访问控制信息定义的访问权力；
- 保存访问控制信息。

访问权限的实施应用于控制对以下各项的访问：

- 与名字相关的目录信息；
- 目录用户信息；
- 包括访问控制信息的目录操作信息。

机构可能使用全部或部分的标准化访问控制模式来执行安全策略，或者可以自由定义他们自己的模式。

ISO/IEC 9594-2 中定义的基础访问控制模式是一个基于模式的访问控制列表，它能使目录管理员将权限与绑定到目录上的认证级别联系起来。本部分中定义的公钥证书构架为这种绑定提供了强认证模式。

ISO/IEC 9594-2 中定义的基于规则的访问控制模式使用本部分中定义的属性证书构架来传输制定访问控制决策所用到的属性。基于规则的访问控制也可用于连接基础访问控制。

20 目录操作的保护

本部分所定义的公钥证书构架用于所有推荐标准定义的目录协议，此目录有选择性地保护请求操作，响应操作和错误操作。完整性保护由发送方的数字签名提供，且收件人通过使用相应的公钥证书验证签名。机密保护通过使用接收方的公钥证书加密来实现，并且通过收件人相应的私钥来解密。

在协议交换中保护元素包括或需要的特殊机制和语法在本系列标准的每个目录协议中分别定义。

国家图书馆借用

附 录 A
（资料性附录）
用 ASN.1 描述的鉴别框架

本附录包括了本目录标准当中所有的 ASN.1 类型、值和信息对象类定义，所有这些定义都以 AuthenticationFrameword、CertificateExtensions 和 AttributeCertificateDefinitions 这三个 ASN.1 模块的形式给出。

A.1 认证框架模型

```
AuthenticationFramework {joint-iso-itu-t ds(5) module(1) authenticationFramework(7) 4 }
DEFINITIONS ::=
BEGIN
-- 全部输出 --
-- 本模块中所定义的类型和值可输出，供本目录标准系列当中包含的其他 ASN.1 模块使用，同时也
-- 可供其他应用用于访问目录服务。其他应用可以将它们用于其各自的目的，但这并不会限制为维
-- 护和改进目录服务而进行的扩展和修改。
    输入
    id-at, id-nf, id-oc, informationFramework, upperBounds, selectedAttributeTypes, basicAccessControl,
    certificateExtensions
        FROM UsefulDefinitions {joint-iso-itu-t ds(5) module(1) usefulDefinitions(0) 4 }
    Name, ATTRIBUTE, OBJECT-CLASS, NAME-FORM, top
        FROM InformationFramework informationFramework
    ub-user-password, ub-content
        FROM UpperBounds upperBounds
    UniqueIdentifier, octetStringMatch, DirectoryString, commonName
        FROM SelectedAttributeTypes selectedAttributeTypes
    certificateExactMatch, certificatePairExactMatch, certificateListExactMatch, KeyUsage,
    GeneralNames,
        CertificatePoliciesSyntax, algorithmIdentifierMatch, CertPolicyid
        FROM CertificateExtensions certificateExtensions;
--公钥证书定义 --
    Certificate                 ::=     SIGNED {SEQUENCE {
        version                 [0]     Version DEFAULT v1,
        serialNumber                    CertificateSerialNumber,
        signature                       AlgorithmIdentifier,
        issuer                          Name,
        validity                        Validity,
        subject                         Name,
        subjectPublicKeyInfo            SubjectPublicKeyInfo,
        issuerUniqueIdentifier  [1]     IMPLICIT UniqueIdentifier OPTIONAL,
                                        --如果存在，版本必须为 v2 或 v3
        subjectUniqueIdentifier [2]     IMPLICIT UniqueIdentifier OPTIONAL,
```

国家图书馆专用

```
                                        ——如果存在,版本必须为 v2 或 v3
    extensions          [3]             Extensions OPTIONAL
                                        ——如果存在,必须是第三版——}}
Version                 ::=     INTEGER {v1(0), v2(1), v3(2) }
CertificateSerialNumber ::=     INTEGER
AlgorithmIdentifier     ::=     SEQUENCE {
    algorithm           ALGORITHM. &id ({SupportedAlgorithms }),
    parameters          ALGORITHM. &Type ({ SupportedAlgorithms } { @ algorithm })
                        OPTIONAL }

——下面的信息客体集合可能要交由已标准化的轮廓或者协议实现的一致性声明来定义。此集合应规
——定一个表,以约束 AlgorithmIdentifier 的 parameters 组分的可能取值。
SupportedAlgorithms     ALGORITHM       ::=  {... }
Validity                ::=     SEQUENCE {
    notBefore           Time,
    notAfter            Time }
SubjectPublicKeyInfo    ::=     SEQUENCE {
    algorithm           AlgorithmIdentifier,
    subjectPublicKey    BIT STRING }
Time    ::=  CHOICE {
    utcTime             UTCTime,
    generalizedTime     GeneralizedTime }
Extensions ::= SEQUENCE OF Extension

——如果对于某些 extensions 来说,SEQUENCE 当中单个 extension 的顺序是有意义的,那么定义那
——些单个 extension 的规范当中应该包含决定顺序的意义的规则。

Extension ::= SEQUENCE {
    extnId      EXTENSION. &id ({ExtensionSet }),
    critical    BOOLEAN DEFAULT FALSE,
    extnValue   OCTET STRING
                —— 包含由 extnID 所标识的扩展对象的类型为
                —— &ExtnType 的值的一个 DER 编码 —— }

ExtensionSet            EXTENSION       ::=  {  ... }
EXTENSION ::= CLASS {
    &id                 OBJECT IDENTIFIER UNIQUE,
    &ExtnType }
WITH SYNTAX {
    SYNTAX              &ExtnType
    IDENTIFIED BY       &id }
——其他 PKI 证书结构
Certificates                        ::=  SEQUENCE {
```

国家图书馆专用

```
        userCertificate                 Certificate,
        certificationPath               ForwardCertificationPath OPTIONAL }
    ForwardCertificationPath            ::=  SEQUENCE OF CrossCertificates
    CrossCertificates                   ::=  SET OF Certificate
    CertificationPath                   ::=  SEQUENCE {
        userCertificate                 Certificate,
        theCACertificates               SEQUENCE OF CertificatePair OPTIONAL }
    CertificatePair                     ::=  SEQUENCE {
        forward                         [0]  Certificate OPTIONAL,
        reverse                         [1]  Certificate OPTIONAL
                                        -- 至少应存在一对 -- }
        (WITH COMPONENTS {..., forward PRESENT }|
        WITH COMPONENTS {..., reverse PRESENT })

-- 证书撤销列表(CRL)
    CertificateList                     ::=  SIGNED {SEQUENCE {
        version                         Version OPTIONAL,
                                            -- 如果存在,必须是第二版
        signature                       AlgorithmIdentifier,
        issuer                          Name,
        thisUpdate                      Time,
        nextUpdate                      Time OPTIONAL,
        revokedCertificates             SEQUENCE OF SEQUENCE {
        serialNumber                        CertificateSerialNumber,
        revocationDate                      Time,
        crlEntryExtensions                  Extensions OPTIONAL }OPTIONAL,
        crlExtensions                   [0]             Extensions OPTIONAL }}

-- 信息对象类 --
    ALGORITHM                           ::=  TYPE-IDENTIFIER
-- 参数类型 --
    HASH {ToBeHashed }                      ::=         SEQUENCE {
        algorithmIdentifier                     AlgorithmIdentifier,
        hashValue                               BIT STRING ( CONSTRAINED BY {
            -- 必须是将散列程序应用于 DER 编码字节的结果 --
            -- ToBeHashed 的值 }) }
ENCRYPTED-HASH {ToBeSigned }    ::=  BIT STRING ( CONSTRAINED BY {
            -- 必须是将散列过程应用于 DER 编码字节(见 6.1 节)
            -- ToBeSigned 的值和将加密过程应用于该字节的结果 -- })
ENCRYPTED {ToBeEnciphered }     ::=  BIT STRING ( CONSTRAINED BY {
            -- 必须是使用加密过程的结果 --
            -- 八位 BER 编码值 -- ToBeEnciphered })
```

国家图书馆

```
SIGNATURE {ToBeSigned } ::= SEQUENCE {
    algorithmIdentifier     AlgorithmIdentifier,
    encrypted               ENCRYPTED-HASH {ToBeSigned }}

SIGNED {ToBeSigned } ::= SEQUENCE {
    toBeSigned              ToBeSigned,
    COMPONENTS OF           SIGNATURE {ToBeSigned }}
```

—— PKI 对象类 ——

```
pkiUser OBJECT-CLASS ::= {
    SUBCLASS OF     {top }
    KIND            auxiliary
    MAY CONTAIN     {userCertificate }
    ID              id-oc-pkiUser }

pkiCA OBJECT-CLASS ::= {
SUBCLASS OF         {top }
    KIND            auxiliary
    MAY CONTAIN {cACertificate |
                    certificateRevocationList |
                    authorityRevocationList |
                    crossCertificatePair }
    ID              id-oc-pkiCA }

cRLDistributionPoint    OBJECT-CLASS ::= {
    SUBCLASS OF             {top }
    KIND                    structural
    MUST CONTAIN            {commonName }
    MAY CONTAIN             {certificateRevocationList |
                            authorityRevocationList |
                            deltaRevocationList }
    ID                      id-oc-cRLDistributionPoint }

cRLDistPtNameForm   NAME-FORM ::= {
    NAMES                   cRLDistributionPoint
    WITH ATTRIBUTES         {commonName }
    ID                      id-nf-cRLDistPtNameForm }

deltaCRL    OBJECT-CLASS ::= {
    SUBCLASS OF     {top }
    KIND            auxiliary
    MAY CONTAIN     {deltaRevocationList }
    ID              id-oc-deltaCRL }
```

国家图书馆专用

```
cpCps    OBJECT-CLASS    ::= {
    SUBCLASS OF        {top }
    KIND               auxiliary
    MAY CONTAIN        {certificatePolicy |
                       certificationPracticeStmt }
    ID                 id-oc-cpCps }

pkiCertPath    OBJECT-CLASS    ::= {
    SUBCLASS OF        {top }
    KIND               auxiliary
    MAY CONTAIN        {pkiPath }
    ID                 id-oc-pkiCertPath }
```

—— PKI 目录属性 ——

```
userCertificate                ATTRIBUTE          ::=        {
    WITH SYNTAX                                   Certificate
    EQUALITY MATCHING RULE                        certificateExactMatch
    ID                                            id-at-userCertificate }

cACertificate                  ATTRIBUTE          ::=        {
    WITH SYNTAX                                   Certificate
    EQUALITY MATCHING RULE                        certificateExactMatch
    ID                                            id-at-cAcertificate }

crossCertificatePair           ATTRIBUTE          ::=        {
    WITH SYNTAX                                   CertificatePair
    EQUALITY MATCHING RULE                        certificatePairExactMatch
    ID                                            id-at-crossCertificatePair }

certificateRevocationList      ATTRIBUTE          ::=        {
    WITH SYNTAX                                   CertificateList
    EQUALITY MATCHING RULE                        certificateListExactMatch
    ID                                            id-at-certificateRevocationList }

authorityRevocationList        ATTRIBUTE          ::=        {
    WITH SYNTAX                                   CertificateList
    EQUALITY MATCHING RULE                        certificateListExactMatch
    ID                                            id-at-authorityRevocationList }

deltaRevocationList            ATTRIBUTE          ::=        {
    WITH SYNTAX                                   CertificateList
    EQUALITY MATCHING RULE                        certificateListExactMatch
```

国家图书馆馆藏

```
    ID                                  id-at-deltaRevocationList }

supportedAlgorithms          ATTRIBUTE      ::= {
    WITH SYNTAX                         SupportedAlgorithm
    EQUALITY MATCHING RULE              algorithmIdentifierMatch
    ID                                  id-at-supportedAlgorithms }

SupportedAlgorithm ::= SEQUENCE {
    algorithmIdentifier                         AlgorithmIdentifier,
    intendedUsage                       [0]     KeyUsage OPTIONAL,
    intendedCertificatePolicies         [1]     CertificatePoliciesSyntax
OPTIONAL }

certificationPracticeStmt ATTRIBUTE    ::=   {
    WITH SYNTAX             InfoSyntax
    ID                      id-at-certificationPracticeStmt }

InfoSyntax      ::=    CHOICE {
    content             DirectoryString {ub-content },
    pointer             SEQUENCE {
        name            GeneralNames,
        hash            HASH {HashedPolicyInfo } OPTIONAL }}

POLICY    ::=   TYPE-IDENTIFIER
HashedPolicyInfo    ::=   POLICY.&Type({Policies})
Policies POLICY ::= {...}——实现者定义——

certificatePolicy   ATTRIBUTE    ::=   {
    WITH SYNTAX       PolicySyntax
    ID                id-at-certificatePolicy }

PolicySyntax    ::=   SEQUENCE {
    policyIdentifier      PolicyID,
    policySyntax          InfoSyntax
}

PolicyID    ::=   CertPolicyId

pkiPath       ATTRIBUTE    ::= {
    WITH SYNTAX       PkiPath
    ID                id-at-pkiPath }

PkiPath       ::=   SEQUENCE OF CrossCertificates
```

国家图书馆专用

```
userPassword    ATTRIBUTE    ::=    {
        WITH SYNTAX                        OCTET STRING (SIZE (0..ub-user-password))
        EQUALITY MATCHING RULE             octetStringMatch
        ID                                 id-at-userPassword }
-- 对象标识符分配 --
-- 对象类 --
id-oc-cRLDistributionPoint                 OBJECT IDENTIFIER ::=      {id-oc 19 }
id-oc-pkiUser                              OBJECT IDENTIFIER ::=      {id-oc 21 }
id-oc-pkiCA                                OBJECT IDENTIFIER ::=      {id-oc 22 }
id-oc-deltaCRL                             OBJECT IDENTIFIER ::=      {id-oc 23 }
id-oc-cpCps                                OBJECT IDENTIFIER ::=      {id-oc 30 }
id-oc-pkiCertPath                          OBJECT IDENTIFIER ::=      {id-oc 31 }

-- 名称窗口 --
id-nf-cRLDistPtNameForm                    OBJECT IDENTIFIER ::=      {id-nf 14 }

-- 目录属性 --
id-at-userPassword                         OBJECT IDENTIFIER ::=      {id-at 35 }
id-at-userCertificate                      OBJECT IDENTIFIER ::=      {id-at 36 }
id-at-cAcertificate                        OBJECT IDENTIFIER ::=      {id-at 37 }
id-at-authorityRevocationList              OBJECT IDENTIFIER ::=      {id-at 38 }
id-at-certificateRevocationList            OBJECT IDENTIFIER ::=      {id-at 39 }
id-at-crossCertificatePair                 OBJECT IDENTIFIER ::=      {id-at 40 }
id-at-supportedAlgorithms                  OBJECT IDENTIFIER ::=      {id-at 52 }
id-at-deltaRevocationList                  OBJECT IDENTIFIER ::=      {id-at 53 }
id-at-certificationPracticeStmt            OBJECT IDENTIFIER ::=      {id-at 68 }
id-at-certificatePolicy                    OBJECT IDENTIFIER ::=      {id-at 69 }
id-at-pkiPath                              OBJECT IDENTIFIER ::=      {id-at 70 }
END
```

国家图书馆专用

A.2 证书扩展名模式

```
CertificateExtensions {joint-iso-itu-t ds(5) module(1) certificateExtensions(26) 4 }
DEFINITIONS IMPLICIT TAGS ::=
BEGIN

-- 输出全部 --

IMPORTS
    id-at, id-ce, id-mr, informationFramework, authenticationFramework,
        selectedAttributeTypes, upperBounds
        FROM UsefulDefinitions {joint-iso-itu-t ds(5) module(1)
        usefulDefinitions(0) 4 }
```

```
        Name, RelativeDistinguishedName, ATTRIBUTE, Attribute, MATCHING-RULE
            FROM InformationFramework informationFramework

        CertificateSerialNumber, CertificateList, AlgorithmIdentifier,
            EXTENSION, Time, PolicyID
            FROM AuthenticationFramework authenticationFramework
    DirectoryString
            FROM SelectedAttributeTypes selectedAttributeTypes
        ub-name
            FROM UpperBounds upperBounds

        ORAddress
            FROM MTSAbstractService {joint-iso-itu-t mhs(6) mts(3)
            modules(0) mts-abstract-service(1) version-1999 (1) };
```

——除非明确注明，否则这个顺序没有意义。
——这个部分中的构造序列的一个组件

——公钥证书和 CRL 扩展名-

```
authorityKeyIdentifier EXTENSION ::= {
    SYNTAX              AuthorityKeyIdentifier
    IDENTIFIED BY       id-ce-authorityKeyIdentifier }

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

subjectKeyIdentifier EXTENSION ::= {
    SYNTAX              SubjectKeyIdentifier
    IDENTIFIED BY       id-ce-subjectKeyIdentifier }

SubjectKeyIdentifier ::= KeyIdentifier

keyUsage EXTENSION ::= {
```

国家图书馆专用

```
    SYNTAX              KeyUsage
    IDENTIFIED BY       id-ce-keyUsage }

KeyUsage ::= BIT STRING {
    digitalSignature        (0),
    nonRepudiation          (1),
    keyEncipherment         (2),
    dataEncipherment        (3),
    keyAgreement            (4),
    keyCertSign             (5),
    cRLSign                 (6),
    encipherOnly            (7),
    decipherOnly            (8) }

extKeyUsage EXTENSION ::= {
    SYNTAX              SEQUENCE SIZE (1..MAX) OF KeyPurposeId
    IDENTIFIED BY       id-ce-extKeyUsage }

KeyPurposeId ::= OBJECT IDENTIFIER

privateKeyUsagePeriod EXTENSION ::= {
    SYNTAX              PrivateKeyUsagePeriod
    IDENTIFIED BY       id-ce-privateKeyUsagePeriod }

PrivateKeyUsagePeriod ::= SEQUENCE {
    notBefore       [0]  GeneralizedTime OPTIONAL,
    notAfter        [1]  GeneralizedTime OPTIONAL }
    ( WITH COMPONENTS {..., notBefore PRESENT }|
    WITH COMPONENTS   {..., notAfter PRESENT })

certificatePolicies EXTENSION ::= {
    SYNTAX              CertificatePoliciesSyntax
    IDENTIFIED BY       id-ce-certificatePolicies }

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

PolicyInformation ::= SEQUENCE {
    policyIdentifier    CertPolicyId,
    policyQualifiers    SEQUENCE SIZE (1..MAX) OF
                        PolicyQualifierInfo OPTIONAL }

CertPolicyId ::= OBJECT IDENTIFIER
```

国家图书馆专用

```
PolicyQualifierInfo ::= SEQUENCE {
    policyQualifierId    CERT-POLICY-QUALIFIER.&id
                         ({SupportedPolicyQualifiers}),
    qualifier            CERT-POLICY-QUALIFIER.&Qualifier
                         ({SupportedPolicyQualifiers}{@policyQualifierId})
                         OPTIONAL }

SupportedPolicyQualifiers CERT-POLICY-QUALIFIER ::= {... }
anyPolicy    OBJECT IDENTIFIER    ::=  {2 5 29 32 0 }
CERT-POLICY-QUALIFIER ::= CLASS {
    &id                  OBJECT IDENTIFIER UNIQUE,
    &Qualifier           OPTIONAL }
WITH SYNTAX {
    POLICY-QUALIFIER-ID      &id
    [QUALIFIER-TYPE      &Qualifier] }

policyMappings EXTENSION ::= {
    SYNTAX               PolicyMappingsSyntax
    IDENTIFIED BY        id-ce-policyMappings }

PolicyMappingsSyntax ::= SEQUENCE SIZE (1..MAX) OF SEQUENCE {
    issuerDomainPolicy       CertPolicyId,
    subjectDomainPolicy      CertPolicyId }

subjectAltName EXTENSION ::= {
    SYNTAX               GeneralNames
    IDENTIFIED BY        id-ce-subjectAltName }

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

GeneralName ::= CHOICE {
    otherName                    [0]  INSTANCE OF OTHER-NAME,
    rfc822Name                   [1]  IA5String,
    dNSName                      [2]  IA5String,
    x400Address                  [3]  ORAddress,
    directoryName                [4]  Name,
    ediPartyName                 [5]  EDIPartyName,
    uniformResourceIdentifier    [6]  IA5String,
    iPAddress                    [7]  OCTET STRING,
    registeredID                 [8]  OBJECT IDENTIFIER }

OTHER-NAME ::= TYPE-IDENTIFIER
```

国家图书馆专用

```
EDIPartyName ::= SEQUENCE {
    nameAssigner        [0]  DirectoryString {ub-name } OPTIONAL,
    partyName           [1]  DirectoryString {ub-name } }

issuerAltName EXTENSION ::= {
    SYNTAX              GeneralNames
    IDENTIFIED BY       id-ce-issuerAltName }

subjectDirectoryAttributes EXTENSION ::= {
    SYNTAX              AttributesSyntax
    IDENTIFIED BY       id-ce-subjectDirectoryAttributes }

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

basicConstraints EXTENSION ::= {
    SYNTAX              BasicConstraintsSyntax
    IDENTIFIED BY       id-ce-basicConstraints }

BasicConstraintsSyntax ::= SEQUENCE {
    cA                      BOOLEAN DEFAULT FALSE,
    pathLenConstraint       INTEGER (0..MAX) OPTIONAL }

nameConstraints EXTENSION ::= {
    SYNTAX                  NameConstraintsSyntax
    IDENTIFIED BY           id-ce-nameConstraints }

NameConstraintsSyntax ::= SEQUENCE {
    permittedSubtrees   [0]  GeneralSubtrees OPTIONAL,
    excludedSubtrees    [1]  GeneralSubtrees OPTIONAL }

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

GeneralSubtree ::= SEQUENCE {
    base                            GeneralName,
    minimum                 [0]  BaseDistance DEFAULT 0,
    maximum                 [1]  BaseDistance OPTIONAL }

BaseDistance ::= INTEGER (0..MAX)

policyConstraints EXTENSION ::= {
    SYNTAX              PolicyConstraintsSyntax
    IDENTIFIED BY       id-ce-policyConstraints }
```

国家图书馆专用

```
PolicyConstraintsSyntax ::= SEQUENCE {
    requireExplicitPolicy      [0] SkipCerts OPTIONAL,
    inhibitPolicyMapping       [1] SkipCerts OPTIONAL }

SkipCerts ::= INTEGER (0..MAX)

cRLNumber EXTENSION ::= {
    SYNTAX              CRLNumber
    IDENTIFIED BY   id-ce-cRLNumber }
CRLNumber ::= INTEGER (0..MAX)

reasonCode EXTENSION ::= {
    SYNTAX              CRLReason
    IDENTIFIED BY       id-ce-reasonCode }

CRLReason ::= ENUMERATED {
    unspecified                 (0),
    keyCompromise               (1),
    cACompromise                (2),
    affiliationChanged          (3),
    superseded                  (4),
    cessationOfOperation        (5),
    certificateHold             (6),
    removeFromCRL               (8),
    privilegeWithdrawn          (9),
    aaCompromise                (10)}

holdInstructionCode EXTENSION ::= {
    SYNTAX              HoldInstruction
    IDENTIFIED BY       id-ce-instructionCode }

HoldInstruction ::= OBJECT IDENTIFIER

invalidityDate EXTENSION ::= {
    SYNTAX              GeneralizedTime
    IDENTIFIED BY       id-ce-invalidityDate }

crlScope EXTENSION      ::= {
    SYNTAX              CRLScopeSyntax
    IDENTIFIED BY       id-ce-cRLScope }
CRLScopeSyntax      ::=   SEQUENCE SIZE (1..MAX) OF PerAuthorityScope
PerAuthorityScope       ::=     SEQUENCE {
    authorityName               [0]   GeneralName OPTIONAL,
```

国家图书馆专用

```
        distributionPoint       [1]  DistributionPointName OPTIONAL,
        onlyContains            [2]  OnlyCertificateTypes OPTIONAL,
        onlySomeReasons         [4]  ReasonFlags OPTIONAL,
        serialNumberRange       [5]  NumberRange OPTIONAL,
        subjectKeyIdRange       [6]  NumberRange OPTIONAL,
        nameSubtrees            [7]  GeneralNames OPTIONAL,
        baseRevocationInfo      [9]  BaseRevocationInfo OPTIONAL
        }

OnlyCertificateTypes    ::= BIT STRING {
        user            (0),
        authority       (1),
        attribute       (2) }
NumberRange ::= SEQUENCE {
        startingNumber  [0]     INTEGER OPTIONAL,
        endingNumber    [1]     INTEGER OPTIONAL,
        modulus                 INTEGER OPTIONAL }

BaseRevocationInfo ::= SEQUENCE {
        cRLtreamIdentifier      [0]  CRLtreamIdentifier OPTIONAL,
        cRLNumber               [1]  CRLNumber,
        baseThisUpdate          [2]  GeneralizedTime }

statusReferrals EXTENSION       ::= {
        SYNTAX          StatusReferrals
        IDENTIFIED BY   id-ce-statusReferrals }

StatusReferrals ::= SEQUENCE SIZE (1..MAX) OF StatusReferral
StatusReferral      ::=     CHOICE {
        cRLReferral     [0]  CRLReferral,
        otherReferral   [1]  INSTANCE OF OTHER-REFERRAL }

CRLReferral ::= SEQUENCE        {
        issuer          [0]     GeneralName OPTIONAL,
        location        [1]     GeneralName OPTIONAL,
        deltaRefInfo    [2]     DeltaRefInfo OPTIONAL,
        cRLScope                CRLcopeSyntax,
        lastUpdate      [3]     GeneralizedTime OPTIONAL,
        lastChangedCRL  [4]     GeneralizedTime OPTIONAL }

DeltaRefInfo    ::=   SEQUENCE {
        deltaLocation   GeneralName,
        lastDelta       GeneralizedTime OPTIONAL }
```

国家图书馆专用

```
OTHER-REFERRAL ::= TYPE-IDENTIFIER

cRLStreamIdentifier EXTENSION    ::= {
    SYNTAX              CRLStreamIdentifier
    IDENTIFIED BY       id-ce-cRLStreamIdentifier }

CRLStreamIdentifier    ::=   INTEGER (0..MAX)

orderedList EXTENSION    ::= {
    SYNTAX              OrderedListSyntax
    IDENTIFIED BY       id-ce-orderedList }

OrderedListSyntax       ::= ENUMERATED {
ascSerialNum       (0),
ascRevDate         (1) }

deltaInfo EXTENSION       ::=   {
    SYNTAX              DeltaInformation
    IDENTIFIED BY       id-ce-deltaInfo }
DeltaInformation    ::=   SEQUENCE {
    deltaLocation       GeneralName,
    nextDelta           GeneralizedTime OPTIONAL }

cRLDistributionPoints EXTENSION ::= {
    SYNTAX              CRLDistPointsSyntax
    IDENTIFIED BY       id-ce-cRLDistributionPoints }

CRLDistPointsSyntax ::= SEQUENCE SIZE (1..MAX) OF DistributionPoint

DistributionPoint ::= SEQUENCE {
    distributionPoint   [0]  DistributionPointName OPTIONAL,
    reasons             [1]  ReasonFlags OPTIONAL,
    cRLIssuer           [2]     GeneralNames OPTIONAL }

DistributionPointName ::= CHOICE {
    fullName                        [0]  GeneralNames,
    nameRelativeToCRLIssuer         [1]  RelativeDistinguishedName }

ReasonFlags ::= BIT STRING {
    unused                  (0),
    keyCompromise           (1),
    cACompromise            (2),
```

国家图书馆专用

```
        affiliationChanged          (3),
        superseded                  (4),
        cessationOfOperation        (5),
        certificateHold             (6),
        privilegeWithdrawn          (7),
        aACompromise                (8) }

issuingDistributionPoint EXTENSION ::= {
        SYNTAX                      IssuingDistPointSyntax
        IDENTIFIED BY               id-ce-issuingDistributionPoint }

IssuingDistPointSyntax ::= SEQUENCE {
        distributionPoint                   [0] DistributionPointName OPTIONAL,
        onlyContainsUserCerts               [1] BOOLEAN DEFAULT FALSE,
        onlyContainsAuthorityCerts          [2] BOOLEAN DEFAULT FALSE,
        onlySomeReasons                     [3] ReasonFlags OPTIONAL,
        indirectCRL                         [4] BOOLEAN DEFAULT FALSE,
        onlyContainsAttributeCerts          [5] BOOLEAN DEFAULT FALSE }

certificateIssuer EXTENSION ::= {
        SYNTAX                      GeneralNames
        IDENTIFIED BY               id-ce-certificateIssuer }

deltaCRLIndicator EXTENSION ::= {
        SYNTAX                      BaseCRLNumber
IDENTIFIED BY                       id-ce-deltaCRLIndicator }

BaseCRLNumber ::= CRLNumber

baseUpdateTime EXTENSION    ::= {
        SYNTAX                      GeneralizedTime
        IDENTIFIED BY               id-ce-baseUpdateTime }

freshestCRL EXTENSION    ::= {
        SYNTAX                      CRLDistPointsSyntax
        IDENTIFIED BY               id-ce-freshestCRL }

InhibitAnyPolicy   EXTENSION    ::= {
        SYNTAX              SkipCerts
IDENTIFIED BY               id-ce-inhibitAnyPolicy }
```

—— PKI 匹配规则——

国家图书馆专用

```
certificateExactMatch MATCHING-RULE ::= {
    SYNTAX              CertificateExactAssertion
    ID                  id-mr-certificateExactMatch }
CertificateExactAssertion ::= SEQUENCE {
    serialNumber    CertificateSerialNumber,
    issuer          Name }

certificateMatch MATCHING-RULE ::= {
    SYNTAX          CertificateAssertion
    ID              id-mr-certificateMatch }
CertificateAssertion ::= SEQUENCE {
    serialNumber            [0] CertificateSerialNumber     OPTIONAL,
    issuer                  [1] Name                        OPTIONAL,
    subjectKeyIdentifier    [2] SubjectKeyIdentifier        OPTIONAL,
    authorityKeyIdentifier  [3] AuthorityKeyIdentifier      OPTIONAL,
    certificateValid        [4] Time                        OPTIONAL,
    privateKeyValid         [5] GeneralizedTime             OPTIONAL,
    subjectPublicKeyAlgID   [6] OBJECT IDENTIFIER           OPTIONAL,
    keyUsage                [7] KeyUsage                    OPTIONAL,
    subjectAltName          [8] AltNameType                 OPTIONAL,
    policy                  [9] CertPolicySet               OPTIONAL,
    pathToName              [10] Name                       OPTIONAL,
    subject                 [11] Name                       OPTIONAL,
    nameConstraints         [12] NameConstraintsSyntax      OPTIONAL }

AltNameType ::= CHOICE {
    builtinNameForm     ENUMERATED {
                rfc822Name                  (1),
                dNSName                     (2),
                x400Address                 (3),
                directoryName               (4),
                ediPartyName                (5),
                uniformResourceIdentifier   (6),
                iPAddress                   (7),
                registeredId                (8) },
    otherNameForm   OBJECT IDENTIFIER }

CertPolicySet ::= SEQUENCE SIZE (1..MAX) OF CertPolicyId

certificatePairExactMatch MATCHING-RULE    ::=    {
    SYNTAX          CertificatePairExactAssertion
    ID              id-mr-certificatePairExactMatch }
```

国家图书馆专用

```
CertificatePairExactAssertion ::= SEQUENCE {
    issuedToThisCAAssertion        [0] CertificateExactAssertion OPTIONAL,
    issuedByThisCAAssertion        [1] CertificateExactAssertion OPTIONAL }
    ( WITH COMPONENTS              {..., issuedToThisCAAssertion PRESENT }|
    WITH COMPONENTS                {..., issuedByThisCAAssertion PRESENT })

certificatePairMatch MATCHING-RULE ::=  {
    SYNTAX    CertificatePairAssertion
    ID        id-mr-certificatePairMatch }

CertificatePairAssertion ::= SEQUENCE {
    issuedToThisCAAssertion   [0] CertificateAssertion OPTIONAL,
    issuedByThisCAAssertion   [1] CertificateAssertion OPTIONAL }
    ( WITH COMPONENTS         {..., issuedToThisCAAssertion PRESENT }|
    WITH COMPONENTS           {..., issuedByThisCAAssertion PRESENT })

certificateListExactMatch MATCHING-RULE  ::=  {
    SYNTAX          CertificateListExactAssertion
    ID              id-mr-certificateListExactMatch }

CertificateListExactAssertion ::= SEQUENCE {
    issuer                  Name,
    thisUpdate              Time,
    distributionPointDistributionPointName OPTIONAL }

certificateListMatch MATCHING-RULE ::= {
    SYNTAX          CertificateListAssertion
    ID              id-mr-certificateListMatch }

CertificateListAssertion ::= SEQUENCE {
    issuer                          Name              OPTIONAL,
    minCRLNumber            [0]     CRLNumber         OPTIONAL,
    maxCRLNumber            [1]     CRLNumber         OPTIONAL,
    reasonFlags                     ReasonFlags       OPTIONAL,
    dateAndTime                     Time              OPTIONAL,
    distributionPoint       [2]     DistributionPointName   OPTIONAL,
    authorityKeyIdentifier  [3]     AuthorityKeyIdentifier  OPTIONAL }

algorithmIdentifierMatch MATCHING-RULE  ::=  {
    SYNTAX          AlgorithmIdentifier
    ID              id-mr-algorithmIdentifierMatch }

policyMatch MATCHING-RULE  ::= {
```

国家图书馆专用

```
    SYNTAX          PolicyID
    ID              id-mr-policyMatch }

pkiPathMatch MATCHING-RULE      ::= {
    SYNTAX          PkiPathMatchSyntax
    ID              id-mr-pkiPathMatch }

PkiPathMatchSyntax    ::= SEQUENCE {
    firstIssuer     Name,
    lastSubject     Name }
```

—— *对象标识符的分配* —

```
id-ce-subjectDirectoryAttributes      OBJECT IDENTIFIER    ::=  {id-ce 9 }
id-ce-subjectKeyIdentifier            OBJECT IDENTIFIER    ::=  {id-ce 14 }
id-ce-keyUsage                        OBJECT IDENTIFIER    ::=  {id-ce 15 }
id-ce-privateKeyUsagePeriod           OBJECT IDENTIFIER    ::=  {id-ce 16 }
id-ce-subjectAltName                  OBJECT IDENTIFIER    ::=  {id-ce 17 }
id-ce-issuerAltName                   OBJECT IDENTIFIER    ::=  {id-ce 18 }
id-ce-basicConstraints                OBJECT IDENTIFIER    ::=  {id-ce 19 }
id-ce-cRLNumber                       OBJECT IDENTIFIER    ::=  {id-ce 20 }
id-ce-reasonCode                      OBJECT IDENTIFIER    ::=  {id-ce 21 }
id-ce-instructionCode                 OBJECT IDENTIFIER    ::=  {id-ce 23 }
id-ce-invalidityDate                  OBJECT IDENTIFIER    ::=  {id-ce 24 }
id-ce-deltaCRLIndicator               OBJECT IDENTIFIER    ::=  {id-ce 27 }
id-ce-issuingDistributionPoint        OBJECT IDENTIFIER    ::=  {id-ce 28 }
id-ce-certificateIssuer               OBJECT IDENTIFIER    ::=  {id-ce 29 }
id-ce-nameConstraints                 OBJECT IDENTIFIER    ::=  {id-ce 30 }
id-ce-cRLDistributionPoints           OBJECT IDENTIFIER    ::=  {id-ce 31 }
id-ce-certificatePolicies             OBJECT IDENTIFIER    ::=  {id-ce 32 }
id-ce-policyMappings                  OBJECT IDENTIFIER    ::=  {id-ce 33 }
-- deprecated                         OBJECT IDENTIFIER    ::=  {id-ce 34 }
id-ce-authorityKeyIdentifier          OBJECT IDENTIFIER    ::=  {id-ce 35 }
id-ce-policyConstraints               OBJECT IDENTIFIER    ::=  {id-ce 36 }
id-ce-extKeyUsage                     OBJECT IDENTIFIER    ::=  {id-ce 37 }
id-ce-cRLStreamIdentifier             OBJECT IDENTIFIER    ::=  {id-ce 40 }
id-ce-cRLScope                        OBJECT IDENTIFIER    ::=  {id-ce 44 }
id-ce-statusReferrals                 OBJECT IDENTIFIER    ::=  {id-ce 45 }
id-ce-freshestCRL                     OBJECT IDENTIFIER    ::=  {id-ce 46 }
id-ce-orderedList                     OBJECT IDENTIFIER    ::=  {id-ce 47 }
id-ce-baseUpdateTime                  OBJECT IDENTIFIER    ::=  {id-ce 51 }
id-ce-deltaInfo                       OBJECT IDENTIFIER    ::=  {id-ce 53 }
id-ce-inhibitAnyPolicy                OBJECT IDENTIFIER    ::=  {id-ce 54 }
```

电子图书馆专用

```
-- OIDs 的匹配规则 --
id-mr-certificateExactMatch          OBJECT IDENTIFIER   ::=  {id-mr 34}
id-mr-certificateMatch               OBJECT IDENTIFIER   ::=  {id-mr 35}
id-mr-certificatePairExactMatch      OBJECT IDENTIFIER   ::=  {id-mr 36}
id-mr-certificatePairMatch           OBJECT IDENTIFIER   ::=  {id-mr 37}
id-mr-certificateListExactMatch      OBJECT IDENTIFIER   ::=  {id-mr 38}
id-mr-certificateListMatch           OBJECT IDENTIFIER   ::=  {id-mr 39}
id-mr-algorithmIdentifierMatch       OBJECT IDENTIFIER   ::=  {id-mr 40}
id-mr-policyMatch                    OBJECT IDENTIFIER   ::=  {id-mr 60}
id-mr-pkiPathMatch                   OBJECT IDENTIFIER   ::=  {id-mr 62}

-- 本部分未使用以下对象标识符:
-- {id-ce 2}, {id-ce 3}, {id-ce 4}, {id-ce 5}, {id-ce 6}, {id-ce 7},
-- {id-ce 8}, {id-ce 10}, {id-ce 11}, {id-ce 12}, {id-ce 13},
-- {id-ce 22}, {id-ce 25}, {id-ce 26}
END
```

A.3 属性证书框架模式

```
AttributeCertificateDefinitions {joint-iso-itu-t-ds(5)module(1)
attributeCertificateDefinitions(32) 4}
DEFINITIONS IMPLICIT TAGS ::=
BEGIN

-- 全部输出 --

输入
    id-at, id-ce, id-mr, informationFramework, authenticationFramework,
        selectedAttributeTypes, upperBounds, id-oc, certificateExtensions
        FROM UsefulDefinitions {joint-iso-ITU-t ds(5) module(1)
        usefulDefinitions(0) 4}

    Name, RelativeDistinguishedName, ATTRIBUTE, Attribute,
        MATCHING-RULE, AttributeType, OBJECT-CLASS, top
        FROM InformationFramework informationFramework

    CertificateSerialNumber, CertificateList, AlgorithmIdentifier,
        EXTENSION, SIGNED, InfoSyntax, PolicySyntax, Extensions, Certificate
        FROM AuthenticationFramework authenticationFramework

    DirectoryString, TimeSpecification, UniqueIdentifier
        FROM SelectedAttributeTypes selectedAttributeTypes
```

国家图书馆专用

```
    GeneralName, GeneralNames, NameConstraintsSyntax, certificateListExactMatch
        FROM CertificateExtensions certificateExtensions

    ub-name
        FROM UpperBounds upperBounds

    UserNotice
        FROM PKIX1Implicit93 {iso(1) identified-organization(3) dod(6) internet(1) security(5)
mechanisms(5) pkix(7) id-mod(0) id-pkix1-implicit-93(4)}

    ORAddress
        FROM MTSAbstractService {joint-iso-itu-t mhs(6) mts(3)
        modules(0) mts-abstract-service(1) version-1999 (1) };

——除非明确注明,否则该序列没有意义。
——说明书中的结构序列组件。
——属性证书结构 ——

AttributeCertificate ::= SIGNED {AttributeCertificateInfo}
AttributeCertificateInfo ::= SEQUENCE
    {
    version                 AttCertVersion   版本是 V2,
    holder                  Holder,
    issuer                  AttCertIssuer,
    signature               AlgorithmIdentifier,
    serialNumber            CertificateSerialNumber,
    attrCertValidityPeriod  AttCertValidityPeriod,
    attributes              SEQUENCE OF Attribute,
    issuerUniqueID          UniqueIdentifier OPTIONAL,
    extensions              Extensions OPTIONAL
    }

AttCertVersion  ::= INTEGER {v1(0), v2(1) }

Holder  ::=  SEQUENCE
        {
        baseCertificateID     [0] IssuerSerial     OPTIONAL,
        ——发布者和公钥证书持有者的序列号
        entityName      [1] GeneralNames          OPTIONAL,
        ——实体名或角色名
        objectDigestInfo      [2] ObjectDigestInfo   OPTIONAL
        ——如果存在,则必须是第二版
——至少存在一个证书 ID 库,实体名,或者对象摘要信息—— }
```

国家图书馆专用

```
ObjectDigestInfo    ::= SEQUENCE {
        digestedObjectType      ENUMERATED {
                publicKey               (0),
                publicKeyCert           (1),
                otherObjectTypes        (2) },
        otherObjectTypeID               OBJECT IDENTIFIER       OPTIONAL,
        digestAlgorithm                 AlgorithmIdentifier,
        objectDigest                    BIT STRING }

AttCertIssuer ::= [0]     SEQUENCE {
        issuerName                      GeneralNames      OPTIONAL,
        baseCertificateID   [0]         IssuerSerial      OPTIONAL,
        objectDigestInfo    [1]         ObjectDigestInfo      OPTIONAL }
-- 必须存在一个组件
    ( WITH COMPONENTS       {..., issuerName      PRESENT }|
      WITH COMPONENTS       {..., baseCertificateID       PRESENT }|
      WITH COMPONENTS       {..., objectDigestInfo PRESENT })

IssuerSerial        ::=     SEQUENCE {
        issuer          GeneralNames,
        serial          CertificateSerialNumber,
        issuerUID       UniqueIdentifier OPTIONAL }

AttCertValidityPeriod       ::= SEQUENCE {
        notBeforeTime   GeneralizedTime,
        notAfterTime    GeneralizedTime }

AttributeCertificationPath      ::=     SEQUENCE {
        attributeCertificate            AttributeCertificate,
        acPath                          SEQUENCE OF ACPathData OPTIONAL }

ACPathData      ::=     SEQUENCE {
        certificate             [0]  Certificate   OPTIONAL,
        attributeCertificate    [1]  AttributeCertificate   OPTIONAL }

PrivilegePolicy    ::=   OBJECT IDENTIFIER

-- 特权属性 --

roleATTRIBUTE ::= {
        WITH SYNTAX         RoleSyntax
        ID                  id-at-role }
```

国家图书馆专用

```
RoleSyntax ::= SEQUENCE {
    roleAuthority [0] GeneralNames OPTIONAL,
    roleName      [1] GeneralName }

-- PMI 对象类 --

pmiUser OBJECT-CLASS ::= {
    SUBCLASS OF     {top}
    KIND            auxiliary
    MAY CONTAIN     {attributeCertificateAttribute}
    ID              id-oc-pmiUser
    }

pmiAA OBJECT-CLASS ::= {
-- a PMI AA
    SUBCLASS OF     {top}
    KIND            auxiliary
    MAY CONTAIN     {aACertificate |
                     attributeCertificateRevocationList |
                     attributeAuthorityRevocationList}
    ID              id-oc-pmiAA
    }

pmiSOA OBJECT-CLASS ::= {      -- PMI 认证源
    SUBCLASS OF     {top}
    KIND            auxiliary
    MAY CONTAIN     {aACertificateRevocation |
                     attributeCertificateRevocationList|
                     attributeAuthorityRevocationList}
    ID              id-oc-pmiSOA
    }

attCertCRLDistributionPt  OBJECT-CLASS ::= {
    SUBCLASS OF     {top}
    KIND            auxiliary
    MAY CONTAIN     {attributeCertificateRevocationList |
                     attributeAuthorityRevocationList}
    ID              id-oc-attCertCRLDistributionPts
    }

pmiDelegationPath  OBJECT-CLASS  ::= {
    SUBCLASS OF     {top}
    KIND            auxiliary
```

国家图书馆专用

```
        MAY CONTAIN             {delegationPath }
        ID                      id-oc-pmiDelegationPath }

    privilegePolicy         OBJECT-CLASS    ::= {
        SUBCLASS OF             {top }
        KIND                    auxiliary
        MAY CONTAIN             {privPolicy }
        ID                      id-oc-privilegePolicy }
```

—— *PMI 目录属性* ——

```
    attributeCertificateAttribute           ATTRIBUTE       ::= {
        WITH SYNTAX                         AttributeCertificate
        EQUALITY MATCHING RULE              attributeCertificateExactMatch
        ID                                  id-at-attributeCertificate }

    aACertificate                           ATTRIBUTE       ::=  {
        WITH SYNTAX                         AttributeCertificate
        EQUALITY MATCHING RULE              attributeCertificateExactMatch
        ID                                  id-at-aACertificate }

    attributeDescriptorCertificate          ATTRIBUTE       ::= {
        WITH SYNTAX                         AttributeCertificate
        EQUALITY MATCHING RULE              attributeCertificateExactMatch
        ID                                  id-at-attributeDescriptorCertificate }

    attributeCertificateRevocationList      ATTRIBUTE ::= {
        WITH SYNTAX                         CertificateList
        EQUALITY MATCHING RULE              certificateListExactMatch
        ID                                  id-at-attributeCertificateRevocationList }

    attributeAuthorityRevocationList        ATTRIBUTE       ::= {
        WITH SYNTAX                         CertificateList
        EQUALITY MATCHING RULE              certificateListExactMatch
        ID                                  id-at-attributeAuthorityRevocationList }

    delegationPath                          ATTRIBUTE       ::= {
        WITH SYNTAX                         AttCertPath
        ID                                  id-at-delegationPath }

    AttCertPath     ::=     SEQUENCE OF AttributeCertificate

    privPolicy                              ATTRIBUTE       ::= {
        WITH SYNTAX                         PolicySyntax
```

国家图书馆馆藏

```
        ID                                  id-at-privPolicy }
```

—— 属性证书扩充和匹配规则 ——

```
    attributeCertificateExactMatch MATCHING-RULE      ::= {
        SYNTAX                              AttributeCertificateExactAssertion
        ID                                  id-mr-attributeCertificateExactMatch }

    AttributeCertificateExactAssertion    ::=   SEQUENCE {
        serialNumber                        CertificateSerialNumber,OPTIONAL
        issuer                              IssuerSerial
        }

    attributeCertificateMatch     MATCHING-RULE      ::=      {
        SYNTAX                              AttributeCertificateAssertion
        ID                                  id-mr-attributeCertificateMatch }

    AttributeCertificateAssertion      ::=      SEQUENCE      {
        holder              [0]    CHOICE {
          baseCertificateID     [0]  IssuerSerial,
          holderName              [1]  GeneralNames}OPTIONAL,
        issuer              [1]  GeneralNames OPTIONAL,
        attCertValidity     [2]  GeneralizedTime OPTIONAL,
        attType             [3]  SET OF AttributeType OPTIONAL }
```

—— 至少存在一个序列组件

```
    holderIssuerMatch MATCHING-RULE      ::= {
        SYNTAX          HolderIssuerAssertion
        ID              id-mr-holderIssuerMatch }
    HolderIssuerAssertion    ::=   SEQUENCE {
        holder          [0]  Holder   OPTIONAL,
        issuer          [1]     AttCertIssuer   OPTIONAL
        }

    delegationPathMatch MATCHING-RULE      ::= {
        SYNTAX          DelMatchSyntax
        ID              id-mr-delegationPathMatch }

    DelMatchSyntax    ::=      SEQUENCE {
        firstIssuer      AttCertIssuer,
        lastHolder       Holder }

    sOAIdentifier EXTENSION      ::= {
```

国家图书馆专用

```
        SYNTAX                  NULL
        IDENTIFIED BY           id-ce-sOAIdentifier }

    authorityAttributeIdentifier EXTENSION      ::=
        {
        SYNTAX              AuthorityAttributeIdentifierSyntax
        IDENTIFIED BY       {id-ce-authorityAttributeIdentifier }}

    AuthorityAttributeIdentifierSyntax    ::=   SEQUENCE SIZE (1..MAX) OF AuthAttiid

    AuthAttId    ::=    IssuerSerial

    authAttIdMatch MATCHING-RULE      ::= {
        SYNTAX          AuthorityAttributeIdentifierSyntax
        ID              id-mr-authAttIdMatch }

    roleSpecCertIdentifier EXTENSION ::=
        {
        SYNTAX              RoleSpecCertIdentifierSyntax
        IDENTIFIED BY       {id-ce-roleSpecCertIdentifier}}

    RoleSpecCertIdentifierSyntax      ::= SEQUENCE SIZE (1..MAX) OF RoleSpecCertIdentifier
    RoleSpecCertIdentifier      ::= SEQUENCE {
        roleName                    [0]     GeneralName,
        roleCertIssuer              [1]     GeneralName,
        roleCertSerialNumber        [2]     CertificateSerialNumber   OPTIONAL,
        roleCertLocator             [3]     GeneralNames    OPTIONAL
        }

    roleSpecCertIdMatch MATCHING-RULE      ::= {
        SYNTAX      RoleSpecCertIdentifierSyntax
        ID          id-mr-roleSpecCertIdMatch }

    basicAttConstraints EXTENSION ::=
        {
          SYNTAX              BasicAttConstraintsSyntax
          IDENTIFIED BY       {id-ce-basicAttConstraints }
        }
    BasicAttConstraintsSyntax ::= SEQUENCE
        {
          authority             BOOLEAN DEFAULT FALSE,
          pathLenConstraint     INTEGER (0..MAX) OPTIONAL
        }
```

国家图书馆专用

```
basicAttConstraintsMatch MATCHING-RULE ::= {
    SYNTAX          BasicAttConstraintsSyntax
    ID              id-mr-basicAttConstraintsMatch }

delegatedNameConstraints EXTENSION ::= {
    SYNTAX          NameConstraintsSyntax
    IDENTIFIED BY   id-ce-delegatedNameConstraints }

delegatedNameConstraintsMatch MATCHING-RULE ::= {
    SYNTAX          NameConstraintsSyntax
    ID              id-mr-delegatedNameConstraintsMatch }

timeSpecification EXTENSION ::= {
    SYNTAX          TimeSpecification
    IDENTIFIED BY   id-ce-timeSpecification }

timeSpecificationMatch MATCHING-RULE ::= {
    SYNTAX          TimeSpecification
    ID              id-mr-timeSpecMatch }

acceptableCertPolicies EXTENSION ::= {
    SYNTAX          AcceptableCertPoliciesSyntax
    IDENTIFIED BY   id-ce-acceptableCertPolicies }

AcceptableCertPoliciesSyntax ::= SEQUENCE SIZE (1..MAX) OF CertPolicyId

CertPolicyId ::= OBJECT IDENTIFIER

acceptableCertPoliciesMatch MATCHING-RULE ::= {
    SYNTAX          AcceptableCertPoliciesSyntax
    ID              id-mr-acceptableCertPoliciesMatch }

attributeDescriptor EXTENSION ::= {
    SYNTAX          AttributeDescriptorSyntax
    IDENTIFIED BY   {id-ce-attributeDescriptor }}

AttributeDescriptorSyntax ::= SEQUENCE {
  identifier                AttributeIdentifier,
  attributeSyntax           OCTET STRING (SIZE(1..MAX)),
  name              [0]     AttributeName OPTIONAL,
  description       [1]     AttributeDescription OPTIONAL,
dominationRule              PrivilegePolicyIdentifier }
```

国家图书馆专用

```
AttributeIdentifier ::= ATTRIBUTE.&id({AttributeIDs})

AttributeIDs ATTRIBUTE ::= {...}

AttributeName ::= UTF8String (SIZE (1..MAX))

AttributeDescription ::= UTF8String (SIZE(1..MAX))

PrivilegePolicyIdentifier ::= SEQUENCE {
    privilegePolicy          PrivilegePolicy,
    privPolSyntax            InfoSyntax }

attDescriptor MATCHING-RULE ::= {
    SYNTAX          AttributeDescriptorSyntax
    ID              id-mr-attDescriptorMatch }

userNotice EXTENSION ::= {
    SYNTAX          SEQUENCE SIZE (1..MAX) OF UserNotice
    IDENTIFIED BY   id-ce-userNotice }

targetingInformation EXTENSION ::= {
    SYNTAX          SEQUENCE SIZE (1..MAX) OF Targets
    IDENTIFIED BY   id-ce-targetInformation }

Targets ::= SEQUENCE SIZE (1..MAX) OF Target

Target ::= CHOICE {
    targetName      [0]     GeneralName,
    targetGroup     [1]     GeneralName,
    targetCert      [2]     TargetCert }

TargetCert ::= SEQUENCE {
    targetCertificate   IssuerSerial,
    targetName          GeneralName OPTIONAL,
    certDigestInfo      ObjectDigestInfo OPTIONAL }

noRevAvail EXTENSION ::= {
    SYNTAX          NULL
    IDENTIFIED BY   id-ce-noRevAvail }

acceptablePrivilegePolicies EXTENSION ::= {
    SYNTAX                  AcceptablePrivilegePoliciesSyntax
```

国家图书馆专用

```
    IDENTIFIED BY          id-ce-acceptablePrivilegePolicies }

AcceptablePrivilegePoliciesSyntax  ::=  SEQUENCE SIZE (1..MAX) OF PrivilegePolicy
-- 分配对象标识符 --
-- 对象类 --
id-oc-pmiUser                          OBJECT IDENTIFIER ::=  {id-oc 24 }
id-oc-pmiAA                            OBJECT IDENTIFIER ::=  {id-oc 25 }
id-oc-pmiSOA                           OBJECT IDENTIFIER ::=  {id-oc 26 }
id-oc-attCertCRLDistributionPts        OBJECT IDENTIFIER ::=  {id-oc 27 }
id-oc-privilegePolicy                  OBJECT IDENTIFIER ::=  {id-oc 32 }
id-oc-pmiDelegationPath                OBJECT IDENTIFIER ::=  {id-oc 33 }
-- 目录属性 --
id-at-attributeCertificate             OBJECT IDENTIFIER ::=  {id-at 58 }
id-at-attributeCertificateRevocationList OBJECT IDENTIFIER ::=  {id-at 59 }
id-at-aACertificate                    OBJECT IDENTIFIER ::=  {id-at 61 }
id-at-attributeDescriptorCertificate   OBJECT IDENTIFIER ::=  {id-at 62 }
id-at-attributeAuthorityRevocationList OBJECT IDENTIFIER ::=  {id-at 63 }
id-at-privPolicy                       OBJECT IDENTIFIER ::=  {id-at 71 }
id-at-role                             OBJECT IDENTIFIER ::=  {id-at 72 }
id-at-delegationPath                   OBJECT IDENTIFIER ::=  {id-at 73 }
-- 属性证书扩展 --
id-ce-authorityAttributeIdentifier     OBJECT IDENTIFIER ::=  {id-ce 38 }
id-ce-roleSpecCertIdentifier           OBJECT IDENTIFIER ::=  {id-ce 39 }
id-ce-basicAttConstraints              OBJECT IDENTIFIER ::=  {id-ce 41 }
id-ce-delegatedNameConstraints         OBJECT IDENTIFIER ::=  {id-ce 42 }
id-ce-timeSpecification                OBJECT IDENTIFIER ::=  {id-ce 43 }
id-ce-attributeDescriptor              OBJECT IDENTIFIER ::=  {id-ce 48 }
id-ce-userNotice                       OBJECT IDENTIFIER ::=  {id-ce 49 }
id-ce-sOAIdentifier                    OBJECT IDENTIFIER ::=  {id-ce 50 }
id-ce-acceptableCertPolicies           OBJECT IDENTIFIER ::=  {id-ce 52 }
id-ce-targetInformation                OBJECT IDENTIFIER ::=  {id-ce 55 }
id-ce-noRevAvail                       OBJECT IDENTIFIER ::=  {id-ce 56 }
id-ce-acceptablePrivilegePolicies      OBJECT IDENTIFIER ::=  {id-ce 57 }
-- PMI 匹配规则 --
id-mr-attributeCertificateMatch        OBJECT IDENTIFIER ::=  {id-mr 42 }
id-mr-attributeCertificateExactMatch   OBJECT IDENTIFIER ::=  {id-mr 45 }
id-mr-holderIssuerMatch                OBJECT IDENTIFIER ::=  {id-mr 46 }
id-mr-authAttIdMatch                   OBJECT IDENTIFIER ::=  {id-mr 53 }
id-mr-roleSpecCertIdMatch              OBJECT IDENTIFIER ::=  {id-mr 54 }
id-mr-basicAttConstraintsMatch         OBJECT IDENTIFIER ::=  {id-mr 55 }
id-mr-delegatedNameConstraintsMatch    OBJECT IDENTIFIER ::=  {id-mr 56 }
id-mr-timeSpecMatch                    OBJECT IDENTIFIER ::=  {id-mr 57 }
id-mr-attDescriptorMatch               OBJECT IDENTIFIER ::=  {id-mr 58 }
```

国家图书馆专用

```
id-mr-acceptableCertPoliciesMatch    OBJECT IDENTIFIER ::= {id-mr 59}
id-mr-delegationPathMatch            OBJECT IDENTIFIER ::= {id-mr 61}
END
```

国家图书馆专用

附 录 B
（规范性附录）
CRL 的产生和处理规则

B.1 介绍

可信实体（证书用户）要有检查证书状态的能力，以便决定是否信任那张证书。证书撤销列表（CRL）是可信实体用来获取信息的一种机制。其他机制也可以用来获取这一信息，但不在本说明范围之内。

本附录为可信实体检查证书撤销状态描述了 CRL 的用法。不同权威对其撤销列表的发布有不同的策略。例如，在某些情况下，证书发布权威可以授权给一个不同权威来发布它的撤销证书。某些权威可以将终端实体和 CA 证书的撤销合并成一张表，而其他权威将这些列表分解成单独的列表。某些权威可以在 CRL 分段上对证书总体进行划分，某些权威可以在常规 CRL 间隔之间给撤销列表发布更新。因此，可信实体要能确定他所获得的 CRL 的范围，来确保他们有一份完整的撤销信息，该信息含盖了在给定的运行策略下，因为各种原因被撤销的证书。crlScope 扩展可作为一种决定范围的机制。本附录提供了一种在 CRL 中不存在 crlScope 扩展情况下的机制。

本附录是为检查使用 CRL、EPRLs 和 CARLs 的公钥证书的撤销状态而编写的。然而，这种描述也能用于检查使用属性证书撤销列表（ACRL）和属性权威撤销列表（AARL）的属性证书的撤销状态。本附录的目的是考虑用 ACRL 代替 CRL，并用 AARL 代替 CARL。

B.1.1 CRL 类型

基于证书发布权威的证书撤销策略，可信实体可获得以下一个或多个 CRL 类型。

- 完整和完善的 CRL；
- 完整和完善的终端实体 CRL（EPRL）；
- 完整和完善的认证权威撤销列表（CARL）；
- 分布点 CRL，EPRL 或 CARL；
- 间接 CRL，EPRL 或 CARL（ICRL）；
- 增量 CRL，EPRL 或 CARL；
- 间接 Δ-CRL，EPRL 或 CARL。

完整和完善的 CRL 是一张由所有撤销的终端实体和 CA 证书组成的列表，它是由权威发布的。

完整和完善的 EPRL 是一张由权威发布的所有撤销终端实体证书列表。

完整和完善的 CARL 是一张由权威发布的撤销 CA 证书列表。

分布点 CRL，EPRL 或 CARL 含盖了由权威发布的所有或部分证书。证书子集可以基于多种不同的标准。

间接 CRL，EPRL 或 CARL（iCRL）是一个包含部分或全部不是由签发该 CRL 的权威发布的撤销证书列表。

增量 CRL，EPRL 或 CARL 是一个仅包含 CRL 变化的 CRL，即在 Δ-CRL 中调用 CRL 的时候所给定的范围是完整的。请注意，调用的 CRL 对于所给范围可能是完整的，或者它可能是一个 Δ-CRL，用于构造一个在所给范围内完整的 CRL。上述所有的 CRL 类型（除了 Δ-CRL）是一个在所给范围内完整的 CRL 类型。Δ-CRL 必须与相关的 CRL 一起使用，即在相同范围内是完整的，以便构成一张完整的证书撤销状态图。

间接 Δ-CRL，EPRL 或 CARL 是一个仅仅包含一个或多个 CRL 变化的 CRL，它对于所给范围是完整的，其中所有或部分证书不是由签发该 CRL 的权威发布的。

国家图书馆藏章

本附录中,“CRL 的范围”由独立的二维空间定义。其中一维是被 CRL 覆盖的证书集合。另一维是 CRL 包含的撤销原因代码的集合。CRL 的范围可由以下一种或多种方式定义:

- CRL 中的发布的分布点(IDP)扩展;
- CRL 中的 CRL 范围扩展;
- 本说明范围外的其他方式。

B.1.2 CRL 处理

如果可信实体把 CRL 作为决定一张证书是否被撤销的机制,则他们必须确信对该证书使用适当的 CRL。本附录通过一系列特定步骤来描述获得和处理适当 CRL 的程序。执行过程与该程序产生的外部操作是相同的。被特殊执行过程用来从给定的输入(例如,证书本身和本地策略输入)导出正确输出(例如,证书撤销状态)的算法未被标准化。例如,尽管这个过程在处理中被描述为一连串有序的步骤,但执行过程中可使用在本地缓存的 CRL,而不是每次都要去获取 CRL,它为证书范围提供了完整的 CRL,并且没有破坏证书或策略的任何参数。

对于包含 statusReferrals 扩展的 CRL 结构中的指针,本附录不包括该过程。包含该扩展的所有 CRL 将不可作为可信实体源用于检查所有证书的撤销状态。包含该扩展的 CRL 能被可信实体作为一个辅助工具来为检查撤销状态查找适当的 CRL。

以下步骤在后面的 B.2 到 B.5 中描述:

1) 确定 CRL 参数
2) 确定请求的 CRL
3) 获得 CRL
4) 处理 CRL

步骤 1 在证书和其他决定需要哪种 CRL 类型的地方标识参数。

步骤 2 用参数值作决定。

步骤 3 在可获得 CRL 类型的地方标识目录属性。

步骤 4 描述适当 CRL 的处理。

国家图书馆专用

B.2 确定 CRL 参数

证书中的信息与从可信实体操作的策略中获得的信息一样,为决定适当的候选 CRL 提供参数。需要以下信息来决定哪种 CRL 类型是合适的:

- 证书类型(例如,终端实体或者 CA);
- 关键 CRL 分布点;
- 关键的最新 CRL;
- 重要的原因代码。

证书类型可由证书中的基本约束扩展来决定。如果扩展存在,它将指出这个证书是 CA 证书还是终端实体证书。如果扩展缺省,则证书类型可被视为终端实体。如果 CRL,EPRL 或者 CARL 能被用于检验证书的撤销,则它需要该信息。

如果证书包含一个关键 CRL 分布点扩展,可信实体的证书处理系统必须知道该扩展以便能信任这个证书。例如,单纯依靠一个完整的 CRL 是不够的。

如果证书包含一个关键的最新 CRL 扩展,那么在没有获得并检验最新的 CRL 之前,可信实体不能使用该证书。

重要的原因代码由权威的证书撤销策略决定并且通常由应用提供。推荐它包含所有的原因代码。当根据原因代码确定哪个 CRL 足够时,需要包含此信息。

注意:即使当 freshestCRL 扩展被标识成非关键或不在证书中时,策略也可以规定可信实体是否要为撤销状态检验 Δ-CRL。尽管该步骤中不包括这些,可选 Δ-CRL 的处理在步骤 4 中描述。

B.3 确定请求的 CRL

B.2 中描述的参数值决定了一个准则，据此可以确定检验给定证书撤销状态所需的 CRL 类型。CRL 类型的确定可由以下基于 B3.1～B3.4 中描述的标准来决定：

- 含有关键 CRL DP 声明的终端实体证书；
- 不含关键 CRL DP 声明的终端实体证书；
- 含有关键 CRL DP 声明的 CA 证书；
- 不含关键 CRL DP 声明的 CA 证书。

保留参数(关键的最新 CRL 扩展和重要原因代码作用的集合)的处理已在每个部分中完成。

注意：每种情况下，可以有一个以上的 CRL 类型能够满足其需求，此时可信实体可以选择任何适当的类型来使用。

B.3.1 含有关键 CRL DP 的终端实体

如果证书是终端实体证书，且 cRLDistributionPoints 扩展存在于证书当中并被标识为关键的，那么需要得到以下 CRL：

- 一个源于推荐分布点 CRL 的 CRL，它含盖了一个或多个重要的原因代码。
- 如果所有重要的原因代码都没有被包括在 CRL 中，以下 CRL 的任意组合就可满足剩余原因代码的撤销状态：
  a) 辅助分布点 CRL
  b) 辅助的完整 CRL
  c) 辅助的完整 EPRLs

如果最新的 CRL 扩展也出现在证书中并被标识为关键的，则同样需要从一个或多个扩展中的推荐分布点获得一个或多个 CRL，来确保对所有重要原因代码的最新撤销信息进行了检验。

B.3.2 不含关键 CRL DP 的终端实体

如果证书是一个终端实体证书并且 cRLDistributionPoints 扩展不存在于证书中或出现了但没有被标识为关键的，则以下 CRL 的任意组合均可满足重要原因代码的撤销状态信息：

- 分布点 CRL(如果存在)；
- 完整的 CRL；
- 完整的 EPRLs。

如果最新的 CRL 扩展也出现在证书中并被标识为关键的，则同样需要从一个或多个扩展中的推荐分布点获得一个或多个 CRL，来确保对所有重要原因代码的最新撤销信息进行了检验。

B.3.3 含有关键 CRL DP 的 CA

如果证书是一个 CA 并且 cRLDistributionPoints 扩展存在于证书中并被标识为关键的，则可以获得以下 CRL/CARLs：

- 源于推荐分布点的 CRL 或 CARL，它含盖了一个或多个重要的原因代码；
- 如果所有重要的原因代码没有包括在 CRL/CARL 中，则以下 CRL/CARLs 的任意组合均可满足剩余原因代码的撤销状态：
  a) 辅助分布点 CRL/CARLs；
  b) 辅助的完整 CRL；
  c) 辅助的完整 CARLs。

如果最新的 CRL 扩展不存在于证书中且未被标识为关键的，则可从扩展中的一个或多个推荐分布点获得 CRL/CARLs，来确保所有重要原因代码的撤销信息已被检验。

B.3.4 不含关键 CRL DP 的 CA

如果证书是 CA 证书并且 cRLDistributionPoints 扩展不存在于证书中或出现但未被标识为关键

国家图书馆专用

的,那么以下 CRL 的任意组合均可满足重要原因代码的撤销状态:

- 分布点 CRL/CARLs(如果存在);
- 完整的 CRL;
- 完整的 CARLs。

如果最新 CRL 扩展存在于证书中并被标识为关键的,则可从一个或多个扩展中的推荐分布点获得一个或多个 CRL/CARLs,来确保重要原因代码的最新撤销信息已被检验。

B.4 获得 CRL

如果可信实体从目录获得了适当的 CRL,则这些 CRL 是从 CRL DP 或证书发布者的目录入口获得的,该入口是通过获得适当属性得来的。例如,一个或多个以下属性:

- 证书撤销列表;
- 授权撤销列表;
- Delta 撤销列表。

B.5 处理 CRL

在考虑了 B.2 中所讨论的参数,鉴别出 B.3 中所描述的适当 CRL 类型,获得了 B.4 中所描述的适当 CRL 集合之后,可信实体准备处理 CRL 了。该 CRL 的集合至少包含一个基本的 CRL 和一个或多个 Δ-CRL。对每个正在处理的 CRL,可信实体必须确保 CRL 的范围是准确的。可信实体通过前面B.2 和 B.3 已经确定了该 CRL 是符合重要证书范围的。另外,有效性检查必须在 CRL 上执行并且他们必须通过检验来决定证书是否已被撤销。这些检查已在 B.5.1~B.5.4 中描述。

B.5.1 验证基本 CRL 的范围

如 B.3 中所述,存在多种类型的 CRL 可以作为基本的 CRL 用来检验证书的撤销状态。根据发布权威机构发布 CRL 的策略,可信实体可能会有一个或多个以下类型的基本 CRL。

- 对于所有实体都是完整的 CRL;
- 完整的 EPRL;
- 完整的 CARL;
- 基于分布点的 CRL/EPRL/CARL。

B.5.1.1~B.5.1.4 提供了这套环境,它对可信实体必须是非常真实的,可信实体将每种类型的 CRL 用于检验重要原因代码的证书撤销状态。

间接的基本 CRL 在各个子章节中描述。

B.5.1.1 完成 CRL

为了确定一个 CRL 对终端实体和 CA 证书是一个完整的 CRL,对于所有的重要原因代码,以下必须是真实的:

- 不存在增量 CRL 指示器扩展;
- 存在发布的分布点扩展;
- 发布的分布点扩展必须不包含分布点域;
- 发布的分布点扩展必须不包含设置成 TRUE 的 onlyContainsUserCerts 域;
- 发布的分布点扩展必须不包含设置成 TRUE 的 onlyContainsAuthorityCerts 域;
- 发布的分布点扩展必须不包含设置成 TRUE 的 onlyContainsAttributeCerts 域;
- 如果 reasonCodes 域存在于发布的分布点扩展中,原因代码域必须包括使用的所有应用的原因代码;
- 发布的分布点扩展可以包含或不包含 indirectCRL 域(因此,该域不必被检查)。

B.5.1.2 完整的 EPRL

为确定 CRL 对重要原因代码是否是一个完整的 EPRL，以下所有必须是真实的：

- 不存在增量 CRL 指示器扩展；
- 必须存在发布的分布点扩展；
- 发布的分布点扩展必须不包含分布点域；
- 发布的分布点扩展必须包含 onlyContainsUserCerts 域。该域必须被设置成 TRUE；
- 发布的分布点扩展必须不包含被设置成 TRUE 的 onlyContainAuthorityCerts 域；
- 发布的分布点扩展必须不包含被设置成 TRUE 的 onlyContainsAttributeCerts 域；
- 如果 reasonCodes 域存在于发布的分布点扩展中，原因代码域必须包括所有应用的原因代码；
- 发布的分布点扩展可以包含或不包含 indirectCRL 域(该域不必被检查)。

此 CRL 仅当可信实体已经确定客体证书是一张终端实体证书时才能被使用。因此，对于第三版本的证书，如果客体证书包含 basicConstraints 扩展，其值应为 cA＝FALSE。

B.5.1.3 完整的 CARL

为了确定一个 CRL 对重要原因代码是一个完整的 CARL，所有以下条件必须是真实的：

- 不存在增量 CRL 指示器扩展；
- 必须存在发布的分布点；
- 发布的分布点必须不包含分布点域；
- 发布的分布点必须不包含被设置成 TRUE 的 onlyContainsUserCerts 域；
- 发布的分布点扩展必须不包含设置成 TRUE 的 onlyContainsAttributeCerts 域；
- 发布的分布点必须包含设置成 TRUE 的 onlyContainsAuthorityCerts 域；
- 如果 reasonCodes 域存在于发布的分布点扩展中，原因代码域必须包括所有应用的重要原因代码；
- 发布的分布点扩展可以包含或不包含 indirectCRL 域(因此，这个领域不必被检查)。

仅当客体证书是一张 CA 证书时，CARL 才能被使用。这样，对第三版本的证书，客体证书必须包含值为 cA＝TRUE 的 basicConstraints 扩展。

B.5.1.4 基于分布点的 CRL/EPRL/CARL

为确定一个 CRL 是证书中 CRL 分布点扩展指示的一种 CRL，以下条件都必须是真实的：

- 要么 CRL 发布的分布点扩展中不存在分布点域(仅当不寻找关键 CRL DP 时)，要么证书中 CRL 分布点扩展的分布点域的命名必须匹配 CRL 发布的分布点扩展中分布点域中的命名。换句话说，证书 CRL DP 中 cRLIssuer 域的一个命名能匹配 IDP 的 DP 中的一个命名。
- 如果证书是一个终端实体证书，CRL 必须不包含 CRL 发布的分布点扩展中被设置成 TRUE 的 onlyContainsAuthorityCerts 域。
- 如果 onlyContainsAuthorityCerts 在 CRL 发布的分布点扩展中被设置成 TRUE，被检查的证书必须包括带有被设置成 TRUE 的 cA 组件的 basicConstraints 扩展。
- 如果原因代码域出现在证书的 CRL 分布点扩展中，该域要么在 CRL 的发布的分布点扩展不存在，要么至少包括证书的 CRL 分布点扩展中声明的一个原因代码。
- 如果 cRLIssuer 域在证书的 CRL 分布点扩展不存在，则 CRL 必须由签名该证书的相同 CA 签名。
- 如果 cRLIssuer 域在证书的 CRL 分布点扩展不存在，CRL 必须由证书 CRL 分布点扩展中指定的 CRL 发布者签名，并且 CRL 必须包含发布的分布点扩展中的 indirectCRL 域。

B.5.2 验证增量 CRL 范围

可信实体也可检查 Δ-CRL，或者为证书中的关键 freshestCRL 扩展所需要，或者因为可信实体运行下的策略支持 Δ-CRL 检查。

如果满足以下所有条件，可信实体能一直确信它拥有该证书的适当 CRL 信息：

国家图书馆专用

- 可信实体使用的基本 CRL 对证书来说是适当的(在一定范围之内)。
- 可信实体使用的增量 CRL 对证书来说是适当的(在一定范围之内)。
- 此基本 CRL 是在 Δ-CRL 调用基本 CRL 时或晚于此时发布的。

为确定 Δ-CRL 对证书来说是否合适,以下所有条件必须是真实的:

- 存在增量 CRL 指针扩展。
- Δ-CRL 必须在基本 CRL 之后发布。一种确认方式是检查 Δ-CRL 的 crINumber 扩展中的 CRL 编号是否大于可信实体正在使用的基本 CRL 的 crINumber 扩展中的 CRL 编号,并且基本 CRL 和 Δ-CRL 中的 cRLStreamIdentifier 域匹配。这个方法可能需要辅助逻辑来计算编号。另一种方法是比较可信实体使用的基本 CRL 和 Δ-CRL 中的 thisUpdate 域。
- 可信实体使用的基本 CRL 必须是 Δ-CRL 发布时或之后发布的。一种确认方式是检查 Δ-CRL 的 deltaCRLIndicator 扩展中,CRL 编号是否小于或等于可信实体所使用的基本 CRL 的 crlNumber 扩展中的 CRL 编号,并且基本 CRL 和 Δ-CRL 中的 cRLStreamIdentifier 域是否匹配。这种方式可能需要辅助逻辑来计算编号。另一种方式是将可信实体拥有的基本 CRL 的 thisUpdate 域和 Δ-CRL 所指向的基本 CRL 进行比较。还有另一种方法是将可信实体拥有的基本 CRL 中的 thisUpdate 域和可信实体拥有的 Δ-CRL 中的 baseUpdateTime 扩展进行比较。

应注意可信实体通常可以通过将 Δ-CRL 应用于基本 CRL 上来构造一个基本 CRL,只要通过检验 crlNumberc 和 cRLStreamldentifer 能够满足上述两条规则。在这种情况下,新的基本 CRL 的 crlNumber 扩展和 thisUpdate 域都由 Δ-CRL 指定。为了与其他 Δ-CRL 建立连接,可信实体不知道并且也不需要知道新的基本 CRL 的 nextUpdate 域。

- 如果 Δ-CRL 包含一个发布的分布点扩展,发布的分布点的范围应该与 B.5.1.4 中描述的证书一致;
- 如果 Δ-CRL 包含 CRL 范围扩展,证书应该在 CRL 范围内;
- 如果 Δ-CRL 不包含以下任何扩展:streamIdentifier,crlScope 和 issuingDistributionPoint,它将仅能与一个完整的基本 CRL 协作工作。

B.5.3 基本 CRL 上的有效性和当前状态检查

为了验证一个基本 CRL 是否是正确的并且自发布以后就未被修改,必须满足以下所有条件:

- 可信实体必须能通过验证机制获得在 CRL 中指定的发布者的公钥;
- 基本 CRL 上的签名必须使用已验证的公钥来进行校验;
- 如果 nextUpdate 域存在,当前时间必须优先于 nextUpdate 域。

CRL 中的发布者姓名必须匹配因撤销而被检查的证书中的发布者姓名,除非 CRL 是由证书中的 CRL DP 检索到的,并且 CRL DP 扩展包含 CRL 发布者组件。即使那样,CRL DP 扩展中 CRL 发布者组件的命名必须匹配 CRL 中发布者的姓名。

B.5.4 增量 CRL 的有效性检查

为了验证 Δ-CRL 是否是正确的,以及自从发布后就未被修改,必须满足以下所有条件:

- 可信实体必须能通过验证机制获得在 CRL 中指定的发布者的公钥。
- Δ-CRL 上的签名必须用已验证的公钥进行校验。
- 如果存在 nextUpdate 域,当前时间应该小于 nextUpdate 域。
- 在 Δ-CRL 中的发布者姓名必须匹配因撤销而被检查的证书中发布者的姓名,除非以下条件是真实的:
  - 增量 CRL 是从证书中的 CRL DP 检索到的,并且 CRL DP 扩展包含 CRL 发布者组件。即使那样,CRL DP 扩展中 CRL 发布者组件的命名必须匹配 CRL 中发布者的姓名。
  - DeltaCRL 是从证书中的 CRL DP 检索到的,并且 Δ-CRL 中的 CRL 范围扩展包含每一个带有匹配证书中发布者姓名的 authorityName 权威范围组件。

国家图书馆专用

附 录 C
（资料性附录）
增量 CRL 发布实例

C.1 介绍

对给定范围的证书，采用增量 crL 来发布 CRL 的模式有两种。

第一种模式，每个增量 CRL 都引用该证书集的最新发布的完整 CRL。在新的完整 CrL 发布之前，对给定范围的证书可以发布多个增量 CRL。给定范围的新发布的完整 CRL 是下一增量 crl 序列的基础，并且被引用在增量 crl 的扩展域中。当发布给定范围的最新的完整 CRL 时，也要签发前一个完整 CRL 的最后一个增量 CRL。

第二种模式，与第一种模式非常相似，不同之处在于增量 CRL 引用的 CRL 不必是完整 CRL，（即增量 CRL 所引用的 CRL 也可以是增量 CRL）。如果增量 CRL 引用的 CRL 是完整 CRL，那么它也可以不必是最新发布的完整 CRL。

证书应用系统在处理时必须同时拥有给定范围内的完整 CRL，该完整 CRL 至少与要处理的增量 CRL 中引用的 CRL 一致。这个完整 CRL 或者由可靠的权威机构发布或者通过证书应用系统本地构建。注意，在某些情况下，完整 CRL 与增量 CRL 中可能有重复的撤销信息。例如证书应用系统所拥有的完整 CRL 是在增量 CRL 所引用的 CRL 之后发布的。

下表演示了增量 CRL 应用的三个实例。例 1 是上述第一种模式所描述的传统的方法，例 2 和例 3 是第二种模式的两种不同的变化形式。例 2 中，权威机构每两天发布一次完整 CRL，增量 CRL 引用次新的完整 CRL。这种机制对于减少为了获得完整 CRL 同时访问存储库的用户数量很有用。在例 2 中，拥有最新的完整 CRL 与拥有次新的完整 CRL 的用户都可以使用相同的增量 CRL。在所使用的增量 CRL 发布时，这两类用户都可以获得给定范围内完整的证书撤销信息。

在例 3 中，与例 1 相同，完整 CRL 一周发布一次，但是每个增量 CRL 引用比自己早 7 天发布的作废信息。

在这里没有提供间接 CRL 使用增量 CRL 的例子，但是，这只是以上例子的扩展。

这些仅仅是一些演示示例，根据具体的策略还可以有其他的变化形式。建立策略时考虑的因素包括：用户数目、访问 CRL 的频率、CRL 的复制、存储 CRL 目录服务系统的负载均衡、性能、响应时间的要求，等等。

表 C.1 增量 CRL 应用实例

	例 1：增量 CRL 引用最新的完整 CRL		例 2：增量 CRL 引用次新的完整 CRL		例 3：增量 CRL 引用 7 天前的发布的证书作废信息	
	给定范围内完整 CRL	增量 CRL	给定范围内的完整 CRL	增量 CRL	给定范围内的完整 CRL 完全 CRL	增量 CRL
8	thisUpdate = day 8 nextUpdate = day 15 crlNumber=8	thisUpdate = day 8 nextUpdate = day 9 crlNumber=8 BaseCRLNumber=1	ThisUpdate = day 8 nextUpdate = day 10 crlNumber=8	thisUpdate = day 8 nextUpdate = day 9 crlNumber=8 BaseCRLNumber=6	thisUpdate = day 8 nextUpdate = day 15 cRLNumber=8	thisUpdate = day 8 nextUpdate = day 9 cRLNumber=8 BaseCRLNumber= 1

表 C.1(续)

	例 1:增量 CRL 引用最新的完整 CRL		例 2:增量 CRL 引用次新的完整 CRL		例 3:增量 CRL 引用 7 天前的发布的证书作废信息	
	给定范围内完整 CRL	增量 CRL	给定范围内的完整 CRL	增量 CRL	给定范围内的完整 CRL 完全 CRL	增量 CRL
9	未发布	thisUpdate = day 9 nextUpdate = day 10 crlNumber=9 BaseCRLNumber=8	未发布	thisUpdate = day 9 nextUpdate = day 10 crlNumber=9 BaseCRLNumber=6	未发布	thisUpdate = day 9 nextUpdate = day 10 cRLNumber=9 BaseCRLNumber= 2
10	未发布	thisUpdate = day 10 nextUpdate = day 11 crlNumber=10 BaseCRLNumber=8	ThisUpdate = day 10 nextUpdate = day 12 crlNumber=10	thisUpdate = day 10 nextUpdate = day 11 crlNumber=10 BaseCRLNumber=8	未发布	thisUpdate = day 10 nextUpdate = day 11 cRLNumber=10 BaseCRLNumber= 3
11-14	继续前几天的模式					
15	thisUpdate = day 15 nextUpdate = day 22 crlNumber=15	thisUpdate = day 15 nextUpdate = day 16 crlNumber=15 BaseCRLNumber=8	未发布	thisUpdate = day 15 nextUpdate = day 16 crlNumber=15 BaseCRLNumber=12	ThisUpdate = day 15 nextUpdate = day 22 cRLNumber =15	thisUpdate = day 15 nextUpdate = day 16 cRLNumber=15 BaseCRLNumber= 8
16	**未发布**	**thisUpdate = day 16 nextUpdate = day 17 crlNumber=16 BaseCRLNumber =15**	**ThisUpdate = day 16 nextUpdate = day 18 crlNumber=16**	**thisUpdate = day 16 nextUpdate = day 17 crlNumber=16 BaseCRLNumber =14**	**未发布**	**thisUpdate = day 16 nextUpdate = day 17 cRLNumber=16 BaseCRLNumber = 9**

国家图书馆

附 录 D
（资料性附录）
特权策略和特权属性定义实例

D.1 介绍

对特权管理而言，特权策略精确的定义在什么时候特权验证者应该能推断所存在的一组权限是充分的，并可以依此将资源（所请求的对象，资源，应用等）访问权授予特权声称者。正式的特权策略规范包含决定接受或拒绝特权声明者请求的规则，授予特权声明者的特权以及资源的敏感度等，因此它有助于特权验证者对特权声称者的特权相对于请求资源的敏感度进行自动的评估。

由于要确保特权验证者作决定所使用的特权策略的完整性，因此不仅要在签名对象中包括以对象标识符的形式存在的特权策略标识符和整个特权策略的散列值，还要存储在目录的条目中。但是，在本规范中没有标准化用于定义特权策略实例的特定语法。

D.2 语法实例

特权策略可使用包括纯文本的任何语法定义。为了帮助定义特权策略的人理解有关特权策略定义的多种选择，本附录提供了两个例子。必须强调这些仅仅是例子，通过使用属性证书或公钥证书的subjectDirectoryAttributes扩展来实现的特权管理不需要支持这些或任何其他的特定语法。

D.2.1 例 1

以下的 ASN.1 语法是全面、灵活定义特权策略的工具的一个例子。

```
PrivilegePolicySyntax      ::= SEQUENCE {
    version                 Version,
    ppe                     PrivPolicyExpression }
PrivPolicyExpression       ::= CHOICE {
    ppPredicate             [0] PrivPolicyPredicate,
    and                             [1] SET SIZE (2..MAX) OF PrivPolicyExpression,
    or                              [2] SET SIZE (2..MAX) OF PrivPolicyExpression,
    not                             [3] PrivPolicyExpression,
    orderedPPE                      [4] SEQUENCE OF PrivPolicyExpression }
```

注："Sequence"定义了验证特权的临时顺序。

```
    --privilege shall be examined
PrivPolicyPredicate        ::= CHOICE {
    present                         [0] PrivilegeIdentifier,
equality                            [1] PrivilegeComparison, --single/set-valued priv.
    greaterOrEqual          [2] PrivilegeComparison,-- single-valued priv.
    lessOrEqual             [3] PrivilegeComparison,-- single-valued priv.
    subordinate             [4] PrivilegeComparison,-- single-valued priv.
    substrings              [5] SEQUENCE {      -- single-valued priv.
        type                        PrivilegeType,
        initial             [0] PrivilegeValue OPTIONAL,
        any                             [1] SEQUENCE OF PrivilegeValue,
        final               [2] PrivilegeValue OPTIONAL },
```

国家图书馆提供

```
        subsetOf                                [6] PrivilegeComparison, -- set-valued priv.
        supersetOf                              [7] PrivilegeComparison, -- set-valued priv.
        nonNullSetInter         [8] PrivilegeComparison, -- set-valued priv.
        approxMatch             [9] PrivilegeComparison,
-- single/set-valued priv. (应用程序定义近似值)
    extensibleMatch             [10] SEQUENCE {
        matchingRule              OBJECT IDENTIFIER,
        inputs                    PrivilegeComparison }}
PrivilegeComparison     ::= CHOICE {
    explicit                                [0] Privilege,
-- the value(s) of external pribilege identified by
-- Privilege PrivilegeId is(are) compared with the value(s)
-- explicitly probided in Pribilege privilegeValueSet
byReference             [1] PrivilegeIdPair }
-- the value(s) of an external privilege identified by
-- PrivilegeIdPair firstPrivilege is(are)compared with
-- the value(s)of a second external privilege identified by
-- PrivilegeIdPair secondPrivilege
Privilege               ::= SEQUENCE {
    type                                    PRIVILEGE.&id ({SupportedPrivileges }),
    values                                  SET SIZE (0..MAX) OF
                                            PRIVILEGE.&Type ({SupportedPrivileges }{@type })
}

SupportedPrivileges     PRIVILEGE ::=       {... }
PRIVILEGE ::=           ATTRIBUTE
-- Privilege is analogous to Attribue

PrivilegeIdPair         ::= SEQUENCE {
    firstPrivilege          PrivilegeIdentifier,
    secondPrivilege         PrivilegeIdentifier }
PrivilegeIdentifier     ::= CHOICE {
    privilegeType           [0] PRIVILEGE.&id ({SupportedPrivileges }),
    xmlTag                              [1] OCTET STRING,
    edifactField            [2] OCTET STRING }
-- PrivilegeIdentifier extends the concept of AttributeType to other
-- (e.g,taggd) enbironments,such as XML and EDIFACT

Version                 ::= INTEGER {v1(0) }
```

具体的例子可以帮助阐明上述 PrivilegePolicy 结构的创建和使用。

考虑批准增加工资的特权。为简单起见,假定所实施的策略规定仅仅高级管理人员或更高的级别能批准增加工资,且仅能批准职位比自身低的(例如,主管可批准高级管理人员的,而不能批准副总经理的工资增加)。例如,假定有 6 个等级("技术职员"= 0,"管理人员"= 1,"高级管理人员"= 2,"主管"

国家图书馆专用

=3,“副总经理”= 4,“总经理”=5)。

假定在属性证书中验证级别的属性类型(“特权”)是对象标识符 *OID-C*,且在要修改工资域的数据库记录中验证级别的属性类型(“敏感度”)是对象标识符 *OID-D*(这些将在实际实现中用真正的对象标识符替换)。下列布尔表达式表示的“工资批准”策略(在 PrivilegePolicy 表达式中是很清楚的):

AND (NOT (lessOrEqual (value corresponding to OID-C, value corresponding to OID-D))

subsetOf (value corresponding to OID-C, {2, 3, 4, 5 }))

这套策略编码说明批准人的等级必须比被批准人高(像“NOT less-than-or-equal-to”那样表达),批准人必须是{高级管理人员,...总经理}之一,以便布尔表达式可以赋值为 TRUE。第一个特权比较是“参考”,比较两个实体所包括的属性类型“rank”;第二个特权比较是“明确的”,在此比较批准人特权“等级”的值和明确包含列表的值。因此在这种情况下,特权验证者需要用带有两个属性的结构对策略进行编码,一个属性与批准人有关,另一个与被批准人有关。批准人的属性(包含在属性证书中)值为{*OID-C* 3},并且,被批准人的属性(也许包含在数据库记录中)值为{*OID-D* 3}。比较批准人属性类型的属性值(在这个例子中是 3)和被批准人属性类型的属性值(在这个例子中是 3)导致了“lessOrEqual”表达式的值为假,这样第一个主管无权批准为第二个主管增加工资。另一方面,如果被批准人属性是{*OID-D* 1},主管将被授权批准为管理人员增加工资。

构想一些对上述表达式的有用添加项是不困难的。例如可以添加第三个‘and’,定义“当前时间”这个环境变量,——从本地时钟读取,然后作为对象标识类型 *OID-E* 的属性进行编码,“当前时间”必须位于特定的时间段,这个特定的时间段作为对象标识类型 *OID-F* 的属性在表达式里明确的指定。例如,当满足上述条件并且在上班时间提出加薪请求,允许增加工资。

D.2.2 例 2

最简单形式的安全策略是一套提供安全服务的标准。关于访问控制,安全策略是更高系统级安全策略的子集,它定义了在发送方和目标间实施访问控制策略的方法。访问控制机制必须:允许特定策略承认的通信,拒绝未明确承认的特殊策略的通信。

安全策略是访问控制机制作出决定的基础。特殊域的安全策略信息经由安全策略信息文件(SPIF)进行传送。

SPIF 是一个签名对象用以防止未授权修改。SPIF 包括用于解释包含在安全标签和清除属性中的访问控制参数的信息。如同安全策略定义的那样,清除属性中的安全策略标识符必须和特定的实现语法和语义相关。与特定安全策略相关的实现语法保留在 SPIF 中。

如同安全策略决定的那样,SPIF 传递授权和敏感交叉安全策略域间的等效性 ;提供安全标签的可打印表达式;并在选择数据对象的安全属性时,对可显示字串到安全等级以及表达式种类到终端用户进行映射。表达等效性映像,例如在安全策略域中产生的标签,域可由其他安全策略域中的应用操作解释。SPIF 将清除属性映射到信息安全标签域和显示给用户的表示标签中。如果映射成功,则检查接受者是否有接受数据对象的授权。

SPIF 包括以下的序列:

versionInformation —ASN.1 语法的版本。

updateInformation —SPIF 规范的语法和语义的版本。

securityPolicyIdData — SPIF 适用的安全策略。

privilegeId —包括在清除属性安全范畴中标识语法的 OID。

rbacId —和 SPIF 一起使用的安全种类语法的 OID。

securityClassifications —将安全标签映射到清除属性的分类中,并提供等价映射。

securityCategoryTagSets —将安全标签的安全分类映射到清除属性的安全范畴中,并提供等价映射。

equivalentPolicies —巩固所有 SPIF 内的相等策略。

国家图书馆检索中

defaultSecurityPolicyIdData—标识应用于不用安全标签就可接收数据的安全策略中。

extensions —提供一种机制用于包括所确定的未来需求的附加性能。

安全策略信息文件用以下语法定义：

```
SecurityPolicyInformationFile ::= SIGNED {SPIF }
SPIF ::=    SEQUENCE      {
    versionInformation                  VersionInformationData DEFAULT v1,
    updateInformation                   UpdateInformationData,
    securityPolicyIdData                ObjectIdData,
    privilegeId                          OBJECT IDENTIFIER,
    rbacId                              OBJECT IDENTIFIER,
    securityClassifications   [0]       SEQUENCE OF SecurityClassification OPTIONAL,
    securityCategories                  [1]  SEQUENCE OF SecurityCategory   OPTIONAL,
    equivalentPolicies                  [2]  SEQUENCE OF EquivalentPolicy   OPTIONAL,
    defaultSecurityPolicyIdData         [3]  ObjectIdData OPTIONAL,
    extensions                          [4]  Extensions OPTIONAL }
VersionInformationData ::= INTEGER {v1(0) }
UpdateInformationData      ::= SEQUENCE {
    sPIFVersionNumber                   INTEGER,
    creationDate                        GeneralizedTime,
    originatorDistinguishedName         Name,
    keyIdentifier                       OCTET STRING OPTIONAL }
ObjectIdData                ::=         SEQUENCE      {
    objectId                            OBJECT IDENTIFIER,
    objectIdName            DirectoryString {ubObjectIdNameLength }}
SecurityClassification    ::=    SEQUENCE      {
    labelAndCertValue       INTEGER,
    classificationName      DirectoryString {ubClassificationNameLength },
    equivalentClassifications           [0]  SEQUENCE OF EquivalentClassification OPTIONAL,
    hierarchyValue                      INTEGER,
    markingData                         [1]  SEQUENCE OF MarkingData OPTIONAL,
    requiredCategory                    [2]  SEQUENCE OF OptionalCategoryGroup OPTIONAL,
    obsolete                            BOOLEAN DEFAULT FALSE    }
EquivalentClassification ::= SEQUENCE {
    securityPolicyId        OBJECT IDENTIFIER,
    labelAndCertValue       INTEGER,
    applied                                 INTEGER {
                                            encrypt    (0),
                                            decrypt    (1),
                                            both       (2)}}
MarkingData ::= SEQUENCE {
    markingPhrase           DirectoryString {ubMarkingPhraseLength }OPTIONAL,
    markingCodes            SEQUENCE OF MarkingCode OPTIONAL }
MarkingCode ::= INTEGER {
```

国家图书馆专用

```
        pageTop                         (1),
        pageBottom                      (2),
        pageTopBottom                   (3),
        documentEnd                     (4),
        noNameDisplay                   (5),
        noMarkingDisplay                (6),
        unused                          (7),
        documentStart                   (8),
        suppressClassName               (9) }
OptionalCategoryGroup ::= SEQUENCE {
        operation               INTEGER {
                                onlyOne         (1),
                                oneOrMore       (2),
                                all             (3)},
        categoryGroup           SEQUENCE OF OptionalCategoryData }
OptionalCategoryData ::= SEQUENCE {
        optCatDataId            OC-DATA.&id({CatData}),
        categorydata            OC-DATA.&Type({CatData}{@optCatDataId}) }
OC-DATA ::= TYPE-IDENTIFIER
CatData OC-DATA ::= {... }
EquivalentPolicy                ::= SEQUENCE {
        securityPolicyId        OBJECT IDENTIFIER,
        securityPolicyName      DirectoryString {ubObjectIDNameLength }
OPTIONAL }
Extensions ::= SEQUENCE OF Extension
Extension ::= SEQUENCE {
        extensionId                             EXTENSION.&objId ({ExtensionSet}),
        critical                                BOOLEAN DEFAULT FALSE,
        extensionValue                          OCTET STRING }
```

注意 SPIF 实例是展开的语法和完整的定义，并且人们能在 ITU-T Rec. X841 或者 ISO/IEC 15816 安全信息对象中找到每个元素完全的定义和描述。

D.3 特权属性实例

以下传输特殊特权的属性实例仅作为实例而被提供。该语法和联合属性的规范包含在国际标准化组织/IEC 9594-2 | 国际电信联盟-T 的 Rec X.501 的 17.5 条款中。该特殊属性传递与命名实体相关的检查，包括与 DSA 通信的 DUA。

清除属性把清除和包括 DUAs 的命名实体联系起来。

```
clearance       ATTRIBUTE ::=       {
        WITH SYNTAX                     Clearance
        ID                              id-at-clearance }
Clearance       ::=     SEQUENCE {
        policyId                OBJECT IDENTIFIER,
        classList               ClassList DEFAULT {unclassified },
```

国家图书馆专用

```
        securityCategories      SET SIZE (1MAX) OF SecurityCategory OPTIONAL }
    ClassList       ::=     BIT STRING {
        unmarked            (0),
        unclassified        (1),
        restricted          (2),
        confidential        (3),
        secret              (4),
        topSecret           (5) }
```

个别的组件用参考文献中的特权规范描述。

国家图书馆专用

附 录 E
（资料性附录）
公钥密码学介绍

在传统的密码体制中，秘密消息的发送方使用密钥加密信息，合法接收方使用相同的密钥解密消息。

不过在公钥密码体制中(PKCS)，密钥成对出现，一个用于加密，另一个用来解密。每个密钥对都和特定用户 X 相联系。其中一个密钥，即所谓的公钥(Xp)是公开的，能被任何用户用来加密数据。仅仅拥有相应的私钥(Xs)的用户 X 可以解密数据。(可被描述为：D = Xs[Xp[D]])。不可能从公钥通过计算能够推导出对应的私钥。这样任何用户都能用 Xp 加密传送信息，这些信息只有 X 可以解密。进一步说，两个用户能用彼此的公钥来加密数据进行秘密地通信。如图 E.1 所示。

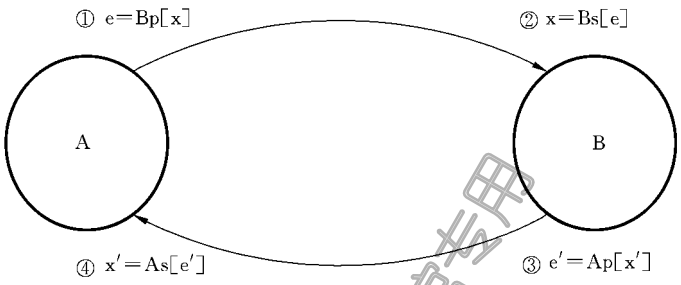


图 E.1 使用公钥密码体制交换秘密信息

用户 A 有公钥 Ap 和相应的私钥 As，用户 B 有另一对密钥 Bp 和 Bs。A 和 B 二者都知道彼此的公钥，但不知道对方的私钥。因此 A 和 B 可以用下列步骤交换秘密信息(如图 E.1 的图解)。

1) A 想发送一些秘密信息 x 给 B。因此 A 用 B 的公钥加密 x，并且将加密后的信息 e 发送给 B。可表示为：

$$e = Bp[x]$$

2) 现在 B 通过使用自己的私钥 Bs 得到信息 x。注意 B 是私钥 Bs 的唯一的拥有者，并且，由于私钥决不会公开，并且不被发送，因此任何第三方都不可能得到信息 x。Bs 的拥有决定 B 的身份。解密操作表示为：

$$x = Bs[e], \text{ or } x = Bs[Bp[x]]$$

3) 现在 B 同样地使用 A 的公钥 Ap，发送一些秘密信息 x′给 A：

$$e' = Ap[x']$$

4) A 解密 e′得到 x′：

$$x' = As[e'], \text{ or } x' = As[Ap[x']]$$

通过这种方法，A 和 B 已经交换了秘密信息 x 和 x′。如果他们的私钥不被泄密，信息不会被 A 和 B 以外的任何人得到。

这样的交换，除了当事方之间传递秘密信息，也可以用于验证彼此的身份。A 和 B 通过拥有私钥 As 和 Bs 而分别验证身份。通过在 B 发送给 A 的秘密信息 x′中包含 A 发送给 B 的秘密信息 x 中部分信息，A 可以确认 B 是否拥有私钥 Bs，这将告诉 A，他正在和 Bs 的拥有者通信，B 通过同样的方式可以验证 A 的身份。

某些公钥密码体制中解密和加密的步骤能颠倒过来，如 D=Xp[Xs[D]]。这使得由 X 产生的信息，能被任何拥有 Xp 的用户阅读。因此，这可用于信息来源的认证，并作为数字签名的根据。附录 D 中有关于这样算法的描述。

国家标准专用

附 录 F
（规范性附录）
算法对象标识符的参考定义

在没有正式注册前，该附录定义了分配给认证和加密算法的对象标识符。如果它可用，则必须进行了注册。该定义用 ASN.1 模块 —“AlgorithmObjectIdentifiers”描述。

```
AlgorithmObjectIdentifiers {joint-iso-itu-t-ds(5)module(1)
algorithmObjectIdentifiers(8) 4 }
DEFINITIONS ::=
BEGIN
—— EXPORTS All——
——定义在该模块中的类型和值被输出用于其他所包含的 ASN.1 模块。
——在该目录规范内部，以及适合于用其进行访问的其他应用程序的使用。
——目录服务。其他应用程序可以为其自身目的使用它们，但是并不强制使用。
——扩展和更改需要维护或改善这个目录服务。
    IMPORTS
        algorithm, authenticationFramework
        FROM UsefulDefinitions {joint-iso-itu-t ds(5) module(1) usefulDefinitions(0) 4 }
        ALGORITHM
        FROM AuthenticationFramework authenticationFramework    ;

——对象标识符的种类——
        encryptionAlgorithm         OBJECT IDENTIFIER    ::=    {algorithm 1 }
        hashAlgorithm               OBJECT IDENTIFIER    ::=    {algorithm 2 }
        signatureAlgorithm          OBJECT IDENTIFIER    ::=    {algorithm 3 }
——同义字——
        id-ea                       OBJECT IDENTIFIER    ::=    encryptionAlgorithm
        id-ha                       OBJECT IDENTIFIER    ::=           hashAlgorithm
        id-sa                       OBJECT IDENTIFIER    ::=           signatureAlgorithm
——算法——
    rsa   ALGORITH          ::=         {
        KeySize
        IDENTIFIED BYid-ea-rsa }
    KeySize                         ::=      INTEGER
——对象标识符的分配——
        id-ea-rsa                   OBJECT IDENTIFIER      ::=      {id-ea 1 }
——下列对象标识符分配保留值，这个保留值分配在与之相抵触的函数中。
    id-ha-sqMod-n                   OBJECT IDENTIFIER    ::=    {id-ha 1 }
id-sa-sqMod-nWithRSA                OBJECT IDENTIFIER    ::=    {id-sa 1 }
END
```

国家图书馆专用

附 录 G
（资料性附录）
认证路径约束的使用实例

G.1 例 1:基本约束的使用

设想一个装饰有限公司想与 Acme 集团的中心 CA 进行交叉认证,而且装饰有限公司只想使用由中心 CA 直接签发的终端实体证书,而不想使用由中心 CA 所认证的其他 CA 所签发的证书。

这家装饰有限公司可以通过以下这个方法满足上述需求,即为 ACME 集团的中心 CA 签发带有下列扩展项的证书:

基本约束扩展的值:

{cA TRUE, pathLenConstraint 0 }

G.2 例 2:名称约束的使用

假设这个装饰有限公司想与 Acme 集团的中心 CA 进行交叉认证,而且装饰有限公司只想使用 Acme 签发的、证书的主题满足下列标准的证书:

- 位于美国的 Acme 有限公司,除了采购部主体外,其他主题都是可接受的;
- 位于法国的 EuroAcme,只有那些是 EuroAcme 总部直接下属的主题是可接收的(这包括直接属于总部的个体,不包括属于下级组织的个体);
- 位于英国的 Acme 有限公司,除了那些 R&D 组织单位的下级组织,所有的主题都是可接受的(这包括直接属于 R&D 的个体,,但是不包括那些属于 R&D 下级单位组织的个体)。

这家装饰有限公司可以通过以下这个方法满足需求,即为 ACME 集团的中心 CA 签发证书,该证书包括下列扩展:

基本约束扩展的值:

{cA TRUE }

名字约束扩展的值:

{**permittedSubtrees** {{**base --Country=US, Org=Acme Inc--**},
{**base --Country=France, Org=EuroAcme--, maximum 1** },
{**base --Country=UK, Org=Acme Ltd--**}},
excludedSubtrees {{**base --Country=US, Org=Acme Inc, Org. Unit=Purchasing-**},
{**base --Country=UK Org=Acme Ltd., Org. Unit=R&D--, minimum 2** }}}

G.3 例 3:策略映射和策略约束的使用

假设下列交叉认证需要在加拿大和美国政府之间进行:

a) 对于被称做 *Can/US-Trade* 的加拿大政府策略,加拿大政府的 CA 希望确认美国政府签名的用途。

b) 美国政府有一个策略叫 *US/Can-Trade*,这是加拿大政府预定的,考虑同其 *Can/US-Trade* 策略相等同的策略。

c) 加拿大政府希望实施如下安全措施:所有美国证书都必须明确声明支持以上策略并能抑制美国领土内的其他策略的映射。

一个加拿大政府的 CA 可以为美国政府的 CA 签发带有下列扩展的证书:

证书策略扩展的值;

国家标准信息

{{**policyIdentifier -- object identifier for Can/US-Trade --**}}

策略映射字段值：

{{**issuerDomainPolicy -- object identifier for Can/US-Trade --** ,
subjectDomainPolicy -- object identifier for US/Can-Trade -- }}

策略约束字段值：

{{**policySet** {-- ***object identifier for Can/US-Trade*** -- }, **requireExplicitPolicy** (0),
inhibitPolicyMapping (0)}}

国家图书馆专用

附 录 H
（资料性附录）
信息术语定义字母表

附录 H 提供了一个按英文字母排序的关于证书和 CRL 格式的定义、证书扩展、对象类，名称格式、属性类型和定义在目录规范中的匹配规策的索引。

表 H.1 信息术语定义字母表

项 目	条 款
证书和 CRL 格式	
属性证书格式	12.1
证书撤销列表	7.3
公钥证书格式	7
证书，CRL& CRL 实体扩展	
可接受的证书策略扩展	15.5.2.3
可接受的特权策略扩展	15.1.2.4
属性描叙扩展	15.3.2.2
权威属性标识符扩展	15.5.2.4
权威密钥标识符扩展	8.2.2.1
基本更新扩展	8.6.2.5
基本属性约束扩展	15.5.2.1
基本约束扩展	8.4.2.1
证书发布者扩展	8.6.2.3
证书策略扩展	8.2.2.6
CRL 分布点扩展	8.6.2.1
CRL 号扩展	8.5.2.1
CRL 范围扩展	8.5.2.5
CRL 流标识符扩展	8.5.2.7
委托名称约束扩展	15.5.2.2
Delta CRL 指示器扩展	8.6.2.4
Delta 信息扩展	8.5.2.9
扩展密钥使用扩展	8.2.2.4
新 CRL 扩展	8.6.2.6
控制指示代码扩展	8.5.2.3
约束及策略扩展	8.4.2.4
无效日期扩展	8.5.2.4
发布者选择名扩展	8.3.2.2
发布者分布点扩展	8.6.2.2

国家图书馆专用

表 H.1(续)

项　　目	条　　款
密钥使用扩展	8.2.2.3
名字约束扩展	8.4.2.2
不撤销信息扩展	15.2.2.2
顺序列表扩展	8.5.2.8
策略约束扩展	8.4.2.3
策略映射扩展	8.2.2.7
私钥使用期扩展	8.2.2.5
原因代码扩展	8.5.2.2
角色规范证书标识符扩展	15.4.2.1
SOA 标识符扩展	15.3.2.1
状态参考扩展	8.5.2.6
主体选择名扩展	8.3.2.1
主体公钥标识符扩展	8.2.2.2
主体目录属性扩展	8.3.2.3
目标信息扩展	15.1.2.2
时间规范扩展	15.1.2.1
用户通知扩展	15.1.2.3
对象类和命名形式	
属性证书 CRL 分布点对象类	17.1.4
证书策略和 CPS 对象类	11.1.5
CRL 分布点对象类和名称格式	11.1.3
Delta CRL 对象类	11.1.4
PKI CA 对象类	11.1.2
PKI 认证路径对象类	11.1.6
PKI 用户对象类	11.1.1
PMI AA 对象类	17.1.2
PMI 委托路径	17.1.5
PMI SOA 对象路径	17.1.3
PMI 用户对象路径	17.1.1
特权策略对象类	17.1.6
目录属性	
AA 证书属性	17.2.2
AA 证书撤销列表属性	17.2.5
属性证书属性	17.2.1
属性证书撤销列表属性	17.2.4

国家图书馆专用

表 H.1(续)

项　　目	条　　款
属性描叙符证书属性	17.2.3
权威撤销列表属性	11.2.5
CA 证书属性	11.2.2
证书实践声明属性	11.2.8
证书策略属性	11.2.9
证书撤销列表属性	11.2.4
交叉证书对属性	11.2.3
委托路径属性	17.2.6
Delta 撤销列表属性	11.2.6
PKI 路径属性	11.2.10
特权策略属性	17.2.7
支持算法属性	11.2.7
用户证书属性	11.2.1
匹配规则	
AA 标识符匹配	15.5.2.4.1
合理的证书策略匹配	15.5.2.3.1
算法标识符匹配	11.3.7
属性证书完全匹配	17.3.1
属性证书匹配	17.3.2
属性描叙符匹配	15.3.2.2.1
基本属性约束匹配	15.5.2.1.1
证书完全匹配	11.3.1
证书列表完全匹配	11.3.5
证书列表匹配	11.3.6
证书匹配	11.3.2
证书对完全匹配	11.3.3
证书对匹配	11.3.4
委托名称约束匹配	15.5.2.2.1
委托路径匹配	17.3.4
持有者发布者匹配	17.3.3
PKI 路径匹配	11.3.9
策略匹配	11.3.8
角色规范证书 ID 匹配	15.4.2.1.1
时间规范匹配	15.1.2.1.1

国家图书馆专用

中 华 人 民 共 和 国
国 家 标 准
信息技术 开放系统互连 目录
第8部分:公钥和属性证书框架
GB/T 16264.8—2005/ISO/IEC 9594-8:2001
*
中国标准出版社出版发行
北京西城区复兴门外三里河北街16号
邮政编码:100045
http://www.spc.net.cn
电话:63787337、63787447
2005年9月第一版 2005年9月电子版制作
*
书号:155066·1-23583

版权专有 侵权必究
举报电话:(010)68533533