西北师范大学

硕士学位论文

PKI中证书撤销机制和具有前向安全性的数字签名研究

姓名：贾续涵

申请学位级别：硕士

专业：计算机软件与理论

指导教师：王彩芬

20070501

摘　要

公开密钥基础设施(Public Key Infrastructure, PKI)是以公钥密码系统为基础、提供安全服务的通用性安全基础设施，在网络传输与信息保密过程中提供密钥的产生、分发、管理、撤销的服务，也提供身份鉴别、信息完整性、不可否认性和机密性的信息安全服务。公钥证书通过数字签名和加密方式把用户身份信息与公钥绑定在一起，确保持证者身份真实、合法、有效，这样就可以降低网络环境下信息传输的风险。

证书在发布后，其合法性会随着时间的推移以及突发的特殊原因而成为无效的证书。因此，证书在发行之后，CA 须为用户提供证书撤销服务供查询以验证证书的有效性。如何实施并维护一个高效的证书撤销机制成为 PKI 中的一个重要研究课题。

在分析了传统的证书撤销机制的基础上，提出了自根向下压缩的二叉排序树的证书撤销方案，为 PKI 的实现提供了几种高效可行的证书撤销方案，使用我们的证书撤销方案可以有效的减少通信代价和维护证书撤销后数据的工作量。这样就可以构建一个迅速、准确和安全的证书撤销信息的检查机制，以适应不同规模的 PKI 应用的需求。

前向安全的数字签名方案可以解决普通数字签名密钥泄露后对系统的影响，将损失降低到最小程度。它将密钥有效期划分为若干个时间段，不同的时间段所用的秘密钥由前一个时间段的秘密钥经一个单向函数运算生成，而公钥在整个签名过程中保持不变，既保证了系统的前向安全性，又简化了验证过程。在对已有的方案进行分析之后，发现了其中的安全漏洞，总结得出具有前向安全性质的签名方案应具备的特征。依此提出了一种基于双线性映射的具有前向安全的概率签名方案，方案具有实现简单，安全性高，密钥更新快的特点。同时还给出了方案的有效性、安全性及前向安全性证明。

关键词：公钥基础设施(PKI)；证书撤销；证书吊销二叉排序树；前向安全

Abstract

Public Key Infrastructure (PKI) offerring the general secure service is based on public key cryptography. It not only provides secure services for network, such as confidentiality, integrity, authentication and non-repudiation, but also generates, distributes, manages and withdraws digital certificate. The public key certificate binds public key and the user's ID information together through digital signature and encryption technique. The verification party verifies the validity of the certificate and ensures the rightness, validity and availability of the identity who holds the certificate before dealing with E-business affair. Thus the risk of transaction on Internet will be reduced and the security of the exchange in E-business can be ensured.

After CA has issued a certificate, it probably changes to invalid for some special reasons over time passes. Therefore, CA must publish Certificate Revocation List (CRL) for customers to download and verify the validity of the certificate. In consequence, how to maintain and implement an efficient certificate revocation is very an important subject in PKI.

Based on the analysis of several revocation schemes, such as Over-Issued-Segmented CRL, random CRL, improved Delta-CRL, CRT, Binary sorted certificate revocation tree and other traditional certificate revocation schemes, the study proposes a new binary sort certificate revocation tree scheme which compressed from root to leaf. The scheme can reduce both the work when CA updates the tree and communication between Directory and the verifier. It is a good revolution for certificate revocation in PKI. It can build a prompt, accurate and secure certificate revocation system which can offer good query service to users.

In order to reduce loss when the secret key of the common digital signature is revealed, the forward secure signature scheme is proposed.After pointing out the flaws of the exist scheme, this paper proposes a forward secure digital signature scheme based on bilinear pairings. The study divides the signature course into several periods. In each period, the public key is steady, but private key always changes. Furthermore, the next private key is produced by one-way function on the current secret key, so this scheme satisfies forward security. The new signature scheme realizes easily and updates the secret key promptly. The thesis also gives the proof on security and efficiency forward security.

Key words: Public Key Infrastructure (PKI); certificate revocation; binary sort certificate revocation tree; forward security; probability signature

独创性声明

本人声明所呈交的论文是我个人在导师指导下进行的研究工作及取得的研究成果。尽我所知，除了文中特别加以标注和致谢的地方外，论文中不包括其他人已经发表或撰写过的研究成果，也不包含为获得西北师范大学或其他教育机构的学位或证书而使用过的材料。与我一同工作的同志对本研究所做的任何贡献均已在论文中作了明确的说明并表示了谢意。

签名：贾续涌　　日期：2007.5.20

关于论文使用授权的说明

本人完全了解西北师范大学有关保留、使用学位论文的规定，即：学校有权保留送交论文的复印件，允许论文被查阅和借阅；学校可以公布论文的全部或部分内容，可以采用影印、缩印或其他复制手段保存论文。

（保密的论文在解密后应遵守此规定）

签名：贾续涌　导师签名：王录芳　日期：2007.5.20

第一章 绪论

1.1 课题研究的背景、目的和意义

计算机网络技术、微电子技术的发展推动了信息革命，信息化把人们带进高速度、多媒体、智能化、全球一体化的信息环境，使人们跨入数字化生存的新环境，信息技术已渗透到人类生活的各各方面。电子商务、电子政务等新技术的应用就是成功范例。但随着网络应用不断扩展，安全问题倍受到人们的特别关注。通过网络进行交易和商业活动以及信息传输，面临的最大问题是如何建立相互之间的信任关系以及如何保证信息的真实性、完整性、机密性和不可否认性。它关系到电子商务主体的利益与电子交易的合法性、公平性、有效性以及信息的真实性、有效性和可靠性，直接影响到电子商务平台是否可行及信息社会正常的秩序。构建一个提供安全服务技术平台将为网络应用提供一个安全保障，是信息交换、电子商务和电子政务实施的先提条件。

完善的公钥基础设施(PKI)是目前唯一的能够基本全面解决安全问题的服务设施[1-6]。PKI 通过管理这些电子证书，维持网络世界的秩序。通过提供一系列的安全服务，为网络中的信息传输、电子商务、电子政务提供有力的安全保障[7-10]。各国政府和许多民间机构都在纷纷研究和开发 PKI 产品，以望走在信息安全领域的前列。以 PKI 为基础服务的商业需求正日益增加，PKI 的应用也将随之日益增加，它能更好地满足未来的网络应用的安全需求。

尽管国内外许多公司已开发出 PKI 产品，并得到应用，但 PKI 市场尚未发育成熟，处于一个待发展和完善的阶段。许多新的技术和方案的不断出现，PKI 的结构、采用的算法、证书撤销的机制、密钥管理的方案等也在不断发展变化。过去的两年和未来的几年将是 PKI 的高速成长和完善的时期，大量的 PKI 支持协议标准将进一步提出并加以完善，建立在这些统一标准上，真正意义上的具有互操作性的产品也会很快出现。国外、国内许多有远见的公司也正开始研发 PKI 的产品。在这样一个应用背景下，进行一些改进算法的研究是有意义的。

1997 年，Anderson 首次提出了前向安全签名的概念[11]。前向安全签名的基本思想是：把整个有效时间分成若干个时间段，每一时段的密钥由前一个时段的密钥经一单向函数生成，在每个时间段内使用当前时段的的签名密钥产生签名，而验证签名的公钥在整个有效时间内都保持不变[11-18]。即使当前时间段的签名密钥被泄露，

由于无法得到以前的密钥，就不能伪造以前的签名，以前时间段的签名仍是有效的，从而减少了由于签名密钥泄露而对系统带来的负面影响。由于具有前向安全性质的密码体制有着广泛的应用，对此进行研究并对已有方案进行改进是有意义的。

1.2 国内外研究现状和发展趋势

总的来说，PKI 是建立在公钥密码体制的基础上的，用来给各种需要安全保证的实体提供安全服务的一整套基础设施。它具有统一化、标准化的特点，还应具备良好的互操作性和可扩展性。目前公钥密码技术已经成熟并且得到广泛的应用，因此，PKI 存在的密码基础是比较完善了。它存在的问题主要就是体系结构的问题、信任模型问题、认证机构建设问题、标准化问题、法律化问题等，这些问题严重影响着 PKI 的互操作性，从而影响着 PKI 应用范围的扩展，其成熟的商业化应用也是举步维艰。不可忽视的是这些问题成为阻碍 PKI 发展的重要因素，解决这些问题将使 PKI 成为公认的在大型开放网络环境下解决信息安全问题最可行、最有效的办法，因而，各国政府也都非常重视 PKI 技术的完善与实施。从 90 年代初期以来，美国、加拿大、英国、德国、日本和新加坡等国相继开展了可信第三方认证体系的研究和建设工作。

在研究的同时，国外的 PKI 应用也已经开始，很多厂家如 E3altimore Technologies, Entrust 和 Microsoft 等都推出了相应的 PKI 产品;有些公司如 VeriSign 已经开始提供 PKI 业务;由美国 National Security Agency 推动的 DOD PKI 研究也正积极地进行着;加拿大政府公开密钥基础设施 GOCPKI(Government Of Canada Public-Key Infrastructure)是世界上最早的大规模政府的 PKI 计划，已在各行各业应用取得了成效。

在我国，很多城市已经建立了 CA 认证中心，以便为本地化通信网络提供安全服务(CA认证中心是PKI的重要核心组成部分,有时将CA认证中心称作简要的PKI。)在国家直属部门，以中国人民银行为首的 12 家金融机构推出了“中国金融认证中心（CFCA）”，中国电信、中国邮政也在开展 CA 机制的试验与运作。另外，许多网络通信公司正在积极开发自己的基于 PKI 的安全产品。但是可以看到，各地方推出的 PKI 基本上只是为了满足局部需要，全国还没有由政府或金融机构推出的统一的信任认证机制，这将会影响今后跨部门之间的信息流通与合作，也会阻碍企业电子商务、经济信息一体化的发展，不过目前政府也已经开始出面组织全国

范围的PKI建设工作。

不仅如此欧美等国及亚洲的韩国在这方面的立法工作已经比较健全，各国、各洲、全球都为了推动相应领域的互操作性问题而创办了 PKI 论坛。随着各方面的努力以及各种电子化应用的不断深入，法律对PKI的支持，PKI同时也为电子商务中出现的各种纠纷提供依据，这将使 PKI 实现成功的商业化运作和发展提供机遇。

但随着PKI的范围的不断扩大，原来PKI的一些体系结构出现一些新的问题，例如：证书吊销采用的CRL是X.509标准方案[19-21]。CRL随着PKI中吊销的证书的数量的增加而增加，数据统计表明，证书吊销的数量为PKI发行数量的百分之十。庞大的证书吊销列表及不完善的证书吊销体制，带给查询者诸多不便。通信代价太大、网络带宽需求过高等因素，严重地制约了PKI发展的规模。

前向安全在加密、签名、秘密共享等技术中有着广泛的应用。它在密钥的生存期内，将密钥生存期分为 T 个时段，每一时段的私钥由前一时段的私钥经一单向函数运算生成，使得整个密钥生存周期内，各时段的秘密钥均不相同，而公钥保持不变，增加了安全性。由于根据当前时段不能计算出以前的密钥，使得系统具有前向安全的特点。

1997年，Anderson首次提出了前向安全签名的概念[11]。Bellare 和 Miner 第一次给出了前向安全签名的正式定义，并基于 A. Fiat 和 A. Shamir 的签名方案给出了两个前向安全签名方案[23]。此后，在各种加密体制下，都提出了具有前向安全性质的方案，并将之引用到各种安全技术中。然而，在已提出的许多方案中，有些算法存在着漏洞，这些方案不具备前向完全性，甚至已不具备一般意义上的完全性，如文献[24][25]。

1.3 本文的主要工作

在对 PKI 概念、标准、应用实施等理论进行研究的基础上，对各种证书撤销机制进行了详细的分析，并提出了自根向下压缩的二叉排序HASH树的证书撤销方案。

本文所提方案使用安全的HASH函数从树根向叶子结点方向压缩，树中所有结点的信息汇聚到叶子结点，每个叶子结点包括从根结点至此结点路径上所有结点的关键字值的信息。然后，利用二叉树中叶子结点的空链域重新组织成一种拓扑结构，CA 对其中一个汇集了整个树中结点信息的结点进行签名，就可以保证用户

的验证性查询的需要。对树中结点的更新，也不需要再向根方向的上层结点进行更新，只向下更新到叶子结点。因为二叉树的结点大部分在底层，这就使得更新效率大幅度提高。方案中二叉排序树的结构没有变化，查询效率就不受影响。返回的查询结果也不需要整个二叉树，只是相关的路径上的结点，没有增加传递信息的数据量。所提出方案给一些较大规模的 PKI 提供了可行的证书吊销方案。

在对已有的前向安全方案经过分析和研究的基础上，发现其中的安全漏洞，总结得出了前向安全的一般原则，并提出了一种新的基于双线性映射的具有前向安全性质的概率签名方案。

1.4 本文的内容结构

文章内容与章节安排：

第一章介绍了课题研究的背景、目的和意义以及国内外研究现状和发展趋势。

第二章介绍了 PKI 公钥基础设施及其体系结构，指出了证书的签发方式，并综述了 PKI 中的证书撤销。

第三章给出了证书撤销模型及评价标准，讨论了在工程应用中最为广泛的证书撤销列表机制，定量比较了证书撤销列表、增量 CRL、分段 CRL 及重复发布 CRL。同时分析了基于动态数据结构的证书撤销机制，包括 CRS, CRT 以及在线证书状态协议和简单证书验证协议等，指出了各种证书吊销机制的优缺点。

第四章针对传统的证书吊销所存在的不足，提出了自根向下压缩的二叉排序 HASH 树的证书吊销方案，并分析了方案的通信代价与效率。

第五章给出一个基于双线性映射的具有前向安全的概率签名方案。

结束语部分总结了相关工作和进一步研究的方向与需要解决的问题。

第二章 PKI概述

2.1 PKI概述

2.1.1 公钥基础设施(PKI)

公钥基础设施(PKI)是一个基于公钥密码算法原理和技术来实现并提供安全服务的具有通用性的安全基础设施[1-6]。PKI 是一种遵循一定的标准，并利用公钥加密技术为信息传输、电子商务、电子政务的开展提供一整套安全服务的基础设施，用户利用 PKI 平台提供的安全服务进行安全通信。PKI 是一种遵循标准的密钥管理平台，体现的是一种先进的密钥管理思想，它能够为所有网络应用透明地提供采用加密和数字签名等密码服务所必需的密钥和证书管理等服务。PKI 作为目前公共网络应用环境中信息安全最全面的解决方案，具有普适性的特点。

使用基于公开密钥技术平台的用户建立安全通信信任机制的基础是：网上进行的任何需要提供安全服务的通信都是建立在公钥的基础之上的，而与公钥成对的私钥只掌握在他们与之通信的对方。这个信任的基础是通过公钥证书的使用来实现的。公钥证书就是用户的身份及其所持有的公钥的绑定，在绑定之前，由一个可信任的权威机构——认证机构(CA)核实用户的身份，然后由可信任的 CA 对该用户身份及对应公钥进行绑定，对绑定的信息进行数字签名，用来证明证书的有效性，就形成了公钥证书。

PKI 首先必须具有可信任的认证机构，在公钥加密技术基础上实现证书的产生、管理、存档、发放以及证书撤销管理等功能，并包括实现这些功能的硬件、软件、人力资源、相关政策和操作规范以及为 PKI 体系中的各成员提供全部的安全服务，例如，身份认证、数据保密、完整性以及不可否认性等服务。

2.1.2 PKI 的体系结构

构建实施一个 PKI 系统主要包括以下内容：

(1)认证机构

认证机构 CA(Certification Authority)是证书的签发机构，它是整个 PKI 系统的核心，是 PKI 应用中权威的可信任的公正的第三方机构[36]。CA 负责生成和管理 PKI 结构下的所有用户(包括各种应用程序)的证书，把用户的公钥和用户的其他实体身份信息捆绑在一起，供网上验证用户的身份。

(2)注册中心 RA

注册中心 RA (Registration Authority)是认证中心 CA 的延伸部分，它与 CA 在逻辑上是一个整体，执行不同的功能。RA 按照特定的政策和管理规范对用户的资格进行审查，以决定是否为该用户发放证书。如果审查通过，即可实时或批量地向 CA 提出申请，来为用户签发证书。

(3)证书发布库

证书发布库是证书的集中存放地，提供公众查询。证书库可以是关系数据库，也可以是目录(目录服务器)、响应器等。

(4)密钥备份及恢复系统

对用户的解密密钥进行备份，当丢失时进行恢复，而用于签名的密钥则不能备份和恢复。

(5)证书撤销处理系统

证书由于某种原因需要作废、终止使用，将通过证书撤销系统如证书撤销列表 CRL (Certificate Revocation List)来实现。

(6) PKI 应用接口系统

为各种各样的应用提供安全、一致、可信任的方式与 PKI 交互，确保所建立起来的网络环境安全可靠并降低管理成本。

一个典型、完整、有效的 PKI 应用系统是由多种认证机构及各种终端实体组件所构成，其结构模式一般为多层次的树状结构。PKI 公钥证书系统的构成是基于 X.500 系列目录协议的，PKI 公钥证书是基于用于目录存储认证的 X.509 公钥证书标准发布的。图 2.1 给出了 PKI 证书管理和应用系统的一个通用模型结构[37]。

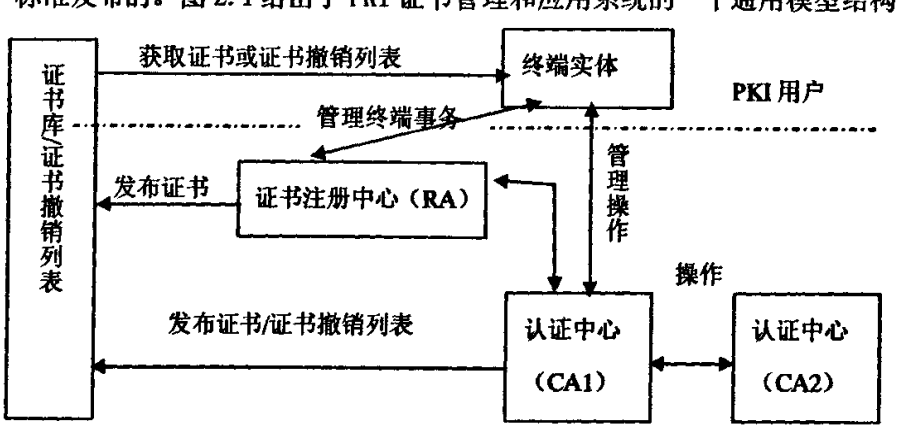


图 2.1 公钥基础设施 PKI 管理模型

Fig. 2.1 PKI management model

2.2 基于 X.509 标准的公钥证书

2.2.1 证书定义

证书的概念[4]是由 MIT 的 Kohnfelde 在他 1978 年的学士论文“发展一种实用的公钥密码系统”中第一次引入的。本文所说的证书为公钥证书，即是将证书持有者的身份信息和其所拥有的公钥进行绑定的文件。证书文件还包含签发该证书的权威机构认证中心 CA 对该证书的签名，通过签名保障了证书的合法性和有效性。证书包含的持有者公钥和相关信息的真实性和完整性也是通过 CA 的签名来保障的，这使得证书发布依赖于对证书本身的信任，也就是说证书提供了基本的信任机制。证书和相关的私钥可以提供诸如身份认证、完整性、机密性和不可否认性等安全服务。证书中的公钥可用于加密数据或者验证对应私钥的签名。

证书是随 PKI 的形成而新发展起来的安全机制。目前定义和使用的证书有很大的不同，例如 X.509 证书，WTLS 证书(WAP)和 PGP 证书等。但是大多数使用的证书是 X.509 v3 公钥证书。属性证书在 X.509 v3 中就已经提出了，但是直到 X.509 v4 才具有实用价值。本文中，数字证书、电子证书、证书等都是指基于 X.509 标准的公钥证书。

2.2.2 证书结构[40]

X.509 v3 证书采用 ASN.1 语法进行编码[43]，为签名计算，将证书按照 ASN.1 语法(DER) [X.208]规则进行编码，在实现中对于序列(SEQU－ENCE)是采用不定长编码格式[1][2][5]。X.509 v3 证书包含 3 个字段串的组合，这 3 个字段是 tbsCertificate,signatureAlgorithm, signature，其基本语法如下:

```
Certificate::=SEQUENCE
{
    tbsCertificate          TBSCertificate,
    signatureAlgorithm      AlgorithmIdentifier,
    signature               BIT STRING
}

TBSCertificate::=sequence
```

```
{
    Version[0]                  Version DEFAULT v1(0),
    serialNumber                CertificateSerialNumer,
    signature                   AlgorithmIdentifier,
    issuer                      Name,
    validity                    Validity,
    subject                     Name,
subjectPublicKeyInfo            SubjectPublicKeyInfo,
issuerUniqueID      [1]         IMPLICIT UniqueIdentifier OPTIONAL,
subjectUniqueID     [2]         IMPLICIT UniqueIdentifier OPTIONAL,
extensions          [3]         Extensions OPTIONAL
}
Version::=INTEGER {v1 (0), v2(1),v3 (2) }
CertificateSerialNumber::=INTEGER
Validity::=SEQUENCE
{
    notBefore           Time,
    notAfter            Time
}
Time::=CHOICE
{
    utcTime             UTCTime,
    generalTime         GeneralizedTime
}
UniqueIdentifier::=BIT STRING
SubjectPublicKeyInfo::=SEQUENCE
{
    algorithm           AlgorithmIdentifier,
    subjectPublicKey    BIT STRING
}
```

```
Extensions::=SEQUENCE SIZE(1.MAX) OF Extension
Extension::=SEQUENCE
{
        extnID                  OBJECT IDENTIFIER,
        critical                BOOLEAN DEFAULT FALSE,
        extnValue               OCTET STRING
}
```

X.509 定义了证书标准字段和扩展字段具体要求，在实践中仍然有很多证书在颁发时需要一个专门的协议子集来进一步定义说明。PKIX 工作组就定义了这样一个子集，即 RFC2459[19]。虽然 RFC2459 是专门为 Internet 的应用环境而制定的，但它里面的很多建议都可以应用于企业环境。下面给出 tbsCertificate 字段(子集)的说明。

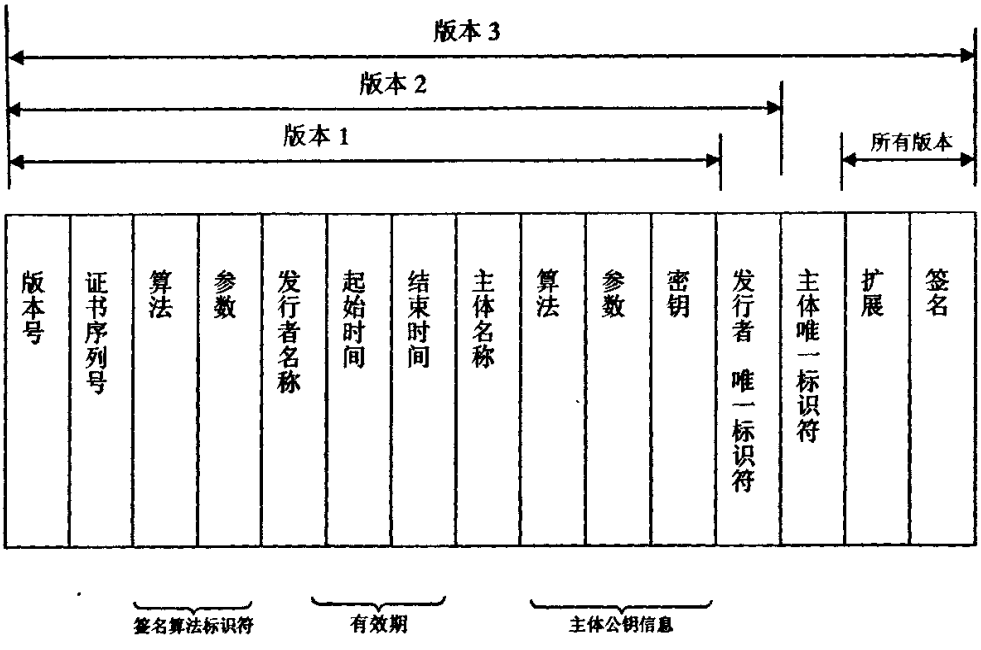


图 2.2 X.509 证书结构

Fig. 2.2 X.509 certificate structure

下面对图 2.2 中的各项字段做出说明:

(1)版本号:描绘编码证书的版本((V1, V2 或 V3);

(2)证书序列号:由证书颁发者(CA)分配的每一个证书的唯一标识符，一般是一个整数;

(3)签名:签名字段算法标识符，是由 CA 在证书上所用签名的算法，由算法和参数组成。例如，SHA-1 和 RSA 的对象标识符就用来说明该数字签名是利用 RSA 对 SHA-1 杂凑加密.

(4)发行者:用来标识在证书上签名和发行的实体，该字段含有一个非空的可识别名(DN)，这是必须说明的;

(5)有效期:证书有效期是时间段。本字段由“notValidBefore”和"notValidAfter”两项组成，它们分别由 UTC 时间或一般时间表示;

(6)主体(主题):证书拥有者的可识别名。此字段必须是非空的，除非使用了其它的命名形式;

(7)主体公钥信息:主体的公钥以及算法标识符，这是必须说明的;

(8)发行者唯一标识符:证书颁发者的唯一标识符，属于可选项，在实际应用中很少使用;

(9)主体唯一标识符:证书拥有者的唯一标识符，属于可选项，在实际应用中很少使用;

(10)扩展项:可选的标准和专用扩展，包括，授权密钥标识符、主体密钥标识符、密钥使用、证书策略、CRL 分布点、主体目录属性等等。

2.3 证书签发

证书的发放分为两种方式::一是离线方式发放，即面对面发放，特别是企业高级证书，最好是面对面的离线方式发放。二是在线方式发放，即通过 Internet 使用 LDAP，在 i500 目录服务器上下载证书。

1.离线方式发放离线方式发放的步骤如下:

(1)一个企业级用户证书的申请被批准注册以后，审核授权部门(RegistryAuthority, RA)端的应用程序初始化申请者信息，在 LDAP 目录服务器中添加企业证书申请人的有关信息。

(2) RA 将申请者信息初始化后传给 CA, CA 为申请者产生一个参照号和一个认证码。参照号 Ref.number 及认证码 Auth.code 在 PKI 中有时也称做 user ID 及 Password。参照号是一次性密钥。RA 将 Ref.number 和 Auth.code。使用加密电子邮件或打印在保密信封中，通过可靠途径传递给企业高级证书的申请人。企业高

级证书的申请人输入参照号及认证码，从 RA 哪儿面对面领取证书。证书介质可以存入软盘或者存放于 IC 卡中。

2.在线方式发放

在线方式发放证书的步骤如下:

(1)个人证书申请者将个人信息写入 CA 的申请人信息数据库中，RA 端即可接收到从 CA 中心发放的 Ref.number 和 Auth. code，并将在屏幕上显示的参照号和授权码打印出来，当面提交给证书申请人;

(2)证书申请人回到自己的微机上，登录到网站，通过浏览器安装 Root CA 证书;

(3)申请人在网页上按提示填入参照号和授权码，自助式地下载自己的证书。

2.4 PKI 中的证书撤销

公钥证书由 CA 签发，绑定了用户实体和公钥信息，是对实体身份的一种证明。证书可能由于一些特殊的原因需要停止使用，即证书吊销，导致证书吊销的原因有以下几种情况：①证书都有一定的生命周期，过期的证书必须作废;②在用户私钥泄露、工作调动等情况下，即使证书仍处于有效期，也必须提前停止使用。③颁发证书的 CA 遭受攻击或泄密，致使密钥无法使用。④证书拥有者的身份相关的信息发生变化。⑤用户提前申请停止使用。⑥证书被新证书取代、证书被冻结或挂起等等。

为保证交易的安全性，在使用证书之前，需要对证书进行一系列检查，称为证书验证。检查项目主要包括以下五个方面：

(1)签发证书的 CA 是否可信，或者 CA 的数字签名被验证是正确的。包括对证书路径上所有证书的验证。

(2)证书是否具有完整性，即能否通过证书的数字签名验证。

(3)证书是否在有效期内，由证书里的 notValidBefore 和 notValidAfter 两个参数限定有效期。

(4)证书的使用方式与预期的使用目的或声明的策略是否一致。

(5)证书当前是否被撤销，由证书使用者向颁发该证书的 CA 查询。

在具有多级层次结构的 CA 中，验证方需反复获取路径上的所有证书，从以上五方面验证，直到颁发证书的 CA 是可信的。在证书路径中，如果有一个证书验证失败，就终止交易。在 CA 层次和数量较多时，证书当前状态检查，即证书是否被

撤销可能成为证书有效性验证的瓶颈。

引入撤销机制以后，证书的状态分为两种:正常和撤销。正常状态的证书只代表证书没有被撤销，还需使用者进一步验证证书的有效性。如不作特殊说明，本文中的撤销证书都是指未到期而被撤销的证书。

证书撤销申请可以由终端用户发起，直接提交给 RA 或 CA 中心。RA 或 CA 要对用户的申请进行审核。在某些环境下，CA 也有权力直接撤销终端实体的证书。

图 2.3 是证书撤销申请的简单示意图。

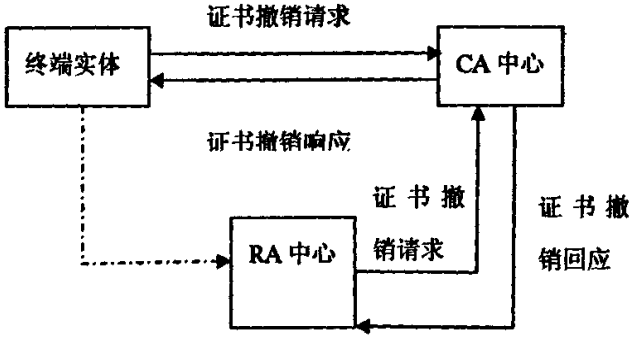


图 2.3 证书撤销申请示意图

Fig. 2.3 Certificate revocation

证书状态由颁发该证书的 CA 管理，为保护 CA 安全，CA 不直接向验证者提供证书状态信息，离线签署证书状态信息后，发布到证书撤销库并定期更新，由证书撤销库响应验证者的查询请求。整个过程如图书 2. 4 所示。

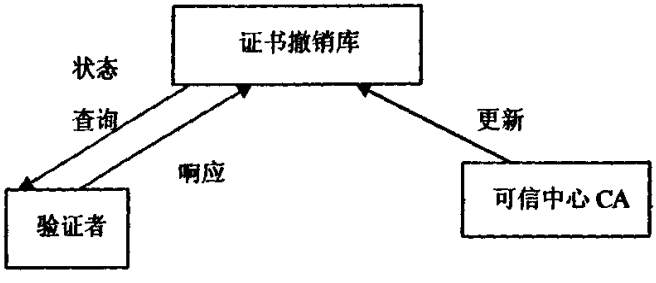


图 2.4 证书状态查询示意图

Pig. 2.4 Certificate status query

证书状态的验证过程涉及三方面实体:

CA 中心:负责所有证书的状态信息，及时更新被撤销证书的状态。为确保私钥安全，CA 必须和外部的网络断开，CA 对发布的证书状态信息签名，确保信息传输

的完整性和不可否认性。

证书撤销库:根据 CA 提供的证书状态信息，应答验证者对证书状态的查询。在这里证书撤销库可以是目录服务器，也可以是响应器或终端用户所保存的证书撤销库，需要依据不同的模型而论。

验证者或称为验证方:使用证书之前，向证书撤销库提交对证书状态的查询请求。在这里验证方一般指点终端用户。

第三章　常见证书撤销机制分析

3.1 证书撤销模型

证书撤销可以抽象为[26-27]：假设 S 代表固定集合，由可信实体定义 S，非可信实体为 P,P 是代表 CA 为 V 提供 S 的查询。验证者为 V，其中 S 对 P 已知而对 V 未知，由 V 向 P 查询集合 S 中的元素是否存在元素 x。目的是在 P 和 V 之间找到协议 P:对于给定的 x，P 可以证明 $x\in S$ 或 $x\notin S$,非可信实体 P 可以动态更新 S，但是在 P 和 V 交互时，假设 S 是固定的，交互过程如图 3.1 所示：

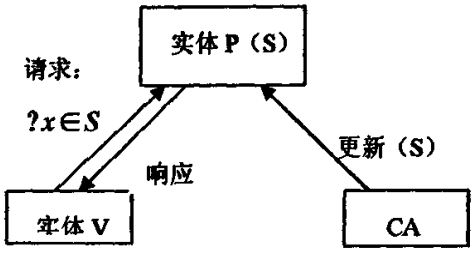


图 3.1 证书撤销模型

Fig. 3.1 Certificate revocation model

假设 U 为全集，其中 S 属于 U，Ds 代表集合 S 的数据结构，V 对 P 的操作包含以下几种：

(1)成员查询

对元素<e>的查询，应答的结果为字符串<a>，其中 a∈{YES,NO}，对应于 e∈S 和 e∉S。

(2)认证成员查询

对元素<e>的查询，应答为字符串<a, P>，a∈{YES, NO}， P 为由 CA 提供的对应答 a 的证明。

(3)更新操作包括两种

① <Insert, e>，其中 e∈U 但 e∉S。Ds 代表新的 S'=S U {e}。

② <Remove,e>，其中 e∈S，产生新的 Ds，代表新的 S'=S-{e}。

定义 1:字典(Dictionary)是集合 S 的数据结构 Ds,规定了对集合 S 中元素的查询和更新操作。

定义 2:认证字典(Authenticated Dictionary)是集合 S' 的数据结构 Ds'，规定了对集合 S' 中元素的认证查询和更新操作。

在证书撤销模型中，可信实体CA定义注销证书集合S，对于不可信实体P证书撤销库S已知，对验证实体V未知，由实体CA为P提供集合S中元素的证明P。

集合S是动态更新的，需采用一种机制保证证明P是最新的。假定：①CA定期更新证书撤销库②验证实体知道最新状态信息是何时发布，并能证明 P 是否最新。

3.2 基于CRL的证书撤销机制

3.2.1 证书撤销列表CRL

证书撤销列表(Certificate Revocation List，简称CRL)是较早的周期性发布撤销证书的一种机制[40]，也是x.509的标准证书吊销方案[51]。CRL由CA周期性发布处于有效期内被提前撤销的证书，是目前 PKI 中应用最广的证书注销方式，属于离线、黑名单方式。CRL经CA签名，只包括被撤销证书的证书序列号，撤销日期和撤销原因等。当已经撤销的证书有效期终止时，就从CRL中去掉。CRL还指明了发布时间及下次更新时间。第一版是1988年由ITU引入，第二版在1997年定义。现在CRL格式遵循X.509定义的v2版，如图3.2所示：

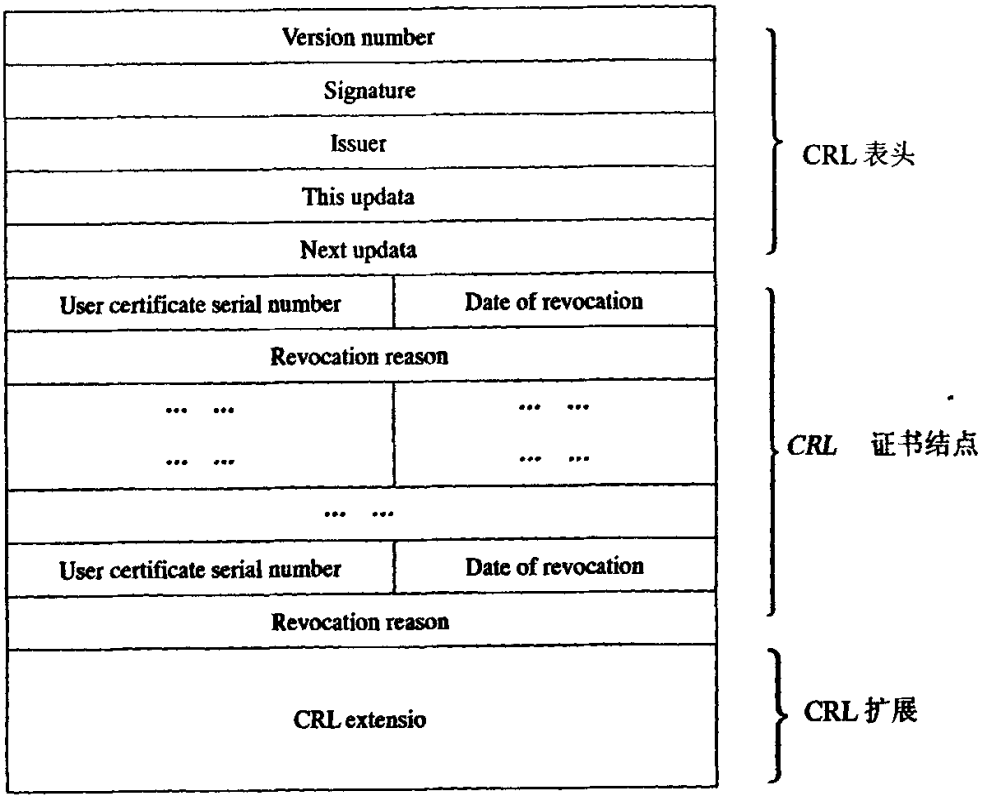


图 3.2 证书撤销列表 CRL v2 版格式

Fig. 3.2 Certificate revocation list CRL v2

其中:

(1)Version number: CRL 的版本号。

(2)Signature:签名算法标识。标识对证书撤销列表进行签名的散列函数和签名算法的类型。

(3)Issuer:签名撤销列表机构的可识别名称。

(4)This update:定义本次撤销列表发布的日期。

(5)Next update:定义颁发下一个撤销列表的最新日期。

(6)撤销的证书列表，每个被撤销证书都包含三个字段:

User certificate serial number:标识被撤销证书的证书序列号。

Date of revocation:认证机构撤销证书的具体日期(可选);

Revocation reason:证书被撤销的原因(可选):

(7)CRL extensions:可选的扩展项，为撤销的证书提供附加信息。

CRL 采用 ASN.1 抽象语法描述，其数据结构如下:

```
CrtificateList::=SEQUENCE
 {
    tbsCertList             TBSCertList,
    signatureAlgorithm        AlgorithmIdentifier,
    signatureValue          BIT STRING
 }
TBSCertList::=SEQUENCE
{
    version                 Version   OPTIONAL,
                                  --if present, MUST be v2
    signature               AlgorithtnIdentifier,
    issuer                  Name,
    thisUpdate              Time,
    nextUpdate              Time      OPTIONAL,
    revokedCertificates     SEQUENCE OF SEQUENCE {
```

```
        userCertificate         CertificateSerialNumber}
        revocationDate          Time,
        crlEntryExtensions      Extensions OPTIONAL
                                        --if present, MUST be v2
        }OPTIONAL,
        crlExtensions[0]        EXPLICIT Extensions OPTIONAL
                                        --if present, MUST be v2
}
```

CRL 的使用流程为:

(1) CA 将撤销的证书生成 CRL，发布到证书撤销库中;

(2)验证实体从证书扩展项 CRLDistributionPoints 中获取证书撤销库地址，下载 CRL;

(3)验证 CA 签名，查找该 CRL 是否包含所查询证书的证书序列号，如果可以找到，说明该证书已经注销。

CA 对 CRL 采用周期性发布策略。在 CRL 更新周期内，验证实体只需下载一次，缓存在本地，供再次查询证书使用。证书撤销库由不可信第三方目录服务器负责为用户提供查询服务，负责发布 CRL，处理所有实体验证请求。因此，CA 须对 CRL 签名，防止证书撤销库篡改或伪造 CRL。

采用 CRL 最大的优点是实现简单、操作方便,广泛应用在工程实践中。PEM (The Privacy Enhanced Email，简称 PEM)体系模型、Penman 和 Kanfinan,窗口证书撤销就使用了 CRL 管理撤销的证书。在小规模 PKI 中,CRL 能有效提供证书状态信息。在大规模 PKI 中，CA 颁发的证书达到几万、几十万，甚至更多，证书撤销频率难以预料，CRL 变得十分庞大，CRL 就暴露出以下几个主要的缺陷:

(1)撤销信息及时性差。CRL 发布周期通常为一周、一天甚至几个小时，两次证书撤销信息发布之间的间隔被称为撤销延迟。有的应用可以接受较长时间的间隔，而有的则是几分钟的间隔都是不能接受的。从用户申请注销到 CA 发布撤销信息，总有一定的延迟，查询实体得到的是不确定的证书状态。缩短 CRL 的颁发周期可减少时延给验证方带来的风险，但频繁发布庞大的 CRL 给 CA 和验证实体都带来了沉重的负担。

(2)随着注销的证书越来越多，CRL 的尺寸会越来越大。证书有效期终止时，

才能从CRL中移出。在CA层次较多时，需要下载证书链中每个证书的CRL加以验证，大大增加了验证方下载CRL和验证签名的时间。

(3)系统性能较差。所有验证实体下载的CRL会同时到期，当CRL更新发布时，总会引起验证实体新一轮下载，导致高峰请求，增加了处理器负荷，需要较高的峰值带宽，甚至引起网络拥塞。CRL的这些缺陷使它无法满足大规模PKI中及时、有效发布证书状态信息的要求，人们对是否继续使用CRL也经过了激烈讨论。但CRL仍是目前应用最广泛的证书撤销机制。在X.509中，引入了几种以CRL为基础的发布机制，增量CRL、分段CRL、重复发布CRL,分别从不同的方面作了改进。

3.2.2 传统CRL证书撤销机制[48]

证书撤销列表(CRL)是由CA签发的包括已经撤销的未过期证书的证书序列号及撤销日期和撤销原因的数据结构。每个CRL中都包括一个“下一次更新”域((NextUpdate Field)用来指明下一次发布CRL的时间。任何请求证书状态信息的信任实体(实体没有当前的CRL)都可以自由地从储存库获得当前CRL。CRL思想简单、易于实现，是目前应用最广泛的证书撤销方式。在小规模PKI环境中，能有效地为验证方提供证书状态信息。随着PKI规模的扩大，CRL尺寸庞大，发布信息及时性较差，用户规模增加时难以扩展。基于CRL，增量CRL、分段CRL、重复发布CRL分别做了改进。但是，没有一种机制能够满足所有的应用，不同的方法都有一定的适用环境。

为了提高性能，可以把CRL的副本分布到几个站点，这里假设所有的信任实体都从同一储存库中获得CRL 。

传统方法中，每个信任实体为了执行一次确认必须获得最新的CRL。一旦一个信任实体已经有了最新的CRL，在新的CRL发布之前，它就不必再向目录服务器发出请求。因而，在CRL的有效期内，一个信任实体最多只需向储存库发送一个请求。这个请求会在当前CRL发布后信任实体需要执行第一次确认时发出。

3.2.3 分段CRL证书撤销机制[28]

分段CRL也叫CRL分布点(CRL Distribution Point)，每个CRL分布点仅包含一组注销证书，可减小CRL的大小。随着注销证书的增多，证书存在于CRL的时间较长，CRL可能十分庞大。将CRL分段，可减小CRL文件的大小。创建分段的标准可以按照证书用户的地域、证书的重要性、使用范围、注销原因等，将CRL

分成不同的段，用户根据需要，下载相应的 CRL 段，而无需下载全部 CRL 。每段 CRL 的存放地点标识在证书扩展字段中，证书颁发之后，CRL 段的存放地址就已经固定，在证书整个生命周期中不能再改变。为了改变这种不利于扩展的局限性，Adams 提出了动态分段的概念，也称为重定向 CRL，在证书扩展项中不是指明 CRL 段的存放地点，而是指向 CRL 段地址的指针。重定向 CRL 不包括撤销证书的任何信息，而是通过范围声明，指向某个段 CRL 。

分段 CRL 减小了 CRL 长度，具有较好的可扩展性，可发布在不同的证书撤销库中，减少响应器(或称目录、目录服务器)的负载，但分段 CRL 不会降低峰值请求，而且当验证方需查询多个段时，反而增加了验证方的平均请求率和等待时间。

3.2.4 重复发布 CRL 证书撤销机制

重复颁布 CRL (Over-Issued CRL)[2]可降低峰值 CRL 值通常是在上一个 CRL 过期时发布，在更新时刻，验证方的请求率最高，称为峰值请求。随后验证方将 CRL 缓存在本地，请求率逐渐降低。在峰值请求时，可能会引起网络拥塞、服务器过载、响应延迟等问题。

提前更新 CRL，将 CRL 交错发布，即在旧的 CRL 还未到期时，发布新的 CRL，使验证方缓存中的 CRL 在不同时间到期，实际上是将验证方分组，每组在不同的 CRL 到期时间下载自己相应的更新 CRL,，这样可以降低峰值请求率。以下表格是一个重复发布 CRL 的例子:

CRL 序列号	本次更新时间	下次更新时间
1	0:00	次日 0:00
2	6:00	次日 6:00
3	12:00	次日 12:00
4	18:00	次日 18:00

表 3.1 重复发布 CRL 发布示例

Tab. 3.1 Over-Issued CRL example

表中交错发布的 CRL 有 4 个序列号，实际上是将验证方分成四组，验证方下

载 CRL 缓存在本地，当每组的 CRL 到期时，验证方就分别在各自的到期时间进行相应的 CRL 下载，这样就降低了峰值请求率。

但是如果验证用户知道 CRL 的发布机制之后，同样也会产生竞争下载的问题。由于 Over-Issued CRL 将 CRL 交错发布，CRL 存在不同的到期时间，因此也就存在不同的 CRL 版本。而作为验证方的用户为减少交易中不必要的损失，就很有可能竞相下载最新的 CRL，以获得最新的证书认证信息，实际上，这时的 Over-Issued CRL 也就等同于传统的 CRL，只不过是缩短了 CRL 的发布周期。

3.2.5 增量 CRL 证书撤销机制

增量 CRL 即 Delta-CRL[29]，是 X.509 第三版引入的，用于解决 CRL 时延问题。增量 CRL 包含两种类型的 CRL：基准 CRL(Base-CRL)和增量 CRL(Delta-CRL) 。基准 CRL 发布所有注销的证书信息，Delta-CRL 在每两次基准 CRL 更新期间，以较小的时间间隔颁发自上一次基准 CRL 颁发以来新增加的注销证书，发布周期比基准 CRL 的发布周期小。Delta-CRL 含有在基准 CRL 和当前 CRL 之间改变的信息(和 Delta-CRL 一道发出)。它由 CA 决定是否提供 Delta-CRL。当发出一个 Delta-CRL 的时候，CA 必须同时发行一个完整的 CRL。若没有签发完整 CRL 时，就不能签发 Delta-CRL。Delta-CRL 和相应的完整 CRL 证书注销列表序列号 CRLNumber 的值必须是完全相同的。

比如基准 CRL 的颁发周期为一天，假设其有效期为 00: 00～24:00，从 00:00 开始每隔 0.5 小时发布一次增量 CRL ， CA 的发布时刻分别如下:

0:00: Base-CRL
0:30: Delta-CRL_1
1:00: Delta-CRL_2
... ...
23:30: Delta-CRLZa

则验证方将在 0:00 下载 Base-CRL,在一天中其它时刻(每隔 0.5 小时)将下载在此基础上发布的 Delta-CRL_i，在下一天的更新周期中，CA 将发布新的 Base-CRL 和在此新 Base-CRL 基础上的新 Delta-CRL，供用户下载。示例如表 3.2:

Base-CRL 序列号	Delta-CRL 序列号	Delta-CRL
1	1	This update=0:00, Next update=0:30 BaseCrlNumber=1
	2	This update=0:30, Nexi update-1:00 BaseCrlNumber-1
	3	This update=I:00, Next update=1:30 BaseCrlNumbet=1
	.....	.....
	48	This update}23:30, Next update=0:00 BaseCrlNumbet=1
2	1	This update=0:30, Next update=1:00 BaseCrlNumber=2
	2	This update=1:00, Next update=1:30 BaseCrlNumber=2
	.....	.....
.....	.....	.....

表 3.2 增量 CRL 发布示例

Tab. 3.2 Delta-CRL example

Delta-CRLNumber 是发布的增量 CRL 的序列号，Delta-CRL 格式与基准 CRL 类似，定义了本次发布时间及下次更新时间，BaseCRLNumber 为 Delta-CRL 依据的基准 CRL 的发布序列号，BaseCRLNumber 的值所定义的 CRL 号，在产生 Delta-CRL 时作为起始点。

Delta-CRL 提高了 CA 发布注销证书的频率，可以提供较新的证书状态信息；验证方已经获得了一个有效的基准 CRL 之后，只需下载最新的增量 CRL 即可，减小了 CRL 下载尺寸。但每一次 Delta-CRL 的发布也会出现类似 CRL 的峰值请求高峰;而且当验证方请求完一次基准 CRL 后，还需请求增量 CRL，增加了平均请求率。

另外，一方面，如果证书的撤销频率很高，那么 Delta-CRL 同样也会比较大，为减少 Delta-CRL 的大小就不得不通过缩短 Delta-CRL 的发布周期来减少 Delta-CRL 的大小;另一方面，在 Base-CRI}发布周期中，仍然需要传送到发布时

为止包含CA所有撤销证书信息在内的整个Base-CRL，网络通信量依然很大。

3.3 CRL证书撤销评价

3.3.1 评价标准

对于如何评价证书撤销机制，人们进行了许多研究和探讨。一般而言，一个好的证书撤销机制通常要满足以下几个方面的要求：

1.安全性

安全性是PKI对证书撤销机制的最基本的要求，验证方得到的证书状态信息必须由可信CA发布或经过CA签名，不能由其它实体伪造证书状态信息，比如:有效证书被伪造为已经注销状态;已经注销的证书被伪造为正常状态。证书撤销机制设计要确保准确的安全的证书状态信息发布，验证实体通过验证CA签名以确保信息可靠。

2.计算代价

证书验证过程中，CA中心、证书撤销库、客户端会执行一定的计算，包括创建证书状态信息、产生应答信息、验证应答信息签名等。计算需简单快速，避免给各方带来过重负担。加密、签名消耗CPU最多，可能会成为计算的瓶颈，应尽量使得这些操作在多项式时间完成。

3.通信代价

通信代价主要指CA和证书撤销库之间发送证书状态更新信息，证书撤销库和验证方之间要发送查询和应答信息。应尽可能减少双方的通信量，以满足网络带宽较紧张的用户。

4.延迟

不同应用对证书状态信息及时性的要求不同，撤销机制应该能够满足这些需求，另外，验证方获取证书的状态信息时有一定的时间限制，不能无限制的等待，尤其是在线交易业务，时延要求较高。

5.兼容性

证书撤销机制能够与现有证书标准和密钥管理协议有良好的兼容性，尤其是与PKI基础设施中已经广泛使用的证书撤销列表、在线证书状态协议兼容，在异构环境中，具有良好的互操作性。

3.3.2 评价模型

David A. Cooper 提出了一个数学模型分析证书撤销体系[27]，主要针对验证方请求证书验证高峰时的请求率。在建立模型之前，假定：

(1)验证方需要验证证书状态信息时，才查询证书撤销库，即初始时没有缓存 CRL。

(2)验证方有较好的缓存机制，一旦从证书撤销库中获得 CRL，就将其缓存在本地直到该 CRL 过期。在本周期内，再次验证证书时，只需查询该 CRL，不必再从证书撤销库获取。

(3)证书撤销库的缓存很大，不会丢弃任何终端实体的请求验证，即任何实体都会得到响应。

假设 CRL 发布起始时间为 0，从 0 时刻起，验证方不断请求 CRL 直到下一次新的 CRL 发布，实体向证书撤销库请求验证的行为可用排队模型分析，如图 3.3

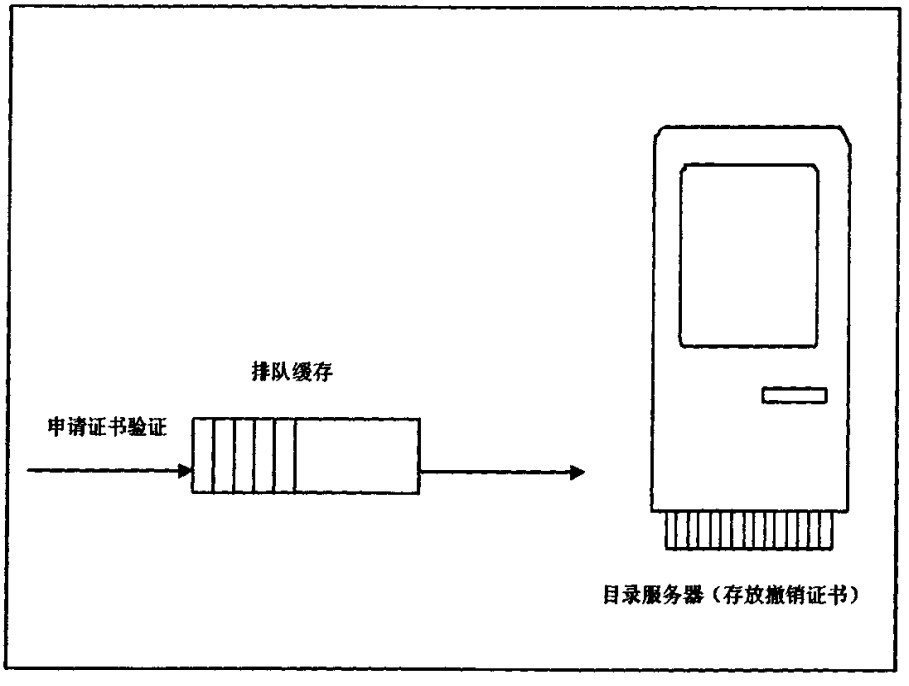


图 3.3 用户排队缓存

Fig. 3.3 User queue model

其中参数见表 3.3 所示。

参数	参数含义
N	请求验证的终端实体数目
v	每个终端实体的平均请求率
L_c	证书有效期
P	证书被提前撤销的概率
B	带宽
O	重复发布的同时有效的 CRL
CRL_H	CRL 头部长度
CRL_e	撤销证书的证书序列号、撤销日期、撤销时间占 9byte
T	CRL 更新周期
c	证书撤销珍表中包含的撤销证书个数
S	CRL 签名所占字节数
segment	CRI 分段数
w	增量 CRL 发布时间间隔
T	CRL 更新周期

表 3.3 证书撤销参数表

Tab. 3.3 Certificate revocation parameters

如果请求验证的验证方 N 的数目很大，每个验证请求是随机的、相互独立的，实体请求率服从指数分布规律即 ve^{-vt}，总请求率为

$$R(t)=Nve^{-vt} \qquad \text{公式 3.1}$$

从公式 3.1 可以看出，在初始时刻 t=0，即 CRL 颁发时刻，每个验证方都想请求得到最新的证书状态信息，请求率最高时，称为峰值请求。此后，验证方在本地缓存下载到的 CRL，在每个周期内不必再请求 CRL，请求率呈指数下降。峰值请求时刻，证书撤销库所需的带宽最多，称为峰值带宽。系统性能指峰值请求时，处理器计算负载;查询代价指验证方所需下载的 CRL 大小。风险指从证书撤销申请到 CA 发布证书撤销状态的时间间隔，间隔越大，验证方承担的风险越高。

影响这些参数的因素主要包括两方面:

(1)通信代价

①CA 更新证书撤销库时所需发送的信息量，在 CRL 中，每次更新需发送全部的证书撤销信息。

②证书撤销库发送给验证方的信息量，以及证书撤销库响应多个用户查询时所需的总的带宽。在 CRL 中，验证方查询单个证书状态时，需要下载所有的撤销证书信息。

(2)计算代价

① CA 生成 CRL 的计算量，需要对撤销证书执行 hash 运算和签名操作；

②证书撤销库响应请求的计算量采用 CRL 方式，不需执行计算；

③验证方验证信息的计算量，需验证签名，从撤销列表中查找目标证书信息所需的计算量。

1、 CRL 的大小

CRL 大小和 CA 颁发的证书数目、证书生命周期、撤销比率成正比。每个 CRL 的大小可以由 CRL 的头部大小+注销的证书个数*每个注销证书所占的长度+签名长度=$CRL_{\#}+n\times CRL_{e}+S$。

2、峰值带宽(B)

峰值带宽 B=峰值请求率×平均 CRL 长度

峰值请求率基本公式 $R(t)=Nve^{-rt}$，在初始时刻 t=0 时，请求率达到最高值。

3.3.3 对几种 CRL 方案的评价

用以上的评价标准对完全 CRL、分段 CRL、基准 CRL、增量 CRL 的进行比较，结果如下：

(1)完全 CRL 的峰值请求率以及峰值带宽、验证实体下载 CRL 都是最大的。

(2)分段 CRL 不会降低峰值请求，但是分段 CRL 比完全 CRL 的尺寸要小，所以所需峰值带宽较小;平均网络负载也要小一些。

(3)基准 CRL 和增量 CRL 的峰值请求并没有降低，但每个增量 CRL 包含的注销证书数目并不多，因此峰值带宽也较小，但增量 CRL 以更小的时间间隔颁发，峰值请求的频率比较高。但它的风险相对较小。

(4)重复发布 CRL 显著降低了峰值请求，减轻了网络负载，但不会减少 CRL 大小。

(5)CRL 增加验证方下载时间和等待时间。采用增量 CRL 和分段 CRL 都可以降低下载量，减轻客户端缓存大量证书撤销信息的压力，但这两种方式都会提高实体的平均请求率。

(6)采用 Delta-CRL，以较短时间间隔发布，可满足对证书状态信息及时性要

求较高的应用，减轻由于延迟而造成风险，其它三种机制发布信息都有一定的延迟，但频繁发布 Delta-CRL 给 CA 和网络带来额外代价。

3.4 基于动态数据结构的证书撤销机制

3.4.1 CRS 证书撤销机制

证书撤销状态 CRS (Certificate Revocation Status 简称 CRS)[30]由 Micali 1995 年提出，是第一个在线提供证书撤销信息的机制。CA 定义证书的有效时间 n，一般取 n=365 天;证书状态发布频率 i，指每天发布一次。在公钥证书扩展项中加入了一对参数，100bit 的初始值 Y 和 N 。 CA 随机选取两个初始秘密值 Y0 和 N0，选用一个 hash 函数对其运算，

$Y = H_n(Y0)$， $N=H(N0)$

Y 进行 n 次 hash 运算，N 运算一次，将结果保存在证书中。

CA 更新证书撤销库的信息为带有时间戳和签名的所有证书列表；包括所有有效和已经被撤销的证书的证书序列号；在时间间隔 i 内新颁发的证书/每个证书 100bit 的证明信息 V，V 的计算方式为：对有效证书 V，由 $V=H_{n-i}(Y0)$ 计算;已经注销的证书由 V=N0 计算。发送信息还可以附加证书注销的时间和原因。

证书状态的验证时，首先获取经 CA 签名的 L 表，验证 L 的签名通过后，从表中查询目标证书，并计算 $H_i(V)$ 是否等于 Y，如果相等，证书有效;否则计算 H (V)，如果等于 N，证书已经注销。

CRS 发送经过 CA 签名的证书列表 CRL，保证了信息完整性；证书状态的相关证明数值较小，只有 100bit，验证方可在线验证签名值。证书的初始值 Y0 和 N0 只有 CA 知道，能保障信息的安全性。CRS 存在一些问题：CA 需每天更新每个证书的状态值，计算量很大；CA 和证书库通信时需要很高的传输资源(不管证书状态是否改变，都需传送该证书的状态信息)；证书颁发时在证书中加入参数，导致证书在生命周期中不能改变验证策略，系统缺乏灵活性；系统的安全性取决于初值 Y0 和 N0 的安全性及 HASH 函数的抗碰撞的能力。

HCRS 系统对 CRS 进行了改进，减少了证书状态更新时所需的传输资源，还是无法解决 CRS 存在的其它问题。

3.4.2 CRT 证书撤销机制

1998 年，美国著名 Valicert 公司的 Kocher 提出了采用证书撤销树

(Certificate Revocation Tree,简称CRT)发布证书状态信息[31]。CRT基于Merkle hash树的思想[32]，叶子节点存储撤销证书的基本信息，内部节点采用单向hash函数OWHF(One Way Hash Function)对孩子节点进行hash运算得到节点值，一直向上到根结点，然后由CA对根节点进行签名。应答包括目标证书的相关路径节点值，由验证方重新构造出根值验证。

CRT最初思想采用叶子节点存储有效证书范围，例如假设一个CA中心证书序列号为5，12，13，15，20，25的证书被注销，用L(5,12)表示证书序列号为5,12的证书被注销，而所有大于5且小于12的证书是正常的。当证书数目非常多时，这种表示十分繁琐，将叶子节点直接存储注销证书的序列号可简化存储结构。

假如CA吊销了两个证书,其证书序列号分别为x1，x2,且x1<x2,则,若Cax=CA且x1<x<x2，则证书x正常，而x1，x2则表明已经撤销。一个撤销证书的证书序列号和撤销信息如撤销原因、撤销日期等用数据结构L0，L1,……，Ln表示，将其按顺序存储在树的叶子节点。一个简单的CRT如图3.7所示。

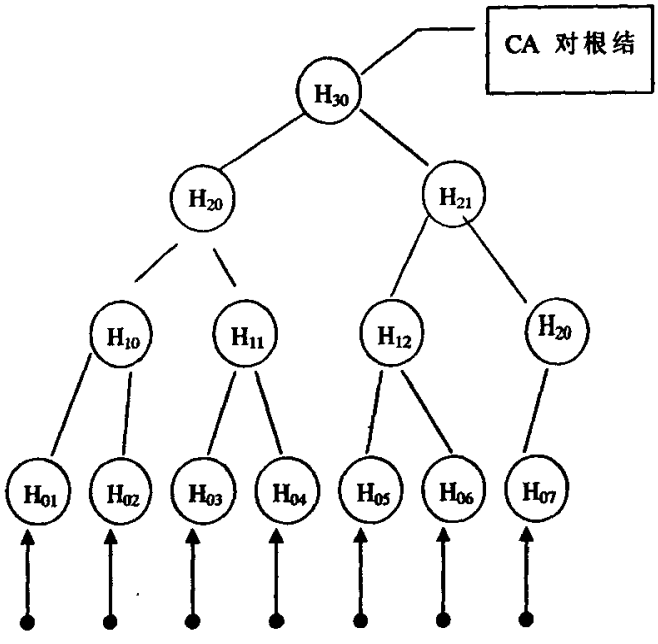


图3.7 证书撤销树示意图

Fig. 3.7 An example of CRT

最底层的节点表示撤销证书的撤销信息的数据结构L，采用单向hash函数如

SHA-1 或 MD5 进行 hash 运算产生杂凑值 H_{01}，H_{02}……，H_{07}。如图中箭头所示，同一级的相邻节点合为一个，继续 hash 运算，产生新的杂凑值作为上层节点。如果没有一对节点，剩余的节点就直接进入树的上一级如节点 H_{07}，H_{20}，反复计算直到唯一的根节点为止，即上图中的 H_{30} 节点。最后，由 CA 对根节点的 hash 值及撤销树的生成日期、下次更新日期签名。更新时，CA 将新注销证书的证书序列号及相关信息及 CA 新的签名值发送给证书撤销库，证书撤销库按照相同的方法重新生成 CRT，并与 CA 发送的根签名值比较，如果一致表明生成了正确的 CRT。

对证书状态的验证，只需检查该证书是否处于树的叶子节点，包括两种类型：

(1)已经撤销的证书

位于叶子节点的证书已经被撤销。证书撤销库需发送叶子节点及从根到叶子节点路径上相关节点值给验证方，由验证方重新构造出根节点 hash 值，与 CA 的根签名值比较是否一致。例如，查询证书序列号为 12 的证书时，证书撤销库需要发送的信息如图标记的数据 L(12)，H_{01}，H_{11}，H_{21}，H_{01} 是叶子节点 5 的 hash 值。使用这些值，验证方可方便地生成根节点，并和 CA 生成的节点值比较，如相等，验证通过。该证书的状态信息为已经注销，以及相关注销的时间、原因等。、

(2)有效的证书

两个叶子节点范围之间的证书是没有被撤销的，需发送包含比查询证书 x 大的最小叶子节点和比 x 小的最大叶子节点的相关值，分别验证根值。如查询的证书序列号为 7，需发送包含 7 的较大值和较小值的撤销证书节点，在图中为 5,12 的两个节点的相关信息，和 CRL 一样，不需在线签名:可用一种简洁方式表示大量证书撤销信息，在通信的角度上，CRT 大小在数量级上是 $\log_2 N$，其中 N 代表已撤销证书的数目。

CRT 的缺点：CA 和证书库计算量比较大，任何撤销证书增加或删除都需要重新构建整棵树。不可信的证书撤销库可能会欺骗验证方，对有效证书需要发送两个节点的相关信息量，是撤销证书的两倍。证书撤销树与 CRL 相比，应答信息较小，但是它主要的缺陷是：

(1)更新成本较高。CRT 采用了静态数据结构，叶子节点的撤销证书按证书序列号顺序排列，当新增加注销证书或者将已到期的证书从证书撤销树中移出时，需重新构建整棵树。证书数目较多时，计算量较大。

(2)响应有效证书通信量较高。证书撤销库发送未注销证书的证明信息时需发

送两个节点的相关信息，而已经撤销的证书只需发送一个节点的相关信息，对有效证书响应的通信量是注销证书的两倍。在 CA 中，注销的证书比例一般为 10%，终端实体查询的证书大部分是没有注销的，经常需要响应有效证书。验证方也需两次构建根 hash 值进行比较。

3.4.3 2-3 CRT

Naor Nissm 给出了 CRT 的变异方案 2-3CRT[28]，将 CRT 由二叉树改为 2-3 树，以解决 CRT 更新时整个树需要重新建立的问题。但 2-3 树的结构会给实际操作带来困难，其数据结构不如二叉树简捷，对有效证书的证明也不够简明。

3.4.4 基于二叉排序树的证书吊销方案

基于二叉排序树的证书吊销方案[33]中，它构建一个二叉排序树，树中的每一结点都包含了一个已撤销的证书的编号(这个编号是唯一的)和它的孩子结点的信息的 HASH 值，若此结点是叶子结点，则它孩子结点的信息值就为空。

基于二叉排序树的证书吊销方案继承地利用了 CRT 证明一个证书的状态时，不须全部二叉树的结点，只须相关的路径。整个树的结点是 CRT 的一半，树的规模得到有效控制，而且在更新结点时，不需要将整个二叉树进行重建，有效地降低了维护 CRT 的计算代价。

但是，基于二叉排序树的证书吊销方案也有一些不足，当二叉排序树插入结点时，插入的新结点都是叶结点，更新路径从树的最深层更新至根结点。当欲删除的结点有两个子树时，则使用文献[52]的标准算法删除的结点，须将欲删除结点的前驱结点来替换删除结点，这样必须更新两条变化结点到根结点的路径上的所有结点。其中一条路径是从欲删除的结点至根结点；另一条路径是从欲删除结点的中序遍历的前驱结点向上至根结点。由此，CA 对二叉排序证书吊销树进行更新时，若更新的结点稍多时，工作量很大，离根结点近的结点被多次重复修改，更新结点代价甚至可能超过建树的代价，此方案的数据维护量相当大。

3.4.5 在线证书状态协议

IETF 的 PKIX 工作组提出了在线证书状态协议 OCSP(Online Certificate Status Protocol，简称 OCSP)[34-36]。OCSP 使得应用程序可以测定所需要检验证书的(是否撤销)状态，这个协议描述了在应用程序检查证书状态和服务器提供状态之间所需要交换的数据。具体的实现方式通常为 Client/Server 结构:CA 设立一个

验证方信任的 ICSP 响应器，查询时，一个 OCSP 客户端按照协议定义的格式发送请求给 OCSP 响应器，包括所要查询证书的序列号，然后侦听当前证书状态直到响应器提供了一个响应:响应器查询到证书信息后，用自己的证书签名在线响应。但响应器如何获得证书状态信息协议并没有具体定义，可直接从 CA 的证书存储库获得信息，或采用 CRL 方式。

采用 OCSP 方式可以获得最及时的证书状态信息。提供证书最新信息程度，一般依靠响应器后端实现方式。OCSP 定义证书状态只有正常、否以及不知道三种病况。正常含义为:证书没有被注销、证书不在有效期内和证书并不存在，验证方还需采取另外的手段来判断证书的有效性。而且，OCSP 要求发布方是可信第三方;响应用户需要在线签名，容易泄露私钥，在大规模环境中有大量验证请求的情况下，严惩影响系统性能。且繁重的计算量可能成为系统的瓶颈。

3.4.6 简单证书验证协议

简单证书验证协议 SCVP (Simple Certificate Verification Protocol，简称 SCVP)由 IETF 提出的[36]，结合 OCSP 协议，可以在线为用户提供证书和证书路径上的证书链验证。SCVP 可能是 OCSP 的替代者，目前还处于草案阶段。

第四章 自根向下压缩的二叉排序证书吊销树方案

本章基于二叉排序证书吊销树的方案[33]，利用 HASH 函数自根向下压缩，整个树中结点的信息汇集至叶子结点，对树中的叶子结点重新利用其空链域在已有的树结构中建立一种新的拓朴结构。在新的拓朴结构中，有一个特殊的结点，它又集成了新的拓朴结构中的所有结点的信息，CA 对此结点进行签名。保证整个树中结点的信息的完整性与正确性。克服了 CRT 方案中树中结点发生变化后，树需要重新建立，树的维护代价太大，既能使用二叉排序树进行高效率的查询，又能有效地减少了名录服务器至查询者的通信代价问题。

4.1 引言

在第三章中对已有的各种证书吊销方案进行了详细的介绍和分析。我们的方案是在基于二叉排序树的证书吊销方案[33]的基础上提出的，以下便是对文献[33]的分析：

基于二叉排序树的证书吊销方案，继承地利用了 CRT 证明一个证书的状态时，不需全部二叉树，只须相关的路径。整个树的结点是 CRT 的一半，树的规模得到有效控制，而且在更新结点时，不需要将整个二叉树进行重建，有效地降低了维护 CRT 的代价。

但是，基于二叉排序树的证书吊销方案也有一些不足，当二叉排序树插入结点时，插入的新结点都是叶子结点，更新路径从树的最深层更新至根结点。当欲删除的结点有两个子树时，则使用文献[62]的标准算法删除结点，须用欲删除结点的前驱结点来替换删除结点，这样必须更新两条到根结点的路径上的结点。其中一条路径是从欲删除的结点至根结点；另一条路径是从欲删除结点的中序遍历的前驱结点向上至根结点。这样，CA 对二叉排序证书吊销树进行更新时，若更新的结点稍多时，工作量很大，离根结点近的结点被多次重复修改，更新结点代价甚至可能超过建树的代价。

本文所提三种方案使用安全的 HASH 函数从树根结点向叶子结点方向压缩，树中所有结点的信息汇聚到叶子结点，每个叶子结点包括从根结点至此结点路径上所有结点的关键字值的信息。然后，利用二叉树中叶子结点的空链域重新组织一种新的拓朴结构。这样，对树中结点的更新，勿需再向根结点方向更新路径中各

结点的值，从而对上层结点不需进行更新。只向下更新到叶子结点(由于大部分结点处于树的下层，它的路径比从叶子结点至根结点路径要短)。由于二叉排序树的结构没有变化，查询效率不受影响。返回查询结果也不需要整个二叉树，只是相关的路径上的结点，没有增加传递信息的数据量，而树维护工作量大大降低，通信成本降低。

4.2 自根向下的证书吊销树概述

文献[51]中证书吊销采用二叉排序树结构后，其优点相对于 CRT 减少了树的近一半的结点，有效证明路径减短。但在二叉排序树中插入的结点均是叶子结点；删除结点时，用它左子树中最右的结点（欲删除结点的前驱结点）来替代它，造成两条至根结点的路径需要更新，所需修改的结点过多，维护的工作量大。在该方案中，对二叉排序树线索化，线索化的目的是能够证明有效证明区间中的结点是未被吊销的，这一点在实际上是没有意义的。有效证书通过 CVT（证书验证树）来证明，不需要在证书吊销系统中进行证明。

本文针对于此，所提方案利用 HASH 函数自根向下压缩，则每一个叶子结点就包含从根结点至该叶子结点的所有结点的信息，再利用二叉排序树的叶子结点的空链域重新建成一个新的拓扑结构。新的拓朴结构中的结点数少于原证书吊销集中证书数目的一半，而新的拓朴结构建立在原来的二叉排序树上，因此不需要增添额外的结点。

根据新的拓朴结构类型，本文提出了三种方案。方案一，把叶子结点组成一个双向链表，CA 对链表最后的一个元素进行签名。方案二，应用文献[49][50]的算法，把叶子结点组成一个 CRT。方案三，把叶子结点建成一个用文献[54]中完全二叉树，再对此树的树根进行签名。所有方案并没有影响原来二叉排序树的结构，是在原来数据结构中构造一个新拓朴结构，这样叶子结点就是两种拓朴结构中双重结点，并没有增添新结点。

4.2.1 证书吊销问题的形式化定义

证书吊销问题的形式化定义引用文献[32]中的定义。

定义：设 u 是一个全集，s 是 u 的一个子集，$s \subset u$，设 D_s 是代表 s 的一个数据结构。

- 设〈e〉代表对 $e \in S$ 的成员查询，其回答为 $<a>, a \in \{YES, NO\}$，分别对

应于$e\in s$ 或$e\notin s$ 若其回答为$<a,p>$，其中a同上，且p是由CA 签名的一个关于a的证明，则称这样的查询为成员认证。

·更新运算是指如下两种运算：

(1)<Insert, e>，其中e∈u \ s，则插入e 后的数据结构为$D_{s'}$，这里s′ = s∪{e}；

(2)<Remove, e>，其中 e∈s，则删除后的数据结构为 Ds′，这里 s′ = s \{e}.

下面设u 是CA 发行的全部证书编号集，s是CA 吊销的证书编号集，我们用二叉排序树及其线索信息以及Hash 树来构造数据结构D_s，并讨论其认证查询以及插入、删除运算。

4.3 方案一：叶子结点组成双向链表结构

4.3.1 数据结构

设CA 吊销的证书的编号集为$S = \{n_0, n_1, \ldots, n_{n-1}\}$，为了使二叉树的左右保持基本的平衡，根结点$n_0$是一个被吊销证书集中的中间值。下面以证书编号为关键字，以n_0为根结点，对S 构造排序二叉树[51]，使每个结点的数据结构为：

Lchild	Parent	Cnum	Hashdata	Nstrhashdata	NodeTag	Rchild

表4.1 方案一中结点的数据结结构

Table 4.1 Data structure of Tree' s node in first theme

说明：

1)Lchild, Rchild：指向该结点的左孩子，右孩子的指针。

2)Parent：双亲结点的指针。

3)Cnum：该结点所代表的吊销证书的编号。

4)Hashdata：该结点与其双亲结点的证书编号的HASH 值（Hashdata =HASH(Cnum|| Parent->Hashdata））。

5) Nstrhashdata：利用叶结点的空链域建立的新拓朴结构的HASH值，这些HASH值用以验证树中结点所代表的证书被吊销的有效证明。

6) NodeTag：表明该结点的类型：当结点是非叶子结点时NodeTag=NODE；当结点是叶子结点时NodeTag=LEAF。

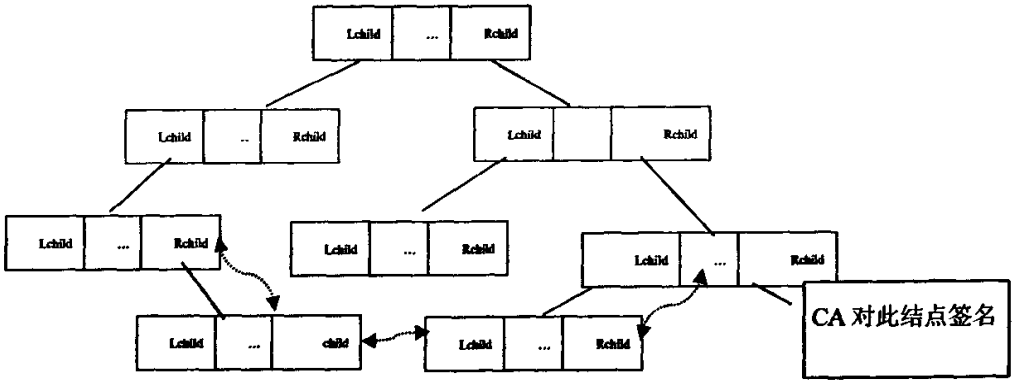
数据结构定义如下：

```
typedef structure CRTFBDHT
```

```
{
    structure CRTFBDHT *Lchild,*Parent,*Rchild;
    datatype Cnum;
    char *Hashdata,*Nstrhashdata;
}
```

4.3.2 具体方案

先将各结点的 Cnum 的内容写入欲吊销的证书编号，再按文献 [54] 中的标准算法建立二叉排序树。证书编号自根向下压缩时，除根结点的 Hashdata=Cnum，其余结点的 Hashdata=HASH(Cnum|| Parent->Hashdata)，每一个结点都包括上层结点的信息。再对叶结点从最左的叶结点向右压缩，并修改指针，建立为双向链表，使得每一结点的 Lchild 指向它的左面的最右的叶结点，而它的 Rchild 指向它右面最左的叶结点。最左的叶结点的 Nstrhashdata 等于 Hashdata，向右的每个叶结点的 Nstrhashdata=HASH (Lchild->Nstrhashdata ||Hashdata)。最后 CA 对该双向链表的最后的结点进行签名，整个树的结点的有效性就可以得到证明。图 4.1 是一个基于该方案的示意图：



注：——— 表示树的结构

<——> 表示使用二叉树叶结点的空链域建立的双向链表结构

图 4.1 二叉排序树的叶子结点组成双向链表

Fig 4.1Double linked list using leaf node in binary sort tree

4.3.3 吊销证书的查询与验证

当用户欲了解某一证书是否是被吊销时，名录服务器从根目录向下按照二叉

排序树的一般算法进行，这样会有三种情况：1）欲查询的证书不在证书吊销树中，查询结果返回 no。2)欲查询的证书为叶子结点时，把它的双亲结点和左边的结点，无左边结点的则用空值，再沿该结点的 Rchild 域向右至树中最右的叶子结点(CA 签名的结点)作为查询结果并返回。3)当查询的结点为非叶子结点时，从该结点的双亲结点开始，经由本结点按如下方法返回数据：当经过的每一个结点有右孩子时从右孩子向下，否则左孩子开始向下，重复直至叶子结点；将该叶子结点的双向链表的前驱结点(叶子结点的 Lchild 所指的结点)与它右边的所有的后继结点返回。

用户接收到名录服务器返回的结果后，若为空值，则表明此证书未被吊销。若返回结果非空，用 hash 函数进行数据核查。最后对 CA 的签名进行验证，只要验证通过就可以确定此证书已被吊销。

4.3.4 插入结点

当 CA 在二叉排序证书吊销树中插入结点时，由文献 [54] 可知新插入的结点均为叶子结点。若新插入的结点的双亲结点原来是叶子结点，则用新插入的结点在双向链表中替换其双亲结点，并修改结点中的 Hash 值。修改双亲结点的标志为 NODE. 若插入结点比双亲结点小，则使双亲结点的 Lchild 域指向新插入的结点，而 Rchild 的值为空。否则，使双亲结点的 Rchild 域指向新插入的结点，而 Lchild 的值为空。

若插入的新结点的双亲结点已有一个孩子。有两种情况：如果新增结点为其双亲结点的右孩子，则必须找到其双亲结点的左子树中最右的叶子结点，再将新增的结点作为它的后继结点加入到双向链表中。如果新增结点为其双亲结点的左孩子，则找到它的双亲结点的右子树中最左的叶子结点，使新增的结点作为它的前驱结点加入双向链表中。

由于插入的结点是叶子结点，则直接修改双向链表中插入结点及以后的结点的 Nstrhashdata 值，最后 CA 对双向链表中最右的结点签名。不同于文献 [33] 自底向上，需要更改许多结点。尤其当一次插入的结点很多时，其效率是很低的。但本方案中，当一次插入的结点很多时，可以先对二叉树进行插入结点的操作，再对双向链表进行一次性更新操作，其效率是非常高的。

插入结点的算法描述如下：

```
int InsertNode(Binarytree &T,Node *e)
```

```
//将结点 e 的插入 Binarytree
{
if(e->Cnum<T->Cnum)
{
 if(T.NodeTag==LEAF)
{
  swapnode (T,e);
  return SUCCESS;   //若 T 为叶子结点，交换 T 和 e 的位置,并修改指针。
}
else  if(T->Lchild)
      InsertNode(Binarytree &T->Lchild,Node e);
           //若 T 的左子树非空，采用递归算法将 e 插入 T 的左子树
else
{
    insertnode(T->Lchild,e);  //将 e 插入 T 的左子树中
    p=Searchleftleaf(T->Rchild);
//Searchleftleaf()返回 T 的右子树中最左的叶子结点
    InsertDoubleLinkedList(e,p,left);
//将 e 插入至双向链表 P 结点的左侧。
    return SUCCESS;
}     //T 有右子树，左子树为空，将新的结点插入左子树。
}
else if(e->Cnum>T->Cnum)
{
if(T.NodeTag==LEAF)
{
   swapnodeR(T,e);
   return SUCCESS;   //若 T 为叶子结点，交换 T 和 e 的位置,并修改指针。
}
else  if(T->Rchild&&T.NodeTag==NODE)
```

```
    InsertNode(Binarytree &T->Rchild,Node e);
     //若T的右子树非空，采用递归算法将e插入T的右子树
  else
  {
    insertnode(T->Rchild,e);  //将e插入T的右子树中
    p=Searchrightleaf(T->Lchild);
  //Searchleftleaf()返回T的左子树中最右的叶子结点
    InsertDoubleLinkedList(e,p,right);
   //将e插入至双向链表P结点的右侧。
  }
  }
  else
    return FAIL;  //欲吊销的证书在二叉证书吊销排序树已存在
  }
```

4.3.5 删除结点

删除的结点有两种情况：

(1)删除的结点为叶子结点，若它的双亲结点只有一个孩子，则修改双亲结点，用双亲结点来代替此结点在双向链表中位置，若它的双亲结点有两个孩子，删除此结点，并修改它的前趋及后继结点的指针，在双向链表中也删除此结点。

(2)删除的结点的若为非叶子结点，如果此结点有一个孩子，则直接按照文献[54]中的标准算法去删除，用它的孩子替代欲删除的结点。如果此结点有两个孩子，先按文献[54]中的算法，将此结点的左子树中最右边的叶子结点的值复制给欲删除的结点，问题就转化为删除一个叶子结点用1)的方法对此叶结点进行删除。

删除结点时，必须从变化的结点向下直至叶子结点的值，修改数据hashdata项的值。最后，再对整个双向链表中各结点的Nstrhashdata进行更新并对最后一个结点签名。但如果一次进行批量删除操作，可以将这些需要更新的结点存储在一个数组中。当一新删除的结点需要插入数组中时，先将它与数组中已保存的元素做以比较。如果出现有的元素在欲删除结点的子树中，就把删除的结点用来替换发现的这个数组元素。比较的方法是，找到删除的最左的结点和最右的结点，

判断数组元素的关键字值是否在这一对数据的范围之内(树中所有的结点的关键字值都在树的最左和最右的结点的关键字值之间)。最后，对这些元素采用自根向下压缩，更新各结点的Nstrhashdata值，这样可以使得更新数据的工作量大为减少。

关于删除叶子结点的算法描述如下：

```
void DeleteLeafNode(Binarytree &Leaf)
// 从二叉排序证书吊销树中删除叶子结点 Leaf
{
if(Degree(Leaf->Parent)==1)    //Degree(Node) 返回结点 Node 的度
  SubstituteNode(Leaf,Leaf->Parent);
//用 Leaf 的双亲结点来代替欲删除的结点在双向链表中的位置
DeleteNodeOfTree(Leaf->Parent,Leaf);    //从树中删除 Leaf.
else  //双亲结点有两个孩子，只须在树中和双向链表中同时删除此结点
{
       Leaf->Rchild->Lchild=Leaf->Lchild;
       Leaf->Lchild->Rchild=Leaf->Rchild;
       DeleteNodeOfTree(Leaf->Parent, Leaf); //从树中删除结点 Leaf.
}
}
```

关于删除叶子结点的算法描述如下：

```
int DeleteNode(Binary &T,Node *e)
{ if (T==NULL)
  return FAIL;
else if(T->NodeTag==LEAF)
DeleteLeafNode(Binarytree &T);
else if(T->NodeTag==NODE)
{
      if(e->Cnum<T->Cnum)
        DeleteNode(T->Lchild,e); //递归调用从左子树中删除结点 e
```

```
        else if(e->Cnum>T->Cnum)
          DeleteNode(T->Rchild,e);  //递归调用从右子树中删除结点 e
        else   //e->Cnum=T->Cnum,即要删除的结点是 T
         if(Degree(T)==0)
            DeleteLeafNode(T);
         else if(Degree(T)==1)
           SubstitudeNode(T,T->Parent);
  //T 的双亲仅有一个孩子，用双亲结点替换此结点
        else
          {
            p=searchPredecessorNode(T);  //将 T 的前驱结点返回
            SubstitudeNode(T,p);
            Delete(p);
          }
    }
}
```

本算法中，没有包括从修改的结点至叶子结点的修改 Hashdata 的操作，也没有给出对双向链表的 Nstrhashdata 修改的过程。若一次若删除的结点过多，可以将这些操作放在一起来调整，可以提高效率，避免重复修改。

4.4 方案二：叶子结点组成一个 CRT 树

4.4.1 数据结构

设 CA 吊销的证书的编号集为 $S = \{n_0, n_1, \ldots, n_{n-1}\}$，为了使二叉树的左右保持基本的平衡，根结点 n_0 是一个被吊销证书集中的中间值。下面以证书编号为关键字，以 n_0 为根结点，对 S 构造排序二叉树[54]，使每个结点的数据结构为：

Lchild	Parent	Cnum	Hashdata	NodeTag	Rchild

表 4.2 方案一中结点的数据结结构

Table 4.2 Data structure of Tree' s node in second theme

说明：

1)Lchild,Rchild：指向该结点的左孩子，右孩子的指针。

2)Parent：双亲结点的指针。

3)Cnum：该结点所代表的吊销的证书的编号。

4)Hashdata：该结点与其双亲结点的证书编号的 HASH 值（Hashdata =HASH(Cnum ||Parent->Hashdata)）。

5) NodeTag：表明该结点的类型：当结点是非叶子结点时 NodeTag=NODE ；当结点是叶子结点时 NodeTag=LEAF。

数据结构定义如下：

```
typedef structure CRTFBDHT
{
   structure CRTFBDHT *Lchild,*Parent,*Rchild;
   datatype Cnum;
   char *Hashdata;
}
```

4.4.2 具体方案

先将各结点的 Cnum 的内容写入欲吊销的证书编号，再按文献 [54] 中的标准算法建立二叉排序树。接下来按方案一的自根向下压缩方案修改每个结点的的 hashdata 项。最后将二叉排序树的叶子结点建立成的一个 CRT 树。但与文献中的 CRT 不同的是结点的个数近似地等于原来的一半，结点个数大大减少。

4.4.3 查询

查询时，按照二叉排序树的算法 [54]，从根结点开始向下开始搜索。当找到符合条件的结点，返回沿着此结点的双亲结点一直向下找到一个叶子再按 CRT 的方案向 CRT 的根结点方向直至根结点(CRT 树的根结点)路径上的所有相关结点，作为查询结果。查询者根据返回查询内容进行验证。

4.4.4 操作

插入结点与删除结点使用文献 [54] 中的标准算法操作，再对叶子结点重新组织 CRT 树。

4.5 方案三：叶子结点组成一个二叉排序树结构

4.5.1 数据结构

设 CA 吊销的证书的编号集为 $S = \{n_0, n_1, \ldots, n_{n-1}\}$，为了使二叉树的左右保持基本的平衡，根结点 n_0 是一个被吊销证书集中的中间值。下面以证书编号为关键字，以 n_0 为根结点，对 S 构造排序二叉树[54]，使每个结点的数据结构为

Lchild	Parent	Subparent	Cnum	Hashdata	Nstrhashdata	NodeTag	Rchild

表 4.3 方案三中结点的数据结结构

Table 4.3 Data structure of Tree' s node in third theme

说明：

1) Lchild, Rchild 是指向该结点的左孩子，右孩子的指针。

2) Parent 是双亲结点的指针。

3) Subparent 是新的二叉树中结点指向其双亲结点的指针。

4) Cnum 该结点所代表的吊销的证书的编号。

5) Hashdata 是该结点与其双亲结点的证书编号的 HASH 压缩值（Hashdata = HASH(Cnum|| Parent->Hashdata)）。

6) Nstrhashdata 是利用叶结点的空链域建立的新数据结构的 HASH 值，这些 HASH 值用以验证树中结点所代表的证书被吊销的有效证明。

7) NodeTag 是表明该结点的类型：当结点是非叶子结点时 NodeTag=NODE；当结点是叶子结点时 NodeTag=LEAF。

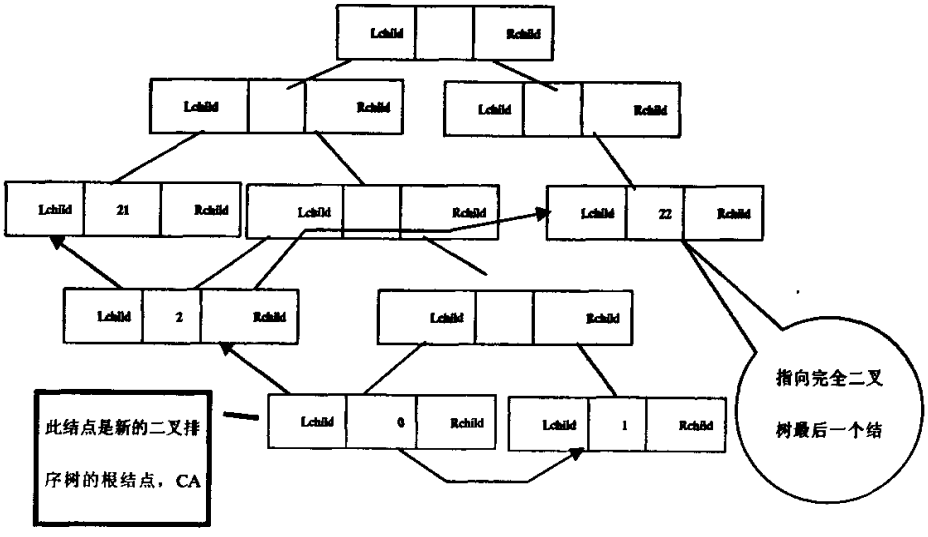
数据结构定义如下：

```
typedef structure CRTFBDHT
{
    structure CRTFBDHT *Lchild,*Parent,*Rchild, *Subparent;
    Datatype Cnum;
    char *Hashdata,*Nstrhashdata;
}
```

4.5.2 具体方案

先使用标准的二叉排序树建立算法建立一个各个结点与吊销证书相符的二叉排序证书吊销树，再对二叉树按前述方法对各结点采用自根向下压缩。除根结点

的 Hashdata=Cnum,其余结点的 Hashdata=HASH(Cnum|| Parent->Hashdata),每一个结点都包括上层结点的信息。然后对叶子结点按文献[54]的标准算法建立完全二叉树(完全二叉树可以使树高度降低，查询时返回结果长度最小)，使新的完全二叉树中叶子结点的 Nstrhashdata=hashdata,其余结点的结点的 Nstrhashdata=hash(Lchild->Nstrhashdata|| Rchild->Nstrhashdata)。这样，就又将整个二叉树中所有结点的信息汇集在新的二叉树的根结点中，形成一个树中树的双重二叉树的数据拓朴结构，CA 再对新的二叉树的树根进行签名，作为对整个证书吊销树的签名。新树不需要新增结点，它利用原叶子结点的空链域建立。



注：——— 原二叉排序树

——▶ 利用原二叉树的叶子结点建立的新二叉树

图 4.2 用二叉排序树的叶子结点组成完全二叉树的证书吊销树

Fig4.2 the certificate revocation Tree of complete tree using the leaf node of binary sort tree

在上图的二叉排序树中，结点序号为 21、2、0、1、22 为二叉排序树的叶子结点。用它们组成了如图所示完全二叉树，结点 0 是完全二叉树的根结点。新的完全二叉树不需要新增结点，它利用原叶子结点的空链域建立。CA 对完全二叉树的根结点签名，用于验证数据的完整性。而指针 End_point 指向完全二叉树中最后的指针，用于插入和删除结点的算法中。

4.5.3 查询与验证

当用户查询证书是否被吊销时，从原二叉排序树的根结点开始向下查找。如果有结点符合查询条件时，返回它的双亲结点向下查找至叶子结点再从该叶子结点根据 Subparent 指针经过新的二叉树的根结点路径的相关结点作为查询结果。

用户收到返回的查询结果后，验证第一个结点是否是要查询的证书，然后检查压缩数据及签名是否正确。

4.5.4 插入结点

为了便于操作，名录服务器可以使用一个 End_point 的指针，用它来指向完全二叉树中最后一个结点(叶子结点所在层的最右边的结点)。

插入结点为原二叉排序树的叶结点，根据插入结点的双亲结点是否有孩子分为两种情况：

当插入结点的双亲结点原来有一个孩子，则将新增结点加入二叉排序树中，为了使新结点加入到完全二叉树中，而且新的二叉树插入结点后仍是完全二叉树，采用回溯算法根据 End_point 所指结点很快找到新的插入位置(Endnode 所指结点的右边的第一个空结点位置，当为满二叉树时，Endnode 指下一层的第一个结点)插入结点，并修改指针使它成为完全二叉树中的结点，并使 End_point 指向新增的叶结点。

当插入结点的双亲结点原来是一个叶子结点，则新增结点替换原来双亲结点在完全二叉树的位置。将双亲结点的值与新增结点交换，原二叉排序树中指向双亲结点的指针指向它，让双亲结点作为它的孩子，其中一个指针修改为指向它，另一个指针为空。

插入的结点必须修改指针及 Nstrhashdata，使原来的结点的二叉排序树的结构性保持不变。而且修改从变化的结点至完全二叉树的根结点所有结点的 Nstrhashdata 的值。如果一次新增结点的数量较多时，可以先插入结点，再一次性更新完全二叉树中的 Nstrhashdata 的值。

方案三的插入结点的算法如下：

```
int InsertNodeOfBinarySortTree(BinarySortTree &T,Node *e)
//将结点 e 插入自根向下压缩的二叉排序证书吊销树中
{
  if(e->Cnum==T->Cnum)
```

```
        return FAIL;     //证书已存在二叉排序证书吊销树中
      else if(e->Cnum>T->Cnum)    //将 e 插入 T 的右子树中
         {
    if(T->Rchild)
    /*若 T 的右子树非空，递归调用 InsertNodeOfBinarySortTree()将 e 插入 T
的右子树*/
    InsertNodeOfBinarySortTree(T->Rchild,e);
    else  //把 e 做为 T 的右孩子
    {
       if(IsLeafNode(T))
     //若 T 为叶子结点，交换 e 和 T 的 Cnum，并修改指针,完全二叉树结构不变
       {
           swap(t->Cnum,e->Cnum);
           if(IsLeftChild(T->parent,T)
    //若 T 是它的双亲结点的左孩子
           {
             e->Rchild=T;
             e->parent=T->parent;
             T->parent->Lchild=e;
              ->parent=e;
           }
    else   //若 T 是它的双亲结点的右孩子
       {
         e->Rchild=T;
         e->parent=T->parent;
         T->parent->Rchild=e;
         T->parent=e;
    }
    }
    else  //T 不是叶子结点
```

```
    {
        T->Rchild=e;
        p=SeekNextPosition(End_point);
/*根据End_point用SeekNextPosition()找到完全二叉树中下一个插入位置p,
使e做为p的孩子*/
 if(!p->Lchild) //p的左孩子为空，e就是p的左孩子
      {
          p->Lchild=e;
          e->Subparent=p;
      }
    else  //p的左孩子为空，e就是p的右孩子
     {
          p->Rchild=e;
          e->Subparent=p;
     }
    End_point=e;  //修改End_point,使它指向e
    }
    }
  }
   else  //将e插入T的左子树中
 {
    if(T->Lchild)
/*若T的左子树非空，递归调用InsertNodeOfBinarySortTree()将e插入T的左
子树.*/
    InsertNodeOfBinarySortTree(T->Lchild,e);
    else   //把e做为T的左孩子
       {
          if(IsLeaf(T))
             {  //若T为叶子结点，交换e和T的Cnum，并修改指针
             swap(t->Cnum,e->Cnum);
```

```
if(IsLeftChild(T->parent,T)
//若T是它的双亲结点的左孩子
   {
     e->Lchild=T;
     e->parent=T->parent;
     T->parent->Lchild=e;
        T->parent=e;
      }
    else
  //若T是它的双亲的右孩子,e就是T的双亲结点的右孩子
    {
     e->Lchild=T;
     e->parent=T->parent;
     T->parent->Rchild=e;
        T->parent=e;
      }
   }
else    //T不是叶子结点
    {
     T->Lchild=e;
  p=SeekNextPosition(End_point);
       /*根据End_point用SeekNextPosition()找到完全二
    叉树中下一个插入位置p,使e做为p的孩子*/
  if(!p->Lchild)  //p的左孩子为空，e就是p的左孩子
      {
       p->Lchild=e;
       e->Subparent=p;
      }
  else   //p的左孩子为空，e就是p的右孩子
  {
```

```
                p->Rchild=e;
                e->Subparent=p;
            }
            End_point=e;  //修改 End_point,使它指向 e
        }
    }
}
```

4.5.5 删除结点

根据删除结点在原二叉排序树的位置，可以分为两类来处理：

当删除的结点是二叉排序树的叶子结点时，如果删除的结点的二叉排序树的双亲结点又变为叶子结点时，则修改其双亲结点使它代替欲删除的结点指针域的值，这样就不用改动新的完全二叉树的结构。否则，先从二叉排序树中删除该结点[64]，将它与 End_point 所存放的结点进行交换，将删除 End_point。这样就从两种数据拓朴结构中将此结点删除了。然后修改 End_point，使它指向刚删除结点的在完全二叉树前边的第一个结点(如果为最下一层第一个结点，则指向上层最后一个结点，否则使它指向新的完全二叉树的最后一个结点)。

当删除的结点是二叉排序树中的非叶子结点时，则按文献 [54] 的方案，用它的前驱结点(原二叉排序树中的叶结点)代替欲删除的结点，删除它的前驱结点的问题是一个删除叶结点的问题用前述方法删除该结点。

以下算法是在二叉排序树中删除叶子结点 e:

```
int DeleltLeafNode(BinarySortTree &T,Node *e)
{
   if(Degree(e->parent)==1) / /e 的双亲在删除 e 之后变成叶子结点,
    {
    swap(e->Cnum,e->parent->Cnum); //交换 e 和 e 的双亲的关键字值
    if(IsLeftChild(e->parent->parent,e->parent)
//e 的双亲是左孩子，交换后，删除 e 的双亲不影响新的完全二叉树的结构
        {
        p=e->parent;
        p->parent->Lchild=e;
```

```
        e->parent=p->parent;
        delete(p);
        }
        else   //e 的双亲是右孩子
        {   p=e->parent;
        p->parent->Rchild=e;
        e->parent=p->parent;
        delete(p);
        }
}
   else   //e 的双亲有两个孩子
    {
    if(IsLeftChild(e->parent,e))  //从二叉排序树将 e 删除
       {
          e->parent->Lchild=NULL;
       }
     else
      {
       e->parent->Rchild=NULL;
       p=SeekPrior(End_point);
       swap(End_point,e);
       }
   if(IsLeftChild(End_point->Subparent,e))  //从二叉完全树将 e 删除
     {
       e->Subparent->Lchild=NULL;
      }
     else
       {
       e->Subparent->Rchild=NULL;
      Delete(End_point );
```

```
        End_point=p;
        }
      }
}
```

以下算法是在二叉排序树中删除结点 e:

```
int DeleltNode(BinarySortTree &T,Datatype e)
{
  if(e==T->Cnum)  //T 就是要删除的结点
  {
    if(IsLeafNode(T))
       {
        DeleteLeafNode(T->Parent,T);
        }
    else
    {
    p=seekPriorNode(T); //查找 T 在二叉排序树中的前驱结点 p
    swap(t->Cnum,p->Cnum);
    DeleteLeafNode(p->parent,p);
    }
 }
else if(e>T->Cnum)
   {
DeleteNode(T->Rchild,e);  //递归调用 DeleteNode(),在右子树中删除 e
    }
else
{
   DeleteNode(T->Lchild,e);  //递归调用DeleteNode(),在左子树中删除e
 }
}
```

4.6 效率分析

由于二叉树上叶子结点的个数近半数，近百分之七十五的结点在最下面两层，结点的插入、删除的结点几乎都在底层，不用更新太多的结点，效率提高了许多。

由于CA维护证书吊销树，必是一次更新多个结点，本方案在批量更新时，效率更高。

以下将从通信长度，插入与删除操作的算法复杂度与CRT与文献[33]的方案分别进行比较(表3-1)：

	通信的结点数	插入操作复杂度	删除操作的复杂度
CRT[49][50]	log(2n)	n个结点重新建树	n个结点重新建树
线索二叉排序证书吊销树	2log(n)	O(mlog(n))	O(2mlog(n))
方案一	Log(n)~n/2	O(n/2)	O(3n/2)
方案二	Log(3n/2)	n/2个结点重新建树	n/2个结点重新建树
方案三	Log(n/2)	O(n/2)	O(3n/2)

表4-4 效率分析

Fig 4.4 Efficience analysis

m为一次批量操作的结点数。

4.6 三种自根向下压缩的证书吊销方案在PKI中应用

4.6.1 插入与删除同时进行时对算法的优化

当一次插入多个结点与删除多个结点同时进行，可以将插入与删除操作间隔进行。先删除一个结点，用一个指针P保存此结点不要先删除此结点，再插入一个结点，当需要一个将它插入至完全二叉树的位置时，就可以把P所指的结点的指针域的值复制给新增的结点，再删除P所指的结点。这样既可以减少增删结点操作步骤，又可以减少需修改Nstrhashdata结点的个数。提高了算法的效率。

用一个数组记录更改的位置，每当有一个结点进行更新时，将它与以前的变动位置进行比较，如果此位置与以前的更新途径重复，只保留一条更新路径，可以提高更新的效率。

4.6.2 三种方案的应用范围

本文所出的三种方案，均可以很好地用于PKI中。方案一、二在目录服务器到验证者之间返回的查询结果所用结点较多，因此用在证书数量较少的情况下。方

案三，采用双重树结构，证明查询结果所用的结点少，更适合一些用于管理证书数量庞大的 PKI 环境中。

4.6.3 客户端的设计

所提出的三种方案都是动态维护的证书吊销方案。它的特点是及时性强，风险小。由于传输的信息少，对带宽要求不是很高。但它又与其它动态的证书吊销方案一样，提高了效率。为了降低请求率，对客户端(查询验证者)做如下设计：

在客户端添加一个缓冲池，每次请求查询时，对返回的结果进行验证后，将返回结点中没有存入缓冲池的结点存入缓冲池。并根据证书的有效期定期删除已过期的结点。有了缓冲池的客户端，查询过程如图 4.3 所示：

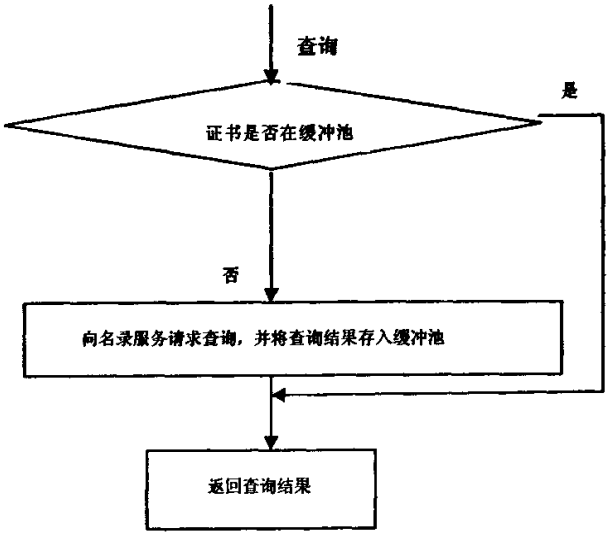


图 4.3 带缓冲池的查询流程图

Fig 4.3 Query system with Buffer

4.7 小结

公钥密码以其安全性高，公钥公开，应用越来越广泛。但 CA 发行的证书的吊销问题将成为公钥基础设施(PKI) 中日益重要的课题。当使用公钥时，必须确认公钥未被吊销。供查询的名录服务器既要保证安全性，又要提高查询效率。因此构造快速、高效的证书吊销机制具有重要意义. 而且在电子商务中证书系统广泛使用，证书吊销机制必将是维护系统安全的重要保证。

所提方案，利用二叉排序树中叶子结点的空链域，对叶子结点重新组织成一

个新的数据拓朴结构，查询输出结点少，安全性高，可防止证书吊销树被破坏性地修改。而且由于将签名位置移向一个叶子结点，增删结点时不需要向根方向修改，只将叶子结点进行更新，减少维护树的代价。因此我们的方案是可以有效地解决证书吊销问题的方案。

第五章　基于双线性映射的具有前向安全的概率签名方案

5.1 引言

普通数字签名能为消息的传输提供完整性和不可否认性，然而一旦签名密钥由于人为疏忽或系统本身的缺陷而遭到攻击致使密钥被泄露，攻击者就可以仿造以前的签名。因此，以前的签名就会全部作废；签名者有时为了否认对自己不利的签名，主动地泄露自己的密钥，然后以自己的密钥被盗，否认自己的签名，使他人的利益受损。为了保证签名的安全性和有效性，如何减少由于密钥泄露所带来的对系统安全的影响，一直是人们十分关注的研究课题。

1997年，Anderson首次提出了前向安全签名的概念[11]。Bellare 和Miner第一次给出了前向安全签名的正式定义，并基于A.Fiat 和A.Shamir的签名方案给出了两个前向安全签名方案[12][50]。此后，在各种加密体制下，都提出了采用前向安全的体制。

前向安全签名的基本思想是：把整个有效时间分成若干个时间段,在每个时间段内使用当前时段的的签名密钥产生签名,而验证签名的公钥在整个有效时间内都保持不变。即使当前时间段的签名密钥被泄露,由于无法得到以前的密钥，因而不能伪造以前的签名，以前时间段的签名是有效的，从而减少了由于签名密钥泄露而对系统带来的负面影响。

在各个时间段内使用不同的签名密钥，这就要求在系统中增加密钥进化算法。密钥进化算法在公钥的生存期为T之内的各个时段中，每个时段（每个时段根据系统的安全要求可以规定为若干个小时、一天、一周或一个月或任意一段时间等），由前一个时段的密钥经一个单向函数运算得来，当新的私钥产生后，立刻删除以前的私钥。

在前向安全签名方案中，攻击者既使在i时段入侵系统获得签名密钥，由于前一时段的密钥经一向单向函数计算出下一个时段的密钥。因此，不能由t时段的密钥计算出以前的密钥，就无法伪造i时段之前的签名。既使i时段的签名密钥泄漏后，系统在i时段之前所作的签名仍然有效。正是由于采用这样的安全机制，前向安全签名增加了系统的安全性，最大限度地降低了由于密钥泄漏而造成的损失。

以下是前向安全签名过程中秘密钥在生命周期的使用及进化情况，如图 5.1 所示。

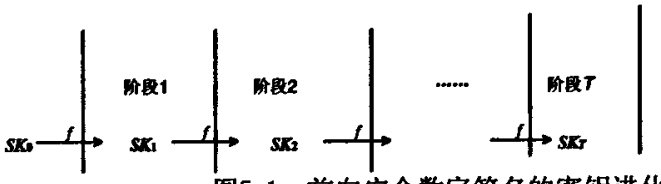


图5.1 前向安全数字签名的密钥进化过程

Fig5.1 updating key of forward security signature

5.2 前向安全的签名方案算法描述

在公钥密码体制中，前向安全的签名方案包括以下四个算法：

(1)密钥生成算法GEN(T,A,q)：这是一个概率算法,输入时段总数T和安全系数A，算法返回一个初始公私钥对(SK_0,P_k)。

(2)签名算法SIG(SK_i,i,T, q,M)：在第i时段，输入密钥SK_i,i,T和消息M,算法返回第i时段关于消息M的签名SIG(i,SIG).

(3)验证算法VER(M,i,SIG, q, P_k)：这是一个确定性算法,输入公钥P_k,消息M,时段i和签名SIG(i,SIG),如果签名是有效的，则VER(M,i,SIG, q, P_k)＝true;否则,VER(M,i,SIG, q, P_k)＝false;

(4)密钥进化算法UPDATE(SKi, i)：此算法输入当前时段i(0<i≤T)和密钥SKi，返回下一个时段的密钥SKi+1。

5.3 前向安全数字签名的几种解决方案

以下的方案[40]采用的数字签名方案(例如:RSA 或 DSS 签名方案)中的密钥生成算法为 GEN，签名生成算法为 SIG，签名验证算法为 VER。

1. 长公钥长秘密密钥方案：

系统建立时，签名者重复执行密钥生成算法产生 T 个密钥对：(p_1, s_1)，(p_2, s_2)，…，(p_T, s_T)，其中 P_i 为签名的公开密钥，s_i 为相应的签名秘密密钥(i =1，2，…，T)。我们设密钥进化方案的公钥为 PK =(p_1, p_2, …, p_T }Pr)以及初始密钥 Sk_0=(s_0, s_1, s_2, …, s_T)，这里 s_0 为空串。密钥的进化过程如下进行，当进入到第 i 时段，签名者删除 s_{i-1}，那么签名者 i 时段的密钥为 SK_i=(s_i, s_{i+1}, …, s_T)。在 i 时段消息 m 的签名为<i, SIG(m, s_i) >。消息 m 的签名<i, ξ >(其中 ξ = SIG(m, s_i))的验证就是检验 VER(m, p_i,)=1 是否成立。这个方案是前向安全的，但该方案的问题在于公钥和密钥的长度随着总时段数 T 线性地增长。

2. 短公钥长秘密密钥方案:Andersor 提出的方案对以上方案进行了改进，它的特点是公钥短，但秘密密钥长度依然随着 T 线性增长。签名者同以上的方案一样

首先产生T个密钥对(p_1, s_1)，(p_2, s_2)，…，(p_T, s_T)，并产生一个附加的密钥对(p, s')，令$\delta_i = SIG(i||P_i||s)(i=1, 2, \cdots, T)$，即$\delta_i = SIG(i||P_i)$，然后删除s。系统的公钥为P(短公钥)，系统的初始密钥为$SK_0=(s_0, \delta_0, s_1, \delta_1, \cdots, s_T, \delta_T)$，其中$s_0$，$\delta_0$为空串。

密钥的进化过程如下：一旦进入i时段，签名者完全删除S_{i-1}，δ_{i-1}，那么i时段的秘密密钥$SK_i=(s_i, \delta_i, s_{i+1}, \delta_{i+1}, \cdots, s_T, \delta_T)$。i时段消息m的签名为$<(i, (SIG(m, s), p_i, \delta_i)>$，则对消息m的签名$< i,(\alpha, q, \delta)>$的验证为检验$VER(i||q, P, \delta)$=true以及$VER(m, q, \alpha)$=true是否同时成立。这个方案提供了前向安全性，但问题在于签名秘密密钥长度随总时段数T线性增长。

3. 长签名方案:通过使用证书链签名的公钥和私钥的长度都很短，但是产生了长的签名.签名者开始生成密钥对(p_0, s_0)。签名者设密钥进化方案的的公钥P_0，初始私钥s_0。在每一个时段一个新的密钥及证书就生成了，然后删除以前的密钥及证书。证书链包含在签名中。具体的说：在第1时段的开始，签名者生成一个新的密钥对(p_1, s_1)，设$\delta_1 = SIG(1||(p_1||s_0)$，然后删除$s_0$。在1时段消息m的签名$<1, (SIG(m, s_1), p_1, \delta_1)>$。那么消息m的签名$<1, (\alpha, q_1, \tau_1)>$被验证通过检验$VER(m, \alpha, q_1)$=true和$VER(1||P_0, q_1, \tau_1)$=true是否成立。当时段$i \geqslant 2$，签名者拥有$(P_1, \delta_1, P_2, \delta_2, \cdots, P_{i-1}, \delta_{i-1})$和前一个时段的密钥$s_{i-1}$，创建一个新的密钥对$(p_i, s_i)$，设$\delta_i = SIG(i||(p_i||s_{i-1})$并删除$s_{i-1}$。时段i消息m的签名为$< i, (SIG(m, s_i), p_1, \delta_1, \cdots, p_i, \delta_i)>$。那么消息m的签名$<<i,(\alpha, q_1, \tau_1, \cdots, q_i, \tau_i)>$的验证通过检验$VER(m, \alpha, q_i)$=true和$VER(i||q_1, \tau_1, q_{i-1})$=true以及$VER(1||q_1, \tau_1, P_0)$=true是否成立。这个方案明显的可以得到前向安全性，并且签名的公钥和私钥的长度独立于总时段数T。但是签名的长度随着T线性增长。

5.4 对文献[24][25]的前向安全性的分析

文献[24]中基于零知识证明的安全协议，所提出的前向安全签名方案。原方案的签名过程如下(对参数的描述可参阅原文)：

签名过程如下:签名者任选k,计算$c=H(m \| g \| y_T \| i \| g^k)$，这里$H:\{0,1\}^* \rightarrow \{0,1\}^l$为单向安全Hash函数,则,令$s=k-c \times sk_i^{2^{T+1-i}} \bmod N$,则签名者对信息m的签名为$<i, SIG(m, sk_i), yT>=\{c, s\}$.

分析:在上述方案上对s的计算中,其中$sk_i^{2^{T+1-i}} = sk_{i-1}^{2^{T+1-(i-1)}} = \cdots\cdots = sk_0^{2^{T+1}} = yT$(该

系统下公钥)。则：$s=k-c\times sk_i^{2^{T+1-i}} \bmod N= k- yT\times c \bmod N$,而c中本身不包含私钥的信息。也就是说，该方案中使用了公钥对m进行签名，不仅仅不具备前向安全性，而且由于签名结果中没有私钥信息，已经不具备一般意义上安全性。

在方案[25]中基于椭圆曲线提出的前向安全的签名方案中，同样是在签名的过程中，最终结果是采用了公钥进行签名，也不具备前向安全性和安全性。

通过以上分析得到以下结论：

结论一：在签名过程中，计算$s=k-c\times f(sk_i)$时，私钥运算后的结果$f(sk_i)$，用来代替私钥进行签名，而$f(sk_i)$必须是一个只有私钥拥有者知道而对其它实体不可知或在通过运算得到是困难的一个秘密。否则,系统就存在伪造签名,不具备安全性。

结论二：在第i时段的前向安全的签名的验证过程中，验证运算必须包含时段i的信息。且i是不可替代的。若i可以被别的值（如i-1,i-2等）替代，攻击者就可以用获得的密钥对以前的签名进行伪造，从而不具备前向安全性。

5.5 基于双线性对的具有前向安全的概率签名方案

5.5.1 预备知识

1.定义

强安全素数:设p'，q'为两个大素数，$q=2p'+1$，若q为素数，则称q为强安全素数。

2.双线性对知识

(1) GDH群

在群G中，P是G的生成元。

CDHP(计算Diffie-Hellman问题)：给定三元组$(P,aP,bP)\in G^3$,对$a,b\in Z_q^*$,计算abP。

DDHP(确定Diffie-Hellman问题)：给定四元组$(P,aP,bP,abP)\in G^4$，对$a,b,c\in Z_q^*$，判断c=ab mod q是否成立。

GDH(Gap Diffie-Hellman问题)：这是一类CDH困难而DDH容易的问题。

GDH群：GDH群是CDH问题困难而DDH问题容易的群。

本文提出的一个具有前向安全性质的签名方案，是基于双线性对密码体制中所用的GDH群的性质（在GDH群中CDHP是困难的，而DDHP是容易的）。

3.双线性映射

令G1为由P生成的循环加法群,阶为q ,G2为具有相同阶q的循环乘法群。G1和G2这两个群都是GDH群。

双线性映射是指在G1和G2这两个群中存在满足下列性质的一个映射e:G1×G1→G2e满足:

(1) 双线性: e(aP,bQ)=e(P,Q)ab;

(2) 非退化性:存在$P,Q\in G_1$,使得e(P,Q)≠1;

(3) 可计算性:对所有的P,Q∈G1存在有效的算法计算e(P,Q)。

5.5.2 签名方案

1. 系统参数

强安全素数:设p'，q'为两个大素数，$q=2p'+1$，若q为素数，则称q为强安全素数。

(1) 令P为G1的生成元,G1,G2的阶均为q,q是一个G1,G2上满足CDH问题困难的强安全素数。在G1,G2之间构造一个映射e:G1×G1→G2.选择两个Hash函数H1:{0,1}*×G1→Z_q^*, H2:{0,1}*→G1。将签名密钥的有效期分为T个时间段,选择随机数s, $s\in_R Z_{q-1}^*$。SK0=s, Pk =$SK_0^{2^{T+1}}$ mod (q-1).

(2) 公开系统公钥及相关参数{P,q,T,Pk,G1,G2,e,H1,H2},初始私密钥为SK0.

2. 签名者私钥的更新算法

当系统进入i时间段时,1≤i≤T,签名者使用拥有的i-1时间段的密钥SKi-1,计算SKi=SK_{i-1}^2mod (q-1).删除系统中i-1时间段的密钥SKi-1,保密并使用新生成的第i时段的密钥SKi.

3. 签名过程

签名者选一个随机数x, $x\in_R Z_{q-1}^*$,计算X=($x^{2^{T+1-i}}$ mod (q-1))P，令h=H1(M,X),Y=x×h×SKi,mod(q-1),输出(X,Y)作为对消息M的签名。

4. 验证过程

接收到消息M和签名(X,Y)后,验证者检验下列等式是否成立h=H1(M,X),Q=H2(M),判断以下等式是否成立:e(P,$Y^{2^{T+1-i}}$Q) =e(((Pk×$h^{2^{T+1-i}}$)mod (q-1))X,Q)是否成立,i为当前的时段,若该等式成立,则接受签名,否则,签名是无效的。

5.5.3 方案的有效性证明

证明：签名的结果为(X,Y),$h=H_1(M,X)$,$Q=H_2(M)$.若签名正确,则有:

$$e(P,Y^{2^{T+1-t}}Q)=e(P,((x\times h\times SK_t)^{2^{T+1-t}} \bmod (q-1))Q)$$

$$=e(P,((x\times h)^{2^{T+1-t}}\times SK_t^{2^{T+1-t}})\bmod (q-1))Q)$$

$$=e(((x\times h)^{2^{T+1-t}}\times SK_0^{2^{T+1}})\bmod (q-1))P,Q)$$

$$=e(((P_k\times h^{2^{T+1-t}})\bmod (q-1))X,Q)$$

故签名是正确的,有效的。

5.5.4 方案的安全性证明

以下给出本方案存在的攻击：

(1)攻击者伪造一个消息M的签名，由签名过程可知，攻击者伪造一个Y的前提是攻击者知道私钥，这是不可能的。

(2)攻击者根据签名结果去计算私钥，当攻击者从(M,X,Y)计算sk_t,攻击者必须从X中计算中计算出x,而由X计算$x^{2^{T+1-t}}$是G_1群上的DLP问题，在计算上是不可行的。

(3)由签名验证的过程来分析，若攻击者可通过$(P_k\times h^{2^{T+1-t}})\bmod (q-1)X=Y^{2^{T+1-t}}P$来计算Y或由Y来计算x，其问题等同于解决了$G_1$群上的DLP和模合数的平方剩余问题。

通过以上分析，基于双线性映射的签名方案是安全有效的。

5.5.5 方案的前向安全性分析

假设在第i时段，窃听者入侵系统后，成功获得密钥SK_t，由于$SK_t=SK_{t-1}^2 \bmod (q-1)$，由$SK_t$计算$SK_{t-1}$是模合数的平方剩余问题，在计算上是困难的。从而,以前的密钥的是安全有效的，既可以防止攻击者伪造第i时段以前的签名,又可以防止签字方对以前的签名以密钥泄漏为由予以否认，满足签名的不可否认性和抗伪造性。

5.6 小结

本文首先针对已有的方案，分析了它们的安全性的缺陷，并得出了相关的结论。基于上述的分析，给出了基于双线性映射的一种具有前向安全的概率签名方案　，并证明是安全　有效地。双线性映射是一种新的公钥密码体制，具有安全性高，密钥短，计算量小等优点。因此本文基于双线性映射，提出一种能够提高系统的安全性而实现简单的一种具有前向安全性质的概率签名方案。

第六章 结 束 语

1.在围绕证书撤销机制在 PKI 中的应用及证书撤销的技术，我们做了以下方面的工作：

(1)对 PKI 中的几个主要实体进行了介绍，并且对它的工作原理和运行的各个环节进行介绍；介绍了 X.509 的证书结构、证书的格式及证书的主要用途。

(2)针对当前应用广泛的证书撤销列表 CRL 及基于 CRL 的几种方案:增量 CRL、分段 CRL、重复发布 CRL、CRT、二叉排序证书吊销树从峰值请求、峰值带宽、验证方下载代价及风险方面进行了比较，指出了各种机制的特点和优缺点。

(3)证书撤销树是一种具有较好应用前景的证书撤销机制，与 CRL 相比，客户端只需下载要查询的目标证书的状态信息，下载代价较小;CA 和证书撤销库之间更新每次只需传送以上一次为基准的变化量，更新时间无具体的限制,更新方式灵活,有效地避开了峰值带宽,使得请求率几乎呈均匀分布.但每次新增加的撤销证书或删除撤销证书时，CA 必须为目录服务器端重新构建证书撤销树，建树的工作量大。

(4) 基于二叉排序树的证书吊销方案中更新结点都涉及叶子结点，更新路径从树的最深层更新至根结点。若更新的结点稍多时，工作量很大，离根结点近的结点被多次重复修改，更新结点代价甚至可能超过建树的代价，该方案的数据维护量相当大。分析这个方案的基础上，我们提出了三种新的自根向下压缩的二叉排序 HASH 证书吊销树方案，利用二叉排序树中叶子结点的空链域，对叶子结点重新组织成一个新的数据拓朴结构，查询输出结点少，安全性高，可防止证书吊销树被破坏性地修改。由于将签名位置移向一个叶子结点，增删结点时不需要向根结点方向更新，只需向叶子结点进行更新，减少维护树的代价和工作量。因此我们所提方案是可以有效地解决大规模的 PKI 中证书吊销问题的方案。

2. 具有前向安全性质的签名方案

前向安全是一种新的安全体制，被引入多种安全领域,有着广泛的应用前景。但已有的有些数字签名方案存在缺陷，在分析这些方案存在的安全漏洞的基础上，总结得出了前向安全体制的一般原则，并给出了一个基于双线性映射的具有前向安全性质的数字签名方案。

3. 今后的工作

尽管我们对 PKI 中证书撤销和具有前向安全性质的数字签名的研究做了一些

工作，但由于本人水平和时间有限，有许多工作还需进一步研究和完善。

主要包括：

(1)如何建立一种效率更高的二叉排序证书吊销树。在更新时，减少维护的工作量；在查询时减少和通信代价。

(2)对本文所提的算法进一步优化，变得效率更高。

(3)研究前向安全在其它密码系统中应用和更有效的方案。

参考文献：

[1] William Stallings，密码编码学与网络安全一原理与实践(第三版)[M]，刘玉珍等译，北京：电子工业出版社，2004,210-227.

[2] 谢冬青，冷健.PKI 原理与技术附].北京:清华大学出版社.2005,22-27.

[3] C.Adams,R.Zuccherato,Internet X.509 Public Key Infrastructure Data Certification Server Protocols,Internet Draft,PKIZ Working Group,1998.

[4] Kohnfelder,L.M.,Towards A Practical Public-key Cryptosystem,B.S.Thesis, supervised by L. Adleman.M1T Laboratory for Computer Science,May,1978.

[5] Carlisle Steve.公开密钥基础设施一概念、标准和实施[M],冯登国等.北京:人民邮电出版社，2001, 87-97.

[6] Gassko I, Gemmell P S., MacKenzie P.Efficient and Fresh Certification.In:Proc of International Workshop on Practice and Theory in Public Key Cryptography(PKC2000) volume 1751 Lecture Notes in Computer Science. Melbourne,springer-Verlag, 2000, 342-353.

[7] John Iliadis, Diomidis Spinellis, Evaluating Certificate status information mechanisms.In: Proc of the 7th ACM conference on Computer and communications security. ACM Press, 2000, 1-8.

[8] Merkle R.Secury,Authentication and Public-Key Systems:[dissertation].Department of Electrical Engineering,Standford University,1979.

[9] Merkle R C. Protocols for Public Key Cryptosystems.In:Proc of Symp on Security and Privacy. IEEE Computer Society Press, 1980, 122-134.

[10] 卢开澄，计算机密码学[M]，北京:清华大学出版社，1998, 1-229.

[11] Anderson R. Invited lecture.In : Proceedings of the 4th ACM Conference on Computer and Communications Security,Zurich,Switzerland,1997,1～7.

[12] Bellare M ,Miner S K. A forward secure digital signature scheme. In:Proc of the CRYPTO' 99. Berlin:Springer Verlag,1999,431～448.

[13] 谭作文,刘卓军.一个前向安全的强代理签名方案.信息与电子工程.2003,1(4).257～259.

[14] M.Abdalla and L.Reyzin.A New Forward-Secure Signature Scheme.In:ASIA CRYPT2000, LNCS 1976, pp Digital.116-129. Springer-Verlag,2000.

[15] G.Itkis and L.Reyzin.Forward-secure signatures with optimal signing and verifying.In: CRYPTO'01,LNCS 2139, pp.332-354.Springer-Verlag,2001.

[16] H. Krawczyk.Simple forward-secure signatures from any signature scheme.In:Proc. of the 7th

ACM Conference on Computer and Communications Security(CCS 2000), PP108-115. ACM, 2000.

[17]T. Malkin, D. Micciancio, and S. K. Miner. Efficient Generic Forward-Secure Signatures with an Unbounded Number Of TimePeriods.In:EUROCRYPT2002,LNCS2332,pp.400-417. Springer-Verlag,2002.

[18] Y.Dodis,J.Katz,S.Xu, and M. Yung. Key-Insulated Public Key Cryptosystems.In:EUROCRYPT 2002, LNCS 2332, pp.65-82. Springer-Verlag,2002.

[19] Housley R,Ford W, Polk W,Solo D,Internet X.509 Public Key Infrastructure Certificate and CRL Profile,IETF PKIX Working Group,RFC2459,Jan.1999.http://www. ietf.org/rfclrfc2459. htm1.

[20] Stubblebine S. B. Recent-secure Authentication: Enforcing Revocation in Distributed Systems. In:Proc of 1995 IEEE Symposiums on Research in Security and Privacy.Ockland,1995, 224-234.

[21] Ronald L.Rivest.Can We Eliminate Certification Revocation Lists In: Intern- ational Conference on Financial Cryptography (FC98) volume 1465.Anguilla, Springer-Verlag,1998,178-183.

[22] Ohta K, Okamoto T. In:A digital multisignature scheme based on the Fiat Shamir scheme. In : Proceedings of Advances in Cryptology ASIA Crypt 91, Berlin: Springer Verlag, 1991, 139-148.

[23] A. Fiat and A. Shamir.How to prove yourself: practical solutions of identification and signature problems.In: Crypto'86, LNCS 263, pp.186-194. Springer-Verlag,Berlin,1987.

[24] 秦波,王尚平,王晓峰,罗喜召.一种新的前向安全可证实数字签名方案.计算机研究与发展 2003, 40(7). 1016～1020.

[25] 詹雄泉,洪景新.基于椭圆曲线密码体制的一种具有前向安全的数字签名方案.厦门大学学报(自然科学版) 2005.44(2),189～192.

[26] Moni Naor, Kobbi, Nissim.Certificate revocation and certificate update.IEEE Journal on Selected Areas in Communications,2000,18(1) : 561～ 170.

[27] David A. Cooper. A Model of Certificate Revocation [A], The Proceedings of Fifteenth Annual Computer Security Application Conference[C], Phoenix, 1999. 256-264.

[28] Naor M, Nissim K.Certificate Revocation and Certificate Update. IEEE Journal on Selected Areas in Communications, 2000, 18(4): 561-570.

[29] Cooper A. Coper. A More Efficient Use of Delta-CRLs [A], The Proceedings of the 2000 IEEE Symposium on Security and Privacy[C], Berkeley, 2000. 190-202.

[30] Micali S, Silvio, Certificate Revocation System, United States Patent.5666416.1997-09-09.

[31] Paul Kocher. A Quick Introduction to Certificate Revocation Trees (CRTs) [R].Technical report, ValiCert, 1999.

[32] Merkle R. A Certified Digital Sigature.In: Proc on Advances in Cryptology.New Work: Spring-Verlag, 1989, 218-238.

[33] 王尚平,张亚玲,王育民.证书吊销的线索二叉排序 Hash 树解决方案.软件学报, 2001, 12(9):1343-1350.

[34] Myers M,Anlcney R,Malpani A,et al.X.509 Internet Public Key Infrastructure Online Certificate Status Protocol--OCSI'. RFC2560.1999.

[35] Berbecaru D, Lioy A, Marian M. Security Aspects in Standard Certificate Revocation Mechanisms: A Case Study for OCSP. In: Procof 7th International Symposium on Computers and Communications. Taormina/Giardini Naxos, 2002, 484-489.

[36] X.509:The Directory's Authentication Framework..

[37] S.Chokhani,W.Ford.Internet.X.509 Public Key Infrastructure Certificate Policy and Certification Paractices Framework.RFC2527,March 1999.

[38] Andre Prnes, Mike Just, Stein J, et al. Selecting Revocation Solutions for PKI[A). Proceedings of The Fifth Nordic Workshop on Secure IT Systems (NORDSEC 2000) . Reykjavik, Iceland, 2000.360-376.

[39] Draft Revised ITU-T Recommendation X.509.Information Technology Open Systems Interconnection.The-Directory:Public-Key and Attribute Certificate Frameworks.2000

[40] draft-ucft-ojux-dcs-00,txt Ubterbet X,509 Public Key Infrastructure Data Certification Server Protocols. PKIX Standard.

[41] P Kocher, "On Certificate Revocation and Validation," Financial Cryptography, LNCS 1465, 1998,172-177.

[42] Malpani A, Housley R, Freeman T. Internet Draft:Simple Certificate Validation Protocol (SCUP), RFC 2026, 2002.

[43] John Larmouth. ASN.1 Communication between Heterogeneous Systems.Academic Press,2001.

[44] Andrew S. Tanenbaum, Maarten van Steen, Distributed Systems: Principles and Paradigms [M].

北京:清华大学出版社(影印版),2003,245-246

[45] Grady Booch,Jams Rumbaugh,Ivar Jacobson.UML 用户指南[M].第一版.邵维忠，麻志毅，张文娟等，北京:机械工业出版社,2001. 73-186.

[46] 钱海峰,曹珍富,薛庆水.基于双线性对的新型门限代理签名方案.中国科学 E辑 信息科学 2004,34(6).711～720.

[47] 纪家慧,李大兴,王明强.来自双线性映射的新的代理多签名、多代理签名和多代理多签名体制.计算机学报2004, 27(10) .1429～1434.

[48] Adams C,Zuccherato R. A general,flexible approach to certificate revocation [EB/OL].URL: http://www.entrust.com/ resourcecenter/pdf/certrev.pdf. 2001.

[49] Micali S.Efficient Certificate Revocation [A].RSA Data Security Conference [C].San Francisco : California,1997.

[50] Heidroon Ong and Claus P. Schnorr. Fast signature generation with a Fait Shamir-like scheme. In Damgard, page 432-440

[51] RFC2459:Internet X.509 Public Key Infrastructure Certificate and crl Profile.

[52] Y.Dodis,J.Katz,S.Xu,and M.Yung.Strong Key-Insulated Signature Schemes.In: Public Key Cryptography-PKC 2003, LNCS 2567, pp.130-144.Springer-Verlag,2003.

[53]贾爱库,王彩芬,于成尊,刘军龙.基于双线性映射的一种具有前向安全的概率签名方案.计算机工程,第 33 卷第一期,2006,1(173-175).

[54] 严蔚敏,吴伟民.数据结构(C 语言版).北京:清华大学出版社,1996.118～135.

致　谢

首先感谢我的导师王彩芬教授。王老师知识渊博、治学严谨、为人和蔼，在我三年读研期间，王老师一步一步将我们领入信息安全这个让我值得奋斗终生的领域。王老师采用一些行之有效的方法，使我们在三年就读过程中，就很快地学习和了解了本学科最新的研究动态。尤其在论文的选题上，王老师凭其对密码学、信息安全知识领域的深刻理解和敏锐的洞察力给予我及时的帮助。王老师精益求精的治学态度、勤勤恳恳的工作作风给了我潜移默化的熏陶，必将使我终生受益。在论文完成之际，再次对王老师的精心指导和悉心帮助表示最衷心的感谢。

在此，我还得感谢西北师范大学文学院的龚喜平教授，龚教授在我们读研期间在学习和生活上给出许多指导和关注，使得三年学习过程顺利进行。

感谢开题报告中给予我指导的冯百明、马满福、段富海、蒋芸等各位老师，因诸位老师所提出的宝贵意见，才使得本文更加完善。

感谢甘肃省商业学校诸位领导的支持与关心，才能顺利完成学业。

同时感谢我的同窗好友刘军龙，于成尊，李海峰及其它师兄弟热情帮助，他们是我的良师益友。三年学习过程中，相互学习，互相鼓励，共同讨论一些学习内容，结下深厚的友谊。他们是我终生难忘的朋友。

最后要感谢我的家人，因为她们在生活上任劳任怨才为我营造了一个学习环境，使我无后顾之忧，能够安心顺利度过三年的学习生活。

附　录

攻读学位期间科研成果：

1. 已发表的论文

[1] 贾爱库, 王彩芬, 于成尊, 刘军龙. 基于双线性映射的一种具有前向安全的概率签名方案. 计算机工程, 第 33 卷第一期, 2007, 1.

[2] 贾续涵, 王彩芬, 于成尊, 刘军龙. 一种自根向下压缩的证书吊销树方案. 计算机工程, 拟于 2007, 10 发表.

[3] 王彩芬, 贾爱库, 刘军龙, 于成尊. 基于签密的多方认证邮件协议. 电子学报，第 33 卷第 11 期, 2005, 11.

[4] 王彩芬, 于成尊, 贾爱库, 刘军龙. 一种新的认证邮件协议. 计算机应用，2005, 7.

[5] 王彩芬, 刘军龙, 贾爱库, 于成尊. 具有前向安全性质的秘密共享方案. 电子与信息学报, 2006, 9.

[6] 于成尊, 王彩芬, 刘军龙, 贾爱库. 基于椭圆曲线的有代理的多重签名. 计算机应用与研究, 2006, 10.

2. 参与科研项目

[1] 王彩芬教授主持的甘肃省自然科学基金项目(3ZS051-A25-042)：电子商务协议的设计与分析。

[2] 王彩芬教授主持的甘肃省科技攻关项目(2GS064-AS2-035-03)：基于自认证公钥体系下的可取签名算法的研究及其实现系统。