

中南大学

硕士学位论文

关系数据库数字水印技术的研究与应用

姓名：胡斌

申请学位级别：硕士

专业：信号与信息处理

指导教师：施荣华

20070401

摘要

随着信息化程度的提高,关系数据库的版权保护问题变得十分重要。关系数据库数字水印作为信息隐藏技术研究领域的重要分支,能够在原始数据库中嵌入版权认证信息,并且不破坏数据库的使用价值或商用价值。因此,关系数据库数字水印技术在版权保护方面的应用具有较高的现实意义和理论意义。

本文收集和分析了近年来国内外数字水印方面的文献,在现有关系型数据库水印算法进行研究的基础上,提出了一种基于密钥分存思想的关系数据库水印技术,该算法通过对水印信息的分存,以及最低有效位的修改,使水印信息分布得更“广泛”,而且原始水印的恢复只须部分子水印即可,能明显提高水印的鲁棒性。接着提出了一种使用图像作为水印的关系数据库水印算法。在算法中引入了三元组的思想,在一个字段值中就可嵌入 3 个 bit,极大地扩大了水印信息的容量。为了进一步拓展研究应用领域,本文设计了一种基于非数值型字段的关系数据库水印算法。该算法最小化了在嵌入水印时需修改的数据量,并且允许数据的拥有者自己定义一个相似函数选择元素进行水印嵌入。

最后,本文设计开发了水印系统软件,将本文提出的算法应用于水印系统中,实现了关系数据库中数字水印信息的嵌入和提取。

关键字 版权保护, 数字水印, 数据库水印, 密钥分存, 认证

Abstract

With the development of information technology, protection of copyright for relational database has been more and more important. As an important branch of information hiding technology, watermarking based on relational database can embed copyright information into the original database, without destroying the usability of database. Therefore, studying on the watermarking based on relational database has great significance in terms of both theory and application.

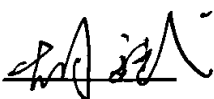
On the basis of researching and improving the current watermarking algorithms of relational database, a new algorithm for relational database based on Blakley secret sharing scheme is proposed in this paper. By secret sharing, the watermarking information is distributed more far-flung, and only part content of watermarking is needed when recovering the watermarking. This algorithm can improve the robustness. The paper presents a database watermarking method by using image. In this method, three-bit group idea is introduced, which can imbed 3 bit in every element's value. In order to broaden the application fields, a new watermarking method for protecting non-numerical databases is designed in this paper. The proposed watermarking system allows the data owner to define a similarity function in order to reduce the distortion caused by watermark embedding and, at the same time, reduce the number of element modifications needed by the embedding process.

Finally, this paper develops software of watermarking system, applies the algorithm carried out in this paper into watermark system and realizes the embedding and recovering of watermark information in relational database.

KEY WORDS Copyright protection, Digital watermarking, Database watermarking, Secret sharing, Authentication

原创性声明

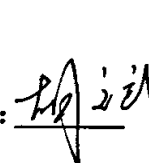

本人声明，所呈交的学位论文是本人在导师指导下进行的研究工作及取得的研究成果。尽我所知，除了论文中特别加以标注和致谢的地方外，论文中不包含其他人已经发表或撰写过的研究成果，也不包含为获得中南大学或其他单位的学位或证书而使用过的材料。与我共同工作的同志对本研究所作的贡献均已在在论文中作了明确的说明。

作者签名： 

日期： 2007 年 5 月 28 日

关于学位论文使用授权说明

本人了解中南大学有关保留、使用学位论文的规定，即：学校有权保留学位论文，允许学位论文被查阅和借阅；学校可以公布学位论文的全部或部分内容，可以采用复印、缩印或其它手段保存学位论文；学校可根据国家或湖南省有关部门规定送交学位论文。

作者签名：  导师签名：  日期： 2007 年 5 月 28 日

第一章 绪论

1.1 概述

20世纪90年代以来,多媒体数据的数字化为多媒体信息的存取提供了极大的便利,同时也极大地提高了信息表达的效率和准确性。随着网络的日益普及,多媒体信息的交流达到了前所未有的深度和广度,其发布形式也更加丰富。人们可以通过Internet网发布自己的作品,传递重要信息,进行网络贸易等。但是其暴露出的问题也更明显:作品侵权更加容易,篡改更加方便。如何既充分利用Internet网的便利,又能有效地保护知识产权,已受到人们的高度重视。

利用传统的基于私有或公共密钥的密码学(Cryptography)理论所开发出来的加解密系统可以用来进行数据访问控制,加密后的产品是可以访问的,但只有那些具有正确密钥的人才能解密。但是首先其在使用上不具有方便性,而且数据被解密后,不再受到密码的保护,也不能制止任意复制、修改,因而无法进行版权的认证和盗版的制止^[1-2]。

数字签名(Signature)技术已经用于检验短数字信息的真实可靠性,但其需要在原始数据中加入大量的签名,因此这种数字签名在数字图像、视频、音频中的应用,既不方便也不实际。

一般的信息隐藏技术^[3]主要采用密钥(用户控制),通过嵌入算法将秘密信息隐藏于公开信息中,然后通过公开信息的传输来传递秘密信息。信息隐藏技术对稳健性的要求可以相对降低,允许隐藏信息的丢失,这对有意义的版权保护,是不能成立的^[1-2]。

数字水印^[4-5](digital watermarking)作为信息隐藏技术研究领域的重要分支,是实现数字产品版权保护的有效办法。它通过在原始数据中嵌入版权认证信息—水印(watermark),水印通常是不可见或不可察觉,它与原始数据(如图像、音频、视频数据)紧密结合并隐藏其中,既不破坏其原始品质而且不会被察觉,并可以经历一些不破坏源数据使用价值或商用价值的操作而存活下来。从而成为近几年来著作权保护的热点问题。

本课题基于国家自然科学基金《计算机网络分布式资源认证存取控制问题的研究》(编号:60173041)。

1.2 数字水印的研究背景

信息全球化的飞速发展,但是随之而来的副作用是通过网络传输的数字产品很容易被恶意的个人或团体在未经许可的情况下非法使用,因而网络世界中数字产品版权保护问题成为人们亟待解决的问题。数字水印技术就是在这种情况下产生的^[4]。

1990年, Tanaka 等提出了在图像中加入秘密信息以确认图像所有权的想法。Trikel 等在 1993 年首次明确地提出了“水印”这一概念,并描述了利用数字水印实现版权保护的一种方法。数字水印,简单地说,就是包含于原始数字数据(包括文档、图像、声音、视频等)中的不可消除的标识信息。正逐渐成为信息与网络时代实现数字产品版权保护的希望,成为今学术界的一个研究热点。

1.3 研究的目标与应用前景

目的:对用于数据库的数字水印技术进行研究,并将研究成果应用于软件系统之中,同时提出新的水印嵌入和检测算法。并且对基于大型关系型数据库(例如: Oracle 9i),进行优化,实现快速的水印嵌入和检测。

意义:数字水印技术的发展虽然只有短短十几年的时间,国际上却已有许多家公司在研制自己的数字水印产品。我国在该领域的研究这些年也从跟踪逐步转向自主研究,许多大学和研究所等科研机构纷纷致力于水印技术的研究,并在国内也开办了生产水印产品的公司,但水印在我国还是刚刚起步的一个新领域,特别是数据库数字水印,还有很多不完善的地方。

关系数据库已经广泛应用于社会的各个行业,如那些提供信息服务(如气象信息、医疗信息、人才市场信息、股票交易信息、电子元器件参数信息等等)的公司,还有国际的大公司(Oracle 公司的 Oracle9、IBM 公司的 DB2、微软的 SQL Server 等)。因此,研究关系数据库水印以实现关系数据库的版权保护有着非常重要的学术意义和应用价值。

1.4 本文所作的工作

本文研究的目标是对基于关系数据库的数字水印技术的研究,并将研究成果应用于关系数据库的水印系统之中,实现数据库的版权保护。

本文对首先对基于数值型属性的关系数据库数字水印算法进行了分析与研究,对现有的算法进行了改进,并且提出了一种基于密钥分存的关系数据库数字

水印算法和一种基于图像的关系数据库水印算法，通过试验仿真和算法分析表明，这两种算法能很好的解决在关系数据库中嵌入水印信息的要求。

目前的关系数据库数字水印算法的研究基本上都集中于对数字型字段的嵌入研究，而对非数值型字段的嵌入算法研究比较少。为了能在数据库中嵌入更多的信息，扩大水印容量，本文尝试着设计了一种基于非数值型字段的数据库水印算法。

本文最后给出了一个数据库数字水印系统的实现方案，并完成了相应的测试分析。

1.5 论文的结构

论文的编排结构如下：

第一章主要介绍了数字水印的基本概念，并对数字水印技术的研究进展和现状进行了总结与回顾。

第二章介绍了数字水印技术的基本原理，基本概念，通用的评价指标，并总结了现有的数字水印算法。

第三章主要讨论介绍了基于数值型字段的数据库公开水印算法研究，提出了一种基于密钥分存思想的算法和一种基于图像的数据库数字水印算法。

第四章介绍了基于非数值型字段的数据库水印算法研究。

第五章介绍了水印系统的设计实现及其性能评测。

最后，对论文做了总结，并指出了今后的努力方向。

第二章 数据库数字水印技术研究现状

2.1 引言

数字水印概念是1990年首次提出的,1993年首次提出数字水印算法,人们对数字水印技术的研究兴趣不断高涨,出现了大量的水印算法和实施技巧。水印的应用也从最初的对媒体数据的版权保护推广到数据认证、访问控制等更广泛的领域。本章将结合近几年的关于水印理论和应用的最新研究成果,对数字水印技术的概念、基本原理、分类、应用和研究内容做全面的分析。

2.2 数字水印的分类

数字水印的分类方法^[1~4]有很多种,分类的出发点不同导致了分类的不同,它们之间既有联系又有区别,有的分类方法还直接反映了水印嵌入算法的不同。目前常见的分类方法有如下几种:

(1) 按水印所用不同的媒体载体分类

根据数字水印所附载体的媒体不同,可以将数字水印划分为图像水印^[7~11]、音频水印^[12]、视频水印^[13]、文本水印^[14~15]、用于三维网格模型的网格水印^[16]、软件水印^[17]及数据库水印^[18~25]等。随着数字技术和多媒体技术的发展,将会有更多种类的数字媒体出现,同时相应地也会产生更多新的数字水印技术。

(2) 按水印的主观形式分类

从数字水印的主观形式的角度出发,可将数字水印分为可见数字水印和不可见数字水印两种。更准确地说应该是可觉察数字水印和不可觉察数字水印。可觉察数字水印嵌入到媒体后会在媒体中留下明显的印记,主要用于标识版权,防止非法使用,虽然降低了资料的商业价值,却不妨碍使用者的使用,如电视台的台标等。不可觉察性数字水印嵌入到数字作品中,人的感观不能明显地觉察,不影响作品的质量,具有较高的使用价值^[26]。

(3) 按水印的抗攻击能力分类^[27]

按水印的抗攻击能力,可以将数字水印分为鲁棒数字水印、半脆弱数字水印和脆弱数字水印。鲁棒数字水印主要用于解决在数字作品中标识著作权信息问题,是目前研究比较多的一种水印技术,它要求嵌入的水印能够有效抵抗各种有意或无意的攻击,脆弱水印主要用于完整性保护,要求对信号的改动敏感,人们根据脆弱水印的状态可以判断数据是否被篡改过,半脆弱数字水印是具有这两

者的一些特点。

(4) 按数字水印的嵌入位置分类^[16]

按数字水印的嵌入位置分类法, 可以将水印技术划分为时/空域数字水印^[28]和频域数字水印^[29-32]。

时/空域数字水印: 时/空域数字水印主要是通过直接修改媒体数据采样值的强度实现水印嵌入的。这种方法无需对原始媒体进行变换, 计算复杂度低, 实施效率高, 有较好的不可感知性, 但由于可修改的属性范围较小, 生成的水印具有局部性, 因而鲁棒性较差。

频域数字水印: 也叫变换域数字水印, 这类算法先对原媒体进行某种形式的正交变换, 在变换得到的系数上嵌入水印, 再经过相应的逆变换得到含水印的媒体。常用的变换包括离散傅立叶变换(DFT)、离散余弦变换(DCT)、离散小波变换(DWT)等。由于变换后的图像具有能量分布集中和良好的分频特性等优点, 易于和人类视听觉的感知模型^[33] (human visual/ auditory system, HVS/HAS) 相适应, 因而可以方便地调节水印的不可见性和鲁棒性的平衡。此外, 由于流行的压缩标准中的核心算法都是在频域中进行的, 因而对频域水印的研究具有更加突出的理论意义和应用价值。

(5) 按数字水印的内容分类

按数字水印的内容可以将数字水印划分为有意义水印和无意义水印。有意义水印是指水印本身也是某个数字图像(如商标图像)或数字音视频片段的编码。无意义水印则只对应于一个序列号。有意义的水印的优势在于: 当媒体水印化信息受到攻击或其他原因致使解码后的水印破损时, 人们仍然可以通过视觉观察确认是否含有水印。但对于无意义水印来说, 如果解码后的水印序列有若干码有错误, 则只能通过统计决策的方法来确定信号中是否含有水印。

(6) 按水印的检测提取过程分类

按照数字水印的检测提取过程是否需要原始媒体信息, 可以将数字水印划分为有源检测水印、无源检测水印和半源检测水印。

无源检测水印: 也叫盲检测水印。水印的检测和提取由含水印的待测媒体本身确定, 而不需要原始媒体的参与。这种水印的检测可以在任何拥有检测环境的平台上进行, 使用范围较广。但此类算法常常选取数据的固有特征进行水印的嵌入和检测, 在数据固有特征被破坏时, 水印检测较为困难, 生成水印的鲁棒性不高。

有源检测水印: 也叫非盲检测水印。水印的检测和提取是在分析原始媒体数据与含水印媒体数据差别的基础上进行的, 检测和提取过程必须在原媒体的参与下完成。这类水印技术可嵌入水印的位置选择范围较大, 可以充分考虑到水印的

鲁棒性和不可见性,生成水印的鲁棒性较好。但由于水印检测和提取必须提供原媒体,因而一定程度地限制了它的应用。

半源检测水印:也叫半盲检测水印。水印的检测无需原始媒体数据,但是需要某些与原始媒体数据有关的信息,这些信息可能是原始数据嵌入水印时的某些参量,也可能是表征原始数据某些特征的信息。

(7) 按照水印的用途分类

不同的应用造就了不同的水印技术。按水印的用途,可以将水印划分为版权保护水印、认证水印^[34]和访问控制水印^[35-36]等。

版权保护水印:版权水印是目前研究最多的一类水印,版权保护水印要求水印具有较好的隐蔽性和鲁棒性。

认证水印:认证水印是一种脆弱水印,其目的是标识宿主信号的完整性和真实性。

访问控制水印:访问控制水印是在媒体中通过嵌入水印,引入不同级别的扰动。访问时,必须先提出水印,恢复扰动才能获得不失真的媒体。

2.3 数字水印技术的主要应用

数字水印技术主要用于多媒体版权保护和隐蔽通信两个领域,具体说来,有以下几个方面应用^[1-4]:

(1) 媒体所有权的版权认证和保护^[37]

目前,版权保护是数字水印最主要的应用。由于数字作品的拷贝、修改非常容易,而且可以做到与原作品完全一致,所以如何对数字作品(如电脑美术、扫描图像、数字音乐、视频、三维动画)的版权进行保护一直是一个难点问题。数字水印技术通过在被保护的作品中嵌入数据的来源信息以及有代表性的版权信息,从而可以有效地防止其他团体宣称对该作品拥有版权。媒体创作者可以在媒体传播前嵌入水印,用于识别合法用户信息,对媒体的传播进行跟踪。例如,如果在数字照相机中实现水印技术,就可以使所拍摄的照片带上摄影师的信息。

(2) 防止非法拷贝

在多媒体发行体系中,建立禁止未授权的媒体拷贝的拷贝保护机制非常重要。在这样的系统中,可以用数字水印来说明数据的拷贝情况。例如,在媒体的录/放设备的设计中应用数字水印技术,当录放设备工作时,检测媒体是否带有水印,以决定该媒体应不应该被录/放,从而拒绝非法拷贝媒体的流行和使用。同样的原理可应用于广播、电视、计算机网络在线多媒体服务中的听、看和访问权限的控制,如在广播、电视接收机和计算机网络的浏览器中应用数字水印技术,可以控制音、视频媒体的听、看权限。

(3) 盗版跟踪

与软件产品的序列号类似,数字水印可用于监控和跟踪流通数据的非法拷贝。主要用来识别数据的单个发行拷贝。在发行的每个拷贝中嵌入不同的水印,通常称之为“数字指纹”。该应用要求水印算法易于提取,复杂度低。例如对于WWW应用,有专门的Web搜索者寻找已嵌入数字水印的盗版图像。

(4) 基于内容的真伪鉴别

随着高质量图像输入输出设备的发展,特别是高精度的彩色打印机和复印机的出现,使得货币、支票以及其他票据的伪造变得更加容易。另一方面,在从传统商务向电子商务转换过程中,会出现大量过度性的电子文件,如各种纸质票据的扫描图像等。数字水印技术可以用于鉴别支票、合同等重要文档的来源的真实性、内容的真实可靠性等。在真伪鉴定应用中,使用数字水印的目的是对数据的修改进行检测,该技术可利用所谓的“脆弱性水印”来实现。数字水印还可以为各种票据提供不可见的认证标志,从而大大增加了伪造的难度。

(5) 隐蔽通信及其对抗

网络情报战是信息战的重要组成部分,其核心内容是利用公用网络进行保密数据传送。迄今为止,学术界在这方面的研究思路一直未能突破“文件加密”的思维模式。然而,经过加密的文件往往是混乱无序的,容易引起攻击者的注意。数字水印所依赖的信息隐藏技术不仅提供了非密文的安全途径,更引发了信息战,尤其是网络情报战的革命,产生了一系列新颖的作战方式,使得利用网络进行保密通信有了新的思路,利用数字化音视频信号相对于人的视觉、听觉冗余,可以进行各种时(空)域和变换域的信息隐藏,从而实现隐蔽通信。

(6) 多语言电影系统和电影分级

利用图像数字水印技术,可以把电影的多种语言配音和字幕嵌入到视频图像中携带,在保证图像视觉质量不受影响的情况下节省了声音的传输信道。与此类似,把电影分级信息嵌入到视频图像中,可以实现画面放映的控制从而实现电影的分级播放。

(7) 数字媒体附加描述和参考信息的携带

可以把感兴趣的图像特征(或区域)的位置和识别信息直接嵌入到图像中,实现特征的定位和识别。

2.4 数字水印的研究内容

数字水印是近年来多媒体信号处理领域兴起的一个新的研究方向,它综合了信息论、密码学、通信原理、信号处理和模式识别等领域的理论和技术。它的研究内容主要有以下几个方面:

(1) 水印基本理论的研究

包括三个部分:

a) 水印的产生:即水印信号的设计与生成,应根据实际应用的需要选择不同的水印信息,包括文本,图像与语音、视频等信息。

b) 水印的嵌入算法:包括三个方面:水印的嵌入位置,即把水印嵌入到原始数据的什么位置上。水印的嵌入强度,即应该嵌入多强的水印以实现水印不可见性和鲁棒性的折中。水印的嵌入方法,即应采用何种数学或逻辑模型来嵌入水印。

c) 水印的提取与检测算法:根据水印嵌入的位置、强度和方法的不同,采用相应的算法将水印从数字媒体中提取出来,作为版权信息的验证或作其它用途。

(2) 依赖与视觉或听觉系统的自适应水印算法的研究

充分利用人类器官系统的冗余特性自适应地嵌入水印信号,以期在一定程度上,既能增强水印的鲁棒性,又能确保水印的不可见性。

(3) 数字水印攻击和抗攻击能力的研究

主要研究对水印实施攻击以消除水印以及水印抗攻击的能力。即希望通过对水印攻击的研究发现和改善现有水印算法的弱点,提高其性能。

(4) 数字水印的评测基准和系统的研究

即研究评测水印性能的有关参数指标和系统平台。

(5) 数字水印技术的应用研究

研究如何在工业中运用水印技术来进行版权保护或隐藏通信等应用,进一步完善水印对数据安全的保护机制,开拓更广泛的应用领域。

2.5 数字水印技术的特征与通用模型

2.5.1 数字水印技术的特征

数字水印技术是信息隐藏技术的一个应用分支,虽然国内外有很多文章讨论过数字水印方面的问题,但对数字水印及其技术始终没有一个明确统一的定义。通过综合一些学者提出的定义^[1-4]和分析已有的数字水印方案,给出如下的关于数字水印和数字水印技术的定义。数字水印是指具有不可见性、鲁棒性、抗检测性的,含有认证敏感信息的数字标记,诸如数字签名、日期、商标或随机序列等。数字水印技术是指利用信号处理的方法,在数字化的多媒体数据(主要包括数字图像、音频和视频)中嵌入数字水印,并且在需要时可以提取出水印或能够证明水印的存在性的一种数字编码技术。为了更好地实现数字媒体的真伪验证、安全存储、保密传输等目的,一般认为在数字媒体中嵌入的数字水印应具有如下的特性:

(1) 鲁棒性

也称作稳定性或抗攻击性,是指数字水印信号在经历多种无意或有意的信号处理或攻击后,仍能保持其完整性或仍能被准确鉴别的特性。可能的信号处理和攻击包括信道噪声滤波、数/模与模/数转换、重采样、剪切、位移、尺度变化以及有损压缩编码等。

(2) 不可见性

又称为知觉透明性,数字水印的嵌入不应引起原数字作品的视觉/听觉质量下降,即不向原始载体数据中引入任何可察觉的附加数据。这样的数字水印将难以用统计的方法发现和篡改。

(3) 安全性

水印的嵌入过程(嵌入方法和水印结构)应该是秘密的,嵌入的数字水印应该是统计上不可检测的,非授权用户无法检测和破坏水印。对于通过改变水印图像(嵌有水印的图像)来消除和破坏水印的企图,水印应该能一直保持存在,直到水印图像已严重失真而丧失使用价值。

(4) 低复杂性

数字水印的嵌入和提取算法应该比较容易实现,计算复杂度较低,在某些应用场合(如视频水印)下,甚至要求水印算法的实现满足实时要求。

(5) 可证明性

数字水印所携带的信息应该能够被唯一地、确定地鉴别,从而能够为已经受到版权保护的数字产品提供完全的和可靠的所有权归属证明的证据。数字水印算法应该能够正确识别出被嵌入到保护对象中的有关信息,例如经过注册的用户的编码、产品的标识或者其它任何有意义的文字图像等,并且能在需要时将其提取出来作为证据。

(6) 水印的容量和负荷

这两个术语目前尚无一致的明确定义,水印容量一般指可以在载体中嵌入的信息量,而负荷则表示在一定量的载体数据中可以嵌入的信息量。水印的容量或负荷和水印的鲁棒性、不可见性之间相互影响。水印算法应该根据具体情况在三者间作出折衷。

2.5.2 通用模型

数字水印算法各异,但原理基本相同,主要包括数字水印的嵌入和数字水印的提取两个部分。

数字水印的嵌入过程就是将数字水印信号叠加或自适应地叠加到图像的灰度(亮度)或者色彩上,其过程可以发生在空间域或者变换域上,具体过程如图2-1

所示。

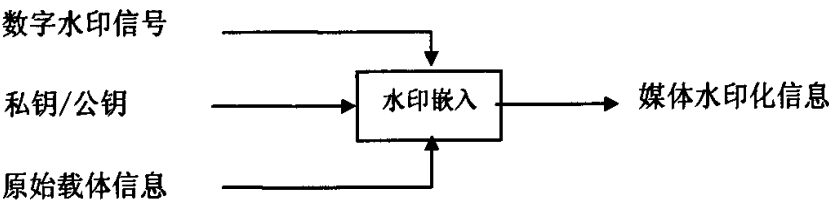


图2-1 水印嵌入

系统的输入是数字水印、被保护的原始载体数据和一个可选的公钥或私钥^[38]。水印信号可以是数字序列、数字标识、文本或图像等信息，其生成和选择一般有两类方法，一类是独立的与原始载体数据无关的水印标记，另一类则是依赖于原始载体。

数据内容的水印，比如在图像数字水印中通过计算一副图像的散列(Hash)值来生成水印。数字水印算法通常都要与加密/解密算法相结合，密钥可以用来加强安全性，利用密钥可禁止水印的非法提取，避免未授权方恢复和修改水印。水印嵌入系统可以使用一个密钥，也可以使用几个密钥的组合。这样，即使非授权用户能够提取出水印，但是在没有密钥的情况下，也无法读出水印信息，从而可以为原始载体提供双层的保护。当水印算法与私钥或公钥结合时，嵌入水印的技术通常分别称为秘密水印技术和公开水印技术。水印嵌入系统的输出是加载了水印信息的水印化信息。图2-2描述的是数字水印的提取和检测过程。

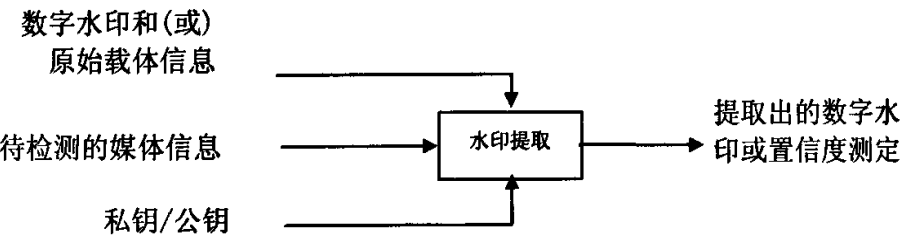


图2-2 数字水印的提取和检测过程

水印检测和提取系统的输入是待检测媒体信息、私钥或公钥，以及原始水印和(或)原始载体信息。输出的是提取出的水印信号，或者某种可信度值，它表明了所考察的待检测媒体信息中存在给定水印信号的可能性。通常利用信号的相关性实现水印的提取，如相关接收器或匹配过滤器等。在水印验证过程中，通过计算检测到的水印信号与已知水印信号的相似度或相关性，可以判断待测媒体信息中是否含有已知的水印信号。

2.6 关系数据库数字水印的特征

2.6.1 关系数据库数据与多媒体数据的差异

为了更好地探讨关系数据库数字水印技术,首先来分析一下关系数据库数据与多媒体数据之间的不同^[39-40],这主要有:

(1)多媒体数据对象是由大量的位组成的,并且其中存在较多的冗余位。而关系数据库是由许多独立的元组组成(当然每个元组是唯一的)。

(2)多媒体数据对象各个点的相对空间位置是不可变的,而关系数据库中的元组和属性值集合组成的,它们之间没有一定的顺序,并且元组之间也是无序的。

(3)多媒体数据对象某个部分的删除或替换,很容易被知觉所察觉。而关系数据库却可以简单地去掉一些元组或者用其他类似的关系数据中的元组来代替而不易被发觉。

(4)多媒体数据一般不能更新,而关系数据经常要维护更新,基于这些不同点,关系数据库数字水印技术有很大的难度,这样多媒体数据的数字水印技术就不可能直接用于关系数据库。但可以参考借鉴多媒体数字水印技术原理和思想,研究适于关系数据库中关系数据的数字水印技术。

2.6.2 关系数据库数字水印特性

在关系数据库中加水印应该充分考虑到关系数据库自身的特殊性以及各种攻击方式。基本要求就是水印算法要在鲁棒性、不可见性之间折衷考虑。

(1)鲁棒性要好,加入的水印就越能够经受住各种恶意的攻击和善意的数据更新;

(2)不可见性要高,不可见性越高说明加入的水印方法越好,水印越不被用户察觉,越不会因为加了水印而影响关系数据的使用;

(3)嵌入的信息要多,加入的信息量越多水印就越不容易被破坏,误判的概率越小;

(4)误判率要小,误判有两种,一种是有水印时不能够检测提取出来,另一种是无水印时却检测到水印存在。

(5)盲测:由于数据库数据需要经常更新维护,关系数据库的数字水印技术最好能够在水印提取时实现盲测,即水印检测既不需要原始数据库数据(加水印前数据)也不需要水印。该特征关键是在关系数据库数据拷贝中检测到水印,而不用考虑原始关系数据库的更新。

2.7 国内外数据库数字水印研究动态

数字水印技术^[33]是网络环境下保护版权的一种新型技术,可以确立版权所有者、识别购买者或提供关于数字内容的其他附加信息。数字水印技术通常是在满足知觉效果的情况下,将标识信息秘密地嵌入载体数据中,用于确认版权等,数字水印作为一种新兴的信息安全技术主要集中在多媒体领域。

关系型数据库^[40]是以二维表作为数据模型的数据库系统。关系数据库技术出现于 20 世纪 70 年代,经过 80 年代的发展,到 90 年代已经比较成熟。随着关系型数据库的广泛使用,数据库版权保护的问题也越来越受到人们的重视。在因特网上允许指定用户远程查询和访问数据库已成为普遍要求,但数据容易被窃取和非法拷贝,版权得不到有效保护。因此,迫切需要一种有效的数据库数据版权保护技术。

目前的水印算法主要是利用一定失真范围内的数据变形来嵌入水印。IBM Almaden研究中心的Rakesh. Agrawal等在这方面作了开创性的研究。他们于2002年首次进行了向关系数据库嵌入比特位模式的实验。该实验利用数据库关系中数值型元组存在的冗余空间,通过在某些数值型属性值中引入少量误差,对其最低有效位(Least Significant Bits, LSB)进行位操作,实现水印信息的嵌入。实验针对一个只包含数值型数据的数据库,首先选用单向Hash函数,根据用户给定的密钥和元组主键值以及需要标记的元组比例来确定哪些元组需要标记,然后根据可以标记的属性数和比特位数确定标记的属性及其比特位位置。在整个关系数据库中许多个比特位标记组合的比特位模式就是嵌入的水印信息。

这类方法采用基本的LSB嵌入算法,易于实现,但难以嵌入有实际意义的水印信息。

2.7.1 关系数据库的几种数字水印技术

(1) R. Agrawal 的关系数据库数字水印技术

R. Agrawal 提出了对关系数据库中数值型属性值进行标记的策略^[24]。其基本思想是:首先假定可以标记的关系数据库的某些数值性属性的属性值允许一定的误差,在其误差范围内不影响关系数据库数据的具体使用。标记策略的基本思想是:首先运用加密中的单项哈希函数,如MD5 或SHA 函数,根据用户给定的密钥和元组的主键值以及需要标记的元组比例来确定哪些元组需要标记,然后根据可以标记的属性数和比特位数确定标记的属性及其比特位置。这样就将关系数据库中符合条件的某些元组的某些数值型属性值的比特位数值置1或0,作为一个标记。这样,在整个关系数据库中许多个比特位标记组合的比特位模式就是嵌入的

水印信息。而需要加标记的元组、元组的属性、属性的比特位置以及具体的比特值都是由密钥、元组主键值和需要标记的元组比例控制的算法来决定的,这里密钥、元组标记比例、可标记属性数和比特位数只有关系数据库的所有者才知道。只有知道了密钥,才可能检测出所加的水印(并且假定攻击者没有修改元组的主键值)。

(2) R. Sion 关系数据库数字水印技术

R. Sion 等人进一步对关系数据库水印的研究,提出对数值型属性进行标记的策略^[22]。其基本思想是:给定数值型项目集合 $S = \{s_1, s_{i+1}, \dots, s_n\} \subset R$ 和一个秘密的排序密钥 K ,首先根据标准化项目的最大意义比特位的加密键值哈希对其进行秘密排序,如 $index(s_i) = H(K, msb(norm(s_i)), K)$ 。然后构造子集 S_i 用来嵌入比特位水印标记。假定水印信息是 m 个比特位长,则整个水印带宽将是 m 个比特位,每个比特位嵌入/隐藏到每个标记的 S_i 中这种秘密排序,通过对数据的分散效果,提高了防止像“选取”或“增加”等不同类型攻击的能力。这样,如果攻击者去除项目的5%就将导致每个子集 S_i 也将变小5%。如果 S_i 足够小或者初始水印方法可以将水印标记编码到 S_i 中,这就会使得最小的变换有一定的鲁棒性。将水印比特位值编码到 S_i 中的比特位编码过程如下:

设 $V_{false}, V_{true} \in (1, 0), V_{false} < V_{true}$ 是实数(如 $c = 90\%, V_{false} = 7\%, V_{true} = 10\%$)。称 c 是一个置信因子(水印检测提取时也用到这些参数值)。设

$$avg(S_i) = \frac{\sum x_j}{|S_i|}, \delta(S_i) = \sqrt{\frac{\sum (avg(S_i) - x_j)^2}{|S_i|}} \quad \forall x_j \in S_i, v_c(S_i) \quad \text{为 } S_i \text{ 中大于}$$

$avg(S_i) + c\delta(S_i)$ 的项目数。则有,如果 $v_c(S_i) > (v_{true} \times |S_i|)$, 则

$mark(S_i) \in \{true, false, invalid\}$ 是 true, 如果 $v_c(S_i) < (v_{false} \times |S_i|)$, 则

$mark(S_i) \in \{true, false, invalid\}$ 是 false, 如果 $v_c(S_i) < (v_{false} \times |S_i|, v_{true} \times |S_i|)$, 则

$mark(S_i) \in \{true, false, invalid\}$ 是 invalid。

假定对每个子集 S_i 嵌入一个比特位,就需要确定选择的子集大小即 $|S_i|$ (子集大小直接决定整个标记编码带宽),这样可用标记带宽即就是 $|S|/|S_i|$ 个比特位。通常 S 要足够大,以便将原始水印多次嵌入,这样可以防止子集剪切攻击(也就是在一个剪切的数据子集中保存标记)。嵌入的水印不超过 $|S|/(|S_i| \times m)$ 次。

水印检测期内,从给定的数据中恢复所有水印备份后,在所有恢复的水印比特位上用多数选举方式配置,从而确定对可能的原始水印比特位。关系数据库另一个好处就是在秘密选择过程中能用实际关系键,而代替提出的数据最大意义比特位(也就是水印属性数据)。攻击者不可能完全改变数据库方案并替换键属性。

(3) 牛夏牧关系数据库数字水印技术

牛夏牧等人对关系数据库数字水印进一步研究加入小量有实际意义的水印的技术^[18],其基本思想是:首先利用现有的关系型数据库水印算法,在所选属性

值的特定比特位中嵌入一种匹配关系,即使得该属性值编号的 Hash 值(奇/偶)与其某一最低有效位的值(1/0)相匹配。以上水印算法是一种验证算法,验证的结果只能是“有”或“无”的信息,而不是有一定具体意义的水印信息。显然,利用这种信息证明数据库的版权所有,并不是十分令人信服的。如果能在数据库中嵌入一些有实际意义的比特值(比如代表公司名称的字符串),水印的使用价值会更高。实际上,可以把验证算法的结果看成 1 位比特信息,如果有水印,结果为“1”,没有水印,结果则为“0”,也就是说,水印嵌入算法相当于在数据库关系中嵌入了一个“1”。考虑到原算法只利用了数据库关系中的一个很小的子集(编号可以被 λ 整除的元组的集合),如果能够利用数据库关系的 m 个子集($m < \lambda$),每个子集中嵌入 1 位比特信息,便可嵌入一个长度为 m 的比特串。首先对各个可嵌水印的属性值进行编号,然后利用编号模除参数 λ 的结果对各个属性值进行分组。对于属性值 A_i ,编号为 $index(A_i)$,设 $j = index(A_i) \bmod \lambda + 1$,分组的结果使得 A_i 位于子集 S_j 中。设水印为 w 。长度为 m ($m < \lambda$) m ,水印的每一位为 w_k ($1 \leq k \leq m$),依据 w 是否为“1”决定是否在数据库关系的第 k 个子集中嵌入水印。也就是说,如果 $w_k = 1$,则按照以上所描述的水印嵌入算法在子集 S_k 中嵌入水印(即在 S_k 的各个属性值中嵌入预定的匹配关系)。

(4) Zhang 关系数据库数字水印技术

Zhang 研究一种新型的关系数据库水印^[41],其基本思想就是把一幅图像嵌入到关系数据库中水印数据库 R 表示为 $R(K, A_1, A_2, \dots, A_n)$,其中 K 为主关键字,它是不能改变的,图像的像素值为 $I(V_0, V_1, \dots, V_m)$, m 远小于 n 。 $r.A_i$ 表示属性值。关系数据库分成与图像大小的相等的许多块也就是一个属性值对一个像素点。每一块都作为是一个水印块。图像的每个像素值 V_i 嵌入到对应的数据库中的属性值中。具体实现为:

```

For each tuple  $r \in R$  do
  If  $V_i = 255$  then
    Mark( $r.A_i \bmod 3$ ) = 1;
  Else if  $V_i = 0$  then
    Mark( $r.A_i \bmod 3$ ) = 2;
  elseif ( $V_i \neq 0$  and  $V_i \neq 255$ ) then
    Mark( $r.A_i \bmod 3$ ) = 0;
     $r.A_i = \text{int}(r.A_i) + \text{unitary}(V_i)$ ;
  end
end

```

按照以上所描述的水印嵌入算法在数据库中嵌入水印。

2.7.2 关系数据库数字水印的攻击

各种数字媒体的数字水印技术都应具有一定的鲁棒性,关系数据库数字水印技术也不例外,也应该可以防御各种各样的攻击,包括正常的数据库更新和恶意的攻击。因为关系数据库数据需要经常维护更新,所以包含在一个关系数据库中的标记不能因为正常的数据库更新而在无意中被去除掉,否则其效果就如同恶意攻击一样。即如果数据窃取者不知道关系数据加了水印,那么在对偷来的关系数据进行正常更新时不能因数据的更新而丢失嵌入在关系数据库里的水印信息。如果数据窃取者知道偷来的关系数据加了水印,就会试图擦掉水印或者用其他方法声明对关系数据的伪所有权^[31]。水印系统应该能保护数据的原始所有权而阻止数据窃取者各种形式的恶意攻击。对水印关系数据库常见的恶意攻击方式有^[17, 20]:

子集选取:数据窃取者不使用水印关系库的全部属性和元组,仅仅使用其属性或元组的子集,从而希望擦除水印。

子集增加:增加一些属性和元组水印关系数据库中,从而希望破坏水印。

子集更新:更新水印关系数据库中的属性值,从而希望破坏水印。

混合和匹配攻击:数据窃取者从包含相似信息的多个关系库中获取不相连的元组创建自己的关系库。

添加攻击:数据窃取者在窃取来的已经加水印的关系数据库上再简单的加上他自己的水印,并声明自己对关系数据库的所有权。

可逆性攻击:如果数据窃取者在其偷来的关系数据库内发现了一个虚幻的水印,就可以采取可逆性攻击,声称自己对关系数据库的所有权。而实际上数据窃取者声称的水印只是随机出现的水印。当然,针对关系数据库不同的水印算法,对其攻击方式还很多并且各不相同,每个水印算法都应能够防御各种攻击,都应具有很强的鲁棒性。

2.8 小结

本章分析了国内外关系数据库水印工作者研究现状和数据库数据和多媒体载体数据各有特点,总结出关系数据库水印所具有的特性。

第三章 基于数值型属性的数据库数字水印技术的研究

本章针对关系数据库的特点,提出了两种数字水印算法,通过在关系数据库的数值型字段中嵌入版权信息(文本,图像信息),实现了关系型数据库的版权保护。

3.1 引言

数字水印技术^[42-43]是实现版权保护的一种有效手段。目前,基于多媒体的数字水印技术研究已经取得了很好的研究成果。由于关系数据库的特殊性,基于关系数据库的数字水印技术研究进展却十分缓慢。在保证不可见性的前提下,如何提高数据库水印算法的鲁棒性,如何拓展关系数据库的冗余空间,是数据库数字水印技术研究发展的重要方向。

目前的水印算法主要是利用一定失真范围内的数据变形来嵌入水印。IBM Almaden研究中心的Rakesh. Agrawal等在这方面作了开创性的研究。他们于2002年首次进行了向关系数据库嵌入比特位模式的实验。该实验利用数据库关系中数值型元组存在的冗余空间,通过在某些数值型属性值中引入少量误差,对其最低有效位(Least Significant Bits, LSB)进行位操作,实现水印信息的嵌入。实验针对一个只包含数值型数据的数据库,首先选用单向Hash函数,根据用户给定的密钥和元组主键值以及需要标记的元组比例来确定哪些元组需要标记,然后根据可以标记的属性数和比特位数确定标记的属性及其比特位位置。在整个关系数据库中许多个比特位标记组合的比特位模式就是嵌入的水印信息。

这类方法采用基本的LSB嵌入算法,易于实现,但难以嵌入有实际意义的水印信息。本文研究了两种在数据库中嵌入具有实际意义的水印信息的算法来解决该问题,该方法所嵌入的水印信息具有保密性强、能盲提取等特征。

3.2 一种基于密钥分存的关系数据库数字水印算法

3.2.1 算法基本思想

(1) Blakley 秘密分享方法^[44]

Blakley 秘密分享方法的本质是几何图形的性质,其秘密分享方法是一个在 m 维的空间,使用 n ($n > m$) 个超平面(hyperplane)交集于一点,在 m 个平

面可以交集於一点, 则可以得到秘密值 (即是交集点)。

Blakley 秘密分享原理(如图 3-1) 说明, 在二维平面有 s_1, s_2, \dots, s_n, n 条线, 交于一点, 任意选取 2 条直线即能计算出秘密值。

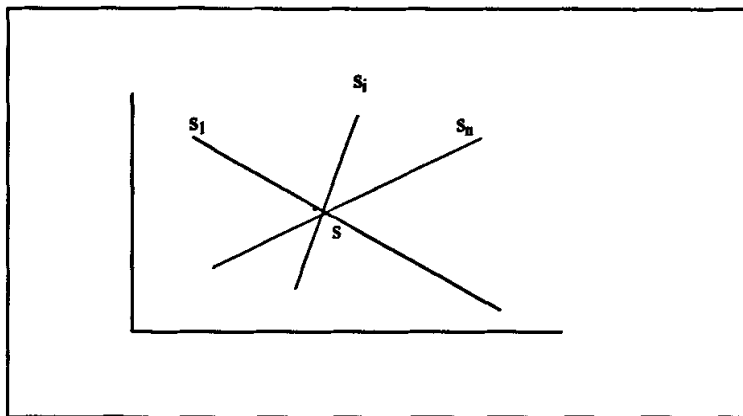


图3-1 Blakley 秘密分享原理

举例说明, 使用 Blakley 秘密分享门槛(3,6)方法, 将主密钥 $K = (120, 131, 135)$, 分解密钥为 6 个子密钥, 任意选取 6 个子密钥中的 3 个即能恢复主密钥(120, 131, 135)。

密钥分享方法如下:

(1) 选取主密钥 Y 为(120, 131, 135)。

(2) 密钥分解

取 $r=3, n=6$, 矩阵 $\alpha = (\alpha_{ij})_{6 \times 3}$ 为:

$$\alpha = \begin{pmatrix} 1 & 2 & 3 \\ 4 & 1 & 2 \\ 5 & 3 & 1 \\ 1 & 3 & 1 \\ 2 & 3 & 1 \\ 1 & 3 & 4 \end{pmatrix}$$

$$\text{则} \quad \mathbf{aY}^T = \begin{pmatrix} 1 & 2 & 3 \\ 4 & 1 & 2 \\ 3 & 3 & 1 \\ 1 & 3 & 1 \\ 2 & 3 & 1 \\ 1 & 2 & 4 \end{pmatrix} \begin{pmatrix} 120 \\ 131 \\ 135 \end{pmatrix} = \begin{pmatrix} 787 \\ 881 \\ 888 \\ 648 \\ 768 \\ 922 \end{pmatrix}$$

- (3) 通过计算得到的 (787, 881, 888, 648, 768, 922) 即为密钥 (120, 131, 135) 分解所得到的子密钥, 不可暴露。

密钥恢复方法如下:

- (1) 现在根据 6 个子密钥中的任意 3 个子密钥 (譬如: 787, 888, 768), 即可恢复密钥 (120, 131, 135)。
- (2) 根据矩阵 \mathbf{a} 和子密钥 (787, 888, 768), 可得方程组:

$$\begin{cases} x_1 + 2x_2 + 3x_3 = 787 \\ 3x_1 + 3x_2 + x_3 = 888 \\ 2x_1 + 3x_2 + x_3 = 768 \end{cases}$$

解方程组可得:

$$\begin{cases} x_1 = 120 \\ x_2 = 131 \\ x_3 = 135 \end{cases}$$

即恢复了密钥 (120, 131, 135)。

(2) 水印分存

把水印分成 n 份, 存入载体, 在水印提取过程中, 只要检测到 k ($k < n$) 份子水印就可以恢复原始水印。当然, 检测到的子水印可设为 m 份 ($k < m \leq n$), 我们可以从 m 份中取 k 份可靠性最高的进行原始水印的恢复, 然后每一位编码后的水印信息嵌入到一个元组集合中 (在集合中重复嵌入)。

3.2.2 水印嵌入算法

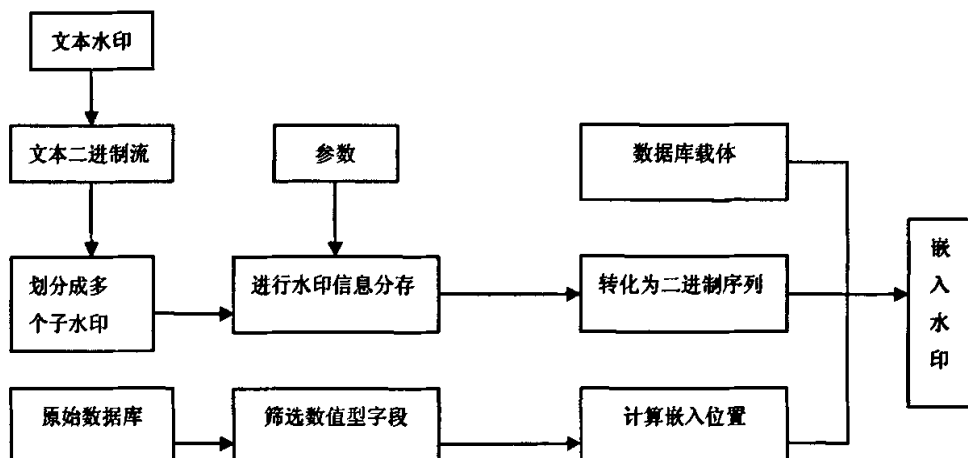


图 3-2 水印嵌入算法

水印嵌入算法的过程如下：

- 1) 首先读取要嵌入的文本水印信息，再将文本信息转化为二进制流；
- 2) 按照按二进制流的长短，将比特流等分成几部分，转化为十进制表示；
- 3) 按照 Blakley 秘密共享门限体制，选取合适的参数值 r, n ，将秘密分存，得到分存值 $k_i (i=1, 2, \dots, n)$ ，再将 k_i 转化为二进制序列，得到分存的水印信息；
- 4) 根据数据库的使用性、精确性以及容错性的的要求，选取数据库中满足条件的数值型字段；
- 5) 在第一个满足条件的字段 A_1 中，对数据库中的所有的元组纪录，选取字段值 $ri.A_1$ ，计算编号 $Index = Hash(key, Primary, ri.A_1)$ [43, 45]，并在该字段值中嵌入信息；
- 6) 继续选取满足条件的字段 A_i ，选取字段值 $ri.A_i$ ，计算编号 $Index = Hash(key, Primary, ri.A_i)$ ，重复 6)，直到所有满足条件的字段都嵌入了水印；

3.2.3 水印提取算法

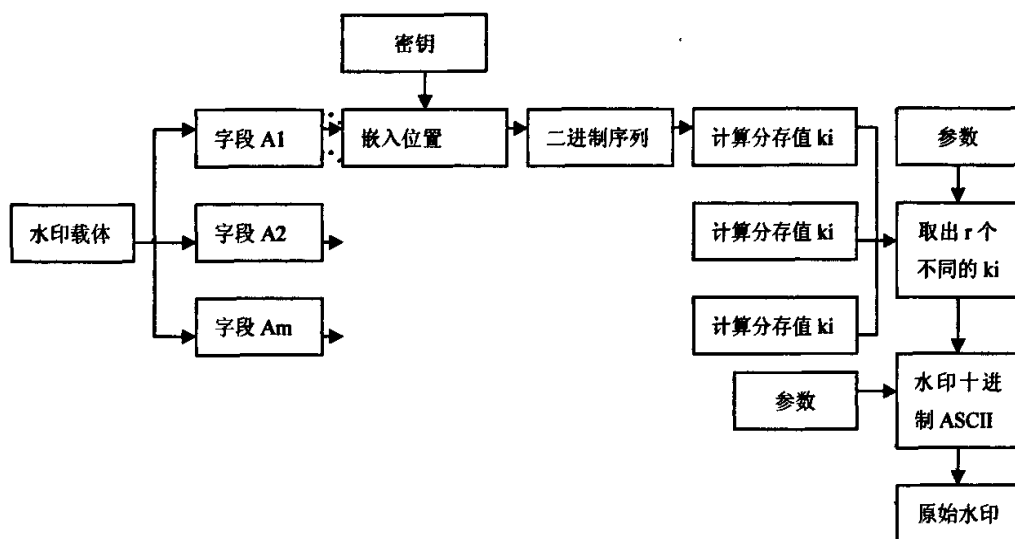


图 3-3 水印提取算法

水印的提取，是水印嵌入的逆过程。水印恢复时，首先从水印载体的各个嵌入水印的字段中按取出分存序列，计算出子水印的分存值，然后根据参数选择 r 个不同的 k_i ，计算出原始水印信息的十进制 ASCII 码值，恢复水印。

- 1) 从数据库中的 A_i 字段中，对每个元组计算编号 $\text{Index} = \text{Hash}(\text{Key}, \text{Primary}, r_i, A_i)$;
- 2) 根据计算得到的 Index 值，顺序将各 r_i, A_i 中嵌入的信息取出，得到二进制的子水印序列；
- 3) 根据二进制序列，计算出十进制的 k_i 的值，根据参数，判定选取 r 个不同的 k_i 值，如果少于 r 个，则继续从下一个字段中计算 k_i ，直到取出 r 个不同的 k_i 为止；若取完所有的嵌入水印的字段， k_i 数 $< r$ ，则水印提取失败；
- 4) 根据参数和 r 个 k_i 值，计算出原始水印的十进制 ASCII 码值，恢复得到原始水印；

3.2.4 算法性能分析

(1) 鲁棒性

该算法采用了 (r, n) 门限方案，在恢复水印时仅仅需要部分分存子水印，大大提高了抗子集的攻击能力，而且具有较好的鲁棒性，可以防御各种各样的攻击，包括正常的数据更新和恶意的攻击^[20, 32]。

但是，该算法有一定的局限性，由于嵌入的是文本消息，每一个字母对应一个 ASCII 码值，如果嵌入的信息较多，则对数据库载体的容量要求较高，否则提取的水印就不够精确。

(2) 不可见性高

选取有效的字段在误差允许范围之内对字段值进行嵌入, 水印不易被用户察觉, 也不会因为加了水印而影响关系数据的使用;

(3) 误判率小

采用了 (r, n) 门限方案和各有效字段都嵌入水印, 恢复和检测时, 根据参数值, 只需要各有效字段中有 r 份不同的子水印, 即可提取出完整的水印;

(4) 盲测: 该算法在检测关系数据库的水印时, 不需要原始数据库

3.3 一种基于图像的关系数据库水印

目前的关于关系数据库的水印方法都是很脆弱的, 主要有以下两个原因:

(1) 水印方法的数量是有限的, 因此他们就显得没什么意义, 而且凭直觉就很容易识别。

(2) 总的来说, 水印方法需要一个密钥来识别数据的合法性。但是在很多实际场合中, 这样的密钥是很难获得的。那么就迫切的需要一种不需要提供密钥的水印检测方法。

本文提出了一种新颖的用于关系数据库的水印方法, 我们称之为基于水印的图像 (IBW)。在本文的方法中, 在关系数据中嵌入一张可识别的图像, 用来描述版权信息。为了具有较高的效率, 水印的检测处理过程应该是可视的, 极细微的, 安全的, 可信的且能够抵抗攻击的。

3.3.1 水印嵌入算法

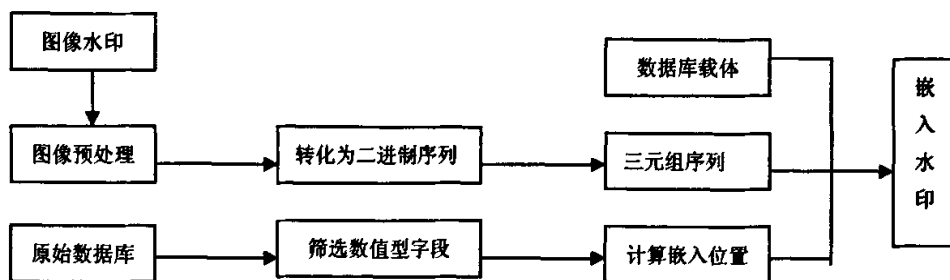


图 3-4 水印嵌入算法

- 1) 读取图像水印信息, 进行水印信息的预处理, 转化为二进制序列;
- 2) 将得到的二进制序列按每 3 个比特分为一组, 如果最后一位不足 3 个比特, 则可添上一个或两个 0 补足;

- 3) 选取数据库中满足有效位数要求的数值型字段;
- 4) 在第一个满足条件的字段中, 选取字段值 $r_i.A_i$, 计算编号 $\text{Index}=\text{Hash}(\text{key}, \text{Primary}, r_i.A_i)$;
- 5) 根据 Index 值, 将三元组分组序列按照表 3-1, 顺序替换满足条件的 $r_i.A_i$ 值的最低有效位;

表 3-1 三元组对应表

根据倒数第二位数的取值, 修改末位数的值										
要嵌入的信息	0	1	2	3	4	5	6	7	8	9
000	00	11	22	33	44	55	66	77	88	99
001	01	12	23	34	45	56	67	78	89	90
010	02	13	24	35	46	57	68	79	80	91
011	03	14	25	36	47	58	69	70	81	92
100	04	15	26	37	48	59	60	71	82	93
101	05	16	27	38	49	50	61	72	83	94
110	06	17	28	39	40	51	62	73	84	95
111	07	18	29	30	41	52	63	74	85	96

- 6) 继续选取满足条件的字段 A_i , 选取字段值 $r_i.A_i$, 计算编号 $\text{Index}=\text{Hash}(\text{key}, \text{Primary}, r_i.A_i)$ ^[48], 重复步骤 6), 直到所有满足条件的字段都嵌入了相同的水印;
- 7) 向可信中心申请注册保护, 提供关系数据库的关键字段、概要说明和水印提取程序; 可信中心根据是否以往有类似的注册的数据库和水印提取程序, 判定该数据库是否盗版, 若不是则在中心注册成功。

3.3.2 水印提取算法

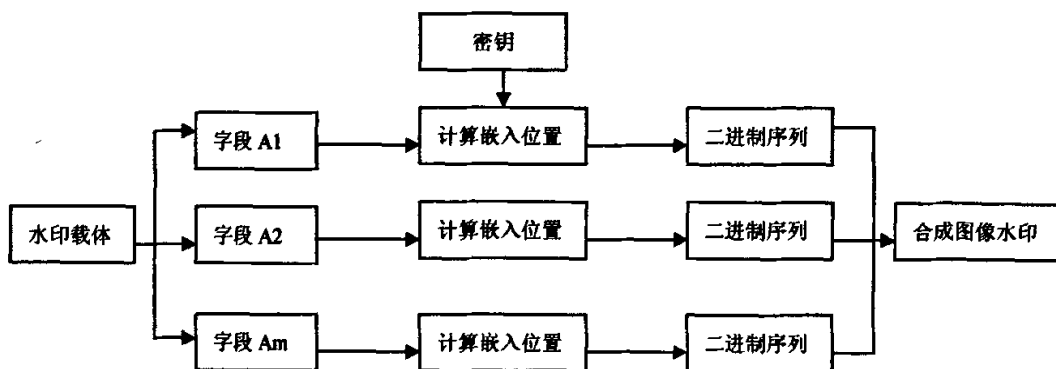


图 3-5 水印提取算法

水印恢复时，首先从水印载体中计算出水印的嵌入位置，然后取出值转化为二进制序列，最后恢复成图像水印。

- 1) 从数据库中的 A_i 字段中，计算 $\text{Index} = \text{Hash}(\text{Key}, \text{Primary}, r_i, A_i)$ [48]；
- 2) 根据 Index 值，顺序将各 r_i, A_i 的末两位数取出，对照表 3-1，得到二进制三元组序列；
- 3) 将各个字段中提取出的三元组序列组合起来，将合成后的二进制序列合成图像水印；

3.3.3 算法性能分析

(1) 鲁棒性

该算法采用三元组序列，在一个数值型字段值中可嵌入三个 bit 位，使可嵌入的水印信息量加大，而使用的水印信息是图像信息，一定程度的图像信息的丢失并不会引起水印信息的检测。大大提高了抗子集的攻击能力，具有较好的鲁棒性，可以防御各种各样的攻击，包括正常的数据库更新和恶意的攻击。特别地，该算法提出了基于可信中心的防添加攻击的思想，即盗版者对盗版数据库嵌入自己的水印，宣称自己对该数据库的所有权时，发生版权纠纷时，数据库的原始所有者可以通过可信中心检测出该数据库含有自己的水印而取得所有权；

(2) 不可见性高

选取有效的字段在误差允许范围之内对字段值进行嵌入，水印不易被用户察觉，也不会因为加了水印而影响关系数据的使用；

(3) 嵌入的信息多

采用三元组的思想，在一个字段值中就可嵌入3个bit，在各个有效的字段中都

可嵌入相同的信息;

(4) 误判率小

采用图像作为水印信息, 只要提取出一部分水印信息, 就能判定数据库版权的归属;

(5) 盲测: 该算法在检测关系数据库的水印时, 不需要原始数据库。

3.4 小结

在本文中, 我们首先提出了一种基于密钥分存思想的关系数据库水印技术, 该算法通过对水印信息的分存, 以及最低有效位的修改, 使水印信息分布得更“广泛”, 分散了水印遭破坏的风险, 能够很好的抵抗各种子集攻击原始水印(部分)的恢复只须任意 r 个子水印即可, 所以我们可以对子水印进行挑选, 能明显提高水印的鲁棒性。但是该算法也存在缺陷, 可嵌入的水印信息较少, 将水印信息分存时, 导致了嵌入的水印信息量扩大几倍。

接着提出了一种使用图像作为水印的, 新颖的关系数据库水印算法。我们的方法比先前的方法更直观, 也很容易地支持水印识别。我们通过引入各种攻击模型, 来评估我们的算法的表现。而且, 这也表明我们的算法比先前的算法更具有优势。但是, 该算法还可以进一步改善, 我们可以通过改进嵌入图像的组织方法, 来进一步改进水印本身的健壮性。

第四章 基于非数值型属性的数据库数字水印技术的研究

本章针对关系数据库的非数值型字段特点,提出了一种数字水印算法,通过在关系数据库的非数值型字段中嵌入版权信息,实现了关系型数据库的版权保护。

4.1 引言

目前的关系数据库数字水印算法的研究基本上都集中于对数字型字段的嵌入研究,而对非数值型字段的嵌入算法研究比较少,这是由于数字型字段容许一定的失真,对数据库的正常使用不造成影响,而非数值型字段则不同,任何微小的修改,都会导致字段值所标示的意思失真,所以目前在该方面的算法研究比较少,也还没有很好的解决方案。而非数值型字段数据在数据库中是很常见的,譬如网络节点数据库,医药数据库等等。本文通过保护关系数据库的非数值型字段提出了一种新的水印算法。提出的水印系统允许嵌入者定义相似函数来减少嵌入时引起的失真,同时减少嵌入过程中所需要修改的数据量。算法分析显示该算法能够较有效的抵抗各种攻击。

4.2 一种基于非数值型字段的关系数据库数字水印算法

4.2.1 算法基本思想

我们首先提出一种基于非数值型字段的数据库数字水印模型。我们在模型中考虑了以下因素:

(1) 数据,我们需要处理哪些数据。

非数值型字段的特点是,他们的值不能轻易的修改,任何修改都会改变原有的意思。我们认为关系数据库是由关系 R , 由主键 $R.Pk$ 和一个或多个属性 A_i 组成的。只要关系数据库的主键 $R.Pk$ 不允许修改,我们的水印系统就是有效的。

(2) 我们的目标:我们需要取得什么。

在关系数据库中嵌入水印,而不需要修改数据库中任何属性的值,同时,嵌入的水印又要足够强壮,能够抵抗各种类型的攻击。尽可能减少需要修改的数据量,可以通过一些相关数据的相似函数来减少嵌入水印时的失真。

(3) 难题:有哪些攻击来破坏我们的水印。

对水印关系数据库常见的恶意攻击方式有^[20, 32]:

子集选取: 数据窃取者不使用水印关系库的全部属性和元组, 仅仅使用其属性或元组的子集, 从而希望擦除水印。

子集增加: 增加一些属性和元组水印关系数据库中, 从而希望破坏水印。

子集更新: 更新水印关系数据库中的属性值, 从而希望破坏水印。

混合和匹配攻击: 数据窃取者从包含相似信息的多个关系库中获取不相连的元组创建自己的关系库。

添加攻击: 数据窃取者在窃取来的已经加水印的关系数据库上再简单的加上他自己的水印, 并声明自己对关系数据库的所有权。

可逆性攻击: 如果数据窃取者在其偷来的关系数据库内发现了一个虚幻的水印, 就可以采取可逆性攻击, 声称自己对关系数据库的所有权。而实际上数据窃取者声称的水印只是随机出现的水印。当然, 针对关系数据库不同的水印算法, 对其攻击方式还很多并且各不相同, 每个水印算法都应能够防御各种攻击, 都应具有很强的鲁棒性。

(4) 符号表示

在本文的算法描述中所用的符号表示的意思如表4-1所示:

表 4-1 算法描述中符号含义

符号	含义
E	所有非字段值元素的集合
G	一个伪随机数产生器
N	可以嵌入水印的元素总数
K1	用来嵌入水印时的密钥
K2	用来计算嵌入位置时的密钥
P	s, x_i 值的一维表
R	数据库中的关系
sf	相似函数
T	数据中的所有元素的数目
t	关系中的一个元组
V	选择嵌入元素时的一个二进值指示器

4.2.2 水印嵌入算法

(1) 计算嵌入位置

我们假设关系中的主键 $R.Pk$ ，在不对数据造成不可接受的破坏的情况下是不能修改的，我们希望所选择的元素与在关系中的相对位置是独立的。

所以，我们使用主键来唯一的确定一个元素，为了使这个过程变得安全，我们采用了密钥 K 和主键 $R.Pk$ 连接起来，得到一个值，用来嵌入到虚拟随机数生成函数 G 中。

表 4-2 算法 1 选择要嵌入水印元素的算法

```

GetElements( $K_1, R$ )
//选择要嵌入水印元素的函数
For each tuple  $t \in R$  do
Seed  $G$  with  $R.Pk \parallel K_1$  //对每个元组进行计算
If next( $G$ )  $\leq r$  then
 $V_t = 1$ 
Else
 $V_t = 0$ 
End if
End for
Return  $\bar{V}$  //得到嵌入位置的一维表
End GetElements( $K_1, R$ )

```

在执行完算法 1（如表 4-2 所示）后，我们可以得到一个一维向量表 \bar{V} ，表的大小取决于数据表中元组的数量，表 \bar{V} 中每一个元素的值为 0 或 1，表示选择的结果。所有的元素值为 1 表示被选择用来嵌入水印。为了控制嵌入水印的影响，我们通过参数 r （关系中选择嵌入水印的元素和总的元素的比值）来控制。期望选择的元素 $N=r*T$ 。因此，如果 $r=0.25$ ，就是 25% 的元素被选择嵌入水印。

(2) 相似函数

当通过数值型字段来嵌入水印时，我们可以很轻易的通过对字段值的轻微的增加，减少来隐藏水印。然而，在非数值型字段中，隐藏一个水印是非平滑的，因为任何轻微的修改，意味着是被另一个值完全取代。我们认为造成失真的主要

原因是用来取代的类别的值和原类别的值所表示的意思相似性太小。为了使失真最小，我们提出了用户定义的相似函数：

$$sf(e_1, e_2) \rightarrow [0, 1] \quad (4-1)$$

给定 e_1 和 e_2 ，相似函数返回一个0和1之间的相似值，0 表示非常的不同，1 表示非常的相似。通过相似函数，我们就可以量化和减少交换两个元素时所产生的失真。

(3) 嵌入水印

水印嵌入过程由 2 步组成：

- 1) 找到元素用来取代原有的元素；
- 2) 执行取代；

$$Embed(R, k_2, sf, M) \rightarrow R' \quad (4-2)$$

为了在关系 R 中隐藏水印，我们需要一个不同于选择嵌入元素时的密钥 k_2 ， sf 和 M，我们采用算法 1 进行选择哪些元素来嵌入水印，在用新的元素代替原来的元素时，应该满足下面的约束：

$$\sum_{i=0}^N s_i x_i \geq M' \quad (4-3)$$

其中 $\vec{X} = \{x_i\}$ ，是伪随机数，统一分布在 $[-\lambda, \lambda]$ ，而 λ 是一个健壮性参数， $S = \{s_i\}$ 是被取代元数中的最不显著位，就像整数中的 $\{-1, 1\}$ ，M 是用户定义的安全参数，决定了水印的健壮性。下面介绍 s_i ， x_i 的计算算法。

1) 计算 \vec{X}

为每一个被选择的元组计算 x_i ，对每个索引 i，在算法 1 计算中 $V_i=1$ 。使用密钥 K_2 和主键 R.Pk 和虚拟随机数生成器 G。通过 G 可以得到一系列 N 虚拟随机数，在 \vec{S} 间，换句话说，我们用一个哈希函数，把主键和密钥 k_2 串联起来：

$$H(R.P_k | K_2) \rightarrow [-\lambda, \lambda] \quad (4-4)$$

我们要求哈希函数 $H(.)$ 是一个单向哈希函数，也就是说从值 $H(R.P_k | K_2)$ 计算出 $R.P_k | K_2$ 是不可能的。

2) \vec{S} 的选择

一旦 $\vec{X} = \{x_i\}$ 确定了，我们必须确定 s_i 的值满足约束条件。我们必须最小化

嵌入水印对数据的影响。

我们通过用来嵌入水印的原始元素 e_i 的最不显著位 $lsb(e_i)$ 初始化每一个 s_i

表 4-3 算法 2 计算 \vec{S} 的算法

```

SetSElements( $\vec{S}, \vec{X}, M$ )

//计算  $\vec{S}$  的算法

P=ComputerProducts( $\vec{S}, \vec{X}$ )//得到 p 的一维表

SortInIncreasingOrder( $\vec{P}$ )

While  $\hat{M} < M$  do

    I=ObtainIndexOfmostNegativeProduct( $\vec{P}$ )

    Swap( $\vec{S}, i$ )

    RemoveIFromP( $\vec{P}, i$ )

     $\hat{M}$ =ComputerMark( $\vec{S}, \vec{X}$ )

End while

```

$$s_i = \begin{cases} -1 & \text{if } lsb(e_i) = 0 \\ 1 & \text{if } lsb(e_i) = 1 \end{cases} \quad (4-5)$$

初始化好 \vec{S} 后, 我们计算

$$\sum_{i=0}^N s_i x_i = \hat{M} \quad (4-6)$$

然后, 如果 $\hat{M} > M$, 我们把 \hat{M} 代替 M , 不对数据做任何修改, 最小失真度。当 $\hat{M} < M$ 时, 我们需要修改 \vec{S} 一些值来满足限制, 通过在算法 2 (如表 4-3 所示) 中调用算法 3 (如表 4-4 所示) 和算法 4 (如表 4-5 所示)。

表 4-4 算法 3 给定一个 \vec{S} , \vec{X} 计算水印

```

function ComputeMark ( $\vec{S}$ ,  $\vec{X}$ ) return  $\hat{M}$ 
//给定一个  $\vec{S}$ ,  $\vec{X}$  计算水印
 $\hat{M} = 0$ 
For i =1 to N do
 $\hat{M} = \hat{M} + s_i x_i$ 
end for
return  $\hat{M}$ 
end function ComputeMark

```

表 4-5 算法 4 使用最相似的替换原有的元素

```

function Replace ( $\vec{S}$ , i)
//使用最相似的替换原有的元素
lsb=0 //初始化最不显著位
if  $s_i = 1$  then //更改  $s_i$  的值
 $s_i = -1$ 
Lsb=0
Else
If  $s_i = -1$  then
 $s_i = 1$ 
Lsb=0
End if
newElement=getMostSimilarElement( $e_i$ , lsb, sf)
 $e_i = \text{newElement}$ 

```

4.2.3 水印提取系统

水印恢复必须确定, 在哪个位置上嵌入了水印。为了确定位置, 我们需要一些参数: R , M , 只有数据拥有者才知道的密钥 K_1 , K_2 , 因此水印的恢复过程

$$\text{Recovery}(\hat{R}, K_1, K_2, M) \rightarrow (\text{yes/no}) \quad (4-7)$$

恢复过程和嵌入过程相似,也需要使用算法 1,获得嵌入水印的元素,在提取过程中,我们不需要原始的数据库,这是因为在水印嵌入过程中,我们并没有修改关系 R 中的主键。一旦标记的元素确定,每个 x_i 的值都可以根据嵌入时的算法,使用密钥 k_2 计算出来。最后, s'_i 的值可以通过使用 $lsb()$ 函数获得的最不显著位获得,总的来说 $s'_i \neq s_i$,这是因为数据的生命周期中的一些偶然的和有意的失真造成的,一旦上述的信息得到恢复,系统就可以计算

$$\hat{M}' = \sum_{i=0}^N s'_i x_i \quad (4-8)$$

一旦 $\hat{M}' \geq \frac{M}{2}$, 恢复系统就会认为数据中包含了一个水印,否则认为不存在水印。

4.2.4 算法性能分析

本水印系统能够抵御随机修改,水平取样,垂直取样等攻击。入侵者能够执行大范围的各种恶意的攻击,下面描述下本水印系统如何能够抵抗各种攻击。

(1) 垂直取样

我们的水印系统可以应用到各种至少有一个主键和属性 A 的关系 R 中,插入的水印不取决于任何属性之间的关系,可以单独地嵌入到任何你想嵌入的属性中。因此,通过选择一些属性,擦除一些属性的方法对我们的水印没什么影响,因为至少有一个标记的属性仍然存在。

(2) 水平和垂直的重排序

水平的重排序攻击,包括交换一些元组的位置,而不修改值。我们的水印系统是不易受影响的,因为关系 R 中的相对位置是不被用来决定哪些元素被嵌入水印的。

同样地,垂直重排序,包括交换一些属性的顺序,而不修改值。对这种攻击,我们的方法是,我们在属性中嵌入水印时与该属性在关系 R 中的相对位置是无关的。

(3) 随机选择元素修改攻击

水印检测时只有当 $\hat{M}' \geq M/2$, 才能检测到水印的存在。入侵者想通过随即选择元素进行修改,来破坏水印,则必须修改足够多的元素。因此,一个简单有效的做法是,增加嵌入水印的元素 n , 可以通过减小 λ , 和增加元素可嵌入水印的元素 N 。

(4) 水平取样

这种攻击是基于随机的选择关系 R 中的一些元组。这种通常对数据库数字

水印算法攻击很有效的方法，对本系统却不是非常有效。因为，为了破坏水印，入侵者必须选择尽可能少的元组，而且这些被选择的元组中包含尽可能少的被嵌入了水印的元组，这是比较困难的。而一个没被选择的元组就像一个被改变的元组，所以水平取样分析攻击就类似于上面讨论的随机修改元组攻击。

4.3 小结

现有的数据库水印技术大多是对数值型字段进行标记，而在非数值数据中因为难以找到可辨认的冗余空间，给水印的安全嵌入带来困难。只有解决了非数值型数据的水印嵌入问题，数据库水印技术才具有真正的实用性。

在本文中，我们提出了一种基于非数值型字段的关系数据库水印算法。该算法最小化了在嵌入水印时需修改的数据量，并且允许数据的拥有者自己定义一个相似函数选择元素进行水印嵌入，在水印恢复时不需要原始数据库。

通过对算法的健壮性分析表明，本水印算法可以抵御各种攻击。相似函数是由用户自己定义的，相似函数的定义取决于受保护的数据库的特点。如何选取最佳的相似函数，引入人工智能领域中的神经网络，遗传算法等算法，将是基于非数值型字段的关系数据库算法下一步的研究方向。

第五章 数据库数字水印系统的应用

本章我们详细介绍数据库数字水印系统的方案设计,并介绍了该方案在版权保护方面的应用。

5.1 系统方案设计

5.1.1 引言

数字水印技术是近十几年来兴起的一门崭新的技术,它通过在数字产品中嵌入可感知或不可感知的信息来确定数字新产品的所有权或检验数字内容的原始性。当前数字水印技术主要集中在多媒体信息领域(如:图像、音频、视频)并取得较好的进展。随着信息化程度的提高,关系数据库版权保护问题变得十分重要,如何在关系数据库中嵌入数字水印从而实现版权保护,已成为人们亟待解决的问题。目前关系数据库的数字水印算法普遍存在以下问题:

- (1) 通常嵌入水印信息少量,且没有实际意义;
- (2) 水印的健壮性较差,不能抵抗正常的数据修改;
- (3) 无法抵抗二次水印的攻击。

为了能更好的解决数据库的版权保护问题,本章将数据库数字水印算法应用于实际。

5.1.2 系统采用的算法

综合前面讨论的结果,我们在设计水印系统时,采用的算法是在第三章中提到的基于图像的关系数据库数字水印算法。针对数值型字段,选择合适的元素进行嵌入。采用三元组序列,极大的扩大了数据库中嵌入水印的容量,而且提高了系统的健壮性。

水印系统主要分成四个模块:水印生成,水印嵌入,水印提取和水印认证模块。

5.2 系统的实现

5.2.1 软硬件平台

(1) 硬件平台

本系统既可以运行在局域网环境下，也可以运行在广域网环境下。其客户部分可运行在通用的PC上，服务器部分可运行在通用的中高档服务器上。

(2) 软件平台

操作系统：

Windows 2000, Windows XP及以上操作系统。

数据库系统：

SQL Server 2000。

开发环境：

Microsoft Visual Studio .Net 2003。

5.2.2 系统设计实施

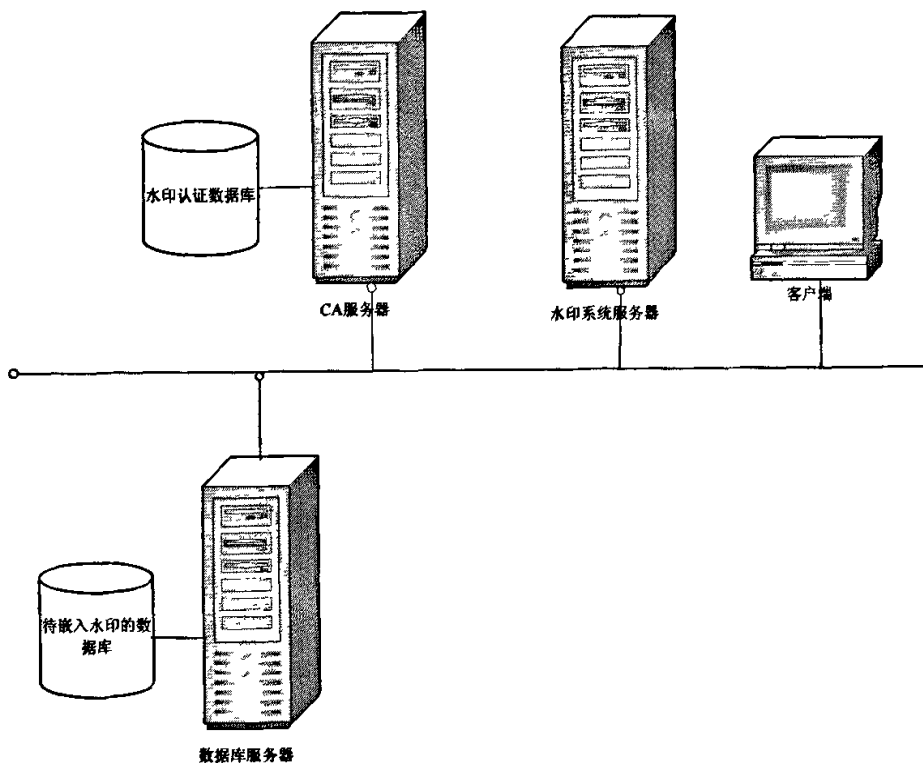


图5-1系统架构图

水印系统的架构图如5-1所示。根据系统的总体设计和各功能模块的具体设计,下面将逐一介绍系统的各个功能模块的关键实现部分。

(1) 水印生成模块

水印生成模块是数据库水印系统的关键模块之一。该功能模块的实现分为三个步骤:获取用户要嵌入的水印信息、水印预处理。

A) 获取用户嵌入的信息

用户要嵌入信息通常是文本或者图像信息,我们通过开发平台中的控件和编写的函数获得用户的信息。主要的代码如下:

```
private void btnInsert_Click(object sender, System.EventArgs e)
{
...
//读取水印图像,并转化为二进制序列
Image image = Image.FromFile("C:\\1.jpg");
MemoryStream ms = new MemoryStream();
image.Save(ms, ImageFormat.Jpeg);
ms.Flush();
ms.Seek(0, SeekOrigin.Begin);
byte [] buffer = new byte[ms.Length];
ms.Read(buffer, 0, (int)ms.Length);
//这里已经转成了字节
...
}
```

B) 嵌入信息预处理^[46-47]

为了能将用户要嵌入的信息嵌入到关系数据库中,我们需要将信息进行一系列的处理。下面给出处理的流程(如图5-2所示):

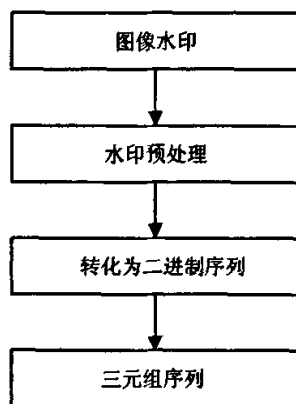


图5-2 信息预处理过程

将得到的图像的二进制序列按每 3 个比特分为一组，如果最后一位不足 3 个比特，则可添上一个或两个 0 补足；主要的代码如下：

```
private void BytetoBit ( )
{
    ...
    //byte[] 转为二进制字符串表示
    string strResult=string.Empty;
    string strTemp;
    for(int i=0;i< buffer.Length;i++)
    {
        strTemp=System.Convert.ToString(buffer [i], 2);
        strTemp
=strTemp.Insert(0,new string('0',8-strTemp.Length));
        strResult+=strTemp;
    }
    ...
}
```

我们先通过函数BytetoBit()，将Byte数组转化为二进制字符串，然后使用Convert.ToInt32函数将字符串每三位截取出来并转为int值，存在三元组中。

```
private void BittoThreeBit ( )
{
    //将二进制序列分为三元组，再转化为十进制表示
    ...
    For ( i=0; i< strResult.length;i=i+3)
    {
        j=i/3;
        ThreeBit[j]=Convert.ToInt32(strResult .Substring (i + 2, i), 3);
        //二进制字符串中每三位截取出来并转为int值，存在三元组中
    }
    ...
}
```

(2) 水印的嵌入

要在数据库表中嵌入信息，我们首先要选择嵌入水印的位置，为了能够更好

的抵抗子集选择, 数据重排序等攻击, 我们使用主键来唯一的确定一个元素, 为了使这个过程变得安全, 我们采用了密钥 K 和主键 $R.Pk$ 连接起来, 得到一个值, 用来嵌入到伪随机数生成函数 G 中。

在执行完第四章中算法 1 后, 我们可以得到一个一维向量表 \bar{V} , 表的大小取决于数据表中元组的数量, 表 \bar{V} 中每一个元素的值为 0 或 1, 表示选择的结果。所有的元素值为 1 表示被选择用来嵌入水印。为了控制嵌入水印的影响, 我们通过参数 r (关系中选择嵌入水印的元素和总的元素的比值) 来控制。

A) 读取数据库数据

```
private void btnsearch_Click(object sender, System.EventArgs e)
{
    //获取要嵌入水印的数据表, 将其存放在DataSet中
    ...
    strsearch="select * from tbl_it where l=1 ";
    //获取要嵌入水印的表
    SqlDataAdapter MyAdapter=new SqlDataAdapter (strsearch, conn);
    //建立数据集
    dss =new DataSet ();
    MyAdapter.Fill (dss,"tbl_it");//将表的信息读入DataSet中
    dgd_id.DataSource =dss.Tables ["tbl_it"].DefaultView ;
    dgd_id.DataBind();
    conn.Close();
    ...
}
```

B) 伪随机数生成函数 G

在生成伪随机数生成函数 G 时, 我们采用了.NET 自带的随机数类 `System.Random` 类提供的方法产生各种满足不同要求的随机数, 随机数类 `System.Random` 产生的方法主要有以下几种, 如表 5-1 所示:

表5-1 System.Random类提供的各种方法

编号	方法名称	功能描述
1	Next ()	返回一个 0~2147483647 之间的整数
2	Next (i)	返回一个 0~i 之间的整数
3	Next (i, j)	返回一个 i~j 之间的整数
4	Nextdouble (Seed)	返回一个 0~1 之间的随机小数

C) 计算水印嵌入位置

计算水印嵌入位置的代码如下所示:

```
private void GetElements (K,R)    //选择要嵌入水印元素的函数
{
    ...

    for(int i=0;i<ds.Tables[0].Rows.Count;i++)
        // 对所有的记录进行循环
        {

            // 判断 next(G) ≤ r, 如果是, 则  $V_i=1$  否则  $V_i=0$ 
            If (System.Random.Nextdouble (ds.Tables[0].Rows[i][0]) <=r)
                {
                    arr[i] = 1; //在该元组中嵌入水印
                }
            Else
                {
                    arr[i] = 0; //不在该元组中嵌入水印
                }
        }
    ...
}
```

D) 水印的嵌入

该模块主要完成水印的嵌入功能, 首先通过前面的方法计算嵌入水印的位置后, 然后判断按照表 1, 顺序替换满足条件的 ri.A1 值的最低有效位; 水印嵌入函数的主要代码如下:

```

private void EmbedMark ()
{
    //通过修改 1sb 位在数据库中嵌入水印
    ...
    for(int j=1;j<n;j++)
    {
        Int k=0;
        for(int i=0;i<ds.Tables[0].Rows.Count;i++)
        {
            Set1sb(ds.Tables[0].Rows[i][j],ThreeBit[k]);
            //将数值字段的最不显著位的值修改为三元组内的值
            K++;
            ...
        }
    }
}

```

(3) 水印的提取

恢复过程和嵌入过程相似,也需要使用算法 1,获得嵌入水印的元素,将各 ri.A1 的末两位数取出,对照表 1,得到二进制三元组序列;将各个字段中提取出的三元组序列组合起来,将合成后的二进制序列合成图像水印;

A) 提取二进制序列

```

private void Getbit()
{
    ...
    int k=0;
    for(int j=1;j<n;j++)
    {
        for(int i=0;i<ds.Tables[0].Rows.Count;i++)
        {
            if (arr[i] = 1)
            {
                GetThreeBit(ds.Tables[0].Rows[i][j],ThreeBit[k]);
            }
        }
    }
}

```

```
        //将数值字段的最不显著位的值提取出来，转化为二进制字符串
        K++;
    }
    ...
}
}
```

得到二进制字符串序列后，再转存为 Byte 数组中。

```
private void BittoByte( )
{
    ...
    //二进制字符串转化为 byte[]
    byte[] buffer =new byte[strResult.Length/8];
    for(int i=0;i< buffer.Length;i++)
        buffer [i] =Convert.ToByte(strResult.Substring(i*8, 8), 2);
    ...
}
```

B) 显示水印图像的代码如下：

```
private void btnRecover_Click(object sender, System.EventArgs e)
{
    //获取二进制序列，并转为图像显示出来
    MemoryStream ms2 = new MemoryStream(buffer, 0, buffer.Length);
    ms2.Seek(0, SeekOrigin.Begin);
    Image image2 = Image.FromStream(ms2);
    image2.Save("C:\\2.jpg", ImageFormat.Jpeg);
}
```

5.3 系统测试

本系统的测试环境如图5-3所示。

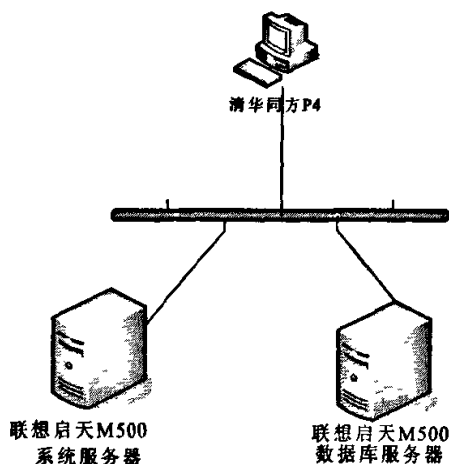


图5-3 系统测试环境示意图

本测试是在局域网环境内进行的。测试用的系统服务器、数据库服务器都是联想启天M500(Athlon 3000+, 512M)，其操作系统为Windows2000 Server。测试所需的软件有Visual Studio .Net 2003, SQL Server 2000。

我们对系统进行了功能测试、性能测试和安全性测试。下面是系统测试的结果。

1、功能测试

主要对本系统所提供的水印的嵌入，水印的提取分别进行测试，结果基本达到了该系统的设计目标。

2、性能测试

为了测试本系统的性能，本文基于关系数据库进行了水印的嵌入和提取实验。实验中用到的关系数据库共有记录6500条，每条记录有数值型字段值8个，非数值型字段2个。实验时，测得数据库中共有14121个字段值可以嵌入信息。嵌入的水印信息为如图5-4所示：

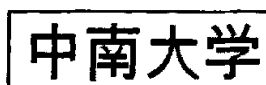


图5-4 嵌入的水印图像信息

(1) 子集选取攻击测试

子集攻击时，分别按10%、20%、30%、40%、50%、60%、70%、80%、90%、100% 随机选择数据库中不同的记录。仿真结果如下：

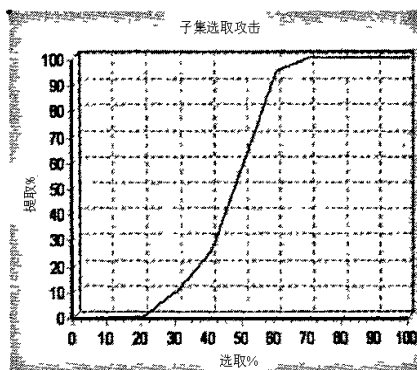


图5-5子集选取攻击

由图5-5可知，对于子集选取攻击，选取的数据比例越大，选取的字段值就越多，相应的可用提取水印信息的字段越多，因此选取的比例越大时，准确恢复的比例越高。

(2) 子集改变攻击

子集改变攻击时，分别按10%、20%、30%、40%、50%、60%、70%、80%、90%、100% 随机更改数据库中不同的记录。仿真结果如下：

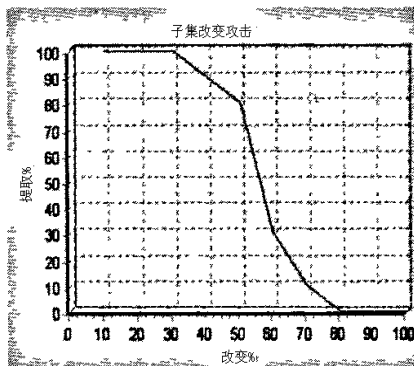


图5-6 子集改变攻击

由图5-6可知，在受到子集修改攻击时，攻击的比例越大，对水印数据破坏的程度越高，因而准确恢复的比例越低。

(3) 子集添加攻击

子集添加攻击时，分别按10%、20%、30%、40%、50%、60%、70%、80%、90%、100% 随机增加数据库中不同的记录。仿真结果如下：

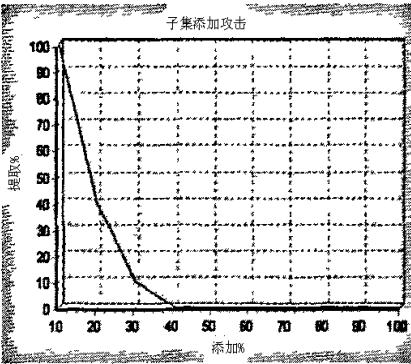


图5-7 子集添加攻击

由图5-7可知，在受到子集增加攻击时，攻击的比例越大，对水印数据破坏的程度越高，因而准确恢复的比例越低。

（4）数据库可用性验证

我们的算法是通过修改数据库中的数值型字段值中的最低有效位，来嵌入水印信息。在嵌入水印后应不影响原数据库的使用。在我们的实验中，水印前的数据库如表5-2所示，加水印后的数据库如表5-3所示，嵌入水印后虽然对数据库有轻微的修改，但是在数据库的允许范围之内，所以并不影响数据库的正常使用。

表 5-2 加水印前气象信息情况表

城市	年	月	月平均气压 (0.1百帕)	月平均气温 (0.1摄氏度)	月极端最高气温 (0.1摄氏度)	月平均相对湿度 (百分率)	月平均相对湿度 (百分率)	月平均总云量 (0.1成)	月平均风速 (0.1米/秒)	月降水量 (0.1毫米)
北京	2002	1	10242	-37	129	-183	44	27	26	27
北京	2002	2	10220	-17	198	-160	44	36	28	49
北京	2002	3	10174	58	264	-150	46	45	31	83
北京	2002	4	10100	142	330	-32	46	47	32	212
北京	2002	5	10057	199	368	26	53	52	28	342
北京	2002	6	10012	244	392	98	61	59	24	781
北京	2002	7	9997	262	419	166	75	67	20	1852
北京	2002	8	10037	248	361	114	77	61	18	1597
北京	2002	9	10105	200	344	43	68	46	20	455
北京	2002	10	10167	131	293	-35	61	39	21	218
北京	2002	11	10213	46	220	-106	57	34	24	74
北京	2002	12	10238	-15	195	-156	49	29	25	28

表 5-3 加水印后的气象信息情况表

城市	年	月	月平均气压 (0.1百帕)	月平均气温 (0.1摄氏度)	月极端最高气温 (0.1摄氏度)	月平均相对湿度(百分率)	月平均相对湿度(百分率)	月平均总云量 (0.1成)	月平均风速 (0.1米/秒)	月降水量 (0.1毫米)
北京	2002	1	10242	-37	129	-183	44	27	26	27
北京	2002	2	10221	-18	195	-161	45	31	22	47
北京	2002	3	10174	58	264	-150	46	45	31	83
北京	2002	4	10100	142	330	-32	46	47	32	212
北京	2002	5	10052	193	367	21	50	54	27	348
北京	2002	6	10012	244	392	98	61	59	24	781
北京	2002	7	9997	262	419	166	75	67	20	1852
北京	2002	8	10037	248	361	114	77	61	18	1597
北京	2002	9	10105	200	344	43	68	46	20	455
北京	2002	10	10169	135	299	-31	62	33	20	216
北京	2002	11	10213	46	220	-106	57	34	24	74
北京	2002	12	10238	-15	195	-156	49	29	25	28

5.4 本章小结

本章给出了数据库数字水印系统的设计方案，详细描述了解决方案的工作流程，并分析了方案的安全性，最后简要介绍了作者开发的数据库数字水印系统，同时也给出了系统测试阶段的功能和性能测试结果。

第六章 总结与展望

6.1 本文所做工作的总结

数字水印技术是实现版权保护的一种有效手段。目前,基于多媒体的数字水印技术研究已经取得了很好的研究成果。由于关系数据库的特殊性,基于关系数据库的数字水印技术研究进展却十分缓慢。在保证不可见性的前提下,如何提高数据库水印算法的鲁棒性,如何拓展关系数据库的冗余空间,是数据库数字水印技术研究发展的重要方向。本文针对这两点,对以往的数据库数字水印算法进行了深入的研究分析,提出了三种数据库水印算法,并在文章中给出了一个数据库数字水印系统的设计方案,完成了相应的测试分析。

本文首先提出了一种基于密钥分存思想的关系数据库水印技术,该算法通过对水印信息的分存,以及最低有效位的修改,使水印信息分布得更“广泛”,分散了水印遭破坏的风险,能够很好的抵抗各种子集攻击原始水印(部分)的恢复只须任意 r 个子水印即可,所以本文可以对子水印进行挑选,能明显提高水印的鲁棒性。但是该算法也存在缺陷,可嵌入的水印信息较少,将水印信息分存时,导致了嵌入的水印信息量扩大几倍。

接着提出了一种使用图像作为水印的,新颖的关系数据库水印算法。我们的方法比先前的方法更直观,也很容易地支持水印识别。在算法中我们引入了三元组的思想,在一个字段值中就可嵌入 3 个 bit,极大的扩大了水印信息的容量。我们通过引入各种攻击模型,来评估我们的算法的表现。特别地,该算法提出了基于可信中心的防添加攻击的思想,即盗版者对盗版数据库嵌入自己的水印,宣称自己对该数据库的所有权时,发生版权纠纷时,数据库的原始所有者可以通过可信中心检测出该数据库含有自己的水印而取得所有权。而且,这也表明我们的算法比先前的算法更具有优势。但是,该算法还可以进一步改善,我们可以通过改进嵌入图像的组织方法,来进一步改进水印本身的健壮性。

第三种算法是基于非数值型属性的数据库数字水印算法。

目前的关系数据库数字水印算法的研究基本上都集中于对数值型字段的嵌入研究,而对非数值型字段的嵌入算法研究比较少,这是由于数字型字段容许一定的失真,对数据库的正常使用不造成影响,而非数值型字段则不同,任何微小的修改,都会导致字段值所表示的意思失真,所以目前在该方面的算法研究比较少,也还没有很好的解决方案。而非数值型字段数据在数据库中是很常见的。本文提出的算法允许嵌入者定义相似函数来减少嵌入时引起的失真,同时减少嵌入

过程中所需要修改的数据量,算法性能分析显示该算法能够较有效的抵抗各种攻击。

算法分析结果表明,如能将基于数值型的和非数值型的数据库数字水印算法结合起来,能够在数据库中嵌入更多的信息,也使水印系统的鲁棒性大大提高。

6.2 未来工作的展望

本文对关系数据库数字水印技术的研究还仅仅是起步,还有大量的工作值得展开。就本人的研究过程和目前的研究状况中,认为还有以下几个方面需要进一步深入研究:

(1) 对提高数据库水印的鲁棒性和安全性的研究。目前的一个基本共识是:在隐藏算法公开的前提下,水印算法的安全性依赖于密钥的安全性。只有将透明性、完整性、鲁棒性等逻辑有机集成到形式化的水印安全模型中,水印数据库系统的安全性才能得到可靠保证。

(2) 对数据库水印的信号容量的研究。信号容量问题是信息隐藏中的一个关键技术。到目前为止,还没有对某种信息载体可以隐藏多少信息量进行准确计算的理论方法,同样,数据库水印的信息容量也有待进一步的理论研究。

(3) 对水印宿主数据类型的扩展。现有的数据库水印技术大多是对数值型字段进行标记,而在非数值数据中因为难以找到可辨认的冗余空间,给水印的安全嵌入带来困难。只有解决了非数值型数据的水印嵌入问题,数据库水印技术才具有真正的实用性。

(4) 对数据库脆弱水印(Fragile Watermark)的研究。相对于研究较多的鲁棒水印而言,脆弱水印在具有较强的抗攻击能力的同时,还要求有较强的敏感性。要实现对数据的完整性验证,需要依靠脆弱水印的特点来进行篡改提示与定位乃至数据复原。

(5) 数据库水印的性能评价和测试基准的建立。对水印的性能建立合理的评估方法和测试基准将有助于数据库水印研究的发展。水印的性能评估主要包括水印的透明性、安全性和鲁棒性。要保证各个水印系统在可比较的条件下进行测试,就需要建立通用的性能指标、测试数据库和测试基准。

总之,随着信息化程度的提高,关系数据库版权保护问题变得越来越重要,关系数据库数字水印技术作为版权保护的重要手段是一项很有发展前途的技术,目前还处于初步的研究阶段,还有很多研究工作可以做,还需要去进一步的研究。

参考文献

- [1] 孙圣和, 陆哲明. 数字水印处理技术. 电子学报, 2000, 8:85~90
- [2] 尹浩, 林闯, 邱锋. 数字水印技术综述. 计算机研究与发展, 2005, 42 (7): 1093~1099
- [3] Fabien A. P. Petitcolas, Ross J. Anderson and Markus G. Kuhn. Information Hiding-A Survey, Proc. of IEEE, 87(7):1062~1078, 1999
- [4] 易开祥, 石教英, 孙鑫. 数字水印技术研究进展. 中国图像图形学报, 2001, 6(2):111~117
- [5] 陈明奇, 钮心忻, 杨义先. 数字水印的研究进展和应用. 通信学报, 2001, 22 (5): 71~79
- [6] Frank Hartung. Multimedia Watermarking Techniques. IEEE, Vol. 87, No. 7:1079 ~1107, July 1999
- [7] Hartung F, Girod B. Watermarking of MPEG-2 encoded video without decoding and re-encoding. SPIE Proceeding on Multimedia Computing and Networking, San Jose, 1997, 30(20):264~273
- [8] Lintian Qiao, Klara Nahrsted. Non-invertible watermarking methods for MPEG encoded audio. SPIE Proceedings on Security and Watermarking of Multimedia Contents, San Jose, 1999, 36(57):194~202
- [9] Zhao. J, Koch. E. Embedding robust labels into images for copyright protection. In: Proceedings of the Know Right' 95 Conference on Intellectual Property Rights and New Technologies, Vienna, Austria, 1995: 241~251
- [10] 潘蓉, 高有行. 数字图像水印技术研究. 湖南大学学报 (自然科学版), 2002, 29(2):117~123
- [11] 易开祥. 数字图像加密与数字水印技术研究. 浙江大学博士学位论文, 2001 (7)
- [12] 肖湘蓉, 孙星明. 基于内容的英文文本数字水印算法设计与实现. 计算机工程, 2005, 22:29~31
- [13] 刘豪, 孙星明, 刘晋飏. 基于字体颜色的文本数字水印算法. 计算机工程, 2005, 15:129~131
- [14] Cox. I. J, Kilian. J, Leighton. T et al. A secure, robust watermark for multimedia. In: Proceedings of InfoHiding' 96, Cambridge University,

London, England, 1996: 185~206

[15] Cox. I. J, Kilian. J, Leighton. Tetal. Secure spread spectrum watermarking for multimedia. IEEE Trans. on Image Processing, 1997, 6(12): 1673~1687

[16] 尹康康, 潘志庚, 石教英. 一种强壮的网格水印算法. 计算机辅助设计与图形学学报, 2001, 12:38~42

[17] 张立和, 杨义先. 软件水印综述. 软件学报, 2003, 14(2):268~277

[18] 牛夏牧, 赵亮, 黄文军, 张慧. 利用数字水印技术实现数据库的版权保护. 电子学报, 2003, 31(12A):2050~2053

[19] 李德毅, 史雪梅, 孟海军. 隶属云和隶属云发生器. 计算机研究和发展, 1995, 32(6): 15~20

[20] 张勇, 赵东宁, 李德毅. 基于云模型的水印关系数据库技术. 解放军理工大学学报(自然科学版)2003, 4(5):1~4

[21] 肖湘蓉, 孙星明. 基于水印的数据库安全控制研究. 计算机工程与应用, 2005, 6:175 ~178

[22] R. sion, et al. Watermarking Relational Databases. CERIAS Technical Report 2002~28

[23] R. G. van schyndel, A. Z. Tirkel, C. F. Osborne. A digital watermark. in: international Conference on image Processing, 1994(2): 86 ~90

[24] Rakesh Agrawal, Jerry Kiernan. Watermarking Relational Databases. Proceeding of the 28th VLDB Conference, Hong Kong, China, 2002:155~166

[25] 刘启原, 刘怡. 数据库与信息系统的的天. 北京:科学出版社, 2000. 131~137

[26] Cox. I. J, MattLMiller. A review of watermarking and the importance of perceptual modeling. SPIE Proceeding on Human Vision and Electronic Imaging, 1997, 30(16):92~99

[27] Cox. I. J, Jean-Paul M. G. Linnartz. Some General Methods for Tampering with Watermarks. IEEE Journal of Selected Areas in Communication, 1998, 16(4): 573~586

[28] Barni. M, Bartolini. F, Cappellini. V etal. ADWT-based technique for spatial - frequency masking of digital signatures. SPIE Proceeding on Security and Watermarking of Multimedia Contents, 1999, 36(57):31~39

[29] 王惠琴. 频率域图像数字水印技术的研究, 西安交通大学博士学位论文, 2002

[30] 王秋生, 变换域数字水印嵌入算法研究, 哈尔滨工业大学博士学位论文,

2001

[31] 牛夏牧, 数字水印处理算法及测试研究, 哈尔滨工业大学博士学位论文, 2000

[32] Cox. I. J, Kilian. J, Leighton. T et al. Secure spread spectrum watermarking for images, audio and video. IEEE Proceeding on International Conference on Image Processing, 1996, 3:243~246

[33] ChristineI Podilchuk, ZengWenjun. Digital image watermarking using visual models. SPIE Proceeding on Human Vision and Electronic Imaging, 1997, 30(16):100~111

[34] 李中献, 詹榜华, 杨义先. 认证理论与技术的发展. 电子学报, 1999, 27(1): 98~102

[35] Bender. W, Gruhl. D, MorimotoN et al. Techniques for data hiding. IBM System Journal, 1996, 35(3&4):313~336

[36] Van SchyndelR, TirkelA, OsborneC. A digital watermark. In: IEEE Proceeding on International Conference on Image Processing, Austin, Tex., IEEE Press, 1994:86~90

[37] Sion. R, Atallah. M, Sunil Prabhakar. Rights protection for relational data. Proceedings of the 2003 ACM SIGMOD International Conference on Management of Data San Diego, California: ACM SIGMOD, 2003. 98~109

[38] C. Asmuth, J. Bloom. A Modular Approach to Key Safe guarding. IEEE Transaction on Information Theory, 1983, 29(2):208~210

[39] Rakesh Agrawal, Jerry Kiernan. Watermarking Relational Databases. Proceeding of the 28th VLDB Conference, Hong Kong, China, 2002:155~166

[40] 萨师煊, 王珊. 数据库系统概论. 北京: 高等教育出版社, 1990年

[41] Zhi-Hao Zhang, Xiao-Ming Jin, Jian-Min Wang. Watermarking relational database using image. Proceedings of the Third International Conference on Machine Learning and Cybernetics, Shanghai, 26-29, August 2004, 1739 ~ 1744

[42] 卢开澄. 计算机密码学——计算机网络中的数据保密与安全. 第二版. 北京: 清华大学出版社, 1998. 20~26

[43] 王育民, 刘建伟. 通信网的安全——理论与技术. 西安: 西安电子科技大学出版社, 2000. 5~19

[44] 施荣华, 胡湘陵, 王国才. 一种不用大小比较的快速模乘算法. 小型微型计算机系统, 1999, 6: 468~471

-
- [45] 张浩, 黄敏, 曹加恒. 数据库水印中的标记算法. 计算机应用研究, 2005, 42(5): 42~44
- [46] 张宏林. Visual C++ 数字图像模式识别技术及工程实践. 北京: 人民邮电出版社, 2002. 215~236
- [47] 苏彦华等. Visual C++ 数字图像识别技术典型案例. 人民邮电出版社, 2004. 123~204

致 谢

岁月荏苒，转眼间三年的硕士研究生阶段即将结束，马上就要踏入人生新的驿站。回顾这三年的日子，紧张而又充实的学习生活历历在目，接触了许多新的知识，我无愧于心，面对新的生活，我更充满信心，充满了希望。

当这篇论文终于可以付梓成册的时候，我首先要把一份最深的谢意献给我的导师施荣华教授。施老师是一位密码学和网络安全领域的资深专家，他以自己深厚的理论造诣、丰富的实践经验和对前沿科学敏锐的洞察能力，为我的研究工作提供了有力的指导和帮助；他治学严谨、为人谦和的优秀品质，将成为我今后工作、学习和做人的好榜样。

同时，我还要感谢王国才老师，在我的论文写作过程中，给予我很多帮助。我的师兄和同学们，尤其是杨政宇、周诚、罗俊、姜悦、胥磊、周玉、康晶、胡芳，在我面对难题时给予帮助，学习之余共同讨论，工作之时并肩作战，使我受益匪浅。

衷心感谢我的父母和亲人。在我学习、工作的人生道路上，他们一直无私的给予我无微不至的关怀。成功时的赞许，失败时的鼓励，懒惰时的促醒，迷惘时的引导，以及生活上的悉心照料，正是这浓浓亲情和殷殷期望，始终鞭策着我，使我不敢懈怠。

最后，谨以本文献给我的父母和亲人！并向所有在我成长过程中帮助我、关心我、关注我的老师、同学和朋友们表示最真挚的谢意！

攻读学位期间主要的研究成果

论文:

[1] 胡斌, 施荣华, 姜悦. 一种改进的基于中国剩余定理的群签名方法. 计算机工程与应用. 2006, 42(24):115~117

[2] 胡斌, 施荣华, 姜悦. 一种基于密钥分存的关系数据库数字水印. 计算机应用研究. (已录用, 将于 2007 年 6 月发表)

[3] 姜悦、施荣华、胡斌. 基于智能卡的 RSA 数字签名算法. 现代计算机. 2006(1):42~45

项目:

[1] 2004.9~2005.3 中国铝业中州分公司设备管理系统 系统调研与设计

[2] 2005.1~2005.6 中国铝业河南分公司设备管理系统 系统调研与设计

[3] 2005.9~2006.1 中国铝业中州分公司电话线路管理系统 项目负责人, 并参与设计实现

[4] 2006.9~2006.12 中国移动常德分公司综合资源管理系统 项目负责人, 并参与设计实现