摘 要

数据泄露防御技术是通过一定的技术或管理手段，防止企业的指定数据或信息资产以违反安全策略规定的形式流出企业。现有的网络数据泄露防御方案主要基于关键词内容过滤、扩展名过滤、信息等级过滤，这些方案都存在一定的局限性和效率的问题。而基于指纹算法对数据流进行检测，可以提高执行效率，同时对防御内容有很高的匹配率，能够有效地阻止网络数据的外泄。

本文对数据泄露防御技术进行研究，以保证企业信息资产的安全为目标，基于 Winnowing 算法，提出了一种新的数据泄露防御算法。该算法以指纹模式来匹配文件，能以较少的资源消耗完成对资源的监控过程。所采用的指纹算法对需防御的数据进行矩阵求解，之后生成这段防御数据的“指纹”。当用户向外发送相关信息时，系统会提取部分数据流，通过指纹算法生成这段数据流的“指纹”，然后将两个指纹进行比较，如果达到预定的相似度则对数据流进行拦截，否则就允许数据发送出去。在实现方面，这种新的基于指纹的数据泄露防御算法可以高效地完成数据的对比过程，同时有较高的匹配性。

本文通过仿真实验证明了把指纹算法引用到数据泄露的防御中是较为可行和可靠的方案。可以预见，将该算法嵌入相应的监视系统中，可以有效地降低企业数据的流失。

关键词： 安全；数据泄露；防御；指纹算法

ABSTRACT

Data leakage prevention technology can prevent certain data or information asserts of the enterprises from leakage by a certain technique or management tools. At present, network data leakage defense program mainly falls back on keywords filtering, extension filtering, and leveled-information filtering and so on. These schemes have limitations and lack efficiency. The fingerprint algorithm used to detect data flow can improve efficiency. As it has a very high matching rate while filtering the content, it can effectively prevent the leakage of data in certain network.

This thesis studies data leakage defense technology. In order to guarantee the security of enterprise information, a new algorithm of data leakage defense is designed by taking advantage of Winnowing algorithm. The algorithm uses fingerprint pattern to match documents, hence less resources are consumed to monitor network. The fingerprinting algorithm can generate fingerprint of the defense data by solving data matrix. If a user sends out some information, the system will extract part of the data flow to generate a fingerprint by means of the fingerprint algorithm. Then, this fingerprint is compared with fingerprint of the defense data. If the similarity between them is as high as the expected value, the data flow will be intercepted, otherwise it will go smoothly. The algorithm can efficiently complete the process of data comparison with a high matching rate.

The results of experiment prove that applying fingerprint algorithm to data leakage defense is effective and reliable. It can be foreseen that the algorithm can be widely deployed in monitoring system to effectively reduce leakage of corporate data.

Keywords: security; data leakage; defense; fingerprint algorithm

缩略词

缩略词	英文全称	译文
B2D	Backup To Tape	磁带备份
B2D2T	Backup To Disk Library To Tape	多级备份
B2T	Backup To Disk	磁盘备份
CDP	Continuous Data Protection	持续数据保护
DLP	Data Leakage Prevention	数据泄露防御
DL	Disk Library	磁盘库
EDLP	Endpoint Data Leakage Prevention	终端数据泄露防御
FTP	File Transfer Protocol	文件传输协议
HDLP	Host Data Leakage Prevention	主机数据息泄露防御
HTTP	Hypertext Transfer Protocol	超文本传输协议
ILP	Information leakage prevention	信息泄露防御
LAN	local area network	局域网
MSA	Mail Server Agent	邮件服务代理
NDLP	Network Data Leakage Prevention	网络数据泄露防御
RPO	Recovery Point Object	恢复点目标
RTO	Recovery Time Object	恢复时间目标
VTL	Virtual Tape Library	虚拟磁带库

图表清单

南京邮电大学学位论文原创性声明

本人声明所呈交的学位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，论文中不包含其他人已经发表或撰写过的研究成果，也不包含为获得南京邮电大学或其它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。

研究生签名：张睿 日期：2009.4

南京邮电大学学位论文使用授权声明

南京邮电大学、中国科学技术信息研究所、国家图书馆有权保留本人所送交学位论文的复印件和电子文档，可以采用影印、缩印或其它复制手段保存论文。本文电子文档的内容和纸质论文的内容相一致。除在保密期内的保密论文外，允许论文被查阅和借阅，可以公布（包括刊登）论文的全部或部分内容。论文的公布（包括刊登）授权南京邮电大学研究生部办理。

研究生签名：张睿 导师签名：李晓辉 日期：2009.4

第一章 引言

1.1 课题的意义

网络的快速发展，使得网上的信息量成倍地增长，伴随而来的网络安全问题接踵而至。其中网络中企业的信息安全问题越来越受企业的关注。为了保护企业的保密信息不被有意或意外泄露，一种称为“数据泄露防御”(Data leakage prevention, DLP)，有时也称为“信息泄露防御”(Information leakage prevention, ILP)的技术便应运而生。

根据所部署的位置的不同，数据泄露防御方案可以分成基于网络的数据泄露防御方案(NDLP)和基于主机的数据泄露防御方案(HDLP)。目前大部分数据泄露防御方案都是基于网络的，有少部分是基于主机类型。基于网络的数据泄露防御方案通常部署在保存有保密数据的企业内部网络和外部网络的接口处，所针对的对象是进出企业内部网络的所有数据，如果发现有违反企业安全策略的数据流入流出，NDLP 便会将违规数据予以拦截或采取报警等其它行为。而基于主机的数据泄露防御方案则部署在存放敏感数据的主机上，当发现被保护主机上的数据被违规转移出主机时，HDLP 会采取拦截或报警等行为。

目前，在网络的数据泄露防御方面，已有人做了相关的研究。而且很多的研究侧重于企业外部的网络数据泄露防御，减少外部对企业的冲击。有人提出对企业内部的信息设置访问控制权限以及对信息进行加密，一方面可以减少重要数据丢失的可能性，另一方面即使重要的数据泄露也不会造成太大损失，因为数据进行了加密，以此防止数据泄露。例如文献[1-2]中提出了一个基于密码隔离的防信息泄露的内网安全模型，该模型利用访问控制和密码手段,合理控制了用户的行为,从而不会发生敏感信息被有意或无意地泄露出去的事故，具有一定实用价值。

目前，国内外对企业外部的网络安全防御及企业内部信息管理有较深入的研究，并取得了较好的研究成果，但针对企业内部的网络安全问题却研究甚少，而企业内部的网络安全问题正是网络数据泄露防御的重点。本课题从企业内部出发，采用指纹算法，对企业内部的信息进行保护，解决企业内部信息的安全问题。

1.2 论文的主要工作

本文针对企业内部信息内容的保护对网络数据的泄露防御进行研究。分析网络数据泄露防御已有的研究成果及特点；基于泄露防御算法的已有研究成果，确定了借助关键词内

容过滤的算法来提高关键数据的匹配性以及解决匹配的效率问题的研究思路，以使数据泄露防御达到更好的境界。在此基础上对数据泄露防御方案进行了研究，提出了一种新的关键词内容过滤算法，该算法在原有类似算法的基础上进行改进，针对不同的信息内容，能够更好地进行匹配。新的数据泄露防御方案因其算法的执行时间较短，能使防御体系的整体效率有所提高。

所设计的新的数据泄露防御方案分为以下三个步骤：

首先，用户对系统输入相关信息。信息内容由用户自由设置。用户把自己担心有可能会泄露的信息内容填入系统中，系统会在用户填入内容后进行防御。如果系统所需要防御泄露的内容为空，则系统对所有信息不进行处理。

其次，当用户填入了相关信息后，对信息进行处理。利用新的关键词内容过滤算法对所填入的信息进行处理。完成处理后，生成指纹矩阵。

最后，利用所产生的指纹对资源进行比较，对相似度过高的信息进行拦截等处理。本文设计的算法，能够有效提高资源匹配的速度，提高匹配效率。

本文还用C++语言编写了相应的仿真程序，仿真结果表明：这种将指纹算法引入数据泄露防御系统的设计思想能够有效提高匹配的效率和适应性。

1.3 论文的组织

第一章 引言：首先介绍了本课题的意义，确立了本文的主要研究内容；随后简要概括了本文所做的主要工作。

第二章 数据泄露防御技术概述：首先介绍了数据泄露防御算法的演变历史及其特点，然后对数据泄露防御技术的系统结构作了概括描述，叙述了数据泄露防御研究的现状，重点介绍了国内相关技术的研究项目及其发展过程；同时对数据泄露防御运行进行概述：叙述了数据泄露防御的基本原理；介绍了数据泄露防御的种类；最后，对数据泄露防御系统的结构做了一个较全面的阐述。

第三章 数据泄露防御算法及应用原型：介绍了关于数据泄露算法的起源，以及使用这些算法的原型系统。

第四章 数据泄露防御算法的研究：首先介绍了几种经典调度算法的思想；然后重点介绍了本文涉及的相关算法及所提出的新算法，给出了算法的设计思想，对算法进行了描述。

第五章 仿真及结果分析：用C++语言编程实现了指纹算法，对相应算法进行了仿真，

分析了仿真结果。

第六章 总结与展望：对本文工作进行了总结，并指出需要进一步改进的内容。

第二章 数据泄露防御技术概述

2.1 数据泄露防御概念

随着技术的发展，企业的业务流程及信息处理越来越依赖于 IT 设施，企业的信息也从最开始的以纸介质为保存媒体逐渐转变为纸介质和电子介质保存的局面。许多企业出于快速处理信息的考虑，甚至将所有的信息进行电子化，所有的业务流程也依赖于 IT 设施。因此，IT 设施的正常运作及电子信息的良好保护便成为企业业务顺利进行和发展的关键因素之一。信息在企业业务中扮演着如此重要的角色，因此我们可以认为信息也是一种资产，并称之为信息资产。信息资产尽管是无形的，但由于它包含了大量的业务数据、客户信息、商业秘密等对企业的业务乃至存亡密切相关的内容，所以信息资产也面临大量的威胁和风险，包括有意或无意的销毁、黑客攻击、恶意软件造成的数据丢失、内部人员的泄露等。这些威胁和风险中，最有可能发生并造成严重后果的便是保密信息及有关的数据有意或意外的泄露，一旦发生数据泄露事件，企业不但要承担保密数据本身价值的损失，严重的时候还会影响企业的声誉和公众形象，并有可能面临法律上的麻烦。2007 年在美国 TJX 零售公司发生了 4500 多万客户的信息卡及保密资料丢失，进而导致其客户和银行业对其提起诉讼，最终 TJX 公司需要向客户赔付 1.01 亿美元，并承受严重的企业声誉损失。因此，如何有效地防止企业数据泄露，尤其是终端数据泄露也就成为企业安全防护链条中的关键环节，这也使得以数据为主的内容管理市场持续增长。由于企业的安全威胁一直在发生转变，网络安全性问题已不再只是外部攻击和简单的病毒防护。虽然目前企业花费大量资金和精力为企业构建起庞大的网络架构和安全防护体系，但是真正威胁企业发展和生存的往往是来自内部网络的安全隐患，而且将造成远大于黑客攻击和病毒骚扰的危害。这种潜在的安全隐患对于企业来讲，数据的泄露危害是最大的。图 2-1 和图 2-2 显示了被泄露数据的类别以及数据泄露的方式[3]。图 2-3 给出了不同的数据泄露的形式[4]。

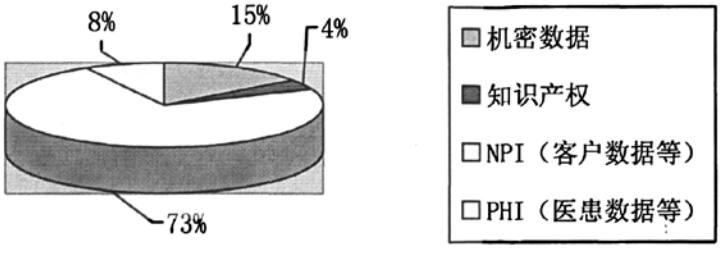

图 2-1 数据泄露的信息类别

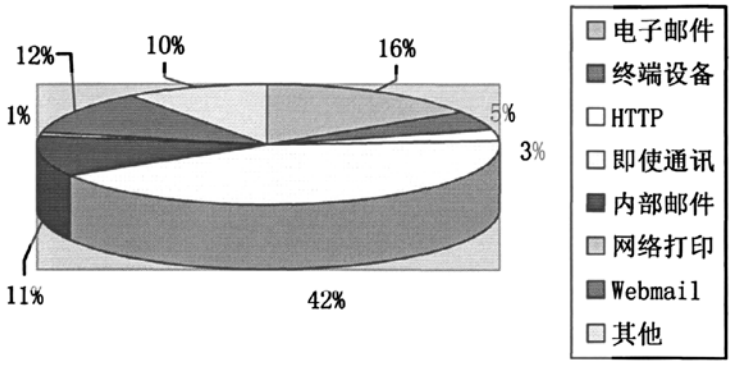

图 2-2 数据泄露的方式

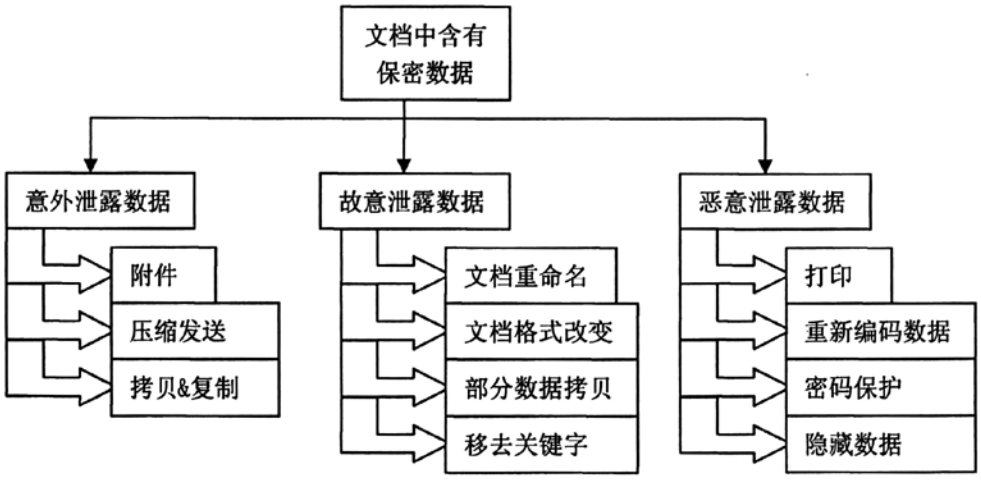

图 2-3 保密数据泄露的三种形式：意外泄露、故意泄露、恶意泄露

数据泄露防御技术(Data Leakage Prevention，DLP)是通过一定的技术或管理手段，防止企业的指定数据或信息资产以违反安全策略规定的形式流出企业。它是一种保护企业的保密信息不被有意或意外泄露的技术。数据泄露防御技术可以直接防护企业关键信息的泄露，从而避免由于错误点击而给间谍软件、键盘记录程序以可乘之机，从而造成安全风险。利用数据泄露防御技术，即使无法阻止恶意软件的安装，也可以在企业机密信息外泄之前，阻止恶意行为，最大限度地保护企业的信息资产。系统将载有客户信息等的数据作为机密数据进行管理，控制有可能导致信息泄露的访问操作。系统的机密信息对来自外部的访问是全部保密的，对内部的机密信息访问实行按职责来进行访问控制。在机密模式下，通过策略使有可能导致信息泄露的机密数据的保存以及复制、打印等操作得到控制。这样可以防止不慎将机密数据泄露出公司。另外，为了防止使用者的非法操作，会将客户端的

访问操作履历以及访问机密数据记录到日志服务器中，而在普通模式下，没有控制访问数据，可以访问并打印普通数据，不会影响业务处理效率。数据泄露防御技术不仅可以用于企业内部网络上，也可以应用于多个点的外部网络，把分布在不同地点的各个服务器和客户端保护起来。利用数据泄露防御技术，不仅可以防止组织内部的用户“故意”或者“由于疏忽”导致信息通过内网或外网的泄露，而且还可以防止利用恶意攻击盗取数据的行为。

2.2 数据泄露防御方案的分类及其特点分析

根据所部署的位置的不同，数据泄露防御方案可以分成基于网络的数据泄露防御方案(NDLP)和基于主机的数据泄露防御方案(HDLP)[5]，这和入侵检测系统的分类是很相似的。目前大部分数据泄露防御方案都是基于网络类型的，有少部分是基于主机类型的。基于网络的数据泄露防御(简称 NDLP)方案通常部署在保存有保密数据的企业内部网络和外部网络连接的接口处，所针对的对象是进出企业内部网络的所有数据，如果发现有违反企业安全策略的数据流入流出，NDLP 便会将违规数据予以拦截或采取报警等行动。而基于主机的数据泄露防御(简称 HDLP)方案则部署在存放敏感数据的主机上，当发现被保护主机上的数据被违规转移出主机时，HDLP 会采取拦截或报警等行动。

数据泄露防御方案可以简单地分成以下三类[6]：

(1)关键词内容过滤

此类数据泄露防御方案根据网络或主机上所保存和流动的数据内的特定关键词进行过滤，来确定是否存在不符合企业安全策略的数据流动，比如根据“保密”这个关键词进行过滤。

(2)扩展名过滤

此类数据泄露防御方案根据网络或主机上所保存和流动的数据文件的扩展名进行过滤，判断是否存在不符合企业安全策略的数据流动，比如根据“以 DOC 为扩展名的文件不能流出企业内部网络”这样的规则进行过滤。

(3)信息等级过滤

此类数据泄露防御方案根据网络或主机上所保存和流动的数据所属的保密级别进行过滤，判断是否存在不符合企业安全策略的数据流动，比如标记为“秘密”等级的数据不能流出企业内部网络。这种数据泄露防御方案的实现需要部署此方案的企业先对自己的所有信息资产进行分类和标记，是最有效同时也是最费力的数据泄露防御方案。图 2-4 给出了采用信息等级过滤方案的防御系统对文档的保护过程。通过对大量信息设置等级，限制高

等级信息的泄露，从而达到数据泄露防御的效果。该系统需要输入文档的相关信息，利用部分内容对数据库中的信息进行匹配，从而获得文档的信息等级，之后进行保护。

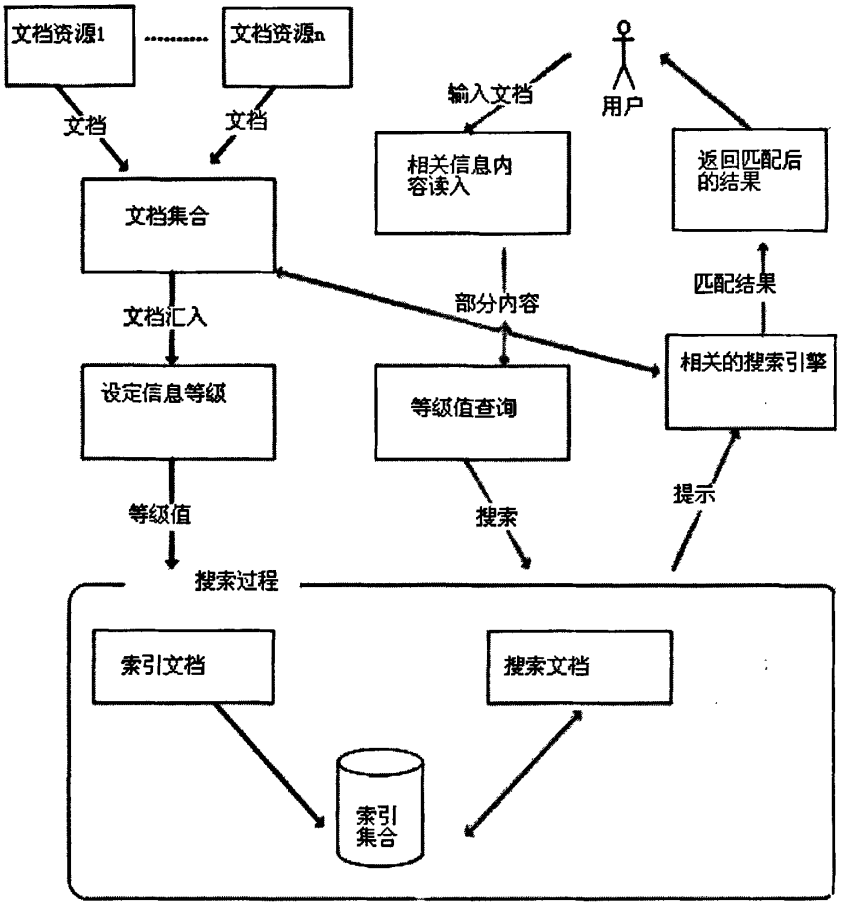

图 2-4 信息等级过滤的过程示意图

依据上述 3 种方案来实现对数据泄露的防御，都存在着局限性和效率的问题。信息等级过滤需要对大量信息进行等级的设置，从而使得实施较为复杂，对海量数据进行配置，既有效率问题的存在，也会使得匹配性下降不少。对于扩展名过滤，系统不可能把所有的文件扩展名都禁止掉，因此只要用户不小心把修改扩展名后的文件发送出去，系统依然无法对其进行有效的阻止，由于存在无法辨识的修改后的文件类型，使得这种数据泄露不能有效实施。关键词内容过滤在传输大型文件时，无法通过关键字有效地捕捉机密文件，因此可行性也大打折扣，也存在着局限性和执行效率问题。

2.3 数据泄露防御系统概述

2.3.1 数据泄露防御系统的部署

如图 2-5 所示，企业的敏感数据通常存放在文件服务器上(Company sensitive data)，用户通过自己的终端(LAN Desktop)进行访问。LAN Desktop 周边的打印机、可移动存储设备、摄像头、调制解调器和无线网络便是潜在的本地数据泄露源，企业可以通过在用户的终端上部署基于主机的数据泄露防御方案来进行控制。LAN Desktop 和 Internet 进行的通讯，尤其是 E-mail、FTP、HTTP 和即时通讯是最常见的网络数据泄露源，在这种场合企业就需要在内部网络和 Internet 接口处部署基于网络的数据泄露防御方案进行控制。

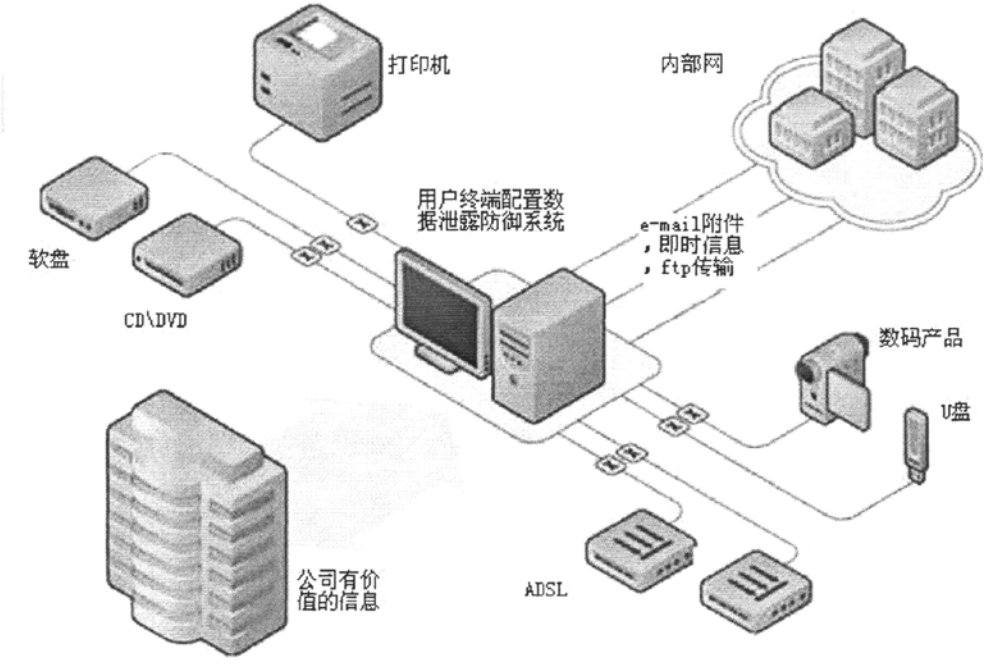

图 2-5 企业数据泄露防御系统的部署

如上图所示，在用户自己的终端部署数据泄露防御系统，可以避免以上各类泄露方法泄露公司有价值的数据。数据泄露防御系统可以阻止这些数据被移动盘符复制后泄露，可以防止用户把这些数据放入 E-mail 的附件中发送出去，可以阻止这些数据通过即时信息泄露出去等。通过部署数据泄露防御系统，可以减少企业由于信息泄露所产生的损失。

企业如果部署数据泄露防御方案，将可以在现有的安全方案上，再为信息资产添加一层安全保障，即使在防火墙、反病毒等方案失效的情况下，仍能最大程度地保护企业内的

保密数据不会因为有意或无意的原因而泄露，同时由于数据泄露防御是防止数据泄露到外界，它也对目前愈演愈烈的内部人员违反安全策略导致的数据泄露有相当好的防御效果。

2.3.2 企业中数据泄露防御系统的发展

信息是一个企业生存与发展的基本元素。对企业来说，凡是企业在进行大量人力、物力投入后所得到的数据信息资源，都可以列入机密信息的范畴。如：技术图纸、财务报表、账户信息、客户资料、渠道资源等。这些信息散落在企业的各个角落。对于早期的防御系统，主要是从身份认证、数据加密、权限管理、日志审计、数据备份等五个步骤进行系统的部署。

（1）身份认证

身份认证是安全管理体系建设的基础。首先要区分出哪些用户是经过授权的，进而加强对终端登录的安全管理和控制。企业中的用户只有通过合法身份认证，才能进入企业内部网络访问企业的内部文件和核心数据信息，以此实现企业级文件系统的安全性。如果不对用户身份进行安全认证，任何人都拥有完全的开放权限，那么企业和机构的安全管理将无从谈起。

（2）数据加密

数据加密的技术是在身份认证的基础上进一步确保信息的安全。目前，比较先进的加密技术是基于操作系统底层的驱动级透明加解密。这种驱动级的透明加解密技术与一般的应用层加解密技术相比，具有安全、可靠、稳定、效率高的特点，并且所有加解密过程均在系统底层完成，不改变用户原有的操作习惯。对数据信息使用权限的管理是一个权限区分和细分以及依据权限进行访问控制的过程。

（3）权限区分

这是指对于数据信息使用者的权限进行区别对待，区分出哪些人是经过授权的，哪些人是没有经过授权的，做到只有经过授权的用户才可以使用数据，这个权限区分的过程，将数据信息的应用范围缩小到了授权人群范围之内。权限细分，在权限区分的基础上进一步细化。包括：只读权限、编辑权限、复制权限、存储权限、完全控制权限、打印权限、解密权限、可以应用的时间权限等等。总之，可以做到让指定的用户，在指定的时间，对指定的数据信息进行指定的应用操作。

（4）日志审计

日志审计可以对所有的文件操作进行详细记录，包括：文件创建、编辑、存储、传输、

删除等整个生命周期的全过程，为事后追踪提供有力的依据。

（5）文件备份

文件备份是整个信息安全管理过程中的重要一环，也是企业在面临数据毁坏这种致命安全风险时的有力保护手段。它能够帮助企业保护其内部核心数据信息的完整性和安全性，提升企业重要文档的抗破坏能力。

以上的防御技术都存在着一些缺陷，早期的防御过分依靠用户行为，使得其可行性变得很低。对用户权限划分，在一定范围内减少了信息的丢失，但是却无法保证高权限用户的行为。用户的权限越大，所带来的危险性也越高。加密技术在保护文件信息方面起了重要的作用，可以有效防止信息的泄露，但加密和解密本身带来的系统开销却无法避免；同时也无法阻止用户发送解密后的信息出去。因此，近代的防御技术开始侧重于对内容的监测。

（6）内容监测

通过对出入网络的数据以及个人主机的数据进行监测，拦截违规数据的出入，完成对数据泄露防御的过程。以内容监测为防御策略可使数据泄露防御系统的实时性得到提升。相对于其它策略，可以更高效率地完成数据泄露防御任务。

在大规模的企业信息网络中，如何对有价值信息进行拦截的同时不影响用户的正常工作，如何实现在网络中对数据进行高效率的匹配尽可能减少有价值信息的泄露，是基于内容监测的数据泄露防御技术面临的一个重要课题和挑战。

2.3.3 数据泄露防御系统的数据保护

保护数据不丢失，可分为多个保护级别。这些级别是根据数据的可用性包括 RTO(恢复时间目标:使系统恢复所需要的时间)和 RPO(恢复点目标:可接受的数据丢失量)来划分的。

保护级别越高，RTO 和 RPO 也就越少；不过实施的相对成本也就越高。这些级别分别是备份、本地复制、远程复制和实时连续复制。

1．备份

备份是为了在系统出现故障时进行数据恢复，包括磁带备份和磁盘备份等。磁带备份(Backup To Tape，B2T)是经典的备份方式，将数据备份到磁带上。磁带备份具有容量大、成本低的好处，但是备份和恢复的时间较长，而且恢复成功的概率较小。任何备份技术的恢复机制都需要一个和备份过程相反的过程，这个过程一般时间会很长，如果用户对恢复

时间 RTO 要求很高，采用磁带备份就捉襟见肘了，为了满足要求就必须采用磁盘备份。

磁盘备份(Backup To Disk，B2D)指的是把磁盘阵列作为备份设备，它改善了备份和恢复的性能，提供高可靠性和可用性。

虚拟磁带库(Virtual Tape Library，VTL)，又称为磁盘库(Disk Library)，用磁盘来存储数据，并且能够仿真成物理磁带库。这种备份方式是磁盘备份的主流方式，它的优点是：相对磁带备份的性能大幅提高；同时，还能沿用原有的磁带备份软件和备份策略。

还有一些衍生的备份方式，例如多级备份(Backup To Disk Library To Tape，B2D2T)，先备份到虚拟磁带库，再备份到磁带。采用这种方式可以充分利用存储空间，把信息存放到适当的存储设备上。

备份关注的有以下几点：备份的数据量；备份和恢复的速度和可靠性；备份、恢复操作的方便性等。为了减少备份数据量、加快备份和恢复的速度,可以通过全面的备份、恢复和归档策略及采用新一代的重复数据删除备份技术。其中，全面的备份、恢复和归档把不活动的、最终形式的数据进行归档，以缩小生成数据的大小，这样就减少了备份的数据量，恢复时间也更短，性能更加稳定，并且可利用分层存储的优势。

2．本地复制

复制技术会为源数据制作一份副本，这个副本除了能够提供近于即时的恢复外，还可用于无中断备份、决策支持、应用程序测试和开发、第三方软件更新等；它们的目的在于保证系统数据和服务的“在线性”，即当系统发生故障时，仍然能够提供数据和服务，使系统恢复正常。

本地复制包括快照和克隆。快照是数据在某个时间点（拷贝开始的时间点）的映像。它是基于指针、节省空间的逻辑拷贝，通常要求少于 30% 的源卷容量，速度较快。克隆是数据的完整复制，是真实的拷贝，它的过程较慢，而且每个副本需要与源卷容量相同的存储空间。

3．远程复制

远程复制为业务连续性和灾难备份提供了强有力的保证，通过远程站点故障切换确保数据和系统的可用性，在中心停机后数分钟内，数据能够恢复，业务继续运营。远程复制包括同步远程镜像和异步远程镜像。同步远程镜像是指通过远程镜像软件，将本地数据以完全同步的方式复制到异地，每一本地的 I/O 事务均需等待远程复制的完成确认信息方予以释放。异步远程镜像保证在更新远程存储系统前完成向本地存储系统的基本 I/O 操作，而由本地存储系统提供给请求镜像主机的 I/O 操作完成确认信息，远程的数据复制以后台同步的方式进行。

4．实时连续复制

这是一种持续数据保护（Continuous Data Protection，CDP）技术，它的关键是持续。就给定的数据集而言，CDP 提供恢复点的连续体，能够存取任何时间点上的数据，而不仅仅针对那些由快照流程预先确定的特殊时刻。CDP 允许数据恢复到特定的时间点之前，而不是恢复到预先确定的时间点上。恢复点在事件发生后选定并动态重建。它提供了粒度无限的恢复点（RPO）。

2.4 数据泄露防御技术的研究现状和发展趋势

2.4.1 数据泄露防御技术的研究现状

数据泄露防御是一套完整的体系，也是多种系统的集成，用以解决不同类型的用户的不同需求。数据泄露防御的方案多数是基于网络的，这主要是由于用户在网络环境下信息的泄露率十分高。在网络环境下，不论是用户自身导致数据泄露，或者是外部因素窃取数据使得相关数据泄露，其危害性都比较严重。因此数据泄露防御技术逐渐成为防火墙、杀毒软件之后的第三类主流安全产品。

在我国，数据泄露防御首先是应用在政府和军队、军工企业，例如外交部、解放军二炮和酒泉卫星发射中心。但是从目前市场应用的分布来看，使用最多的是制造企业，占到总数的 48%，其次是通信制造（含手机研发行业）占 22%，大型企业集团也非常重视 DLP，占到总数的 18%。如图 2-6 所示。

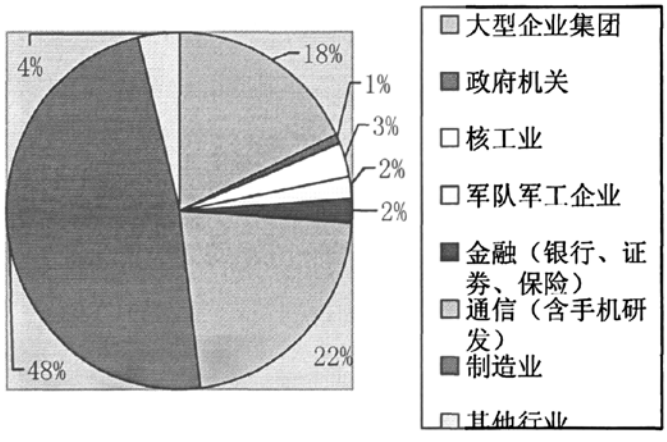

图 2-6 数据泄露防护在行业中的分布

目前对数据泄露防御技术进行研究的公司包括 Reconnex（被 McAfee 收购），Verdasys、

Vericept、 Websense、RSA（被 EMC 收购）和 Symantec 等，都是通过在网络端部署数据泄露防御机制来减少企业的信息泄露，同时不断地增加数据泄露防御的范围。今后的数据泄露防御技术的研究目标是，能够监视用户的操作，使得系统不仅能控制网络方面的信息，对于单个机器也能减少其信息的泄露。例如：阻止用户拷贝有价值的信息至移动存储器上，阻止用户对有价值信息进行复制等。

数据泄露防御系统要截获违反规则的文档，则要求系统中要有较好的文档匹配算法，用来筛选不同的文档。不少学者提出了较有建树和参考价值的思想和方法。比如文献[7-8]提出了基于字符串匹配的方法，该方法要求从文档中选取一些字符串，然后把这些字符串映射到散列表中，一个字符串对应一个数字，最后统计散列表中相同的字符串数目或者比率，根据相似性度量公式，得出文档的相似度，从而判断文档是否相同。文献[9-10]提出了基于词频统计的方法，该方法源于信息检索中的向量空间模型。这类方法用向量来表示数字文档：(W1, W2, W3, …, Wn)，其中 W_i 表示第 i 个特征的权重。它的工作过程是首先把数字文档分解成单词序列，然后统计出每篇文档中各个单词的出现次数，然后根据单词频度构成文档特征向量，最后采用点积、余弦或者类似方式度量两篇文档的特征向量，以此作为文档相似度的依据。文献[11]提出了以串匹配方法检测相同文档的算法。通过分解，可以把文档划分成若干个文本块，在文献[11]中每个文本块是长度固定的字符串，再通过匹配文档间的文本块，依据相似性公式得出待检测文档与数据库中已有文档的相似程度，从而判断待检测文档是否与数据库中的文档数据一致。该算法首先定义一个散列函数，保证不同文本块的散列值发生冲突的可能性很小。然后通过散列函数把待检测文档和数据库中已有文档的文本块映射成比文本块短得多的散列值。再通过比较待检测文档与数据库中已有文档的散列值来确定文档是否匹配。如果两个文本块的散列值不相等，说明这两个文本块的文本内容一定不相同。否则，就认为这两个文本块的文本内容是相同的。

总而言之，数据泄露防御技术的研究在不断加深，该技术能够对企业所需要保护的数据信息进行有效的防御泄露，降低企业丢失有价值信息的可能。同时另一方面也促进了搜索引擎的发展，因为数据泄露防御算法核心是利用文档的匹配算法，在一些方面和搜索算法中利用字符串匹配还是很相似的。随着技术的发展，不同的领域的技术进行很多的融合，也使得数据泄露防御系统能起到的作用也在不断的增大，对网络安全的发展有极大的帮助。

2.4.2 数据泄露防御技术的发展趋势

数据泄露防御的目的是为了减少数据的泄露，其防御方式由原来的多样化逐渐走向单一化。最初的数据泄露防御技术主要是依靠约束用户个人的操作行为，使用户的操作行为限制在有限的行为中，以此减少数据泄露的可能性，或者封闭外部的条件，以此达到防御泄露。例如：禁止用户拷贝行为，禁止传送文件至网络，屏蔽外网等种种手段。但弊端也比较明显，限制了用户的行为，不仅影响用户的操作性，其本身也有其不合理性，属于过分地保护本身资源。于是数据泄露防御的方式逐渐转换成其它方式的数据防护，侧重于记录日志、用户权限、数据加密等方面。记录日志通过对用户的行为进行记录，防止数据泄露，但本身的滞后性使得该方式并不实用。设定权限行为是通过事先设定好用户的权限，使得某些用户无法看到较为机密的信息，以此达到需求，但其复杂性以及用户使用的不方便性依然给用户带来很多麻烦。随后防御技术侧重于加密技术，加密技术则是通过对企业数据的加密使得信息无法被其他人看到，即使不小心把数据发送出去，没有解密的情形下，别人依然很难看到信息。能够较好的阻止泄露，但是其加密解密的开销比较大，对大量的数据进行加密对企业以及用户来说，都是比较麻烦的。同时，加密技术要求用户一定要保护好密钥，07 年 TJX 公司由于密钥问题，导致数据大量泄露。这使得防御技术由侧重加密技术逐渐转向为通过在网络端或个人客户端进行防御系统的部署来防止数据泄露。

新的数据泄露防御逐渐以内容监测为重。输出内容的监测技术通常以智能网络专用设备的形式出现。这类设备执行政策驱动的控制，并在一些情况下使用行为分析来确定雇员是否让保密数据面临风险。这些设备发出报警，将可疑的输出内容放在一个存储区内，或者立即阻止可能使敏感数据面临风险的活动。现在，大量用户发现，输出内容监测器对于他们的分层安全架构越来越重要。

在内容监测的模型中，数据泄露防御的部署位置和内容匹配算法是模型的核心[12]。目前，一种基于指纹的防御成为新的研究方向。这种方案基于指纹算法对数据流进行检测，首先对相应的文件做处理，生成文件的特征值（文件的指纹），之后和数据库的机密文件的指纹进行比较，进行评估后，根据文件匹配度进行拦截。这样的处理方式，既可以完成对关键字的捕捉过程，也能对大型机密文件进行有效扑捉，还可以对文件类型进行匹配，在防御程度上要高于上述的三个防御方案。此外，指纹匹配方案本身对防御内容有很高的匹配率，可以有效地提高效率，能够有效地阻止网络数据的外泄。指纹算法可以对所防御的数据进行矩阵求解，之后生成这段防御数据的“指纹”。如果有用户把相关的信息发送出去，系统会提取部分数据流，通过指纹算法生成这段数据流的“指纹”，然后进行比较，

如果相同则对其进行拦截，否则就允许数据发送出去。在实现方面，可以高效的完成数据的对比过程，同时有较高匹配性，因此把指纹算法引用到泄露数据的防御，可以认为是可行而且可靠的方案。

本文研究的数据泄露防御技术，侧重于对内容进行监测，运用指纹算法，对敏感内容进行提示以及拦截，以此达到数据泄露防御的效果。

2.5 基于内容监测的数据泄露防御系统的结构

基于内容监测的数据泄露防御系统对内容匹配的信息进行拦截，对触犯规则的信息予以拦截和提示。在内容监测的模型中，数据泄露防御的部署位置和内容匹配算法是模型的核心。根据数据泄露防御的部署位置不同，可分为网络数据泄露防御模型以及本机终端数据泄露防御模型[13]。两种防御随着部署位置的不同，所起到的作用也不同。网络数据泄露防御侧重对发送出网络的数据进行监测，如果发现违规数据则进行拦截和提示。本机终端的数据泄露防御侧重对本地数据的流向进行监测。如果用户试图拷贝机密文件到移动设备上，会被防御系统拦截，同时提示用户。以此达到数据泄露防御的效果。

如果侧重防御数据流向网络的企业及用户，会选择在网络端进行部署数据泄露防御机制，这样可以阻止用户把有价值信息无意识地发到网上。如果侧重于本地系统的保护，为防止对机密文件进行移动或复制，可以选择本地终端数据泄露防御系统的部署。可以使得本地用户对机密文件的操作都得到应有的提示。同时，由于企业的需求，研究人员逐渐试图把两种部署的防御模型整合到一起，使得数据泄露防御能够既对网络数据进行防御，又能对本地数据泄露进行防御。

2.5.1 网络数据泄露防御（NDLP）模型

如图 2-7 所示，这样的方案主要侧重于离线检测，以及与对应的邮件服务代理(Mail Server Agent, MSA)和 Web 代理服务器 Web Proxy Server 相集成，以此达到网络数据泄露防御的效果。网络数据泄露防御侧重于在网络网关接口处部署，通过对数据的监测，对触犯了规则的数据进行阻止并提示，以此减少数据在网络中的泄露。

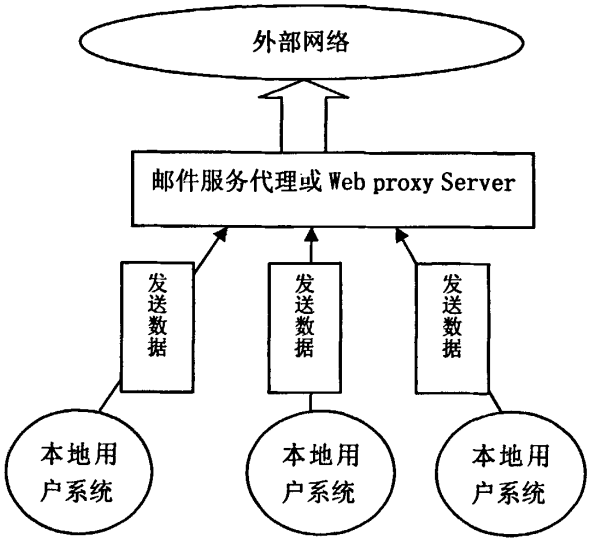

图 2-7 NDLP 的部署

网络数据泄露防御的目标就是减少有价值信息被用户无意识的发送出去，用户需要把有价值信息的拷贝放入到 NDLP 中进行处理，NDLP 会对存入的文档或者数据文件进行处理，生成文件的特征值。之后当数据试图发送至网络时，NDLP 会把这些数据和数据库的有价值信息的指纹进行比较，进行评估后，根据文件匹配度进行拦截。这样的处理方式，既可以完成对关键字的捕捉过程，也能对大型机密文件进行有效扑捉，还可以对文件类型进行匹配。

网络数据泄露防御采用的算法包括基于词频统计的方法、基于字符串进行匹配的全文匹配方法、通过特定的 hash 算法进行文档之间的匹配等。采用合适的匹配方案对防御内容会有很高的匹配率，对有效地阻止网络数据的外泄非常重要。

2.5.2 本机终端数据泄露防御（EDLP）模型

把 DLP 设置在服务器端，监控客户端。这样的防御方案主要侧重于利用分布计算、本地协议的解析、解析加密协议的数据，之后对数据泄露进行防御。对于单个用户，DLP 也可以设置在本机中，即服务器和客户端都是同一台机器。如图 2-8 所示，企业可通过在终端部署 DLP，减少员工把机密文件带出公司的可能，减少文件因为带出公司而泄露的危险，同时记录用户对机密文件的操作行为，以此防止用户的疏忽导致文件被无意识地篡改。这样，可以减少本地系统把机密数据泄露出去的概率，提高本地系统的安全性，同时保证了企业信息的安全。

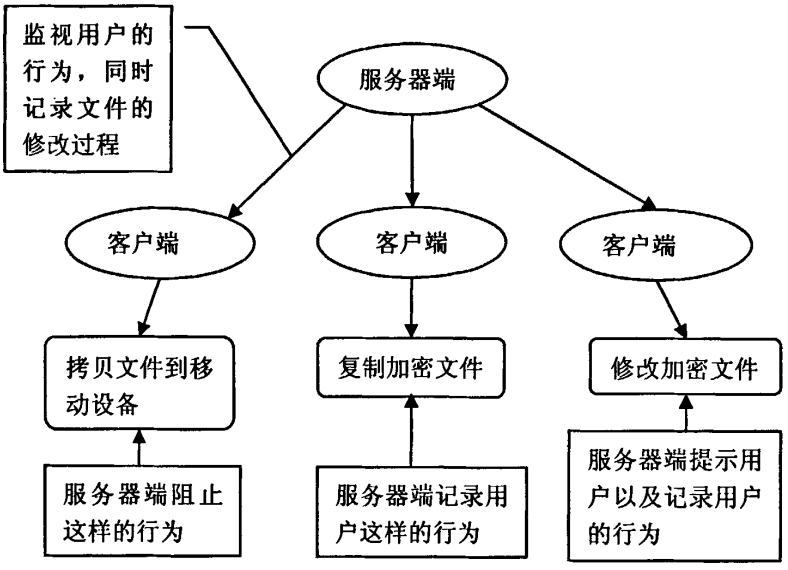

图 2-8 EDLP 的部署

本地终端数据泄露防御的算法和网络数据泄露防御的算法相似，用户把有价值信息的拷贝放入到泄露防御的系统中，之后对这些信息进行处理，生成这些信息的特征值。之后对有相同的特征值的文件进行监视，一旦用户触犯了对应的法则，系统会提示用户或者阻止用户，保证用户不会在无意识的情形下泄露公司的机密信息。系统会记录用户对机密文件的操作行为，有利于以后对文件的操作进行审查。

2.5.3 数据泄露防御的混合模型

本地的防御系统主要是对本地的用户泄露进行防御，缺点是不能够完全监视数据的发送过程；网络防御的优点是侧重于监视网络端的数据输入输出，有效地减少数据从网络中泄露出去，缺点是对本地系统泄露数据缺乏有效的防范手段。因此有必要将两种模型综合运用，以便同时解决本地和网络的数据泄露防御问题。

目前，为了使得企业使用更方便，防御范围更加广泛，不少从事数据泄露防御研究的公司已经开始逐步把网络防御和本地防御整合到一起，整合后的系统对企业的日常工作会起到更高级的保护，保护的范围的增加，能够使得企业机密数据的泄露更少。图 2-9 是两种数据泄露防御模型整合后的混合模型。

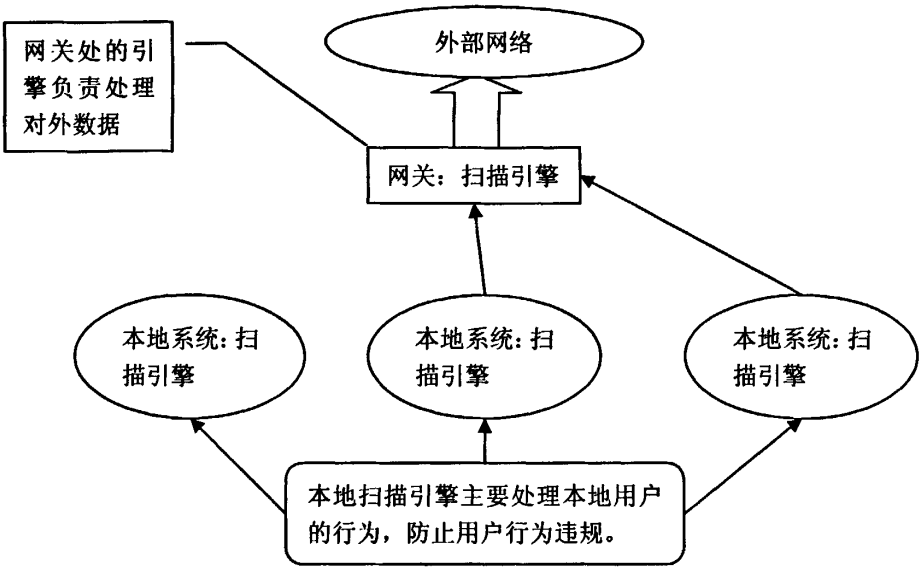

图 2-9 数据泄露防御的混合模型

整合后的模型着重于用扫描引擎处理各种事务，扫描引擎的位置不同，所处理的行为也不同。放置到客户端的扫描引擎主要负责对用户的行为进行监测，保证本地数据不会泄露。放置到网关处的扫描引擎主要对数据的内容进行监测，防止有价值的信息被发送出去。混合后的模型既保证了对本地数据流失进行监控，同时又能够实时地保护网络数据，从而有效地减少了数据的泄露。

引擎本身用的算法依然是之前的指纹算法，指纹算法本身具备高效的工作效率，同时有较高的匹配率，能够在复杂的环境下高效地处理文档数据。

2.6 本章小结

本章对数据泄露防御技术进行了综述，介绍了数据泄露防御的相关概念、数据泄露防御方案的分类及特点、数据泄露防御系统的组成；概述了数据泄露防御技术的研究现状和发展趋势。鉴于基于内容监测的数据泄露防御技术是较新的技术，本章对其系统的结构进行了重点介绍了，分析了相关模型各自的特点。

本文后续章节对数据泄露防御技术的研究，将侧重于对内容进行监测，运用指纹算法，对敏感内容进行提示以及拦截，以此达到数据泄露防御的效果。

第三章 数据泄露防御算法及应用原型

如上一章所述，在内容监测的数据泄露防御模型中，内容匹配算法是模型的核心之一。用于内容监测的数据泄露防御系统的算法是由文档匹配转换而来的。通过使用文档匹配算法，检测进出系统的数据是否触发违禁条件，之后对该数据做出相应的处理。文档匹配模型最初用于剽窃检测系统，以下是这些系统的相关介绍。

3.1 现有的主要原型系统

3.1.1 Plague 系统

Plague 系统[14]是 G.Whale 于 1988 年开发的，Plague 的检测分为三个阶段：第一个阶段：将文本转换成描述其结构特征的标识序列和结构度量列表。第二个阶段：比较结构特征，结合使用特定的函数找到最相似的语句端。预期大多数文本都是不相似的，如果有语句段是相似的，将其留到下一个阶段进行处理。第三个阶段，使用最长公共子序列算法（LCS）的改进算法对标识符序列进行比较。

Plague 系统使用了结构度量技术，且比较了程序间更详细的结构信息。其缺陷如下：

①Plague 系统仅可以用于 PASCAL, Prolog, Bourne Shell 和 Llama，如果要扩展到对其它语言的检测，工作量会很大；

②Plague 系统的检测结果是两列按 H 和 HT 索引排序的列表，需要对其作进一步的解释，不能做到一目了然；

③Plague 系统执行效率不高，又依赖于太多的 UNIX 工具，因此可移植性较差。

3.1.2 MOSS 系统

MOSS（Measure Of Software Similarity）[15]系统是由 Alex Aiken 于 1994 年开发的，主要用于检测 C，C++，Java，Pascal，Ada，ML，Lisp，Scheme 等八种编程语言编写的程序代码的相似性。MOSS 抄袭检测系统可以检测增加空格、增加无关语句、调整语句序列等的抄袭行为[16]。它使用的算法是基于文献[17-18]的串匹配算法。该算法可描述如下：

①程序分割成长度为 K 的邻接子字符串，参数 K 由用户自己定义。如：k=2，则“left”可以分割成“le”、“ef”、“ft”；

②每一个长度为 k 的子字符串放入相应的哈希表中；

③在哈希表中选择一个子集作为程序的指纹；

④比较这些指纹。

MOSS 系统提供 Linux、Windows 等多种平台的服务，这个服务是在线的、免费的。用户可以使用相应的命令发送请求。在命令中包含有待检测的文件夹或文件的指定位置，编写语言等信息。连接成功后，MOSS 会将测试结果以网页的形式反馈给用户。在反馈信息中，可以查看被比较的程序相似度的值，并且对相似度大于某个阈值的程序对的相似的代码段进行同种颜色的标记。

3.1.3 SIM 系统

SIM(software similarity Testor)[19]系统是由 Dick Grune 开发的，它支持的程序设计语言有 C、Java、Pascal、Modula-2、Lisp、Miranda 等，同时也可以对文本文件进行检测。该系统采用的算法是一种应用于 DNA 序列相似性检测的串匹配技术[20]：串排列（string alignment）。

SIM 系统首先使用标准语法分析器将源程序转换成对应的语法分析树，然后通过语法分析树得到一个字符串。接下来，SIM 通过在字符串的字符中间插入空格的方法将这些字符串进行排列，这样就得到一系列的排列，然后将两个程序得到的排列进行比较，得到最大的匹配，这个比较使用的方法为动态程序设计方法（dynamic programming）[21]。

SIM 系统运行的时间复杂度为 $O(s^2)$，s 是 SIM 系统开始比较前生成的语法分析树的最大长度。实验结果表明，SIM 可以用来检测计算机科学专业低年级学生程序作业中的抄袭，包括变量改名，调整语句或函数顺序，增加或删除空格和注释等。SIM 在检测较小的程序时，运行速度比较快，在检测 56 个平均长度为 3415 字节的程序中每两对之间的相似性时，花费的时间大约是 3.5 分钟。

3.1.4 YAP 系统

Michael Wise 于 1992 年开发了 YAP1[22]，不久又开发了更优越一些的 YAP2，1996 年开发出了 YAP 的最终版本 YAP3。YAP1 和 YAP2 用于对程序代码的抄袭检测，YAP3 不仅可以对程序代码进行抄袭检测，还可用于对自然语言文本进行相似性的计算。YAP 系列比较两个程序后，会给出两个程序匹配结果即相似度的值，这个值是一个在 0 到 100 之间的数，表示两个程序从无关到完全相同。YAP 系列的基本操作过程是如下两个步骤：

（1）标识化处理

以上三个版本的处理过程相同。详细的步骤如下：

①删除程序中的注释；

②将所有的大写字母转换成小写字母；

③删除程序中所有不合法的标识符；

④生成基本标记串表；

⑤将同义词映射成相同的形式，如在C语言中，将strncmp映射为strcmp。

（2）对标识化后的字符串进行匹配，计算相似度的值

在这一步中，对标识化后的字符串进行匹配，得到一个最大匹配子串，然后再计算相似度的值。以上三个版本所采用的算法不同，分别是：

YAP1：混合使用了UNIX通用程序和命令程序脚本，如DIFF。

YAP2：使用的是Heckel算法。

YAP3：使用的是RKR－GST（Running Karp Rabin Greedy String Tiling）算法。

RKR－GST算法的基础算法是GST算法，该算法是由澳大利亚悉尼大学的Michael J.Wise设计的，被用在计算两个字符串之间的相似度上。该方法被YAP3及Jplag等系统优化，同时该算法还用于比较DNA序列。根据该算法的应用，可以返回两个标记串的相似度以及那些相似的代码段。

YAP的作者Wise用实验验证系统后声称，YAP可以处理程序抄袭中的如下几种变换：

①变换程序中的注释或程序的格式；

②变换程序中使用的标识符；

③改变操作数的顺序；

④改变变量的数据类型；

⑤用等价的表达式替换程序中原来的表达式。YAP完全能检测到，也可能出现很小的偏差；

⑥在程序中增加无用的语句或变量。增加无用变量，YAP完全能检测到；增加无用的语句，YAP检测结果的准确性可能会有很小的偏差；

⑦改变程序中独立语句的顺序，YAP检测此种变换会出现一些问题。

⑧改变重复语句的结构，改变选择语句的结构，用过程体代替过程调用语句，YAP能检测到大部分这三种变换；

⑨引入非结构化的语句，YAP检测结果的准确性可能会有很小的偏差；

⑩组合原来的和复制后的程序段。

实验结果证明，YAP用于检测学生程序作业中的抄袭是非常有效的。

3.1.5 JPLAG 系统

Jplag[23]是由德国卡尔斯鲁厄（karlsruhe）大学的 L.Prechelt,G.Malpohl 和 M.Phippsen 用 Java 语言编写的，在互联网上提供在线的程序代码抄袭检测服务。使用的比较算法与 YAP3 相同，也是 Greedy String Tiling，但却优化了 YAP3 的时间复杂度。该系统也是用来度量一个程序集合中代码文件的相似性的，当前的版本支持 Java，C，C++，Scheme 编写的源程序和自然语言文本。

Jplag 是按如下两个步骤计算两个程序之间的相似度的：

①由源程序生成标记字符串（token string）；

②比较每一对由程序所生成的标记字符串，以此来计算每一对程序之间的相似度。

在 Jplag 系统里，两个源程序文件 A 和 B 之间的相似性按如下公式计算：

$$Sim(A,B)=\frac{2*Coverage(tiles)}{|A|+|B|},Coverage(tiles)=\sum_{match(a,b,length)\in tiles} length$$

|A|,|B|：代表文件 A 和 B 中符号串的长度。

a,b：分别代表最大匹配串在 A 和 B 中的起始位置。

Length：最大匹配的长度。

实验测试证实，JPlag 在很多方面的功能与 MOSS 和 YAP3 是一样强大的，当提交的程序很多时，JPlag 比 MOSS 更出众。

3.1.6 Siff 系统

Siff[24]最初用于在大规模文件系统中寻找内容相似的文件，后来结合其它的方法来度量 Java 字节码文件的相似性。这个工具提出了“近似指纹(approximate fingerprints)”方法,要求从文档中选取一些字符串，这些字符串被称为“指纹”(fingerprint)，然后把指纹映射到 Hash 表中，一个指纹对应一个数字，最后统计 Hash 表中相同的指纹数目或者比率，作为文本相似度依据。这个思路被很多后来的文本复制检测系统所采用。

计算文本相似度的决策函数有很多种[25]，最简单的两种如下：令 F(A)表示文档 A 的指纹集，F(B)表示文档 B 的指纹集，S(A,B)表示文档 A 和 B 的相似度，则第 1 种决策函数为：

$$S_2(A,B)=|F(A)\cap F(B)|$$

第 2 种决策函数为：

$$S_1(A,B)=\frac{|F(A)\cap F(B)|}{|F(A)\cup F(B)|}$$

显然，无论使用哪种决策函数都有 S(A, B)=S(B, A)。

上述的 Siff 系统以 Winnowing 算法为原型。本文的后续研究中将参考 Siff 系统的算法原型 Winnowing 算法，提出新的指纹算法来完成文档之间的匹配。之所以选择指纹算法进行研究是因为指纹算法在匹配数据时所需要的空间相对于其它算法来说少很多，且匹配准确率方面也不亚于其它算法。

第四章 数据泄露防御算法的研究

数据泄露防御系统主要是用来保护企业有价值的数据。基于内容监测的数据泄露防御技术是目前比较先进的技术，用于防御数据泄露的内容匹配算法是此类系统的核心技术之一。在基于内容监测的数据泄露防御系统中，系统首先会对企业指定的有用数据进行处理，把处理后的结果存入系统中，之后系统会对进出系统的数据进行检测，会抽取对应的数据信息和系统存入的结果进行匹配，如果达到足够的相似度，则会做出相应的反应以阻止有用数据泄露。因此，对文档匹配算法的要求应运而生，系统需要比较高效的文档匹配效率以此提高本身性能。

本章将对常用的文档匹配算法进行分析，在此基础上借鉴已有算法的思想，设计新的数据泄露防御算法。

4.1 常用的文档匹配算法

文档内容的匹配效率是用户最为关心的指标之一，也是体现系统性能的重要标志，而字符串匹配算法则是优化内容匹配的基础。

字符串的匹配指的是从文本中找出给定字符串（称为模式）的一个或所有出现的位置。根据先给出模式还是先给出文本，字符串匹配分为四类方法：第一类方法是基本匹配，对模式和文本都未进行任何处理；第二类方法基于自动机或者字符串的组合特点，其实现上，通常对模式进行预处理；第三类方法对文本建立索引，这也是现在搜索引擎采用的方法。第四类方法是对模式和文本都进行处理，以此到达高效匹配的效果。以往的数据泄露防御研究算法的重心由第一、二、三类算法的研究转向第四类。第四类算法有效地弥补了之前三类算法所存在的不足之处，同时在效率方面得到极大的提高，并且相对于前三类算法，第四类算法所需要的空间也小很多。以下对第一、二、四类算法做介绍和分析。

4.1.1 第一类算法

此类算法对模式字符串和文本都不进行处理，只是以基本算法对模式字符串和文本进行比较，之后做出判定。朴素字符串匹配算法[26,27]是第一类的典型算法，朴素字符串匹配算法的思想非常直白，既然是查找模式字符串在文本字符串中的出现，那么就从文本字符串的每一位开始依次判断是否与模式字符串相同，如果相同那么就找到了模式字符串的一次出现，可以看出此算法的时间复杂度为 O(mn)，m,n 分别为文本字符串与模式字符串的

长度，但是通过更加深入的分析可以得出此算法的平均时间复杂度为 O（n+m），所以此算法在模式字符串与文本字符串都是随机选择的情况下会工作得相当不错；此外，该算法并没有其它算法所必须的预处理时间。但是，该算法本身存在工作效率的问题，所以逐渐被第二类算法和第三类算法所取代。

4.1.2 第二类算法

第二类算法侧重于对模式字符串进行预处理，之后利用处理后的字符串和文本进行比较。属于第二类算法范围的有字符串匹配自动机、KMP 算法[28]、后缀树算法[29]、Boyer-Moore 算法[30]等算法。其中：

1. 字符串匹配自动机

字符串相关的自动机是一个五元组（Q,q,A,E,%）。其中 Q 是状态的有限集合，q<Q 是开始状态的集合，A<Q 是接受状态的集合，E 是输入字符集合，%是一个 QXE->Q 的映射，称为转移函数。通过模式字符串，我们可以构造出识别此字符串的自动机，此自动机对应于字符表中的任何一个输入在某个状态下都由一个转移到另一个状态（包括它自身），当进行到接受状态时就找到了模式字符串在目标字符串中的一次出现，由于自动机需要对于任何一个字符集中出现的字符都给出相应的转移，也就意味着构造自动机的时间复杂度与|E|有关，而事实上，构造自动机的时间复杂度为 O（m^3|E|），但当此自动机构造完成后，搜索目标字符串的时间为 O(n)。

2. KMP 算法

KMP 算法采用类似上面所提到的字符串自动机的原理，但完全避免了计算转移函数，它仅仅用了一个只需要 O(m)就能计算出来的辅助函数 pai[1..m]就达到了搜索时间复杂度为 O(n)。

函数 pai 称为前缀函数，是根据模式字符串构造的，对于一个给定的模式字符串 p[1..m]，它的前缀函数定义如下：pai[q]=max{k:k<q and p(k)>p(q)}即：pai[q]是 p 的最长的与后缀 p(q)相同的前缀。

通过均摊分析可以证明计算前缀函数的时间复杂度为 O(m)，而同样把此方法用于分析 KMP 算法的时间复杂度，我们可以得出 KMP 匹配算法的时间复杂度为 O(m+n)。

3. 后缀树算法

后缀树算法分为三类：

1）后缀字典树算法

对于目标字符串的所有后缀，可以把它们构造成字典树，此字典树的每一个节点（除叶子节点外）都可能有 | E | 个分支，这样在目标字符串中搜索模式字符串的出现变得非常简单，只需要从字典树的根节点出发，按照对应的分支向下查找，最多比较 m 次，就可以确定此模式字符串是否在目标字符串中出现。所以查找字符串的时间复杂度为 O(m)，与目标字符串 n 无关，这种好处是非常明显的，它意味着对于一个目标字符串，一旦建立它的后缀字典树后，在此目标字符串查找任意字符串都只需要 O(m)的比较，但是构造后缀字典树无疑是很花费时间与空间的，通过分析可以知道，构造此字典树的时间花费为 $O(n^2)$，对于很大的目标字符串来说，这是无法忍受的。

2）非在线后缀树算法

为了减少构造后缀字典树所花费的时间与空间，Edward McGreight 于 1976 年发表了后缀树的论文，此后缀树与后缀字典树在代表的含义上完全相同，但由于它采用了路径压缩的方法，即每条边不只代表一个字符，有可能是一个字符串，大大节省了构造此后缀树所花的时间与空间，可以证明此后缀树最多有 2n 个节点，构造的时间为 O(n)，但是此算法有个非常明显的缺点，就是它是倒序处理字符串的，也就是它是非在线的。

3）在线后缀树算法

1995 年 Ukkone 提出了在线的后缀树算法[31]，支持从左到右地输入字符串，并在输入的同时构造出后缀树，至此给后缀树画上了完美的句号，通过分析可以知道基于后缀树的字符串匹配算法的时间复杂度为 O(m+n)。

4. Boyer-Moore 算法

Boyer-Moore 算法从模式字符串字串 P 后面开始比对。如果字串 P 的最后一个字元 P[M-1] 不等于文本 T[M-1]，而且 T[M-1]不等于 P[0]…P[M-2]之间任一个字元的话，下一轮的比对就可以直接从 T[M]开始。在理想的情况下，只要花费 N/M 回合就可以决定搜寻的结果。字符匹配过程如下图 4-1 所示：

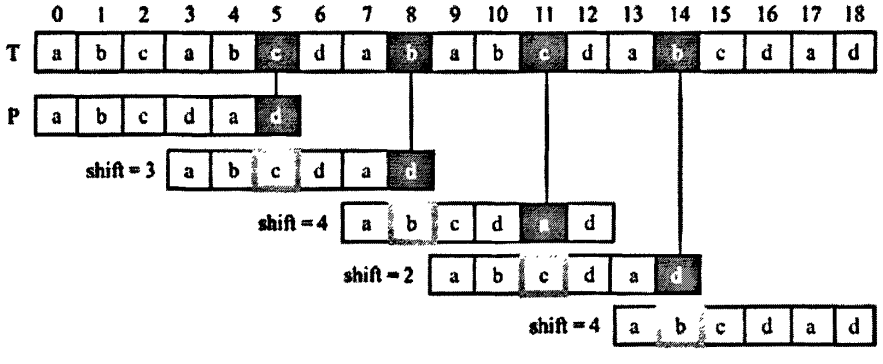

图 4-1 字符匹配过程

Boyer-Moore 算法启动了 5 次搜寻，但却只花费 11 次字元的比对。Boyer-Moore 算法需要一个 skip 表格，当 T[i]比对失败时，用来计算位移大小。如果遭遇到无法继续前进的窘境，就要使用暴力法强迫前进一步。使用 skip 表格计算位移量的方式是发生在字元不相符的时候，因此称之为“不相符字元位移”。另外一种位移计算方式是相符字尾位移，它根据文件字尾与字串 P 中间相符的部分作为位移的考量。这个方法跟 KMP 法类似，差别在于 Boyer-Moore 算法是从字串后面来考虑。不相符字元位移和相符字尾位移独立运作的话，都可以得到不错的效率。但是各自都有其最坏的情况。Boyer 和 Moore 建议同时使用这两个表格，并取其较大的位移量用来计算下一回合的比对起点。使用 Boyer-Moore 算法，文件 T 的索引 i 也是维持递增的，因此执行搜寻的复杂度为 O(N)。加上构建表格的运算，总复杂度为 O(M+N)。

4.2 第四类算法研究

属于第四类算法研究范围的有：Rabin-Karp 算法[32-34]、Winnowing 算法[35-38]（指纹算法）等。

4.2.1 Rabin-Karp 串匹配随机算法

在介绍 Rabin-Karp 串匹配随机算法之前，首先介绍几个概念：

(1)字符串的模式匹配：设有两个字符串，即文本串与模式串，要求在文本串中查找是否有与模式串相等的子串，并给出子串在文本串中出现的位置，这一过程称为字符串的模式匹配。

(2)文本串：是指在字符串匹配过程中待搜索的串，它通常是 n 个字符的一个序列。

(3)模式串：是指在字符串匹配过程中需要查找的子串，它通常是 m 个字符的一个序列(m≤n)。

Rabin-Karp 算法是 1987 年图灵奖获得者 KARP 教授和著名学者 RABIN 教授合作在 IBM 研究与发展杂志上发表的一种直观、快速的串匹配随机算法,简称 KR 算法。算法十分简单，但是效率却很高。Rabin-Karp 算法是一个在实际中比较好用的字符串匹配算法。数学家 Rabin 和 Karp 从不同的角度研究字串比对的方法,他们透过杂凑函数将字串 P 和文件 T 的部分字串转换成数字，然后直接比对 hash 值来判断字串是否相等。

该算法的基本思想是把长度为 m 的模式串看作是一个键(key)，而把文本串中每 m 个字符也看作是一个键。定义一个散列函数把这些键都映射为它们对应的散列值，那么只有那

些与模式串具有相同散列值的文本字段才有可能与模式串匹配。

Rabin-Karp 算法对散列函数的处理：假设文本串与模式串中出现的字符集合为 Σ，设集合的大小为 S，那么可以把字符串看作是一个 S 进制的数，对于子串：

$P(k)=p_k\ p_{k+1}p_{k+2}\cdots p_{k+m-1}$

可以把各个字符取其内码(如 ASCII 码)，则 P(k)可以数值化为 P(k)`:

$P(k)`=p_k*s^{m-1}+p_{k+1}*s^{m-2}+p_{k+2}*s^{m-3}+\ldots+p_{k+m-1}$

取一个较大的素数 q，P(k)的散列函数定义为：

$Hash(P(k))=mod(P(k)`,q)=mod(p_k*s^{m-1}+p_{k+1}*s^{m-2}+p_{k+2}*s^{m-3}+\ldots+p_{k+m-1},q)$

同理，若将子串 P(k)往后移一个位置，得到新的子串 $P(k+1)=p_{k+1}\ p_{k+2}\cdots p_{k+m-1}$，则有：

$Hash(P(k+1))=mod(p_{k+1}*s^{m-1}+p_{k+2}*s^{m-2}+p_{k+2}*s^{m-3}+\cdots+p_{k+m},\ q)$

显然，Hash(P(k))与 Hash(P(k+1))满足如下关系：

$Hash(P(k+1))=Hash(P(k)-p_k*s^{m-1})*s+p_{k+m})$

假设要匹配的文本串与模式串都是数字串。令 T="75610163225371468459721 23"，P="25371"。

此时的字符集为 Σ={0,1,2,3,4,5,6,7,8,9}，则 S=10；定义散列函数为：

Hash(key)=mod(key,17)

每个字符的内码定义为数字本身，那么首先可以预先计算 S 的幂的散列值

Hash(10)=4，Hash(10)=14，Hash(10)=15，Hash(10)=10

接着可以计算模式串的散列值如下：

Hash(25371)=mod(25371,17)=7

从文本串左端开始，取 m 个字符作为一个键，则第一个子串的散列值

Hash(75610)=mod(75610,17)=11

向右移动一个字符，则下一个子串为“56101”，其散列值计算方法如下：

$Hash(56101)=mod((75610-7\times10^4)\times10+1,17)$

$=mod((mod(75610,17)-mod(7\times mod(10^4,17),17))\times10+1,17)$

$=mod((11-mod(7\times4,17))\times10+1,17)$

$=mod(1,17)$

$=1$

这样不断地向后移动一个字符，可以求得左右的长度为 m 的连续字符串的散列值，其中的过程省略，结果如表 4-1 所示。其中第一行的字符表示子串以该字符为起始字符。

表 4-1 子串散列值

子串	75610	56101	61016	10163	01632	16322	63225
散列值	11	1	3	14	0	2	2
子串	32253	22537	25371	53714	37146	71468	14684
散列值	4	12	7	11	1	0	13
子串	46845	68459	84597	45972	59721	97212	72123
散列值	10	0	5	4	19	6	9

从表格中可以看到，子串散列值与模式串散列值相同的只有一个子串，此时比较这两个子串，显然刚好是相等的。对于 Rabin-Karp 算法，预处理时间为 O(m),如果不考虑执行匹配检查的时间复杂度，它的在最坏情况下的时间复杂度为 O((2n-m+1)m)，而平均复杂度接近 O(m+n)，此算法的主要思想就是通过对字符串进行 hash 运算，使得算法可以容易地排除大量的不相同的字符串，假设模式字符串的长度为 m，利用 Horner 法则 p = p[m] + 10(p[m -1] + 10(p[m-2]+...+10(p[2]+10p[1])...))，求出模式字符串的 hash 值 p，而对于文本字符串来说，对应于每个长度为 m 的子串的 hash 值为 t(s+1)=10(t(s)-10^(m-1) T[s+1])+T[s+m+1]，然后比较此 hash 值与模式字符串的 hash 值是否相等。若不相同，则字符串一定不同；若相同，则需要进一步的按位比较，整个 Rabin-Karp 法的效能显然是 O(M+N)。然而，不可避免的是，仍有极小的机率是连续 N 个相同的 hash 值遇到杂凑函数碰撞问题。此时最差的效能为 O(NM)。Rabin-Karp 法最令人注目的特色在于，它能够处理多维阵列资料的搜寻。而且它也适用于需同时搜寻多组长度相同的字串的场合。

4.2.2 Winnowing 算法

Winnowing 算法是把模式字符串进行处理，改变成指纹数据，之后对文本进行同样的处理，对指纹数据进行匹配。Winnowing 算法是比较经典的划动窗口匹配算法[39]的修改，在原有划动窗口的基础上增加对原文本的处理，使其算法更加高效。Winnowing 算法首先要求满足以下 2 个条件：

（1）要求字符串足够的长，其长度 t 使得算法能检测出结果。

（2）文本的长度要足够的长，同时窗口长度 k 要设置足够的长度。

由于算法用于文件内容的匹配，所以以上 2 个条件属于必定会满足的情形。之后对文本或模式字符串进行初始处理，遵循选择窗口最小的 hash 值，同时只选唯一的 hash 值，保存所有选出来的 hash 值作为文本或模式字符串的指纹。其步骤如下：

（1）文本中有如下内容：

A do run run run, a do run run

（2）把不相关的内容去除，忽略大小写：

adorunrunrunadorunrun

（3）设置窗口为 5，筛选文本，得到：

adoru dorun orunr runru unrun nrunr runru

unrun nruna runad unado nador adoru dorun

orunr runru unrun

（4）假设这些窗口中的队列 hash 值为：

77 74 42 17 98 50 17 98 8 88 67 39 77 74 42 17 98

（5）以 4 为标准组合 hash 值，得到：

(77,74,42,17) (74,42,17,98)

(42,17,98,50) (17,98,50,17)

(98,50,17,98) (50,17,98,8)

(17,98,8,88) (98,8,88,67)

(8,88,67,39) (88,67,39,77)

(67,39,77,74) (39,77,74,42)

(77,74,42,17) (74,42,17,98)

（6）通过 Winnowing 算法筛选出的 hash 值为：

17 17 8 39 17

（7）记录 hash 值在原文中的位置如下：

[17,3] [17,6] [8,8] [39,11] [17,15]

通过以上步骤，Winnowing 算法获取当前文本的指纹，之后就利用指纹进行文件与文件之间的比较。图 4-2 是使用 Winnowing 算法处理文本后需要的 hash 值个数：

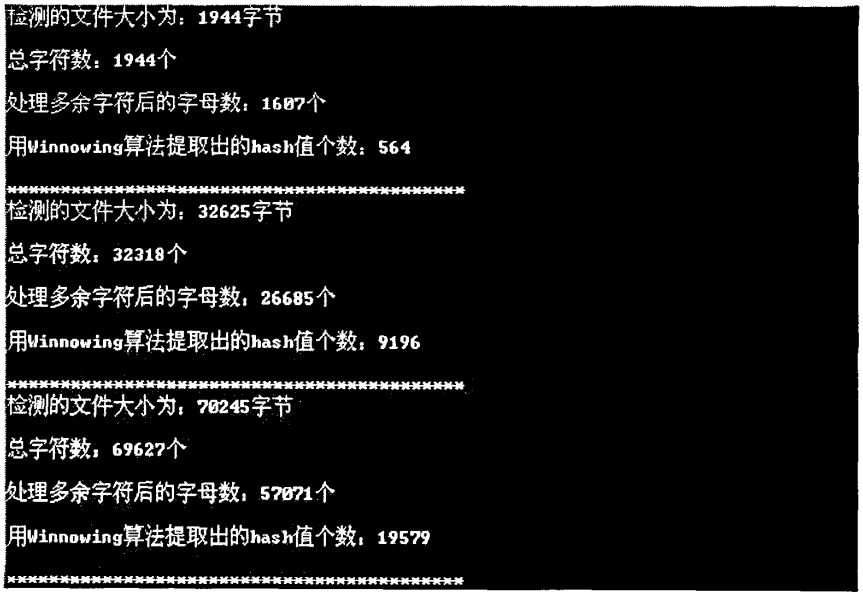

图 4-2 Winnowing 算法生成的 hash 值个数

通过上图可以清楚发现，文档匹配的情况下 Winnowing 这类指纹算法所需要的空间仅仅是用来存储hash值的空间,对于字串匹配则需要原文档作为副本存储起来,用Winnowing可以节省 2/3 的空间。Winnowing 算法只需要保存文件的 hash 值，之后利用 hash 值进行比较，能更有效地节省使用空间。同时利用该算法，可以以很小的标签来代替很大的文件，以下是该算法的另一个优势体现：

X=所对应的文件

n=文件转换成指纹的长度

f:X->{0,1}n 指纹算法的函数

对于文件 A，B，如果 f(A)≠f(B)则 A≠B，发生误判情况的概率为：

$Pr(f(A)=f(B)\,|\,A\neq B)\approx 1/2n$

根据以上数据可以看出，指纹算法本身可以更加高效地处理文件和提高效率与准确率。

4.3 一种新的基于指纹的数据泄露防御算法

鉴于指纹类算法具有节省存储空间同时对于文档之间有较高匹配率的优点，本文以上节介绍的 Winnowing 算法为基础，以生成文档特定的指纹为目标，通过对 Winnowing 算法加以改进，提出了一种新的基于指纹的数据泄露防御算法。

Winnowing 算法是基于 Rabin-Karp 和 hash 值算法衍生而出的,在求解 hash 值的同时，其 hash 值的个数会随着文件的增大而增多，如表 4-2 所示，由于文档内容的增多，

Winnowing 求解 hash 值的个数也随着增大，存储 hash 值所需要的空间也随之增加。

表 4-2 Winnowing 需要的 hash 值个数

文件大小	Hash 值求解个数
1944 字节	564
32625 字节	9196
70245 字节	19579

本文提出的新算法对文档进行处理后，生成固定字节的指纹值，同时对文档中的指纹值进行权重评估，为进一步提高算法自学习能力提供基础。

4.3.1 算法的伪代码

算法的伪代码如下：

输入：散列值序列，序列长度和窗口长度值；

输出：(指纹，位置)序列。

```
Step1：//从第一个窗口中提取特征
    hashVal[min]=hashVal(c1c2…ck);
    for(int i=1;i<=w;i++)
    {
    if(hashVal(cici+1…ci+k-1)<=hashVal[min])
    hashVal[min]=hashVal(cici+1…ci+k-1);
    }
    return(hashVal[min],global-pos[min]);
Step2：int r=w
Step3：//从其它窗口中提取特征
    while(r<length){
    r=r++;//指针右移
    hashVal[r]=next_hashVal();//添加一个新的散列值
    //前一个提取出的特征不在当前窗口中，从右向左查找最小值
    if(min+w==r){
    hashVal[min]=hashVal[r];
    for(int i=r-1;i<r-w+1;i--){
```

```
if(hashVal[i]<hashVal[min]){
min=i;
return(hashVal[min],global-pos[min]);
}
//比较前一个提取出的特征与窗口中最右边散列值
else{
if(hashVal[r]<=hashVal[min]){
min=r;
return(hashVal[min],global-pos[min]);
}}}//获得指纹集合
Step4://对指纹值进行权重评估
Short int num[7][2];//存放指纹和评估值
for(i=0;i<7;i++)
for(j=0;j<7;j++)
//把指纹值存入至 int 型 2 维数组 num[0][0]-num[7][0]中
SaveNum(fingerNum,num[i][j]);
//num[1][j]-num[i][j]存放对应指纹的初始评估值
ModifyNum(num[1][1],InitialNum);
Step5://随着系统的运行，动态的修正指纹评估值
for(i=0;i<7;i++)
for(j=i+1;j<7;j++)
//把指纹值存入 temp[0]-temp[7]中
{SaveNum(fingerNum,temp[i]);
//比较指纹值之后进行评估修正
If(temp[j]!=num[i][j])
{
num[i+1][j]=temp[j];
restructNum(num[i][j])
}}
//此后，可利用矩阵求解平均向量，获得对应的指纹权值。
```

4.3.2 提取指纹值的流程

上述算法中对文档提取指纹值的流程可用图 4-3 描述如下：

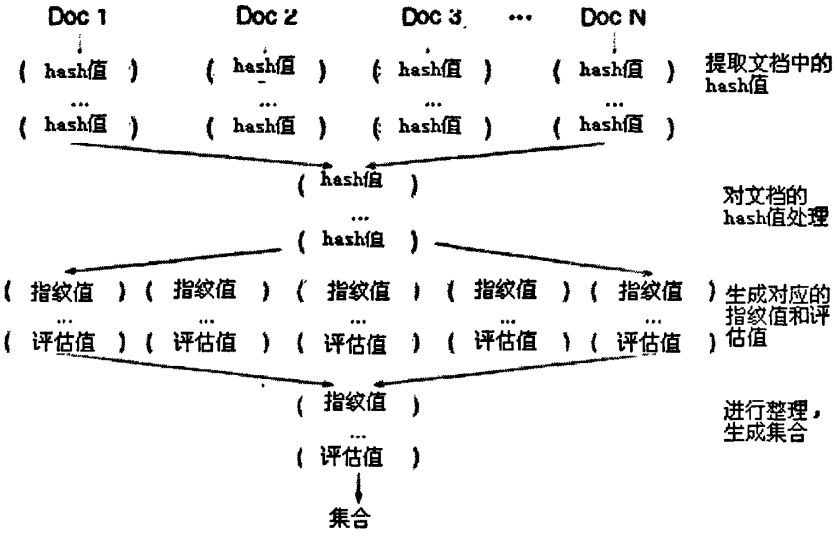

图 4-3 提取文档中独立的指纹值

4.3.3 文档检测过程

图 4-4 是利用本文提出的新算法对文档进行检测的过程。通过对比文档之间的指纹值，以此判断文档是否相似，同时在数据库中对文档的指纹值进行权值评估，随着系统的运行，对于文档匹配的效果将会得到提升，能够更有地的获得文档的相似值。

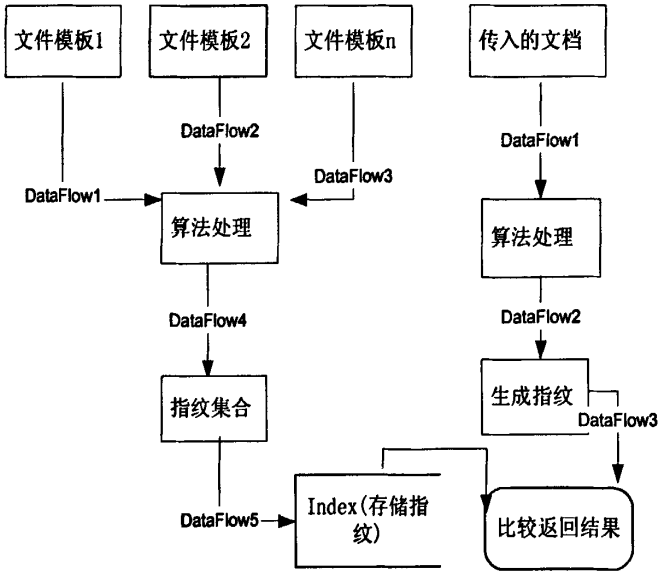

图 4-4 文档检测过程

以下是用本文提出的新算法对文档处理的过程：

(1)整合文档，去除文档中无意义单词，例如“the”，“do”，“am”等，之后把文档字符全部最小化。

(2)去除文档中的空格等无意义字符。生成新的列表 L1。

(3)从 L1 中经过筛选（Robin 算法求值），把结果存入列表 L2 中。

(4)对于 L2 的每个值记录它在 L1 中的位置，如 L1={t1, t2, t3…t_e}, L2={T1, T2….T_f}，对应的 T_i~<P(i, 1), P(i, 2), ….P(i, S_i)>记录所处的 L1 的位置。

对于 i=1, …f, S_1 +S_2+…+S_f=e。

(5)对列表 L2 的值进行计算：

Score (T_j) = [P(j, S_j) - P(j, 1)] * S_j * Weight(T_j) / Sqrt (D_j),

D_j=[P(j, 2)-P(j, 1)]^2+ [P(j, 3)-P(j, 2)]^2…. + [P(j, S_j)-P(j, S_{j-1})]^2

score 函数功能是用来度量在文档中 j 端数据的频率使用和重要性，这主要取决与 S_j * Weight(T_j)。score 函数同时确保所选段落可以在文档中占有足够的分数（确保段落在文档中的重要性），这取决与[P(j, S_j) - P(j, 1)]/ Sqrt (D_j)。

(6)对 L2 中的分数进行排序。

(7)取出前 m 个分数，存入列表 L3 中。

(8)对于 L3 中的 T_j 从 L1 中找出对应的：P(i, 1), P(i, 2), …., P(i, S_i)。之后如下：

For each k ε { P(i, 1), P(i, 2), …., P(i, S_i) }，在 L1 中选取相邻的第二个字段，把它们组合成一个字符串，t_{k-d} + … + t_{k-1} + t_k + t_{k+1}+ … + t_{k+d} 把这段字符转换成特征值 $F_{j,k}$。

(9)从 L3 的 T_j 中，对列表 L3 的{ $F_{j,1}$, $F_{j,2}$, …. $F_{j,sj}$ }排序，选取最大的 n 个特征值。

(10)对于所有 L3 的成员，已有 m*n 个特征值，其值就是从文档中计算出的结果。

(11)随着匹配次数的增加，对指纹中的权值进行评估，提高匹配的准确率。

其中第(1)-(2)步骤是对文档的初始操作，第(3)-(7)步骤是从文档中选取最有信息的字段。第(8)-(10)步骤是把上面步骤得出的字段进行特征值的求解，其中第(8)步保证求解的特征值具有文档的唯一性，第(9)步骤是在有限的空间里获得最有信息价值的特征值，从而达到节约空间的目的。

对文档进行处理后，完成特征值的求解，对以后输入的相似文档进行判断，根据相似度的大小，判断文件是否该做处理。同时对于系统要监控大量文件的情况，使用本算法后，可以大幅度减少样本存储空间，还可以有不低于全文匹配方法的匹配度。

4.3.4 与全文检测算法的比较

（1）检测准确率比较

对于大小为 t 的文档，首先划分为 m 个组的 n 个元素组合，对每个分组进行指纹算法，得出其文件的特征,如果这两文件相似，必有大于 r 个的共同特征。现在假设两个文档基本相似度为 β 的话，文献[40]中提出的全文检测和指纹检测的判定概率分别为：

利用全文检测技术获得的概率为：

$$P=\sum_{i=t}^{m*n}\left(\frac{m*n}{i}\right)\beta^{i}(1-\beta)^{m*n-i} \tag{4-1}$$

利用指纹检测技术获得的概率为：

$$P=\sum_{i=r}^{m}\left(\frac{m}{i}\right)\beta^{n-i}\left(1-\beta^{n}\right)^{m-i} \tag{4-2}$$

指纹检测技术的概率值十分接近全文检测技术的概率，但全文检测技术要求一定空间去存储所检测的文件样本，所消耗的空间和指纹算法就无法比较了。假设当文档的大小 t 为 100,划分成 m 为 8 个分组,每个分组 n 为 10,制作成 8bytes 的指纹,只需要存在 64bytes/文档。对于以字符串匹配的全文检测需要原文档做存储副本完成匹配过程，因此所需要的存储空间大小和原文档大小一样。可见，将需要防范丢失的数据文档转换成指纹后可以大幅度减少空间，实用性比全文检测技术高；同时本算法所需要的空间只需要固定的 64 字节，相对于 Winnowing 算法，本文的新算法在空间复杂度上更具有优势。

（2）文档检测效果的比较

本文算法是面向英文文档。在把文档分解成 k-grams 文本块的过程中，选择合适的 k 值是非常重要的。因此，首先使用不同的 k 值分解同一个纯文本文档。在处理文档时，对于不同 k 值对文档的影响进行了研究。发现不同 k 值对文档判断的影响如图 4-5 所示，其中纵坐标表示每 100 个匹配中发生错误的数目，横坐标表示 k 的取值。

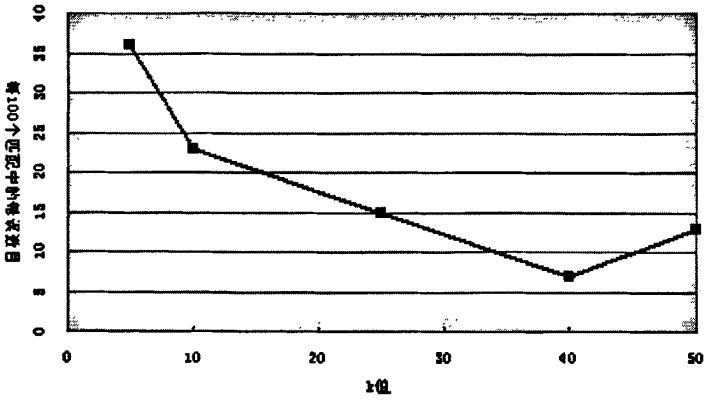

图 4-5 k-grams 的取值与匹配出错数目的关系

从图 4-5 中可以看出，恰当地选取 k 值，可以使发生错误的可能性下降。在下面的实验中，取 k=40。下例给出了一个文本最开始的两个 40-grams。

假设文本为：Often, publishers are reluctant to offer valuable digital documents on the Internet for fear that they will be re-transmitted or copied widely….

第一个 40-grams 为：oftenpublishersarereluctanttooffervaluab。

第二个 40-grams 为：ftenpublishersarereluctanttooffervaluabl。

英文文档中出现的字符集合就是从字母 a 到 z 的 26 个字母，所以可以把文档中的每一个 k-grams 字符串看作是一个 26 进制的数。在散列函数中令 asc(a)=0；asc(b)=1；…；asc(y)=24；asc(z)=25，取基数 b=2，把生成的散列值定义为 long int 型。令 $q=25\times(2^{50}-1)=28147497671065575$，用来避免不同文本块的散列值发生冲突。图 4-6 是用本文的新算法与全文检测算法检测以上文本的效果比较。

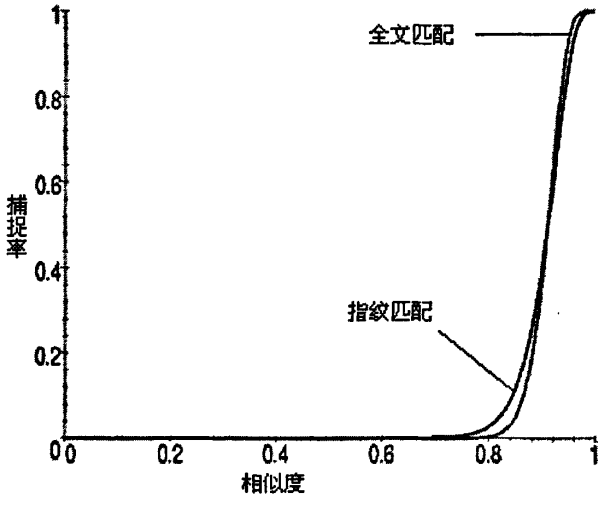

图 4-6 检测相似文件的对比

从上图的比较可以看出，全文匹配和指纹匹配工作结果很接近，但指纹算法所节省的空间却远比全文匹配多，指纹算法需要数据库存储的是文件的指纹，而全文匹配则是存储整个文件，因此在存储方面全文匹配存在着明显的不足。

4.4 本章小结

本章着眼于基于内容监测的数据泄露防御技术，分析了几种常见的文档匹配算法的优缺点，并重点研究了指纹算法。

由于一个好的匹配算法应能合理地利用时间和空间来优化其工作效率，本文基于Winnowing 算法这种指纹算法，通过改进提出了一种新的基于指纹匹配算法的数据泄露防御算法。该算法对文档进行处理，所防御的数据进行矩阵求解，生成被防御数据的“指纹”，指纹值字节固定。如果有用户把相关的信息发送出去，系统会提取部分数据流，通过指纹算法生成这段数据流的“指纹”，然后进行比较，如果相同则对其进行拦截，否则就允许数据发送出去。

以指纹模式来匹配文件，能以较少的资源消耗完成对资源的监控过程，可以高效地完成数据的对比过程，同时有较好的匹配性。

下一章将对这个新算法进行仿真。

第五章 仿真及结果分析

本章采用面向对象的方法，实现了基于指纹匹配方法的文档复制检测系统，并进行了两组实验，以对所提出的基于指纹的数据泄露防御算法的有效性进行验证。第一组实验用来验证系统具有的特性。第二组实验用来与 Winnowing 算法进行比对，并高效地检测出原有的文档和复制文档的相似性。利用系统定义的文档相似性度量方法对不同文档进行复制检测。实验结果反映了新算法的检测高效性。

5.1 实验环境和参数设置

前台利用 Visual C++，后台用 Microsoft SQL Server 数据库管理系统开发了基于指纹匹配方法的文档复制检测原型系统。

系统运行环境如下：CPU 为 Inter P4 2.4 GHz，内存为 256 M，操作系统为 Microsoft Windows XP。

5.2 验证算法的特性

数据泄露防御技术是为防止企业有用数据信息被传送出去，导致企业受到损失。本实验以 IT 企业的程序源代码为例进行说明。数据泄露防御技术对企业的源代码进行保护，检测出入的内容数据是否和保护的源代码一致；同时检测内容数据和保护的源代码相似度，如果足够高则进行阻止。

5.2.1 实验基础数据设置

对于源程序的修改，在 2001 年 Edward L. Jones 将更改手段（在不影响结果的情况下）重新总结，分为如下十类[41]：

①逐字拷贝。

②更改注释语句。

③更改空白区域。

④重新命名标识符。

⑤改变代码块的顺序。

⑥改变代码块中语句的顺序。

⑦改变表达式中操作符和操作数的顺序。

⑧更改数据类型控制逻辑。

⑨增加冗余的语句和变量。

⑩用等价的控制结构替换原有控制结构。

可以肯定地说，这些更改都是表面的，是少量的，而程序中内含的属性和结构是没有改变的。我们所说的程序的属性，就是根据程序的内在性质所定义的不随表达形式而变化的特性。一般来说，这些属性不易被改变，即使改变也是少数的。程序的结构代表了问题解决的逻辑和步骤，即使程序的属性可以做微弱的改动，但解决问题的逻辑即程序的结构是不会发生改变的。所以这样的程序具有内在的相似性。

依据 Faidhi 和 Robinsons 的六层修改谱系图，收集了如下 5 个用 C 语言实现的折半查找程序作为测试用数据集。

原创程序 1（文件名 binarysearch.c）

```
#include"stdio.h"
/*para:a 数组
para:value 要查找的值
para:length 数组长度*/
int BinarySearch(const int A[],int key,int N)
{int Low,Mid,High;
Low=0;High=N-1;
while(Low<=High)
{Mid=(Low+High)/2;
if(A[Mid]<key) Low=Mid+1;
else if(A[Mid]>key) High=Mid-1;
else return Mid;}
return -1;}
main()  //测试:
{int a[]={-2,-4,0,5,11,13};
printf("BinarySearch:%d\n",BinarySearch(a,13,6));}
```

第一层修改:改变程序语句(文件名 bsearch1.c)

```
#include"stdio.h"
void BinarySearch(int inarray[],int value,int length)
```

```
{int down=0;
int up=length-1;
int mid=(low+high)/2;
for(;down<=up;)
{ if(value<inarray[mid])
up=mid-1;
else if(value>inarray[mid])
down=mid+1;
else {
printf("Find!\n");return;}
mid=(down+up)/2;}
printf("Can't Find!\n");}
main()
{int a[]={1,2,3,4,5,6,7,8,9};
int key;
printf("\nPlease input the value you want to search:");
scanf("%d",&key);
for(;;)
{if(value==0)break;
BinarySearch(a,key,9);
printf("\nPlease input the value you want to search:");
scanf("%d",&key);}}
```

第二层修改：改变标识符（文件名 bsearch2.c）

```
#include"stdio.h"
void BinarySearch(int inarray[],int value,int length)
{int down=0;
int up=length-1;
int mid=(down+up)/2;
while(down<=up)
{if(value<inarray[mid])
up=mid-1;
```

```
else if(value>inarray[mid])
down=mid+1;
else{printf("Find!\n");
return -1;}mid=(down+up)/2;
}printf("Can't Find!\n");}
main()
{int a[]={1,2,3,4,5,6,7,8,9};
int key;
printf("\nPlease input the value you want to search:");
scanf("%d",&key);
while(value!=0)
{BinarySearch(a,key,9);
printf("\nPlease input the value you want to search:");
scanf("%d",&key);}}
```

第三层修改：改变单个语句或代码块位置（文件名 bsearch3.c）

```
#include"stdio.h"
void BinarySearch(int inarray[],int value,int length)
main()
{
int a[]={1,2,3,4,5,6,7,8,9};
int key;
printf("\nPlease input the value you want to search:");
scanf("%d",&key);
while(value!=0)
{BinarySearch(a,key,9);
printf("\nPlease input the value you want to search:");
scanf("%d",&key);}
}
void BinarySearch(int inarray[],int value,int length)
{int down=0;
int up=length-1;
```

```
int mid=(down+up)/2;
while(down<=up)
{if(value<inarray[mid])
up=mid-1;
else if(value>inarray[mid]) down=mid+1;
else{printf("Find!\n");
return;}mid=(down+up)/2;}
printf("Can't Find!\n");}
```

第四层修改:过程合并（文件名 bsearch4.c）

```
#include"stdio.h"
main()
{int a[]={1,2,3,4,5,6,7,8,9};
int key,down,up,mid;
printf("\nPlease input the value you want to search:");
scanf("%d",&key);
while(value!=0)
{down=0;
up=a.length-1;
mid=(down+up)/2;
while(down<=up)
{if(value<a[mid])
up=mid-1;
else if(value>inarray[mid])
down=mid+1;
else{printf("Find!\n");
break;
}mid=(down+up)/2;
}if(down>up)printf("Can't Find!\n");
printf("\nPlease input the value you want to search:");
scanf("%d",&key);}}
```

5.2.2 不同 k 值下对程序进行相似性度量的结果

在把标记串分解成长度为 k 的相互重叠子串的过程中，选择合适的 k 值是非常重要的。在实验中，k 值分别取 5，25，50 时的相似度度量结果分别如表 5-1，表 5-2，表 5-3 所示。

表 5-1 k=5 时数据代码相似度

	bsearch1.c	bsearch2.c	bsearch3.c	bsearch4.c
Binarysearch.c	0.2560	0.2714	0.2630	0.2137
Bsearch1.c	1	0.8127	0.7358	0.4454
Bsearch2.c		1	0.9115	0.6039
Bsearch3.c			1	0.5691
Bsearch4.c				1

表 5-2 k=25 时数据代码相似度

	bsearch1.c	bsearch2.c	bsearch3.c	bsearch4.c
Binarysearch.c	0.7467	0.7670	0.7679	0.7554
Bsearch1.c	1	0.9038	0.8723	0.7959
Bsearch2.c		1	0.9182	0.8758
Bsearch3.c			1	0.7873
Bsearch4.c				1

表 5-3 k=50 时数据代码相似度

	bsearch1.c	bsearch2.c	bsearch3.c	bsearch4.c
Binarysearch.c	0.5767	0.5470	0.5679	0.5554
Bsearch1.c	1	0.8638	0.7834	0.612
Bsearch2.c		1	0.9033	0.8544
Bsearch3.c			1	0.6717
Bsearch4.c				1

5.2.3 K 值对检测结果的影响分析

分析上述实验结果，不同的 k 值，将会对检测结果产生不同的影响：

k=5：在程序源代码中，经常有许多长度较小的标记串出现，如C++程序中的“);}”，在源代码和目标代码中检测出这些子串的匹配，不一定是完全复制，在许多情况下这样的相似性仅仅是巧合而已，K 值若取大于这些经常出现的短标记串，将会减少虚假数值的出现，从而使检测结果更加准确。

k=25：k 值选用 25 时，结果准确率开始上升。因为 K 值设置适当，可以有效地检测出目标程序的相似度，从而增加了系统匹配率。

k=50：当 k=50 时，k 值选取过大，产生大量的冗余值，影响系统对文本的匹配准确率。

可见，当 k 值太小或太大，将产生大量的虚假值或冗余值导致无法正确获取代码程序之间的匹配值，在代码检测过程中可根据需要选择合适的 k 值，一般选用 10 到 40 之间的值。

5.3 算法响应时间

为了对 Winnowing 算法与本文提出的新的指纹匹配算法的检测效率进行比较，本实验对上述数据集中的 8 个用 C 语言编写的折半查找程序进行两两相似度度量，记录所需运行时间。

Winnowing 算法中的 K 值和本文的指纹匹配算法中的 K(代码分割长度)分别取 5、10、25、40、50，每个长度、每个算法都运行多次，对运行时间取较稳定的值，结果如图 5-1 所示。

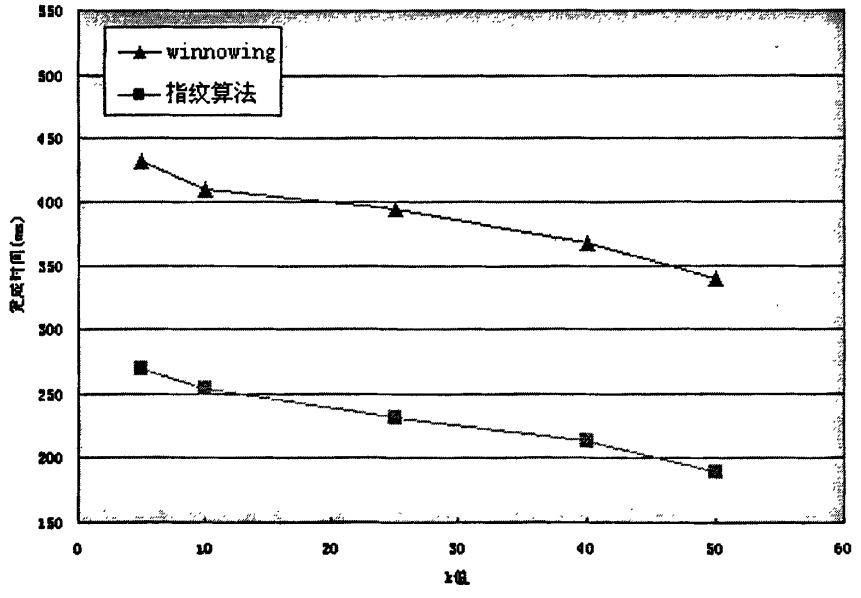

图 5-1 算法响应时间表

5.4 本章小结

本章用 Visual C++、Microsoft SQL Server 开发了基于指纹匹配方法的文档复制检测原型系统，基于该系统设计了两组实验。首先通过实验确定分割标记串的长度 k，选择 k 值的长度保证系统检测结果中产生错误的数目很少，然后用实验证明了指纹匹配算法比 Winnowing 匹配算法具有更高的效率。

第六章 总结与展望

由于数据泄露防御是一种新出现的安全防御技术，它具有许多区别于传统安全防御方案的特点，更侧重于保护用户数据的安全。因此，在数据泄露防御算法中面临着一些新的挑战。本文以数据泄露防御算法为研究对象，以提高工作效率为前提，尽可能减少冗余数据所需要的空间，提出了一种改进的指纹算法，在数据泄露防御算法的研究方面做了有益的尝试，研究成果有较高的参考价值。具体工作内容如下：

（1）介绍了数据泄露防御技术的背景、数据泄露防御的关键技术以及数据泄露防御系统的结构等基础知识，明确了与传统的安全防御系统相比数据泄露防御系统所独有的特征。

（2）介绍了数据泄露防御系统，分别从防御原理、系统分类、系统结构和匹配算法等多方面加以描述。

（3）提出了一种改进的指纹算法，该算法的优点是将原有的文件转化成指纹进行存储，从而减少存储文件所需的空间，同时该算法有较高的匹配效率，能够提高数据泄露防御系统的工作效率。仿真实验结果表明：这种设计思想能使防御效果有较好的提升。

网络安全问题是一个复杂的研究课题，由于研究的时间有限，本文提出的数据泄露防御算法还不够完善，可以就以下各方面继续研究：

（1）匹配成功率的提高方法，对中文匹配的支持技术。

（2）指纹算法中指纹值求解过程的优化和指纹值准确性的进一步提高。

（3）基于本文的权值评估方法设计自学习功能。

致 谢

光阴荏苒，三年紧张而又充实的研究生生活即将结束，在硕士论文完成之际，我向所有支持、关心和帮助过我的人表示最诚挚的谢意。

衷心感谢我的导师李玲娟教授两年多来的谆谆教诲。李老师严谨的治学态度、兢兢业业的工作精神、待人真诚、襟怀坦荡的为人品质是我今生学习的榜样，她在科研上孜孜不倦的追求和勤奋不辍的精神使我受益匪浅、终生铭记。李老师在我学习期间不仅传授了做学问的方法，还传授了做人的准则，这些都将使我终生受益。无论是在理论学习阶段，还是在论文的选题、资料查询、开题、研究和撰写的每一个环节，无不得到导师的悉心指导和帮助。在此，谨向恩师致以最诚挚的谢意！

感谢共同学习、讨论和生活的师兄妹，在学习讨论中我们通力合作，在日常生活中我们其乐融融，在此感谢他们在长期学习生活中给予的支持、理解和帮助！

在我的成长和整个学习生活中，我的家人一直对我付出了无私的爱和关怀，本文的完成也是他们心血的回报。

最后，向所有给我的学习和生活带来帮助的人表示衷心的感谢！

攻读硕士学位期间的学术论文

本人攻读硕士学位期间发表的学术论文：

[1]张睿, 李玲娟. P2P 网络资源传播的研究. 全国计算机新技术与计算机教育论文集, 2007, 16(卷)下:79-82.

[2]李玲娟, 张睿. 数据泄露防御算法的研究. 计算机应用研究, 已录用.

攻读硕士学位期间参加的项目

(1)国家 863 课题：新型对等计算网络安全关键技术（No.2006AA01Z439）

(2)南京邮电大学校科研基金项目：网络安全性能评估的研究（No.NY207051）

参考文献

[1] San Mateo,Contact Informatin[OL], URL:http://www.tablus.com.

[2] Jim Nisbet. The Security Role For Content Analysis[R]. November 17, 2004. Company TABLUS.

[3] 中国数据泄露防护（DLP）市场分析[OL]. URL:http://netsecurity.51cto.com.

[4] Edward L.Jones.Metrics Based Plagiarism Monitoring[R].JCSC 16, 4(May 2001)by the Consortium for Computing in Small Colleges. 2001,253-261.

[5] 数据泄露防御如何保护企业信息资产[OL]. URL: http://safe.csdn.net/safe/data/Protect/200902/23-655.html

[6] 成功部署DLP技术的关键问题[OL]. URL: http://safe.csdn.net/safe/data/Protect/200902/20-639.html

[7] 李旭. 基于串匹配方法的文档复制检测系统研究[D]. 燕山大学：2005 年 11 月

[8] 邓爱萍,徐国梁,肖奔.基于串匹配方法的源代码复制检测技术研究[J].科学技术与工程,2007,7(10):2251-2254.

[9] Narayanan Shivakumar and Hector. SCAM:A copy detection mechanism for digital documents[J]. In Proceedings of the Second Annual Conference on the Theory and Practice of Digital Libraries,1995.

[10] N.Shivakumar, H.G.Molina. Building a Scalable and Accurate Copy Detection Mechanism[C]. The 1st ACM Conference on Digital Libraries, Bethesada, Maryland, USA, 1996:34-41

[11] C.Y.Cho, S.Y.Lee, C.P.Tan. Network forensics on packet finngerprints[J]. In 21st IFIP Information Security Conference (SEC 2006) Karlstad, Sweden, 2006.

[12] Lindoso A, Entrena L, Liu-Jimenez J, Millan, E.S. Increasing security with correlation-based fingerprint matching[J]. Security Technology, 2007 41st Annual IEEE International Carnahan Conference on 8-11 Oct. 2007 Page(s): 37 - 43 2007. IEEE Computer Society Press.

[13] Clough P. Plagiarism in Natural and Programming Languages:An Overview of Current Tools and Technologies[J]. Research Memoranda:CS-00-05, Department of Computer Science, University of Sheffield, 2000, 8(5):91-97.

[14] G Whale. Plague:plagiarism detection using program structure[R]. Dept of Computer Science Technical Report, University of NSW, Kensington, Australia, 1988.

[15] Boywer, Kevin W and Lawrence O. Experience using'MOSS'to Detect Cheating on Programming Assignments[C]. In 29th ASEE/IEEE Frontiers in Education Conference,San Juan,Puerto Rico,1999,18-22.

[16] Aiken A. Moss:A System for Detecting Software Plagiarism[OL]. http:// www.cs.berkeley.cdu/~aiken/moss.htm1.2006-04-02.

[17] Peter Mork, Beitao Li, Edward Chang, Junghoo Cho, Chen Li, and James Wang. Indexing tamper resistant features for image copy detection[OL], 1999. URL: http://citeseer.nj.nec.com/mork99indexing.html.

[18] Yamamoto T, Matsushita M, Kamiya T. Measuring Similarity of Large Software Systems Based on Source Code Correspondence[J]. Division of Software Science, Graduate School of Engineering Science, Osaka University,2002,4-5.

[19] David Gitchell and Nicholas Tran.Sim:A Utility For Detecting Similarity in Computer Programs[J]. In Proceedings of the 30th SIGCSE Technical Symposium, March 1999.

[20] E. Anderson and M. Arlitt. Full packet capture and offline analysis on 1 and 10 gb networks[R]. Technical Report HPL-2006-156, 2006.

[21] Park S.H, Woo G.. Source Code Similarity Detection Using Adaptive Local Alignment of Keywords[C]. In Eighth International Conference on Parallel and Distributed Computing, Applications and Technologies. 2007,179-180.

[22] WISE, MICHAEL J. YAP3:Improved Detection of Similarities in Computer Program and other Texts[R].Department of Computer Science,University of Sydney,2003.

[23] L.Prechelt, G.Malpohl and M.Phippsen. Finding Plagiarisms among a Set of Programs with Jplag[J]. Journal of Universal Computer Science, vol 8, no.11:1016-1038,2002.

[24] Brenda S. Baker and Udi Manber. Deducing similarities in java sources from bytecodes[M]. In Proc. of Usenix Annual Technical Conf, pages 179–190, 1998.

[25] Griswold. A Method for Protecting Copyright on Networks[C]. The Conference of Joint Harvard MIT Workshop on Technology Strategies for Protecting Intellectual Property in the Networked Multimedia Environment, Cambridge, England, 1993:214-221

[26] 史彦军,滕弘飞,金博.抄袭论文识别研究与进展[J].大连理工大学学报,2005,45(1):50~57.

[27] 车万翔,刘挺,秦兵,等.基于改进编辑距离的中文相似句子检索[J].高技术通讯,2004,(07):120-124.

[28] D.Gitchell, N.Tran. A Utility for Detecting Similarity in Computer Programs[J]. In the 30th SIGCSE Technical Symposium on Computer Science Education, New Orleans, Louisiana, USA, 1999:266-270.

[29] Andrei Broder. Algorithms for duplicate documents[R]. A Broder-Algorithms for near-duplicate documents February 18, 2005.

[30] P.Clough. Plagiarism in natural and programming languages:an overview of current tools and technologies[J]. Department of Computer Science, University of Sheffield, 2000, 1-31.

[31] Shan He, Kirovski D, Min Wu. Colluding Fingerprinted Video using the Gradient Attack[J], Acoustics, Speech and Signal Processing, 2007. ICASSP 2007. IEEE International Conference on Volume 2, 15-20 April 2007 Page(s):II-161 - II-164.

[32] Z.B.Andrei.Some Applications of Rabin's Fingerprinting Method[J]. Methods in Communications, Security and Computer Science, 1993,3(1):143-152

[33] 梁田贵,张鹏.算法设计与分析[M].北京:冶金工业出版社,2004:145-146

[34] S.B.Brenda. A Theory of Parameterized Pattern Matching:Algorithms and Applications[J]. The 25th Annual ACM Symposium on Theory of Computing, San Diego, CA, USA, 1993:592-601

[35] S.Schleimer, D.Wilkerson, and A.Aiken. Winnowing: Local algorithms for document fingerprinting[J]. in Proceedings of the ACM SIGMOD International Conference on Management of Data, 2003, pp 76–85.

[36] Samuel Mann, Zelda Frew. Similarity and Originality in Code:Plagiarism and Normal Variation in Student Assignments[C]. In Proceedings of the 8th

Australasian Conference on Computing Education. 2006, 143-150.

[37] Aleksi Ahtiainen,Sami Surakka,Mikko Rahikainen.Plaggie:GNU-licensed Source Code Plagiarism Detection Engine for Java Exercises[C]. In Proceedings of the 6th Baltic Sea Conference on Computing Education Research. Koli Calling, 2007, 141-142.

[38] Younhee Gil, Dosung Ahn, Sungbum Pan, Yongwha Chung. Access control system with high level security using fingerprints[J]. Proceedings of the 32nd Applied Imagery Pattern Recognition Workshop (AIPR'03) 2003 IEEE.

[39] Nevin Heintze. Scalable document fingerprinting[J]. In 1996 USENIX Workshop on Electronic Commerce, November 1996.

[40] Jiansheng Chen，Yiu-Sang Moon，A Minutiae-based Fingerprint Individuality Model[J]. Computer Vision and Pattern Recognition, 2007. CVPR '07. IEEE Conference on 17-22 June 2007 Page(s):1 – 7.

[41] Sebastian Niezgoda, Thomas:A Software Tool for Detecting Cut and Paste Plagiarism[J]. In Proceedings of the 37th SIGCSE Technical Symposium on Computer Science Education(SIGCSE'06). 2006, 51-55.