策略驱动的可扩展网络漏洞扫描研究

摘 要

随着信息技术的飞速发展和网络文化的普及，网络安全日益引起人们的广泛关注。在信息技术为政治、经济、文化等各个领域带来方便和高效的同时，安全保障问题逐渐成为制约其进一步发展的瓶颈。尤其当今对黑客技术感兴趣的人越来越多，而且伴随黑客站点的日益增多和黑客工具的免费下载，成为一名黑客的门槛也变得越来越低；与此同时，各种各样的病毒层出不穷，且传播速度越来越快，造成的危害也越来越大，信息安全管理正面临着巨大的压力与严峻的考验。

通常，计算机黑客和电脑病毒利用安全漏洞入侵计算机系统。因此只要找到并修补所有的安全漏洞，就可以抵御绝大部分的黑客攻击及病毒入侵。安全漏洞扫描技术能够检测网络系统潜在的安全漏洞，使网络管理员可以预先了解系统的脆弱性所在，及时修补漏洞，从而确保系统的安全。

本文首先介绍了漏洞的定义、产生原因、特征、分类，并详细分析了网络漏洞扫描器实现中的一些关键技术：端口扫描、操作系统识别、漏洞扫描的两种方法（知识匹配和模拟攻击）的原理。

其次，本文结合漏洞扫描，具体分析了漏洞的各个属性，讨论了应该如何综合利用漏洞的多个特征来多层次判断漏洞。并在此基础上，进一步分析了策略驱动的漏洞扫描原理，给出了一个采用策略驱动的安全漏洞扫描系统的总体设计方案，介绍了该系统的各组成模块及其功能实现。该系统基于信息收集的结果调度漏洞扫描插件，可减少漏洞扫描的盲目性，提高扫描效率。扫描程序以独立的插件形式实现，有利于系统的扩展。

此外，本文在分析漏洞扫描实现原理及对现有漏洞扫描工具扫描过程抓包分析的基础上，探讨了对常见的一些漏洞如 CGI 滥用、缓冲区溢出、拒绝服务及简单口令等的扫描实现原理，分析了如何从多方面提取特征码来判断漏洞，降低扫描误报率，并最终给出了多个漏洞的扫描实现代码。

最后通过扫描测试，证明了本扫描器可以有效地工作，并通过与

Xscan 扫描器选取相同漏洞扫描插件对同一台主机进行扫描的总时间对比，证明了本文所采用的策略驱动漏洞扫描确实提高了漏洞扫描的效率，且在针对开放端口较少的目标主机时，这种优势较为明显。

总之，本文设计的扫描器采用了可扩展的策略驱动网络漏洞扫描总体设计模型，通过先执行信息收集模块进行端口扫描与操作系统探测，再根据收集的主机开放端口、操作系统信息，针对该操作系统开放的网络服务调用相应的扫描插件进行安全漏洞检测，并在针对各个漏洞的扫描插件编写时尽量从多个特征值，多个方面来判断漏洞。最后通过实验证明这种方法能够减少漏洞扫描的盲目性，对漏洞进行多层次判别，提高扫描效率及准确性。

关键词：网络安全，漏洞扫描，信息采集，策略驱动

RESEARCH ON EXTENSIBLE POLICY-DRIVEN NETWORK VULNERABILITY SCANNING

ABSTRACT

With the popularization of information technology and network culture, system security, network intrusion and hacker skill are becoming hot topics. While information technology makes people more convenient and effective, the requirement for information technology security is also getting more urgent. Nowadays more and more people are getting interested in hacker skill, along with the increasing of the hacker websites in the Internet and free downloading of the hacker tools, the threshold to be a hacker is getting lower and lower. Meanwhile, various viruses emerge with more harm, higher frequency and spread speed, information security management is suffering tremendous pressure.

Generally, hacker and virus intrude a computer system by exploiting some security vulnerability. As long as all the security vulnerabilities are found and fixed, almost all the hacker attack and virus intrusion will be prevented. Security vulnerability scanning can detect the underlying security vulnerabilities in a system, so the network manager can know in advance where the weakness of the network is by using security scanning, then fix and patch these vulnerabilities in time, and the network security will be ensured consequently.

This thesis at first introduces the definition, the cause, characteristic, category of vulnerability, and some key technologies to implement network vulnerability scanner, including the principle of port scanning, vulnerability scanning and the two methods of vulnerability scanning such as knowledge matching and attack simulation.

Then, each attribute of vulnerability is analyzed in detail combined with vulnerability scanning, how to detect a vulnerability using the various attributes of the vulnerability is discussed in the thesis. Based on all these analysis, the

principle of policy-driven vulnerability scanning is analyzed, a design scheme of policy-driven vulnerability scanning system is proposed, all the system modules and their function implementation are introduced. The scanner is based on information gathering. Before the network vulnerabilities scanning is launched, the Ping test, OS detection and port scanning are first carried on. It can reduce blindness. The scanning plug-ins are used to examine the system vulnerabilities which are compiled to dynamic link library files. When new security vulnerability appears, we made the corresponding plug-in, and then put it into the plug-in storehouse. In this way the scanner has a good extensibility.

In addition, based on the analysis of the implementation principle of vulnerability scanning and the packet analysis captured when some scanner are working, the scanning methods to some common vulnerabilities such as CGI abuse, buffer overflow, DOS and simple password are put forward, how to use more characteristic code to detect vulnerability is also analyzed, and the programming using WINSOCK API is also introduced.

At last, the scanner's validity is proved through system testing. Moreover, compared to Xscan in scanning time with two tools both selecting the same vulnerability scanning plug-ins, it is proved that the policy-driven vulnerability scanning is more efficient, this advantage is more noticeable especially when the target host has less open port.

In a word, the design model of extensible policy-driven vulnerability scanner is introduced in this thesis, which makes use of the information of gathering module including port scanning, OS detection at first; then call the corresponding plug-ins to scan security vulnerability based on the information about the target host open ports and OS type. At last, it is proved that this model can reduce blindness, and it is more efficient and accurate by experiment.

KEY WORDS: network security, vulnerability scanning, information gathering, policy-driven

声　　明

本人郑重声明：所呈交的学位论文，是本人在指导教师的指导下，独立进行研究所取得的成果。除文中已经注明引用的内容外，本论文不包含其他个人或集体已经发表或撰写过的科研成果。对本文的研究做出重要贡献的个人和集体，均已在文中以明确方式标明。本声明的法律责任由本人承担。

论文作者签名：闫美凤　　日期：2006.4.29

关于学位论文使用权的说明

本人完全了解太原理工大学有关保管、使用学位论文的规定，其中包括：①学校有权保管、并向有关部门送交学位论文的原件与复印件；②学校可以采用影印、缩印或其它复制手段复制并保存学位论文；③学校可允许学位论文被查阅或借阅；④学校可以学术交流为目的，复制赠送和交换学位论文；⑤学校可以公布学位论文的全部或部分内容（保密学位论文在解密后遵守此规定）。

签　　名：闫美凤　　日期：2006.4.29

导师签名：彭新志　　日期：2006.4.29

第一章 绪论

1.1 课题背景

随着计算机网络和通讯技术的快速发展，网络应用的日益丰富，利用开放的网络环境进行全球通信已成为时代发展的必然趋势。近几年内，上网人数以几何级数的数量急剧增长，网络已经成为人们生活的一部分。然而网络在提供便利与高效的同时，也带来了许多安全隐患。

调查显示，黑客攻击系统的方式往多样化方面发展，黑客工具和病毒、木马的差异区分在减少，利用系统漏洞攻击的手段也层出不穷[1]。2003年1月爆发的SQL Slammer利用的是SQL Server缓冲区溢出漏洞，8月份蔓延的“冲击波”病毒利用了Windows系统的DCOM RPC缓冲区漏洞，这些利用系统漏洞的攻击给安全管理带来了巨大的压力。可见，进行漏洞扫描，评估系统的安全性，及时修补这些安全漏洞可以在很大程度上减少系统被攻击的可能性，进而保护系统的安全。因此，对漏洞扫描技术的研究具有重要的意义。

1.1.1 国际及国内网络安全现状

黑客攻击、计算机病毒破坏和网络金融犯罪已构成对世界各国的实际威胁。在各领域的计算机犯罪和网络侵权方面，无论是数量、手段，还是性质、规模，都已经到了令人咋舌的地步。据统计，全球平均每20秒就发生1次网上入侵事件，黑客一旦找到系统的薄弱环节，所有用户均会遭殃。而且近年来，黑客攻击已从单一的利用计算机病毒搞破坏和用黑客手段进行入侵攻击转变为使用恶意代码与黑客攻击手段结合的方式，使得这种攻击具有：传播速度惊人、受害面惊人和穿透深度惊人的特点，往往一次攻击就给受害者带来严重的破坏和损失[2]。

我国虽然在国家范围的网络建设方面发展迅速，取得了巨大的成绩。但是，由于基础薄弱，工作人员防范意识淡薄，安全隐患巨大，计算机系统也多采用开放式的操作系统，安全级别较低，很难抵抗黑客的攻击。目前我国 95%与互联网相联的网络管理中心都遭受过境内外黑客的攻击或侵入，其中银行、金融和证券机构是黑客攻击的重点。另外，网络安全存在的严重隐患还表现在，不能快速有效抑制各种计算机病毒的入侵，2003 年爆发的“蠕虫王”、“冲击波”病毒，2004 年 5 月爆发的“网络天

空”、“震荡波”病毒，以及2005年8月的被称为历史上最快利用微软漏洞攻击电脑的病毒“狙击波”等病毒及其变种，都给计算机用户造成了较严重的损失。根据国家计算机病毒应急处理中心的调查显示：2003年中国计算机病毒感染率高达5.57%，因病毒感染造成经济损失的比率达到了63.57%。所以说，我国面临的网络安全问题是非常严峻的。

1.1.2 网络安全问题的原因

有诸多的原因会导致网络安全问题，主要原因如下：

首先，网络协议自身的缺陷是造成网络不安全的主要原因，如IP层协议就有许多安全缺陷。IP地址可以软件设置，这就造成了地址假冒和地址欺骗两类安全隐患；IP协议支持源路由方式，即源点可以指定信息包传送到目的节点的中间路由，这就提供了源路由攻击的条件。再如应用层协议TELNET，FTP，SMTP等协议缺乏认证和保密措施，这就为否认、拒绝等欺瞒行为开了方便之门。

其次，随着应用软件、操作系统变得越来越复杂，使得软件设计者在设计阶段无法预料到程序运行时的系统状态，更无法精确的预测在不同系统状态下会发生什么结果，所以系统往往存在着很多的漏洞。我们使用的操作系统无论是Windows还是UNIX几乎都或多或少地存在安全漏洞，众多的服务器、浏览器、桌面软件都发现过安全隐患，可以说任何一个软件系统都可能会因为程序员的一个疏忽、设计中的一个缺陷等原因而存在漏洞。计算机紧急响应组（Computer Emergency Response Team，CERT）统计发布显示，最近几年漏洞数目有异常显著的提高：2001年2437个漏洞，2002年4129个漏洞，2003年3784个漏洞。可见，安全漏洞的存在是在所难免的[3]。

再次，黑客的门槛越来越低。系统的安全漏洞和系统的加密措施已不再像以前那样仅为为数不多的专业人士知道。在互联网上，有数以万计的黑客站点在时时刻刻地发布这些信息，并提供各种工具和技术以利用这些安全漏洞来进行攻击。一个普通的计算机用户能轻易地获取这些信息，轻松地变为一个具有很大威胁的潜在黑客。尤其是现在还缺乏针对网络犯罪卓有成效的反击和跟踪手段，使得黑客攻击的隐蔽性好，“杀伤力”强。所以黑客的攻击是网络安全的一个主要威胁。

最后，网络管理不完善造成安全隐患。据IT界企业团体美国信息技术协会ITAA（Information Technology Association of America）的调查显示：美国90%的IT企业对黑客攻击准备不足。目前，美国75%-85%的网站都抵挡不住黑客的攻击。

因此，从根本意义上讲，绝对安全的计算机不存在，绝对安全的网络也是不可能

的。从理论上说，开放的系统都会有安全漏洞[4]。而且正是这些安全漏洞被一些拥有很高的技术水平和超强耐性的黑客所发现，又被他们用网络发布，被大多数攻击者所利用，从而使网络入侵事件层出不穷，愈演愈烈。

1.1.3 现有的安全技术

我们要保护网络的安全，不能单独地依靠某一种技术或方法，而应该从各个层面来提高整体的安全性。综合来看，目前最常用的网络安全防护技术主要有三种：

1） 防范技术：主动防范黑客进入被保护的网络，例如防火墙（Firewall）、物理隔离等；

2） 探测技术：帮助系统管理员探测网络存在那些脆弱性，例如漏洞扫描器（Vulnerabilities Scanner）、端口扫描器（Port Scanner）等；

3） 检测技术：当网络发生异常时能及时检测到入侵行为，例如入侵检测系统（Intrusion Detection System）等。

以上三种技术构架了网络安全防御的综合体系结构，缺一不可。防范技术属于第一级的防护措施，它的功能主要是阻断非法的连接，是主动防范的措施。但是，防火墙存在一些不足之处，首先是容易被黑客研究其结构；其次是一旦防火墙被攻陷，如果没有其他保护措施，则内部网络将暴露无遗；还有就是对于内部的安全隐患并不能有效的控制。检测技术属于第二级的防护措施，它的功能是能够及时检测入侵并能留下入侵者入侵时的记录。它的缺点是：误报率及漏报率较高，缺乏智能化的响应和自身的安全问题。探测技术相对于前两种技术更为灵活，它是一种主动的防范措施，是一个网络安全的评估系统，其主要功能是探测网络安全的隐患，能够在被黑客攻击前事先对系统和网络进行安全扫描，分析它们存在的安全问题，及早发现漏洞并加以改进，防患于未然。只有将这三者有机地结合起来，才能更为有效地提高网络的整体安全，这是未来网络安全的一个发展趋势。

综上所述，漏洞扫描可以发现系统潜在的安全漏洞，防患于未然，是网络安全的重要技术之一，对安全漏洞扫描技术的研究具有重要的意义。

1.2 本文的主要研究内容

本文详细论述了网络漏洞扫描的理论基础知识，并在分析漏洞特征属性和网络漏洞扫描系统的设计目标的基础上，给出了安全漏洞扫描系统的总体设计方案。整个系统

的设计与实现采用了策略驱动的扫描调度方法，漏洞检测采用插件的方式，当有新安全漏洞出现时，编写相应的插件存入插件库就完成了新漏洞的扩展，使系统具有良好的扩展性。

具体来说，本文的研究内容如下：

1) 在深入分析了漏洞的定义、产生原因、特征、分类的基础上，论述了漏洞和扫描的关系。

2) 对网络安全扫描技术的基本原理进行了全面、细致的分析。详细介绍了实现网络漏洞扫描系统所需要的一些关键技术：端口扫描、操作系统识别、漏洞扫描的两种方法（知识匹配和模拟攻击）的原理及系统实现涉及到的 Winsock 编程技术。

3) 在分析漏洞各种特征属性的基础上，分析了漏洞扫描应该如何利用漏洞的多个特征来综合多层次判断漏洞。并在此基础上，分析了策略驱动的漏洞扫描，给出了安全漏洞扫描系统的总体设计方案。对各个模块以及其实现作了比较详细的陈述。在进行真正的安全漏洞检测前，先进行Ping连通测试和端口扫描、操作系统识别。Ping连通测试主机是否处于活跃状态，端口扫描收集活跃主机的开放端口服务信息，操作系统识别判断目标的操作系统类型。在漏洞检测时，根据活跃主机的操作系统及其上开放端口信息，调用相应的漏洞扫描插件，减少了漏洞扫描的盲目性，提高了系统效率。漏洞检测采用插件的方式。所有的插件放在扫描插件库中，这些插件提供统一的接口为扫描引擎所调用，保证了系统的扩展性。

4) 此外，本文在分析漏洞扫描实现原理及对现有漏洞扫描工具扫描过程抓包分析的基础上，探讨了对常见的一些漏洞如 CGI 滥用、缓冲区溢出、拒绝服务及简单口令等的扫描实现原理，分析了如何从多方面提取特征码来判断漏洞，并最终给出了多个漏洞的扫描实现代码。

5) 最后，通过扫描测试，证明了本扫描器可以有效地工作，并通过与 Xscan 扫描器选取相同漏洞扫描插件对同一台主机进行扫描的总时间对比，证明了本文所采用的策略驱动漏洞扫描确实提高了漏洞扫描的效率，且在针对开放端口较少的目标主机时，这种优势较为明显。

第二章 漏洞及漏洞扫描概述

2.1 漏洞概述

网络攻击、网络入侵等安全事故的频繁发生，多数是由于系统存在安全漏洞导致的。对系统安全漏洞的研究有利于发现未知漏洞和及时修补已发现的漏洞，从而减少系统安全漏洞被攻击者利用的可能性，降低攻击事件的发生率。漏洞研究是实现网络漏洞扫描的基础，只有对系统漏洞有充分的了解，才能有针对性的给出完整的漏洞扫描方案。

2.1.1 漏洞概念及产生原因

系统安全漏洞（vulnerability）也叫系统脆弱性，简称漏洞，是指在计算机系统中硬件、软件及协议的具体实现或系统安全策略上存在安全方面的缺陷，从而使攻击者可以非法入侵系统或未经授权访问或破坏系统[5]。

系统安全漏洞是针对计算机安全而言的，广义的系统安全漏洞是指一切导致威胁或破坏计算机系统安全（可靠性、可用性、保密性、可控性、不可抵赖性）的因素。每个平台无论是硬件还是软件都存在着漏洞，没有绝对安全的系统。

系统安全漏洞的存在本身不会对系统安全造成危害，问题的关键在于攻击者可以利用这些漏洞达到其非法目的，例如：攻击者通过利用某种漏洞获得计算机系统的额外权限，在未授权的情况下访问或破坏系统，从而影响或破坏系统的安全性。

漏洞的产生主要是由于程序员不正确和不安全编程引起的。大多数程序员在编程初始就没有考虑到安全问题。在后期，由于用户不正确的使用以及不恰当的配置都可导致漏洞的出现。漏洞产生的原因主要归为以下几种：

1) 输入验证错误

漏洞的产生是由于未对用户提供的输入数据的合法性作适当的检查，这种错误导致的安全问题最多。

2) 访问验证错误

漏洞的产生是由于程序的访问验证部分存在某些可利用的逻辑错误或用于验证的条件不足而造成的。这类缺陷使得非法用户绕过访问控制成为可能，从而导致未经授权的访问。

3) 竞争条件错误

漏洞的产生是由于程序在处理文件等实体时，在时序和同步方面存在问题，在处理的过程中可能存在一个机会窗口使攻击者能够施以外来的影响。

4) 意外情况处置错误

漏洞的产生是由于程序在它的实现逻辑中没有考虑到一些本应该考虑的意外情况。这种错误比较常见，如没有检查文件是否存在就直接打开设备文件导致拒绝服务，没有检查文件是否存在就打开文件提取内容进行比较而绕过验证等。

5) 配置错误

漏洞的产生是由于系统和应用的配置有误，或是软件安装在错误的地方，或是参数配置错误，或是访问权限配置错误等。

6) 环境错误

由一些环境变量的错误或恶意设置造成的漏洞，导致有问题的特权程序可能去执行攻击代码。

7) 设计错误

这个类别是非常笼统的，严格来说，大多数漏洞的存在都是设计错误。

2.1.2 漏洞的分类

由于对漏洞研究的抽象层次不同，会对漏洞提出不同的分类。以下主要根据漏洞的属性进行分类：

1) 按漏洞可能对系统造成的直接威胁分类[6]

(1) 远程管理员权限

攻击者不需要本地用户权限就可直接获得远程系统的管理员权限，通常通过攻击以root身份执行的有缺陷的系统守护进程来完成。漏洞的绝大部分来源于缓冲区溢出，少部分来自守护进程本身的逻辑缺陷。

(2) 普通用户访问权限

攻击者可以获得系统的普通用户存取权限，通常是利用服务器的某些漏洞。攻击者利用服务器的漏洞，取得系统的普通用户存取权限，对UNIX类系统通常是shell访问权限，对Windows系统通常是cmd.exe的访问权限，能够以一般用户的身份执行程序，存取文件。

(3) 本地管理员权限

攻击者在已有一个本地账号能够登录到系统的情况下，通过攻击本地某些有缺陷

的程序，得到系统的管理员权限。

(4) 权限提升

攻击者在本地通过攻击某些有缺陷的程序，把自己的普通权限提升为管理员权限。

(5) 本地拒绝服务

在攻击者登录到系统后，利用本地拒绝服务漏洞，可以使系统本身或应用程序崩溃。这种漏洞主要是因为程序对意外情况的处理失误，如写临时文件之前不检查文件是否存在，盲目跟随链接等。

(6) 远程拒绝服务

攻击者利用远程拒绝服务漏洞，无须登录即可对系统发起拒绝服务攻击，使系统或相关的应用程序崩溃或失去响应能力。这类漏洞通常是系统本身或其守护进程有缺陷或设置不正确造成的。

(7) 读取受限文件

攻击者通过利用某些漏洞，读取系统中本来没有权限读取的文件，且这些文件通常是安全相关的。这些漏洞的存在可能是文件设置权限不正确，或者是特权进程对文件的不正确处理。

(8) 远程非授权文件存取

攻击者可以不经授权地从远程存取系统的某些文件。

(9) 口令恢复

因为采用了很弱的口令加密方式，使攻击者可以很容易的分析出口令的加密方法，从而使攻击者通过某种方法得到密码后还原出明文来。

(10) 欺骗

利用欺骗漏洞，攻击者可以对目标系统实施某种形式的欺骗。这通常是由于系统的实现上存在某些缺陷。

(11) 服务器信息泄露

利用服务器信息泄露这类漏洞，攻击者可以收集到对于进一步攻击系统有用的信息。这类漏洞的产生主要是因为系统程序有缺陷，一般是对错误的不正确处理。

(12) 其它

虽然以上的几种分类包括了绝大多数的漏洞情况，可还是有可能存在一些上面几种类型无法描述的漏洞，把它们归到这里。

事实上，一个系统漏洞对安全造成的威胁远不限于它的直接可能性，如果攻击者

获得了对系统的一般用户访问权限，就极有可能再利用本地漏洞把自己升级为管理员权限。

2) 根据漏洞被攻击者利用的方式分类

漏洞的危害主要就在于被攻击者利用，使得攻击者有可能入侵系统。于是依据漏洞被利用的方式，系统安全漏洞可以分为以下几类：

(1) 本地（Local）攻击漏洞

利用此类漏洞，攻击者需要是系统的本地合法用户或已经通过其它攻击方法获得了本地权限的非法用户。

(2) 远程（Remote）攻击漏洞

利用此类漏洞，攻击者可通过网络，对连接在网络上的远程主机进行攻击。

3) 按漏洞严重性分类

按漏洞严重性的等级，可以将漏洞的严重性分成高，中，低三个级别。

(1) 高

大部分远程和本地管理员权限大致对应为高。

(2) 中

大部分普通用户权限，权限提升，读取受限文件，远程和本地拒绝服务大致对应中级。

(2) 低

大部分远程非授权文件存取，口令恢复，欺骗，服务器信息泄露漏洞属于低级别。

但这只是一般的情况，具体情况还需要具体分析。

2.1.3 漏洞数据库及漏洞命名标准

已有许多不同的漏洞库可以用为科学研究的基础，容易得到的漏洞数据的主要提供者之一是美国计算机应急反应小组（http://www.cert.org），此外还有一些其他的可靠的漏洞信息源，如：事件响应与安全组织论坛（http://www.first.org）；ISS（Internet Security Systems）公司（http://xforce.iss.net）；美国能源部下属的计算机事件咨询中心CIAC（Computer Incident Advisory Capability, http://ciac.llnl.gov/ciac/index.html）；Securityfocus公司（http://www.securityfocus.com）等。

为了使不同数据库之间共享统一标准的漏洞数据，可采用一个统一的标准。而当

前得到大家认可和广泛应用的当属CVE[7]（Common Vulnerabilities and Exposures 公共漏洞和暴露）这一标准。CVE是一个对安全漏洞和其他安全问题的标准化的统一名字列表，CVE致力于所有安全漏洞及安全问题的命名标准化。目前这个命名方案由MITRE公司（http://www.cve.mitre.org/）主持。

CVE实际上是一个字典，其主要作用是为已知的安全弱点或漏洞提供通用名称，其目的是促进不同安全工具和漏洞数据库之间的数据共享。对于同一个弱点或者漏洞，在CVE中只存在一个名字，并且对该弱点或者漏洞有一个标准的描述。其命名过程为：对每一个安全弱点或漏洞，CAN（CVE Candidate Numbering Authority CVE权威候选编号机构）会为其分配一个编号，其形式为CAN-xxxx-编号（其中xxxx为发现该漏洞的年份，编号是一个唯一的序号），然后提交给CVE Editorial Board（CVE编辑委员会）以确定是否列为CVE的一个条目。若通过，CAN编号将以CVE编号形式给出该漏洞（不一定与原来CAN编号一致），并编入CVE正式列表在其站点公布。

2.1.4 漏洞的特征与属性

漏洞和具体的系统环境密切相关。漏洞会影响到很大范围的软硬件设备，包括系统本身及其支撑软件，网络客户和服务器软件，网络路由器和安全防火墙等。在不同种类的软、硬件设备中，同种设备的不同版本之间，由不同设备构成的不同系统之间，以及同种系统在不同的设置条件下，都存在各自不同的安全漏洞问题。

漏洞问题与时间紧密相关。随着时间的推移，旧的漏洞会不断得到修补或纠正，新的漏洞会不断出现，因而漏洞问题会长期存在。脱离具体的时间和具体的系统环境来讨论漏洞问题是毫无意义的。只能针对目标系统的操作系统版本、其上运行的软件版本以及服务运行配置等实际环境来具体谈论其中可能存在的漏洞。

漏洞的上述特点决定了漏洞完整描述的独特性。在对漏洞进行研究时，除了需要掌握漏洞本身的特征属性，还要了解与漏洞密切相关的其它对象的特点。漏洞的基本属性有：漏洞类型、造成的后果、严重程度、利用方式、环境特征等。除此之外，漏洞还与其它对象密切相关，比如：存在漏洞的软（硬）件、操作系统、相应的补丁程序和修补漏洞的方法等。

2.1.5 漏洞的发现

漏洞是客观存在的，但是在一定时间内是不一定能被发现的。漏洞是不断地被人们发现并被公布出来。

发现漏洞也会因目的不同而有不同的出发点。从黑客的角度看，他们会不断地去发现系统漏洞，从而利用漏洞来进行入侵。事实上，在漏洞的发现史上最初漏洞是被攻击者先发现的。从系统安全维护的角度来看，系统维护者也需要努力发现可能存在的漏洞，在漏洞被攻击者发现和利用之前将之修补好。

总体来说，漏洞的发现者主要是程序员、系统管理员、安全服务商组织、黑客以及普通用户。程序员、系统管理员和安全服务商组织主要是通过测试不同应用系统和操作系统的安全性来发现漏洞。而黑客主要是想通过发现并利用漏洞来从事攻击活动。普通用户只是偶尔在不经意间发现漏洞，可能性比较小。

漏洞扫描器对于管理员来说则是尽可能地在漏洞被他人利用之前发现，提供具体的漏洞检测报告并给出修补方案。

2.2 漏洞扫描的概念及分类

漏洞扫描（Vulnerability Scanning）就是对计算机系统或者其它网络设备进行相关的安全检测，以找出安全隐患和可被黑客利用的漏洞[8]。

漏洞扫描技术可以从不同角度进行分类。从扫描对象来分，可以分为基于网络的扫描（Network-based Scanning）和基于主机的扫描（Host-based Scanning）[9][10]：

基于主机的扫描是从一个内部用户的角度来检测操作系统级的漏洞，主要用于检测注册表和用户配置中的漏洞。基于主机的扫描的优势在于它能直接获取主机操作系统的底层细节（如重要文件和目录的权限、共享文件系统、特殊服务和配置的细节等），此类漏洞扫描器与相应主机的操作系统紧密相关。

基于网络的扫描是从外部攻击者的角度对网络及系统架构进行的扫描，主要用于查找网络服务和协议中的漏洞[11]。基于网络的扫描可以及时获取网络漏洞信息，有效的发现那些网络服务和协议的漏洞，同时能够有效的发现那些基于主机的扫描不能发现的网络设备漏洞（如路由器、交换机、远程访问服务和防火墙等存在的漏洞）。

从扫描方式来分，可以分为主动扫描（Active Scanning）与被动扫描（Passive Scanning）[12]：

主动扫描是传统的扫描方式，拥有较长的发展历史，它是通过给目标主机发送特定的包并收集回应包来取得相关信息的，当然，无响应本身也是信息，它表明可能存在过滤设备将探测包或探测回应包过滤了。主动扫描的优势在于通常能较快获取信息，准确性也比较高。缺点在于[13]：①易于被发现，很难掩盖扫描痕迹；②要成功实施主动扫描通常需要突破防火墙，但突破防火墙是很困难的。

被动扫描是通过监听网络包来取得信息。由于被动扫描具有很多优点，近来倍受重视，其主要优点是对它的检测几乎是不可能的。被动扫描一般只需要监听网络流量而不需要主动发送网络包，也不易受防火墙影响。而其主要缺点在于速度较慢而且准确性较差，当目标不产生网络流量时，就无法得知目标的任何信息。

本文研究的是基于网络的主动扫描技术，下面将主要介绍网络漏洞扫描技术的有关方法及原理。

2.3 网络漏洞扫描方法

网络漏洞扫描是网络安全领域中一种换位思考的新的安全解决思路，由Dan Farmer和Wiestse Verema在1995年提出。其基本思想是模仿入侵者的攻击方法，从攻击者的角度来评估系统和网络的安全性，网络漏洞扫描能够更准确地向网络安全管理人员指明系统中存在安全问题的地方以及加强管理的方向，对网络安全管理人员来说比其它安全方法更有指导性和针对性。

其原理为[14]：通过模拟攻击者的攻击行为，根据系统对合法和不合法数据包的回应来探测存在的漏洞。从判别目标主机是否存在漏洞的依据来看，漏洞扫描方法可以分为知识匹配和模拟攻击。

2.3.1 知识匹配

前面介绍过漏洞和存在的系统环境有密切的相关性，一定的安全漏洞存在于一定的系统环境之中。基于知识匹配的漏洞检测，就是根据每个漏洞存在的基本环境形成相应的漏洞知识上下文，比如操作系统类型、版本，所开放服务的应用程序、版本等，然后通过收集目标主机的相关基本信息，如果满足知识库中所描述的基本前提，那么就认为目标具有相应的安全漏洞，否则就认为没有。

可见，知识匹配的前提就是要首先收集目标主机相关信息，信息收集方法主要有：通过端口扫描确定目标主机的开放端口、通过操作系统探测确定目标主机的操作系统类型等。

这种方法的缺点在于：由于信息收集的准确率以及相关漏洞在运行时刻安全上下文的不确定性导致检测结果的可靠性较低。它的好处就是没有攻击性，不会对真正具有相关漏洞的目标带来任何影响。

2.3.2 模拟攻击

模拟攻击就是模拟黑客攻击手法对目标主机实施攻击，逐项检查目标系统可能存在的漏洞。如果模拟攻击达到了漏洞利用的结果，那么就认为目标具有相应的安全漏洞，反之就认为没有。

相对于基于知识匹配的漏洞检测而言，模拟攻击检测具有更大的准确率，因为它确确实实显示了某个漏洞能否被利用的事实，其缺点在于攻击性较强，可能对存在漏洞的系统造成破坏。

常用的模拟攻击方法有：缓冲区溢出、拒绝服务攻击、口令攻击等。下面将分别介绍这几种攻击方法。

1) 缓冲区溢出攻击

缓冲区溢出（buffer overflow）是一种非常普遍和严重的漏洞，在各种操作系统以及应用程序中广泛存在，缓冲区溢出攻击也是网络攻击技术中最有效的一种攻击技术[15][16]。通过缓冲区溢出攻击，可以使远程攻击者有机会获得目标系统的部分或者全部控制权。

它的原理是向一个有限空间的缓冲区复制了超长的字符串，而程序自身却没有进行有效的检验，从而导致程序运行失败，系统重新启动，甚至停机。因此，缓冲区溢出这种程序设计缺陷变成了黑客进行系统攻击的一种手段，他们有意识地通过往程序的缓冲区写超出其长度的内容，破坏程序的堆栈，从而使程序转而执行其它指令，以达到攻击的目的。

造成缓冲区溢出的根本原因是程序中没有仔细检查用户或程序接口的输入参数。例如下面一个C程序：

```
void function(char*str){
char buffer[16];
strcpy (buffer,str);
}
```

上面的strcpy()将直接把str中的内容复制到buffer中。这样只要str的长度大于16，就会造成buffer溢出，使程序运行出错。

当然，随便往缓冲区中填写东西造成它溢出一般只会出现Segmentation fault（分割失败）错误，而不能达到攻击的目的。最常见的手段是通过制造缓冲区溢出使程序运行一个用户shell，再通过shell执行其他命令。如果该程序属于root且有suid权限的话，攻击者就获得了一个有root权限的shell，可以对系统进行任意操作了。

缓冲区溢出漏洞和攻击有很多种形式。最简单和常见的缓冲区溢出攻击类型就是在一个字符串里综合代码植入和活动纪录技术。

在检测远程主机的缓冲区溢出漏洞时，可以模拟攻击者实施缓冲区溢出攻击。当目标系统被攻击后出现异常，则此漏洞存在。

另外，网络漏洞扫描程序在对目标主机进行安全扫描时，一般不期望目标主机出现异常。在这种情况下，我们也可以分析目标应用程序在漏洞修补前后对一些特定发送代码的反映，依据其不同来判断是否存在漏洞。

2) 拒绝服务攻击

拒绝服务（Denial of Service DoS ），从网络攻击的各种方法和所产生的破坏情况来看，DoS算是一种很简单但又很有效的进攻方式。这种攻击行动使网站服务器充斥大量要求回复的信息，消耗网络带宽或系统资源，导致网络或系统不胜负荷以至于瘫痪而停止提供正常的网络服务。

DoS攻击的基本过程是：首先攻击者向服务器发送众多的带有虚假地址的请求，服务器发送回复信息后等待回传信息，由于地址是伪造的，所以服务器一直等不到回传的消息，分配给这次请求的资源就始终没有被释放。当服务器等待一定的时间后，连接会因超时而被切断，攻击者会再度传送新的一批请求，这个过程周而复始，最终导致服务器因资源被耗尽而瘫痪。

分布式拒绝服务攻击（Distributed Denial of Service DDoS ）：它是一种基于DoS的特殊形式的拒绝服务攻击，是一种分布、协作的大规模攻击方式，主要攻击比较大的站点。DDoS攻击是利用一批受控制的主机向目标主机发起攻击，这样来势迅猛的攻击令人难以防备，因此具有较大的破坏性。

DDoS攻击分为3层：攻击者、主控端、代理端。攻击者所用的计算机是攻击主控台，可以是网络上的任何一台主机，甚至可以是一个活动的便携机。攻击者操纵整个攻击过程，它向主控端发送攻击命令。主控端是攻击者非法侵入并控制的一些主机，这些主机还分别控制大量的代理主机。主控端主机的上面安装了特定的程序，因此它们可以接受攻击者发来的特殊指令，并且可以把这些命令发送到代理主机上。代理端同样也是攻击者侵入并控制的一批主机，它们上面运行攻击器程序，接收并执行主控端发来的命令。代理端主机是攻击的执行者，真正向受目标主机发送攻击。

攻击者发起DDoS攻击的第一步，就是寻找在INTERNET上有漏洞的主机，入侵有安全漏洞的主机并获取控制权。第二步在入侵主机上安装攻击程序，其中一部分主机充当攻击的主控端，一部分主机充当攻击的代理端。最后各部分主机各司其职，在攻

击者的调遣下对攻击目标发起攻击。由于攻击者在幕后操纵，所以在攻击时不会受到监控系统的追踪，身份不容易被发现。

3) 口令攻击

口令攻击也是漏洞扫描中模拟攻击的一种，但是因为其原理与其他的攻击相差较大，而且最终若攻击成功的话，可以直接得到登陆目标主机系统的用户名和口令。所以这里单独讨论对网络服务的远程字典攻击。

当前很多应用和网络服务都设置了用户名和口令，如TELNET，FTP，POP3和Windows SMB等服务。入侵者如果获得某些服务的用户名和口令，就可以轻而易举地取得对这些服务的操作权。由于用户和网络管理员的疏忽，某些用户标识和口令的设置太过简单，入侵者可以通过设置一个由一系列可能的用户名和口令组成的字典文件，让程序自动读取该字典对目标主机的网络服务进行试探登陆，这样就有可能发现一些弱用户名/口令。另外，通过Unix的Finger服务或Windows的NetBIOS（网络基本输入输出系统服务）服务可以获取目标主机上的用户列表，这样对这些指定用户进行字典攻击的成功率将会大大的提高。

2.4 现有漏洞扫描器的介绍及优缺点分析

国内外开发了很多的网络安全扫描工具，早期较为著名的是1995年4月Dan Farmer和Weitse Venema推出的SATAN（Security Analysis Tools for Auditing Network）。它运行在Unix环境下，扫描的目标是Unix系统，SATAN为用户提供一个HTML界面，是第一个把扫描结果以用户友好的格式来显示的扫描工具，SATAN的推出引起很大的轰动，但缺点是只能扫描几个比较老的漏洞，而且没有更新扩展功能，所以，该系统虽风光一时，但现在已经落伍。

NMAP （Network Mapper）是当前较受欢迎的扫描器，其特点是提供了比较全面的端口扫描方法及利用指纹技术的远程操作系统识别功能，它提供的端口扫描方法包括 TCP Connect扫描、TCP SYN扫描、TCP FIN扫描、TCP FTP Proxy扫描、利用IP Fragments的SYN/FIN扫描、TCP ACK和Windows扫描、ICMP扫描等诸多扫描技术。Nmap虽然是一个强大的端口扫描工具，但遗憾的是它没有扫描漏洞的功能，无法对目标主机的脆弱性进行深入挖掘。

Nessus[17] 是法国人Renaud Derasion编写的，是第一个使用漏洞插件的扫描器。并且漏洞数据是与CVE标准兼容的。该系统采用C/S架构，在服务端采用了插件（plug-in）的体系，允许用户加入执行特定功能的插件，这些插件可以进行更快速和更复杂的

安全检查。优点是能够扫描较多的漏洞，且具有较强的扩展性。但它主要运行在UNIX系列操作系统平台，造成了一定的局限性。

X-Scan是由国内“安全焦点”编写的一个运行于Windows环境下的能够扫描漏洞的检测工具。它提供图形界面和命令行两种操作方式，但他的许多功能还需进一步完善，扫描的漏洞也较少。

总的来说，现有的扫描工具都有各自的特点和局限性。而我国开展扫描技术的研究工作起步比较晚，研究工作不够深入、系统性不强，而且综合性能全面的远程扫描工具非常少。因而，鉴于扫描技术在网络安全中的重要地位，我们必须深入开展远程扫描的关键技术研究。

2.5 漏洞扫描器的发展趋势

从最初专门为Unix系统编写的一些简单小程序，发展到现在，已经出现了可在多种操作系统平台上运行，且具有复杂功能的漏洞扫描系统。结合当前安全扫描器的评测因素，可以预测出安全扫描器的发展趋势[18][19][20]：

1) 使用插件（plugin）或者专用脚本

插件或者专用脚本有其特殊的优点：当需要增加更多的漏洞时，只需添加新的插件或者脚本即可；而且，若编写插件的语言规范化，则用户自己也可编写插件。这样，若充分采用这种技术将会增加扫描的漏洞数和增强安全扫描器的可扩展性。

2) 使用信息共享标准CVE

随着越来越多的安全组织和安全产品支持CVE编号标准，CVE已成为网络安全漏洞表示的事实上的标准。因此，采用CVE标准是当前漏洞扫描器的趋势之一。

3) 由安全扫描程序到安全评估专家系统

当前较成熟的扫描系统能整理扫描结果并形成报表，并对具体漏洞提出一些解决方法，但对网络的安全状况缺乏一个整体的评估，对网络安全没有系统的解决方案。未来的安全扫描系统，应不仅能扫描安全漏洞，还应能智能化地协助网络管理人员评估本网络的安全状况，并给出具体建议，修补安全漏洞，从而逐步发展成为一个安全评估专家系统。

第三章 关键技术的原理介绍及分析

3.1 主机扫描

这一阶段就是通过发送不同类型的 ICMP 或者 TCP、UDP 请求，从多种不同的方面检测目标主机是否存活。在这一阶段使用的技术通常包括 ICMP 扫射、广播 ICMP、非回显 ICMP、TCP 扫射、UDP 扫射。

3.1.1 ICMP 扫射

在说明ICMP扫射之前先简单介绍一下ICMP协议。互联网控制报文协议（ICMP）是IP层的一个组成部分，用来传递差错报文和其他需要注意的信息。经常用到的ping命令就是使用的ICMP。ICMP报文是在IP数据报内部传输的。ICMP的正式规范参见RFC792。

ICMP扫射利用了类型为8的ICMP报文，即ICMP回显请求。通常网络上收到ICMP回显请求的主机都会向请求者发送ICMP回显应答（类型为0）报文。这样，如果发送者收到来自目标的ICMP回显应答，就能知道目标目前处于存活状态，否则可以初步判断主机没有在线。使用这种方法轮询多个主机称为ICMP扫射。这是用来发现目标的最原始的方法。

ICMP扫射虽然非常简单，但它并不十分可靠。因为目标可以阻止对ICMP回显请求作出应答。

3.1.2 广播 ICMP

实现原理：将 ICMP 请求包的目标地址设为广播地址或网络地址，则可以探测广播域或整个网络范围内的主机。

缺点：只适合于 UNIX 类系统，Windows 会忽略这种请求包；且容易引起广播风暴。

3.1.3 非回显 ICMP

如果目标主机阻塞了 ICMP 回显请求报文，仍然可以通过使用其他类型的 ICMP

报文探测目标主机是否存活，例如类型为 13 的 ICMP 报文（时间戳请求）和类型为 17 的 ICMP 报文（地址掩码请求）。ICMP 时间戳请求允许系统向另一个系统查询当前的时间，ICMP 地址掩码请求用于无盘系统引导过程中获得自己的子网掩码。

对于 ICMP 地址掩码请求报文而言，虽然 RFC1122 规定，除非是地址掩码的授权代理，否则一个系统不能发送地址掩码应答（为了成为授权代理，必须进行特殊配置）。但是大多数主机在收到请求时都会发送一个应答，甚至有些主机还会发送差错报文应答。所以也可以使用类型为 17的 ICMP 报文来探测主机是否存活。

3.1.4 TCP 扫射

传输控制协议（TCP）为应用层提供一种面向连接的、可靠的字节流服务。它使用“三次握手”的方式建立连接。TCP 的正式规范参见RFC793。

建立 TCP 连接的“三次握手”的过程如下：客户端首先发送一个序号为 SEQ1 的 SYN标志置位的数据包到服务器端。服务器接收到该数据包后，如果相应端口正在侦听，则响应一个序号为 SEQ2 的 SYN/ACK数据包，此数据包的确认序号为 SEQ1+1；如果相应端口没有侦听，则响应一个 RST 数据包，拒绝建立连接。客户端如果接受到正确的 SYN/ACK 数据包，则再发送一个 ACK 数据包到服务器，此数据包的确认序号为 SEQ2+1。至此，连接建立成功。

从建立连接的过程可以知道，如果向目标发送一个 SYN 数据包，则无论收到一个 SYN/ACK 数据包或者一个 RST 数据包，都表明目标处于存活状态。这就是 TCP 扫描的基本原理。与此类似，也可以向目标发送一个 ACK数据包，按照 RFC793 的规定，如果目标存活，则会收到一个 RST 数据包。

TCP 扫射看起来比利用 ICMP 协议进行探测更加有效，事实也正是如此。但 TCP 扫射也不是百分之百的可靠，因为有的防火墙能够伪造 RST数据包，从而造成防火墙后的某个主机存活的假象。

3.1.5 UDP 扫射

用户数据报协议（UDP）是一个面向数据报的传输层协议。UDP 数据报也被封装在 IP 数据报之中，RFC768 是 UDP 协议的正式规范。

UDP 协议的规则之一是，如果接收到一份目的端口并没有处于侦听状态的数据报，则发送一个 ICMP 端口不可到达报文，否则不做任何响应。这样，如果向目标的特定端口发送一个 UDP 数据报之后接收到 ICMP 端口不可到达错误，则表明目标处

于存活状态，否则表明目标不在线或者目标的相应 UDP 端口是打开的。由于 UDP 和 ICMP 错误都不保证能到达，因此在一个数据报看上去是丢失的时候，还应该重新发送新的 UDP 数据报以确认目标没有发送错误消息。这种方法很不可靠，因为路由器和防火墙都有可能丢弃 UDP 数据报。另外，逐一扫描 UDP 端口通常是很慢的，因为 RFC1812 的 4.3.2.8 节对 ICMP 错误消息的速率做了规定（Windows 系统并没有遵守 RFC 的规定，因此对 Windows 系统例外）。例如 Linux 的内核（在 net/ipv4/icmp.h 中）限制产生目的不可到达消息的速率是每 4 秒 80 次，如果超过上限，则再增加 1/4 秒的延迟。Solaris 有着更严格的限制（大约每秒两次就会延迟），所以这要耗费相当长的时间。UDP 扫描也有一个好处，就是它可以使用 IP 广播地址，如果向广播地址的高端端口发送一个 UDP 数据报，在没有防火墙过滤的情况下，将收到很多来自目标网络的 ICMP 端口不可到达的错误消息。当然，这也可能造成扫描者自己的 DoS（Denial of Service 拒绝服务）。

3.2 端口扫描

端口扫描技术是一项自动探测本地或远程系统端口开放情况及策略和方法，通过端口扫描可以获取目标系统的开放端口、运行服务等信息。端口扫描是漏洞扫描中重要的一步：首先，不必要的端口开放本身就可以说是一种漏洞；其次，由于很多漏洞都是针对某个端口，某种服务的，因此端口扫描结果也可以作为判别漏洞的一个条件。

端口有TCP端口和UDP端口两类。由于TCP协议是面向连接的协议，针对TCP扫描方法比较多，扫描方法从最初的一般探测发展到后来的躲避IDS和防火墙的高级扫描技术。针对TCP端口的扫描，最早出现的是全连接扫描，随着安全技术的发展，出现了以躲避防火墙为目的的TCP SYN扫描以及其他一些秘密扫描技术，比如TCP FIN扫描、TCP ACK扫描等。UDP端口的扫描方法相对比较少，通常有UDP ICMP端口不可达扫描，下面将分别介绍他们。

3.2.1 TCP 端口扫描

一次成功的TCP连接应完成完整的三次握手过程，一般的网络编程接口对这个过程进行封装，使得建立端口连接非常简单。但是这个过程很容易被目标主机发现并记录下来。因此，为了有效隐蔽自己的同时又能实现端口扫描，TCP扫描往往利用

TCP/IP协议本身的特征完成一个非正常的握手过程。

TCP数据包的格式如图3-1所示：

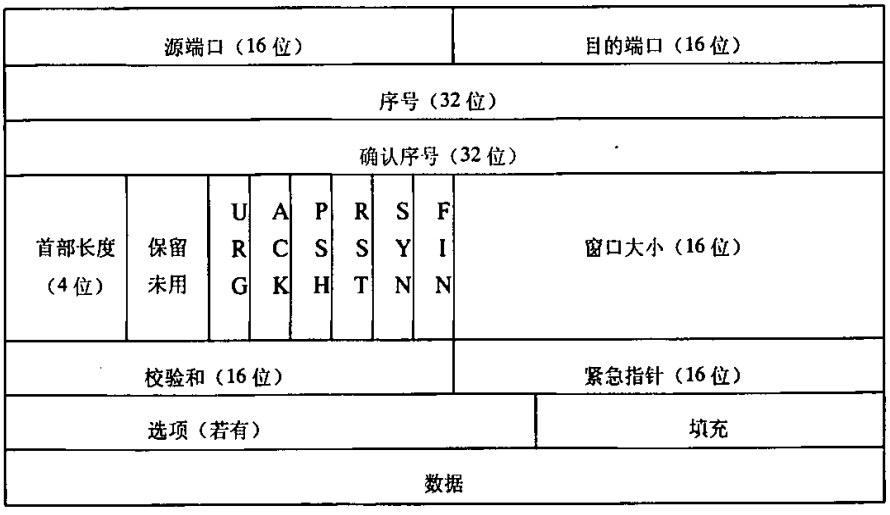

图 3-1 TCP 报文结构

Fig.3-1 TCP segment structure

首部中有 6 个标志位，分别表示为：ACK，SYN，RST，FIN， PSH 和 URG。各标志字段的含义如表 3-1：

表 3-1 TCP 标志字段各位的含义

Table.3-1 The meaning of every symbol field

位	该比特置 1 时的含义
URG	紧急指针字段有效
ACK	确认字段有效
PSH	本报文段请求推(push)操作
RST	连接复位
SYN	序号同步
FIN	发送方字节流结束

其中，ACK被置1表明确认序号有效。若被清0，表明数据包中确认号被忽略。

SYN被置1，用来建立一个连接。在连接请求过程中，SYN=1，ACK=0，指明确认字段没有使用。在对连接请求的应答过程中需要使用确认字段SYN=1， ACK=1。因而，SYN通常用来指明连接请求和连接请求被接受，并用ACK区分这两种情况。

RST被置1，用来重建一个连接（用于一台主机崩溃或一些其他原因而引起的通信

混乱时）。RST还被用来拒绝接收一个无效的TCP数据包，或者用来拒绝一个建立连接的请求。

FIN被置1，用来释放一个连接，表明发送者已没有数据要发送。在SYN和FIN置1的TCP数据包中都有一个顺序号保证数据按照正确的顺序被处理。

PSH被置1，表明数据的接收者被提示将收到的数据直接交给应用程序处理。在使用TCP/IP协议进行数据封装时，通过不同标志位的多种组合演变出许多TCP扫描技巧。

1) TCP connect() 扫描

这是最基本的TCP扫描方法。使用操作系统提供的connect()系统调用来与目标主机的TCP端口进行连接。如果connect()连接成功，说明目标端口处于监听状态。若连接失败，则说明该端口没有开放。

TCP connect()方法的最大的优点是不需要任何特殊权限，系统中任何用户都可以调用connect()函数。另一个优点就是速度快。但TCP connect()方法的缺点是它很容易被发觉，并且容易被防火墙过滤掉。同时目标主机的日志文件会记录一系列的有关该服务的连接建立并马上断开的错误信息。

因为TCP连接建立的三次握手特性，使得此方式的扫描的准确性在所有TCP扫描方式中是最准确，可以达到100%的准确性。本文设计实现的网络漏洞扫描系统主要是采用完全连接的扫描方式。

2) TCP SYN 扫描

这种技术通常认为是半开放扫描，这是因为扫描程序不必要打开一个完全的TCP连接。扫描程序发送的是一个SYN数据包，好象准备打开一个实际的连接并等待反应一样（参考TCP的三次握手建立一个TCP连接的过程）。一个SYN|ACK的返回信息表示端口处于侦听状态。一个RST返回，表示端口没有处于侦听状态。如果收到一个SYN|ACK，则扫描程序必须再发送一个RST信号，来关闭这个连接过程。 这种扫描技术的优点在于一般不会在目标计算机上留下记录。 但这种方法的缺点是：必须要有管理员权限才能建立自己的SYN数据包。

3) TCP FIN 扫描

在很多情况下，既使是 SYN 扫描也不能做到很隐秘。一些防火墙和包过滤程序能够监视到 SYN 数据包访问一个未被允许访问的端口，从而检测到这些扫描。然而，FIN 数据包却有可能通过这些监视程序。它的基本思想是[21][22]：扫描主机向目标端口发送一个 FIN，按照 RFC 793 的规定，目标系统中所有关闭的端口将会返回一个 RST

应答，而打开的端口会忽略这些请求。

优点：由于这种技术不包含标准的 TCP 三次握手协议的任何部分，所以无法被记录下来，从而比 SYN 扫描隐蔽得多，FIN 数据包能够通过只监测 SYN 包的包过滤防火墙。

缺点：跟 SYN 扫描类似，需要自己构造数据包，要求由超级用户或者授权用户访问专门的系统调用；而且，由于微软没有采用 RFC 793 这个标准。所以这种扫描方式对 Windows 系统无效。

4) TCP Xmas 扫描

这种方法是 TCP FIN 扫描的一个变种，其原理与 FIN 扫描类似。扫描主机向目标系统的指定端口发送一个含有 FIN，URG 和 PUSH 三个标志位的数据包。按照 RFC 793 规定，目标系统中所有关闭的端口应该返回一个 RST 分组。这种扫描方法通过三种标记位的组合，有可能通过检测 FIN 标记的包过滤程序，但同样对 Windows 系统无效。

5) TCP NULL 扫描

该方法与前两种方法类似，唯一不同的是向目标主机的指定端口发送的是一个不包含任何标志位的 TCP 数据包。

6) SYN/ACK 扫描

这种方法省掉了 TCP 三次握手的第一步，直接发送 SYN/ACK 包到目标端口。如果端口开放，目标主机返回 RST 数据包；如果关闭，则不返回任何信息。但考虑到收不到应答的原因还可能是超时，发出包被状态检测防火墙过滤掉等原因，这种方法也并不可靠[23][24]。

SYN/ACK 扫描的优点：快速，可绕开基木的 IDS。缺点：需要超级用户权限，不可靠。

3.2.2 UDP 扫描

这种扫描方式利用的是 UDP 协议，可用来发现目标打开的 UDP 端口。通过向目标端口发出一个 UDP 包，如果我们收到端口不可达的 ICMP 消息，端口就是关闭的，否则我们就假设它是打开的。

这种方法需要具有 root 权限，且由于 RFC1812 的有关规定（具体见 3.1.5），造成其扫描速度缓慢。不过 Microsoft 忽略了 RFC1812 的这个建议，所以我们可以快速扫描 Windows 主机上的 UDP 端口。

由于 UDP 是面向非连接的，扫描的准确性难以保证，但是可以通过重复发送 UDP 包来提高其准确性。

此外，端口扫描技术的不断发展在具体实现上也有所发展。在上述的这些基本扫描方式上，综合程序在实现这些扫描方法时的步骤、时序等，出现了慢速扫描和乱序扫描[25]等。

分析以上扫描方式，有些扫描方法只适用于部分操作系统，带有很大的局限性，有些难以保证准确性。综合考虑，本文介绍的漏洞扫描系统中的端口扫描技术上采用的是完全连接扫描、TCP SYN 扫描技术。

3.3 操作系统探测

资料显示，绝大多数的主机安全漏洞是与其运行的操作系统有直接的关系，甚至某些安全漏洞是个别操作系统版本所特有的，只有精确地识别出远程操作系统的类型，才能更有目的性地发掘漏洞和弱点所在，准确地对目标主机进行评估。目前用于探测操作系统的方法主要可以分为三类：利用系统旗标（Banner）信息、根据开放的端口、利用 TCP/IP 堆栈指纹。

3.3.1 利用系统旗标（Banner）信息

利用系统旗标信息是最原始的探测方法，至今仍然被包括 ISS 在内的许多网络安全扫描工具使用，因为在大多数情况下，操作系统的多种服务如 TELNET、HTTP、FTP、SMTP 等等都会暴露其“身份”，比如 TELNET 服务，连接上以后通常会返回类似下面的信息：

Red Hat Linux release 8.0 (Psyche)

Kernel 2.4.18-14 on an i686

根据此信息，可以判断出目标系统运行的操作系统是 Red Hat Linux 8.0。同时，这种方法实现起来也特别简单。但是在很多情况下，管理员出于安全考虑都会修改或者关闭旗标信息，或者目标机器并不提供有旗标信息的服务，在这样的情况下，这种方法就不能发挥作用了。

3.3.2 根据开放的端口

确定操作系统的另一个不太可靠但简单的方法是检查目标系统开放的端口号，并

将这些端口与不同操作系统的通用服务进行最佳匹配，可以大致判断目标系统所运行的操作系统。例如开放111端口（Sunrpc portmap服务）的系统通常是Unix，而开放137、138、139、445端口的系统通常是Windows。

3.3.3 TCP/IP 堆栈指纹

利用 TCP/IP 堆栈指纹识别操作系统是近年来发展迅速的一类技术。从原理上讲，由于对RFC相关文档的不同理解，不同的操作系统厂商在具体实现TCP/IP协议栈的时候就可能存在细微的差别。此外，还由于下面这些因素加大了彼此的差异[25]：

1) RFC规范中存在着一些可选的特性；

2) 某些私自对TCP/IP协议栈实现的改进；

3) 少数不遵从RFC规范的行为。

通过探测并仔细分析这些TCP/IP协议栈指纹的细微差异，就能够精确地辨识出远程操作系统，这就是“堆栈指纹识别”（Stack Fingerprinting）[26]技术。

目前主要的网络堆栈特征探测技术有如下几种：TCP响应分析、ICMP 响应分析、TCP重传延时分析、被动特征探测。

利用TCP/IP协议栈来识别操作系统的技术主要包括[27]：FIN探测、假标记探测、TCP ISN取样探测、IP ID取样探测、TCP时间戳探测、DF位探测、ACK值探测、ICMP错误信息抑制探测、分段控制探测和TCP选项探测等。

ICMP响应分析是通过发送UDP或ICMP的请求报文，然后分析ICMP应答，根据应答包的差异来区分不同的操作系统，具体内容参考文献[28]。

TCP重传延时分析发送SYN 请求三次握手但是放弃响应目标主机的SYN/ACK响应，目标主机会在延时定时器的控制下重传SYN/ ACK包，由于不同操作系统采用不同的重传延迟算法，重传的延时及次数是各异的。根据测量每个重传包之间的延时以及重传的包次数，将得来的信息与特征数据库中的已知指纹数据相匹配，即可获得目标主机操作系统的类型[29]。

被动协议栈指纹探测和主动的协议栈指纹探测很相似，不同之处在于这种方法不主动向目标系统发送分组，而是通过嗅探目标网络的通信，抓取从远程主机上发送的数据报，获取包括 TTL、窗口大小、DF 位、服务类型等在内的数据报属性，构成目标系统的指纹。这种方法主要被入侵者使用，因为它不容易被发现。

3.4 网络编程基础—WINSOCK 规范

安全漏洞扫描器利用TCP/IP协议和Client/Server的实现原理，不断连接目标服务器，而目标服务器通常响应这种连接请求，并返回数据。分析服务器发回的数据，就可判断漏洞是否存在。Socket（套接口）是应用程序与TCP/IP协议交互的编程接口，主要用于开发Client/Server方式的应用程序。

WinSock是从UNIX系统的Socket接口演变过来的。Socket诞生于20世纪80年代初，美国政府的高级研究工程机构（ARPA）给加利福尼亚大学Berkeley分校提供了资金，委托他们在UNIX操作系统下实现TCP/IP协议的开发接口。于是研发人员就为TCP/IP网络通信开发了一个API（应用程序接口）——Socket接口（套接字）。到了20世90年代，当时的一些网络开发商，包括现在的Sun和Microsoft公司等，共同制定了一套WINDOWS下的网络编程接口，即Windows Sockets规范，简称WinSock规范。WinSock规范是一套开放的、支持多种协议的Windows下的网络编程接口。从1991年的1.0版到1995年的2. 0. 8版，经过不断完善并在Intel，Microsoft，Sun等公司的大力支持下，已成为Windows网络编程的事实上的标准。目前，WinSock规范主要有1.1版和2.0版。WinSock库函数都包含在winsock.dll这个动态链接库（ WinSock1.1包含在该动态链接库中，而WinSock2.0中则是在ws2_32.d11中）中。应用程序调用WinSock API实现相互之间的通信。Windows套接字又利用下层的网络通信协议功能和操作系统调用实现实际的通信工作。他们之间的关系如图3-2所示。

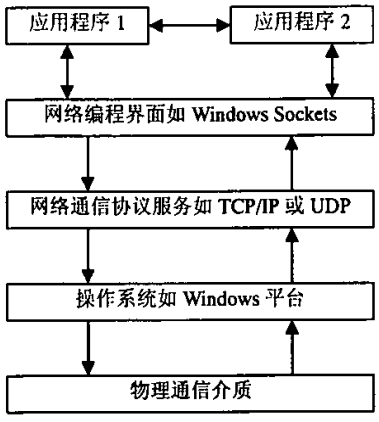

图 3-2 操作系统、应用程序和 Windows socket 的关系
Fig.3-2 The relationship among OS, application program and Windows socket

常用的Socket类型有两种：流式Socket（SOCK_STREAM）和数据报式Socket（SOCK_DGRAM）。流式Socket是一种面向连接的Socket，针对面向连接的TCP服务应用；数据报式Socket是一种无连接的Socket，对应于无连接的UDP服务应用。这两种套接口只能收到用户数据。此外还有一种是原始套接口（SOCK_RAW）保留了数据包中的完整IP头，可以通过原始套接口构造一些数据包的特殊字段，并且可以通过它对接收到的数据进行分析。

这里主要简单介绍一下本文将要用到的一些重要函数：

最基本的是创建套接字函数socket()，其函数原型为：

```
SOCKET socket( int af, int type, int protocol);
```

与服务器端建立连接的函数是connect()，其函数原型为：

```
int connect( SOCKET s, const struct sockaddr * name, int namelen);
```

用来从一个指定的套接字接收数据的函数recv()，原型为：

```
int recv( SOCKET s, char * buf, int len, int flags);
```

向一个已连接的套接字发送数据的函数send，其原型为：

```
int send( SOCKET s, const char * buf, int len, int flags);
```

关闭套接字函数是closesocket()，其原型为：

```
int closesocket( SOCKET s);
```

还有其他的一些辅助函数来保证网络通讯的顺利进行，在此不一一介绍。

第四章 策略驱动的网络漏洞扫描器的原理及系统设计

经过前面对安全漏洞扫描器原理、功能的深入分析，以及对现有网络漏洞扫描工具的实例分析和一些实际的扫描经验，首先提出了要达到的设计目标和要遵循的详细设计原则，然后根据漏洞特征分析了策略驱动的漏洞扫描原理，并根据该原理，提出了安全漏洞扫描系统的总体设计模型。

4.1 系统设计目标

根据前面对漏洞扫描原理及扫描器性能要求的分析，可知网络漏洞扫描系统是一个涉及知识面广、动态性很强的系统，它应具有以下特性：

1) 可扩展性

可扩展性是保证一个网络漏洞扫描器的扫描质量和长久生命力的关键。首先，漏洞扫描器能完成的扫描内容决定了它的功能和可靠性，扫描内容越多越全面，它对系统的安全评估能力也就越强越准确。再者人们在使用系统和应用软件的过程中，不断地发现各种新的安全漏洞，入侵者往往利用这些新的漏洞进行攻击，如果没有相应的扫描措施，将很难了解并防范这些攻击。然而，人们在设计和实现漏洞扫描器时，不可能把所有的安全漏洞网罗其中，而且也没有能力预见各种新漏洞。因此，为了提供扫描器的实用性，必须使它具有良好的可扩展功能，能够方便地增加各种新的必要的漏洞检查程序。

2) 易于配置

因为网络漏洞扫描系统的目标就是简化网络管理员的工作，能够使非专业人员完成网络安全的测评，所以应该提供方便友好的用户界面，使得系统易于配置和使用。

3) 准确性

通过我们前面对漏洞扫描原理的分析，知道无论哪种漏洞扫描方法都不可能100%的保证扫描结果的准确性。因此，我们只能通过综合各种扫描方法，对漏洞进行多层次的判别，提高扫描的准确性。

4) 漏洞描述应与CVE兼容

由于目前存在多种网络安全漏洞库，他们对于同样的安全脆弱性往往有着不同的命名和描述方法。为避免这种混乱，就是采用统一的CVE标准的安全漏洞命名方法。

随着越来越多的安全组织和安全产品支持CVE编号标准，CVE已成为网络安全漏洞表示的事实上的标准。

4.2 策略驱动的漏洞扫描原理

漏洞有多种属性，描述一个漏洞的完整信息包括漏洞所影响的系统信息、使用端口、补丁信息及原理描述等多种信息，其属性示意图如 4-1 所示：

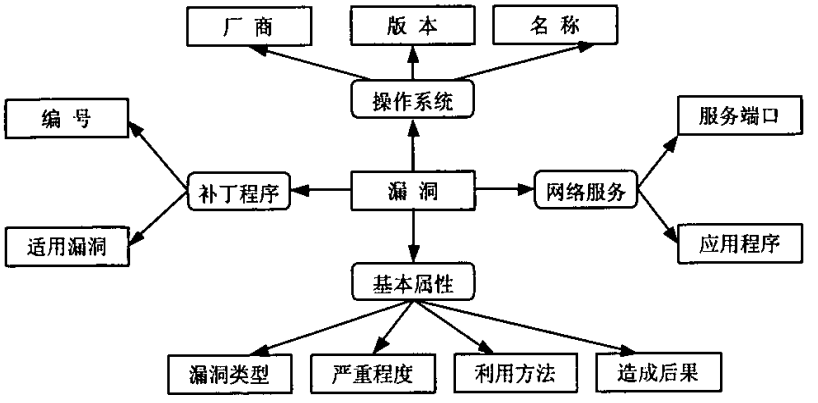

图 4-1 漏洞属性示意图

Fig.4-1 The sketch map of vulnerability attribute

结合漏洞扫描，具体分析漏洞这些属性，可概括如下：

1) 安全漏洞总是与某种版本的操作系统相关。

2) 从漏洞探测的角度看，网络安全漏洞与某种网络服务有关（如标准服务、应用程序等），而网络服务是通过端口进行通信的，所以安全漏洞与开放端口直接关联。

3) 漏洞的原理描述包括漏洞产生的机理、漏洞特征代码、危害性等信息。漏洞探测程序根据漏洞的原理进行编写，可以针对性地探测漏洞。

4) 漏洞补丁程序是漏洞的主要解决方案，用于消除目标系统对应的安全隐患，可以通过目标系统是否打了相应的补丁来间接探测目标系统是否还存在相应漏洞。

以上信息构成了安全漏洞的特征集合，如何利用这些信息探测漏洞或鉴别漏洞是漏洞扫描系统的关键所在。对漏洞扫描技术而言，漏洞的特征信息有主要特征和次要特征的区分，以主要特征探测为主，以次要特征探测为辅，有机地结合，可以合理适宜地探测漏洞。总体而言，相对漏洞扫描来说，漏洞信息的主次关系如下：

第一级：漏洞所影响的端口信息。端口开放是能够探测安全漏洞的前提。

第二级：漏洞所影响的的操作系统信息。正确的操作系统信息是进一步调用相应

操作系统相关的漏洞探测程序的判决条件，有利于提高扫描的准确性和效率。

第三级：漏洞原理描述。漏洞探测程序通常是利用漏洞机理、特征码等编写的，用于扫描或攻击目标系统漏洞。

第四级：漏洞的补丁等其他信息。漏洞补丁探测程序利用补丁特征等编写的。在对该漏洞扫描时，可以利用这些信息从侧面来对目标进行漏洞探测，从多方面判断漏洞。

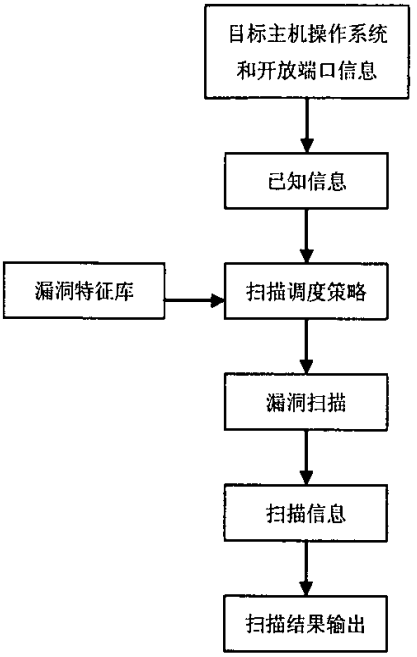

图 4-2 策略驱动的漏洞扫描原理示意图

Fig.4-2 The sketch map of strategy-driven vulnerability scanning

按照漏洞信息级别，根据相关知识逐级判断，多层次地对漏洞特征进行识别，这样可以保证漏洞探测的有效性，提高系统效率，也有利于对目标系统进行全面的安全评价。策略驱动的漏洞扫描方法正是基于上述分析提出的，其基本原理如图 4-2。就是在系统的执行过程中，将端口扫描、操作系统识别与漏洞扫描分开，在进行具体的漏洞扫描前先执行端口扫描、操作系统识别。根据收集的主机开放端口和服务信息对目标主机可能存在的漏洞进行初步的判别，在下一阶段的漏洞扫描时，就可以针对目标操作系统及开放的网络服务调用相应的扫描插件进行扫描，在漏洞扫描中根据已经获得的信息，按照信息的重要性主次关系作用于漏洞库，进行漏洞特征的相关分析，调用插件库中相应的漏洞探测插件，执行漏洞探测。这样相对目前大多数扫描器广泛采

用的依次调用漏洞扫描插件来说，不仅对漏洞进行了多层次的判别，而且可以减少扫描的盲目性，提高漏洞扫描效率。扫描插件的编写可以根据漏洞机理、特征码等编写，通过模拟攻击目标系统来探测漏洞，并同时根据漏洞的补丁信息来从侧面判断漏洞。

4.3 网络漏洞扫描器的总体设计

根据以上分析，我们提出了网络漏洞扫描系统的总体设计模型，如图4-3所示，下面简要说明各部分的功能和设计思想。

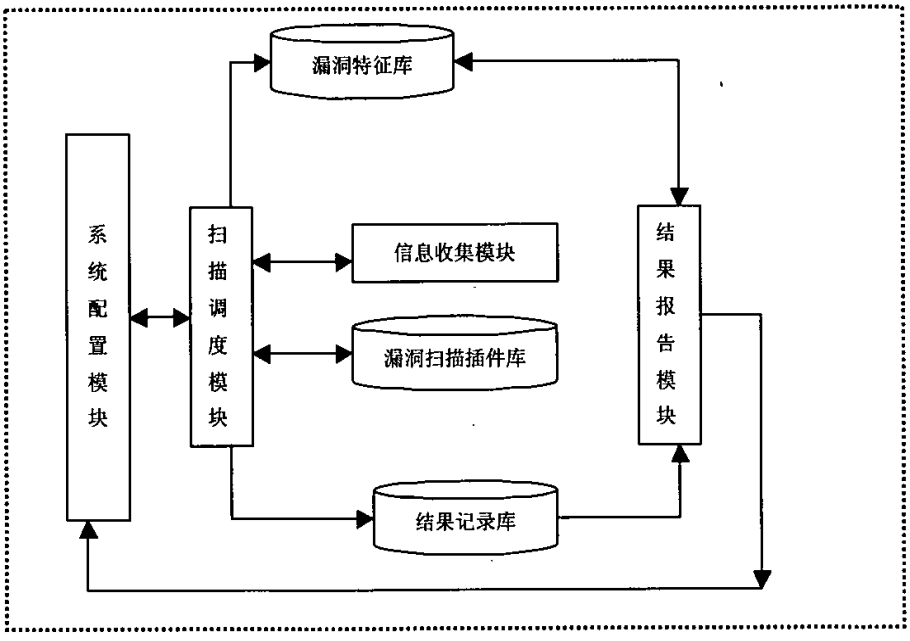

图 4-3 网络漏洞扫描器总体设计模型图

Fig.4-3 The design model of network vulnerability scanner

4.3.1 系统配置模块

系统配置模块为用户提供一个配置和控制扫描全过程的图形化操作界面。扫描系统通过此模块接受任务，控制扫描任务的启动、暂停、终止；配置扫描参数如扫描的目标IP等，获得需要扫描的漏洞模块等信息，以及确定扫描任务结束后的相关输出。

当用户通过系统控制平台发出了扫描命令之后，控制平台即向扫描模块发出相应的扫描请求，扫描模块在接到请求之后立即启动相应的子功能模块，对被扫描主机进

行扫描。在扫描过程中，以图形界面显示扫描进度，其中包括端口扫描进度，漏洞扫描进度，以及正在运行的插件名称等。而且，用户可以选择暂停或终止对某一主机的扫描或者整个扫描。一旦终止对某一主机的扫描，系统会显示到目前为止的该主机的扫描结果。通过对从被扫描主机返回的信息进行分析判断，扫描结果报告模块将扫描结果返回给控制平台，再由控制平台最终呈现给用户。

4.3.2 扫描调度模块

扫描调度模块是系统的主程序，它采用策略驱动的漏洞扫描方法，负责扫描调度执行，即根据信息收集结果，根据调度策略决定调用哪些漏洞扫描插件和插件调用的顺序等等。

扫描调度模块首先调用信息收集模块对目标主机进行初步的信息收集，然后根据信息收集模块提供的目标操作系统、开放端口信息，根据策略驱动机制来负责调度插件的执行，同时在漏洞扫描过程中还起着将扫描结果写入结果记录库及搜索漏洞特征库的作用。

4.3.3 信息收集模块

信息收集模块根据扫描参数对目标主机进行初步的信息收集，其主要是利用第三章介绍的信息收集技术原理，识别远程操作系统及其版本，探测对方的开放端口，为下一步的漏洞检测提供参考。这是因为不同的操作系统的不同服务有不同的系统漏洞，从而可以采取不同的检测措施。

4.3.4 扫描结果报告模块

漏洞扫描结果报告是漏洞扫描器的最后一个重要环节，详实和准确的漏洞报告是评价一个漏洞扫描器的基本标准，这些信息能够让用户全面、深入地了解被扫描目标系统所存在的安全隐患，帮助用户及时准确地了解自己的系统对外提供了哪些服务，而哪些服务存在着潜在的安全风险，如版本过低等，以及时提醒用户关闭对外提供的不必要的服务和端口，或者更换服务对外所显示的服务版本标志等，以免为怀有恶意的用户或黑客提供过多的有用信息。同时根据漏洞扫描器的漏洞描述信息和补救方法来及时补漏和修改必要的设置。该模块根据漏洞特征库和结果记录库来进行，主要分为两部分：一部分为了用户扫描期间查看方便，是在运行界面上显示扫描进度，及本次扫描发现漏洞的具体描述，包括漏洞名称、CVE编号、解决方案、漏洞严重性等级

（分别以3，2，1表示高，中，低）。另一部分是以HTML形式提交给用户的扫描结果报告，可以保存在一个专门的文件夹里，方便日后查看。其主要输出本次扫描发现目标主机的操作系统信息、端口开放信息、发现的漏洞具体信息等。

4.3.5 漏洞扫描插件库

漏洞扫描插件库主要存放的是常见的和最新的检测漏洞和脆弱性的测试程序，实现对目标系统可能存在的安全漏洞和隐患进行检测和分析。对新发现的漏洞及其攻击方法，通过编写相应的漏洞扫描插件，并加上插件服务接口的相应函数调用，加到漏洞扫描插件库中，此后主程序会自动动态加载并调用该插件。

在编写对具体漏洞探测的插件时，应尽量从多方面来判断漏洞，提取多个特征值，利用漏洞原理及补丁程序多方面探测。

4.3.6 结果记录库

该数据库中包含测试的中间结果记录，具体包括被测试的目标系统的IP地址、测试中间结果等。结果分析模块可以利用结果记录库，对系统测试结果进行分析，从而提供方便的结果报告。

4.3.7 漏洞特征库

显然，策略驱动方法的核心是如何根据已知信息进行相关分析，从而决定调用相应的漏洞扫描插件。漏洞特征库除了应包括漏洞的基本属性特征描述外，还应包括漏洞和目标操作系统及开放端口的依存关系。漏洞库与扫描插件库同时扩展，编制某一个扫描插件的同时，同时也应扩充漏洞库的知识。

该模块用来存储当前已知的所有漏洞信息，包括漏洞的序列号、漏洞的名称、漏洞的类别、漏洞的具体描述、漏洞的脆弱性等级、漏洞的修补方法等。为了便于用户根据不同的需求进行检索和查询操作，同时符合业界的命名标准，因此采用了业界公认的CVE列表。漏洞库的定义如表4-1：

4-1 漏洞库的定义

Table 4-1 The definition of vulnerability database

漏洞ID	漏洞名	CVE编号	受影响的系统	依赖端口	漏洞形成原因	漏洞严重程度	修补方案	参考资源	扫描插件名称

4.4 本系统的优点

该模型的优点主要表现在两个方面：

1) 提高扫描的效率

这种策略的最大优势在于缩小了探测的范围，提高了扫描的效率，现有的许多扫描器不管目标是什么系统，开放了什么端口，都会将所有漏洞都检测一遍。这样做虽然保证了漏洞扫描的全面性，但如果目标是Windows平台，将所有检测UNIX系统漏洞的插件都执行一遍，不仅浪费时间，而且增加了不必要的网络流量。而本系统只是有针对性地执行相关的漏洞扫描，能够保证只执行需要执行的扫描插件，从而提高漏洞扫描的效率。

2) 对漏洞进行多层次地探测，提高扫描的准确性

2.3介绍了漏洞扫描的两种方法，并分析了两种方法的局限性，知道无论采用哪种方法都不能保证扫描的绝对准确性，只能采用对漏洞进行多层次的判断，两种方法相结合来提高漏洞扫描的准确性，本模型从漏洞的多个特征来判断漏洞，一定程度上提高了扫描的准确性。

第五章 网络漏洞扫描器的实现及试验

前面分析了策略驱动的漏洞扫描原理以及漏洞扫描器的体系结构及其工作原理，本章将详细介绍主要功能模块的实现。

本文设计的网络漏洞扫描器基于Windows 2000，Visual C++ 6.0开发。可运行于所有Win2000后的Windows平台，具体包括Windows2000系列、WindowsXP系列和Windows2003系列。能探测Windows和UNIX系列等操作系统以及其上运行网络软件的漏洞。

5.1 扫描调度模块的实现

扫描调度模块首先调用信息收集模块，对目标主机进行信息收集，根据信息收集模块收集到的系统开放端口、操作系统信息，根据扫描策略检索漏洞特征库，判断可能存在的漏洞集合，并选择相应的漏洞扫描插件，逐个进行探测，并对返回探测信息进行收集整理；在扫描的过程中，将扫描的各种属性，例如扫描目标设置、扫描中间结果等，保存到结果记录库中；将收集到的漏洞信息传送给下一阶段的扫描结果报告模块，供扫描结果报告模块产生扫描结果报表。

采用策略驱动的漏洞扫描算法如下：

1) 信息收集

调用信息收集模块收集目标主机操作系统类型及开放端口信息。

2) 加载符合条件的插件

根据操作系统及端口开放信息以及系统配置设置的要检测的安全漏洞类型，按照规则从插件库中选取符合条件的扫描插件加载，在上载插件时，运行每个插件的初始化函数初始化插件的基本信息，并将所有插件被上载的地址存入一链表。当上载插件时，从检测代码执行函数入口上载该插件并返回一个句柄，通过这个句柄和入口函数名保存插件被上载的地址。

3) 依次执行加载的扫描插件

4) 将漏洞信息传给扫描结果报告

5) 扫描结束

该模块总的工作流程如图5-1。

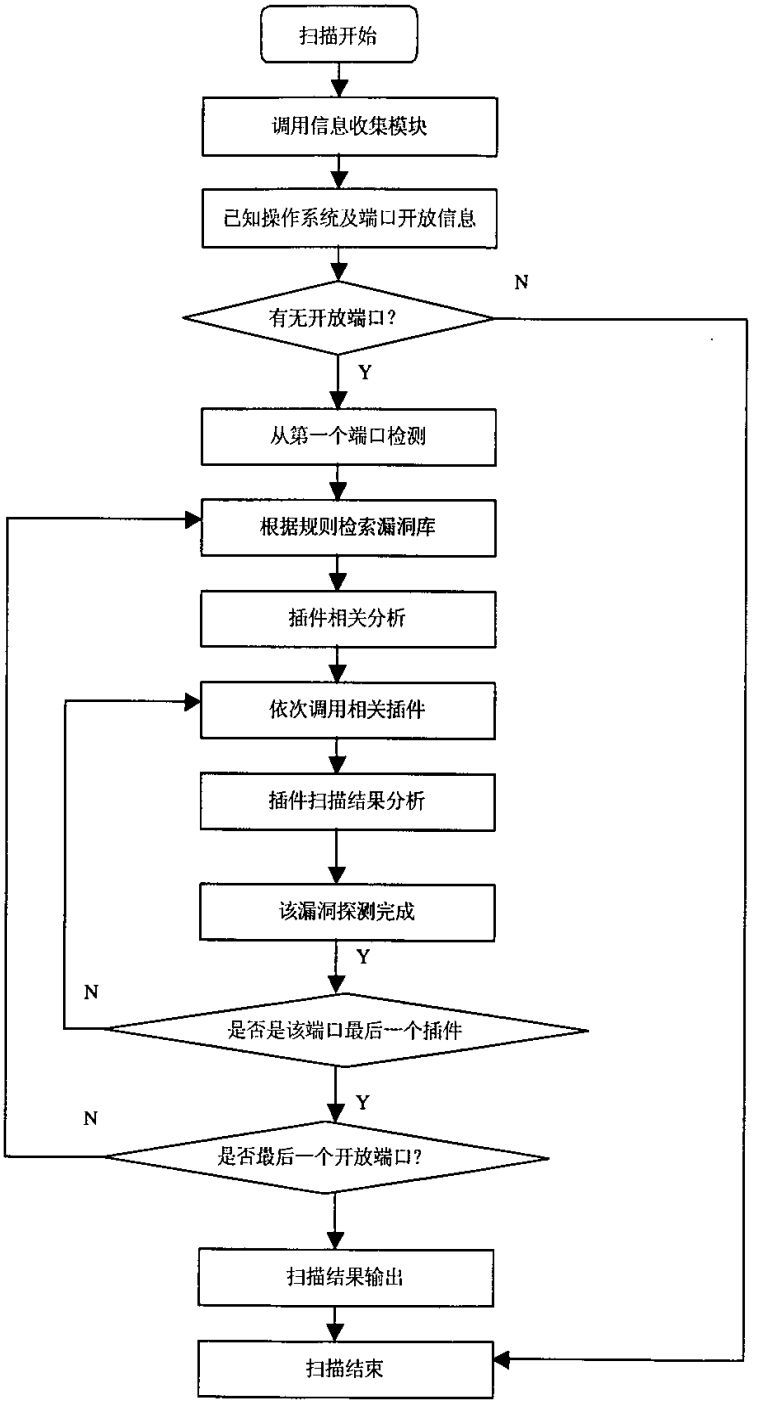

图 5-1 扫描调度模块流程图

Fig.5-1 The flow chart of the scanning scheduling module

漏洞扫描插件在应用的时候被加载，不需要时被释放。所有插件都被一个称为

plugin_link的链表控制，用来处理插件与应用程序交互的问题。plugin_link的数据结构如下：

```
struct   Plugin_Link
        { char FileName[50]; ;    //存放插件名
          int nMark;   //插件是否被放入队列
          struct Plugin_Llink *next
         }plugin_link
```

驱动规则实质是一种基于关系数据库的逻辑驱动机制，按照漏洞信息的主次关系，使用IF...THEN语句，利用已知信息检索漏洞库的相应字段，根据匹配结果，选择对应的扫描插件进行安全探测。其策略原理如图5-2所示：

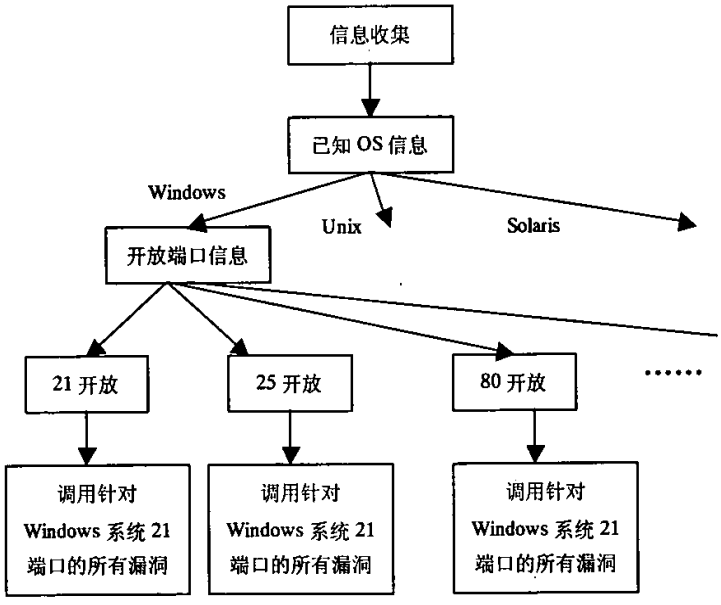

图 5-2 扫描调度策略原理图

Fig.5-2 The principle of the scanning policy

本系统中的扫描插件调度策略的实现依赖于漏洞特征库。只有建立一个完整的漏洞特征库，才有可能从本质上提高漏洞扫描的性能。由于漏洞的数量巨大并且不断有新的漏洞产生，所以漏洞特征库的创建与维护的工作量很大，需要丰富的网络安全方面知识，并且需要不断积累。

5.2 信息收集模块的实现

信息收集模块的目标就是尽可能地探测远程操作系统及其开放的端口等信息，以便于对目标系统进行扫描时的插件选择，缩短扫描时间，同时也减少误报率，

为提高扫描效率，首先用Ping连通测试主机是否在线，对在线的主机才执行进一步的操作系统识别及端口扫描。具体流程见图5-3。

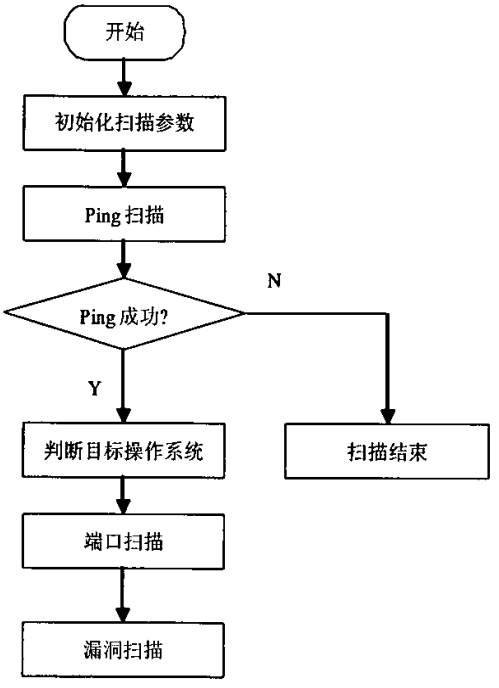

图 5-3 信息收集模块流程图

Fig.5-3 The flow chart of the information gathering model

5.2.1 Ping 扫描模块的实现

该模块主要用来判断目标主机是否在线。本系统主机扫描的实现使用 Ping Sweep 扫描。实现原理：通过向目标主机发送 ICMP Echo Request （type 8）数据包，等待回复的 ICMP Echo Reply 包（type 0）。如果能收到，则表明目标系统可达，否则表明目标系统不可达。

Ping利用的是ICMP协议，因为ICMP报文是直接封装在IP报文的数据区中发送接收

的，必须通过IP层处理它。在实现时，无法通过数据报Socket和数据流Socket来发送和接收ICMP报文，需要使用原始套接字Raw Socket。

Ping 程序的编写步骤概括如下：

1) 使用 socket()创建一个类型为 SOCK_RAW 的原始套接字，同时设定协议 IPPROTO_ICMP；

2) 使用 setsockopt()设置发送和接收超时选项；

3) 自己填充 IP 数据报和 ICMP 数据报并计算校验和，其中 IP 数据报的数据域存放 ICMP 数据报；

4) 调用 sendto()将 ICMP 请求发给远程主机；

5) 调用 recvfrom()接收 ICMP 应答；

6) 分析接收到的数据报是否是 ICMP_ECHOREPLY。

7) 调用closesocket()关闭套接字。

5.2.2 端口扫描程序的实现

本系统提供了两种端口扫描方式：TCP connect()扫描和TCP SYN扫描。

TCP connect()实现最简单，直接使用数据报Socket，通过调用Connect()函数连接目标主机的目标端口，根据函数的返回值来进行判断。如果Connect()调用返回成功标志，说明完成了一次完整的TCP三次握手连接过程，该端口是打开的，否则该端口就是关闭的。读取该端口的响应信息后切断连接，转向下一端口。

此外，本系统还实现了TCP SYN扫描，在实现TCP SYN扫描时，需要使用原始套接字，因为仅原始套接字支持开发者对IP报头的设置，即自己构造IP数据包。同时，原始套接字也支持发送和接收IP数据包。实现时可通过调用setsockopt()函数来设置选项IP_HDRING，从而获取自定义IP报头的权限。具体的实现流程如图5-4所示：

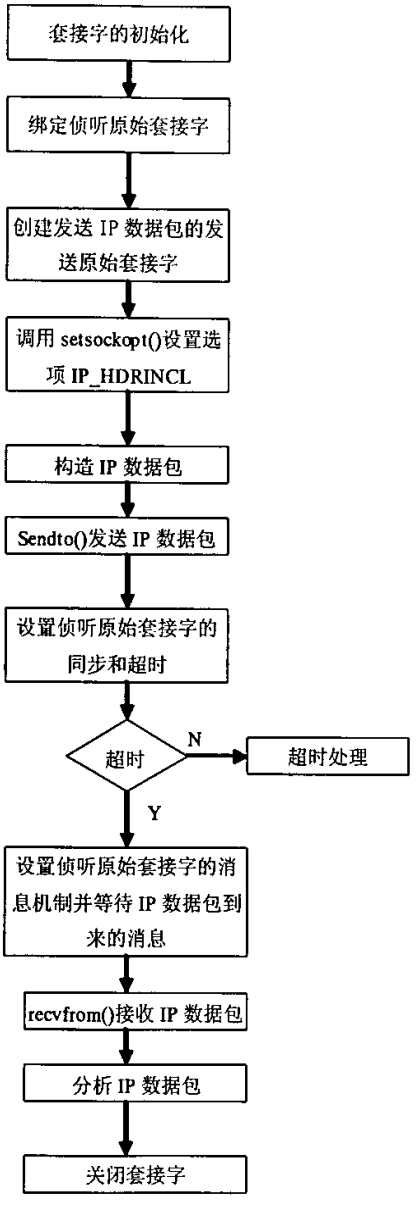

图 5-4 SYN 扫描流程图

Fig.5-4 The flow chart of SYN scanning

5.2.3 操作系统探测的实现

本网络漏洞扫描器中目标主机操作系统的探测实现利用向目标主机发送类似Ping命令的ICMP回显报文请求，根据返回的TTL（位于IP首部中的生存时间字段）值来判断

目标主机操作系统类型，5.2.1已经介绍了Ping的编程实现，本系统中，操作系统的识别方法只是通过分析Ping返回数据包中的TTL值，因为TTL值依赖于包从源地址到目标地址所经过的节点数，TTL的值不是常量。我们在实现时，采用的是表5-1所示特征库：

表 5-1 操作系统—TTL 对应值
Table 5-1 OS type – the corresponding TTL value

TTL值	操作系统
<=32	Windows 9X 系列
32-64	Linux
64-128	Windows2000/NT
128-255	Unix

5.3 漏洞扫描插件库的实现

扫描插件的主要功能是完成对目标主机的安全漏洞检测，不同的漏洞扫描插件对应不同的漏洞。插件被编译成动态链接库文件，存放在扫描插件库中。主扫描程序通过调用插件的方法来执行扫描。仅仅是添加新的插件就可以使软件增加新功能，扫描更多漏洞。在插件编写规范公布的情况下，用户或者第三方公司甚至可以自己编写插件来扩充软件的功能，从而使软件的升级维护都变得相对简单，并具有非常强的扩展性。

软件插件技术是现代软件设计思想的体现，其本质是在不修改程序主体的情况下对软件功能进行加强和扩充。从广义的范围来看，插件主要有以下三种类型：

1) 类似批命令的简单插件。事实上这种插件的自由度非常低。使用这种方法的软件有Win Hacker等，运行这种插件后，会一步步要求用户进行选择/输入，最后根据用户的输入来执行一系列事先定义好的操作。这种插件一般是文本文件，功能比较单一，可扩展性极小。优点是插件做起来非常方便，即使是对程序设计了解不多的人也可以制作，如WinAmp的Skin。

2) 使用一种特殊的脚本语言来实现的插件（俗称为脚本插件）。这其实就是一种更高级的插件技术，用户可以使用专门的脚本语言来扩充软件功能。这种插件比较难写，需要软件开发者自己制作一个程序解释内核。例如Perl语言编写的插件，如Nessus扫描器就是用专门的脚本语言NASL语言来编写漏洞插件。这种方法的优点在于无需使用其它工具来制作插件，软件本身就可以实现。

3) 利用已有的程序开发环境来制作的插件。如采用DLL方法来编写插件，使用这种方法的软件在程序主体中建立了多个自定义的接口，使插件能够自由访问程序中的各种资源。这种插件的优势在于自由度极大，编写较为容易。

在本系统中，采用的是第三种类型的插件。这是狭义范围的插件，也是真正意义上的插件。

插件的编写需考虑以下问题：

1) 主体程序的插件处理机制以及用来进行初始化每个插件的过程，每个插件接口的管理，在插件访问时能够做出的正确反馈。

2) 插件接口函数的定义，以及如何对插件接口函数进行封装。

3) 插件和主程序之间的通信机制。

在本系统中，系统主程序相当于一个“容器”，每个插件封装一个或多个漏洞的检测，只要增加漏洞扫描插件就可以扩展漏洞扫描功能。漏洞扫描插件库主要存放的是对各种漏洞进行扫描的插件，对新发现的漏洞及其攻击方法，通过编写相应的漏洞扫描插件，并加上插件服务接口的相应函数调用，加到漏洞扫描插件库中，从而可以方便地实现对扫描插件库的更新和维护。

5.3.1 漏洞扫描插件的接口

在本系统中，插件开发时利用Microsoft Visual C++编译链接后生成DLL文件，由扫描引擎动态加载。为使扫描引擎能够动态加载编译好的插件，每个插件都有统一的模板形式。模板包括插件所要引用的头文件、插件引用的数据结构的定义、初始化函数和检测代码执行函数。

扫描引擎接到扫描请求后，载入检测插件，对被检测目标进行扫描，最后将检测报告发送给用户。在系统配置模块，用户可以根据自己的要求选择扫描主机范围和需要扫描的漏洞类别，扫描模块依照系统配置模块提供的扫描参数和信息收集模块所收集的目标主机操作系统及开放端口服务信息，运行相应的插件对目标主机进行检测。

在本系统中，插件和主程序要分别实现几个接口函数才能完成插件与主程序之间的交互工作。在主程序部分要实现的函数包括：

1) 显示扫描进度或者结果的函数

GetProgress(char *ip, char *chResult);

2) 将扫描结果以报表的形式输出的函数

GetReport(char *ip,char *chMsg);

在每个插件部分要实现的函数包括：

1) 初始化插件的函数

bool PluginInit (INIT *pParam);

参数 pParam 主要包含从主程序里实现的两个函数的指针，以方便在插件里调用主程序里的函数来向主程序输出信息。

2) 获取插件相关信息的函数

bool GetPluginInfo(RECORD *pParam);

参数 pParam 主要包含插件的基本信息，主要是扫描的漏洞名、简单描述、风险等级以及解决方案等基本信息。

3) 插件探测执行的主体执行函数

bool ExecScan(HOST *pParam);

参数 pParam 包含目标地址、最大的等待时间以及最大线程数目。

其中的两个主要数据结构如下：RECORD描述插件的基本信息，用来与调用模块进行通信，从而将插件的各种信息提供给调用模块，具体描述如下：

```
/*插件基本信息*/
typedef struct _RECORD{
    char        ClassName[50];      //插件所属类别
    char        MemberName[50];     //插件成员名称
    char        FileName[50];       //插件文件名
    char        Risk[10];           //漏洞风险等级
    char        Description[300];   //插件对漏洞的文字描述
    char        Advice[300];        //插件对漏洞提供的一些建议
    char        CVE_id              // 漏洞的CVE_id
    int         nMark;              //标识该插件是否被默认选中
        } RECORD;
```

另一个数据结构HOST是插件的运行参数，将调用模块相应的调用信息传递给插件，从而使扫描插件可依据不同的参数来完成相应功能，并将运行的返回信息存入此数据结构。

HOST结构指定了插件运行的参数，包括目标主机的IP地址、最大的等待时间以及最大线程数目。

```
struct _HOST{
```

```
char ip[MAXIP];   //目标主机的IP地址
int threadnum;      // 最大线程数目
int timeout;        //超时控制变量
            }HOST;
```

5.3.2 漏洞扫描插件的具体编写

1) 网络漏洞扫描实现原理分析

网络漏洞扫描的终极目的是找到系统所暴露在网络上的薄弱环节，即从一个入侵者的角度来看，目标系统可被利用的漏洞。

漏洞的探测比较复杂，也没有一个统一的方法来探测，只有单个分析各个漏洞来找出探测方法。根据漏洞的情况，来构造对应的数据包，分析从目标主机返回的数据包。

漏洞探测的流程图如图5-5所示：要检测漏洞，需要构造好适当的数据包，向目标主机发送后，在目标主机修补漏洞的前后会返回不同的数据包，返回的数据包是判断漏洞存在的依据。因此，对于漏洞扫描，本质上主要需要实现三个部分：发送数据包机制、接收数据包机制、漏洞特征码分析和抽象。

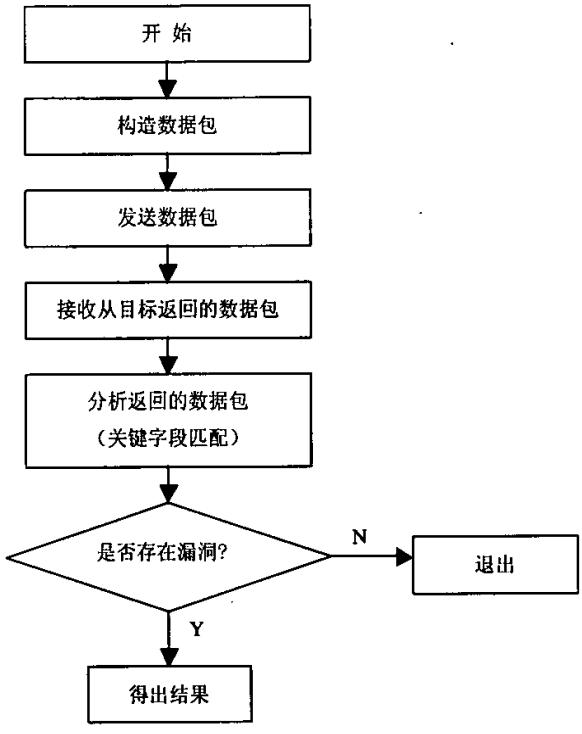

图 5-5 漏洞探测流程图

Fig.5-5 Vulnerabilities detection flow chart

(1) 发送数据包机制

前文已经提到漏洞扫描是一种主动探测行为，需要根据漏洞的特征码来构造数据包并发送到目标主机。发送数据包机制的实现主要有两种方法：其一是建立正常的TCP/IP连接，构造数据段的内容，另外一种要求能够构造一些数据包的特殊字段，例如IP的分片标志和偏移量以及TCP首部的一些特殊标志位等等。

(2) 接收数据包机制

对于网络远程扫描，对方主机所回应的数据包是评估漏洞的原始资料，快速准确地接收数据包是漏洞扫描的基本保证。除通过正常的TCP\IP协议栈进行处理外，还需要接收原始数据包。例如Windows系统下，要通过原始套接字Raw Socket接收这种原始数据包，其原因是对不同的漏洞所要分析的数据包的层次是不相同的，如查看端口号或者查看TCP返回的标志位等。

漏洞扫描的发送数据包机制和接收数据包机制与端口扫描方法的发送接收机制类似，根据不同情况分别使用Winsock API提供的三种套接口来实现，这里就不再赘述。

(3) 漏洞特征码分析抽象

漏洞扫描的关键是对漏洞特征码的分析与抽象。发送机制需要根据特征码来发送探测数据包，接收数据包后也需要根据返回的特征码来进行判断。所以，特征码应该能够保证探测数据包的有效性和接收到应答信息后判断的准确性。

我们利用抓包工具Iris（Eeye公司发布）及Winsock Expert通过对一些扫描工具的扫描过程进行抓包，分析其发送特征码，并通过分析其发现漏洞与未发现漏洞时接受到的数据包的差异来分析其接收特征码。通过大量的实验，我们认为：大多数扫描器通常将特征码最终归结为数据包中某一字段的数值或者字符串，但这些扫描工具或多或少都存在一定的误报。要提高扫描的准确性，特征码决不应该是一个单一的值，而应尽量取几个字段与其他判断条件相结合的值的集合，从多方面来提取特征码，多层次来判断漏洞，尽可能地减少误报。

由于漏洞种类的千差万别，漏洞的出现也层出不穷，这使提取漏洞的特征码值的工作非常困难。所以说，实现漏洞扫描的关键其实是一个分析漏洞的过程，只有单个分析漏洞来找出其探测方法，根据各漏洞的具体情况来构造相应的数据包，发送并分析从目标主机返回的数据包。为此我们需要分析同一类型系统环境在修补漏洞前后从目标返回的数据包，找出其中的不同来作为判断是否存在漏洞的特征码，在提取特征码时应尽量从多方面考虑，综合多个特征值，所以，漏洞扫描的关键就归结到如何找到这个特殊的探测包。

漏洞的种类很多，据CVE的统计资料现在共有三千多种，其中以CGI漏洞最多，而缓冲区溢出漏洞的出现频率最高，下面是几种比较典型的漏洞扫描的实现过程，每一类中又有很多种漏洞，但实现原理基本相同，只是代码的编写有些差别。下面以几个实例来介绍我们分析得出的几种具体的漏洞扫描插件编写中特征码的提取及扫描具体实现。

2) 拒绝服务攻击漏洞扫描实现

在检测这类漏洞的时候多是采用模拟攻击方式来评估目标系统对抗DoS的能力。即首先模拟黑客对目标进行拒绝服务攻击致使目标瘫痪，等待几秒后，再次试图与服务器建立连接，如果失败则认为服务器存在拒绝服务漏洞。

这里以检测Microsoft SQL Server是否有DDOS漏洞的插件的主要函数代码为例，说明这种DOS扫描插件的编写方法。

漏洞描述：Microsoft SQL Server 7.0接收连续0信号TCP包时被非法关闭。

CVE ID：CVE-1999-0999

漏洞影响版本：Microsoft SQL Server 7.0；Microsoft SQL Server 7.0 SP1

漏洞产生后果：可造成目标主机拒绝服务。

扫描原理：首先与SQL Server建立TCP连接并判断连接是否成功，然后向服务器发送连续字节的0并关闭连接，等待几秒后，再次试图与服务器建立连接，如果失败则认为服务器已经发生拒绝服务故障。

主要实现：

```
extern "C" _declspec(dllexport)_stdcall boolean   Plugin_run(struct HOST *host)
{
struct sockaddr_in sad;//存放主机地址的结构
struct hostent *ptrh;//指向主机入口的指针
SOCKET sd;
char buf[6];
memset(buf,0,6);
memset((char*)&sad,0,sizeof(sad));//清空sad
ptrh=NULL;
sad.sin_port = htons(1433);        //SQL通信端口号为1433
sad.sin_addr.s_addr=inet_addr(host->ip);
sad.sin_family=AF_INET;
sd=socket(AF_INET,SOCK_STREAM,0); //TCP连接
if(sd<0) { return FALSE;}
if(connect(sd,(struct sockaddr *)&sad,sizeof(sad))<0)
  {
  closesocket(sd);
  return FALSE;
  }
buf[0]=0x00; buf[1]=0x00;buf[2]=0x00;
buf[3]=Ox00;buf[4]=Ox00;buf[5]=0x00;
send(sd,buf,sizeof(buf),0);
closesocket(sd);
Sleep(2000);//等待
sd = socket(AF_INET,SOCK_STREAM,0); //再次建立连接
if(sd<0)
```

```
    {
    return FALSE;
    }
  if(connect(sd,(struct sockaddr *)&sad,sizeof(sad))<0)
    {
    closesocket(sd);
    return TRUE; //连接失败，说明发生了拒绝服务
    }
  closesocket(sd);
  return FALSE;
  }
```

此外，为保证扫描准确性，扫描本漏洞同时采取另一种方法从侧面来探测：如果目标开放139或445端口，通过这两个端口建立SMB（Server Message Block 服务器信息块）会话连接，在注册表中查询是否存在关键的KEY值：SOFTWARE\Microsoft\Windows\NT\CrrentVersion\HotFix\Q248749”。如果存在，则认为该漏洞已经被打了补丁；不存在，则判定对方存在该漏洞。

以上这个实例代表了拒绝服务攻击一类的漏洞，对该类漏洞的探测相对比较困难，因为大多数拒绝服务攻击都是在极短的时间内发送极大的数据量，但是扫描器本身不能作为攻击的工具，所以有两种方法进行探测：第一种就是模拟攻击，即仍然按照漏洞本身的特征构造“非法”的数据包，但是发送量较小，来探测对方的反应。该方法直接切入漏洞本身，准确性相对较高，但是具有潜在的攻击性。第二种就是从侧面来推测，即通过其他的方式进行漏洞的探测，而避开发送可能存在危险的数据包，例如通过判断其是否安装了补丁程序来进行探测，危险性就会大大下降。该方法的缺点即为判断的稳定性比较难以保证，而通过采取两种方法相结合，就可以在一定程度上减小扫描误报率。

3) CGI漏洞扫描实现

通用网关接口 CGI（Common Gateway Interface）是一种规范，允许 WEB 服务器执行其它程序并将输出结果返回给浏览器。

WEB 服务器和客户端的浏览器是通过 HTTP 建立连接和完成超文本的正确传输。由于服务器并不能记住浏览器每次的访问，HTTP1.0 协议被称为无关状态的，每个 HTTP1.0 中的 GET 或 POST 必须携带全部的信息以服务于这些请求。

一个完整的 HTTP 响应包含 HTTP 版本和状态码、通用头标信息、实体头标和实体[30]。

例如：HTTP/1.0 200 OK　　//HTTP 版本，状态码

Server: NCSA/1.0a6　　//通用头标信息

Content_type: text/plain　　//实体头标

Text goes here…　　//实体

一些常见的 HTTP 响应状态码如表 5-2 所示。

表 5-2 常见的 HTTP 响应状态码

Table 5-2 the common HTTP response code

代码	结果	描述
200	OK	请求执行正常
202	Accepted	请求已接纳但仍在处理
301	Moved	文档已被移到一个新的位置
302	Found	文档在服务器上,但在不同位置上
400	Bad Request	请求语法错误
401	Unauthorized	服务器对文档有限制
403	Forbidden	由于访问权和其它原因请求被禁止
404	Not Found	请求不能找到匹配项
500	Internal Error	服务器执行请求的不可预期失败
502	Service is Overload	服务器不能再处理另外的要求

对于WEB服务器，在远程用户权限允许的情况下，如果用户访问的是可执行程序和CGI脚本，那么程序或CGI脚本将被执行，服务器输出结果。如果用户访问的是文本文件，那么服务器将输出该文本文件的内容。

由于CGI程序开发者的疏忽，许多CGI程序存在有各种危险程度的漏洞。这些漏洞有的允许远程攻击者在WEB服务器上执行任意的命令，可能对服务器造成破坏；有的漏洞能给攻击者提供WEB服务上目录或者文件列表，以及文件的内容，为攻击者提供攻击的信息等，对WEB服务器的安全造成威胁[31]。

对于该类漏洞的扫描，一般过程如图5-6所示：

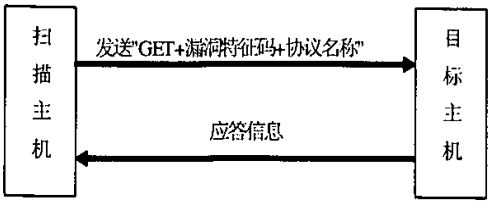

图 5-6 CGI 漏洞扫描

Fig.5-6 CGI vulnerability scanning

通常来说，如何提取发送特征码相对比较容易，通常黑客的入侵也是首先发送特殊的请求，在发送含特征码的检测包时，与目标主机进行连接时有二种方法：一种是使用TCP/IP中的connect()与目标主机的80端口进行连接，发送“GET+漏洞特征码+协议名称”等待回应包；另一种使用GetHttpConnection()与目标主机建立HTTP连接，使用OpenRequest(_T ("GET", 特征码))发送特征码。

但是如何提取接收特征码，我们通过对其它工具的抓包分析，发现大多数工具是利用HTTP响应状态码来进行漏洞检测，比较应答包中是否含“200”或“202”进行判断，这种方法在大多数情况下是比较有效的，但我们在实验中发现：虽然目标存在CGI漏洞，但服务器返回的信息中并不包含“200”或“202”，因此虽然目标存在该漏洞，但却没有被检测到。

我们采用的方法是：具体分析每个CGI漏洞返回信息的特征，采用状态码与返回信息相结合的接收特征码。例如对于那些能给攻击者提供WEB服务上目录的CGI漏洞，因为漏洞造成的后果会导致泄露目标主机的目录信息，其中包含“<DIR>”的字符串，所以可同时将返回数据包中是否存在“200”或“202”及“<DIR>”作为漏洞判别的依据。

我们以 IIS directory traversal漏洞为例来说明我们对此类漏洞扫描方法的具体实现。

漏洞描述：通过向目标IIS服务器发送请求时替换UNICODE编码可以允许非法执行恶意的命令。

CVE ID：CAN-2000-0884

漏洞影响版本：Microsoft IIS 5.0-Microsoft Windows NT 2000

Microsoft IIS 4.0-Microsoft Windows NT 4.0

Microsoft BackOffice 4.5-Microsoft Windows NT 4.0

Microsoft BackOffice 4.0-Microsoft Windows NT 4.0

漏洞形成原因：IIS4.0和IIS5.0在Unicode字符解码的实现中存在一个安全漏洞，导致用户可以远程通过IIS执行任意命令。当IIS打开文件时，如果该文件名包含unicode字符，它会对其进行解码，如果用户提供一些特殊的编码可导致IIS错误的打开或者执行某些web根目录以外的文件。

漏洞产生后果：利用此类漏洞绕过IIS的路径检查，去执行或者打开任意文件。

(1) 如果系统包含某个可执行目录就可能执行任意系统命令；

(2) 可能查看系统文件的内容。

扫描原理：与目标服务器端口建立TCP连接后一次发送包含特殊代码的数据包：

"GET/scripts/..%c0%af..%c0%af..%c0%af..%c0%af../winnt/system32/cmd.exe?/c+dir+c\\HTTP/1. 0";

"GET/msadc/..%c0%af..%c0%af..%c0%af..%c0%af../winnt/system32/cmd.exe?/c+dir+c\\HTTP/1. 0";

"GET/_mem_bin/..%c0%af..%c0%af..%c0%af..%c0%af../winnt/system32/cmd.exe?/c+dir+c\\HTTP/1. 0";

"GET/_vti_bin/..%c0%af..%c0%af..%c0%af..%c0%af../winnt/system32/cmd.exe?/c+dir+c\\HTTP/1. 0";

接着接收目标主机发送的数据包，进行字符串匹配，查找数据包内容是否包含“<DIR>”、“200”或“202”的字符串，如果匹配，则判定对方存在该漏洞，否则不存在。

主要实现代码如下：

(1) 创建套接字并初始化套接字地址结构

```
sad. sin_port=htons(80);//WWW端口号
sad.sin_addr.s_addr=inet_addr(host->ip);
sad. sin_family=AF_INET;
soc=socket(AF_INET,SOCK_STREAM, 0);
```

(2) 向目标主机发送连接请求

```
connect (soc,(struct sockaddr*)&sad, sizeof (sad))
```

(3) 发送GET请求：“GET+漏洞特征码+协议名称”

```
send(soc, cgiiisname[i],sizeof(cgiiisname[i]), 0)
```

(4) 接收目标返回数据，并查找数据包内容是否包含“200”或“202”或“<DIR>”的字符串，如果匹配，则判定对方存在该漏洞，否则不存在。

```
recv(soc, rbuff, sizeof(rbuff), 0) ;
```

```
if(strstr(rbuff, "<DIR>")!=NULL)
    {return TRUE;}
else if((strstr(rbuff, "200")!=NULL)||(strstr(rbuff, "202")!=NULL))
    {return TRUE;}
     else return FALSE;
```

4) 远程缓冲区溢出漏洞扫描插件编写

通常可采用模拟攻击的方式进行扫描，可使用现有的利用代码（exploit）来对目标进行测试，如果能成功利用或者使目标系统崩溃，则存在漏洞。但这种方式通常会给目标造成一定的破坏，最好的方法是通过抓取操作系统在修补漏洞和没做修补漏洞时从目标返回的数据包，分析其中的不同来判断是否存在漏洞。

以下以Microsoft Windows LSA（Local Security Authority 本地安全授权服务）远程缓冲区溢出漏洞为例来说明此类漏洞的扫描实现：

漏洞描述：Microsoft Windows LSA （lsasrv.d11）是Windows系统的一个关键服务。LSASS DCE/RPC末端导出的Microsoft活动目录服务存在一个缓冲区溢出漏洞，远程攻击者可以利用这个漏洞以SYSTEM权限在系统上执行任意指令。

漏洞形成原因及危害程度：一些活动目录服务在Windows目录中的“debug”子目录里生成调试日志文件，在lsasrv.dll实现的记录函数写信息条目到日志文件中，其中的vsprintf()函数用于建立日志条目。由于对提供给这个函数的字符串参数缺少正确的边界缓冲区检查，超长字符串可导致缓冲区溢出。部分RPC函数接收此字符串作为参数并尝试写调试日志文件，利用这些RPC函数，攻击者可以SYSTEM 权限在系统上执行任意指令。

漏洞影响系统：Microsoft Windows 2000 sp2, sp3, sp4;
Microsoft Windows XP sp1;
Microsoft Windows Server 2003

CVE ID：CAN-2003-0533

漏洞修补：及时下载Microsoft提供的补丁。

主要实现代码如下：

```
char lsassrequest1[]=
"\x00\x00\x00\x57\xff\x53\x4d\x42\x72\x00\x00\x00\x00\x08\x01\xc8"
"\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x66\x7a"
"\x6b\x00\x00\x00\x00\x34\x00\x02\x4c\x41\x4e\x4d\x41\x4e\x31\x2e"
```

```
"\x30\x00\x02\x57\x69\x6e\x64\x6f\x77\x73\x20\x66\x6f\x72\x20\x57"
"\x6f\x72\x6b\x67\x72\x6f\x75\x70\x73\x20\x33\x2e\x31\x61\x00\x02"
"\x4e\x54\x20\x4c\x4d\x20\x30\x2e\x31\x32\x00";
char lsassrequest2[]=
"\x00\x00\x00\x9e\xff\x53\x4d\x42\x73\x00\x00\x00\x00\x08\x01\xc8"
"\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\xff\xfe"
"\x00\x00\x00\x00\x0c\xff\x00\x00\x00\x01\x40\x02\x00\x01\x00\x00"
"\x00\x00\x00\x44\x00\x00\x00\x00\x00\x5c\x00\x00\x80\x63\x00\x60"
"\x61\x06\x81\x06\x2b\x06\x01\x05\x05\x02\xa0\x56\x30\x54\xa0\x1a"
"\x30\x18\x06\x0a\x2b\x06\x01\x03\x86\xaa\x95\xf2\x33\x06\x06\x0a"
"\x2b\x06\x01\x04\x01\x82\x37\x02\x02\x0a\xa2\x36\x04\x34\x4e\x54"
"\x4c\x4d\x53\x53\x50\x00\x01\x00\x00\x00\x97\x82\x08\xe0\x00\x20"
"\x20\x20\x20\x00\x20\x20\x00\x00\x20\x20\x20\x20\x20\x20\x20\x20"
"\x20\x20\x00\x00\x00\x00\x20\x20\x00\x00\x20\x20\x00\x20\x20\x20"
"\x20\x00";
```

(1) 创建套接字并初始化套接字地址结构

```
server.sin_family=AF_INET;
server.sin_addr.s_addr=htonl(host->ip);
server.sin_port=htons(445);
s=socket(AF_INET,SOCK_STREAM,0);
```

(2) 向目标主机发送连接请求

```
connect(s,(struct sockaddr *)&server,sizeof(server))
```

(3) 发送探测特征码到目标主机，接收并判断目标主机返回的数据包

```
if(send(s, lsassrequest1, sizeof(lsassrequest1)-1, 0))
 {
  ul[0] = 1;
  ul[1] = s;
  l = select (0, (fd_set *)&ul, NULL, NULL, &timeout);
  if(l == 1)
   {
   l = recv (s, recvbuf, sizeof(recvbuf), 0);
```

```
if (l >= 0)
   {
   if(recvbuf[9] == '\x00' && recvbuf[10] == '\x00' &&
    recvbuf[11] == '\x00' && recvbuf[12] == '\x00')
    {
    if(send(s, lsassrequest2, sizeof(lsassrequest2)-1, 0))
     {
     l = select (0, (fd_set*)&ul, NULL, NULL, &timeout);
     if(l == 1)
      {
       memset(recvbuf,0,sizeof(recvbuf));
       l = recv (s, recvbuf, sizeof(recvbuf), 0);
       if(l >= 0)
        {
        if(recvbuf[9] == '\x16' && recvbuf[10] == '\x00' && recvbuf[11] ==
        '\x00' && recvbuf[12] == '\xc0')   //有漏洞的系统返回数据包特征
          { return TRUE; }
        else if(recvbuf[9] == '\x0d' && recvbuf[10] == '\x00'
          && recvbuf[11] == '\x00' && recvbuf[12] == '\xc0')
                    //打过补丁的系统返回的数据包
          { return FALSE;   }
       }
      }
     }
   }
 }
 closesocket(s);
}
```

此外，为减少扫描误报率，扫描本漏洞同时采取也采取了另一种方法从侧面来探测：由于此漏洞依赖139、445端口，我们通过这两个端口与目标主机建立SMB（Server

Message Block 服务器信息块）会话连接，在注册表中查询是否存在关键的KEY值：SOFTWARE\Microsoft\Windows\NT\CrrentVersion\HotFix\KB835732”。如果存在，则认为该漏洞已经被打了补丁；不存在，则判定对方存在该漏洞。

5) 简单口令探测

此类漏洞检测主要是先分析出登陆过程中的数据包的构成，然后使用字典中的用户名和口令来模拟构造用户登陆直到登陆成功或者遍历完字典中的单词为止。漏洞特征码的提取主要是通过分析客户端登陆成功及失败时服务器返回信息的特征。

(1) POP弱口令漏洞扫描

POP3协议（邮局协议）是一种允许用户从邮件服务器收发邮件的协议，它用用户名和口令来控制邮件的读取。对于POP3弱口令漏洞扫描前先建立用户标识文档和口令文档，文档内存有一些常见的众多用户惯用的用户标识和口令，两个文档均可不断更新。扫描时，先与目标端口110连接，确认POP3协议当前处于认证状态。步骤一：发送"Username\r\n”给目标主机。步骤二：判断应答信息，如果信息中包含-ERR（表示连接失败或使用了错误的命令），则用户标识错误。如果信息中包含有+OK（表示连接成功或服务器接受客户端计算机所使用的命令及语法），则用户标识正确。步骤三：发送“Password\r\n”给目标主机。步骤四：判断应答信息，如果信息中包含-ERR，则口令错误；如果信息中包含有+OK，则口令正确。

通过上述步骤的不断重复，也许能够找到弱用户名和口令。

(2) FTP弱口令漏洞扫描

FTP协议即远程文件传输协议，是一个用于网络上系统之间文件传送的协议。用户要连接FTP服务器，必须要有FTP服务器授权的用户标识和口令。FTP弱口令扫描过程如下：

步骤一：建立SOCKET连接，确定目标主机有没有开放FTP服务。

步骤二：测试FTP是否允许匿名登陆，发送信息"USER anonymous\r\n"给目标主机。

步骤三：判断应答信息首字符是否为331，331代表“用户名正确，需要口令”，如果是，则可以匿名登陆。

步骤四：测试能否找到弱口令。扫描前先建立用户标识文档和口令文档。发送“USER User_Name\r\n”给目标主机。

步骤五：判断应答信息首字符是否为331，如果是，则用户名正确。

步骤六：发送“PASS Pass_ID\r\n”给目标主机。

步骤七：判断应答信息首字符是否为230，230代表用户登录。如果是，则找到一组弱用户标识和口令。

从以上示例可以看出，实现漏洞扫描需要非常丰富的网络知识，最重要的是必须掌握每种类型的漏洞的产生原理，从原理上去设计实现漏洞扫描，知道了漏洞的产生原理之后要探测该漏洞则不难了。要实现漏洞扫描还需要了解黑客攻击的手法，必要时可以加以借鉴。从各种角度去综合考虑才能提高漏洞扫描的准确性和有效性。

5.4 系统测试

为了测试本漏洞扫描器的有效性及扫描效率，对其进行了扫描测试，并与 Xscan V2.3 做了对比。比较时，两个扫描器采用相同的配置：漏洞扫描模块选取的是两个扫描器都实现了的 Smtp，ftp，rpc，sql，smtp，cgi，iis，finger 扫描插件，分别见图 5-7，5-8。而且扫描的端口选取的都是 1-139，1080，1433，2049，3128，3389，5000，7000-7010，8010，8080，8181 这几个端口，对 ftp 弱口令猜解使用的是相同的口令、密码字典，对 Xscan 的 cgi，iis 扫描设置中的特征码字典 cgi.lst 也设置了与本扫描器相同的特征码字典。

扫描对象具体情况：局域网主机，IP地址为219.226.86.139，操作系统是Windows 2000 server，在系统安装后，没有安装任何其他系统补丁，同时开启IIS服务。

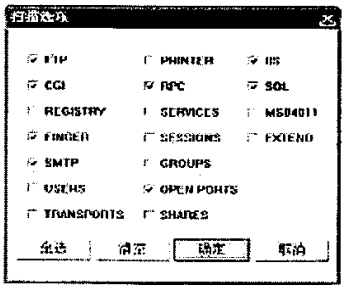

图 5-7 本漏洞扫描器的扫描配置
Fig.5-7 The scanning configuration of our scanner

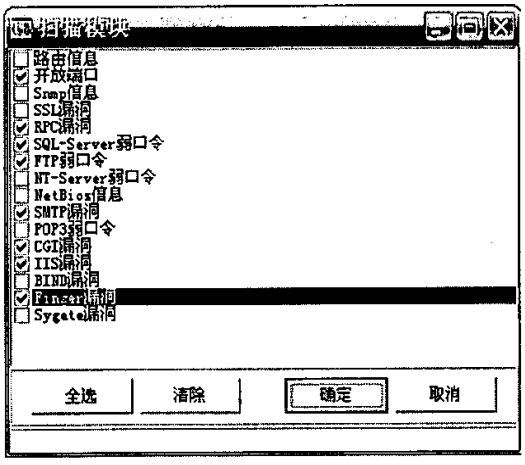

图 5-8 Xscan 扫描模块选择
Fig.5-8 The selected scanning module of Xscan

扫描结果：使用本扫描器扫描，目标主机操作系统类型在扫描中被识别为“Windows 2000/NT”，符合实际情况。发现主机开启了21，25，80，135，139，443，445，1025端口，在扫描中扫描出CGI漏洞与实际情况完全吻合。扫描运行界面如图5-9所示，扫描运行界面包括扫描进度显示区和发现漏洞具体描述区，扫描进度显示区显示扫描进度，发现漏洞具体描述区给出了本次扫描所发现漏洞的具体描述，包括漏洞名称、CVE编号、解决方案、漏洞严重性等级（分别以3，2，1表示高，中，低）。此外，扫描结束后还提供了HTML扫描结果报告，统一保存在log文件夹里，方便我们日后查看。我们用浏览器打开查看结果如图5-10所示。

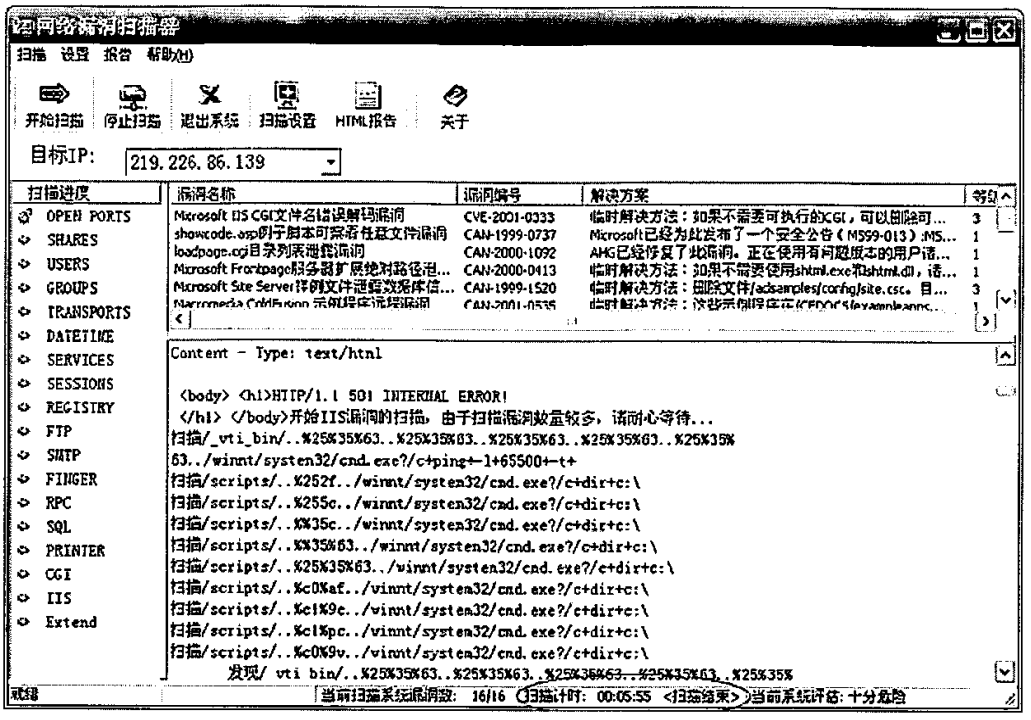

图 5-9 扫描运行界面图

Fig.5-9 The interface of the scanner

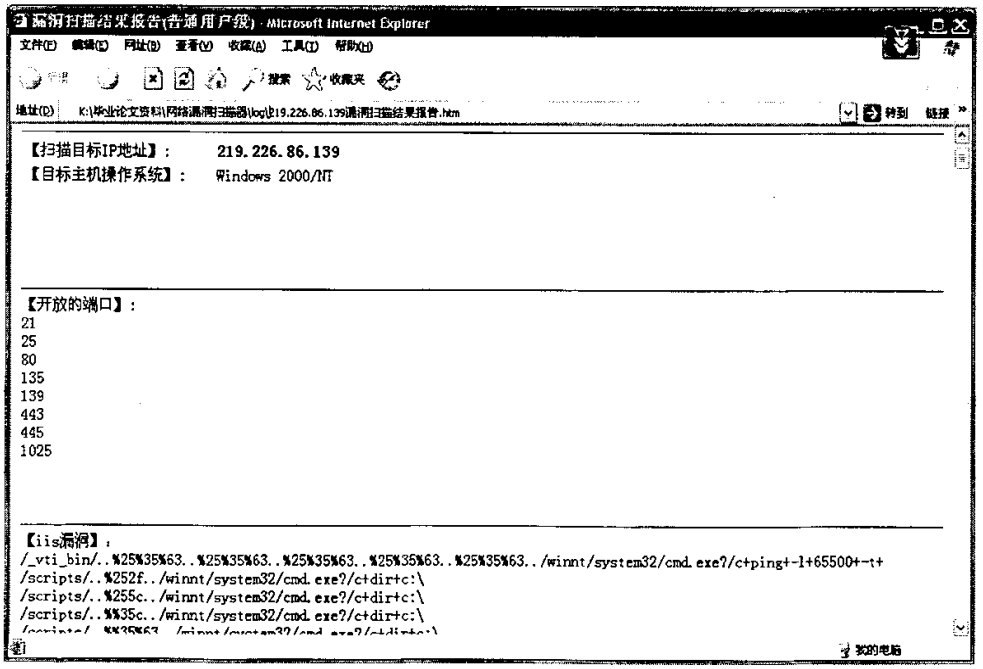

图 5-10 扫描结果报告

Fig.5-10 The scanning report

另外，我们使用 Xscan 对同一台主机进行扫描，采用相同的配置，扫描结果和我们的扫描器结果相同。

我们的扫描器在运行界面上给出扫描时间见图 5-9。通过比较各自的扫描时间，来比较两个扫描器的扫描效率。Xscan 在扫描结束后输出的报告上给出了扫描起始和结束时间如图 5-11。两个图给出的扫描时间是我们针对目标地址 219.226.86.139 的扫描结果，从图中可以看出，Xscan 的扫描时间是 9:01:39 至 9:07:41，共计 6 分 02 秒，本扫描器的扫描时间是 5 分 55 秒。另外扫描时间除了和扫描器本身有关，也和各扫描器运行时的网速有关，为了更好的比较，我们一共做了 5 次试验，具体情况见表 5-3。

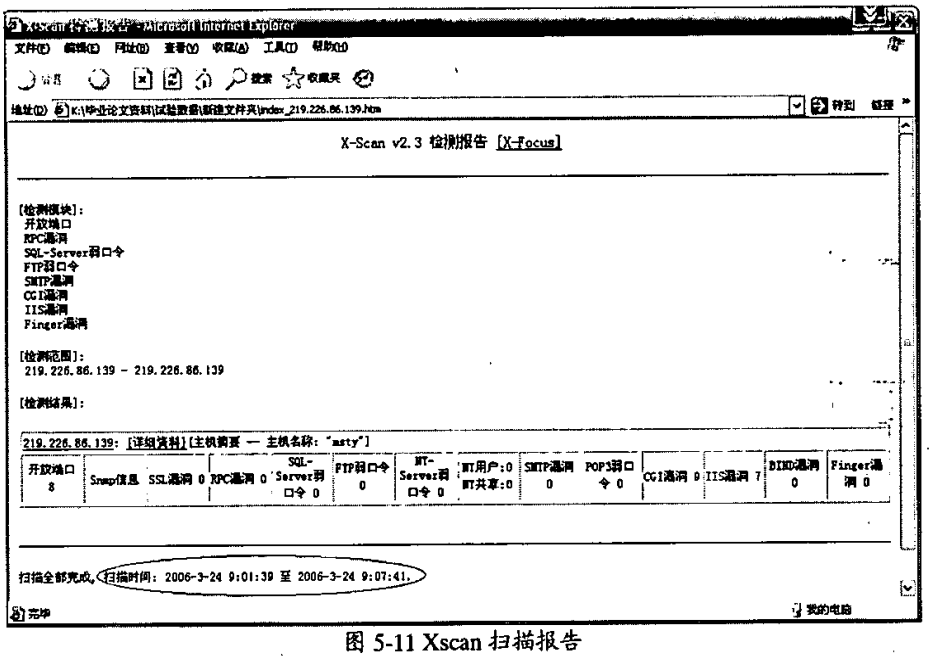

图 5-11 Xscan 扫描报告

Fig.5-11 The scanning report of Xscan

表 5-3 本扫描器与 Xscan 扫描时间对比图

Table 5-3 The contrast of scanning time between Xscan and our scanner

次数	目标主机	开放端口	Xscan 扫描时间	本扫描器 扫描时间
1	219.226.86.139	21，25，80，135，139，443，445，1025	6 分 02 秒	5 分 55 秒
2	219.226.86.136	135，139，445	1 分 52 秒	1 分 38 秒
3	219.226.86.142	135，139，445	2 分 03 秒	1 分 48 秒
4	219.226.86.135	135，139，445	1 分 49 秒	1 分 33 秒
5	219.226.86.129	23，80，25	5 分 10 秒	5 分 03 秒

从表 5-3 可以看出，我们的扫描器由于采用了策略驱动的漏洞扫描插件调度方法，通过有针对性地调用相关的漏洞扫描插件，能够保证只执行需要执行的扫描插件，在一定程度上提高了扫描效率。而且可以看出，因为扫描器根据目标主机开放端口信息调用漏洞扫描插件，在目标主机开放端口较少时，由于许多无关扫描插件会被忽略，这种优势较为明显。

第六章 结束语

本文在深入分析了漏洞的定义、产生原因、特征、分类的基础上，论述了漏洞和扫描的关系。此外，分析了网络漏洞扫描器实现中的一些关键技术：端口扫描、操作系统识别、漏洞扫描和实现这些技术要涉及的网络编程规范 Winsock。

其次，本文结合漏洞扫描，具体分析了漏洞的各个属性，讨论了应该如何利用漏洞的多个特征来综合多层次判断漏洞。并在此基础上，分析了策略驱动的漏洞扫描原理，给出了一个采用策略驱动的安全漏洞扫描系统的总体设计方案，并介绍了该系统的系统模块及各部分的功能实现。该系统基于信息收集结果调度漏洞扫描插件，减少了漏洞扫描的盲目性。扫描程序以独立的插件形式实现，有利于系统的扩展。

此外，本文还在分析漏洞扫描实现原理的基础上，探讨了一些常见的漏洞如 CGI、缓冲区溢出、拒绝服务及简单口令探测等应采取的扫描方法，并结合 Winsock 给出了其编程实现方法。

最后通过系统测试，证明了本系统的有效性，并通过与 Xscan 扫描器选取相同漏洞扫描插件对同一台主机进行扫描的总时间对比，证明本文所采用的策略驱动漏洞扫描确实提高了漏洞扫描的效率，且在针对开放端口较少的目标主机时，这种优势较为明显。

但是由于时间和条件所限，该扫描器的一些功能还需进一步的完善，未实现的部分还需进一步实现并优化，因此还有很多工作需要完成：

1) 对操作系统的识别采取的方法比较简单，识别结果比较粗糙，将来应该采用比较精确的“协议栈” 识别技术，来综合分析得出比较准确的操作系统类型，以提高扫描的效率。

2) 本扫描器对各种漏洞的检查数目和总类不够丰富，及时扩充插件库，保证漏洞扫描的最新性，广泛收集漏洞数据，是今后工作的重点。

参考文献：

[1] Zhang Tao, Hu Mingzeng, Li Dong, Generate attack graph for network security analysis, Journal of Information and Computation Science, 2005, 2(3), pp.605-615

[2] Newson Alan, Network threats and vulnerability scanners, Network Security, 2005, 12, pp.13-15

[3] Denning, Cryptography and Data Security, Addison-Wesley Publishing Company, 1987, pp.1-1

[4] Tian H, Huang L, Luo Y, Arm up administrators: Automated vulnerability management, Proceedings of the International Symposium on Parallel Architectures, Algorithms and Networks, 2004, pp.587-593

[5] 戴祖锋，张玉清，胡予濮，安全扫描器综述，计算机工程，2004，2，pp.5-8

[6] 翟钰，系统脆弱性研究及其数据库的设计与实现，[学位论文]，西安建筑科技大学，2004

[7] Martin Robert, Integrating your information security vulnerability management capabilities through industry standards (CVE and OVAL), Proceedings of the IEEE International Conference on Systems, 2003, 2, pp.1528-1533

[8] 陈铁明，基于插件的安全漏洞扫描系统设计，计算机工程与设计，2004，25 (2)，pp.194-196

[9] 黄著，苏璞睿，蒋建春，可扩展的网络弱点扫描系统设计与实现，计算机工程，2002， 28 (2)，pp.149-151

[10] 邢栩嘉，计算机网络脆弱性评估系统的设计与实现，[学位论文]，清华大学，2004

[11] Tanemo Fumiyuki, Chikira Kazuaki, High-speed scanning system for security diagnosis, NTT Technical Review, 2004, 2(11), pp. 68-71

[12] 张平，蒋凡，一种改进的的网络安全扫描工具，计算机工程，2001，27(9)，pp.107-109

[13] 李鹏，杨献荣，许丽华，网络漏洞扫描器的设计与实现，计算机工程，2003, 29 (8)，pp.116-117, 142

[14] Manes Gavin, W Schulte, Dominic Guenther Seth, NetGlean: A methodology for distributed network security scanning, Journal of Network and Systems Management, 2005, 13(3), pp.329-344

[15] 阀喜戎，Windows系统中基于缓冲区溢出的攻击分析，计算机应用，2002，1，pp.33-35

[16] 蒋卫华，李伟华，杜君，缓冲区溢出攻击：原理，防御及检测，计算机工程，2003，29(10)，pp.5-7

[17] Churning Matthews, Manton M, A Web interface for nessus network security scanner, Proceedings of the International Conference on Internet Computing , 2004, 4, pp. 383-389

[18] 刘宇东，基于 plug-in 的网络漏洞扫描系统信息网络安全，计算机工程与设计，2003，32(12)，pp.49-50

[19] 汪生，孙乐昌，基于CVE的安全脆弱性数据库系统的扩展设计，微电子学与计算机，2005，22(10)，pp.152-155

[20] 洪宏，网络安全扫描技术研究，计算机工程，2004, 30 (10)，pp.54-56

[21] 刘海燕，杨洪路，王崛，一个基于网络的脆弱性扫描系统，计算机应用，2003, 23 (7)，pp.98-99, 102

[22] Panjwani Susmit, Tan Stephanie, Jarrin Keith M, An experimental evaluation to determine if port scans are precursors to an attack, Proceedings of the International Conference on Dependable Systems and Networks, 2005, pp.602-611

[23] 刘健，曹耀钦，网络隐蔽扫描技术研究，计算机工程与设计，2004，25(2)，pp.239-243

[24] 王雨晨，系统漏洞原理与常见攻击方法，计算机工程与应用，2001，38(8)，pp.62-65

[25] 单葱胜，李小勇，李建华，主动检测网络扫描技术，计算机工程，2003，29(19)，pp.8-10

[26] 王轶骏，薛质，李建华，基于TCP/IP协议栈指纹辨识的远程操作系统探测，计算机工程，2004，30(18)，pp.7-9

[27] 邢栩嘉，林闯，蒋屹新，计算机系统脆弱性评估研究，计算机学报，2004，27(1)，pp.1-11

[28] Ofir Arkin, A remote active OS fingerprinting tool using ICMP, http://www.sys-security.com/archive/articles/login.pdf, 2006.3

[29] 濮青，基于包重传延时的主机操作系统探测技术的设计与实现，小型微型计算机系统，2005，26(4)，pp.584-587

[30] 孙百勇，向剑伟，莫正坤等，Web服务CGI安全漏洞分析与检测，华中科技大学学报(自然科学版)，2003，31(3)，pp.66-68

[31] 张吉才，张翔，王韬，网络CGI漏洞扫描器的研究与实现，计算机工程与设计，2003，24 (12)，pp.47-48, 77

致 谢

三年的学习生活即将结束，值此论文完成之际，我心情无比激动。在此，衷心地感谢我的导师彭新光教授，在这三年中，彭老师给了我莫大的教诲和热切的关怀。彭老师治学严谨、学识渊博，谆谆教诲让我受益匪浅；他那严谨求实、孜孜不倦的学习态度、一丝不苟的工作作风、以及热情随和的待人之道都给我留下了深刻的印象，使我不仅学到了专业知识，而且也学到了为人处事、做人的道理，这些对我今后的工作、生活都是一笔宝贵的精神财富。

感谢我的同学张俊丰、王跃云、任凯锋、蔡思飞、赵月爱、吴文刚、李爱方等，在学习上一直给予我帮助，耐心地帮助我解决研究中的问题，不断鼓励我上进，这是我论文得以完成的重要条件。感谢三年来在一起学习、生活的同学们，这是一段我永远难忘的岁月。

感谢我的家人，我的父母、爱人和弟弟对我的支持、关怀和鼓励！没有你们的支持和鼓励，我不可能顺利完成研究生三年的学业，你们的支持是我最大的动力。

再次衷心感谢所有在我求学期间以及论文写作过程中，曾给予我支持、关心和帮助的各位老师、同学和朋友。

攻读学位期间发表的学术论文目录

Peng Xinguang, Yan Meifeng, Trusted Anomaly Detection with Context Dependency, The Journal of Shanghai Jiaotong University (上海交通大学学报英文版). （已录用）