第 24 卷 第 1 期 华侨大学学报(自然科学版) Vol. 24 No. 1
2003 年 1 月 Journal of Huaqiao University (Natural Science) Jan. 2003

文章编号 1000-5013(2003)01-0092-06

利用 VC++ 编程实现防火墙数据包过滤

吴 金 龙

(华侨大学信息科学与工程学院, 福建 泉州 362011)

摘要 随着计算机网络深入社会、经济、国防、科技与文教等各个领域, 计算机系统的安全问题正变得日益复杂和突出. 资源共享和网络分部更增加了网络收到威胁和攻击的可能性, 于是基于包过滤加状态检测的防火墙系统成为保护网络安全的工具. 文中介绍一种集数据包过滤、日志、代理服务于一体的复合型防火墙系统, 着重论述利用 VC++ 编程技术实现数据包过滤的软件方法.

关键词 复合型防火墙, 包过滤, 网络安全

中图分类号 TP 393.08 ; TP 312 文献标识码 A

防火墙(Firewall)是一种有效的网络安全模型. 实现防火墙的技术有多种, 最主要的技术包括数据包过滤和代理服务. 传统的包过滤在遇到利用动态端口的协议时会发生困难, 如文件传输协议 FTP. 它需要实现将所有可能用到的端口打开, 这又会给安全带来不必要的隐患. 动态包通过状态检测检查应用程序的信息(如 FTP 的 PORT 和 PASS 命令), 以便判断此端口是否允许需要临时打开; 而当传输结束时, 端口又恢复为关闭状态. 组成复合型防火墙有两个基本要素——自适应代理服务器与动态包过滤器. 代理可以提供极好的访问控制、登录以及地址转换功能, 对进出防火墙的信息进行记录, 便于管理员监视和管理系统.

1 系统模块的设计

根据小型实用的特性, 我们利用 1 台 Cisco 2600 路由器和 3 台 PC 兼容机, 构建主机过滤型的防火墙. Cisco 2600 路由器作为内部网络和外部网络的中间接口, 充当过滤路由器. 运行防火墙程序的 PC 机作为堡垒主机, 充当代理服务器, 提供 HTTP 代理. 另外两台 PC 机作为内部网络主机. 过滤路由器使用了两个 Ethernet 口, 其中 Ethernet 0/0 口的 IP 地址设为 210.34.251.80, 连接电脑系的路由器, Ethernet 0/1 口的 IP 地址设为 192.168.0.1. 它们作为两台内部网络主机的网关, 两台内部网络主机通过交换机连接在一起, 交换机的 Up Link 口连接路由器.

系统规定, 任何外部网络的主机只能与内部网络的堡垒主机建立连接. 防火墙软件自带一个简单的包过滤系统. 当个人上网时, 如用户机器装有支持 NDIS 规范的网卡, 则可以启动软件包过滤系统, 为用户提供防火墙级的安全保障. 代理服务进行基本身份验证和用户管理, 限

收稿日期: 2002-09-17 作者简介: 吴金龙(1943-), 男, 教授

制最大连接数目, 避免系统超载. 防火墙的体系结构, 如图 1 所示.

系统共分为 4 个模块: 包过滤模块、用户管理模块、日志模块和代理模块. 其中包过滤模块在 OSI 模型的网络层实现, 它分为监听模块和过滤模块. 两个子模块有选择的让数据包在内部和外部主机间进行交换, 根据设定的 IP 地址或端口号, 允许某些数据包通过, 同时又阻断某些数据包. 日志模块对监听子线程和代理子线程建立详细的日志记录, 管理员可以查看这些日志, 并可根据需要将其保存成 SQL 数据库表格, 建立永久日志, 以备日后检查. 代理模块提供 HTTP 1.1 代理, 代理过程中要求基本认证, 用户名及密码从用户管理模块取得, 限制最大连接数为 80. 用户管理模块登记要求使用代理的用户, 代理用户缴纳一定金额后可以获得用户密码. 代理用户在使用代理过程中, 程序自动根据其使用代理的下行流量来进行计费[1].

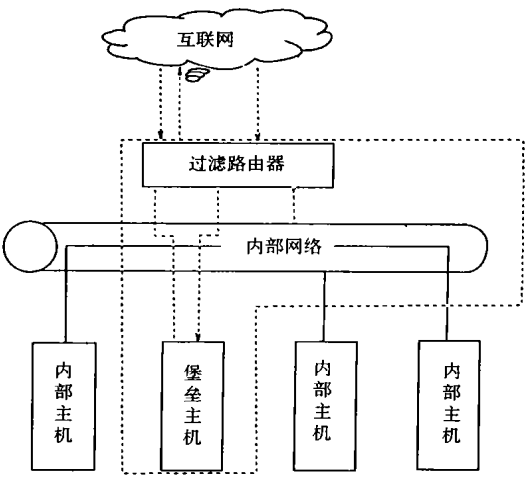

图 1　复合型防火墙的体系结构

2　利用 VC++ 的多线程特性编程实现数据包过滤

大家知道, VC++ 6.0 对网络通信、硬件驱动等方面的程序开发提供了强大的支持, 使用 DDK 和 Winsock 可以开发底层的网卡驱动程序. VC++ 具有多种优良的编程特性, 主要体现在下列几方面.

2.1　多线程编程和执行

系统的各个进程在自己的程序空间内运行可以包括一个或多个线程, 其中至少有一个主线程. 这些线程之间互不干扰. 操作系统根据系统核心的调度算法, 轮流将确定的时间片分配给处于就绪状态的线程, 使得每个线程均有机会执行. 使用多线程可以利用 Windows 的并发执行特点, 极大地提高程序运行效率.

2.2　NDIS 网络驱动程序

NDIS(Network Driver Interface Specification) 是 Micro Soft 和 3COM 公司联合制定的网络驱动规范, 并提供了大量的操作函数. 它为上层的协议驱动提供服务, 同时屏蔽了下层各种网卡的差别. NDIS 向上支持 TCP/IP, NW Link IPX/SPX, NETBEUI 等多种网络协议, 向下支持不同厂家生产的多种网卡. NDIS 还支持多种工作模式, 支持多处理器, 提供一个完备的 NDIS 库. 库中所提供的各个函数都是工作在核心模式的, 不宜直接操作, 需要寻找另外的接口[2].

2.3　采用 VXD 技术

Windows XX 使用 Intel CPU 的两个保护级 Ring0 和 Ring3. 系统进程运行于 Ring0, 因而具有对系统全部资源的访问权和管理权. 普通用户进程运行于 Ring3, 只能访问自己的程序空间, 不允许对系统资源进行直接访问, 许多操作受到限制. 要实现对网卡的直接读写, 进程必须

运行在 Ring0 优先级,为此必须采用 VXD 虚拟设备驱动程序作为 Windows XX 系统和物理设备之间的接口. 它不仅适用于硬件设备,也适用于按 VXD 规范所编制的各种软件"设备". Windows XX 系统的底层功能只能在 VXD 中调用,应用程序如果要用,必须编个 VXD 作为中介. VXD 作为应用程序在系统中的一个代理,应用程序通过它来完成自己做不到的事情,并通过这一手段,Windows XX 系统为普通应用程序留下了扩充接口.

2.4 建立 Winsock2 规范

建立 Winsock2 规范的目的是提供一个与协议无关的传送接口,它包含了一组针对 Windows 的扩展库函数,以使程序员能充分地利用 Windows 消息驱动机制进行编程. Winsock 规范定义并记录了如何使用扩展库函数与 Internet 协议族的连接,所有的 Windows Sockets 实现都支持数据流套接口和数据报套接口,并支持多线程的 Windows 进程. 一个进程包含了一个或多个同时执行的线程. 它有利于多个应用程序在多任务情况下更好地运作. Windows 应用程序调用 Windows Sockets 的 API 实现相互之间的通信,它们之间的关系如图 2 所示[3]. 根据 C^{++} 的特性,结合防火墙的运行机理,我们的研究集中到数据包的过滤和捕获上. 在通常情况下,以太网在同一网络线段的所有网络接口都可以侦听到物理媒体上传输的所有数据,而每一个网卡都有一个唯一的 48 比特的 MAC 地址. 一个网络接口只响应与自己硬件地址相匹配的数据帧或发向所有机器的广播帧. 在一个实际的系统中,数据的收发是由网卡来完成的,网卡接收到传输来的数据,再根据网卡驱动程序设置的接收模式判断该不该接收. 认为该接收的数据就接收,并产生中断信号通知 CPU,认为不该接收的数据就丢掉不管,或者说数据被截断了. 驱动程序接收数据后放入信号堆栈,让操作系统处理. 网卡的接收模式包括广播方式、小组传播、直接发送和混杂模式. 在混杂模式下的网卡能够接收一切通过它的数据,而不管该数据是否要传给它的. 我们正希望网卡能接收一切它所能接收的数据. 如图 3 所示,机器 A,B,C 与集线器 HUB 相连接,集线器 HUB 通过路由器访

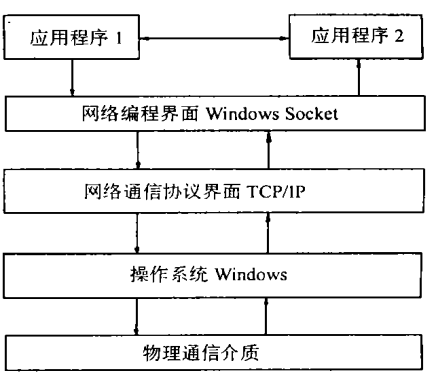

图 2 应用程序与 Winsock 关系图

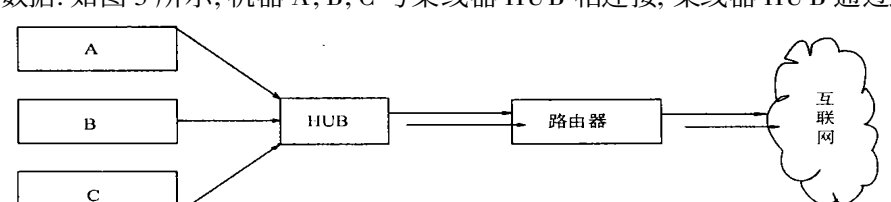

图 3 典型的小型局域网联入互联网的模式

问外部网络. 几个部门的集线器通过路由器连接. 假设机器 A 的管理员为了维护机器 C,使用 FTP 命令向机器 C 进行登录. 其 FTP 口令经过应用层 FTP 协议、传输层 TCP 协议、网络层 IP 协议、数据链路层上的以太网驱动程序逐层的包裹,最后送到了物理层上. 接下来,数据帧再由 HUB 向每一个节点广播由机器 A 发出的数据帧,机器 B 接收到由 HUB 广播发出的数据帧,并检查地址是否和自己的地址相匹配,如不是发向自己的数据帧,则丢弃不予理睬. 而机器

C 也接收到数据帧, 在比较之后发现是发给自己的, 因此 C 就对数据帧进行分析处理. 如果机器 B 上的管理员想知道究竟登录机器 C 上 FTP 口令是什么, 则只需把网卡置于混杂模式, 并对接收到的数据帧进行分析, 从而找到包含在数据帧中的口令信息. 综上所述, 我们对数据包的拦截, 只要把网卡置于混杂模式, 然后捕获数据包, 最后分析数据包. 为节省篇幅, 以下仅列出实现将网卡置于混杂模式, 捕获数据包和分析数据包各项功能的部分源程序代码为

```
# include "headers. h"
# define INTERFACE "eth0"
/* Prototype area* /
int Open _Raw _Socket(void);        //自己定义函数,
int Set _Promisc(char * interface, int sock);    //将网卡置成混杂模式
int main() {              //获取流经网卡的数据的主程序
int sock, bytes -recieved, fromlen;
char buffer[65535];          //数据缓冲
struct sockaddr -in from;
struct ip * ip;
struct tcp * tcp;
sock = Open -Raw -Socket();
Set -Promisc(INTERFACE, sock);
while(1)
{
fromlen = sizeof from;
bytes -recieved = recvfrom(sock, buffer, sizeof buffer, 0, (struct sockaddr * ) &from,
&fromlen);
printf( "\nBytes received ::: %5d\n", bytes -recieved);
printf( "Source address ::: %s\n", inet -ntoa(from. sin -addr));
ip = (struct ip * )buffer;
/* See if this is a TCP packet(看是否为 TCP 包)* /
if(ip- > ip -protocol = = 6) {
printf( "IP header length ::: %d\n", ip- > ip -length);
printf( "Protocol ::: %d\n", ip- > ip -protocol);
tcp = (struct tcp * )(buffer + (4* ip- > ip -length));
printf( "Source port ::: %d\n", ntohs(tcp- > tcp -source -port));
printf( "Dest port ::: %d\n", ntohs(tcp- > tcp -dest -port));}}}
```

以下是设置过滤端口的程序. 请注意, 设置过滤 IP 和设置过滤端口的程序大同小异:

```
void CFilterDlg::OnAddport()
{
UpdateData(TRUE);        //刷新数据
```

```
if (! m - editport. IsEmpty())        // 输入端口非空
{
// 最大的端口数为 32
if (m - listport. GetCount() < = MAX - QUERY)
{
// 该端口是否已在列表中
if (m - listport. FindString( - 1, m - editport)! = LB - ERR)
{
AfxMessageBox("对不起,该端口号已经存在于列表中!");
((CEdit* )GetDlgItem(IDC - FILTER - EDIT)) - > SetSel(0, - 1);
return;
}
// 将端口加入列表
m - listport. InsertString(m - listport. GetCount(), m - editport);
/*
CString strsec= GetKeyName("FilterPort");
int pos= strsec. ReverseFind('t');
if (pos! = - 1)
{
strsec. Delete(0, pos+ 1);
int i;
char*  buf= strsec. GetBuffer(MAX - BUF);
sscanf(buf, "%d", &i);
i= i+ 1;
strsec. Format("%d", i);
strsec. ReleaseBuffer();
}
else strsec= "1";
WritePrivateProfileString("FilterPort", "Port"+ strsec, m - editport, file);
* /
m - editport= "";
}
else AfxMessageBox("对不起,超过最大数目!");
}
else AfxMessageBox("请输入端口号");
}
```

如果输入的端口号为空,则提示出错. 其次,如果已经设置了 32 个端口,同样将提示出错.

最后, 检查输入的端口号是否已设置, 如果端口号已经设置, 再次提示出错. 反之, 将输入的端口号加入.

3　结束语

如上所述, 我们设计的防火墙是一种供个人和小型 LAN 使用的复合型防火墙. 其主要功能是帮上网用户抵挡来自外部网络的入侵和攻击, 防止信息泄露. 它可以探测数据包. 当收到攻击时可以自动报警, 并提供强大的反追踪功能, 将入侵者的踪迹显示出来, 包括时间、端口号、MAC 和 IP 地址等. 它能够对企图入侵的数据包详细的日志记录, 并发出各种攻击警告. 它提供 HTTP 等协议的代理服务, 避免局域网内部主机直接访问外部的因特网, 造成网络系统的安全隐患. 最后应当指出的是, 由于防火墙假设了网络边界和服务, 对内部的非法访问难以有效地控制 . 因此, 防火墙适合于相对独立的 Intranet. 另外, 由于网络实验室所提供的设备和条件限制, 我们所研制的防火墙仅实现常用的部分协议的代理 . 代理的验证模式为基本验证模式, 它的认证内容在网络上以数据包的形式发送, 容易被一些居心不良的网络用户窃取. 因此, 包过滤系统的功能还应进一步完善, 代理的效率和速度还有待提高. 但是, 随着网络技术的不断发展, 利用软件方法实现简单而实用的防火墙是完全能办得到的 . 目前, 动态包过滤技术可以与其数据流相适应, 向着更具柔和性和多功能的方向发展 . 例如, Cisco PIX 防火墙的过滤规则可由路由器快速配置 . 在操作系统的内核级实现了代理服务, 做到协议透明和应用透明, 并提供加密的 IP 通信, 所有这些都是我们的研究内容 .

参考文献

1　李海泉, 李　健 . 计算机网络安全与加密技术[M]. 北京: 科学出版社, 2001. 95～109

2　Chris C 著. Windows WDM 设备驱动程序开发指南[M]. 孙　义等译. 北京: 机械工业出版社, 2000. 152～275

3　薛静锋. VC++ 高级开发教程[M]. 北京: 人民邮电出版社, 1999. 407～465

Implementation of Firewall Packet Filtration by Applying VC++ Programming

Wu Jinlong

(College of Info. Sci. & Eng., Huaqiao Univ., 362011, Quanzhou, China)

Abstract　The safety of computer system becomes increasingly complex and prominent in pace with the penetration of computer network into society, economy, national defense, science and technology, culture and education, and some other fields. Resource sharing and network subsection aggrandize the possibilily that the network will be menaced and attacked. Thus the firewall system based on pack filtration and status detection becomes tool for protecting the security of network. The author presents here a composite type firewall system into which data packet filtration, daily record and proxy service are integrated. Emphasis is put upon the software method in the implementation of data packet filtration by applying VC++ programming

Keywords　composite type firewall, packet filtration, security of computer network