基于 Windows NT 的多级过滤防火墙系统设计与实现

Design and Implement of A Multilevel Filter Firewall Based on Windows NT Stucture

高志伟 毛晚堆 贾玉锋 （石家庄铁道学院计算机系 050043）

摘要：本文基于 Windows NT 平台，提出了一种新的结合内核态和用户态进行多级过滤的防火墙系统，从底层提升防火墙性能。系统分别对网络层、传输层、应用层进行多级过滤，提高了系统的安全性。为提高效率，改进了目前的 BM 模式匹配算法，并结合反向有限自动机，提出了一种新的基于多模式匹配算法。

关键词：防火墙 NDIS TDI Winsock 模式匹配

1 前言

目前国内的防火墙技术大多是通用 Linux 内核的 IPChains 或 Netfilter 架构，针对 Windows NT、2000 结构下的防火墙研究并不是很广泛。而 Windows NT、2000 是目前使用得最广泛的操作系统，因此针对 Windows 平台研究如何保护用户的信息安全具有重要的意义。

2 Windows 常用防火墙实现技术分析

2.1 用户态下主要有以下三种技术

(1) Winsock Layered Service Provider (LSP) 分层技术。这种方法好处是可以获得调用 Winsock 进程的详细信息。因此可以用来实现 QoS，数据流加密等目的。但是，如果应用程序直接通过 TDI(Transport Driver Interface)调用 TCP/IP 来发送数据包，这种方法就无能为力了。对于一些木马和病毒来说要实现通过 TDI 直接调用 TCP/IP 是一件很容易的事情。因此，大多数的防火墙都不使用这种方法。

(2) Windows 2000 包过滤接口。Windows 2000 IPHLP API 提供了安装包过滤器的功能。但是，包过滤的规则有很多限制，对于防火墙来说远远不够。

(3) 用全局的 hook api。hook winsock 函数替换系统自带的 WINSOCK 动态连接库。这是目前在应用态下实现防火墙的常用方法。

很显然，在用户态下进行数据包拦截最致命的缺点就是只能在 Winsock 层次上进行，而对于网络协议栈中底层协议的数据包无法进行处理。对于一些木马和病毒来说很容易避开这个层次的防火墙。

2.2 内核态主要技术

(1) NDIS Hook Driver。这是目前大多数防火墙所使用的方法。Hook 的概念在 Windows9x 下非常流行，而且实现也很容易。在 Windows NT/2000 下面实现 Hook 可以通过修改 NDIS.SYS 的 Export Table。在 Windows NT/2000 下，可执行文件(包括 DLL 以及 SYS)都是遵从 PE (Portable Executable)格式。所有向其他操作系统组件提供接口的驱动程序都有 Export Table，因此只要修改 NDIS.SYS 的 Export Table 就可以实现对关键 NDIS API 的挂接。由于协议驱动程序在系统启动的时候会调用 NdisRegisterProtocol 来向系统进行协议注册，因此这种方法关键在于修改 NDIS.SYS 所提供的 NdisSend 函数的起始地址。在用户态模式要修改 PE 文件格式可以用一些 API 来实现，而 NDIS.SYS 位于系统的核心内存区，要修改 NDIS.SYS 就得通过写驱动程序来实现，也就要求对 PE 文件格式有比较深入的了解。

(2) Win2k Filter - Hook Driver。这是从 Windows2000 开始系统所提供的一种驱动程序，该驱动程序主要是利用 ipfiltdrv.sys 所提供的功能来拦截网络数据包。Filter-Hook Driver 结构非常简单，易于实现。但是正因为其结构过于简单，并且完全依赖于 ipfiltdrv.sys，Microsfot 并不推荐使用 Filter-Hook Driver。

3 系统设计及采用技术分析

根据 Windows2000 的网络体系结构，本系统防火墙采用了一种将核心态和用户态相结合的多级过滤技术。在核心态中，网络层利用 NDIS 技术实现对数据包的过滤，传输层利用 TDI 技术实现对应用进程的过滤；在用户态下，在应用层中利用 WINSOCK 技术实现对数据包内容的过滤，从而通过多级过滤，达到进一步提高系统安全性的目的。

系统在最底层采用 NDIS 技术，这样能够截获到所有从网卡上收发的数据包，可以对网络上的所有数据包进行检查。利用 NDIS 技术，开发一种协议驱动程序，和底层网络接口绑定，并将网卡接收的所有帧，向上传递给此协议驱动程序。这样可以截获流经本机网卡的数据包，并根据需要分析数据包内容。从 Windows 网络层次体系来看，NDIS 层位于链路层的中上方，处于上层协议和下层网络设备之间，能够对任何由本机传输的数据包进行操作。而协议驱动工作在更高的层次，和具体的传输协议相关，对数据过滤不全，不能将主机和外界完全隔离。因此在低层利用 NDIS 驱动截获，能保证对数据包的过滤更完善，可靠。应用 NDIS 驱动技术

构建 Windows 平台的防火墙是未来的一种技术趋势。在 NDIS 对数据包过滤后，将通过规则检查的数据包传向传输层，传输层中利用 TDI 技术开发出建立于 TCP 之上的虚拟驱动层，相当于在应用层和核心层之间建立一接口，使所有通过网络的 IRP 请求包都需要经过虚拟驱动层的判断，才能决定是否继续传输。系统还利用了在 TDI 层的网络数据拦截可以得到操作网络数据包的进程详细信息的特性，可以监视是哪个应用程序正对网络进行访问。数据包经过网络层、传输层的过滤后，向上传输到应用层。在应用层中，实现第三级过滤，针对数据包内容进行过滤，主要通过修改系统 WINSOCK 函数，使系统 WINSOCK 函数指针指向自己编写的函数，达到对数据包内容过滤的目的。因此，本系统通过三层过滤，大大地增加了系统的安全性。

4　系统具体实现

本文提出的多级过滤防火墙系统主要由数据包截获模块、包过滤模块、数据包内容过滤模块等几部分组成，分别完成网络层、传输层、应用层的过滤功能，系统在针对数据包内容过滤模块中，改进了目前的 BM 模式匹配算法，并结合反向有限自动机，提出了一种新的基于多模式匹配算法，提高了系统的效率。

试验数据如下：试验中分别选取字符数为 600,000、60,000,000、600,000,000、6,000,000,000 的英文短文，从中查找指定单词，3 种算法的查找时间如下表所示。系统测试时采用的调试工具为 DebugView。

	600,000 字符	60,000,000 字符	600,000,000 字符	6,000,000,000 字符
普通算法	0.11761ms	382.61176 ms	5100.50468 ms	51411.23187 ms
BM 算法	0.11314 ms	93.07551 ms	2009.06121 ms	22230.01638 ms
改进算法	0.11141 ms	91.17129 ms	1940.84863 ms	20667.42282 ms

从表中可以看出，当文本长度比较小时，改进算法与 BM 算法相比，优势不是很明显，但随着文本长度增大，改进算法的查找时间明显减小。

4.1　数据包截获模块

在网卡驱动程序和传输驱动程序之间有一层 NDIS 驱动，利用 NDIS 驱动层截获网卡收到的数据包，从而可以对数据包进行各种处理。例如，从底层传来的数据包，经过与规则的匹配，符合过滤规则的转发给上层，或截获上层即将发送的数据包，经过匹配，符合的数据包转发给网卡，发送出去。

在该模块中，首先在 NDIS 网络驱动层中注册一个小端口驱动 Miniport 接口和一个协议驱动 Protocol 接口。利用协议驱动与底层的链路层通信，这样，当链路层有数据包向上传输时，协议驱动可以及时截获数据包；利用小端口驱动与高层的协议层通信，当传输层有数据包向外发送时，小端口驱动同样可以及时截获到将要发送的数据包。具体流程如图 1 所示：

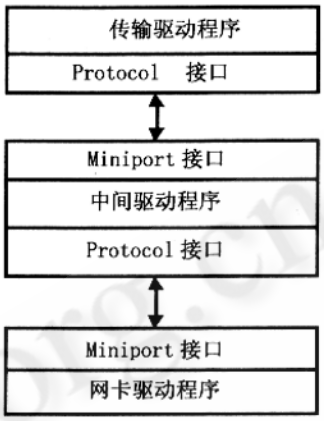

图 1　数据包截获模块流程图

在本系统中，发送数据时，需要在虚拟网卡的 Miniport 接口上截获数据，调用自定义的 MiniportSend 或 MiniportSendPackets 函数可实现截获即将要发送的数据。在截获到数据包后，首先，解析数据包，决定丢弃或通过。然后调用自定义的 NdisSend 或 NdisSendPackets 函数，转发包到链路层。在接收底层发来的数据包时，虚拟网卡首先在 Protocol 接口上调用自定义的函数 ProtocolReceive 或 ProtocolReceivePackets，截获从底层网卡驱动传来的数据包，经过虚拟驱动层的处理后，再调用定义在 Minport 接口层上的 NdisMIndicatePacket 或 NdisMXxxIndicatePacket 函数，将数据包传输到上层的协议驱动层中。具体处理流程图略。

4.2　数据包过滤模块

本模块主要实现在网络层进行数据包过滤。包过滤是在网络数据包的入口处截取 IP 包，对其包头进行分析，根据包的源地址、目的地址、源端口、目的端口、协议等部分进行综合检测，符合安全访问控制规则的包才给予转发，否则予以丢弃。其流程如图 2 所示：

在系统中，为了防止破坏原有数据包，首先建立了自己的 packet，buffer 和 memory，把从网卡上截取的数据包复制到 memory 中，然后通过把 packet 与 memory 建立映射，这样就可以直接对 packet 的数据进行修改，分析，而不必破坏原始数据。

4.3　数据包内容过滤模块

本系统在应用层上实现第三层过滤，针对数据包内容的过滤。当 Internet 上的某些用户提交给主机的信息中含有系统禁止的，非法的，或带危害性的字段时，系统会过滤掉该数据包，禁止该数据包的向上传输，达到保护服务器主机的目的。比如：当主机想针对含有某些信息的数据包禁止时，由于 ip 地址是灵活的，不可能利用单纯的根据 ip 地址进行的包过滤来实现针对内容的数据包的禁止，这样，就需要针对内容的过滤。当发现 Internet 用户的数据包中含有非法的信息时，则过滤掉该数据包，实现禁止用户对主机的访问。同时，在本系统中，针对数据包内容的过滤，还可以实现禁止主机用户对一些非法站点的访问。

本部分实现主要是利用钩挂技术，也就是将 Winsock 提供的函数替换成自己编写的函数。系统中，首先根据系统

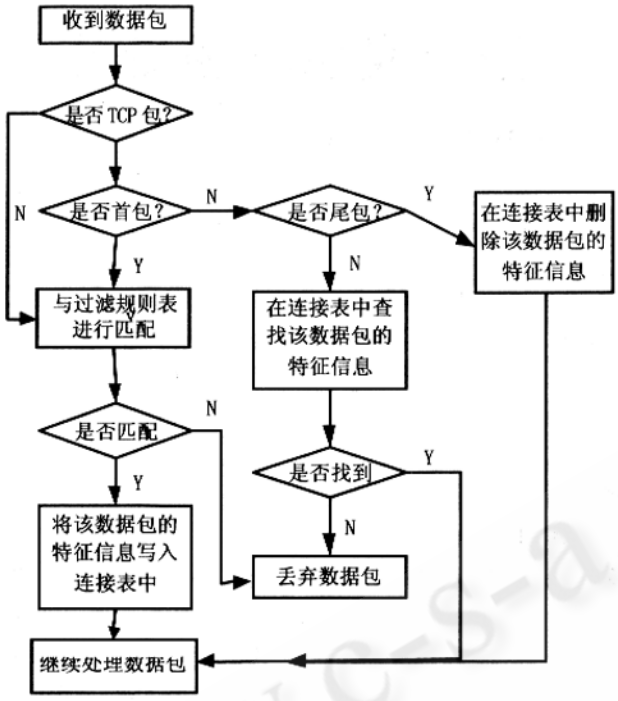

图 2　数据包过滤流程图

提供的 lpProtocolInfo 参数找到相应的系统服务提供者的路径，然后根据这个路径得到系统服务提供者提供的自身服务函数的指针，再将这些指针指向自定义的 Winsock 函数，实现钩挂技术。以后系统对网络的访问都要运行自定义的 Winsock 函数，而不再执行系统本身的的 Winsock 函数。这样，通过自定义的 Winsock 函数实现了基于内容的过滤。整个钩挂部分的流程如图 3 所示：

在数据包到达应用层时，自定义的 Winsock 函数要在数据包中查找是否包含有系统所禁止的模式串，该模块是决定系统效率的一个重要部分，因此，如何能够进行高效率的模式匹配是整个部分的关键。本系统在针对数据包内容过滤模块中，改进了目前的 BM 模式匹配算法，并结合反向有限自动机，提出了一种新的基于多模式匹配算法，提高了系统的效率。

常用的模式匹配算法有 Brute－Force(BF)算法、Knuth－Morris－Pratt(KMP)算法、Boyer－Moore(BM)算法，都是单字符串的模式匹配算法，其中 BM 算法最高效。改进 BM 算法的基本思想主要是利用本次匹配不成功，尽可能多的跳过字符。当 T[1:m]与 S[i:i+m－1]对齐进行匹配，如果匹配失败，则不但分析 S[i+m－1]，还要分析 S[i+m]这个字符，来决定 T[1:m]右移的距离 max(skip1(S[i+m－1]), skip2(S[i+m]))。因为本次匹配失败后，T[1:m]至少要移动 1 个位置，在一般情况下，S[i+m]这个字符会出现在下一次的匹配过程中，于是在匹配失败后，可结合考虑 S[i+m－1]和 S[i+m]，而不单只考虑 S[i+m－1]，这样，

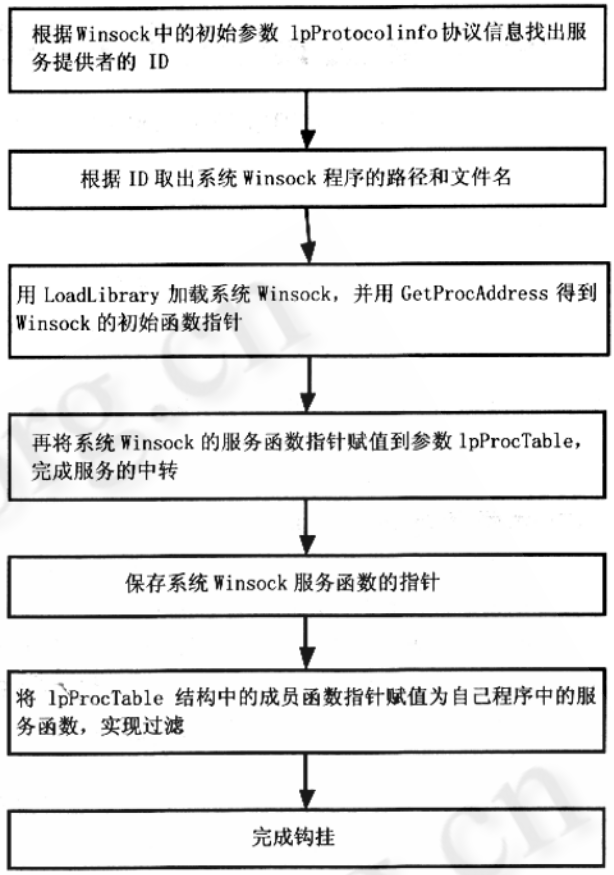

图 3　钩挂流程图

就可以使得模式串最大右移的距离是 m＋1；而 BM 算法中的最大右移距离是 m，效率有所提高。

在本系统中，需要匹配的模式串常常有多个，因此需要多模式匹配，多模式匹配的经典算法是基于有限自动机 DF－SA (deterministic finite state automata)的算法，为了能够高效的实现多模式串的匹配，且能够将改进的 BM 算法与多模式匹配算法结合起来，本系统针对传统的有限自动机进行了改进，生成一种反向有限自动机，并利用反向有限自动机进行多模式的匹配。反向有限自动机的构造中，每个模式串的字符从后到前，加到树形的自动机中，同时，匹配过程中，目标串的输入也是从后到前的顺序，即逆向扫描。具体实现步骤及算法复杂度分析略。

5　小结

本系统以 Windows 操作系统为开发平台。在核心态，网络层利用 NDIS 技术实现对数据包的过滤，传输层利用 TDI 技术实现对应用进程的过滤；在用户态，在应用层中利用 WINSOCK 技术实现对数据包内容的过滤，这样达到多级过滤，进一步提高了系统的安全性。

（下转第 28 页）

(上接第 25 页)

然而作为一个完整的主机防火墙产品,本系统还有许多后续工作要做,还应该包括更多的功能,比如,防病毒功能,以及检测更多攻击的能力、利用人工智能技术来检测网络攻击等。

参考文献

1 Kim E C. The multi-layer VPN management architecture[C], Proceedings of the Sixth IFIP/IEEE International Symposium, 1999:187-200

2 Stalling W., Network and Internetwork Security, Principles and Pratice, New Jersey: Prentice-Hall, Inc, 1995.

3 FAN Jang jong, SU Kehyih. An efficient algorithm for match multiple patterns[J]. IEEE Transon Knowledge and Data Engineering, 1993,5 (2):339-351.

4 石磊等,NDIS 技术在个人信息安全方面的应用,网络安全技术与应用,2002.5。

5 王永成、沈洲、许一震,改进的多模式匹配算法,计算机研究与发展,Vol .39,No.1,Jan.2002。

6 胡宇光,驱动程序怎样和应用程序通信, http://www.driverdevelop.com

7 朱雁辉,Windows 防火墙与网络封包截获技术,电子工业出版社,2002。

8 Art Baker, Jerry Lozano, Windows 2000 设备驱动程序设计指南,机械工业出版社,2001。