基于多 Agent 内核级网络数据包的研究与应用

苏朋程，曹 斌
（贵州大学 信息学院，贵州 贵阳 550003）

摘 要：采用多 Agent 对网络数据包在 Windows 内核级进行有关的研究，使用 NDIS 中间驱动接口技术路线，提高对 TCP/IP 网络数据包的处理速度，减少丢包率，提高 Windows 操作系统下的网络数据包的处理效率，可以应用于 Windows 操作系统下软件，如分布式防火墙（内核级）、VPN 客户端及服务器端软件、VLAN 软件、Windows 网络数据加密等。解决了以前对 Windows 网络数据包的处理只局限于应用程序接口层，如 Winsock 层次，对 Windows 网络数据包的处理效率不高，对 Windows 网络数据的拦截不彻底，不能在网络底层进行拆包和封包等问题。

关键词：Agent；防火墙；VPN；Winsock
中图分类号：TP311 **文献标识码**：A **文章编号**：1673－629X(2007)02－0216－03

Research and Application for Kernel Network Data Packets Based on Multi－Agent

SU Peng-cheng，CAO Bin
（Institute of Information and Computer Science，Guizhou University，Guiyang 550003，China）

Abstract：The multi－Agent and NDIS intermediate drivers interface technique way to research Windows kernel network data packet is presented in this thesis，to improve the speed of handling TCP/IP packets and reduce the losing of packets，so that it can enhance the efficiency of handling network data packets in Windows operation system，the result of research can be used to various use，such as the design of distributed firewall（kerneled），VPN server and client software，VLAN software，encryption Windows network data packets，and so on. In this thesis，solved the insufficiency of handling of Windows network data packets which is limited of application programme interface level beforetime，for instance，Winsock level，of which the handling efficiency of Windows data packets is insufficient，intercepting of Windows network data packets is not thorough.

Key words：Agent；firewall；virtual private network；Winsock

0 引 言

目前国内外对网络安全产品如防火墙、VPN、网络入侵检测（IDS）的需求越来越大，电子商务平台的构建必须建立在安全网络环境的基础上，因此，网络安全越来越重要。根据调查表明，网络安全隐患主要来自于公司或企业内部网络，硬件集中式（网络级）防火墙主要防范来自外网的安全威胁，对于内部网络的黑客攻击，集中式防火墙不能有效地防范这样的攻击。硬件防火墙技术目前已经成熟，其采用的技术是基于 Linux 的嵌入式技术。对于公司或企业内部网络的用户而言，大多数使用的操作系统是 Windows 操作系统，因此研究 Windows 网络数据包的处理问题很有实用价值，在此基础上可以研发 Windows 个人核心态防火墙，在不改变现有网络结构的情况下开发 Windows 操作系统下 IPsec VPN 软件系统，IDS，Windows 网络数据包加解密等。开发以上网络安全产品涉及到 Windows 操作系统下的网络数据包的处理问题，因此必须提高对网络数据包的处理效率，减少 TCP/IP 包的丢包率，提高网络安全产品的性能。

1 中间驱动处理 Windows 核心网络数据包

中间驱动是使用 NDIS 技术，中间层驱动程序，挂接在协议设备对象（包括 TCP/IP 设备对象）和网卡设备对象之间。任何进出网卡的网络封包，均必须首先经过中间层驱动程序的处理。从某种意义上分析，中间层驱动程序更象一个虚拟网卡。该虚拟卡封装了物理网卡，对物理网卡的一切网络访问操作，均必须先经

收稿日期：2006－05－12
基金项目：国家火炬计划项目（2005EB011453）；科技型中小企业技术创新基金资助项目（04c26215201399）
作者简介：苏朋程（1976－），男，贵州余庆人，硕士，研究方向为网络安全。

过虚拟网卡处理。公开源代码的微软官方 2000 DDK 的 Passthru.c 例程,使用中间层驱动技术,实现了网络封包截获。中间驱动是基于链路层和 IP 层之间的驱动,中间层驱动介于协议层驱动和小端口驱动之间,它能够截获所有的网络数据包(如果是以太网那就是以太网帧)。中间驱动可以避免在 Windows socket 层次对网络数据包进行不彻底的拦截和拆包封包处理[1,2],中间驱动对 Windows 网络数据包的所有处理函数将使用微软的开发工具里的 Windows DDK 里的网络库函数,而不是调用 Windows socket 提供的应用程序编程接口(API)里的函数[3],并且不是挂钩由传输服务提供者和名字解析服务提供者实现的 Winsock 服务提供者接口(SPI)和 ws2_32.dll 的相关函数。

中间驱动 IMD 接收网络数据流程图如图 1 所示。中间驱动 IMD 发送网络数据流程图如图 2 所示。

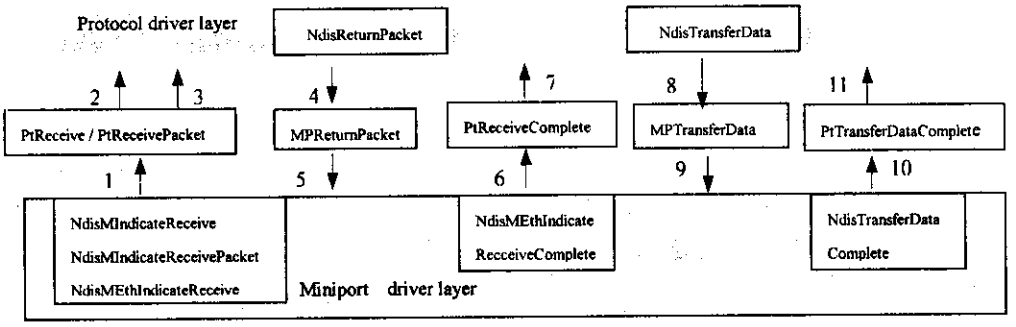

图 1　中间驱动 IMD 接收网络数据流程

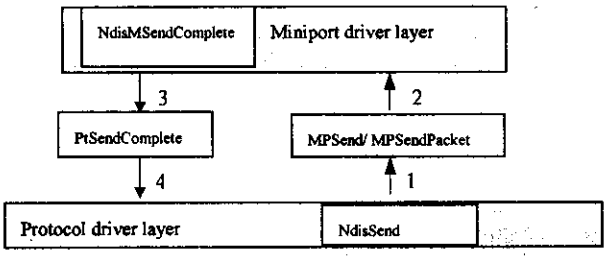

图 2　中间驱动 IMD 发送网络数据流程

2　多 Agent 协作处理 Windows 网络数据包

2.1　处理 Windows 网络数据包的多 Agent

正如文献[4]所说,艾真体(Agent) = 体系结构 + 程序,也可以把艾真体(Agent)定义为:艾真体(Agent) = 数据结构 + 程序,数据结构采取链表和指针等。每一个 Agent 都能独立地完成特定的功能,也就是整个程序中的一个子模块,可以把它们定义为一个类、成员函数或标准 C 的源程序文件 *.c。由于驱动程序不能使用 MFC 来开发,不能支持 VC + +,所以可以把每一个 Agent 定义为一个 C 的源程序文件 *.c 或函数。Agent 相互之间通过发送消息或相互调用(函数调用)来协作处理 Windows 网络数据包。

1)接收 Windows 网络数据包 Agent。

图 1 中,中间驱动程序工作于协议层和微端口层之间,通过调用 NDIS 的库函数来完成对 Windows 网络数据包的拆包及封包,在封包之前,可以对网络数据包进行解密,即嵌入解密 Agent,嵌入的位置是 NdisGetReceivedPacket。得到了一个完整的 packet 以后,就分配自己的 MyPacket,构成一个新的网络数据包,然后再对 MyPacket 进行解密。接收 Agent 的工作流程为:底层驱动使用 NdisMIndicateReceive / NdisMEthIndicateReceive,通知上层已经收到数报文;在 PtReceive 中如果通过 NdisGetReceivedPacket 得到了一个完整的 packet,就分配自己的 MyPacket,根据底下传上来的 packet 设置 MyPacket,然后调用 NdisMIndicateReceivePacket 通知 NDIS,NDIS 会接着调用上层协议驱动的相应 PtReceive 例程;如果此时 MyPacket 的 status 是 NDIS_STATUS_RESOURCES,就在本函数中释放分配的 MyPacket,否则在上层发送 4 的时候,在 MPReturnPacket 中释放 MyPacket;在 PtReceive 中如果通过 NdisGetReceivedPacket 不能得到一个完整的 packet,那就直接调用 NdisMEthIndicateReceive 等函数通知 NDIS;当上层协议驱动得到了一个完整的数据报文并且处理完毕以后,它会调用 NdisReturnPacket,然后 NDIS 会调用 MPReturnPacke;在 MPReturnPacket 中,释放自己分配的 MyPacket,然后同样向下层调用 NdisReturnPacket;下层会释放它们自己分配的 packet,如果 3 发生,当底层 miniport 驱动收到了一个完整的数据报文,它会调用 NdisMEthIndicateReceiveComplete,然后 NDIS 会调用 PtReceiveComplete。PtReceiveComplete 同样会调用 NdisMEthIndicateReceiveComplete,通知 NDIS"我们已经收到了完整的报文";当上层协议驱动得知底层已经收到了完整的数据报文以后,可能会调用 NdisTransferData,要求下层把剩余的数据传上来;8 的调用会导致 NDIS 调用 MPTransferData 例程;在 MPTransferData 中,做同样的调用 NdisTransferData。注意该函数的返回值:如果返回 success,说明剩余的数据立刻就传上来了,此时会立即返回,10、11 两步骤就不会调用;如果返回 pending,表明底层在此阻塞,底层会在稍后的时候调用 10;当底层 miniport 驱动做好了一个完整的 packet,它会调用 NdisTransferDataComplete,同样在 Pt-

TransferDataComplete 中,会作出同样的调用。

2) 发送 Windows 网络数据包 Agent。

图 2 中,与接收 Agent 相比较,发送 Agent 要简单得多。如果对网络数据进行加密,就要在该 Agent 中嵌入加密 Agent,嵌入的位置是在 Protocol driver 调用 NdisSend 向下层发送数据报文之后,在 MPSend/MPSendPacket 例程根据上层传下来的数据报文分配 MyPacket 的时候加密网络数据,组成新的具有安全性的网络数据包。发送 Agent 的具体工作流程为:Protocol driver 调用 NdisSend 向下层发送数据报文;Passthru 的 MPSend/MPSendPacket 例程根据上层传下来的数据报文分配 MyPacket,调用 NdisSend 发送到下层;如果返回 pending,就在 PtSendComplete 中释放 MyPacket 否则就在本函数中紧接着释放 MyPacket;当下层 miniport driver 发送完成 MyPacket 以后,会调用 NdisMSendComplete,NDIS 接着调用 passthru 的 PtSendComplete,在这个函数里边,应该释放 MyPacket,并且通知上层 protocol driver 去释放它们的 packet。

3) TCP/IP 网络数据包识别 Agent。

网络上有各种格式的网络数据包,有 TCP/IP 包和非 TCP/IP 包,例如对于 ICMP 这样的数据包,就要将其过滤掉,如何得到所期望的网络数据包(应用数据),这就需要对网络数据包进行识别,即对不同层次的网络数据包或数据帧进行识别,只允许应用数据通过,避免有害的网络数据。如何识别呢?文献[5]为设计网络数据包识别 Agent 提供了依据,判断依据是针对不同层次的网络数据包或数据帧的包头、帧头等相应字段的特有信息位。由于是安装在以太网卡上的服务驱动程序,在以太网得到的是以太网帧,过滤规则要和网络适配器绑定,对上层协议驱动层下传的数据包 packet,也需要应用规则识别,即是否允许网络数据包通过,最后根据数据的内容进行过滤,即过滤规则的 IP 地址、TCP/UDP 端口等。从微端口层对网络适配器到协议驱动层,在适配器和数据包操作的 NDIS 中间驱动库函数中,都要添加过滤识别的代码(由于篇幅限制,源代码略),如在 MPTransferData、MPReturnPacket 等函数里添加识别源代码。由于网络数据的传输是双向的,所以还必须有一个方向标志,标识出网络数据包是进入还是离开适配器。

4) 拦截 Windows 网络数据包 Agent。

对 Windows 网络数据包的拦截,在上文已经涉及到了,不同于挂钩技术,中间驱动能修改对底层网络数据操作的 NDIS 库函数,而钩子技术只能在 DLL 导出的函数之前或其后嵌入执行处理函数,其缺点是不能修改那些导出的函数,最重要的是不能彻底拦截底层网络数据,如以太网帧。对于拦截 Windows 网络数据包来说,处理的是微端口驱动层和协议驱动层的数据包,调用的函数是中间驱动函数。在微端口驱动层向上发送数据包到协议驱动层时,中间驱动要调用函数 NdisGetReceivedPacket 得到了一个完整的 packet 以后,此时就可以申请内存用于保存创建自己的新数据包,于是就实现了拦截从下层向上层传递的数据包。在上层协议驱动层向下层微端口层传输数据包,Protocol driver 调用 NdisSend 向下层发送数据报文之后,在 MPSend/MPSendPacket 例程根据上层传下来的数据报文分配 MyPacket 的时候,此时申请内存并分配一个自己的数据包,这样就实现了拦截上层传输到下层的网络数据包。源代码略。

2.2 多 Agent 的嵌入协作处理模式

各个 Agent 都能独立完成一定的功能,Agent 相互之间通过发送消息和相互嵌入使用其它的 Agent 提供的服务,同时相互协作提供一定的服务提供给其它的 Agent 使用,共同完成对 Windows 网络数据包的处理。由上面的结果得知,Windows 网络数据包处理的 Agent 分为接收 Windows 网络数据包 Agent、发送 Windows 网络数据包 Agent、TCP/IP 网络数据包识别 Agent、拦截 Windows 网络数据包 Agent,如果要开发 Windows 操作系统下的 VPN 等软件,则增加加解密和认证 Agent 即可。TCP/IP 网络数据包识别 Agent、拦截 Windows 网络数据包 Agent 以及加解密 Agent 和认证 Agent 分别嵌入到接收 Windows 网络数据包 Agent 和发送 Windows 网络数据包 Agent。具体的嵌入位置见上文所述。由于是相互嵌入式 Agent,所以它们融为一体,能很好地协作处理 Windows 网络数据包。

3 预期结果及展望

本程序是在 NTDDK5.0 环境下编译通过,生成驱动文件 passthru.sys,另外编写 inf 文件 netsf 和 netsf_m,这几个文件是安装驱动所必需的。编译结果如图 3 所示,使用观察工具 DebugView 察看运行结果,输出结果如图 4 所示。

从图中可以看出对于除了加解密 Agent 或认证 Agent 没有加入协作处理外,其它的 Agent 能协调地工作,有效地按照所设置的规则将 Windows 网络数据进行识别和拦截,如果加入加解密 Agent,则能实现加解密功能。由于开发中间驱动程序相当难,开发出成熟的 Windows 操作系统下的 VPN 软件还有待解决,但是开发内核级分布式防火墙已经成熟。当然,网络数据包的管理也是必须考虑的问题,幸运的是 NDIS 已

(下转第 222 页)

6 总 结

数据库加密是继访问控制后关于数据库安全的一个重要研究方向。采用数据库加密技术,即使系统的访问控制体系被黑客攻破也可以在一定程度上保证敏感信息不会外泄。文中指出了目前数据库加密技术存在的几点不足,并详细介绍了目前数据库加密的几种常用方法,及每种方法的优缺点。

参考文献:

[1] Rivest R L, Adleman L, Dertouzos M L. On Data Banks and Privacy Homomorphism[C]//In DeMillo R D. Foundations of Secure Computations. [s. l.]: Academic Press, 1978: 169 - 177.

[2] Domingo - Ferrer J. A New Privacy and Homomorphism Applications[J]. Information Processing Letters, 1996, 60(5): 277 - 282.

[3] 杨 勇,方 勇,周安民. 秘密同态技术研究及其算法实现[J]. 计算机工程, 2005, 31(2): 157 - 159.

[4] Hacigumus H, Iyer B, Li C, et al. Executing SQL over Encrypted Data in the Database - Dervice - Provider Model [C]//In Proceedings SIGMOD 02, Madison, International Conference on Management of Data. Wisconsin, USA: [s. n.], 2002: 216 - 227.

[5] Fischmann M, Gunther O. Privacy Tradeoffs in Database Service Architectures [C]//In Proceedings of BIZSEC 03, the First ACM Workshop on Business Driven Security Engineering. Berlin, Germany: [s. n.], 2003.

[6] 戴一奇,尚 杰,苏中民. 密文数据库的快速检索[J]. 清华大学学报, 1997, 37(4): 24 - 27.

[7] Davida G I, Wells D H, Kam J H. A Database Encryption System with Subkeys[J]. ACM Trans on Database Systems, 1981 (6): 31 - 37.

[8] Hwang M - S, Yang W - P. Multilevel Secure Database Encryption with Subkeys[J]. Data and Knowledge Engineering, 1997, 22(1): 117 - 131.

[9] 王庆梅,吴克力,刘凤玉,等. 一种子密钥数据库加密算法及其密钥管理方案研究[J]. 计算机工程与应用, 2003, 39 (11): 52 - 54.

[10] 戴一奇,尚 杰,陈 卫. 一种新的数据库加密密钥管理方案[J]. 清华大学学报:自然科学版, 1995, 35(4): 43 - 47.

[11] 余祥宣,崔永泉,崔国华. 分布式环境下数据库加密密钥管理方案[J]. 华中科技大学学报:自然科学版, 2002, 30(4): 43 - 45.

[12] Agrawal R, Kiernan J. Watermarking Relational Databases [C]//In 28th Int 'l Conference on Very Large Database. Hong Kong: [s. n.], 2002.

[13] Damiani E, Vimercati D S C, Finetti M, et al. Implementation of a Storage Mechanism for Untrusted DBMSs [C]//In Proc of the Second International IEEE Security in Storage Workshop. Washington DC, USA: [s. n.], 2003.

(上接第 218 页)

经解决了该问题[1,6],此处不再介绍。

```
E:\论文\vpn\vpn1>build -cz
BUILD: Object root set to: ==> objfre
BUILD: /i switch ignored
BUILD: Compile and Link for i386
BUILD: Compiling and linking e:\论文\vpn\vpn1
Compiling - vpn.rc for i386
Compiling - revpn.c for i386
Compiling - protocol.c for i386
Compiling - miniport.c for i386
Compiling - receive.c for i386
Compiling - send.c for i386
Compiling - logprint.c for i386
Compiling - packetinfo.c for i386
Compiling - vpn.c for i386
Compiling - wdmdriver.c for i386
Compiling - passthru.c for i386
Compiling - des.c for i386
Linking Executable - objfre\i386\passthru.sys for i386
BUILD: Done

    12 files compiled
    1 executable built - 25 Warnings
```

图 3 IMD 编译结果

```
0.00060203 * * * e:\论文\vpn\vpn1\receive.c(186) * * *
0.00060622 = = = >VPN - PtReceiveRePacket...
0.00065595 * * * e:\论文\vpn\vpn1\receive.c(293) * * *
0.00066070 < = = = VPN - PtReceiveRePacket0.00066573 * * * e:
\论文\vpn\vpn1\receive.c(347) * * * 0.00066964 < = = =
VPN - PtReceive
0.00067523 * * * e:\论文\vpn\vpn1\receive.c(373) * * *
0.00067970 = = = >VPN - PtReceiveComplete
0.00068528 * * * e:\论文\vpn\vpn1\receive.c(407) * * *
0.00068947 < = = = VPN - PtReceiveComplete
0.01641913 * * * e:\论文\vpn\vpn1\miniport.c(174) * * *
```

图 4 IMD 输出结果

参考文献:

[1] Microsoft Corporation. Windows 2000 DDK [M]. US: Microsoft Corporation, 1985 - 2000.

[2] 朱雁辉. WINDOWS 防火墙与网络封包截获技术[M]. 北京:电子工业出版社, 2002: 210 - 228.

[3] 李 凌. Winsock2 网络编程实用教程[M]. 北京:清华大学出版社, 2003: 30 - 38.

[4] 蔡自兴,徐光祐. 人工智能及其应用[M]. 第 4 版. 北京:清华大学出版社, 2004: 311 - 316.

[5] Slattery T, Burton B. Advanced IP Routing in Cisco Networks [M]. Second Edition. Beijing: China Machine Press, 2001: 7 - 25.

[6] Microsoft Corporation. Windows XP DDK [M]. US: Microsoft Corporation, 2001.