Design and implementation of a personal firewall Based on NDIS Intermediate Drivers

He chaokai
Nanjing university of posts & communications
chaokaih@gmail.com

Abstract

Firewall is the core technology of today's network security and the first line of defense against external network attacks and threats. most personal firewall deals with the packets under the user model, there are a lots of limits. in order to protect our privates better some operations need to be done under the kernel model. The NDIS by which we can easy do something under the kernel model is introduced in the windows operating system that is the mostly used system in personal computer. This paper clearly describes the architecture of the NDIS, on the base of the NDIS intermediate drivers presents a personal firewall model which operations under the kernel model.

1. Introduction

The importance of network security has been significantly increasing in the past few years. In the global world of Internet, firewall is the core technology of today's network security and the first line of defense against external network attacks and threats. Firewall controls or governs network access by allowing or denying the incoming or outgoing network traffic according to firewall policy rules. These rules are explicitly written and managed to filter out any unwanted traffic coming into or going from the secure network.

A firewall is a security guard placed at the point of entry between a private network and the outside Internet so that all incoming and outgoing packets have to pass through it. A packet can be viewed as a tuple with a finite number of fields; examples of these fields are source/destination IP address, source/destination port number, and protocol type. By examining the values of these fields for each incoming and outgoing packet, a firewall accepts legitimate packets and discards illegitimate ones according to its configuration. A firewall configuration defines which packets are legitimate and which are illegitimate. An error in a firewall configuration means a wrong definition of being legitimate or illegitimate for some packets, which will either allow unauthorized access from the outside Internet to the private network, or disable some legitimate communication between the private network and the outside Internet. Neither case is desirable. So design a correct firewall configuration is therefore an important security issue.

2. The NDIS

2.1. Architecture of the NDIS

Microsoft and 3Com introduced the NDIS, which defines a standard interface that is used by low-level drivers to interact with layered models, such as a transmission control protocol/internet protocol (TCP/IP) model. The NDIS provides a pair of abstraction layers to connect the network drivers to an overlying protocol stack, such as a TCP/IP and an underlying network adapter. The NDIS performs a set of external functions for network adapter drivers, such as registering and intercepting hardware interrupts or communicating with underlying network adapters. In particular, the Windows library provides a fully standardized interface to implement a customized network adapter driver for the Windows Operating system. The library exports all of the Windows kernel-mode functions that are required for driver development. In addition, the Windows library file maintains binding and state information about all of the underlying network adapter drivers Fig. 1 shows the general NDIS architecture that is implemented in Windows-based platforms. The NDIS interface is located between an upper-level protocol driver, such as the TCP/IP protocol driver on the top of the communications architecture, the intermediate driver and miniport drivers in the middle of the communications architecture, and a network adapter at the bottom of the communications architecture. The NDIS architecture consists of several components:

0-7695-2909-7/07 $25.00 © 2007 IEEE
DOI 10.1109/SNPD.2007.233

IEEE computer society

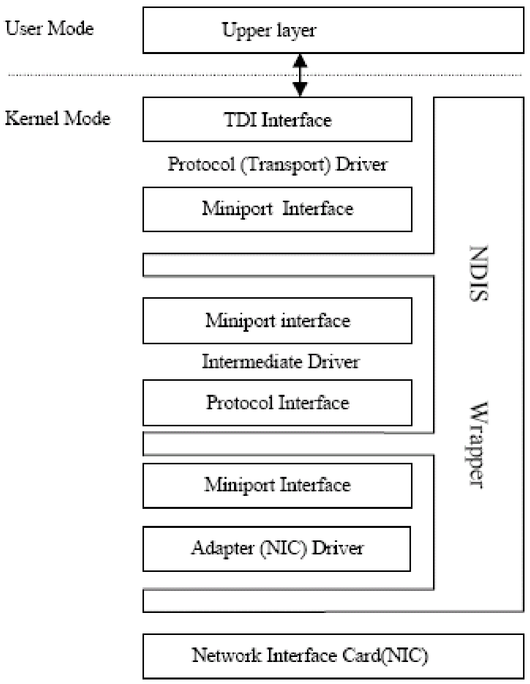


Fig.1. NDIS driver architecture

- The NDIS wrapper component provides the operating environment for drivers that use NDIS.
- The protocol drivers are implemented as applications using the Winsock application interface. This allocates packets, copies data from sending applications into packets, and sends these packets to the lower-level driver via the NDIS wrapper component. It also provides a lower-level interface specified by the NDIS to receive incoming packets from the miniport driver.
- The NIC miniport drivers manage network interface cards directly. They interface through the NDIS to the hardware at the lower level and to the upper layers to send packets, indicate received packets, reset the NIC, halt the NIC, and query or set the operation characteristics of the driver.

2.2. How the NDIS handles packets

At this point, it is important to describe briefly the way NDIS represents and handles packets. Each packet is identied by a packet descriptor (PD). The packet descriptor chains one or more buffer descriptors (BDs), each of them pointing to the memory area that contains the packet portion.

Packets may be indicated to upper protocols as either “full packets” (the lower layer indicates a packet in the form just described providing the PD as parameter), or “lookaheaded packets”(the lower layer provides a buffer containing the beginning part of a network packet, following the upper layer to request the transfer of the remaining portion if the network packet appears to contain relevant data for the upper protocol).

Each time a full packet based operation is initiated from within a certain layer, that layer actually relinquishes possession of its resources to the corresponding adjacent layer. These resources (PDs, BDs and memory areas) will be released almost immediately if the operation is executed synchronously, or at a later moment, through a completion or return operation triggered by the resource “borrower”, if the operation performs asynchronously.

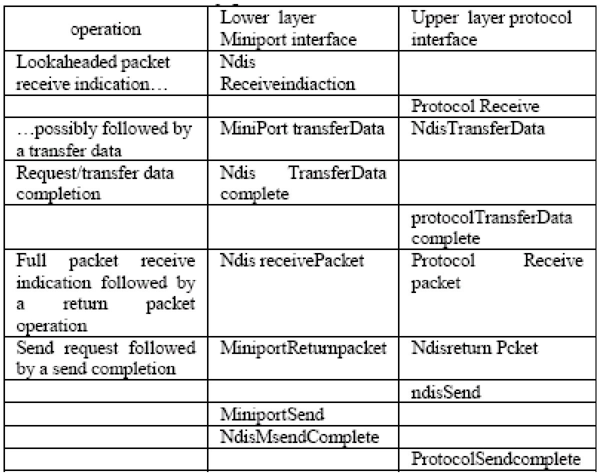
NDIS requires PDs not to cross more than one interface. This is why mipdrv should repackage the incoming full packets into fresh PDs before passing them to any of the djacent levels. The original PD should be stored within a reserved area of the newly allocated PD, in order to make sure that the original packet will be appropriately released at a later moment (as the operation completes).

When indicating lookaheaded packets, NDIS sets up a “receive context” in order to allow a likely transfer data request to retrieve the information in relation to the specific lookaheaded packet the operation refers to (usually, the receive context contains the address of the lower layer indicated lookahead buffer). The receiving context is provided as a parameter by the lower layer via the receive indication , and can be retrieved from the parameter list of the peer transfer data request.

Reserved areas within PDs and receive contexts are the necessary “bookmarks”required in an asynchronous, event driven environment. They provide for restoration of the context a certain operation was initiated within, in order to allow that operation to complete.

Table 1 provides a brief description of those primitives used for packet handling. It lists the NDIS calls and the correspondent entry points of the adjacent layer invoked by the NDIS wrapper as a result of the NDIS calls (the time axis should be imagined as vertically increasing).

Table 1. The correspondence between the NDIS Primitives and the Interface entry points



operation	Lower layer Miniport interface	Upper layer protocol interface
Lookaheaded packet receive indication…	Ndis Receiveindiaction	
		Protocol Receive
…possibly followed by a transfer data	MiniPort transferData	NdisTransferData
Request/transfer data completion	Ndis TransferData complete	
		protocolTransferData complete
Full packet receive indication followed by a return packet operation	Ndis receivePacket	Protocol Receive packet
Send request followed by a send completion	MiniportReturnpacket	Ndisreturn Pcket
		ndisSend
	MiniportSend	
	NdisMsendComplete	
		ProtocolSendcomplete

Now, considering the packet return operation, at the time a packet is returned, mipdrv has no possibility of detecting whether that packet was originated by itself or whether the packet was just used for relaying a NIC driver originated packet. The same situation applies to the transfer data completion: the actual initiator of the transfer data might have likewise been mipdrv or Tcpip.sys. This is why operation codes should be embedded into the packets (within the reserved area, along with the original packet descriptor).

When a packet is returned, mipdrv can decide, by inspecting the code, to either free up all the related resources, or, alternatively,to restore the old packet descriptor, free up its own packet descriptor and then return the original packet to the NIC driver that was the actual initiator.

Similarly, when a transfer data has been completed,mipdrv will know from the operation code whether the data has been transferred by the NIC driver, or whether the transfer data completion should be propagated to Tcpip.sys.

The interaction between various instances of protocol entities located at adjacent levels is managed via bindings.A binding represents the logical link between one instance of a protocol entity on one layer and one instance of protocol entity on the layer below. Thus, in a two NICed configuration, there will be two mipdrv instances, each one exporting at its top a virtual image of the underlying adapter driver instance on which it binds. Tcpip.sys will then bind on both mipdrv virtual adapters.

A binding can be accessed through a handler, and has an associated packet pool, a buffer pool and two more handlers provided by NDIS, which identify the miniport and the protocol interfaces. Additional parameters can be provided, such as the IP and physical addresses of the underlying NIC

3. A personal firewall based on the NDIS Intermediate drivers

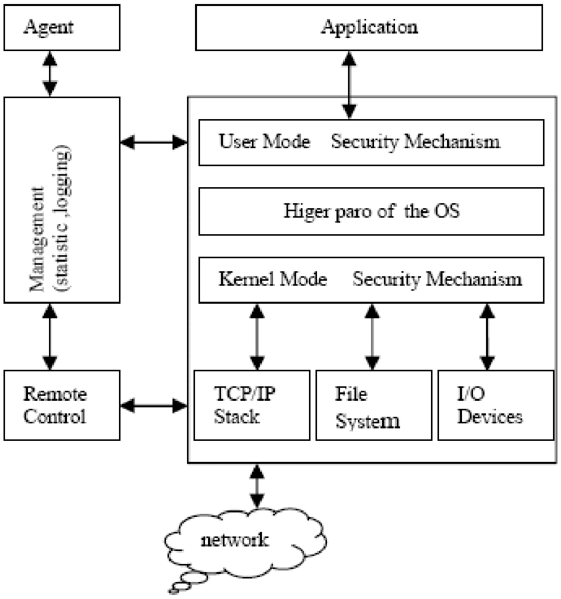


Fig.2. The components of a personal firewall

3.1 Personal firewall

The intermediate miniport drivers have the same upper-level interface as an NIC miniport driver. However, the lower level interfaces with an underlying device driver, such as the serial driver.

Most effective security tools that a network administrator can deploy to limit vulnerability. A firewall is hardware or software that the transmission of packets of digital information that attempt to pass through the perimeter of a network. A can be described as a system for enforcing access control policy between two networks. A firewall can deny unauthenticated request, with potential threats, while permitting authenticated requests, thus protecting the internal network. In most cases, firewalls are used to prevent outsiders from accessing an internal network. However, firewalls can also be used to guard one highly sensitive part of a private network against its other parts. Such sensitive parts are for payroll, payment processing, systems, There are essentially three types of firewalls: 1) filtering firewalls, 2) firewalls and 3) level firewalls A packet-filtering firewall drops packets based on their source or destination address or ports. For example, it blocks all packets from a site that we do not trust, or block all packets to a machine that should be inaccessible from external network. A filtering firewall works at the session layer of the OSI model, or the TCP layer of It monitors TCP handshaking between packets to determine whether a requested session is allowed. It creates a circuit then relays data between external and internal network. Stateful firewalls keep track of sessions and connection since internal state tables. It can protect against certain types of denial of service attacks. An example of this type of firewalls is In addition to lower levels, application-level firewalls inspect packets at the application level. A packet reaches the wall and is passed to an application

specific proxy, which inspects the validity of the packet. Newer types of firewalls include distributed firewalls and personal firewalls, In distributed firewalls, security policy is centrally defined but is enforced at network end points, such as routers, gateways, servers or user Policy can be distributed to endpoints in various forms. For example, policy can be pushed to end points where such policy should be enforced. Policy may also be provided to a drivers. It is thus possible to control all intrusion and contamination scenarios, which results in the maximum operating security and compatibility.

3.2 The Components of a personal firewall

A personal firewall consists of several Components, which are illustrated in the Fig 2:

The 'Agent' is the user interface to the personal firewall and provides the user with information on status. It controls defined accesses to the computer environment and to the resources.

The 'User Mode Security Mechanism' provides protection on a higher level, protecting the runtime environment in a user-oriented way.

The 'Kernel Mode Security Mechanism' provides extended security on a lower level and also protects the runtime environment. By integrating it into the Kernel level, maximum control can be achieved.

The 'Remote Control Module' provides the easy implementation of an organization's security policy by means of central security management.

In the 'Management Module', events are anylised, logged and statistics calculated. Besides that, the management module normally contains a cache and cookie management

A personal firewall can be configured offline or online. Offline configuration of the personal firewall is executed at the PC directly, whereas online configuration is executed centrally.

3.3 Security Components of a Personal Firewall

The firewall component 'Packet Filter interprets the packets and verifies whether the data contained in the packet headers corresponds with the defined communications rules. The rules are defined in such a way as to allow only the necessary communications and to avoid known security threatening settings. On each level, different aspects of the traffic can be checked in accordance with the communications rules. At the network access layer, source and destination addresses and protocol types are controlled.

At the network layer, the following aspects are checked depending on protocol:

- IP protocol: e.g. source and destination addresses, fields and flags
- ICMP: ICMP commands
- IPX protocol: e.g. network/node
- OS1 protocol: OS1 network address

At the transport layer, the following aspects are checked:

-UDP/TCP: e.g. port numbers (source and destination port - basis for the definition of services such as FTP, Telnet, HTTP)

- TCP: e.g. additional check of the direction of the connection setup.

It is also possible to check whether access over the packet filter took place within a defined period of time (e.g. Mondays to Fridays between 8am and 8pm; Saturdays between 7am and 7pm; Sundays, not at all).

The gathered information is taken from the set of rules and compared with the results of the analysis.

Sandbox Model

The sandbox model, which is typically defined in Java, is a concept where programs such as Java Applets, Active-X

Controls and other executables are executed under controlled conditions in an isolated area where they will not affect the rest of the system. This way, a potentially malicious application can only access the protected network (including the system, network files, resources and connected devices) if the personal firewall allows it, thus protecting all system resources against non-confidential, unknown or malicious applications. When the sandbox concept is in operation, it is even possible to protect the system against unknown threats.Display, Logging and Statistics Reporting on Security-Relevant Events

The PC user is alerted when active contents (Java Applets, Active-X Controls, etc.) are installed and/or running on the computer system. All suspicious activities are recorded in a logbook, and statistical information can be evaluated from these records.

4. Conclusion

We clearly describe the architecture of the NDIS, on the base of the NDIS intermediate drivers we present a personal firewall model which operations under the kernel model.

5. Reference

[1] J. W.Floroiu,T.C.Ionescu,R,Ruppelt,B.Henckel Mateescu,"Using NDIS intermediate drivers fro extending the protocol stack a case study", computer communications, 24(2001)703-715.

[2] Suk Lee, Kyoung Nam Ha, Jee Hun Park, Kyung Chang Lee,"NDIS-based virtual polling algorithm of IEEE 802.11b for guaranteeing the real-time equirements

[3] Willy Susilo,Russell James Ang,Cameron Allen Geroge McDonald, Jiangyong Hang, "Personal Firewall for Pocket PC2003: Design & Implementation", Proceedings of the 19th International Conference on Advanced Information Networking and Applications (AINA'05), 2005, pp. 450

[4] Ying QIU, Jianying ZHOU,Feng BAO, "MOBILE PERSONAL FIREWALL", 2000 s (ICDCS'04)

[5] Niklas Steinleitner, Henning Peters, Xiaoming Fu , "Implementation and Performance Study of a New NAT/Firewall Signaling Protocol", Proceedings of the 26th IEEE International Conference on Distributed Computing Systems Workshops (ICDCSW'06), 2006.

[6] Mohamed G. Gouda And ALEX X. Liu, "A Model of Stateful Firewalls and its Properties", Proceedings of the 2005 International Conference on Dependable Systems and Networks (DSN'05), 2005.

[7] Hong Han,Xian-Ling Lu, Li-Yong Ren,Bo Chen, "TAICHI: An open intrusion automatic response system based on plugin", Proceedings of the Fifth

[8] Charles Payne Tom Markham , "Architecture and Applications for a istributed Embedded Firewall"

[9] Mohamed G. Gouda and Xiang-Yang Alex Liu , "Firewall Design:Consistency, Completeness, and Compactness", Proceedings of the 24th International Conference on Distributed Computing Systems (ICDCS'04)

[10] Niklas Steinleitner, Henning Peters, Xiaoming Fu, "Implementation and Performance Study of a New NAT/Firewall Signaling Protocols", Proceedings of the 26th IEEE International Conference on Distributed Computing Systems Workshops (ICDCSW'06)