PART II

Screening Routers and Firewalls

CHAPTER 5

AN INTRODUCTION TO SCREENING ROUTERS

MANY COMMERCIAL routers provide the capability to screen packets based on criteria such as the type of protocol, the source address and destination address fields for a particular type of protocol, and control fields that are part of the protocol. Such routers are called *screening routers*. Screening routers can provide a powerful mechanism to control the type of network traffic that can exist on any network segment. By controlling the type of network traffic that can exist on a network segment, the screening routers can control the type of services that can exist on a network segment. Services that can compromise the network security can therefore be restricted.

This chapter examines how devices such as screening routers can be used to improve network security.

Clarifying Definitions

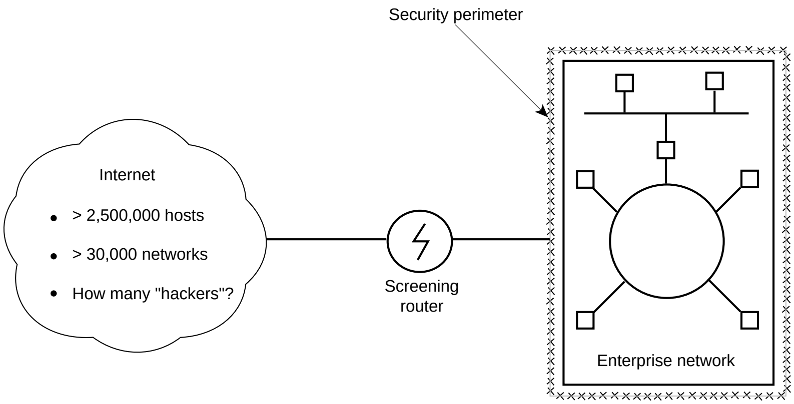
Screening routers can discriminate between network traffic based on the protocol type and the values of the protocol fields in the packet. The router's capability to discriminate between packets and restrict packets on its ports based on protocol-specific criteria is called *packet filtering*. For this reason, screening routers also are called *packet filter routers*.

Zones of Risk

An example of a packet filtering service implemented by a screening router is shown in figure 5.1. This figure shows an enterprise network connected to the Internet through a router that performs packet filtering.

Figure 5.1

A screening router forming a security perimeter.



The latest statistics on the Internet indicate that it consists of over 30,000 networks with a total of over 2.5 million hosts. With so many network users on the Internet, there is, unfortunately, a small segment of users who are malicious hackers. This situation is similar to moving to a large city that has its share of criminals. In this case, it is wise to protect your abode using locked doors. Prudence on your part also demands that if someone knocks on your door, you should have the ability to examine the person before allowing them entrance into your abode. Persons who appear to be harmful or look dangerous (high-security risks) should not be allowed entrance. In a similar manner, the screening router examines incoming packets to determine which of them could be potentially harmful.

In the network in figure 5.1, the enterprise network's boundary is called the *security perimeter*. Because malicious hackers abound on the Internet, it is useful to define a zone of risk. The *zone of risk* includes all TCP/IP-capable networks that are directly accessible through the Internet. *TCP/IP-capable* means that the host supports the TCP/IP protocol and its support protocols. *Directly accessible* means that there are no strong security measures (no "locked doors") between the Internet and hosts on your enterprise network.

From your point of view, the Internet's regional, national, and backbone networks represent a zone of risk. Hosts within the zone of risk are vulnerable to attacks. Placing your networks and hosts outside the zone of risk is highly desirable. However, without a device that can block attacks made against your network, the zone of risk will extend to your network. The screening router is one such device that can reduce the zone of risk so that it does not penetrate your network's security perimeter.

Not all hosts in your enterprise network may be TCP/IP-capable. Even so, these non-TCP/IP hosts can become vulnerable despite the fact that they are not technically part of the zone of risk. This can occur if the non-TCP/IP host is connected to the TCP/IP host. The intruder can use a protocol common to both the TCP/IP host and the non-TCP/IP host to access the non-TCP/IP host from the TCP/IP host (see fig. 5.2). If the hosts are on the same Ethernet segment, for example, the intruder can reach the non-TCP/IP host through the Ethernet protocol.

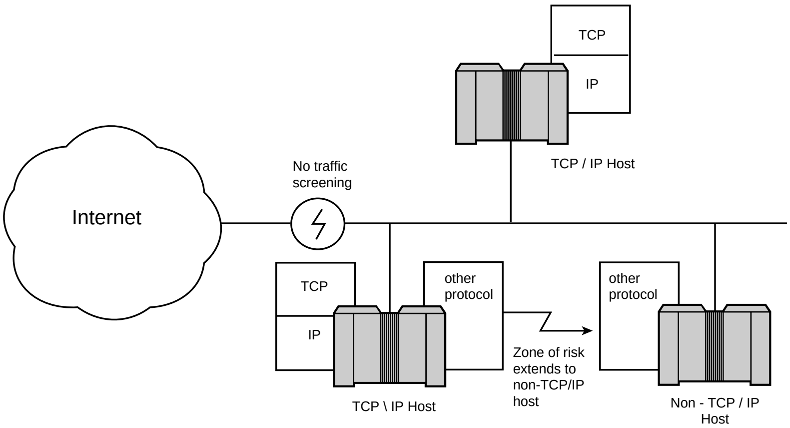


FIGURE 5.2

The zone of risk can extend to non-TCP/IP hosts.

Screening routers by themselves might not be able to eliminate the zone of risk. They can, however, be extremely effective in reducing the zone of risk.

THE OSI REFERENCE MODEL AND SCREENING ROUTERS

The term *router* in *screening router* implies a level of functionality dealing with routing issues. Understanding the overall role of the router in providing network communications helps you understand what types of filtering actions are provided by such routers. The preeminent model for understanding communication functions is the OSI Reference Model. This model can be used to describe the functionality of a router, and it helps explain the extent of the packet filtering capability of a screening router and its limitations.

The OSI Reference Model was developed in 1978 by the International Organization of Standards (ISO) to specify a standard that could be used for the development of open systems and as a yardstick to compare different communication systems. Network systems designed according to OSI framework and specifications speak the same language; that is, they use similar or compatible methods of communication. This type of network system enables systems from different vendors to interoperate.

In the early days of computer networks (prior to the OSI model), the proprietary computer network architecture reigned supreme. In those days, an organization interested in installing a computer network examined the choices available from vendors such as IBM, DEC, HP, Honeywell, and Sperry and Burroughs (now UNISYS). Each of those choices had its own proprietary architecture; the capability to interconnect networks from different vendors was almost nonexistent.

NOTE

> Despite the fact that OSI is a documented standard, TCP/IP is the de facto standard. OSI was largely unaccepted and even ridiculed by the IP community. While OSI was being evaluated and developed into a suite of protocols, TCP/IP was already accumulating momentum in the academic marketplace. Few applications today can actually use OSI—even X.400 e-mail can run on TCP/IP.

Once committed to buying equipment from a specific vendor, the organization was virtually locked in. Updates or modifications to the system were provided by the vendor, and because the vendor had a closed proprietary architecture, no one could compete with that vendor in supplying equivalent services. Prices were determined based on what the customer could bear without complaining too much!

Today's users probably realize that in many areas of the computer industry, this picture has not changed much. Proprietary architecture history is still around, but the OSI model can, at the very least, provide you with a clearer picture of how the different components of a network relate to each other.

Layers of the OSI Model

The OSI model has seven layers, as shown in figure 5.3. The layers, working from the bottom up, are as follows:

Physical
Data link
Network
Transport
Session
Presentation
Application

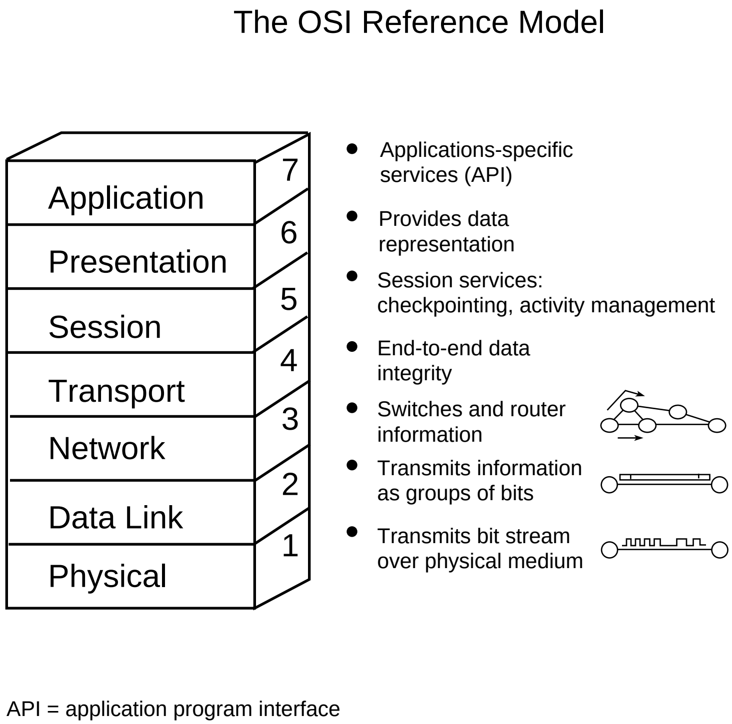


FIGURE 5.3

The OSI Reference Model (graphic courtesy of Learning Tree International).

The ISO applied five principles when structuring the layers of the model:

1. A layer should be created only when a different level of abstraction is needed.
2. Each layer should provide a well-defined function.
3. The function of each layer should define internationally standardized protocols.
4. The layer boundaries should minimize the information flow across layer interfaces.
5. Distinct functions should be defined in separate layers, but the number of layers should be small enough that the architecture does not become unwieldy.

The following is a summary of the functions of the seven layers.

Physical Layer

The *physical layer* deals with the mechanical, electrical, and procedural interfaces over the physical medium. The physical layer transmits bits over a communication channel. The bits might represent database records or file transfers; the physical layer is oblivious to what those bits represent. The bits can be encoded as digital 1s and 0s or in analog form.

In the case of the enterprise network of figure 5.1, the physical layer represents the cabling, physical ports, and attachments of the enterprise network. The physical connection of the screening router to the Internet, such as a leased line of the T1-link, also is part of the physical layer.

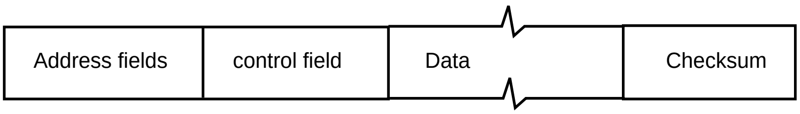
Data Link Layer

The *data link layer* builds on the transmission capability of the physical layer. The bits that are transmitted/received are grouped in logical units called a *frame*. In the context of LANs, a frame could be a token ring or Ethernet frame.

The bits in a frame have special meanings. The beginning and ending of a frame can be marked by special bit patterns. Additionally, the bits in a frame are divided into an address field, control field, data field, and error-control field. Figure 5.4 shows a typical data link frame.

Figure 5.4

A typical data link layer frame.



The *address field(s)* contains the sender and receiving node address. The *control field* is used to indicate the different types of data link frames, which include data frames and frames used for managing the data link channel. The *data field* contains the actual data being transmitted by the frame. The error control field usually detects errors in the data link frame. The data link layer also is the first layer in which you see error control concerns. The error control field usually is a hardware-generated checksum that is used to detect errors in the data link frame.

In figure 5.1, the screening router's Ethernet or token ring connection to the internal network is part of the data link layer. On the Internet port, the protocol used for the data link layer of the screening router could be SLIP, PPP, X.25, Frame Relay, and so on.

Network Layer

The *network layer* builds on the node-to-node connection provided by the data link layer by extending the node-to-node data link services across a network. An additional service provided by the network layer is how to route *packets* (units of information at the network layer) between nodes connected through an arbitrarily complex network.

Besides routing, the network layer helps eliminate congestion and regulate flow of data. The network layer also makes it possible for two networks to be interconnected by implementing a uniform addressing mechanism. Token ring or Ethernet LANs, for instance, have different types of data link addresses. To interconnect these two networks, you need a uniform addressing mechanism that can be understood by both token ring and Ethernet. For TCP/IP-based networks, this capability is provided by the Internet Protocol (IP).

The primary function of a router (including a screening router) resides in the network layer. When this router sees an IP packet, it examines the destination IP address in the packet. As discussed in Chapter 1, "Understanding TCP/IP," an IP address consists of a network number (also called netid) and a host number (also called hostid). The router examines the network number portion of the destination IP address in the IP packet and compares it to the entries in its routing table. If there is a match, the router forwards the IP packet as indicated in the routing table. If there is no match and there is no default route, the IP packet is rejected.

Screening routers can use criteria in addition to the routing table to forward the packet or reject it. The screening router can perform filtering at the network layer based upon the Source IP address, the Target IP address, and IP options (source routing or loose source routing). Figure 5.5 shows the fields within the IP packet on which the screening router can perform filtering, and figure 5.6 shows some of the IP options that can be used for filtering.

Transport Layer

The *transport layer* provides enhancements to the services of the network layer. This layer helps ensure reliable data delivery and end-to-end data integrity. To ensure reliable delivery, the transport layer builds on the error-control mechanisms provided by the lower layers. If the lower layers do not do a good enough job, the transport layer has to work harder. For this reason, this layer is also called the "last chance for error recovery" layer. In fact, when it comes to providing error-free delivery, you could say "The buck stops here" at the transport layer.

FIGURE 5.5

IP packet fields used by screening routers (graphic courtesy of Learnimg Tree International).

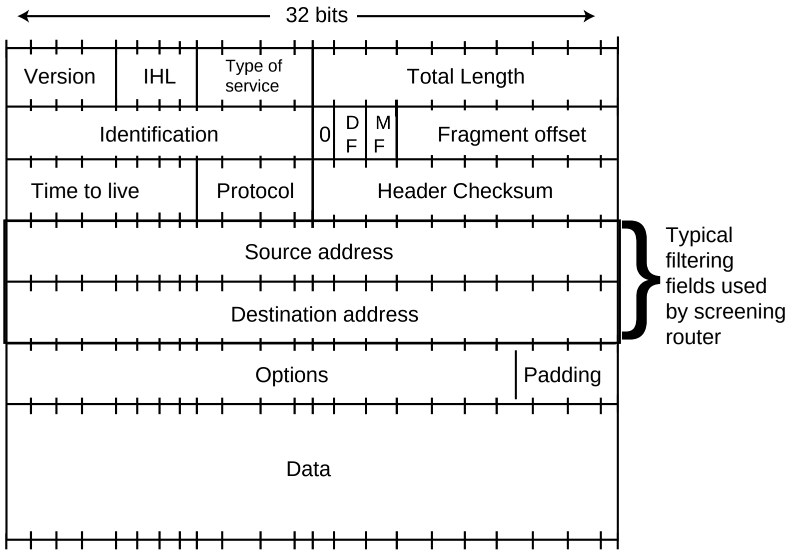


FIGURE 5.6

IP options used by screening routers (graphic courtesy of Learning Tree International).

IP Options Reference

Option Type	**Option Length**	**Option Value**
0	N/A	End of option list.
1	N/A	No operation.
2	11	Security. As required by DoD compartment, user group, handling restriction, label events.
3	variable	Loose source routing. Permits sender to specify *general* route followed by datagram.
4	variable	Timestamp. Permits a time trace of a datagram's *route* through the internet specified by sender.
7	variable	Record route. Traces path taken.
8	4	Stream ID. Permits routers to handle collections of IP datagrams in a similar way. Provides a tailored virtual circuit service.
9	variable	Strict source routing. Packet must *strictly* follow path.

The transport layer is the middle layer of the OSI model. The three lower layers constitute the network, and the three upper layers usually are implemented by networking software on the node. The transport layer usually is implemented on the node also; its job is to convert an unreliable subnet into a more reliable network.

The transport layer also can be responsible for creating several logical connections over the same network connection, a process called *multiplexing*. Multiplexing occurs when a number of transport connections share the same network connection.

Because of multiplexing, several software elements share the same network layer address. To uniquely identify the software elements within the transport layer, a more general form of addressing is necessary. These addresses, called *transport addresses,* usually are a combination of the network layer address and a transport Service Access Point (SAP) number. In TCP/IP, the term *port numbers* is used to identify transport addresses.

Routers normally do not perform processing at the transport layer. Screening routers, however, can examine port number fields in the TCP header. They can perform filtering decisions based on the port number values in the TCP header.

The source and destination port number fields used by screening routers are highlighted in the TCP header shown in figure 5.7. Many TCP/IP application services have standard port number values; for example, the FTP server uses port numbers 20 and 21. If you want to identify FTP sessions for the purposes of performing filtering, you can do so based on the port number values of 20 and 21. The source and destination port number are 16-bits, and can have values that range from 0 to 65535. The lower port numbers are assigned to Well Known Services (WKS) such as FTP, TELNET, and so on. The port numbers from 512 to 1024 usually are reserved for Unix-specific TCP/IP applications.

The port numbers are defined in an Internet Request for Comments (RFC) document that is updated on a frequent basis. As of the printing of this book, the current RFC for assigned numbers is RFC 1700. Most of the RFC documents are available on the World Wide Web at `http://www.cis.ohio-state.edu/htbin/rfc/rfc-index.html`.

Table 5.1 shows a few of the well-known port number assignments.

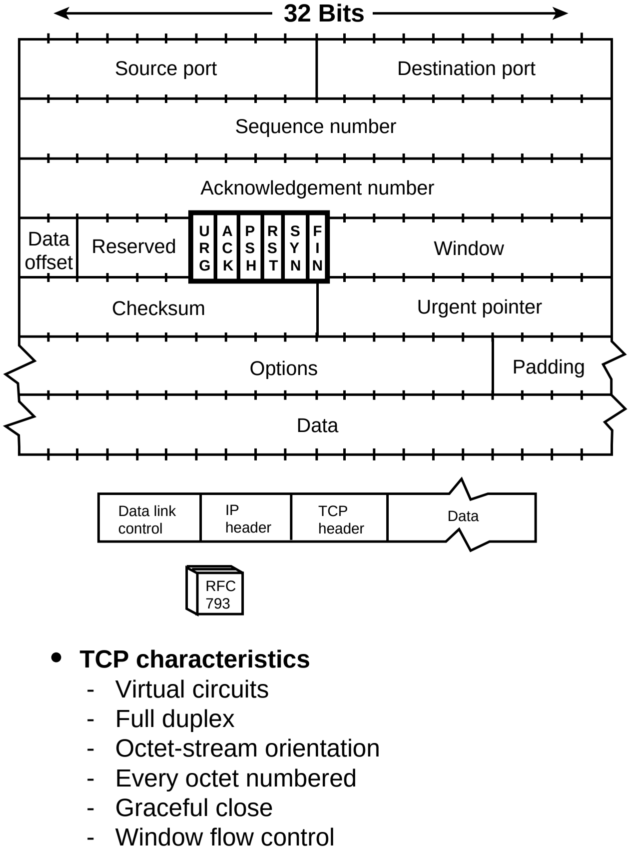


FIGURE 5.7

TCP fields used by screening routers (graphic courtesy of Learning Tree International).

TABLE 5.1

Some Well-Known TCP Port Numbers

Port Number	*Description*
0	Reserved
5	Remote Job Entry
7	Echo
9	Discard
11	Systat
13	Daytime
15	Netstat

Port Number	*Description*
17	Quotd (Quote of the day)
20	ftp_data
21	ftp (Control)
23	telnet
25	smtp (mail)
37	time
53	name server
70	Gopher protocol
79	Finger protocol
80	World Wide Web HTTP
88	Kerberos
102	ISO-TSAP
103	X.400
104	X.400 sending service
111	Sun RPC
123	Network Time Protocol (NTP)
139	NetBIOS session source
144	News
179	Border Gateway Protocol
512	exec
513	rlogin
514	rexec
515	lpd (line printer daemon)
517	talk
518	ntalk
2000	Open Windows (SUN)
X11	6000–6999

Besides port numbers, the screening routers are be able to filter packets based on the TCP flags. The TCP flags are used to indicate the type of TCP packet, such as the following:

- Open connection
- Acknowledgment of open connection
- Acknowledgment packet or data packet

The meanings of the TCP flags are shown in table 5.2. The most important TCP flags are the SYN and ACK flags. When the SYN flag is set to 1, it indicates that a TCP connection is being processed. This information is vital for designing proper filter rules for screening routers. If you want to prevent a TCP connection to an FTP service, for example, the screening router can reject packets to the FTP control port 21 that has the SYN flag set. The advantage of this is that the FTP connection is cut off before the client has the opportunity to reach a server.

TABLE 5.2
TCP Flags

TCP Flag	*Description*
URG	This flag is used to send out-of-band data without waiting for the receiver to process octets already in the stream. When the URG flag is set, the urgent pointer field is valid. RFC 1122 states that the urgent pointer points to the sequence number of the LAST octet (not LAST+1) in a sequence of urgent data, and that RFC 793 describes it incorrectly as LAST + 1. A TCP implementation must support a sequence of urgent data of any length. A TCP layer must inform the application layer asynchronously whenever the TCP layer receives an Urgent pointer with no previous pending urgent data, or whenever the urgent pointer advances in the data stream. There must be a way for the application to learn how much urgent data remains to be read from the connection, or at least to determine whether more urgent data remains to be read. Although the urgent mechanism can be used for any application, it normally is used to send interrupt-type commands to a Telnet program. The asynchronous, or `out-of-band`, notification enables the application to go into urgent mode, reading data from the TCP connection. This enables control commands to be sent to an application whose normal input buffers are full of unprocessed data.
ACK	The ACK flag indicates that the acknowledgment number field is valid.

TCP Flag	*Description*
PSH	This flag tells TCP to deliver data for this message immediately to the upper-layer process. When an application issues a series of send calls without setting the PSH flag, the TCP may aggregate the data internally without sending it. Similarly, when a series of segments is received without the PSH bit, a TCP may queue the data internally without passing it to the receiving application. The PSH bit is not a record marker and is independent of segment boundaries. Some implementations incorrectly think of the PSH as a record marker, however. The transmitter should collapse successive PSH bits when it packetizes data to send the largest possible segment. TCP can implement PSH flags on send calls. If PSH flags are not implemented, then the sending TCP must not buffer data indefinitely and must set the PSH bit in the last buffered segment (for example, when no more queued data is to be sent). RFC 793 erroneously implies that a received PSH flag must be passed to the application layer. Passing a received PSH flag to the application layer is optional. An application program is logically required to set the PSH flag in a send call whenever it needs to force delivery of the data to avoid a communication deadlock. A TCP should send a maximum-size segment whenever possible to improve performance, however. This means that on the sender side, a PSH may not result in the segment being immediately transmitted. When the PSH flag is not implemented on send TCP calls (or when the application/TCP interface uses a pure streaming model), responsibility for aggregating any tiny data fragments to form reasonable-size segments is partially borne by the application layer. Generally, an interactive application protocol must set the PSH flag at least in the last send call in each command or response sequence. A bulk transfer protocol like FTP should set the PSH flag on the last segment of a file, or when necessary to prevent buffer deadlock. At the receiver, the PSH bit forces buffered data to be delivered to the application (even if less than a full buffer is received). Conversely, the lack of a PSH can be used to avoid unnecessary wake-up calls to the application process; this can be an important performance optimization for large time-sharing hosts.
RST	The RST bit resets the virtual circuit due to unrecoverable errors. The reason could be a host crash or delayed duplicate SYN packets.

continues

TABLE 5.2, CONTINUED
TCP Flags

TCP Flag	*Description*
SYN	This flag indicates the opening of a virtual-circuit connection. TCP connections are opened using the three-way handshake procedure.
FIN	The FIN flag terminates the connection. Connection termination in TCP is accomplished by using a graceful close mechanism. Both sides must agree to terminate by sending a FIN = 1 flag before connection termination can occur; doing this ensures that data is not unexpectedly lost by either side by an abrupt connection termination.

When the SYN flag is set to 1, the ACK flag can be set to either 0 or 1. The settings of the ACK flag indicate whether the TCP packet is a connection or connection acknowledgment packet. If the SYN flag is set to 1 and the ACK flag is set to 1, the TCP packet is a connection acknowledgment packet. You can see this in figure 5.8 where a connection is made by a client on the Internet to a server on a local network. The server has a passive open connection and is waiting for a connection packet on a specified port. When the TCP connection packet is received by the server, it sends an open connection acknowledgment packet. The open connection packet has the following flag settings:

SYN = 1

ACK = 0

The open connection acknowledgment packet has the following settings:

SYN = 1

ACK = 1

When the client receives the open connection acknowledgment packet, it acknowledges this packet, completing the three-way handshake. The acknowledgment of the open connection acknowledgment packet has the following TCP flag settings:

SYN = 0

ACK = 1

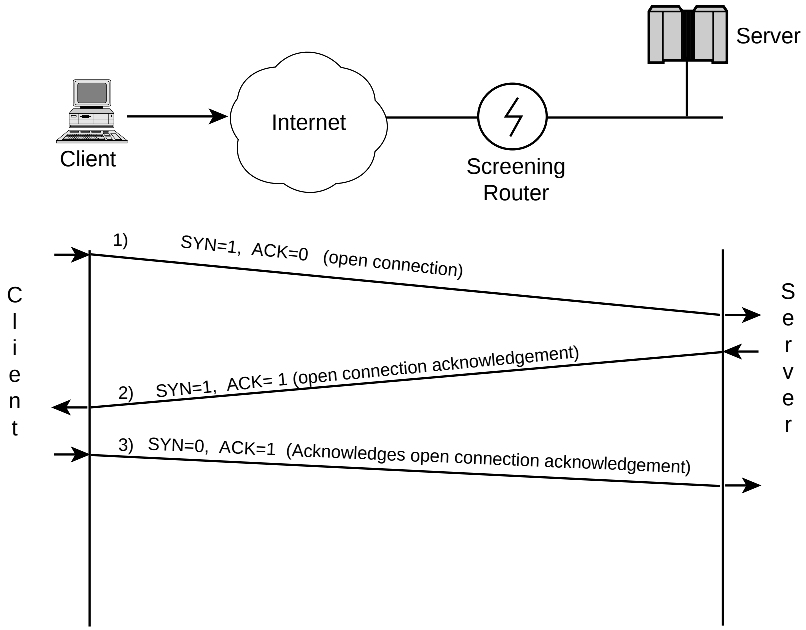


FIGURE 5.8

TCP open connection.

The three major combinations of SYN and ACK flag settings are summarized in table 5.3.

TABLE 5.3
SYN and ACK Flag Combinations

SYN Flag	*ACK Flag*	*Meaning*
1	0	Open connection TCP packet
1	1	Open connection acknowledgment
0	1	Acknowledgment packet or data packet

Figures 5.9 through 5.11 show the establishment of a TCP connection as previously described. They illustrate the SYN and ACK flags in a protocol decode of a TCP connection three-way handshake taken from a live TELNET connection. Figure 5.9 shows the open connection packet to the server, figure 5.10 shows the open connection acknowledgment, and figure 5.11 shows the acknowledgment to this open connection acknowledgment packet. Figure 5.11 shows that besides acknowledging the server connection acknowledgment, the client also sends application data that is part of the telnet protocol (option negotiation).

Thus, screening routers can perform packet filtering based upon the following:

- Source port number
- Destination port number
- TCP flags

Although screening routers can filter on any of the TCP flag settings, the flags that are most frequently used are the SYN and ACK flags. Not all router software has the capability to access the TCP flags. Consequently, such software has limited sensitivity to the SYN and ACK flags.

Besides TCP, another transport protocol that can be used is UDP (User Datagram Protocol). The User Datagram Protocol is a simpler transport protocol than TCP. It is popular in many LAN-oriented applications and is a connectionless transport protocol, meaning that it does not use a preestablished connection to transmit data.

Whereas the TCP protocol is responsible for reliable, simultaneous, full-duplex connections, the UDP protocol provides unreliable connectionless transport services. The term *reliable* means that TCP takes care of transmission errors by resending the portion of data that was in error. Any application that uses TCP does not have to be concerned with reliability of data transmission because this is handled by TCP. TCP provides for simultaneous connections. This means that several TCP connections could be established at a host over which data could be sent simultaneously independent of data on other connections. TCP provides full-duplex connections, which means that data can be sent and received on a single connection.

On the other hand, the UDP protocol is not as robust as TCP and can be used by applications that do not require the reliability of TCP at the host-to-host layer. It is called an *unreliable protocol* because the protocol does not guarantee the reliable delivery of a packet. It makes the best efforts it can to deliver the packet. Although the UDP protocol does provide an optional checksum for data integrity, unlike TCP it does not guarantee the sequenced delivery of the packets. *Sequenced delivery* means that packets are received in the order in which they are sent. UDP cannot guarantee sequenced delivery because it has no provision for keeping track of sequence numbers for packets. UDP has less overhead compared to TCP. If additional reliability such as data being received in the order in which it was sent (sequenced delivery) is required, the application that uses UDP has to provide for it.

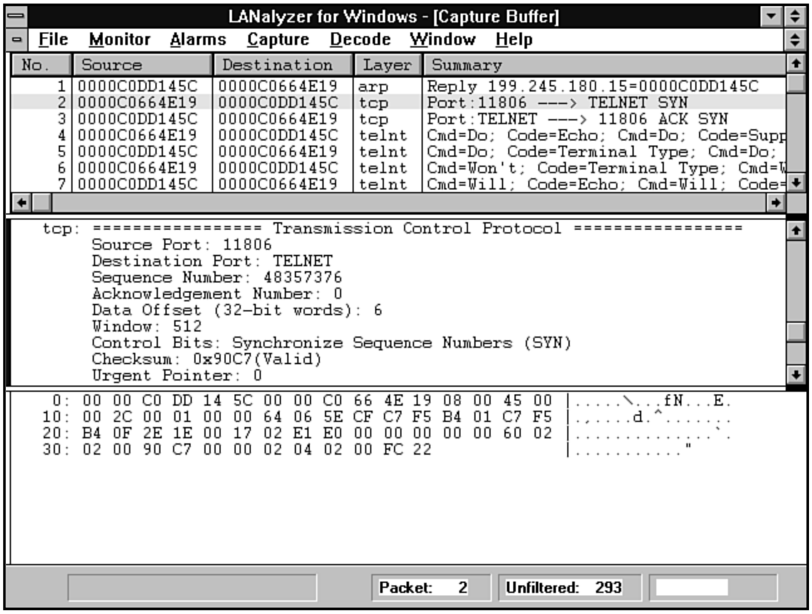


FIGURE 5.9

Three-way handshake: open connection TCP packet with SYN=1, ACK=0.

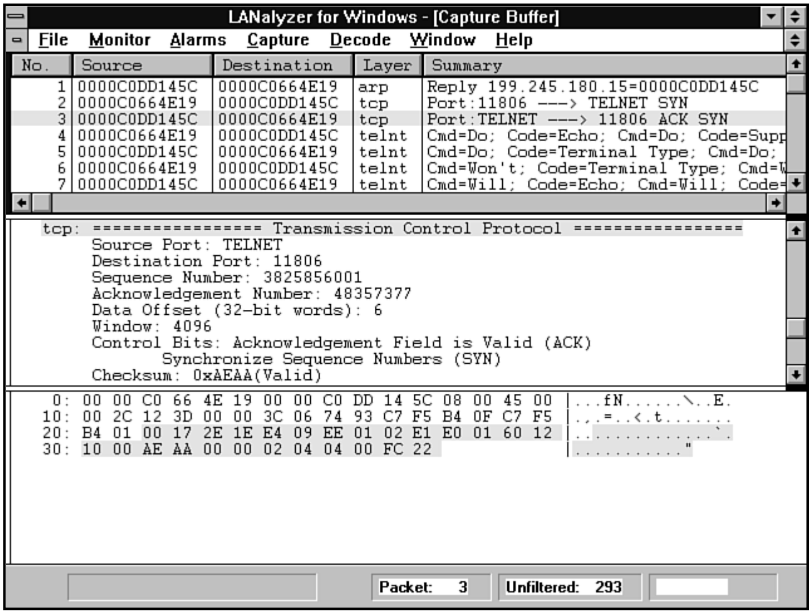


FIGURE 5.10

Three-way handshake: open connection acknowledgment TCP packet with SYN=1, ACK=1.

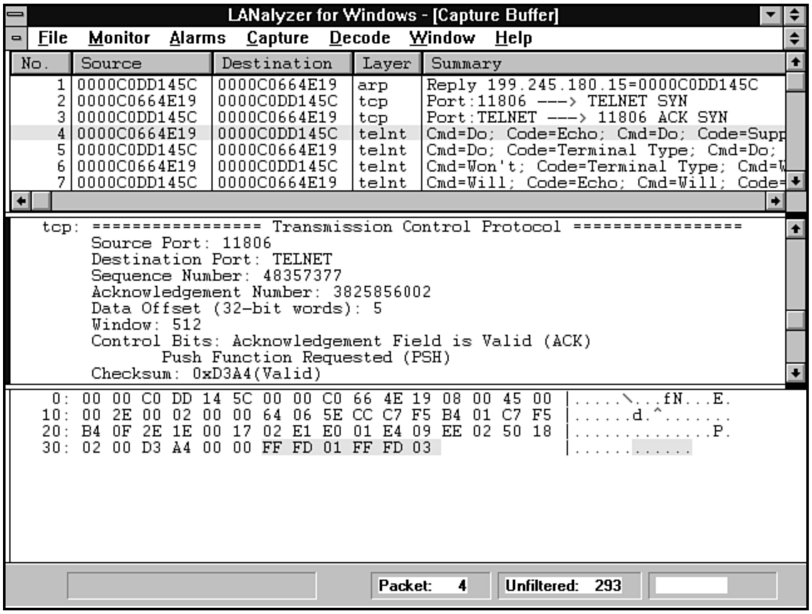


FIGURE 5.11

Three-way handshake: acknowledgment of open connection acknowledgment packet with SYN=0, ACK=1.

Figure 5.12 shows a UDP header. Notice that like the TCP packet in figure 5.7, the UDP header has source and port numbers. Because these port numbers refer to the UDP protocol that is distinct from the TCP protocol, these port numbers are in a different address space than the TCP port numbers. Some application services are available through both the TCP and UDP protocols, in which case they have port numbers defined for both TCP and UDP. Table 5.4 shows some of the examples for the UDP port numbers.

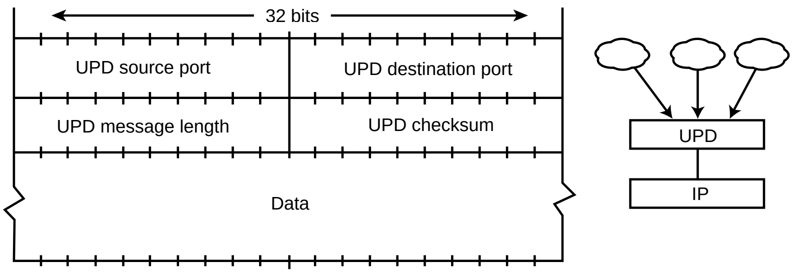


FIGURE 5.12

A UDP packet header (graphic courtesy of Learning Tree International).

- **Notable points**
  - No sequence or acknowledgment numbers
  - No flow control
  - Messages can be duplicated or arrive out of order
  - Checksum includes pseudoheader

Table 5.4
Example UDP Port Numbers

UDP Port Number	*Description*
0	Reserved
2	Management Utility
3	Compression Process
5	Remote Job Entry
7	Echo
9	Discard
11	Active Users (systat)
13	Daytime
17	Quote of the Day (QUOTD)
35	Any private printer server
37	Time
39	Resource Location Protocol
42	Host name server (nameserver)
43	Who Is (nicname)
49	Login Host Protocol (login)
52	XNS Time Protocol
53	Domain Name Server (domain)
54	XNS clearing house
66	Oracle SQL*NET (sql*net)
67	Bootstrap Protocol Server (bootps)
68	Bootstrap Protocol Client (bootpc)
69	Trivial Transfer Protocol (tftp)
80	World Wide Web HTTP

continues

Table 5.4, Continued
Example UDP Port Numbers

UDP Port Number	*Description*
88	Kerberos
94	Trivoli Object Dispatcher (objcall)
95	SUPDUP
108	SNA Gateway Access Server (snagas)
110	Post Office Protocol - Version 3 (POP3)
111	Sun Remote Procedure Call (sunrpc)
119	Network News Transfer Protocol (NNTP)
123	Network Time Protocol (NTP)
134	INGRES-NET Service
137	NETBIOS Naming Service (netbios-ns)
138	NETBIOS Datagram Service (netbios-dgm)
139	NETBIOS Session Service (netbios-ssn)
142	Britton-Lee IDM
161	SNMP
162	SNMP Traps
191	Prospero
194	Internet Relay Chat Protocol (irc)
201	AppleTalk Routing Maintenance (at-rtmp)
202	AppleTalk Name Binding (at-nbp)
213	IPX (used for IP tunneling)
215	Insignia (Soft PC)
217	dBASE Unix
372	Unix Listserv
513	Maintains database on who is logged on to machines on a local net and the load average of the machine (who)

UDP Port Number	*Description*
517	talk
518	ntalk
519	unixtime
525	Time Server (timed)
533	Emergency broadcasts (netwall)
556	RFS server (remoterfs)
565	Who Am I (whoami)
749	Kerberos Administration (kerberos-adm)
767	Phone (phonebook)
1025	Network Blackjack (blackjack)
1352	Lotus Notes (lotusnote)
2000	Open Windows
2049	Network File System (NFS)
6000 to 6999	X11
7000 to 7009	Used by Andrew File System (AFS)
17007	ISODE Directory User Agent (isode-dua)

The term *connectionless,* when applied to a protocol, means that the data can be sent without requiring that an IP data circuit be established. Each data unit is sent with complete source and destination IP addresses and port numbers that identify the application level processes that are involved in the data exchange. Connectionless IP focuses on its relationship with routing rather than the applications involved per se. This means that although the application port numbers are included in the packet, the application itself does not concern itself with routing—only with the exchange and processing of the data in the packet. UDP is similar to ordinary postal services in that complete addressing information is sent with each UDP message.

A big advantage that UDP has over TCP is that UDP is more suited for applications that require broadcast data. A single datagram can be broadcast on the network by specifying a broadcast address on the destination address. UDP is popular in many LAN-based applications that are broadcast-based and do not require the complexity of TCP.

Note

> Examples of applications that use UDP are NFS (Network File System), DNS (Domain Name System), SNMP (Simple Network Management Protocol), and TFTP (Trivial File Transfer Protocol). UDP also is used for IP tunneling.

Session Layer

The *session layer* of the OSI model makes use of the transport layer to provide enhanced session services. Examples of a session include a user being logged in to a host across a network, or a session being established for the purpose of transferring files.

The session layer can provide some of the following enhancements:

- Dialog control
- Token management
- Activity management

A session, in general, enables two-way communications (full duplex) across a connection. Some applications might require alternate one-way communications (half duplex). The session layer has the option of providing two-way or one-way communications, an option called *dialog control*.

For some protocols, it is essential that only one side attempt a critical operation at a time. To prevent both sides from attempting the same operation, a control mechanism, such as the use of tokens, can be implemented. When using the token method, only the side holding a token is permitted to perform the operation. Determining which side has the token and how it is transferred between the two sides is known as *token management*.

Note

> The use of the word *token* here should not be confused with token ring operation. Token management is a much higher level concept than token ring operation, residing at layer five of the OSI model. IBM's Token-Ring operation belongs to layers two and one of the OSI model.

If you are performing a one-hour file transfer between two machines, and network crashes occur approximately every 30 minutes, you might never be able to complete the file transfer. After each transfer aborts, you have to start all over again. To avoid this problem, you can treat the entire file transfer as a single activity with checkpoints inserted into the datastream. That way, if a crash occurs the session layer can synchronize to a previous checkpoint. This operation of managing an entire activity is called *activity management*.

> **Note**
>
> In the TCP/IP world, the RPC (Remote Procedure Call) protocol used by NFS (Network File System) can be considered an example of a session layer protocol.

Presentation Layer

The *presentation layer* manages the way data is represented. Many ways of representing data exist, such as ASCII and EBCDIC for text files, and 1s or 2s complement representation for numbers. If the two sides involved in communication use different data representations, they will not be able to understand each other. The presentation layer represents data with a common syntax and semantics. If all the nodes used and understood this common language, misunderstanding in data representation could be eliminated.

> **Note**
>
> An example of this common language is Abstract Syntax Notation, Rev 1 (ASN.1), an OSI recommendation. In the TCP/IP world, the ASN.1 is used for encoding SNMP messages. Another example of a protocol that corresponds to this layer is the External Data Representation (XDR) protocol that is used in the NFS.

Application Layer

The *application layer* contains the protocols and functions needed by user applications to perform communication tasks. Examples of common functions include the following:

- Protocols for providing remote file services, such as open, close, read, write, and shared access to files
- File transfer services and remote database access
- Message handling services for e-mail applications
- Global directory services to locate resources on a network
- A uniform way of handling a variety of system monitors and devices
- Remote job execution

Many of these services are called application programming interfaces (APIs). *APIs* are programming libraries that an application writer can use to write network applications.

Note

For TCP/IP, examples of application layer protocols are FTP, SMTP, TELNET, SNMP, NFS, and X Windows.

Firewalls used for protecting networks operate at the application layer of the OSI model. Because firewalls operate at the highest level of the OSI model, they have access to more information than screening routers and can be programmed to operate more intelligently than screening routers.

Screening Routers and Firewalls in Relationship to the OSI Model

Figure 5.13 compares screening routers and firewalls in relationship to the OSI model. This figure shows that the screening router functions primarily correspond to the network (IP protocol) and transport (TCP protocol) layers of the OSI model. Screening routers also can include the data link and physical layers, however, because most filtering systems apply to the type of interface, the network media in use, and even the MAC address itself. Firewalls often are described as gateways. Gateways can perform processing at all the seven layers of the OSI model. Typically, gateways perform processing at the seventh (application) layer of the OSI model. This is true for most firewall gateways.

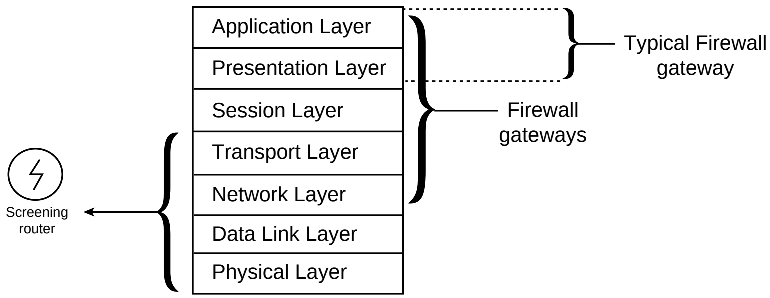


Figure 5.13

Screening routers, firewalls, and the OSI model.

Figure 5.13 also shows that because firewalls cover the network and transport layers, they can perform packet filtering functions. Some vendors, for marketing reasons perhaps, blur the distinction between a screening router and a firewall, to the extent that they call their screening router products firewall products. For the sake of clarity, this book makes the distinction between screening routers and firewalls based on the OSI model.

Sometimes screening routers are also called packet filter *gateways*. Perhaps one justification of the use of the term gateway for the packet filter device is that filtering based on the TCP flags done at the transport layer is not a function of the router that operates at the network layer of the OSI model. Devices that operate above the network layer also are called gateways.

You will learn in Chapter 8, "Firewall Architecture and Theory," how screening routers and firewall gateways can be combined to provide robust network security.

Understanding Packet Filtering

Screening routers can use packet filtering as a means to enhance network security. The screening function also can be performed by many commercial firewall products and by software-based products such as the Karlbridge PC-Based Filters. However, many commercial routers can be programmed to perform filtering. Router vendors such as Cisco, Wellfleet, 3COM, Digital, Newbridge, ACC, and many others provide routers that can be programmed to perform packet filtering functions.

Packet Filtering and Network Policy

Packet filtering can be used to implement a wide variety of network security policies. As discussed in Chapter 3, the network security policy must clearly state the types of resources and services that are being protected, their level of importance, and the people from whom the services are being protected.

Generally the network security policy guidelines are focused more in keeping outsiders out, than trying to police insiders. For example, it is more important to prevent outsiders from breaking in and intentionally exposing sensitive data or disrupting services than preventing insiders from using external network services. This type of network security policy determines where screening routers should be placed and how they should be programmed to perform packet filtering. Good network security implementations also should make it difficult for insiders to harm the network security. This usually is not the major thrust of security efforts.

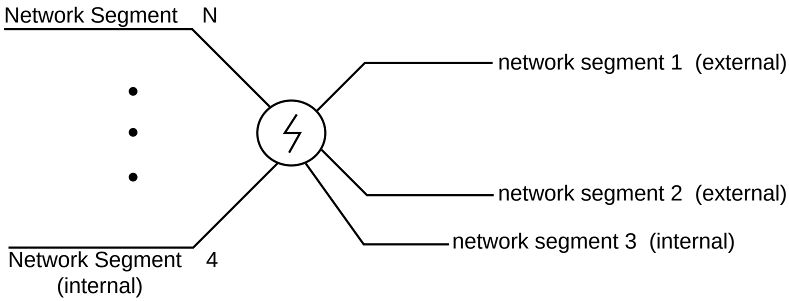
One of the goals of a network security policy is to provide a transparent mechanism so that the policy is not a hindrance to the users. Because packet filtering operates at the network and transport layers of the OSI model, and not at the application layer, this approach generally tends to be more transparent than the firewall approach. Recall that firewalls operate at the application layer of the OSI model, and security implementations at this layer tend not to be as transparent.

A SIMPLE MODEL FOR PACKET FILTERING

A packet filter usually is placed between one or more network segments, as shown in figure 5.14. The network segments are classified as either an external or internal network segment. *External network segments* connect your network to outside networks such as the Internet. *Internal network segments* are used to connect the enterprise's hosts and other network resources.

FIGURE 5.14

A packet filter placed between multiple segments.



Each of the ports of the packet filter device can be used to implement network policies that describe the type of network service that is accessible through the port. If the number of network segments that connect with the packet filter device is large, the policies that the packet filter device implements can become complex. In general, complex solutions to security problems should be avoided because of the following reasons:

- They are harder to maintain.
- It is easy to make mistakes in configuring packet filtering.
- They have an adverse effect on the performance of the device on which they are implemented.

Sheer economics, however, often dictate that a router with extra ports is purchased rather than several smaller routers. The advantages of a router that has several ports are scalability and the processing capacity of its CPU interface. Furthermore, because the packet filtering rules often are bound to one interface, a multiport router can be a manageable solution if designed properly.

In many instances, the simple model shown in figure 5.15 can be used to implement the network security policy. This model shows that the packet filter device has only two network segments connected to it. Typically, one of these network segments is an external network segment and the other is an internal network segment. Packet filtering is done to restrict the network traffic for the services that are to be denied. Because the network policy is written to favor insiders contacting external hosts, the filter on each side of the screening router's ports must behave differently. In other words, the filters are asymmetric.

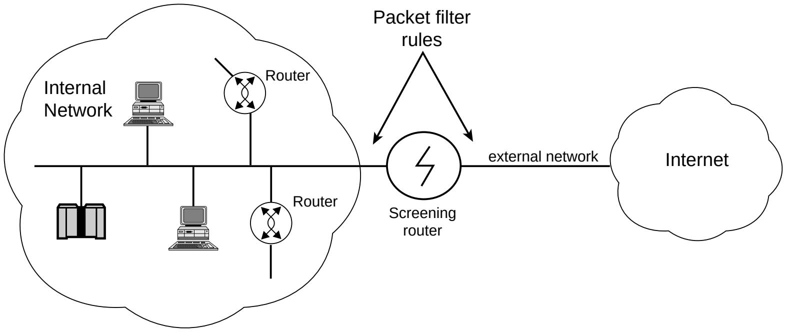


Figure 5.15
A packet filter placed between two network segments.

Packet Filter Operations

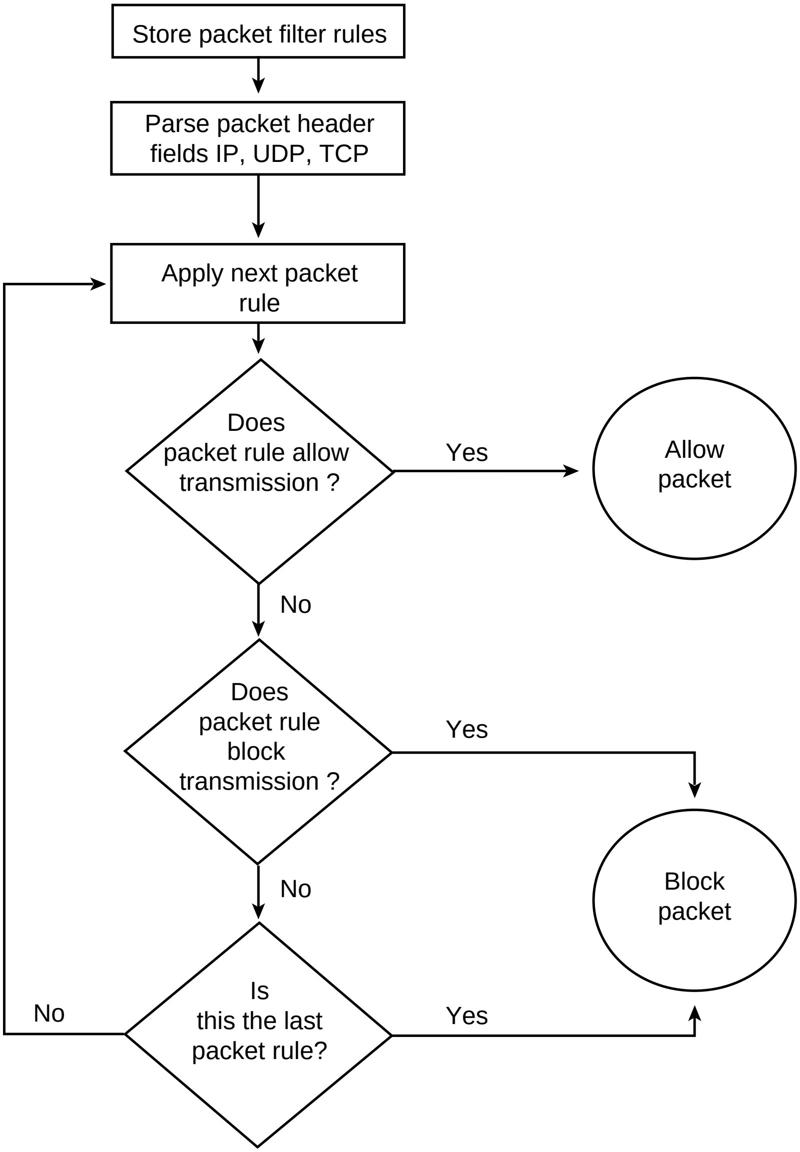
Almost all current packet filter devices (screening routers or packet filter gateways) operate in the following manner:

1. Packet filter criteria must be stored for the ports of the packet filter device. The packet filter criteria are called *packet filter rules.*
2. When the packet arrives at the port, the packet headers are parsed. Most packet filter devices examine the fields in only the IP, TCP, or UDP headers.
3. The packet filter rules are stored in a specific order. Each rule is applied to the packet in the order in which the packet filter rule is stored.
4. If a rule blocks the transmission or reception of a packet, the packet is not allowed.
5. If a rule allows the transmission or reception of a packet, the packet is allowed to proceed.
6. If a packet does not satisfy any rule, it is blocked.

These rules are expressed as a flowchart in figure 5.16.

Figure 5.16

A flowchart of packet filter operation.



From rules 4 and 5, you should realize that it is important to place the rules in the correct order. A common mistake in configuring packet filter rules is to place the rules in the wrong order. If the packet filter rules are placed in the wrong order, you might end up denying valid services, while permitting services that you wanted to deny.

Rule number 6 follows this philosophy:

That which is not expressly permitted is prohibited.

This is a fail-safe philosophy that you should follow when designing secure networks. It is the opposite of a permissive philosophy that says:

That which is not expressly prohibited is permitted.

If the latter philosophy were used for designing packet filters, you would have to think of every possible case not covered by the packet filter rules to make the network secure. And as new services are added, you easily can end up with situations in which no rule is matched. Rather than block this service and hear complaints from users if you have blocked a legitimate service (at which time you can unblock the service), you may end up allowing a service that can be a security risk to the network.

Packet Filter Design

Consider the network in figure 5.17 for which the screening router is used as the first line of defense between the internal protected network and an external untrusted network.

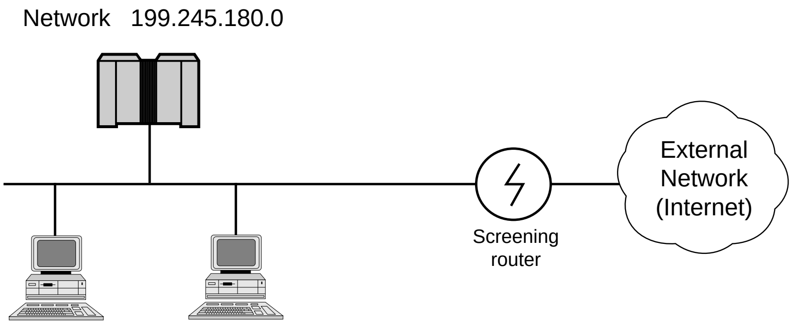


Figure 5.17

Sample network for packet filter design.

Assume that the network security policy requires that Internet mail be received from external hosts on a specific gateway, and you want to deny network traffic originating from the host named CREEPHOST that you do not trust. (Perhaps one reason for this could be that they have a tendency to send large messages that your mail cannot handle. Another could be that you suspect security threats originating from this host.)

In this example, the network security policy on SMTP use must be translated into packet filter rules. You could translate the network security rules into the following English language rules:

[Filter Rule 1]

We do not trust connections from host CREEPHOST.

[Filter Rule 2]

We wish to allow connections to our mail gateway.

These rules can be encoded into the table of rules shown in figure 5.18. The asterisk (*) symbol is used to match any values for that column.

For filter rule 1 (in figure 5.18) there is an entry for the External Host column, and all other columns have the asterisk symbol. The action is to block the connection. This translates to the following:

> Block any connection from CREEPHOST originating from *any* (asterisk) of its ports to *any* (asterisk) of our ports on *any* (asterisk) of our hosts.

For filter rule 2 there is an entry for the Our Host and the Port on Our Host columns. All other columns have the asterisk symbol. The action is to allow the connection. This translates to the following:

> Allow any connection from *any* (asterisk) external host originating from *any* (asterisk) of its ports to port 25 of on our MAIL-GW host.

Port 25 is used because this TCP port is reserved for SMTP (refer to table 5.1).

Figure 5.18

An attempt to encode packet filter rules.

Filter Rule Number	Action	Our Host	Portion on Our Host	External Host	Port on External Router	Description
1	Block	*	*	CREEPHOST	*	Block traffic from CREEPHOST
2	Allow	Mail-GW	25	2	*	Allow connection to our mail gateway
3	Allow	*	*	3	25	Allow outgoing SMTP traffic to remote mail gateway

Legend: * = Matches all values

The rules are applied in the order of their number in the table. If a packet does not match any of the rules, it is rejected.

A problem with the way the rules are specified in figure 5.18 is that it allows any external machine to originate a call from port 25. Port 25 should be reserved for SMTP, as per RFC 1700 (Assigned Numbers), but an external host could use this for other purposes. The third rule illustrates how an internal host would be able to send SMTP mail to an external host's port 25. This enables the internal host to send mail to external sites. If the external site is not using port 25 for SMTP, the SMTP-sender process will not be able to send mail. This is equivalent to mail not being supported on the external host.

As mentioned before, a TCP connection is a full duplex connection and information flows in both directions. The packet filter rules in figure 5.18 do not explicitly specify in which direction the information in the packet is being sent: from our host to the external site, or from an external site to our host.

When a TCP packet is sent in any direction, it must be acknowledged by the receiver. The receiver sends the acknowledgment by setting the ACK flag. Figure 5.8 shows the use of the ACK flag in acknowledging the TCP open connection requests. However, ACK flags are used in normal TCP transmission as illustrated in figure 5.19. In this figure, the sender sends a TCP segment (data sent by TCP is called a *segment*) whose starting byte number (SEQ#) is 1001 and length is 100 bytes. The receiver sends back a TCP acknowledgment packet indicated by the ACK flag set to 1, and the acknowledgment number (ACK#) set to 1001+100 = 1101. The sender then sends two TCP segment numbers that are 200 bytes each. These are acknowledged by a single acknowledgment packet that has the ACK flag set to 1, and acknowledgment number indicating the starting byte number of the next TCP data segment (1101 + 200 + 200 = 1501).

From figure 5.19, you can see that ACK packets will be sent on all TCP connections. When the ACK packet is sent, the sending direction is reversed, and the packet filter rules should take into account the ACK packets that are sent in response to control or data packets.

Based on the previous discussion, the modified set of packet rules can be written as indicated in figure 5.20.

For filter rule 1 (in figure 5.20) there is an entry for the Source Host/Net column of 199.245.180.0, and an entry in the Destination Host Port column of 25. All other columns have the asterisk symbol.

The action in filter rule 1 is to allow a connection. This translates to the following:

> Allow any connection from network 199.245.180.0 originating from *any* (asterisk) of its ports to port 25 on any (asterisk) destination host with any TCP flags or IP options set (including source routing).

Note that because 199.245.180 is a class C network number (also called the *netid*), the 0 in the host number (also called *hostid*) field refers to any host on the class C network 199.245.180.

For filter rule 2 there is an entry for the Source Host Port column of 25, an entry in the Destination Host column, and a TCP ACK entry in the TCP flags/IP options column. All other columns have the asterisk symbol.

FIGURE 5.19

Use of acknowledgments in TCP data transfer (graphic courtesy of Learning Tree International).

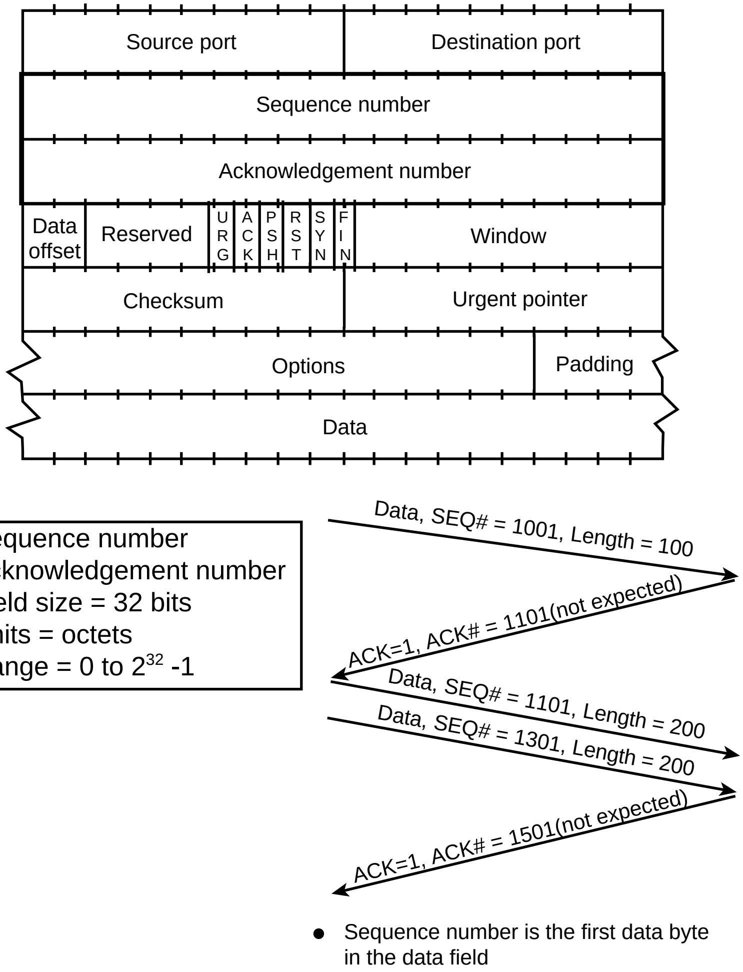


FIGURE 5.20

Packet filter rules for SMTP.

Filter Rule Number	Action	Source Host/Net	Source Host Port	Destination Host/Net	Destination Host Port	TCP Flags/ IP Optional	Description
1	Allow	199.245.180.0	*	*	25	*	Allow packet from network 199.245.180.0 to any destination host port 25
2	Allow	Mail-GW	25	199.245.180.0	*	ACK	Allow return acknowledgement

The action in filter rule 2 is to allow a connection. This translates to the following:

> Allow any connection to continue to be set up from *any* network originating from port 25 and which has the TCP ACK flag set to *any* (asterisk) port on any (asterisk) host on our network (199.245.180.0).

The combined effects of filter rules 1 and 2 of figure 5.20 is to allow TCP packets between the network 199.245.180.0 to the SMTP port on any external host.

Because the packet filter examines only layers 2 and 3 in the OSI model, there is no way to absolutely guarantee that the return TCP acknowledgments are part of the same connection. In actual practice, the scheme works well because TCP connections maintain state information on each side. They know what sequence numbers and acknowledgments to expect. Also, the upper layer application services, such as TELNET and SMTP, can accept only those packets that follow the application protocol rules. It is very difficult (though theoretically possible) to forge return replies that contain the correct ACK packets. For a higher level of security, one can use application-level gateways such as firewalls.

PACKET FILTER RULES AND FULL ASSOCIATIONS

Figure 5.21 shows a worksheet that can be used for designing packet filter rules. Screening routers, in general, can filter based upon any of the field values in the TCP or IP protocol headers. For most network security policies that can be implemented by screening routers, you need to specify only the TCP flags, IP options, and source and destination address values.

Filter Rule Number	Direction	Action	Source	Source Port	Destination	Dest. Port	Protocol Flags Options	Description
1								
2								
3								
4								
5								
6								
7								
8								

FIGURE 5.21

A worksheet for designing packet rules.

If you examine each row in the worksheet, you will notice that it completely describes the TCP connection. Formally, a complete description of a connection is called a *full association*.

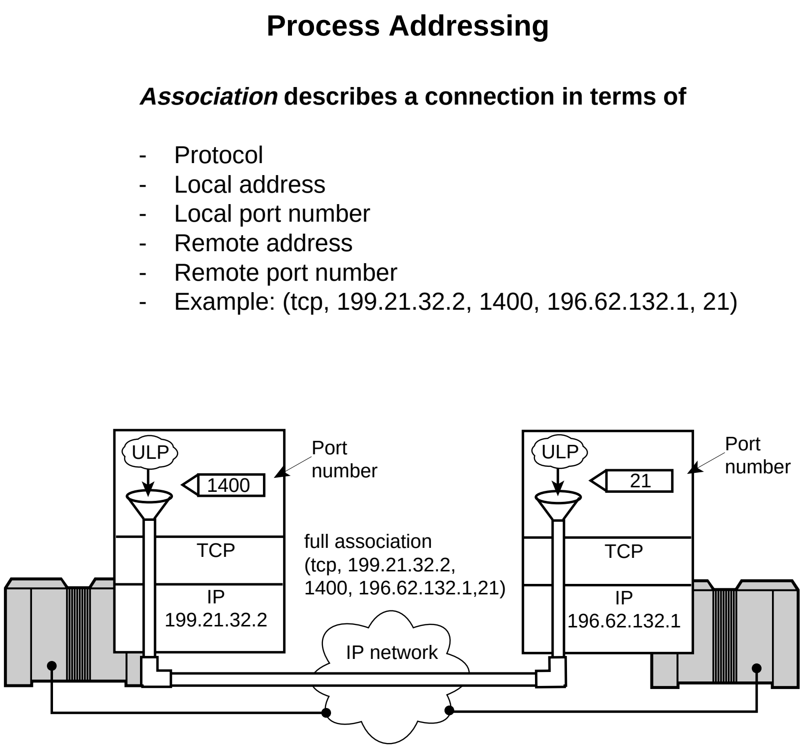
When designing packet filter rules it is helpful to keep in mind the definitions of full association, half association, and endpoints. This helps you better understand the packet filtering rules.

A *full association* is illustrated in figure 5.22, which shows that a TCP connection between two hosts can be described by the following information:

- Protocol type
- Local IP address
- Local TCP port number
- Remote IP address
- Remote TCP port number

Figure 5.22

A full association (graphic courtesy of Learning Tree International).



In figure 5.22, the protocol type is TCP, the local IP address is 199.21.32.2, the local TCP port number is 1400, the remote IP address is 196.62.132.1, and the remote TCP port number is port 21. The full association for this circuit is represented as a 5-tuple. In the example in figure 5.22, this 5-tuple is:

(TCP, 199.21.32.2, 1400, 196.62.132.1, 21)

Comparing this 5-tuple with the entries in worksheet 5.21, you can see that each worksheet describes the full association between hosts. The worksheet also contains the following additional information:

- Action
- TCP and IP options
- Order in which the rule should be executed

Each side of the connection can be described by a half association. A *half association* describes only one end of the connection and consists of the following:

- Protocol type
- IP address
- TCP port number

Thus, the two half associations that form the TCP connection of figure 5.22 are the following:

(TCP, 199.21.32.2, 1400)

(TCP, 196.62.132.1, 21)

The *endpoint,* also called the *transport address,* consists of the following:

- IP address
- TCP port number

The endpoints for the TCP connection in figure 5.22 are the following:

(199.21.32.2, 1400)

(196.62.132.1, 21)

Because each rule in the worksheet of figure 5.21 can have a range of values for any of the values in the fields that describe a full association, a number of different types of TCP circuits can be described by the packet filter rule. This enables a packet filter rule to implement a network security policy by describing a variety of different types of TCP connections.

Summary

This chapter introduced the concepts necessary to understand screening routers and packet filtering. The role of the packet filter was described in reference to the OSI model. You also learned about the typical protocol flags on which routing decisions are made. The next chapter follows this logic and discusses how packet filter rules are implemented and applied.

CHAPTER 6

PACKET FILTERS

THE LAST CHAPTER introduced packet filtering after explaining the information necessary to understand the process. This chapter continues that topic and discusses implementing packet filter rules, protocol-specific issues in packet filtering, and examples of screening router configurations.

To make the screening router configuration examples more practical, these discussions are done in the context of Cisco routers. These examples include defining access lists, using standard access lists, using extended access lists, filtering on incoming/outgoing calls, and IP security options for Cisco routers. You will learn about packet filter placement and filtering on input and output ports. This chapter discusses examples of filtering FTP network traffic and problems in filtering an FTP session.

Implementing Packet Filter Rules

Once you have designed the packet filter rules and described them in the packet filter rule worksheet (shown in figure 5.21 of the preceding chapter), you have to implement them on the screening router or firewall (if it allows packet filter rules to be specified).

Each type of packet filter device has its own set of rules and syntax on how to program the packet filter rules. Therefore, one must read the packet filter device documentation and learn the peculiarities of the packet filter rules syntax for that device. If you change the vendor of the packet filter device, you will have to learn a different set of syntax rules.

NOTE

> One of the goals of this book is to give you practical advice on building Internet firewalls and improving network security. Because of this, the author believes that it is important to show some practical examples of how packet filter rules can be specified. This will be done in relationship to the screening routers from the router vendor Cisco, Inc.

A word or two about the selection of the vendor is perhaps in order. Selection of the Cisco routers in the packet filtering examples is by no means an endorsement of the product. The author would be equally happy discussing another router vendor's product. On the other hand, Cisco dominates the router market and implements many of the packet filter capabilities discussed in this book. The packet filter rules for other vendors' routers are similar in principle to the ones used by Cisco routers, but are syntactically different.

Defining Access Lists

Cisco routers define *access lists* as a sequential collection of permit-and-deny conditions that apply to Internet addresses. These access-list conditions are used to implement the packet filter rules.

When the screening router is programmed with access lists, it tests the packets against the conditions in the access list one by one. The first match determines if the router accepts or rejects the packet. Because the screening router stops testing conditions in the access lists after the first match, the order of the conditions is critical. If no conditions are matched, the packet is rejected.

The Cisco routers have two types of access lists:

- Standard access lists
- Extended access lists

The standard access lists have a single address for matching operations, and the extended access lists have two addresses with optional protocol-type information for matching operations. For many practical filtering operations, you need both the standard and extended access lists.

Using Standard Access Lists

The syntax for the standard access lists is as follows:

```
access-list list {permit ¦ deny} address wildcard-mask
no access-list list
```

The *list* is an integer ranging from 1 to 99 and is used to identify one or more permit/deny conditions. The filter rules as defined in figure 5.21 of Chapter 5 are assigned to an access list. It is possible to assign each rule to its own access list, but this is highly inefficient and prone to errors. Each access list is associated with an interface on the router, such as a network interface, or the console. Access list 0 is predefined; it is the default list for all interfaces, and the only restrictions placed on the interface are what the router operating system will support.

The use of the keywords "permit" and "deny" corresponds to the words "allow" and "block" in the packet filter rules discussed earlier. The IP source address in the packet is compared to the *address* value specified in the access-list command. If the keyword "permit" is used, a match causes the packet to be accepted. If the keyword "deny" is used, a match causes the packet to be rejected.

The *address* and the *wildcard-mask* are 32-bit values and are written using the dotted-decimal notation. The *wildcard-mask* should not be confused with subnet masks that are used to subdivide an IP network number assignment. Address bits corresponding to a 1 in the *wildcard-mask* are ignored in the comparison. Address bits corresponding to a 0 in the *wildcard-mask* are used in the comparison. Consider the following example:

```
access-list 1 permit 199.245.180.0  0.0.0.255
access-list 1 permit 132.23.0.0  0.0.255.255
```

In this example, two address/wildcard-mask values are specified, and they both apply to the access-list number 1. The first access-list command permits access from hosts on the class C network 199.245.180.0, and the second access-list command permits access from hosts on the class B network 132.23.0.0.

If the wildcard-mask value is not specified, it is assumed to be 0.0.0.0; that is, all the bits in the address are compared. Thus, the following two access-list commands have an identical effect:

```
access-list 2 permit 132.23.1.3  0.0.0.0
access-list 2 permit 132.23.1.3
```

Both of the previous commands allow packets for the host with IP address 132.23.1.3 only. If the wildcard-mask value is non-zero, it can specify a range of IP addresses. Therefore, this should be done with care.

Perhaps another example of the use of standard access lists will clarify its use. Assume a class A network 67.0.0.0 connected to a screening router that is using a subnet mask of 255.255.0.0. Consider the following access-list commands:

```
access-list 3 permit 67.23.2.5  0.0.0.0
access-list 3 deny 67.23.0.0  0.0.255.255
access-list 3 permit 67.0.0.0  0.255.255.255
```

The first rule of the access list permits traffic for a single host with IP address 67.23.2.5 on subnet 23 of the class A network. The second access list blocks all traffic to subnet 23. Because this follows the rule that permits traffic to host 67.23.2.5, it does not affect traffic sent to that host. The third rule permits traffic sent to the entire class A 63.0.0.0 network. Therefore, the access lists implement the following network policy:

> "Block all traffic to subnet 0.23.0.0 for class A network 67.0.0.0, with the exception of allowing traffic to host 67.23.2.5 on this network. Allow traffic for all other subnets of 67.0.0.0."

You can use the "no access-list list" command to delete the entire access list, but use it with caution. If an incorrect access list is specified, you might be deleting something you want to keep. In fact, because of how access lists are entered into the router, if you want to make a change to a rule in the middle of the list, you must re-enter the entire list. Consequently, if you have to deal with a large set of access lists, it is easier to test them and then save them in a file so that later edits can be performed and then uploaded to the router.

WARNING

Access lists take effect immediately. If care is not taken, you can lock yourself out of the router, thereby making configuration and operation impossible.

USING EXTENDED ACCESS LISTS

The extended access lists enable you to filter interface traffic based on source and destination IP addresses and protocol information.

The syntax for the extended access lists is as follows:

```
access-list list {permit ¦ deny} protocol source source-mask destination
destination-mask [operator operand]
```

The *list* is an integer ranging from 100 to 199 and is used to identify one or more extended permit/deny conditions. The numbers 100 to 199 are reserved for extended access lists and are outside the range of the numbers 1 to 99 used for standard access lists.

If the keyword "permit" is used, a match with the condition causes the packet to be accepted. This is equivalent to the "allow" rule used in packet filter design rules. If the keyword "deny" is used, a match causes the packet to be rejected. This is equivalent to the "block" rule used in packet filter design rules. The rest of the extended list is not processed after a match occurs.

The *protocol* can represent any of the following values corresponding to the IP, TCP, UDP, and ICMP protocols:

- ip
- tcp
- udp
- icmp

Because IP encapsulates TCP, UDP, and ICMP packets, it can be used to match any of these protocols.

The *source* and the *source-mask* are 32-bit values and are written using the dotted-decimal notation. These are used to identify the source IP address. The *source-mask* should not be confused with subnet masks that are used to subdivide an IP network number assignment. Address bits corresponding to a 1 in the *source-mask* are ignored in the comparison. Address bits corresponding to a 0 in the *source-mask* are used in the comparison.

The *destination* and *destination-mask* are used for matching the destination IP address. These also are written using the dotted-decimal notation, and the *destination-mask* is used in the same way as the *source-mask* for *source* addresses.

The *operator* and *operand* are used to compare port numbers, service access points, or contact names. These values are meaningful for the TCP and UDP protocols. For the tcp and udp protocol key values, the *operator* can be any of the following values:

- lt (less than)
- eq (equal to)
- gt (greater than)
- neq (not equal to)

The *operand* is either a keyword or the decimal value of the destination port for the specified protocol. It also can consist of a range of values, enabling the access list rule to be effective over a range of ports. The following are examples of the use of access-list commands.

Example 1

Suppose network policy requires that you deny incoming SMTP connections from host 132.124.23.55 to your network 199.245.180.0. You can implement this policy by the following extended access list:

```
no access-list  101
access-list 101 any any
access-list 101 deny  tcp 132.124.23.55 0.0.0.0 199.245.180.0 0.0.0.255 eq 25
```

The first command deletes any prior extended access-list 101. The second command accepts any packet from any host. Without this command, the default action would be to deny all packets. The third command denies a TCP packet coming from host 132.124.23.55 to network 199.245.180.0 with a destination port of 25 (SMTP).

Example 2

In this example, the internal network is 133.34.0.0.

```
no access-list 101
access-list 101 permit tcp 0.0.0.0 255.255.255.255 133.34.0.0 0.0.255.255 gt
➥1023
access-list 101 permit tcp 0.0.0.0 255.255.255.255 133.34.12.3 0.0.0.0 eq 25
access-list 101 permit icmp 0.0.0.0 255.255.255.255 133.34.0.0 255.255.255.255
interface ethernet 0 out
ip access-group 101
```

The first access-list command deletes any existing access-list 101. The second access-list command permits any incoming TCP connections with destination ports greater than 1023. The second access-list command permits incoming TCP connections to the SMTP port of host 133.34.12.3. The last access-list command permits incoming ICMP messages for error feedback.

Note that the extended access-list command must be used with the access-group interface command that can take an extended access list number (100 to 199) as an argument. The access-group command is used to apply the access-list definitions to the interface. The syntax of the access-group command is as follows:

```
ip access-group list
```

where *list* is a number from 1 to 199 and specifies the access list to be applied to the interface.

Example 3

In this example, assume that you have a network (181.12.0.0) connected to the Internet. Your network security policy requires any host on the internal network to be able to form TCP connections to any host on the Internet. However, you do not want Internet hosts to be able to form TCP connections to hosts on the internal network, except to the mail (SMTP) port of a dedicated mail host (181.12.34.12).

The keyword "established" can be used for the TCP protocol to indicate an established connection. A match occurs if the TCP datagram has the ACK or RST bits set, which indicates that the packet belongs to an existing connection.

```
access-list 102 permit tcp 0.0.0.0 255.255.255.255 181.12.0.0 0.0.255.255
➥established
access-list 102 permit tcp 0.0.0.0 255.255.255.255 181.12.34.12 0.0.0.0 eq 25
interface ethernet 0
ip access-group 102
```

Filtering On Incoming and Outgoing Terminal Calls

To restrict incoming and outgoing connections between a line into the Cisco router and the addresses in an access list, you can use the access-class line configuration command.

```
access-class list {in ¦ out}
```

The *list* is the number of the access list. Using the value "in" at the end of the command restricts incoming traffic between the Cisco device and the addresses in the access list. Using the value "out" at the end of the command restricts outgoing traffic between the Cisco device and the addresses in the access list. To remove access restrictions for a specified access list, use the following command:

```
no access-class access-list-number {in ¦ out}
```

The following example defines an access list that permits only hosts on network 199.245.75.0 to connect to the virtual terminal ports 1 to 5 on the router:

```
access-list 18 permit 199.245.75.0 0.0.0.255
line 1 5
access-class 18 in
```

The following example blocks connections to networks other than network 156.233.0.0 on terminal lines 1 through 3:

```
access-list 19 permit 156.233.0.0 0.0.255.255
line 1 3
access-class 19 out
```

EXAMINING PACKET FILTER PLACEMENT AND ADDRESS SPOOFING

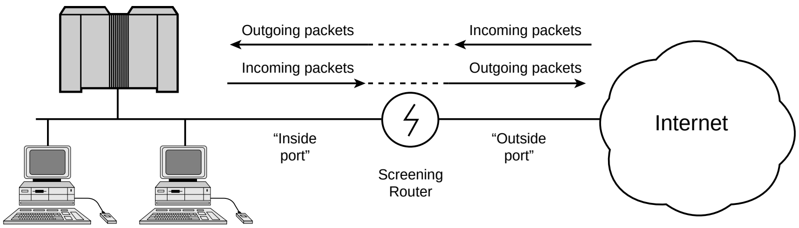
When designing packet filter rules, you must specify whether to perform packet filtering on incoming or outgoing packets. A related issue is determining where to place a packet filter. Packet filter rules should also be written in a manner that prevents address spoofing. These issues are examined in the sections that follow.

PACKET FILTER PLACEMENT

Consider the screening router of figure 6.1. The router can examine packets at any one of its interfaces. In the case of figure 6.1, the router can examine the packet traffic at either the inside port or outside port. Another factor to consider is the fact that packets can be either incoming or outgoing at any of the router's interfaces. The packet filtering, therefore, can be done on either incoming packets, outgoing packets, or both.

FIGURE 6.1

The placement of a packet filter.



Many router vendors implement packet filtering on outgoing packets for efficiency reasons. For outgoing packets, the filter rules can be applied when the router consults its tables to determine the destination of the packet. If the packet is not routable, or there is no match of the filter rules, the packet is rejected, and an ICMP `destination unreachable` message is sent.

If routers filter packets at the time the packets are being sent out of a router port, some information is lost. The router does not know which interface the packet arrived on. This can leave your network vulnerable to a type of attack known as *address spoofing*.

Consider the network in figure 6.2. A class B network 135.12.0.0 is connected to the Internet using a screening router. This class B internal network is using subnetting. The subnet mask is 255.255.255.0 for both subnets 10 and 11. An outside TCP/IP host sends a packet claiming to originate from the IP address 135.12.10.201. This packet is received by the screening router in its outside port. If the router was filtering incoming packets, it could quickly catch this pretender packet because it knows that the network 135.12.10.0 is connected to a different (inside) port, and therefore the packet could not have originated on its outside port. However, if the packet filtering is done on outgoing packets, the router does not check to see that this packet that was received on an outside port could not have originated from the internal network.

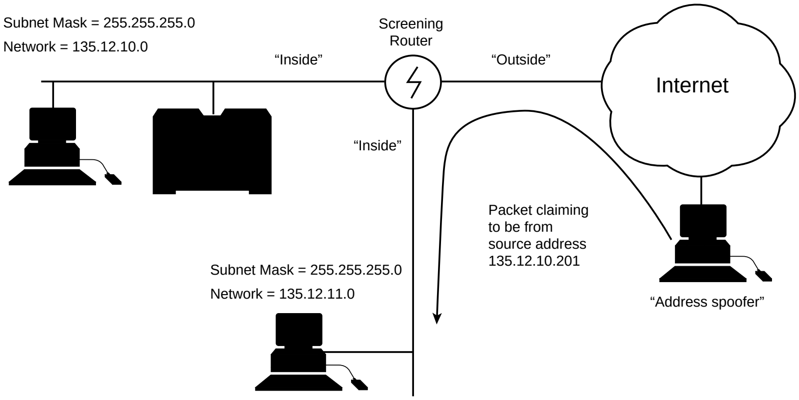


Figure 6.2
The placement of a packet filter.

Therefore, filtering on incoming packets can prevent a class of attack called address spoofing. In general, you should perform filtering as quickly as possible.

In figure 6.2, there are two inside ports and one outside port to the router. If all packets that are exchanged with the outside world are through this single access point, then the screening router can effectively act as a discriminating "choke" between the outside world (screening routers are also referred to as *chokes*) and the internal network.

Also in figure 6.2, if you wrote packet filter rules at the inside port for screening traffic between the internal network and the Internet, you would have to write a set of rules at each of the inside ports.

If the screening router has only two ports—one connecting to the external network and the other connecting to the internal network (see fig. 6.1)—then the packet filter rules are symmetric whether they are written for the inside or outside port. This is because the incoming packets on one port will appear, if not rejected by the router's routing table, as outgoing packets on the other port.

Filtering on Input and Output Ports

Not all routers can filter on both source and destination ports. Many routers filter on the destination port alone. The reason is that because TCP connections require data flow in both directions, the port on which you want to place the filter will appear as a destination port either when data is sent, or when the acknowledgment is received. However, because you are not able to supply source and destination ports simultaneously, this can cause problems.

Consider that your network security policy allows TCP connections for a custom application between an internal and external host. Assume that this custom service uses TCP port number 5555 on both sides. This situation is shown in figure 6.3. You can design the packet filter rule using the worksheet of figure 5.21 in the preceding chapter. This packet filter design table is shown in figure 6.4.

Figure 6.3

A custom application using port 5555.

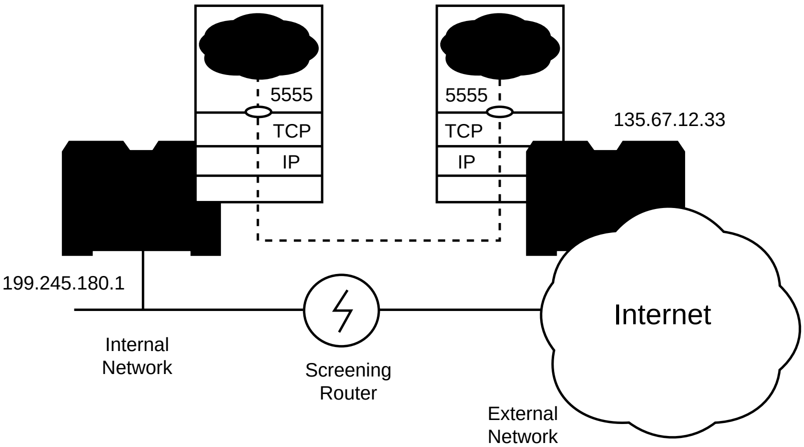


Figure 6.4

The packet filter rule design for custom application when source and destination ports can be specified.

Filter Rule Number	Action	Source	Source Port	Destination	Dest. Port	Protocol Flags Options	Description
1	allow	199.245.180.1	5555	135.67.12.33	5555	TCP	Allow TCP session at port 5555 on specified hosts
2							
3							

The table in figure 6.4 shows that only one packet filter rule is needed. However, if the screening router permits only the destination port to be specified, the single filter rule in the table in figure 6.4 has to be written as two rules as shown in figure 6.5.

If you examine figure 6.6, you can see that although the new rules encompass the rule specified in figure 6.5, they are a lot more permissive. The new rules allow a set of full associations that includes the single full association specified in figure 6.4. This is because the rules allow the following types of connections:

1. Connection from any of the internal hosts' ports to port 5555 on the external host
2. Connection from *any port on the external host* to port 5555 on the internal host

Filter Rule Number	Action	Source	Source Port	Destination	Dest. Port	Protocol Flags Options	Description
1	allow	199.245.180.1	*	135.67.12.33	5555	TCP	Allow TCP session to external host port 5555 from internal hosts
2	allow	135.67.12.33	*	199.245.180.1	5555	TCP	Allow TCP session to internal host port 5555 from external hosts
3							

Figure 6.5

The packet filter rule design for custom application when only destination port can be specified.

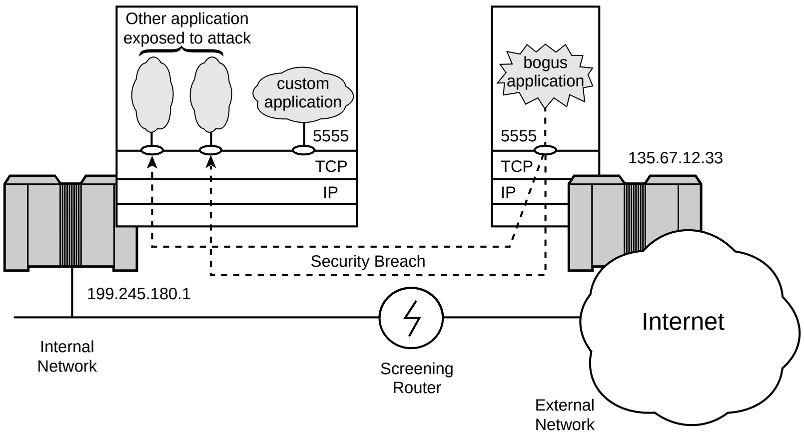


Figure 6.6

A bogus program invading an internal machine as a consequence of rules specified in figure 6.5.

Rule 1 in figure 6.5 is probably not very damaging because you usually can trust an internal host. Rule 2 in figure 6.5, however, can be a breach in network security. Consider the situation in figure 6.6, where a bogus program on the host can connect to any port on the internal host. This type of connection was not intended in the network security policy but is allowed from rule 2 of figure 6.6.

This is one example where it is not possible to achieve 100-percent security. If all of the security checks pass, then there is no way of knowing that the program is bogus. Consequently, the security policy and the router did their jobs. The end result, however, is that the programmer who wrote the program knew how to circumvent the policy and make it work to his or her advantage.

Therefore, to write packet filter rules effectively, screening routers should allow both source and destination ports to be specified in a single matched rule.

Examining Protocol-Specific Issues in Packet Filtering

When you design packet filter rules, you should understand the behavior of the application service that you are trying to filter. Some of the application services, such as FTP, require a call-back mechanism, in which an external host might need to initiate a connection to the internal host on a port that is not known at the time of specifying the packet filter rules.

The X11 protocol used in Unix X-Windows applications also requires an "incoming" call to an internal host from an external host. The internal hosts should be protected from these types of incoming calls. The sections that follow discuss the application services in relationship to packet filtering.

Filtering FTP Network Traffic

The following section discusses the normal behavior of FTP and points out the problem this behavior poses for screening routers. Solutions to solve packet filtering for FTP sessions are discussed.

Understanding the FTP Protocol

Figure 6.7 shows the FTP model. The FTP client makes a connection to the FTP server on the well-known Port 21 that is assigned to the FTP server. This connection is called the *control* connection. The control connection is used for sending FTP commands and receiving replies from the FTP server.

When a file is retrieved or stored on the FTP server, a separate data connection is established. This data connection is established on well-known Port 20 on the FTP server. The data connection exists only for the duration of the data transfer. It is destroyed at the end of the data transfer.

The different phases of an FTP transfer are discussed next for a live FTP session captured using a protocol analyzer. In this file transfer, the FTP client and FTP server have the following IP addresses and were connected on an Ethernet LAN:

FTP client: 199.245.180.1

FTP server: 199.245.180.15

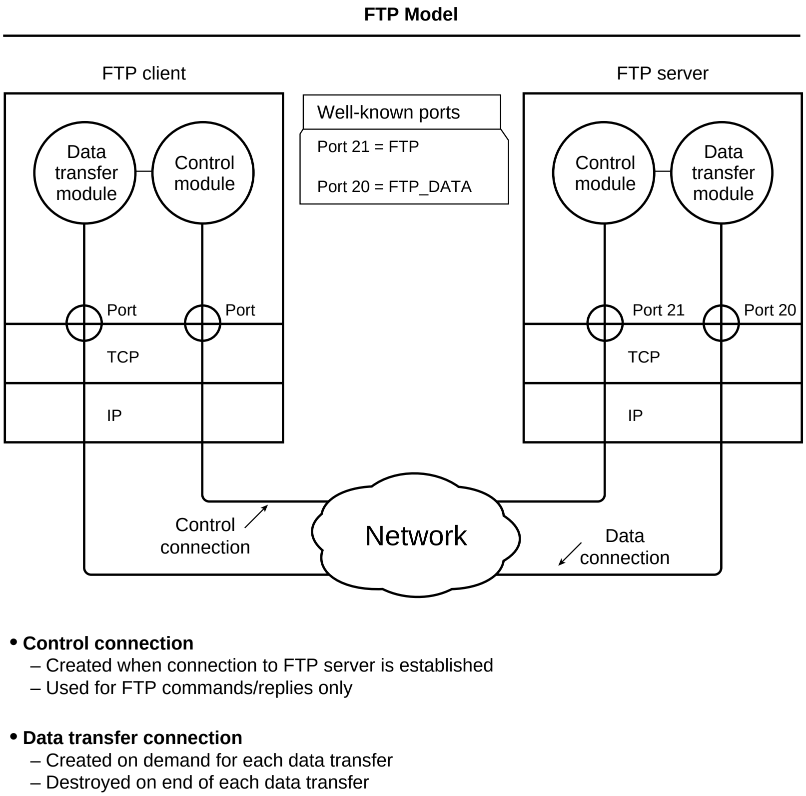


Figure 6.7
The FTP model (graphic courtesy of Learning Tree International).

Figures 6.8 and 6.9 show that the FTP session had 43 packets. Figure 6.8 shows packets 1 through 27, and figure 6.9 shows packets 17 through 43. All references to packet numbers in the discussion that follows is to the first column in these figures.

Packets 1-3:

Packet 1 is the FTP client (199.245.180.1) doing an ARP broadcast announcing its hardware address and IP address association.

Packet 2 is the FTP client doing an ARP request broadcast to discover the FTP server's hardware address.

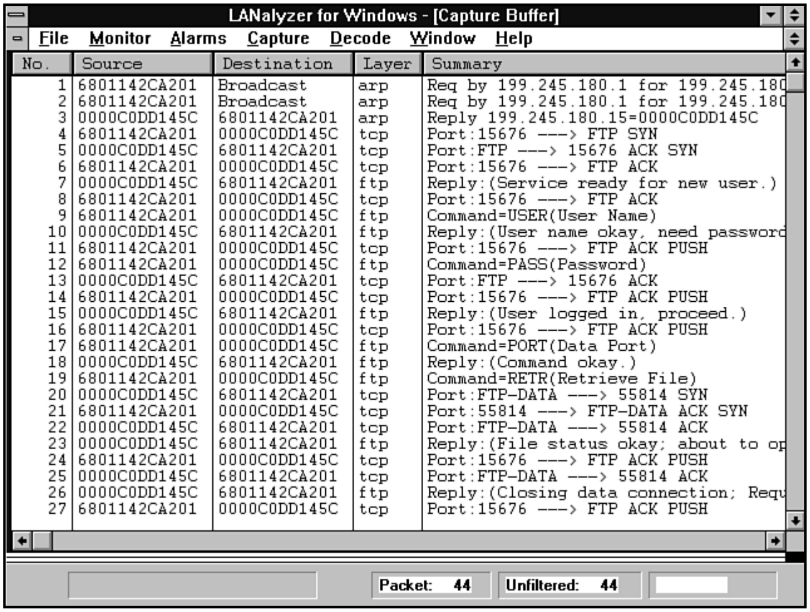
Packet 3 is the ARP reply from the FTP server. This ARP reply contains the server's hardware address (shown as 0000C0DD145C in figure 6.8).

Packets 4-6:

Packet 4 is the FTP client making an FTP control connection to the FTP server. The FTP client local-port number is 15676 and the FTP server port number is 21.

Figure 6.8

The FTP session packets (screen 1 of 2).



LANalyzer for Windows - [Capture Buffer]

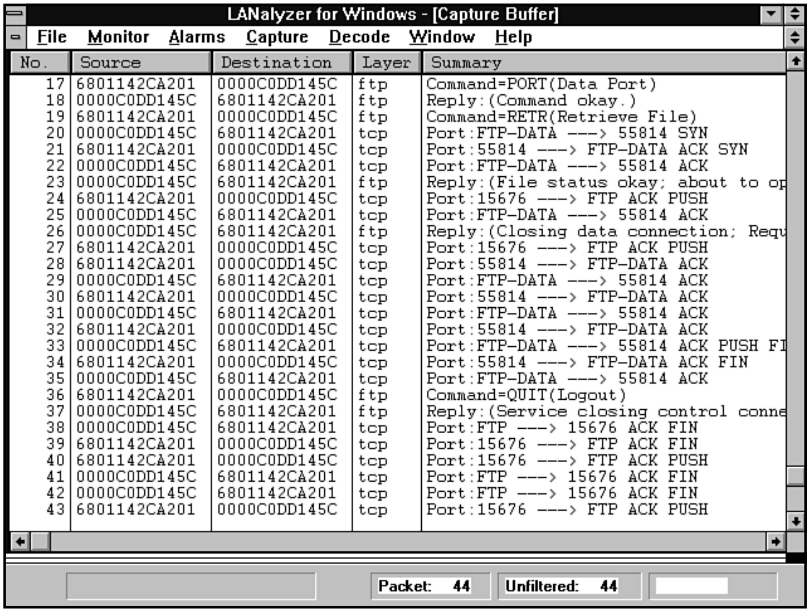
File Monitor Alarms Capture Decode Window Help

No.	Source	Destination	Layer	Summary
1	6801142CA201	Broadcast	arp	Req by 199.245.180.1 for 199.245.180
2	6801142CA201	Broadcast	arp	Req by 199.245.180.1 for 199.245.180
3	0000C0DD145C	6801142CA201	arp	Reply 199.245.180.15=0000C0DD145C
4	6801142CA201	0000C0DD145C	tcp	Port:15676 ---> FTP SYN
5	0000C0DD145C	6801142CA201	tcp	Port:FTP ---> 15676 ACK SYN
6	6801142CA201	0000C0DD145C	tcp	Port:15676 ---> FTP ACK
7	0000C0DD145C	6801142CA201	ftp	Reply:(Service ready for new user.)
8	6801142CA201	0000C0DD145C	tcp	Port:15676 ---> FTP ACK
9	6801142CA201	0000C0DD145C	ftp	Command=USER(User Name)
10	0000C0DD145C	6801142CA201	ftp	Reply:(User name okay, need password
11	6801142CA201	0000C0DD145C	tcp	Port:15676 ---> FTP ACK PUSH
12	6801142CA201	0000C0DD145C	ftp	Command=PASS(Password)
13	0000C0DD145C	6801142CA201	tcp	Port:FTP ---> 15676 ACK
14	6801142CA201	0000C0DD145C	tcp	Port:15676 ---> FTP ACK PUSH
15	0000C0DD145C	6801142CA201	ftp	Reply:(User logged in, proceed.)
16	6801142CA201	0000C0DD145C	tcp	Port:15676 ---> FTP ACK PUSH
17	6801142CA201	0000C0DD145C	ftp	Command=PORT(Data Port)
18	0000C0DD145C	6801142CA201	ftp	Reply:(Command okay.)
19	6801142CA201	0000C0DD145C	ftp	Command=RETR(Retrieve File)
20	0000C0DD145C	6801142CA201	tcp	Port:FTP-DATA ---> 55814 SYN
21	6801142CA201	0000C0DD145C	tcp	Port:55814 ---> FTP-DATA ACK SYN
22	0000C0DD145C	6801142CA201	tcp	Port:FTP-DATA ---> 55814 ACK
23	0000C0DD145C	6801142CA201	ftp	Reply:(File status okay; about to op
24	6801142CA201	0000C0DD145C	tcp	Port:15676 ---> FTP ACK PUSH
25	0000C0DD145C	6801142CA201	tcp	Port:FTP-DATA ---> 55814 ACK
26	0000C0DD145C	6801142CA201	ftp	Reply:(Closing data connection; Requ
27	6801142CA201	0000C0DD145C	tcp	Port:15676 ---> FTP ACK PUSH

Packet: 44 Unfiltered: 44

Figure 6.9

The FTP session packets (screen 2 of 2).



LANalyzer for Windows - [Capture Buffer]

File Monitor Alarms Capture Decode Window Help

No.	Source	Destination	Layer	Summary
17	6801142CA201	0000C0DD145C	ftp	Command=PORT(Data Port)
18	0000C0DD145C	6801142CA201	ftp	Reply:(Command okay.)
19	6801142CA201	0000C0DD145C	ftp	Command=RETR(Retrieve File)
20	0000C0DD145C	6801142CA201	tcp	Port:FTP-DATA ---> 55814 SYN
21	6801142CA201	0000C0DD145C	tcp	Port:55814 ---> FTP-DATA ACK SYN
22	0000C0DD145C	6801142CA201	tcp	Port:FTP-DATA ---> 55814 ACK
23	0000C0DD145C	6801142CA201	ftp	Reply:(File status okay; about to op
24	6801142CA201	0000C0DD145C	tcp	Port:15676 ---> FTP ACK PUSH
25	0000C0DD145C	6801142CA201	tcp	Port:FTP-DATA ---> 55814 ACK
26	0000C0DD145C	6801142CA201	ftp	Reply:(Closing data connection; Requ
27	6801142CA201	0000C0DD145C	tcp	Port:15676 ---> FTP ACK PUSH
28	6801142CA201	0000C0DD145C	tcp	Port:55814 ---> FTP-DATA ACK
29	0000C0DD145C	6801142CA201	tcp	Port:FTP-DATA ---> 55814 ACK
30	6801142CA201	0000C0DD145C	tcp	Port:55814 ---> FTP-DATA ACK
31	0000C0DD145C	6801142CA201	tcp	Port:FTP-DATA ---> 55814 ACK
32	6801142CA201	0000C0DD145C	tcp	Port:55814 ---> FTP-DATA ACK
33	0000C0DD145C	6801142CA201	tcp	Port:FTP-DATA ---> 55814 ACK PUSH FI
34	6801142CA201	0000C0DD145C	tcp	Port:55814 ---> FTP-DATA ACK FIN
35	0000C0DD145C	6801142CA201	tcp	Port:FTP-DATA ---> 55814 ACK
36	6801142CA201	0000C0DD145C	ftp	Command=QUIT(Logout)
37	0000C0DD145C	6801142CA201	ftp	Reply:(Service closing control conne
38	0000C0DD145C	6801142CA201	tcp	Port:FTP ---> 15676 ACK FIN
39	6801142CA201	0000C0DD145C	tcp	Port:15676 ---> FTP ACK FIN
40	6801142CA201	0000C0DD145C	tcp	Port:15676 ---> FTP ACK PUSH
41	0000C0DD145C	6801142CA201	tcp	Port:FTP ---> 15676 ACK FIN
42	0000C0DD145C	6801142CA201	tcp	Port:FTP ---> 15676 ACK FIN
43	6801142CA201	0000C0DD145C	tcp	Port:15676 ---> FTP ACK PUSH

Packet: 44 Unfiltered: 44

The protocol analyzer has translated this port number to the symbolic representation of “21” to “FTP” (see fig. 6.10). You can decode the hexadecimal dump of the packet, showing that the destination port has a value of 15(hex) or 21 decimal.

Packets 4 and 5 are useful when setting up packet filters. If your router supports creating filters to detect an open FTP connection request (Cisco does not), you can filter for the following conditions:

Destination port = 21.

TCP SYN flag is set.

TCP ACK flag is set in open-connection acknowledgment.

Packet 6 completes the three-way handshake. It is the acknowledgment packet that is sent in response to the open-connection acknowledgment packet.

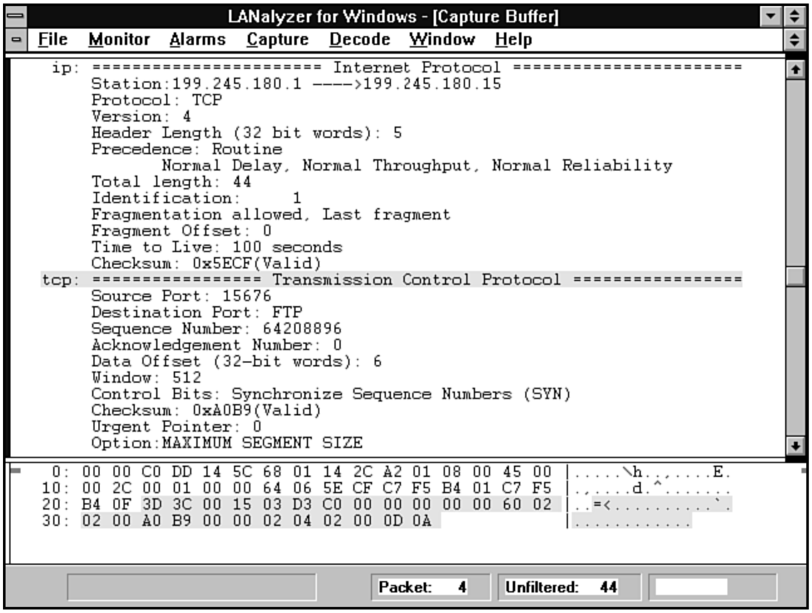


FIGURE 6.10

An FTP Open Control connection packet.

Packets 7-8:

Packet 7 is the reply from the FTP server. The FTP commands are sent using a four-character ASCII text command followed by parameters. The FTP reply is a three-digit decimal status code followed by an optional text message.

The FTP command and FTP reply syntax are shown in figure 6.11. This figure also summarizes the FTP session activities that have taken place so far. Figure 6.12 shows packet number 7 that contains the FTP reply code 220 that means "Service ready for new user." The text message that is sent for displaying at the FTP client console is `ucs FTP server (Version 5.60) ready.`

Packet 8 sent from the FTP client to the server acknowledges the server reply.

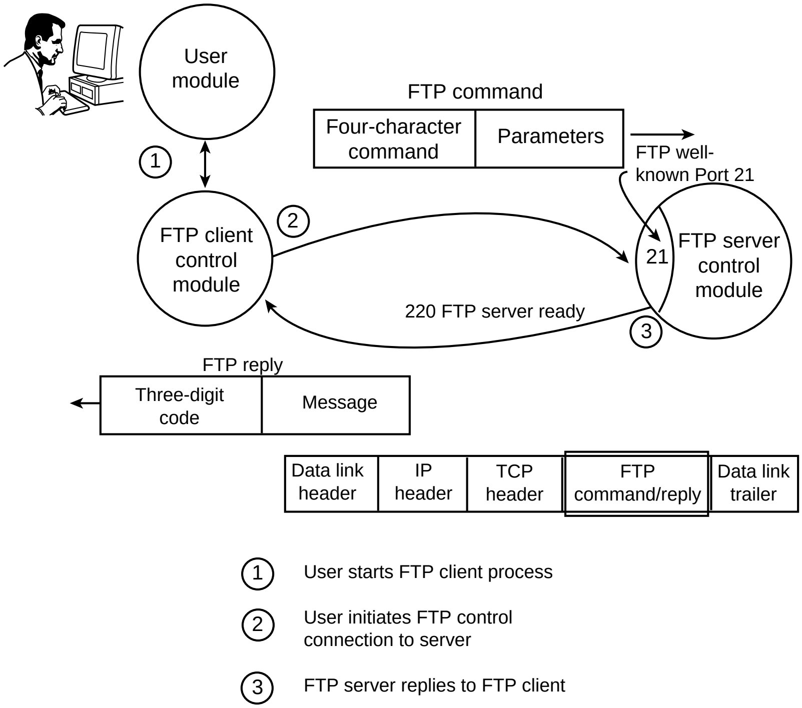


FIGURE 6.11

The FTP command/reply actions (graphic courtesy of Learning Tree International).

Packets 9-11:

Packet 9 is an FTP command USER sent from the FTP client to the FTP server. Figure 6.13 shows the protocol decode for packet 9. This decode shows that the user requesting the FTP session is user1.

Packet 10 is the FTP server's reply to the FTP client's USER command. Figure 6.14 shows the protocol decode for packet 10. This decode shows that the reply code sent back is 331, which means that the user name is fine, but a password is needed.

Packet 11 sent from the FTP client to the server acknowledges the server's response.

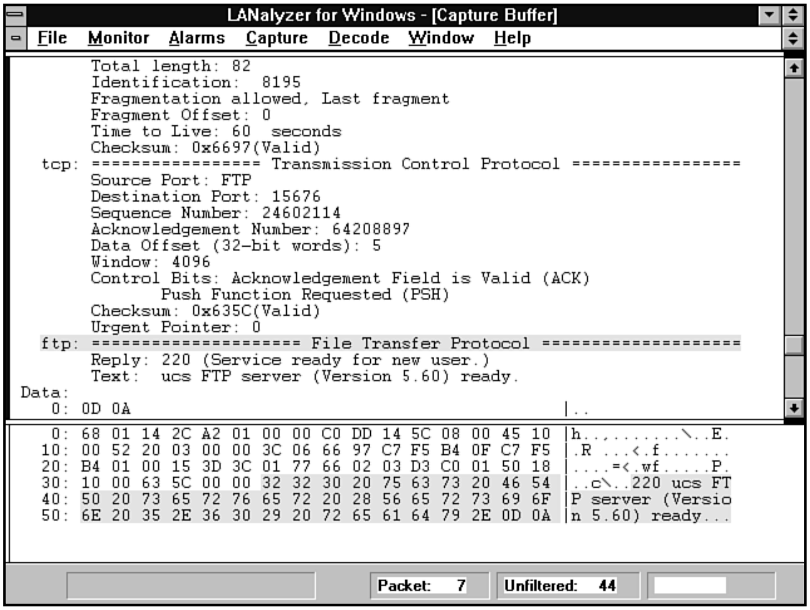


FIGURE 6.12

An FTP server reply-ready packet.

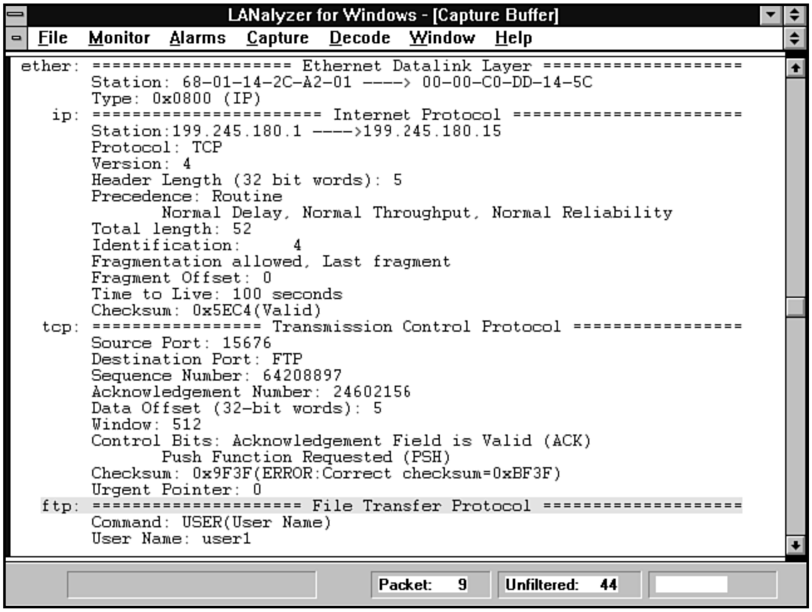
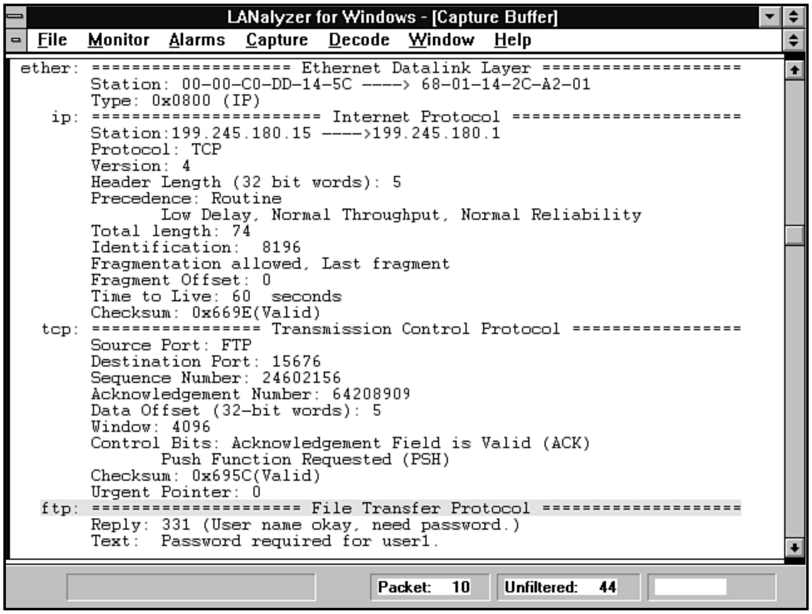


FIGURE 6.13

The FTP USER command.

FIGURE 6.14

The FTP server's response to the USER command.



Packets 12-14:

Packet 12 is an FTP command PASS sent from the FTP client to the FTP server. Figure 6.15 shows the protocol decode for packet 12. This decode shows that the user's password is user1pw.

Please note that the FTP command sends the password in the clear. Any protocol analyzer, such as the one used in this study, can be used to discover the FTP password. For this reason, a stronger authentication mechanism is needed if you want to access an FTP server using your user account over an untrusted network.

Packet 13 is the server's acknowledgment that it received the user password command.

Packet 14, sent from the client to the server, acknowledges this acknowledgment and sets the PSH flag, indicating that for the time being, there are no commands from the FTP client. Figure 6.16 summarizes the actions performed by packets 7 to 14.

Packets 15-16:

Packet 15 is the FTP server's reply to the user password. Figure 6.17 shows the protocol decode for packet 15. This decode shows that the user has logged in successfully.

Packet 16 sent from the FTP client to the server acknowledges the server's response.

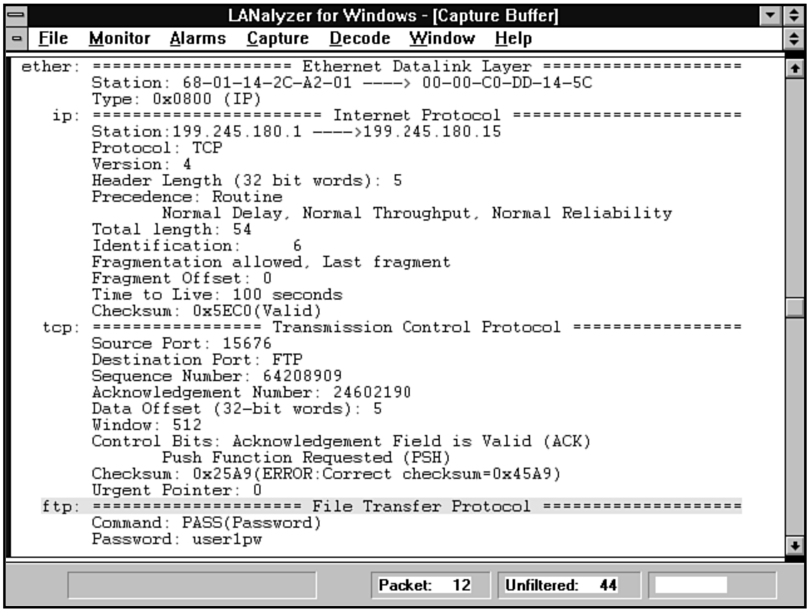


FIGURE 6.15

The FTP PASS command.

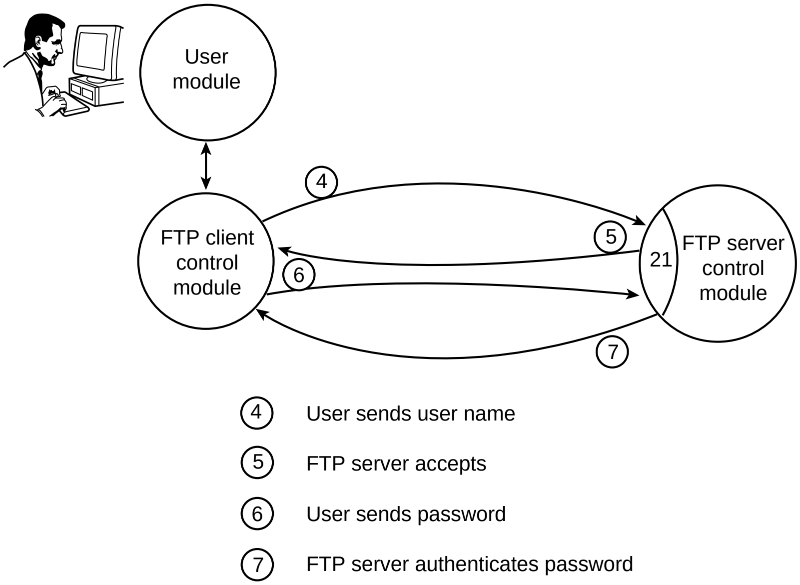


FIGURE 6.16

The FTP client user name authentication (Graphic Courtesy of Learning Tree International).

At this point, the user has entered the following command from the FTP client console in order to retrieve the file "netstart:"

```
get netstart
```

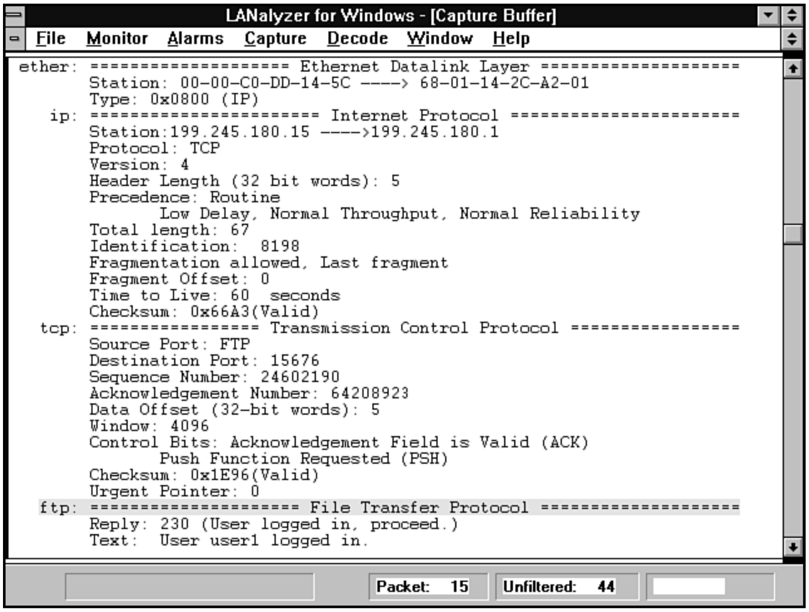


FIGURE 6.17
An FTP server response indicating a successful user login.

As mentioned before, files are retrieved and stored using a separate data connection originating from the server's FTP port 20. However, the FTP server does not know the FTP client's local port to connect to. The FTP protocol solves this problem by having the FTP client send the local end-point address (IP address and port number) to the FTP server. The FTP server then knows the destination port number to open the connection. The FTP client sends its local port number using the PORT command.

Packets 17-18:

> Packet 17 is the FTP command PORT sent by the FTP client to the server. Table 6.1 shows the FTP commands and their meanings. This table contains the definition of the PORT command. The last two numbers, p1 and p2, are the dotted decimal representation of the 16-bit port number. The FTP client selects a dynamic value (greater than 1024) that can be used by the server to make a data connection to the FTP client. Suppose the FTP selects the port number 55814, which is not in use. The PORT command sent by the FTP client with IP address 199.245.180.1 will be
>
> ```
> PORT 199,245,180,1,218,6
> ```
>
> The first four numbers represent the FTP client's IP address 199.245.180.1. The numbers 218 and 6 are the most significant and least significant bytes of the port number. If you were to convert this to a decimal port number, you could do so using the following calculations:

218*256 + 6

= 55808 + 6

= 55814

This value equals the port number that was selected by the FTP client.

Packet 18, sent from the FTP server to the client, acknowledges the client's PORT command.

Table 6.1
FTP Commands

FTP Command	*Description*
USER	Used to identify user for authentication.
PASS	Specifies user's password.
PORT	Specifies FTP client's Internet address and TCP port address as a series of 8-bit decimal numbers: PORT i1,i2,i3,i4,p1,p2. p1 and p2 represent the most significant and least significant bytes of the 16-bit port number.
TYPE	Data type used in file transfer: A = ASCII, E = EBCDIC, I = IMAGE, L = Logical byte size.
STRU	File structure of file to be transferred. F = Unstructured, R = Record, P = Page.
MODE	Transfer mode. S = Stream, B = Block, C = Compressed.
RETR	Gets a file from FTP server.
STOR	Stores a file on the FTP server.
QUIT	Logs user out.
PASV	Specifies that receiver should do a passive TCP open.
NOOP	No operation.

Packet 19:

Packet 19, sent from the FTP client to the FTP server, contains the RETR (Retrieve) command used to download this file. Figure 6.18 shows a protocol decode of this packet. This shows that the name of the file that is retrieved is netstart.

The events described by packets 17-19 are summarized in figure 6.19.

Figure 6.18

The FTP RETR command.

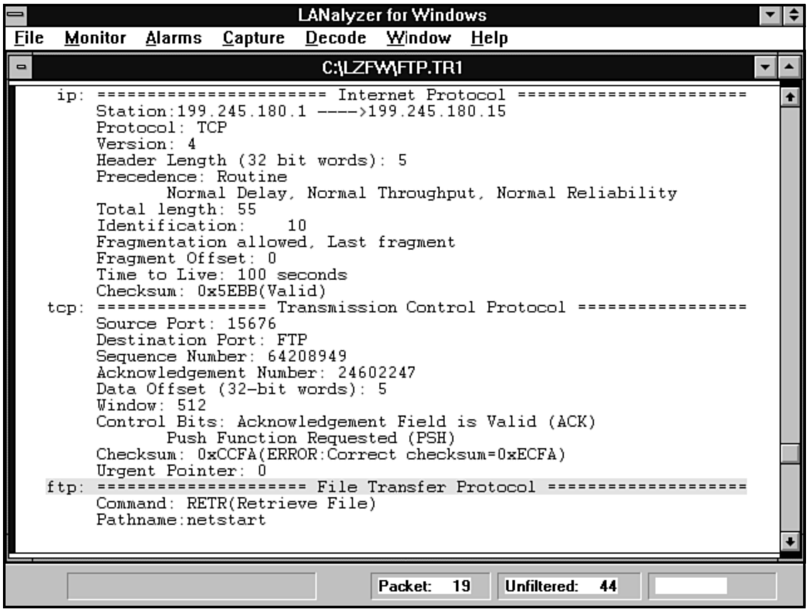
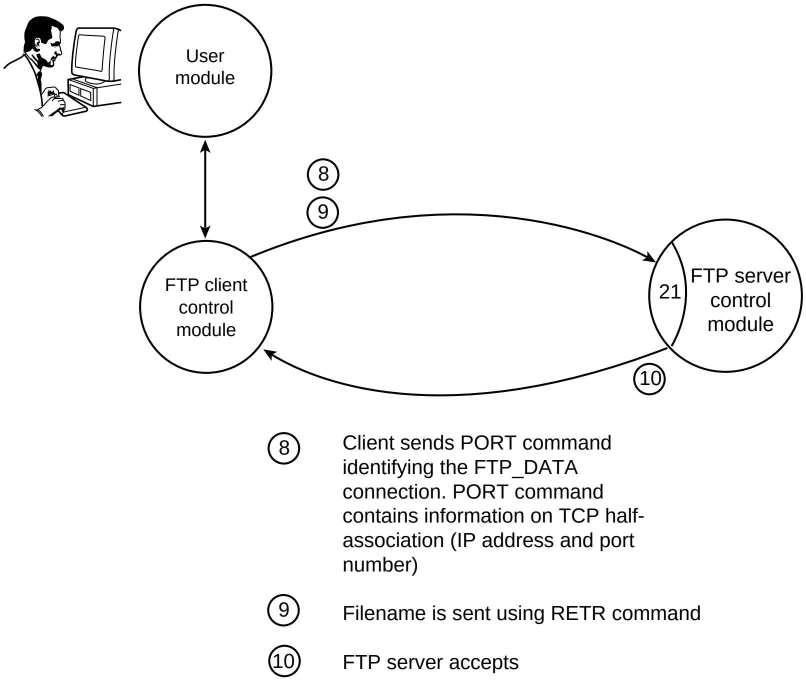


Figure 6.19

The use of the PORT and RETR commands (graphic courtesy of Learning Tree International).



Packets 20-22:

In reply to the RETR command, the FTP server opens a data connection to the FTP client.

Packet 20 shows an incoming TCP connection request from the FTP server to the FTP client. The destination port in the connection request is the one that was specified by the FTP client in the PORT command. Figure 6.20 shows a protocol decode of packet 20. This shows that the source port is port number FTP_DATA (port 20), and the destination port is 55814, specified in the PORT command.

Packets 21 and 22 complete the open connection handshake.

Packets 20 to 22 are very important from a packet filtering standpoint. These packets represent an incoming call from the FTP server. If the FTP server resides on an untrusted network, you must place a filter to allow this connection. The problem is that one does not know beforehand what the port number used by the client is going to be. Because of this, it is difficult to set a proper filter for the incoming call.

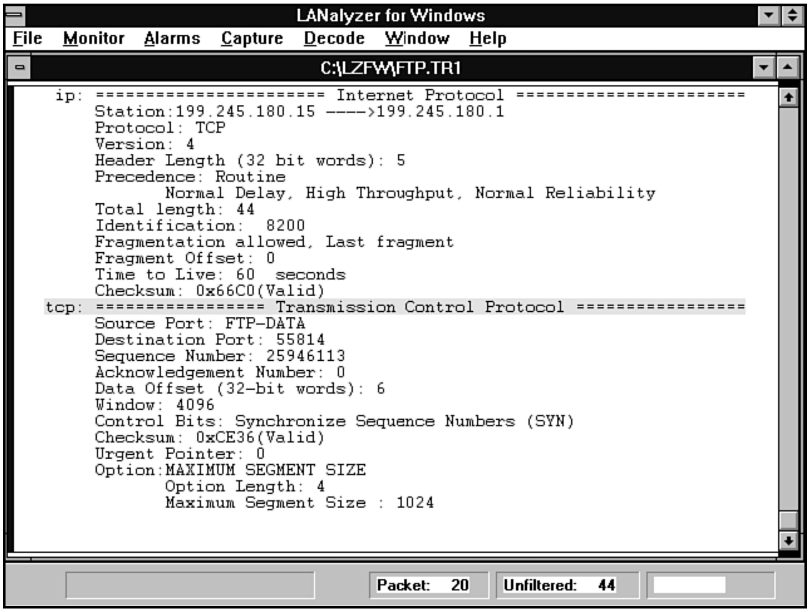


FIGURE 6.20

An FTP data connection from FTP server to FTP client.

Packets 23-24:

In packet 23, the FTP server replies with a status on the file to be retrieved. Figure 6.21 shows a protocol decode of packet 23.

Packet 24 is sent from the FTP client to the FTP server to acknowledge the server's status of file reply. Figure 6.22 summarizes the events in packets 20-24.

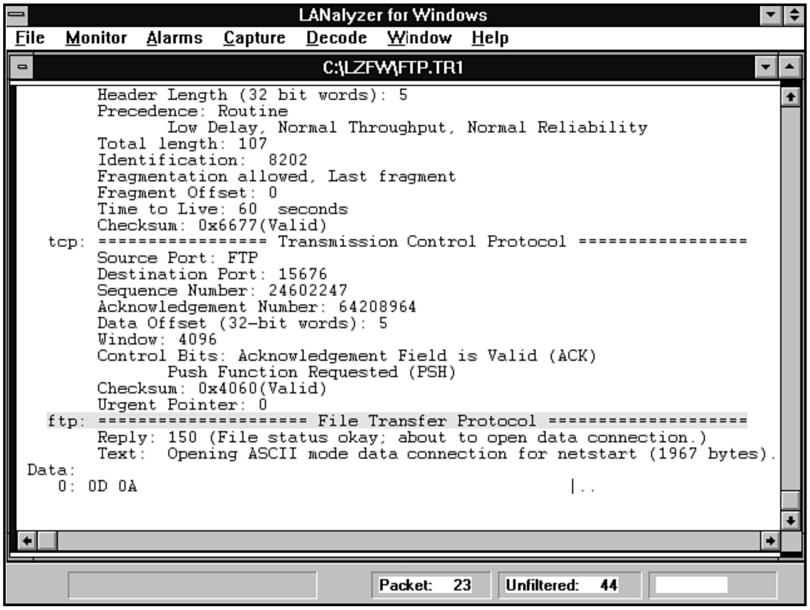


FIGURE 6.21

The status of file reply sent by FTP server.

Packet 25:

Packet 25 commences the transfer of the data in the file to be retrieved. Figure 6.23 shows the data packet containing the contents of the file.

Packet 26:

The FTP server has sent the entire file as a TCP message segment for transmission to the TCP layer. The FTP server then server-announces its intention to break the data connection when file transfer is completed. Figure 6.24 shows a reply code of 226, which indicates that the data connection will close at end of file transfer.

Packets 27-32:

Data is transferred and acknowledged. This phase of the FTP session is illustrated by figure 6.25.

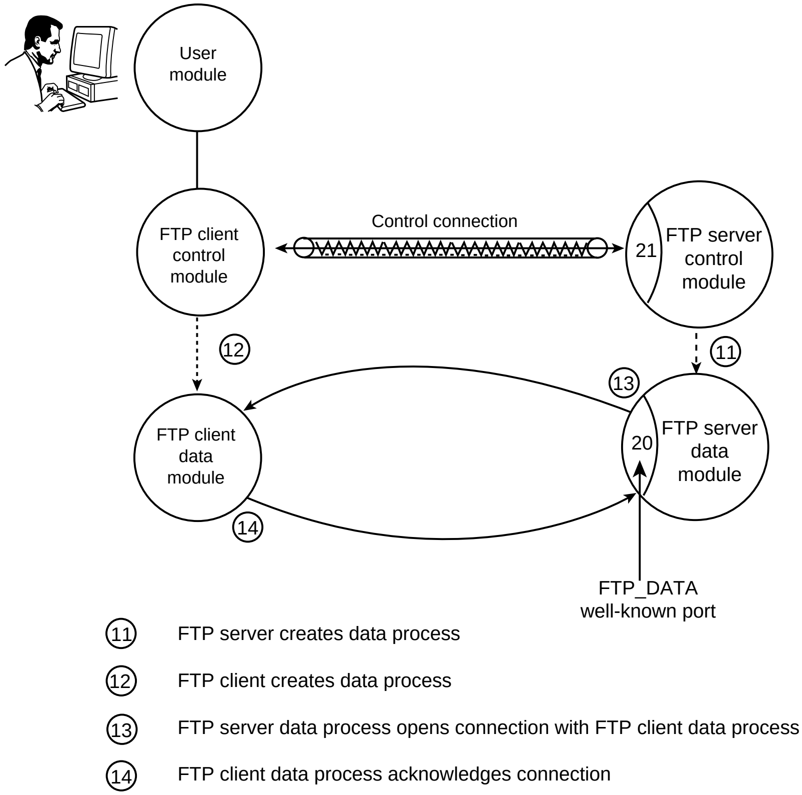


Figure 6.22
The FTP data connection creation (graphic courtesy of Learning Tree International).

Packet 33:

Packet 33 is the last data packet. The FIN flag is set in this packet indicating that the FTP server wants to close the connection (see fig. 6.26).

Packets 34-35:

TCP uses a graceful close mechanism, in which both sides have to agree to break the connection. Packet 34, sent from the FTP client to the FTP server with the FIN flag set, indicates that the FTP client agrees to close the connection (see fig. 6.27). Figure 6.28 summarizes the events described by packets 33 to 35.

Packet 35 is sent from the FTP server to the FTP client to acknowledge the client's agreement in the previous message to break the data connection.

Figure 6.23

An FTP data transfer packet.

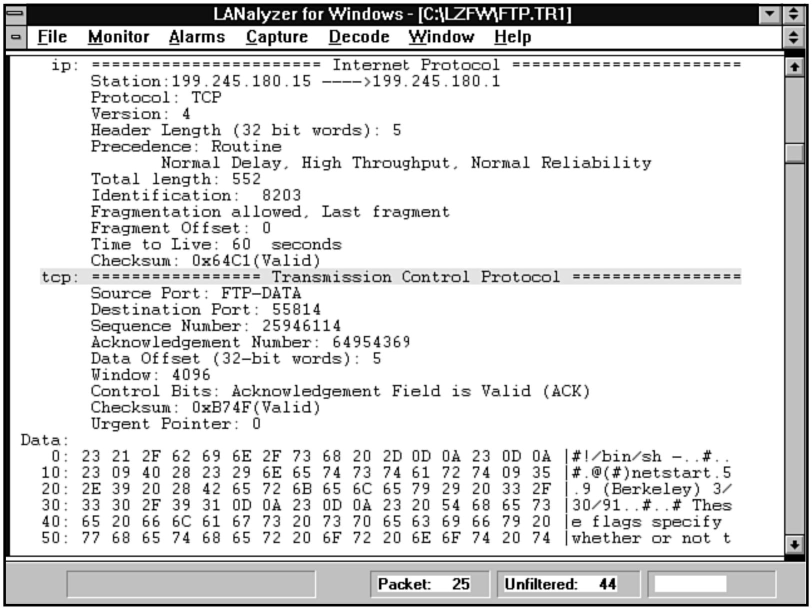
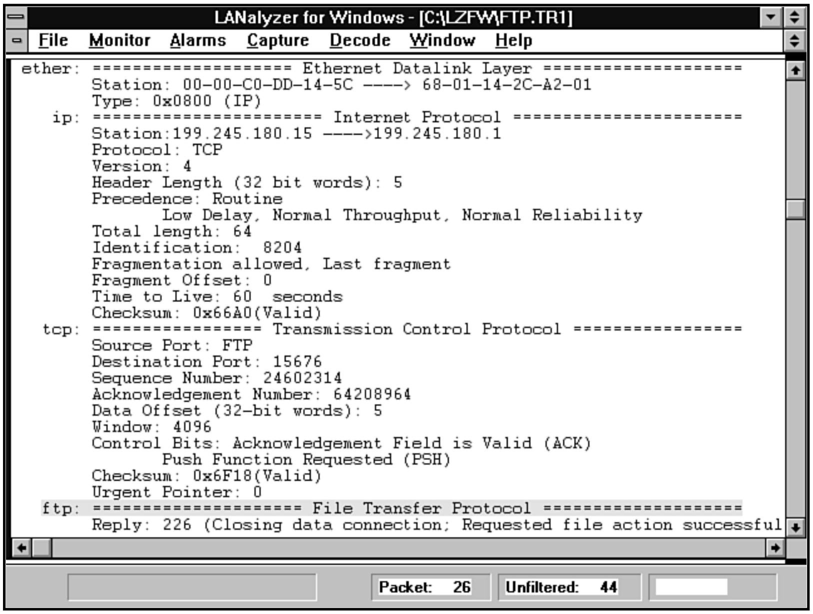


Figure 6.24

The FTP server announces intention to break data connection.



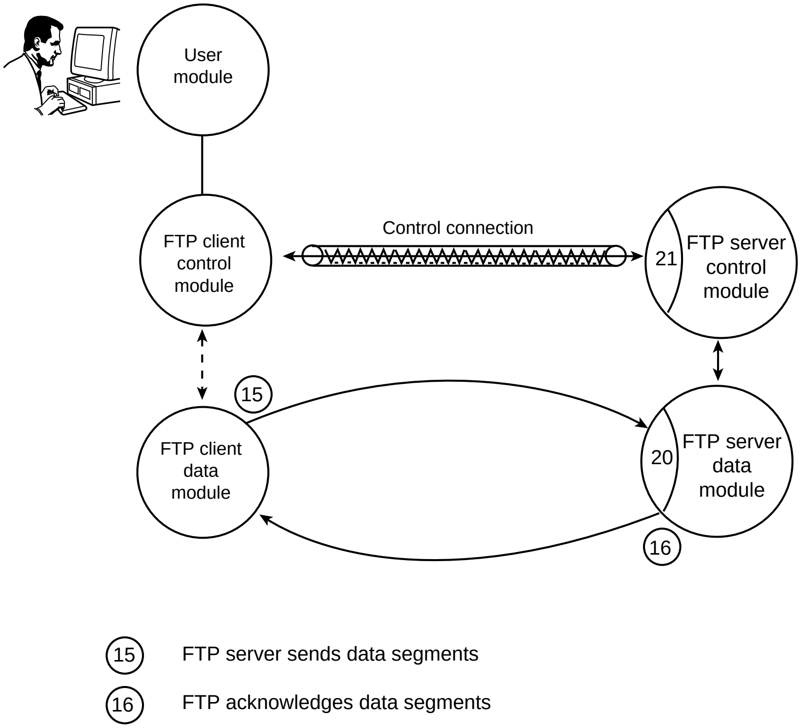


FIGURE 6.25

The FTP data transfer phase (graphic courtesy of Learning Tree International).

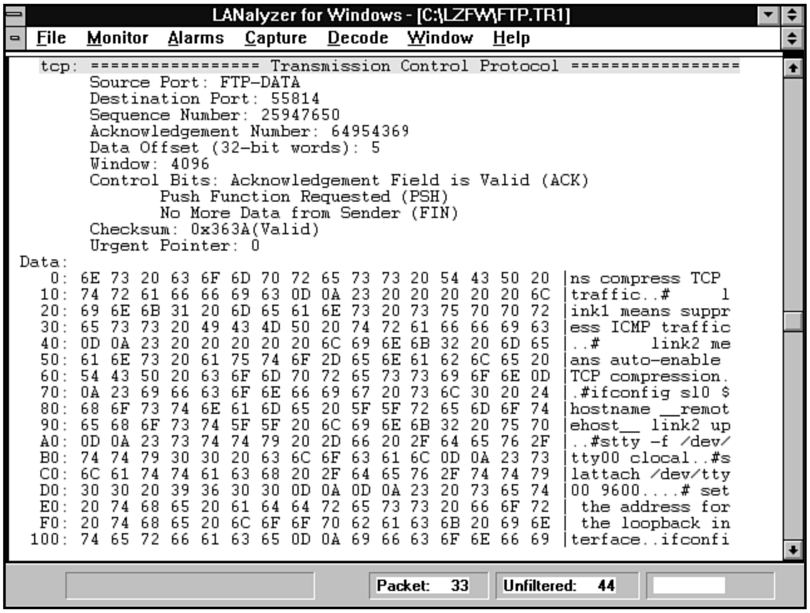
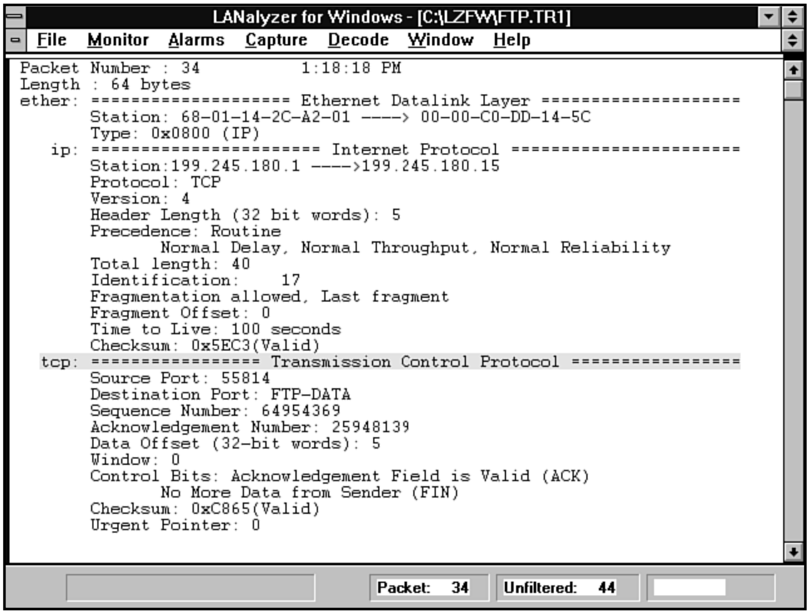


FIGURE 6.26

The TCP layer in FTP sends the FIN flag to break data connection.

FIGURE 6.27

The TCP layer in FTP client sends a reciprocating FIN flag to break data connection.



Packets 36-37:

The user enters the QUIT or BYE command to end the FTP session. In packet 36, the FTP client issues the QUIT command to the FTP server (see fig. 6.29).

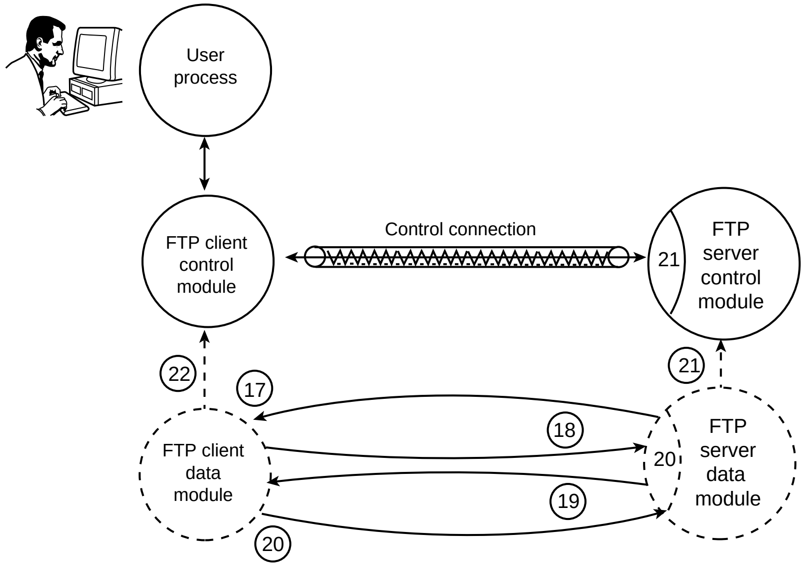
In packet 37, the FTP server replies that it is ready to close the connection (see fig. 6.30).

Packets 38-43:

The FTP server initiates a termination of the FTP connection with FIN flag set, and the FTP client agrees to it with FIN flag set. Additional packets are sent to acknowledge the termination, but the breaking of the control connection is similar to the breaking of the data connection, and therefore, protocol decodes are not shown.

Figure 6.28

The breaking of the FTP data connection (graphic courtesy of Learning Tree International).



17 FTP server sends last data segment

18 FTP client acknowledges last data segment

19 FTP server closes data connection

20 FTP client acknowledges close data connection request

21 FTP server data process terminates

22 FTP client data process terminates

Figure 6.29

The QUIT command from the FTP client.

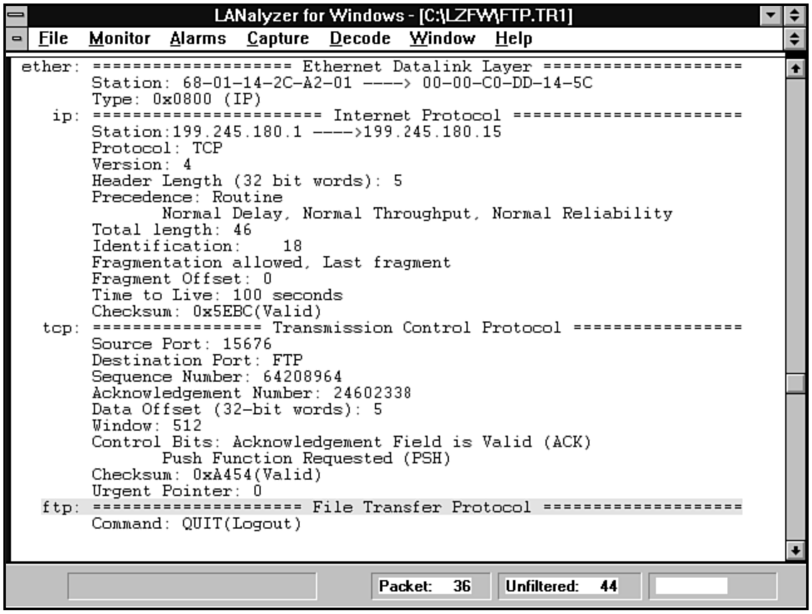
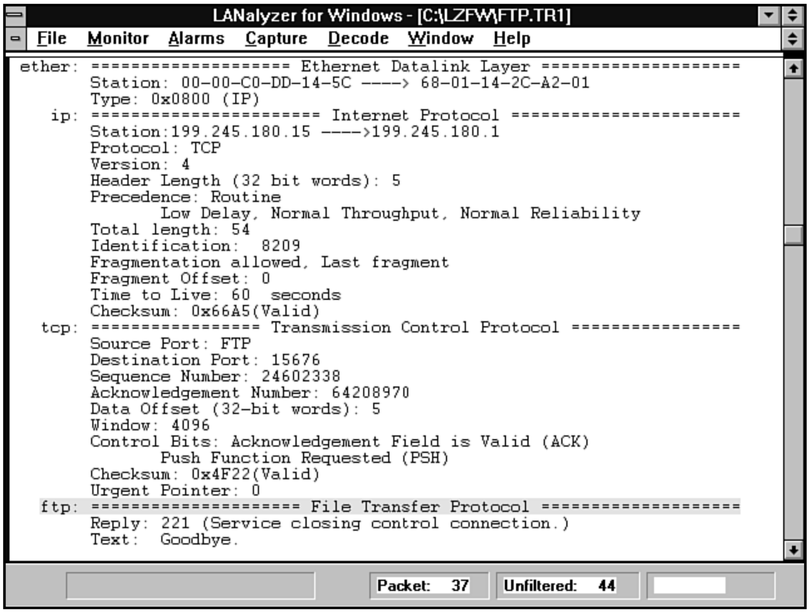


Figure 6.30

The FTP server's response to the client's QUIT command.



Problems in Filtering an FTP Session

The previous section showed the detailed behavior of an FTP session. Consider figure 6.31, in which a screening router is set up between an FTP client on an internal network and an FTP server on an external network. Assume that the network security policy allows internal hosts to initiate FTP sessions with external hosts. Figure 6.32 shows an attempt to set up packet filter rules to implement this policy. From figure 6.32, you can see that you do not know the destination port number for rule 2 because this is set dynamically by the FTP protocol. If you allow the host to call any one of the ports for the FTP "call back" from the server, a program written with evil intent can probe any of the internal network hosts if it originates a call from port 20. This is clearly undesirable.

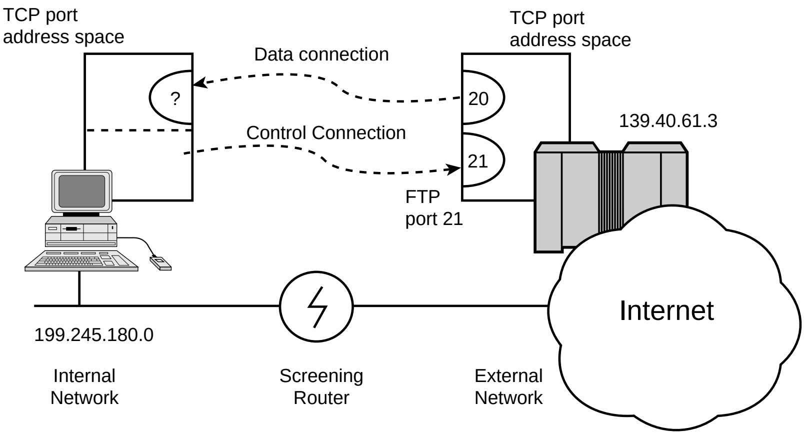


Figure 6.31

An FTP session to an external host through a screening router.

Filter Rule Number	Action	Source	Source Port	Destination	Dest. Port	Protocol Flags Options	Description
1	allow	199.245.180.0	*	*	*	TCP	Permit outgoing TCP connection.
2	allow	*	20	199.245.180.0	*	TCP	Destination port not known.
3							

Figure 6.32

An attempt to set up an FTP connection filter.

FTP PACKET FILTERING USING PORT RANGE RULES AND TCP FLAGS

One way to solve the problem is to use the TCP ACK flag to identify legitimate incoming connections and to block connections to the internal host's standard service ports (usually less than 1024).

Figure 6.33 shows the redesigned packet filter rules for an FTP session. Filter rule 1 allows calls to the external host from any port from the internal network. Filter rule 2 blocks calls to ports less than 1024. This is safe for FTP operation because the local call-back port is greater than 1024 in standard FTP. Filter rule 3 only allows ACK packets from port 20 on the external host.

FIGURE 6.33

The redesigned packet filter rules using the ACK flag.

Filter Rule Number	Action	Source	Source Port	Destination	Dest. Port	Protocol Flags Options	Description
1	allow	199.245.180.0	*	139.40.61.3	21	TCP	Permit outgoing TCP connection
2	block	139.40.61.3	20	199.245.180.0	< 1024	TCP	Block connections to standard services
3	allow	139.40.61.3	20	199.245.180.0	*	TCP ACK=1	Allow ACK packets to any port on internal machine from port 20

A determined intruder could, however, still originate calls from port 20 and probe the port addresses above 1024. Because the ACK flag is on, this attack would require a degree of skill.

FTP PACKET FILTERING USING THE PASV COMMAND

RFC 1579 on "Firewall-Friendly FTP" proposed the use of an FTP command PASV (passive open) that does not require a modification to the FTP protocol, but a modification to the FTP clients. The FTP protocol says that, by default, all data transfers should be over a single data connection. The FTP server does an active open from TCP port 20 to a local dynamic port on the FTP client, which does a passive open on this local port.

Most current FTP clients do not behave that way; they assign a new local port for each transfer and announce this through the PORT command.

If the FTP client sends a PASV command to the FTP server, the server does a passive TCP open on a random port and informs the client of the port number. The client can initiate an active open to establish the connection using a random local port. This mechanism avoids the FTP server call-back to a service port on the FTP client. The FTP client initiates an active open to an external host, and this is usually not a problem with most organizations' network security policies.

If the FTP server does not implement the PASV command, this scheme does not work. PASV is required by STD 3 (RFC 1123); however, not all FTP servers implement it. You usually can detect this problem when you receive a reply code of `500 Command not understood`.

When a PASV command is sent, the server can respond to it with the following:

```
227 Passive i1,i2,i3,i4,p1,p2
```

The *i1*, *i2*, *i3*, and *i4* are the decimal numbers of the server's IP address in dotted decimal notation. The *p1* and *p2* represent a random port assigned by the server. The FTP client can issue an active TCP open with destination port of 256**p1* + *p2* to the server FTP. This mechanism is shown in figure 6.34.

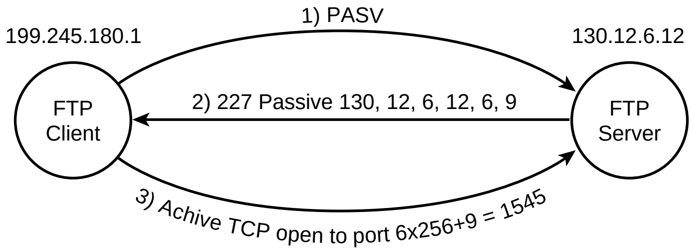


FIGURE 6.34

The use of PASV in Firewall-Friendly FTP.

FILTERING TELNET NETWORK TRAFFIC

TELNET traffic does not require any call-back mechanisms to an unprotected port on the TELNET client, so filtering TELNET sessions on standard port 23 is relatively straightforward.

TELNET can be used as a general mechanism to connect to any port using the following command:

```
telnet host [portnumber]
```

If *portnumber* is not specified, the default TELNET port of 23 is used. If your site security policy does not want internal users to contact services such as Gopher, WWW, or the Weather Underground, these actions are difficult to control if a non-standard port number is used.

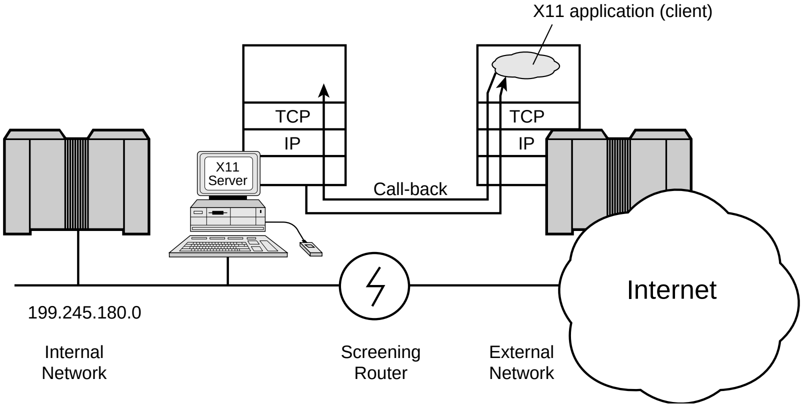
TELNET also can be used by external users to probe what special services, if any, you are providing on TCP ports on your internal machines. You should protect all such services using screening routers or firewall gateway solutions.

Filtering X-Windows Sessions

The X11 protocol uses TCP. Like FTP, the X11 protocol requires an incoming call to the X11 server. The X-server here is the user's display (X-terminal or X-workstation). If internal users want to run an X11 application at an external site (see fig. 6.35), the X11 application needs to make an incoming call to the X11 server. The X11 servers usually use port numbers in the range 6000-6999. You should at a minimum protect the port range 6000-6100.

Figure 6.35

The X11 across a screening router.



Consider the network security policy specifying that X-access from the external site 128.23.0.0 should be allowed, but all other external access should be blocked. If the internal X11-servers use port numbers 6000-6100, you can use the filter design table in figure 6.36 as a start.

Figure 6.36

An example of X11 packet filter rules.

Filter Rule Number	Action	Source	Source Port	Destination	Dest. Port	Protocol Flags Options	Description
1	allow	199.245.180.0	*	*	*	TCP	Permit outgoing TCP connections.
2	block	128.23.0.0	*	199.245.180.0	6000-6100	TCP	Block access to X11 services from 128.23.0.0
3	allow	*	*	199.245.180.0	6000-6100	TCP	Allow X11 access to all other sites.

Packet Filtering and the UDP Transport Protocol

The application services you have examined so far have used TCP. TCP is a connection-oriented protocol. It uses virtual circuits where each side maintains state information, such as sequence and acknowledgment numbers, to determine what data is expected next. This state information specifies a context in which the next packet should occur. This context information is very useful in packet filtering. You can use the TCP ACK flag, for example, to associate a packet as part of an existing TCP session. The ACK flag can be used to distinguish between an incoming or return packet, and the SYN flag can be used to indicate if the packet is part of an open-connection request.

UDP sessions are connectionless. UDP does not use virtual circuits, so it does not retain any state information. There are no sequence and acknowledgment numbers to determine the next packet. Because of this, one has to rely on filtering UDP based on port numbers.

Consider the situation (see fig. 6.37) where host 190.245.180.9 wants to poll an SNMP agent on an external machine at 157.23.13.44, and the network security policy permits such an operation. The SNMP agents use well-known UDP port number of 161. You could write this as a filter rule shown in figure 6.38. Filter rule 1 allows outgoing SNMP polls, and filter rule 2 allows incoming SNMP replies.

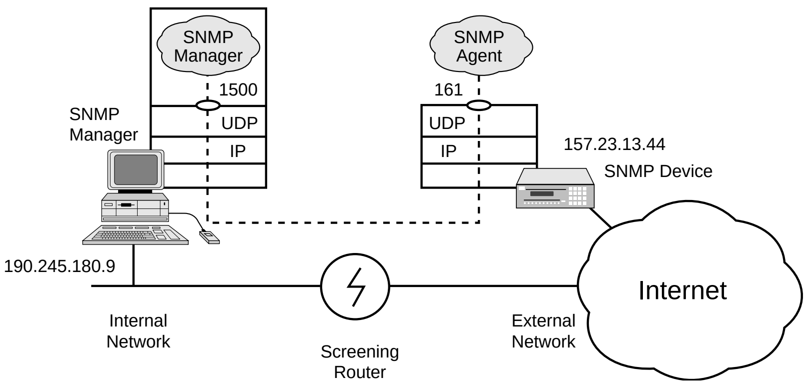


Figure 6.37

SNMP access.

FIGURE 6.38

The implemention of SNMP access security policy.

Filter Rule Number	Action	Source	Source Port	Destination	Dest. Port	Protocol Flags Options	Description
1	allow	190.245.180.9	1500	157.23.13.44	161	UDP	Permit outgoing SNMP polls
2	allow	157.23.1.3.44	161	190.245.180.9	1500	UDP	Allow SNMP replies

Assume the local port used by the SNMP Manager is 1500. The SNMP poll packet consists of the following:

Source IP address = 190.245.180.9

Source port = 1500

Destination IP address = 157.23.13.44

Destination port = 161

The SNMP reply would be the following:

Source IP address = 157.23.13.44

Source port = 161

Destination IP address = 190.245.180.9

Destination port = 1500

What if the following reply was received?

Source IP address = 157.23.13.44

Source port = 161

Destination IP address = 190.245.180.9

Destination port = 1352

Because the source port is 161, by filter rule 2 in figure 6.38, you would allow this packet access. But what if someone has forged a UDP packet with a source port of 161? They could, then, use this to attack port 1352 on the local machine, which is assigned to Lotus Notes.

One way to solve this problem is to explicitly specify rules preventing access to Well-Known Services (WKS) running on internal hosts. The problem here is that it may be hard to keep the packet filter rules updated as new UDP services are added to the network.

TIP

A generally accepted practice is to deny all UDP traffic with the exception of DNS traffic, which is UDP-based for name resolution.

Many of the UDP applications such as NFS, TFTP, and SNMP are LAN-oriented. Therefore, you may consider blocking these UDP services entirely. An exception to this is DNS, which uses UDP, and is definitely not confined to LANs. In this case, you might want to block all UDP traffic from external sites, except DNS. One possible set of filter rules for doing this is shown in figure 6.39. However, as explained earlier, someone can still do address spoofing by impersonating DNS port number 53, and packet filter rule 2 of figure 6.39 will permit this type of packet to reach the internal network 190.245.180.0.

Filter Rule Number	Action	Source	Source Port	Destination	Dest. Port	Protocol Flags Options	Description
1	allow	190.245.180.0	**	*	*	UDP	Allow outgoing UDP packets
2	allow		*	190.245.180.0	53	UDP	Allow DNS queries
3							

FIGURE 6.39

Blocking UDP services with the exception of DNS.

PACKET FILTERING ICMP

The Internet Control Message Protocol (ICMP) is part of the Internet Layer and is described in RFC 792. ICMP must be implemented in all IP protocol modules. This means that all TCP/IP hosts have ICMP support.

ICMP is used to report errors on the IP datagrams. It does not make the IP protocol layer more reliable. It reports errors on the Internet layer, and it is up to an upper-layer protocol such as TCP to make the Internet layer more reliable. ICMP reports information on network parameters and errors on the network; it can also be used to diagnose the network. The following ICMP services are defined:

- Echo test—used to test the availability of a TCP/IP host (ping).
- Time stamp messages for measuring network delay.
- Time to Live expired messages.
- Destination network or host is unreachable messages.
- Errors in IP parameters in the IP header messages.
- Redirect messages for determining better routes.
- Determining subnet address mask of network to which host is attached.
- Source quench messages to inform source to slow down sending of packets. This is an attempt to provide flow control.

ICMP redirect messages generally are sent by routers to other devices, informing them about new routes. If ICMP redirect messages are allowed to filter into the internal network, an external site can send bogus ICMP redirect messages to internal hosts and cause havoc with the internal network's routing tables. This is an example of a "denial of service" attack because it disrupts normal operations. There generally is no good reason for an internal network to listen to ICMP redirect messages from an external network, especially if these originate from an untrusted network. For this reason, you might consider filtering out ICMP redirect packets coming from an external network.

Some hosts are susceptible to ICMP subnet reply messages, even when they have not made an ICMP subnet request. This is obviously a bug in the TCP/IP implementation, and should be identified and eliminated to prevent your network from responding to false ICMP subnet reply messages.

Although the ICMP echo service (popularly implemented by the ping utility) is useful for verifying connections, if you allow external sites to ping your internal network, they can obtain a logical map of your internal network. If this is an important security issue, you might consider denying ICMP requests to the internal network.

PACKET FILTERING RIP

RIP (Routing Information Protocol) is used by many internal networks. RIP exchanges "hop count" information about network and host destinations at periodic intervals (30 seconds). These RIP exchanges are based on trust between the routers. There is no authentication of RIP messages. If a router is mistaken about a route, this error can easily propagate to other routers, producing such ills as routing loops, inefficient routes, and unreachable destinations.

WARNING

> Because of RIP's trustedness between the hosts, it is wise to use an access list to establish trusted hosts or to avoid RIP and use static routes.

It does not take a great deal of imagination to see what can happen if false routing information is deliberately leaked to an internal network. Someone could change the default route information on hosts, for example, so that internal network traffic is diverted to an attacker's host.

You also should disable source routing at the screening router. An intruder can use source routing to force the sending and receiving of packets through the screening router. If you are using Cisco routers, for example, you can disable source routing by using the following command:

```
no ip source-route
```

EXAMPLE SCREENING ROUTER CONFIGURATIONS

This section discusses a few screen routing configuration scenarios to provide you with additional guidance on designing and implementing packet filter rules.

CASE STUDY 1

Figure 6.40 shows a network that has the following security policy:

1. All hosts on internal network 131.44.0.0 can access any TCP service on the Internet.
2. External hosts cannot connect to the internal network except through the mail gateway at 131.44.1.1 where they can access the SMTP mail service only.
3. ICMP messages to the Internet should be blocked.

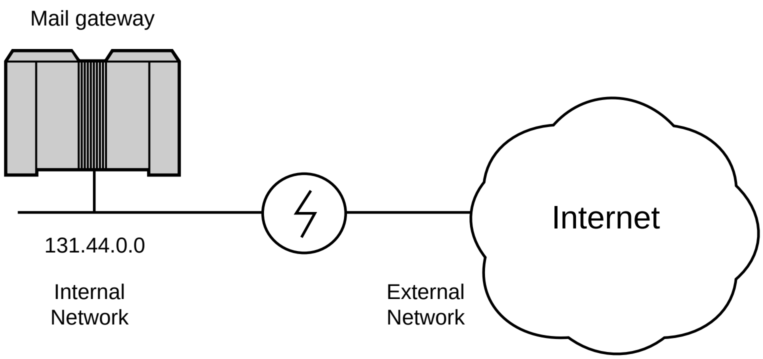


FIGURE 6.40

The Case Study 1 network.

You can express this security policy as filter rules for the Internal (see fig. 6.41) and External port (see fig. 6.42) of the screening router.

FIGURE 6.41

Case Study 1: Filter rules for external port.

Filter Rule Number	Action	Source	Source Port	Destination	Dest. Port	Protocol Flags Options	Description
1	allow	131.44.0.0	*	*	*	TCP	Allow outgoing TCP connections
2	block		NA	*	NA	ICMP	Block ICMP messages to external network
3							

NA = Not Applicable

Figure 6.42

Case Study 1: Filter rules for internal port.

Filter Rule Number	Action	Source	Source Port	Destination	Dest. Port	Protocol Flags Options	Description
1	allow	131.44.0.0	*	*	*	TCP	Allow incoming TCP connections
2	block	131.44.0.0	*	*	*	TCP	Block ICMP from reaching the Internet

Using Cisco routers, the packet filter rules for the external port (see fig. 6.41) can be implemented as shown in the following.

Packet filter rule 1 for external port:

```
access-list 101 permit tcp 131.44.0.0 0.0.255.255 0.0.0.0 255.255.255.255
```

Packet filter rule 2 for external port:

```
access-list 101 deny icmp 131.44.0.0 0.0.255.255 0.0.0.0 255.255.255.255
```

Using Cisco routers, the packet filter rules for the internal port (see fig. 6.42) can be implemented as shown in the following.

Packet filter rule 1 for internal port:

```
access-list 102 permit tcp 0.0.0.0 255.255.255.255 131.44.1.1  0.0.0.0 eq 25
```

Packet filter rule 2 for internal port:

```
access-list 102 deny tcp 0.0.0.0 255.255.255.255 131.44.0.0 0.0 0.0.255.255
```

Case Study 2

Figure 6.43 shows a network that has the following security policy:

1. Incoming e-mail and news are permitted to hosts 144.19.74.200, 144.19.74.201.
2. DNS access to gateway server 144.19.74.202 is allowed.
3. Access to NFS services on internal network is not allowed from external hosts.
4. Internal hosts are allowed all TCP access to external networks, except to Gopher and WWW.

You can express this security policy as filter rules for the External (see fig. 6.44) and Internal port (see fig. 6.45) of the screening router.

Using Cisco routers, the packet filter rules for the external port (see fig. 6.44) can be implemented as shown in the following.

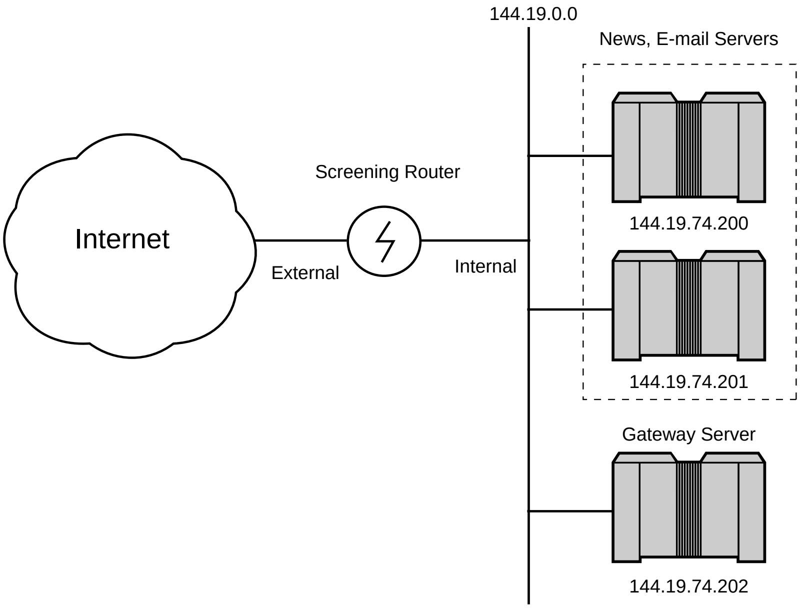


Figure 6.43

The Case Study 2 network.

Filter Rule Number	Action	Source	Source Port	Destination	Dest. Port	Protocol Flags Options	Description
1	block	144.19.0.0	*	*	70	TCP	Block outgoing access to Gopher
2	block	144.19.0.0	*	*	80	TCP	Block outgoing access to WWW
3	allow	144.19.0.0	*	*	*	TCP	Allows all other TCP access

Figure 6.44

Case Study 2: Filter rules for external port.

Filter Rule Number	Action	Source	Source Port	Destination	Dest. Port	Protocol Flags Options	Description
1	allow	*	*	144.19.0.0	*	TCP ACK=1 TCP SYN=1	Permit incoming packets that are part of an established TCP connection
2	allow	*	*	144.19.74.200	25	TCP	Allow e-mail access per security policy
3	allow	*	*	144.19.74.201	25	TCP	Allow e-mail access per security policy
4	allow	*	*	144.19.74.200	119	UDP	Allow NNTP access per security policy
5	allow	*	*	144.19.74.201	119	UDP	Allow NNTP access per security policy
6	allow	*	*	144.19.74.202	53	UDP	Allow DNS access per security policy

Figure 6.45

Case Study 2: Filter rules for internal port.

Packet filter rule 1 for external port:

```
access-list 101 deny tcp 144.19.0.0 0.0.255.255 0.0.0.0 255.255.255.255 eq 70
```

Packet filter rule 2 for external port:

```
access-list 101 deny tcp 144.19.0.0 0.0.255.255 0.0.0.0 255.255.255.255 eq 80
```

Packet filter rule 3 for external port:

```
access-list 101 permit tcp 144.19.0.0 0.0.255.255 0.0.0.0 255.255.255.255
```

Using Cisco routers, the packet filter rules for the internal port (see fig. 6.45) can be implemented as shown in the following.

Packet filter rule 1 for internal port:

```
access-list 102 permit tcp 0.0.0.0 255.255.255.255 144.19.0.0  0.0.255.255
➥established
```

Packet filter rule 2 for internal port:

```
access-list 102 permit tcp 0.0.0.0 255.255.255.255 144.19.74.200 0.0.0.0  eq
➥25
```

Packet filter rule 3 for internal port:

```
access-list 102 permit tcp 0.0.0.0 255.255.255.255 144.19.74.201 0.0.0.0  eq
➥25
```

Packet filter rule 4 for internal port:

```
access-list 102 permit tcp 0.0.0.0 255.255.255.255 144.19.74.200 0.0.0.0  eq
➥119
```

Packet filter rule 5 for internal port:

```
access-list 102 permit tcp 0.0.0.0 255.255.255.255 144.19.74.201 0.0.0.0  eq
➥119
```

Packet filter rule 6 for internal port:

```
access-list 102 permit udp 0.0.0.0 255.255.255.255 144.19.74.202 0.0.0.0  eq
➥53
```

Case Study 3

Figure 6.46 shows a network 144.19.0.0 that is using two subnets. The subnet 1 acts as a demilitarized zone (DMZ) between the internal and external networks. The DMZ is on subnet 15, and the internal protected network is on subnet 16. A firewall gateway, 144.19.15.1 exists in the DMZ. The mail router is on the subnet 16 and has an IP address of 144.19.16.1. The ports of the internal router connecting subnets 15 and 16 has IP address assignments of

144.19.16.81 and 144.19.16.82. The network has the following security policy for the internal router:

1. IP Source routing not permitted.
2. Services originating from subnet 16 are permitted to the firewall gateway on DMZ (subnet 15).
3. Allow all internal traffic to DMZ, except e-mail access to the DMZ (subnet 15).
4. Allow traffic from firewall gateway to internal network.
5. Allow connections from DMZ that show source ports in the range 1024 to 5000. The reason for this could be to allow FTP callbacks (another reason is to show you how to filter on port ranges).

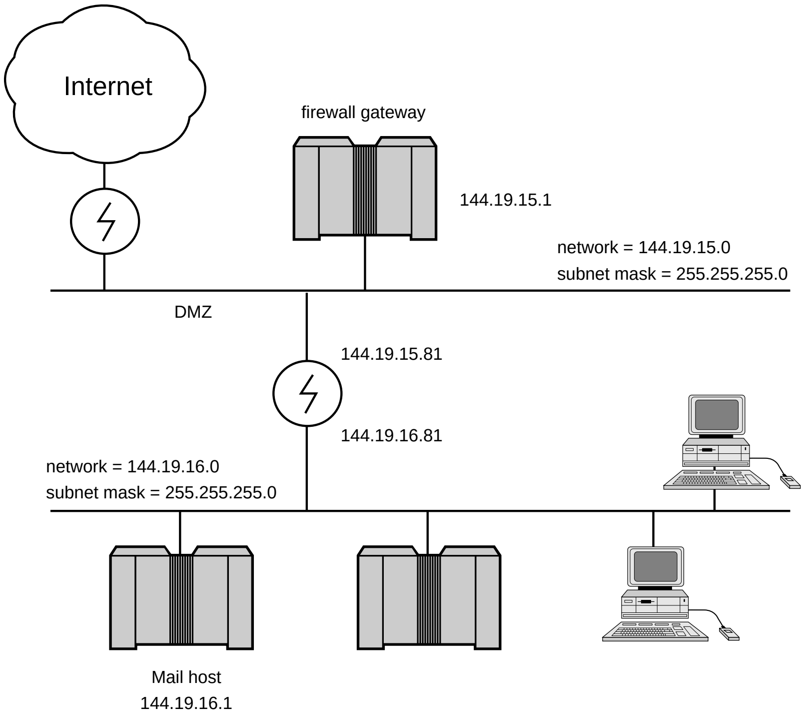


Figure 6.46

The Case Study 3 network.

You can express this security policy as filter rules for Ethernet interface connections subnet 15 (see fig. 6.47) and subnet 16 (see fig. 6.48) of the router.

Figure 6.47

Case Study 3: Filter rules for Ethernet interface on subnet "15."

Filter Rule Number	Action	Source	Source Port	Destination	Dest. Port	Protocol Flags Options	Description
1	allow	144.19.16.0	NA	*	NA	IP	Allow packets from internal network to firewall gateway
2	block	144.19.15.0	*	*	25	TCP	Block e-mail access to DMZ per security policy
3	allow	144.19.15.0	*	*	*	TCP	Allow all other access to DMZ

NA = Not Applicable

Figure 6.48

Case Study 3: Filter rules for Ethernet interface on subnet "16."

Filter rule number	Action	Source	Source Port	Destination	Dest. Port	Protocol Flags Options	Description
1	allow	144.19.15.1	NA	144.19.0.0	NA	IP	Allow traffic to internal network from firewall gateway
2	block	144.19.0.0	*	*	25	TCP	Block E-mail access to DMZ per security policy
3	allow	144.19.0.0	*	*	*	TCP	Allow all other access to DMZ

NA = Not Applicable

The following example shows a more complete procedure on configuring the router interfaces. To configure the router Ethernet interface on subnet 15, you can use the following:

```
no ip source-route      ! Disable source routing for router
interface ethernet 0    ! Identify physical interface
ip address 144.19.15.81 255.255.255.0 ! Assign IP address and subnet mask
ip group access 150     ! Extended access list 150 will apply to this
                        interface
```

To configure the router Ethernet interface on subnet 16, you can use the following:

```
! You do not need to disable source routing as it was done before
interface ethernet 1    ! Identify physical interface
ip address 144.19.16.81 255.255.255.0 ! Assign IP address and subnet mask
ip group access 160     ! Extended access list 160 will apply to this
                        interface
```

The packet filter rules for subnet 15 (see fig. 6.47) can be implemented as shown in the following.

Packet filter rule 1 for Ethernet 0 interface:

```
access-list 150 permit ip 144.19.16.0 0.0.255.255 144.19.15.1 0.0.0.0
```

Packet filter rule 2 for Ethernet 0 interface:

```
access-list 150 deny tcp 144.19.15.0 0.0.255.255  0.0.0.0 255.255.255.255 eq
➥25
```

Packet filter rule 3 for Ethernet 0 interface:

```
access-list 150 permit tcp 144.19.15.0 0.0.255.255  0.0.0.0 255.255.255.255
```

The packet filter rules for the Ethernet interface on subnet 16 (see fig. 5.48) can be implemented as shown in the following.

Packet filter rule 1 for Ethernet 1 interface:

```
access-list 160 permit ip 144.19.15.1 0.0.255.255 144.19.15.0 0.0.255.255
```

Packet filter rule 2 for Ethernet 1 interface:

```
access-list 160 deny tcp 0.0.0.0  255.255.255.255  144.19.15.0 0.0.255.255 lt
➥1024
access-list 160 deny tcp 0.0.0.0  255.255.255.255  144.19.15.0 0.0.255.255 gt
➥5000
access-list 160 permit tcp 0.0.0.0  255.255.255.255  0.0.0.0 255.255.255.255
➥lt 1025
```

Summary

This chapter discussed the implementing of packet filter rules. The packet filter rules were implemented using examples of a commercially available router product. The problems of designing packet filter rules when a "call-back" connection is made from an external network were discussed. Call-back connections are required by most FTP and X11 implementations. Several case studies showing the design and implementation of packet filter rules were discussed.

CHAPTER 7

PC PACKET FILTERING

PACKET FILTERING CAN BE applied to the PC world as well as to Unix towers and workstations. PC-based products are popular because of their relatively low cost compared to RJSC-based Unix platforms or special router products. This chapter explores the ways in which that is possible—in particular, by carefully examining some popular PC-based products that serve as examples.

PC-Based Packet Filter

Packet filtering software packages can be run on IBM-PCs. The IBM-PC, then, becomes a packet filter device. Two popular examples of such devices are:

- KarlBridge
- Drawbridge

The KarlBridge Packet Filter

The *KarlBridge* is a program written by Doug Karl of Ohio State University that runs on a 286/386/486 clone. The KarlBridge provides an inexpensive two-port Ethernet-to-Ethernet bridge that performs protocol filtering.

The KarlBridge filters packets based on any specified Ethernet protocol such as IP, XNS (Xerox Network System), DECNET (Digital Equipment Corporation Network), LAT (Local Area Transport), AppleTalk, NetBEUI (Net BIOS Extended User Interface), and Novell IPX (Internetwork Packet Exchange). As KarlBridge has evolved, some of its features resemble a firewall gateway more than a packet filter device.

KarlBridge can provide filtering of IP packets based upon IP address, network, and subnet combinations and port number values. Besides IP, it can also filter the following protocol types:

- DECNET packets based upon DECNET address, area, object number, and object name
- AppleTalk Phase 1 & 2 NBP packets based upon file server name, printer name, and/or zone name
- NetWare SAP packets based upon IPX network number, server name, and socket service number

The shareware working demo of KarlBridge V2.0 is available through anonymous ftp from `ftp://ftp.net.ohio-state.edu/pub/kbridge`.

The KarlBridge is also a commercial bridge or brouter product. The commercial products are based on the commercial version of the KarlBridge or KarlBRouter code. The Ethernet version of the commercial KarlBridge version comes with AUI/10BASE2 or AUI/10BASE-T connectors. It is implemented in a specially configured PC workstation with two Ethernet cards, a special boot ROM, and a floppy drive.

The KarlBridge Shareware/Demo is a limited-function, free version of the commercially available KarlBridge and KarlBRouter. Even though it is a demo version, it is very functional and, for many situations, has just the right features to be very useful.

Requirements for KarlBridge

To build your own KarlBridge, you need the following:

- 286, 386SX, 386DX, 486, or Pentium PC computer (keyboard and monitor are optional)
- Two SMC Elite 16 Ethernet cards 8013EPC
- KarlBridge software

The PC must be a good, reliable clone. The following lists typical hardware specifications for the KarlBridge:

- 16 MHz, 0 wait state 286 or 386SX motherboard or 386DX motherboard with AMI BIOS, 1 MB of RAM, floppy drive and controller; it must be able to boot with no monitor and no keyboard.
- The speaker connection must be modified to power a front panel LED that signifies LAN traffic. Different motherboards need different types of modification. Without this modification, the speaker clicks for each packet forwarded.
- The floppy drive must be capable of withstanding a dusty environment so that if the system is operating for months at a time and then a power failure causes a reboot, the floppy drive will still work. This usually requires a modification to the case so that the air does not flow through the floppy drive itself.
- For additional reliability, the entire system can be burned in for a minimum of two weeks with cycling temperature with full Ethernet load.
- Two SMC Elite 16 boards can be configured with either of the following configurations:

  Configuration 1:

  First board: IRQ 3 I/O Addr 280, Shared RAM D000 (Remote Port)

  Second board: IRQ 5 I/O Addr 2A0, Shared RAM D400 (Local Port)

Configuration 2:

First board: IRQ 3 I/O Addr 280, Shared RAM E000 (Remote Port)

Second board: IRQ 5 I/O Addr 2A0, Shared RAM E400 (Local Port)

If you chose to construct your own PC clone, be aware that the designers of KarlBridge have not tested the program with hardware other than a 16 MHz, 0 wait state 286-16, 386SX-16, and 386DX40 with the I/O bus running at 8 MHz. The SMC cards may not function properly in some 486 machines.

The software is configured in a PC. The configuration produces a disk that contains properly configured software.

Overview of Setting Up KarlBridge

After you obtain a copy of KarlBridge software, you must configure it on an MS-DOS PC. The current release is compressed ("zipped"), so you must use a tool such as PKUNZIP to extract the files. The distribution software comes with a program called KBRIDGE that you can use to configure the KarlBridge software. The KarlBridge distribution should contain the programs shown in table 7.1.

Table 7.1
KarlBridge Distribution Software

Program Name	*Description*
kbconfig.exe	KarlBridge configuration program
kbc.exe	KarlBridge configuration support program
kbc.sun4	Sun 4 Sparc version of kbc.exe program
kbconfig.cfg	File that contains IP address, mask, and so on
kbhelp.hlp	Help file for the KBCONFIG.EXE program
kbridge.exe	Executable image of KarlBridge for 2 SMC Elite 16 boards with the WD83C690 chip

The following is an outline of the steps to configure KarlBridge:

1. From the DOS prompt, change to the directory that has the KarlBridge software. Issue the following command on your PC:

```
KBCONFIG KBRIDGE.EXE
```

You should see the KarlBridge configuration program. Set up the KarlBridge configuration. The configuration procedure is quite lengthy and can be quite involved. Details of this procedure are presented in the next section.

2. The KBCONFIG program modifies the actual bridge program file KBRIDGE.EXE.

3. Create a bootable floppy. You can do this from the DOS prompt by issuing the following command:

   ```
   FORMAT A: /S
   ```

4. Create an AUTOEXEC.BAT file on the floppy disk that contains the following single command:

   ```
   KBRIDGE
   ```

5. Copy the KBRIDGE.EXE file and the AUTOEXEC.BAT to the bootable floppy:

   ```
   COPY KBRIDGE.EXE A:
   ```

6. You now have a bootable KarlBridge floppy.

7. After setting up the hardware for the KarlBridge, boot with this floppy.

KarlBridge Configuration

The KarlBridge configuration is done by KBCONFIG.EXE. This is a menu-driven program that modifies the KBRIDGE.EXE program. The KarlBridge provides a great deal of flexibility. This section guides you in configuring the more important capabilities of KarlBridge. The shareware product is a bridge that can be configured to perform protocol filtering. If your network requires that the filter device be a router, you must purchase the router version of this product.

The term *remote* in the KBCONFIG program refers to machines connected to the bridge's Port 0. This is the SMC Ethernet board with the I/O address 280. The term *local* refers to machines connected to the bridge's Port 1. This is the SMC Ethernet board with the I/O address 2A0.

The rest of this section illustrates the different configuration options for KarlBridge. Start the configuration by issuing the following command:

```
KBCONFIG KBRIDGE.EXE
```

You should see the opening screen similar to figure 7.1. Select OK.

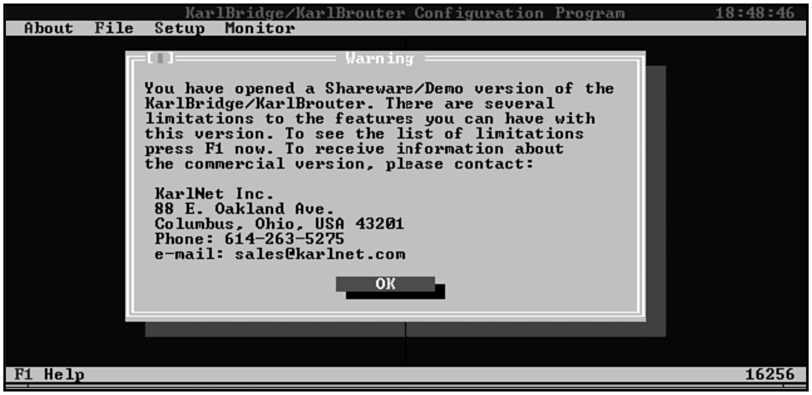


Figure 7.1

The KarlBridge opening screen.

Press Alt+S to invoke the setup menu. You should see a screen showing you the setup options (see fig. 7.2). Select the General Setup option (see fig. 7.3).

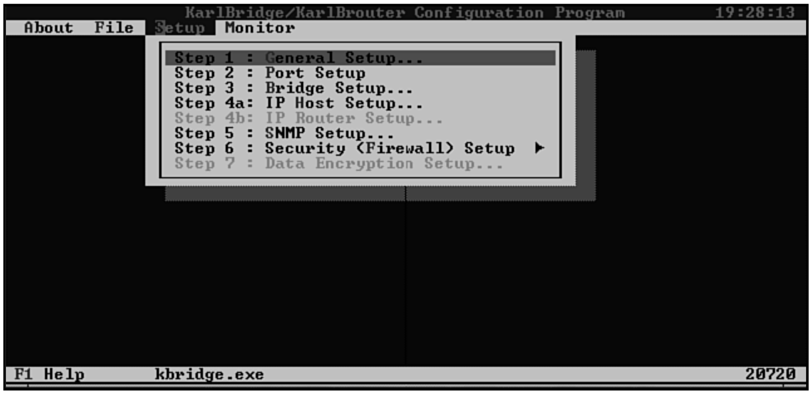


Figure 7.2

The KarlBridge Setup menu.

Figure 7.3

The KarlBridge General Setup screen.

The following options are enabled by default:

Enable Bridging

Enable Security Filters

Enable Watchdog Reboot Timer

Enable VGA real-time display

Select OK after making your selections. Use the spacebar to select/deselect. The meaning of the options in figure 7.3 are described next:

- **Enable Bridging.** The transparent bridging function will be enabled. The device acts as a bridge (examining the data-link layer addresses only), transferring packets between local and remote ports. If you want the Bridge/Router to perform the bridging function, then you must enable this. When bridging is enabled, the Bridge Menu can be used.
- **Enable IP Routing.** If you have purchased the IP Routing option with the KarlBridge, you can enable it with this button. The routing works properly only if the routes are set up in the IP Route menu.
- **Enable Security Filters.** Enabling security filters causes the KarlBridge/KarlBRouter to analyze each of the network layer headers in a packet to determine if it should be passed or dropped. If the KarlBridge or KarlBRouter is to be used as a simple, standard transparent bridge or simple IP Router with no advanced filtering, then this feature should be disabled. If you want to use the advanced filtering, firewall, and security features, then you must enable security filters. Note that the default settings for UDP/TCP, Novell, AppleTalk, and DECNET is to drop the packets (*That which is not explicitly permitted is prohibited.*). After enabling security filters, you then must enable the appropriate protocol-specific security filter.
- **Enable Data Encryption.** The Data Encryption option can be used either to encrypt tunneled data that flows between KarlBridge tunnel partners or to encrypt UDP/TCP packets that flow between KarlBridge/KarlBRouters. Because only the UDP/TCP data portion of the packet is encrypted, the packet is routed correctly by standard IP routers.
- **Enable Remote Bridging using IP Tunnels.** The KarlBridge/KarlBRouter supports a special feature that enables Ethernet packets of any protocol type to be encapsulated in IP and then sent to other KarlBridges for decapsulation. This tunneling behavior is described in RFCs 1226, 1234, and 1241. This method can be used to set up "virtual" Ethernet LANs between several points on an IP network.

- **Enable Watchdog Reboot Timer.** The KarlBridge/KarlBRouter contains a watchdog timer reboot feature. If no packets are seen on the network for more than 10 minutes (a very rare occurrence), the KarlBridge/KarlBRouter reboots itself. After it reboots, the 10-minute reboot timer does not activate again until a packet is seen on one of the ports. This ensures that only one reboot occurs if the entire network is truly shut down.

- **Enable Real-time Display.** Some KarlBridges and KarlBRouters contain a CGA, EGA, or VGA controller board and display. You can enable the displaying of real-time bridge/router statistics with this option. If you do not have a display, it is recommended that you disable this function.

The next step is to configure the network interfaces. For Ethernet, no special hardware setup is required because these settings are fixed in the hardware requirements discussed earlier. If Bridging is enabled, select Bridge Setup (press Alt+S to see the options). You should see a screen showing you the setup options (see fig. 7.4).

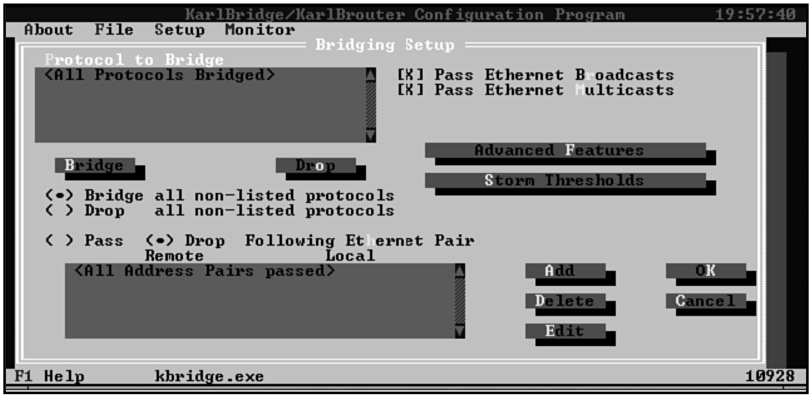


Figure 7.4
The KarlBridge Setup screen.

This is the main screen for configuring the bridge. The different configuration options are described next:

- **Protocol to Bridge/Tunnel.** This menu specifies the Ethernet protocols to bridge, drop, or tunnel. Each protocol can be bridged or dropped by selecting the Bridge or Drop button.

  All other protocols not specified in the menu are then either bridged or dropped depending upon the mode selected by radio buttons labeled Bridge all non-listed protocols or Drop all non-listed protocols.

  It is recommended that you bridge only the protocols that you absolutely need and drop all non-listed protocols. This follows from implementing the security policy

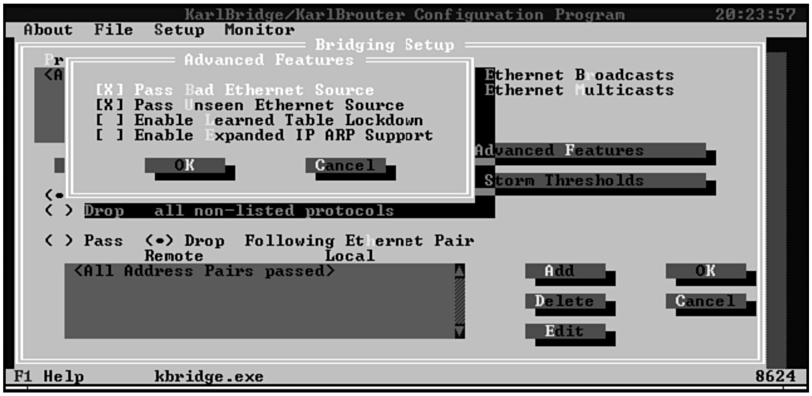
That which is not explicitly permitted is prohibited. If you bridge IP, DECNET, Novell, or AppleTalk, you have the opportunity to set up additional filters under the Security Setup options. At this step, you can specify in more detail the types of services you want to allow or block in accordance with the security policy.

- **Bridge/Drop all non-listed protocols.** This setting determines whether the packets not listed in the Protocol to Bridge or Tunnel menu should be bridged or dropped.
- **Pass Ethernet Broadcast.** Standard Ethernet bridges always forward broadcast packets. Although many protocols do not use broadcasts, many do. For example, the IP, UDP, and ARP protocols use broadcasts. If you do not use IP or any other protocol that requires broadcasts, then you can drop them. Blocking broadcast packets reduces the traffic on your network and the number of interrupts that each computer connected to your network experiences. Networks with a high number of broadcasts slow down the processing of each attached computer even if it is not using the network.
- **Pass Ethernet Multicasts.** Standard Ethernet bridges always forward multicast packets. Some protocols, such as IP and Novell IPX, do not use multicasts. Other protocols, such as OSPF, require multicasts. If you do not use protocols that use multicasts, then you can drop them by shutting off multicasts on the KarlBridge. Shutting off multicast packets reduces the traffic on your network as well as the number of interrupts that each computer connected to your network experiences.
- **Pass/Drop Following Address Pair.** This menu specifies the Ethernet addresses that should be either allowed or blocked. Both source and destination data-link address are checked against this filter. An entire 6-byte Ethernet address can be filtered or just portions of it. This menu can be used to inhibit or promote communication with several particular Ethernet addresses or groups of Ethernet addresses. Because the first 3-bytes of an Ethernet packet (or any IEEE LAN) represent the manufacturer's code, you can filter packets based on the make of the host's Ethernet board. This approach of specifying Ethernet addresses is similar to a standard bridge that supports Ethernet address filtering.

  For example, if the menu is set to Drop following Pair and an address pair of 00-00-C0-00-1A-7B and 00-00-00-4F-XX-XX-XX is specified, then data packets from the address 00-00-C0-00-1A-7A to any addresses that start with 00-00-4F are dropped.
- **Advanced Features.** This menu contains advanced bridging options. These options are described in figure 7.5.

Figure 7.5

KarlBridge Advanced Features Bridging options.



The advanced options are as follows:

- **Advanced Features—Pass Bad Ethernet Source.** Most Ethernet bridges pass Ethernet packets with a broadcast or multicast address as their source (when the first bit is set to 1). The Ethernet specification for Non-Source Routing bridges does not allow these types of packets. These types of packets can be considered "bad" packets. A common failure mode of many Ethernet interfaces and networking software is to transmit packets that have broadcast or multicast addresses in their source address fields. If you do not need the KarlBridge to pass Source Routing packets, you can configure KarlBridge to drop these packets. The default is to pass bad packets.

- **Advanced Features—Pass Unseen Ethernet Source.** Ethernet bridges always forward packets with destination addresses that have not been *learned* (addresses that have not been seen as a source address in a packet). This characteristic is essential for the proper operation of an Ethernet bridge. However, failure mode of many Ethernet interface cards is to send out erroneous packets with good CRCs but with random Ethernet destination and source addresses. Standard bridges pass these erroneous packets because they have not learned the random destination address, and they add this packet's random source address to their finite learned table. This situation can hinder the operation of standard bridges. If you chose to drop unlearned packets, the KarlBridge does not forward unicast packets to an Ethernet address that has not already been seen as a source address. This scheme works for most protocols because it relies on the characteristics of most upper-layer protocols to transmit ARP requests or Hello packets. The default is to pass packets with unseen source addresses.

- **Advanced Features—Enable Learned Table Lockdown.** A standard bridge watches the source addresses of each packet it receives on any of its ports. As new addresses are seen, entries are added in the bridge table. These entries contain the source address and the port number on which that address was received. If that source address is later seen on a different port, the bridge changes the port number in the learned table entry. This condition could happen in a correctly functioning network if someone moved the computer to a different part of the network. This also could happen if someone was trying to capture network packets by spoofing the bridge. Enabling learned table lockdown prevents the port number from being changed after the source address has been seen.

A standard bridge also times-out the learned table records. If learned table lockdown is enabled, then these records are not be timed out; once a record is learned, it cannot be changed or deleted until either the bridge reboots or the learned table becomes completely filled and needs to be reset. A typical KarlBridge learned table can contain over 10,000 records. The default is for this option to be disabled.

✣ **Advanced Features—Enable Expanded IP ARP Support.** Enabling this feature causes the bridge to also watch the IP/ARP packets that occur on the network. No action is taken in response to an IP/ARP packet because that is the role of an IP router. However, the bridge adds the IP address to its IP/ARP table. This feature is helpful on an IP network because it builds a database of MAC layer addresses to IP address pairs. An SNMP monitoring program such as KBCONFIG (Monitor menu) can be used to extract this information. Note that the IP/ARP table is never timed-out in this mode, and this feature is not available if the KarlBRouter is routing IP. The default is that this feature is disabled.

✣ **Storm Thresholds.** One of the features of the KarlBridge/KarlBRouter is its capability to keep Broadcast and Multicast storms from spreading across a network. Network storms are common and can cause bridges, routers, workstations, servers, and PCs to slow down or crash. Storms occur if network equipment is configured incorrectly, network software is not functioning correctly, or programs, such as network games, are not designed correctly.

When you select the Storm Thresholds option in the Bridging Setup menu, you see something similar to figure 7.6.

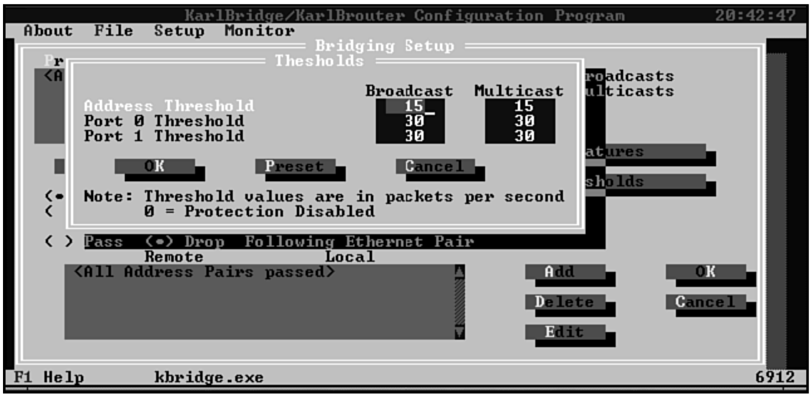


Figure 7.6

KarlBridge Storm Thresholds options.

The options in this screen are explained next:

- **Storm Thresholds—Address Threshold—Broadcast.** This is the number of broadcast packets that can occur in each one-second period before a storm condition is declared for a particular address (host). When a storm is declared, then any additional broadcast packets from that host address are dropped until the storm is determined to be over. The storm is determined to be over when 30 seconds has passed in which every one-second period has less than the stated threshold in broadcast packets.

- **Storm Thresholds—Address Threshold—Multicast.** This is the number of multicast packets that can occur in each one-second period before a storm condition is declared for a particular address. When a storm is declared, then any additional multicast packets from that host address are dropped until the storm is determined to be over. The storm is determined to be over when 30 seconds has passed in which every one-second period has less than the stated threshold in multicast packets.

- **Storm Thresholds—Port Threshold—Broadcast.** This is the number of broadcast packets that can occur in each one-second period before a storm condition is declared for a particular port. When a storm is declared, then any additional broadcast packets received on that port are dropped until the storm is determined to be over. The storm is determined to be over when a one-second period has occurred with no broadcast packets received on that port.

- **Storm Thresholds—Port Threshold—Multicast.** This is the number of multicast packets that can occur in each one-second period before a storm condition is declared for a particular port. When a storm is declared, any additional multicast packets received on that port are dropped until the storm is determined to be over. The storm is determined to be over when a one-second period has occurred with no multicast packets received on that port.

- **Storm Thresholds—Preset.** This button sets the Broadcast and Multicast storm thresholds to the recommended values (see fig. 7.6). These values have been determined to offer good protection without interfering with the operation of the typical network. These values may need to be tuned for your particular network.

If in the General Setup menu the Remote Bridging using IP Tunnels is enabled, then tunnel partners can be set up. The Tunnel Partners menu specifies the IP addresses of each of the KarlBridges set up to participate in the tunnel group. Only specify the other bridges; do not specify the IP address of this bridge.

Some KarlBridges and KarlBRouters contain a special software encryption algorithm that is distinct from the optional WaveLAN DES encryption chip. If Data Encryption is enabled on the General Setup menu and if an Encryption Key is set up in the Data Encryption menu, then enabling encryption here causes all packets transmitted to tunnel partners to be encrypted and any packets received from tunnel partners to be decrypted.

Select IP Hosts Setup (press Alt+S to see the options). You should see a screen showing you the setup options (see fig. 7.7). This screen is used to set up the IP address of the KarlBridge.

FIGURE 7.7

IP Host settings.

The fields that you may need for bridges are described next:

- **OUR IP ADDRESS.** This is the IP address of the KarlBridge itself. If you want to configure or monitor your KarlBridge or if your network supports IP and you want to enable the Ping support and IP/SNMP support, set this to a valid IP address. Setting this address to 0.0.0.0 disables Ping and IP/SNMP support. The KarlBridge is not an IP router. It has only one IP address, and that address applies to both the remote and local networks (both sides of the bridge). Having two Ethernet interfaces with the same IP address is different from a standard IP host, but is appropriate for a transparent bridge.

- **OUR SUBNET MASK.** This is the subnet mask assigned to the IP address. The value is expressed as a hexadecimal pattern. Select a value that is compatible with your subnet.

If you want the KarlBridge to be monitored from an SNMP Manager station, select SNMP Setup (press Alt+S to see the options). You should see a screen showing the setup options (see fig. 7.8). This screen is used to set up the IP address of the KarlBridge.

FIGURE 7.8
SNMP Setup.

The fields that you may need for bridges are described next:

- **READ PASSWORD.** This is the read-only password used for SNMP I support. It is the SNMP password needed to read the MIB variables. The string *public* is the common password used by most SNMP monitors.
- **READ/WRITE PASSWORD.** This is the read/write password used for SNMP support. It is the SNMP password needed to write the MIB variables. The string should be set to a value known only by you. The factory default value for this variable is the string *public* and should be changed to a string known only to you.
- **SYSTEM CONTACT.** This defines the value of the MIB variable, *sysContact*, for system contact. This could be the name of a person or a telephone number.
- **SYSTEM NAME.** This defines the value of the MIB variable, *sysName*, for system name. This is a description of the system.
- **SYSTEM LOCATION.** This defines the value of the MIB variable, *sysLocation*, for system location. This is a description of the system.
- **TRAP HOST IP ADDRESS.** This is the address of the host to which the KarlBridge's trap messages are sent. The host must be set to run an SNMP trap logger that can log these traps. Typically, this is the IP address of the SNMP manager.
- **TRAP HOST PASSWORD.** Only hosts that have this password set for trap messages can receive this trap message. The string *public* is the common password used to receive SNMP trap receivers.
- **ENABLE SNMP COLD/WARM START TRAP.** When enabled, SNMP trap messages are sent for cold and warm boot of the SNMP agent in the KarlBridge. If you have an SNMP manager that can log these events, you should enable these messages in case an intruder causes a shutdown of the KarlBridge.

- ✣ **ENABLE SNMP AUTHENTICATION TRAPS.** If an SNMP manager uses an incorrect password when trying to poll the bridge, it could represent an intruder trying to probe the KarlBridge. In this case, an SNMP authentication failure trap message is sent if this option is enabled.
- ✣ **SNMP IP ACCESS LIST.** This identifies the SNMP managers that can poll the KarlBridge.

If you want the KarlBridge to be used as a screening router, select Security (Firewall) Setup (press Alt+S to see the options). You should see a screen showing the protocol options for which you set filters (see fig. 7.9). The protocol options are TCP/UDP, AppleTalk, DECNET, and Novell's IPX. If you select the TCP/UDP option, you should see a screen for programming the filter (see fig. 7.10).

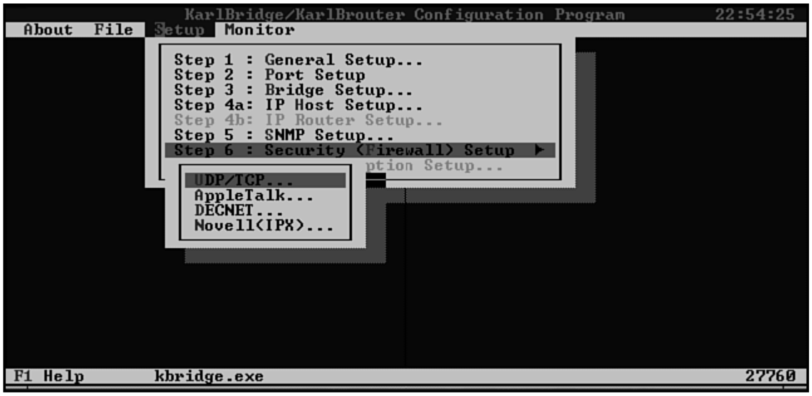


FIGURE 7.9

Protocol options for setting filters.

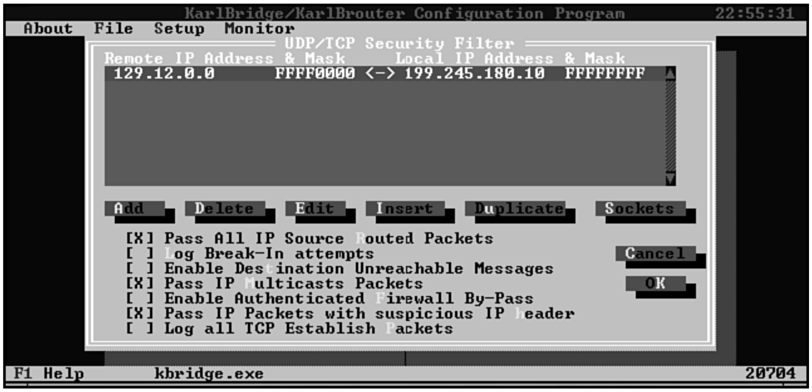


FIGURE 7.10

Protocol configuration options.

These fields are described next:

✣ **Remote and Local Servers.** These menus specify up to 10 IP networks, subnets, host IP address combinations, and server port numbers (referred to as sockets in the interface) that are to be passed or dropped. These two menus must be used together and will be combined into one menu in a future release of the software (see fig. 7.11).

Figure 7.11
Remote and local servers.

Each IP packet that passes through the bridge is checked against the filter entry consisting of an IP Address and an IP Mask. If a packet matches the Remote IP network then the corresponding entry in the Local menu is checked for a match. If a match is found in the Local IP network, the port number conditions set for this entry are matched.

Each IP packet source and destination address is checked against each entry in the list to determine if the packet is to be allowed or blocked. Blocked packets are dropped. Matching is performed on the first entry first and then goes down the list. When a match is found, the action specified on that line is performed immediately. A bit-wise AND operation is performed between the packet's IP addresses and the Mask values. The same is done for the addresses specified in the entry. The two results are then compared. If they are the same, a match has occurred.

Note that the KarlBridge is not an IP Router. This menu specifies the IP networks, IP subnets, and IP hosts on the remote network that hosts on the local network can communicate with.

The following examples can clarify the use of these filter specifications:

Example 1

IP Address	*IP Mask*	*Action Performed*
128.146.0.0	FFFF0000	Pass packets to Network 128.146.x.x
128.150.0.0	FFFF0000	Pass packets to Network 128.150.x.x
Drop All Others		Drop packets to all other networks

Example 2

IP Address	*IP Mask*	*Action Performed*
128.146.10.8	FFFFFFFF	Pass packets to 128.146.10.8
128.146.10.9	FFFFFFFF	Pass packets to 128.146.10.9
128.146.10.0	FFFFFF00	Drop packets to Subnet 128.146.10.x
128.150.25.0	FFFFFF00	Drop packets to Subnet 128.146.25.x
128.146.0.0	FFFF0000	Pass packets to Network 128.146.x.x
128.150.0.0	FFFF0000	Pass packets to Network 128.150.x.x
Drop All Others		Drop packets to all other networks

- **Local IP Address/Mask Menu.** This menu specifies the local IP subnets and hosts that are to be allowed or blocked. Each entry consists of an IP address and an IP mask. A packet that matches is then either passed or dropped as indicated. Each IP packet's source or destination address is checked against each entry in the list to determine if the packet is to be passed or dropped. Matching is performed on the first entry first and then goes down the list. When a match is found, the action specified on that line is performed immediately. A bit-wise AND operation is performed between the packet's IP addresses and the Mask values. The same is done for the addresses specified in the entry. The two results are then compared. If they are the same, a match has occurred.

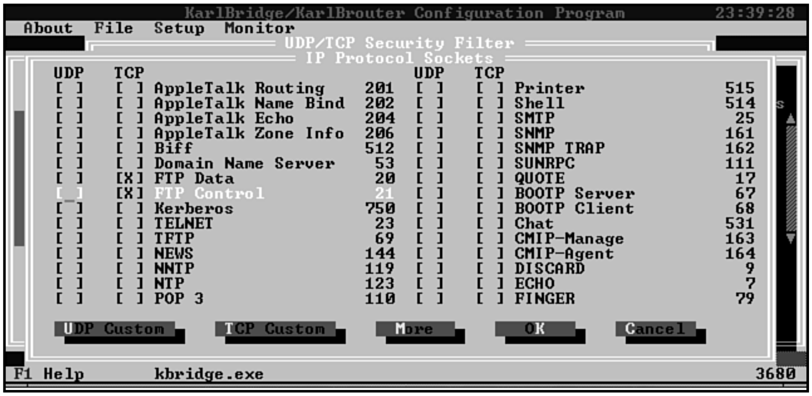
  This option is similar in function to that configured in the Remote IP Address/Mask Menu discussed previously, with the exception that it applies to IP hosts, IP subnets, and IP network on the local network (connected to the local port of the KarlBridge).

- **PASS/DROP IP REMOTE SERVERS MENU.** This menu specifies the remote IP/UDP and IP/TCP server sockets to be passed or dropped. In this context, *server sockets* are sockets assigned numbers less than 1,024, such as the Telnet daemon, the FTP daemon, the SMTP daemon, and many others. *Remote server sockets* are servers located on the remote network. For example, if ftp and Telnet are passed and all other sockets are dropped in this menu, then machines on the local network can initiate an ftp or Telnet session to remote machines, but no other outgoing connections can be initiated.

  If the Sockets button is selected (see fig. 7.12), you can specify the port numbers to be used for filtering.

FIGURE 7.12

Selecting service port numbers.



- **PASS/DROP IP LOCAL SERVERS MENU.** This menu specifies the local IP/UDP and IP/TCP server port numbers to be allowed or blocked. In this context, *server sockets* are sockets assigned numbers less than 1,024, such as the Telnet daemon, the FTP daemon, the SMTP daemon, and many others. *Local server sockets* are servers located on the local network. For example, if ftp and telnet are dropped and all other sockets are passed in this menu, then machines on the remote network cannot initiate an ftp or telnet session on local machines.

- **PASS/DROP IP SOCKETS > 1023 MENU.** This menu specifies the remote and local IP/UDP and IP/TCP sockets greater than 1,023 to be passed or dropped. X11, Multiuser Dungeons (MUD), NFS, and other nonprivileged sockets are in this range. No distinction is made between *local* and *remote* for this menu. You can drop MUD sockets, for example, either by specifying them individually or by setting the menu to pass nothing and Drop All Others. This causes the KarlBridge to drop all packets to or from server sockets > 1,023, which includes MUD, X11, NFS, and all other server port numbers in this range.

You can add a port filter specification by selecting the Add button. Figure 7.12 showed the flexibility available for adding port numbers.

When done setting up security filter, save your changes by pressing Alt+S and selecting Save. Exit the KBCONFIG program by pressing Alt+F and selecting Exit. The options that you have configured are stored in the KBRIDGE.EXE program.

KarlBridge Implementation Considerations

The SMC Elite 16 Ethernet card-based KarlBridge is a general-purpose PC card and is not as fast at forwarding packets as some commercial bridges. At times, extra speed might be needed, such as in situations of network congestion or high network traffic. Commercial versions of KarlBridge are based upon special, ultra-fast Ethernet cards that boost the forwarding rate up to full Ethernet speed.

A commercial clone, such as 286/386/486/Pentium, provides adequate performance for most small networks. If additional performance and reliability are needed to withstand larger extremes of heat, dust, and restarting after power failures, you might want to use the commercial version of KarlBridge.

Currently, if you are using the shareware version of KarlBridge, you must use SMC Elite 16 cards with the 83C690 chip. The SMC Ultra card is expected to be supported in future versions of the shareware/demo KarlBridge.

Other enhancements planned for KarlBridge include the following:

- Drives for:

  COM Port with 16650 UART (SLIP and PPP)

  Aggregation of multiple ports, which means making multiple 56K/T1 or multiple V.Fast modem lines work together to provide faster remote Ethernet bridging and routing

  CATV card for an Ethernet to CATV bridge/router

- Support for:

  Additional wireless card to add to the existing WaveLAN support

  FDDI support

  Token Ring support

  ATM card support

- Adding RIP to the IP router module of KarlBRouter
- Support for spanning tree and source routing
- Filter and firewall enhancements for IP and support for Windows NT server filters

Uses of the KarlBridge

The KarlBridge designers estimate that over 7,000 people worldwide use the shareware and commercial versions of the KarlBridge. The following paragraphs provide some examples that may benefit you in using KarlBridge.

A wireless KarlBridge with the CellWave algorithm was used at Interop, Las Vegas (Spring 1994) to provide a 2 Megabit/sec Internet connection from the Las Vegas Convention center to the MGM Grand hotel (2.5 miles away) and the Bally's hotel (2.0 miles away). Each location had over 25 diskless X terminals. The convention center was configured as a Wireless KarlBridge base station, and the hotels were satellite stations. The CellWave algorithm provides lossless connectivity and takes care of base station repeat and hidden node problems both normally associated with wireless networks.

The Ohio State University network uses over 150 KarlBridges. This network consists of an FDDI ring connecting five hub sites. Each hub has one or more Cisco 7000 Routers with an FDDI interface and several Ethernet interfaces. Each Ethernet network connects through fiber to one or more buildings. A KarlBridge is set up at the entrance to each building to filter out unwanted protocols and to provide firewall and broadcast storm protection and SNMP monitoring. Many of the university buildings have several departments, each with a KarlBridge that can be used to isolate unwanted network traffic between departments. The departments run a combination of AppleTalk, NetWare-based, and TCP/IP-based networks. Protocol problems characteristic for each type of network can be prevented from propagating to other networks. Thus traffic caused by zone name problems need not appear on NetWare networks, and network number collisions on NetWare networks can be prevented from appearing on other networks.

Commercial Version of KarlBridge

The commercial version of KarlBridge supports additional features, such as the following:

- Expanded firewall filters
- Logging of break-in attempts
- Broadcast storm detection and suppression
- Logging of all TCP establish packets

- A new SNMP management, monitoring, and configuration program
- Encryption
- IP Routing
- Supports for high-speed Ethernet cards that enable the KarlBridge/KarlBRouter to forward at full Ethernet speeds
- Support for WAN links such as a dual 56 KB/64 KB card with T1/E1 speeds, ATT/NCR/DEC WaveLAN wireless card, and standard 16550 UART and modem support for dial-up SLIP links
- PPP over the synchronous and asynchronous lines with support for channel aggregation
- RIP for the IP router module
- Intelligent firewall filters that enable the Karlbridge to function like an application level gateway

The commercial version of KarlBridge is also licensed to OEMs by KarlNet, Inc. KarlNet, Inc. is the main commercial supplier of the KarlBridge hardware and software in the United States. KarlBridges and KarlBrouters are manufactured and sold in the United Kingdom by Sherwood Data Systems, Ltd., and other resellers exist worldwide.

KarlNet, Inc. sells the Ethernet-to-Ethernet KarlBridge box for approximately $1,200 (prices vary). KarlBridge models that come with Flash ROM kits cost more.

KarlBridge/Brouter can be obtained with different networking options, such as the following:

- Ethernet-to-wireless bridging and routing using the ATT/NCR/DEC WaveLAN card
- Ethernet-to-56 KB/64 KB/T1/E1 bridging and routing, Ethernet-to-Async SLIP line
- A standard Hayes AT command set compatible modem (supports V.Fast modems, auto dial, dial on demand, and secure lines)

You can obtain additional information on KarlBridge from the following sources:

For the commercial version:

In the United States:

KarlNet, Inc.
614-263-5275
sales@KarlNet.com
URL: `http://www.karlbridge.com`

In the United Kingdom:

Sherwood Data Systems, Ltd. UK
44- (0) 494 464 264

For the shareware version:

Doug Karl
Senior Computer Specialist
Networking Engineering Group
Ohio State University
`kbridge@osu.edu`

THE DRAWBRIDGE PACKET FILTER

The Texas AMU security tools include a package for implementing screening routers called *Drawbridge*. Drawbridge can be found at many security-related sites, so it might be best to research using WWW or Archie to find the closest and latest version. This package is available at the URL `ftp://net.tamu/edu/pub/security/TAMU`.

Version 1.1 of Drawbridge is available in drawbridge-1.1.tar.Z and drawbridge-1.1-des.tar.Z. The drawbridge-1.1.tar.Z package is the Drawbridge base package without DES support. The drawbridge-1.1-des.tar.Z package is a supplemental package that contains the DES support. This package is installed in addition to the drawbridge-1.1.tar.Z package; just extract it on top of the regular package. This adds a few source files and new makefiles to the filter and fm directories. Note that the DES package is not required to operate Drawbridge; it only allows you to manage Drawbridge in a secure manner.

Because of United States export restrictions, only U.S. domestic sites can download the DES package. The package can function without any encryption package, but with reduced security on filter updates across the network.

Drawbridge is a copyrighted but freely distributed bridging filter. It uses a PC with two Ethernet cards to perform the filtering. It is composed of the following three tools:

- Filter
- Filter Compiler
- Filter Manager

Drawbridge was designed and programmed by David K. Hess, Douglas Lee Schales, and David R. Safford of Texas A&M University. You can send comments and suggestions about their package to drawbridge@sc.tamu.edu

Overview of Drawbridge

Drawbridge is a bridging filter that has filtering at the center of the design. It uses custom, table-driven software called the Filter that can run on a dedicated PC. The authors recommend a 33-MHz 486 or better PC. The PC requires two Ethernet cards. Only 16-bit SMC cards are currently supported. The Filter Manager and Filter Compiler both run on Sun workstations.

The Filter Manager communicates and manages Filter (running on a PC) using SUN's NIT support. The Filter Compiler generates filtering tables that are loaded into Filter through the Filter Manager.

The Filter program filters TCP/IP on an incoming and outgoing basis and UDP/IP on an incoming basis. All other protocols are transparently bridged.

Extracting the Drawbridge Software

The file is a Unix "GNU zipped" tar file. After you download this file, you must use the Unix gunzip and tar utilities on the downloaded file. The following commands are listed as a guideline to performing these steps:

```
mkdir  /usr/tamu        # Create a working directory
#Retrieve archive into this directory. The following
#commands assume that the name of the archive is
#drawbridge.tar.Z
gunzip drawbridge.tar.Z                  # This unzips the file
tar -xvBf drawbridge.tar                 # On BSD UNIX. Your UNIX command
                                         # may differ.
```

The following subdirectories and files are created relative to your working directory:

```
drawbridge-1.1/
drawbridge-1.1/drawbridge.README
drawbridge-1.1/fm/
drawbridge-1.1/fm/Makefile
drawbridge-1.1/fm/comm.c
drawbridge-1.1/fm/crypt.c
drawbridge-1.1/fm/fm.8
drawbridge-1.1/fm/fm.c
drawbridge-1.1/fm/fm.h
drawbridge-1.1/fm/lex.h
drawbridge-1.1/fm/lex.l
drawbridge-1.1/fm/nit.c
```

```
drawbridge-1.1/fm/util.c
drawbridge-1.1/doc/
drawbridge-1.1/doc/OVERVIEW
drawbridge-1.1/doc/MANAGER
drawbridge-1.1/doc/FILTER
drawbridge-1.1/doc/COMPILER
drawbridge-1.1/doc/filtering.ps.Z
drawbridge-1.1/doc/DES
drawbridge-1.1/doc/firewall.ps.Z
drawbridge-1.1/COPYING
drawbridge-1.1/filter/
drawbridge-1.1/filter/crypt.c
drawbridge-1.1/filter/filter.8
drawbridge-1.1/filter/filter.c
drawbridge-1.1/filter/message.c
drawbridge-1.1/filter/wd.c
drawbridge-1.1/filter/filter.h
drawbridge-1.1/filter/wd.h
drawbridge-1.1/filter/filter.exe
drawbridge-1.1/filter/makefile.mak
drawbridge-1.1/fc/
drawbridge-1.1/fc/Makefile
drawbridge-1.1/fc/chario.c
drawbridge-1.1/fc/chario.h
drawbridge-1.1/fc/classes.c
drawbridge-1.1/fc/classes.h
drawbridge-1.1/fc/fc.8
drawbridge-1.1/fc/fc.c
drawbridge-1.1/fc/grammar.y
drawbridge-1.1/fc/groups.c
drawbridge-1.1/fc/groups.h
drawbridge-1.1/fc/hosts.c
drawbridge-1.1/fc/hosts.h
drawbridge-1.1/fc/lex.c
drawbridge-1.1/fc/macros.h
drawbridge-1.1/fc/ports.h
drawbridge-1.1/fc/protocols.h
drawbridge-1.1/fc/services.c
drawbridge-1.1/fc/services.h
drawbridge-1.1/fc/Sample
```

The directories fm, doc, filter, and fc are created under the drawbridge directory. In the preceding example, this directory is drawbridge-1.1. This name might change as newer versions of the drawbridge program are released.

The doc directory contains the documentation about Drawbridge. The fm directory contains the source code for the Filter Manager. The fc directory contains the source code for the Filter Compiler. The filter directory contains the source code for the Filter. The Filter program runs under DOS on a PC configured with two network interfaces.

The Filter Compiler (fc) and Filter Manager (fm) both require an ANSI C compiler. The GNU C Compiler (gcc) is recommended by the authors. The Filter requires Borland's Turbo C++ 3.0. An executable version of Filter is provided in case you do not have access to Turbo C++.

To build Filter Compiler and Filter Manager, just go into the respective directories and type the following command:

```
make
```

The preceding command builds the executables. To install the Filter Compiler and Filter Manager, edit the makefiles to set the destination directory, login as root user, and type the following:

```
make install
```

If you have the DES portion of the package, make sure to install that before typing **make**.

To build Filter, copy all of the filter directory to a PC and type **make**.

DRAWBRIDGE DESIGN

Drawbridge compares best to a filtering router firewall configuration as shown in figure 7.13. In a filtering router firewall, a router that has packet filtering support is used to filter packets to an internal network.

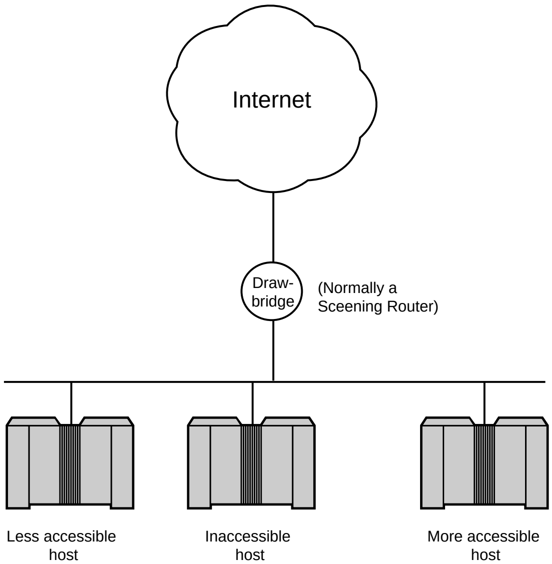


FIGURE 7.13

Drawbridge as filtering router firewall.

In figure 7.13, instead of using a screening router (firewall router), the filtering function is moved from the router into Drawbridge, which acts as a bridging filter.

One problem with packet filtering implementations is that they are difficult to configure. The simple syntax is designed so that a router can efficiently implement it, and not for a network administrator to remember the filter. In Drawbridge, the filter programming is off-loaded. Filter tables are specified using a rich and powerful language. The compiled tables are used for the dedicated filter. This allows Drawbridge to filter in terms of connections, rather than at the level of a packet.

This table design enables arbitrarily complex filters to be defined. In most conventional filtering routers, as filters are added, the performance begins to quickly drop due to how they implement the filtering rules. In Drawbridge, arbitrary numbers of complex filters can be set up, and the performance remains almost constant because simple look ups are performed and only connection establishment packets are filtered for TCP.

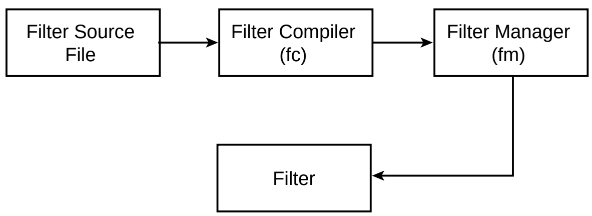
Drawbridge enables the testing of the configuration by enabling the administrator to see the results of a compiled configuration file, and to see if the correct filtering rules have been applied.

Drawbridge, unlike many screening routers, provides support for TCP source port filtering. Because source port filtering does allow the possibility of tunneling, Drawbridge does add the restriction that the destination port must be greater than 900. The authors selected 800 because of "broken" FTP implementations that happen to use FTP data ports beginning at around 900 rather than 1,024.

Figure 7.14 shows the steps involved in developing a filter specification. A source file containing filtering specifications in a special language is generated and maintained by an administrator. This file is then passed through the filter compiler, which generates the tables used by Filter. These tables can be loaded using a Filter Manager or by floppy disk.

Figure 7.14

Developing a Filter Specification.



The Filter Compiler generates four types of tables. The first is a network table, which has an entry for each host in the network. The host portion of an address determines the index into the table. The value in the table defines the "class" that is applied to a host when a TCP connection attempt or UDP packet is directed at that host. Currently, only class B and C networks are supported because class A tables would consume 16 MB of memory, and Filter does not have the capability to use any memory above 1 MB. The tables are defined in terms of *internal hosts* (hosts on the inside of Filter). No filtering is done based on the address of the host outside of the filter, except on a global basis. Filter only controls which inside host services are open, not which outside hosts may access an inside host services.

The *host class* is used as an index into the classes table. This table is composed of four subtables: TCP in, TCP out, TCP source, and UDP in. These subtables are in turn composed of lists that contain port number ranges. A *class* specifies a list out of each subtable that defines a host's filtering. TCP filtering only occurs when packets with SYN=1 and ACK=0 flags (connection initiation) are detected in a TCP header. All other packets of a TCP session are not filtered. Also, all UDP packets are filtered on an incoming basis only.

The last two tables are the allow and reject tables. The *allow table* globally allows packets out from any machine on the inside of the filter to the list of addresses in the allow table. The *reject table* globally rejects packets coming in from any machine on the outside of the filter with an address corresponding to an address in the reject table.

Figure 7.15 shows the algorithm used for filtering outgoing packets, and figure 7.16 shows the algorithm used for filtering incoming packets.

The language used by the Filter Compiler contains constructs for creating the various tables used by the filter. Constructs exist for specifying the network access on a per host basis or a network or subnetwork basis. The language enables groups of services to be created. These groups can be used in cases of related services or to group related machines.

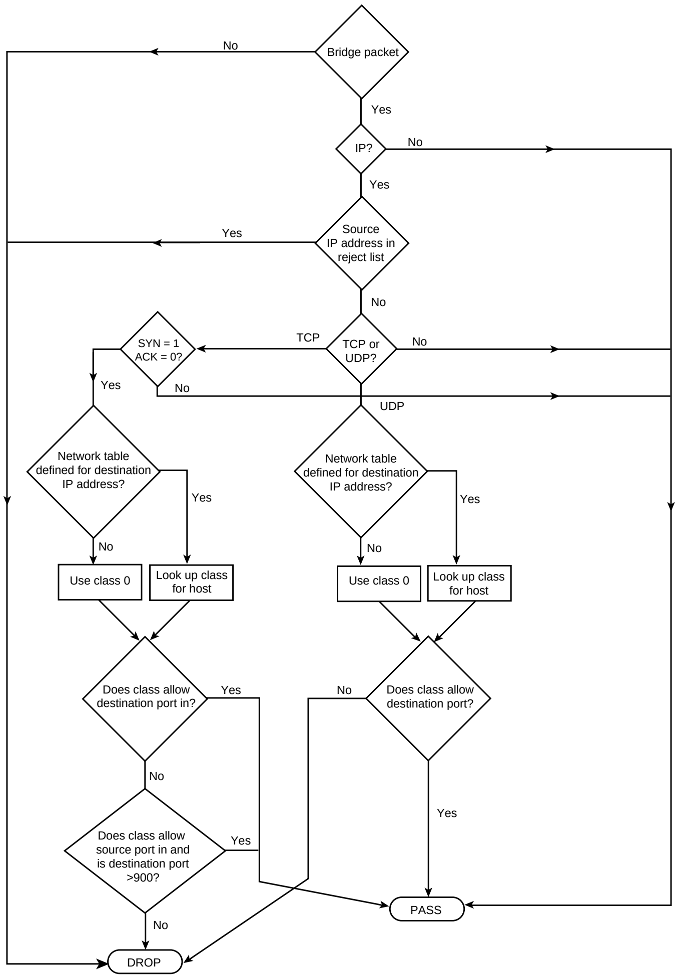
The basic element of the language is a service specification. The service specification contains the following four pieces of information:

- The service
- The protocol
- The source or destination
- The traffic direction

The service can be either an entry from /etc/services or a numeric port. A range of port numbers also can be used. The protocol specifies the protocol that the service uses. The source or destination indicates whether the filter should use the source port or the destination port. The traffic direction indicates whether this is for outbound packets or inbound packets, or both.

Figure 7.15

Filtering of outgoing packets in Drawbridge.



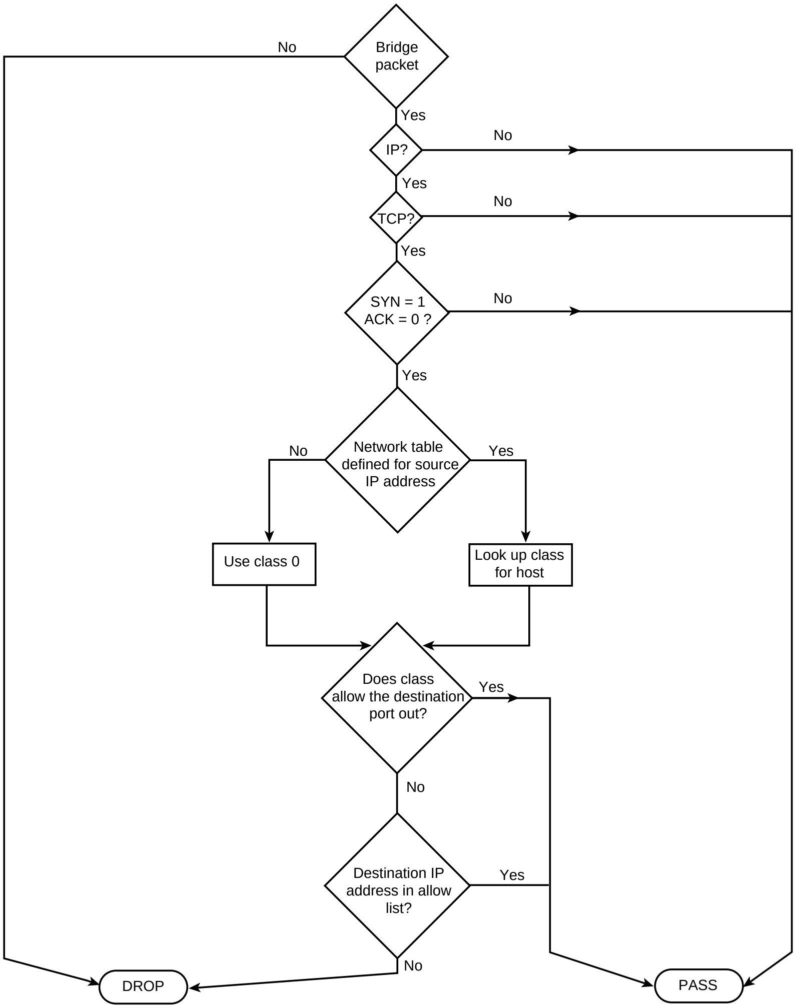


Figure 7.16

Filtering of incoming packets in Drawbridge.

A *group* is a list of comma-separated service specifications or other previously defined groups. Groups can be used to relate services or to categorize machines, enabling quick global changes to a category of machines. Hosts and networks can be granted network access using service specifications or group names.

Drawbridge Filter Language

The basic element of the language is a service specification. The service specification was described in the preceding section. The following are some sample specifications of the filter language.

Example 1 allows SMTP connections in either direction:

```
<smtp in-out>
```

Example 2 allows all outbound connections:

```
<1-65535 out>
```

Example 3 allows UDP-based network time protocol (NTP):

```
<ntp/udp in>
```

Example 4 allows data connect for ftp callback:

```
<src=ftp-data in>
```

Prefacing the service specification with an exclamation mark (!), indicates that this service is not allowed.

Example 5 does not allow TFTP:

```
<!tftp/udp in>
```

Example 6 does not permit rlogin:

```
<!login in>
```

Example 7 does not allow call to an internal NFS server:

```
<!2049/udp in>
```

Groups of service specifications can be used to repeatedly enter the same data. Groups also can be used to quickly change the access characteristics of an entire set of machines.

A *group* is a list of comma-separated service specifications or other previously defined groups, terminated by a semicolon (;).

Example 8 allows incoming mail and gopher connections.

```
define normal <smtp in>, <gopher in>;
```

The preceding example creates a group called normal that includes inbound smtp and inbound gopher. This group can be used in other groups to build up larger groups.

Example 9:

```
define server normal, <telnet in>;
```

In the preceding example, the new group server includes smtp, gopher, and telnet in.

The special group "default" is used to define access class #0, which is the default access for any machine not explicitly defined in the configuration file:

Example 10:

```
define default <1-65535 out>, <src=ftp-data in>, <smtp in>;
```

The preceding example allows all outbound connections, e-mail in, and ftp's data connections in for all machines that do not appear in the configuration file. Normally, all machines want the default services as well; therefore, group default should be added to the machines in the configuration file.

To define the access for a particular host, simply give the hostname and a comma-separated list of service specifications and group names.

Consider the following host services in Example 11:

```
host myhost default, normal;
host newshost default, <nntp in>;
```

Using the preceding definitions for normal and default, host myhost defines the following service definition.

```
<1-65535 out>, <src=ftp-data in>, <smtp in>,
<gopher in>, <telnet in>;
```

And host newshost defines the following service definition:

```
<1-65535 out>, <src=ftp-data in>, <smtp in>,
<gopher in>, <nntp in>;
```

If you want to specify access for an entire range of addresses, you can use the *network* command, which is similar to the host command.

Example 12:

```
network 123.45.67.0-123.45.68.255 255.255.255.0 <!1-65535 in-out>;

network 123.45.69.0 255.255.255.0 <!1-65535 in>;
```

The first entry allows no TCP connections "in" or "out" for 123.45.67.* and 123.45.68.*. The second allows no incoming connections for 123.45.69.*. Notice that the second example is an implicit range of 123.45.69.0–123.45.69.255. In the first example, you must fully specify the full range. For example, the range 123.45.67.0–123.45.68.0 would not include the hosts 123.45.68.1–123.45.68.255.

When multiple service specifications appear in a group definition or host/network command, they are merged. Service specifications are merged by doing an OR on the allowed services and keeping those explicitly disallowed, as shown in the following example:

```
define group1 <telnet in>, <!login in>, <!tftp/udp in>;
define group2 <telnet out>, <smtp in>, <login in>;
define group3 group1 group2 <tftp/udp in>;
group3 will have <telnet in-out>, <smtp in>.
```

Note that even though group2 allows <login in>, it is not possible to override the <!login in> specified in group1. Similarly it is not possible to override the <!tftp/udp in> from group1. group3 inherits the <!login in>. In other words, the actual value of group3 is as follows:

```
<telnet in-out>, <smtp in>, <!login in>, <!tftp/udp in>
```

The <telnet in> and <telnet out> in the combined group are equivalent to <telnet in-out>.

As a host's access capability is specified, either by use of a host or network command, classes of hosts are generated. All the members of a specific class have the same access allowed to them. There can be up to 256 different classes, with class 0 reserved for those hosts not specified in the configuration file. Class 0's capabilities are defined by the special group default. Each class can have up to 32 port ranges for each of incoming TCP, outgoing TCP, incoming TCP srcport, and incoming UDP.

You can break up the configuration file and include the separate files to create the filter control files, as shown in the following example:

```
include file1;
include file2;
```

Nesting of include statements is permitted.

```
reject address netmask
```

You can use the reject command to reject all incoming packets coming from a network address, as in the following example:

```
reject 18.23.0.0 255.255.255.0;
```

You can use the allow command to permit outbound packets from this destination, even if the access for the source host does not allow it. The expected use of the allow subcommand is in situations in which a restricted machine is being granted a network service. For example, if a workstation normally does not have access to the Internet, the allow subcommand can allow these machines to access a service on a specific machine/network. The syntax of the allow command is as follows:

```
allow address netmask [service-specification list];
```

After the rules are specified, you can use the compiler fc to generate the filter control tables from the source configuration file, as in the following example:

```
fc configuration-file
```

The following is a sample configuration file adapted from the Drawbridge documentation:

```
# Example config file
#-------------------------------------------------------------------------
#
# Defaults for any machine not listed in this file.
#
define tcpdefault <1-65535 out>, <src=ftp-data in>, <smtp in>, <auth in>,
                <gopher in>;
define udpdefault <1-65535/udp in>, <!tftp/udp in>, <!2049/udp in>,
        <!sunrpc/udp in>;
define default tcpdefault, udpdefault;
# Include 'default' in telftp since most will need defaults as well.
define telftp default, <telnet in>, <ftp in>;
# Admin requested no access in/out for these subnets
network 123.45.58.0 255.255.255.0 <!1-65535 in-out>;
network 123.45.39.0 255.255.255.0 <!1-65535 in-out>;
network 123.45.40.0 255.255.255.0 <!1-65535 in-out>;
#-------------------------------------------------------------------------
#
host m1.dt.tamu.edu                     telftp, <domain in>;
#-------------------------------------------------------------------------
# NNTP host and CSO phonebook server
host mailnews.tamu.edu                  telftp,
                                        <nntp in>, <time in>,
                                        <csnet-ns in>, <domain in>,
                                        <finger in>;
host slow.tamu.edu                      telftp;
host previous.tamu.edu                  telftp, <domain in>;
host mvt.tamu.edu                       telftp;
# IRC server on here...
host ick.tamu.edu                       <1-65535 out>, <6667 in>, <smtp in>,
<auth in>,udpdefaults;

#
# Machine (PC) in library which use tftp to do document transfers
host sender.tamu.edu                    <1-65535/udp in>;
#-------------------------------------------------------------------------
host h1.tamu.edu                        default, telftp;
#-------------------------------------------------------------------------
# These are all the VMS machines which have been cleared for telnet/ftp.
host h2.tamu.edu                        telftp;
```

```
host h3.tamu.edu                        telftp;
host h4.tamu.edu                        telftp, <domain in>; # zone transfers
host h5.tamu.edu                        telftp;
host h6.tamu.edu                        telftp;
host h7.tamu.edu                        telftp;
host h8.tamu.edu                        telftp;
host h9.tamu.edu                        telftp;
host h10.tamu.edu                       telftp;
# Research group using port 4211
host h11.tamu.edu                       telftp, <4211 in>;
host h12.tamu.edu                       telftp;
host h13.tamu.edu                       telftp;
host h14.tamu.edu                       telftp;
# NTP time server
host h15.tamu.edu                       telftp, <ntp in>,
                                        <time in>;
#---------------------------------------------------------------------------
host s1.tamu.edu                        telftp;
host fast.tamu.edu                      telftp;
host dunno.tamu.edu                     telftp;
host meat.tamu.edu                      telftp, <finger in>;
# A PC FTP server
host somepc.tamu.edu                    default, <ftp in>;
# Has to have X11
host arrow.tamu.edu                     default, <ftp in>, <6000 in>;
host d2.tamu.edu                        default, <ftp in>;
host trouble.tamu.edu                   default, telftp;
#
host td.tamu.edu                        default,
                                        <ftp in>, <auth in>, <domain in>;
# No access in/out
host bee.tamu.edu                       <!1-65535 in-out>;
host g1.tamu.edu                        <!1-65535 in-out>;
host see.tamu.edu                       <!1-65535 in-out>;
host bam.tamu.edu                       <!1-65535 in-out>;
# NTP server
host bird.tamu.edu                      default, <ntp in>, <time in>;
#---------------------------------------------------------------------------
# Gotta have X...
host gotta.tamu.edu                     default, <6000 in>;
host be.tamu.edu                        default, <6000 in>;
host very.tamu.edu                      default, <6000 in>;
host slow.tamu.edu                      default, <6000 in>;
#---------------------------------------------------------------------------
# More name servers for subdomains
host add.tamu.edu                       default, <domain in>;
host bigadd.math.tamu.edu               default, <domain in>;
#
host dead.tamu.edu                      telftp;
#
host someone.tamu.edu                   default, <ftp in>;
```

```
# Nuclear Engineering
host hot.tamu.edu                    default, <ftp in>;
#
# Robotics Lab.  For demo. until 12/19?
host sp1.dt.tamu.edu                 telftp,
                                     <2650 in>, <2655 in>, <2700-2702 in>,
                                     <3200-3202 in>, <3300-3302 in>,
<3500-3502 in>;
# Local MUD's
host someklingon.tamu.edu            default, <2000 in>;
host hmmmmm.tamu.edu                 default, <2000 in>;
```

DRAWBRIDGE FILTER

Filter is an executable program file called FILTER.EXE and should run on most compatibles. The FILTER.EXE is the software component of Drawbridge that runs on a 286/386/486/ Pentium PC and performs packet filtering. This module takes full control of the PC and converts it into a dedicated packet filter device. You can place a line containing the FILTER command in the AUTOEXEC.BAT file of the PC. This implies that the system should be DOS-bootable. DOS versions of 5.0 or higher can be used. By placing the Filter program in the AUTOEXEC.BAT you can ensure that the system boots automatically on power failures of system reboots initiated by the Filter Manager.

The Filter program is written using ANSI C. In its current distribution, Filter is compiled with Borland's Turbo C++ 3.0. No special settings are needed for the CONFIG.SYS file, so you can delete the CONFIG.SYS file or leave it empty. If you are using the Filter on a test PC, you may need to eliminate any device drivers that can cause conflicts. As usual with the PC world, these conflicts are hard to predict because of the many varieties of BIOS systems and device drivers.

The PC on which Filter runs should have 1 MB of memory and 5 MB of hard disk space. A 486 PC is recommended for good performance. Any memory over 1 MB is not used by the filter program because it runs in the DOS-real mode. If you are using the same PC to do the compilation, you will need more memory to modify and recompile the program with Turbo C++.

The Filter program is currently hard-coded to use only the Elite16 (8013) series of Ethernet cards from SMC. Future versions might add support for more cards. Because the PC acts as a bridge, two cards are needed for Filter. One Ethernet interface card is used for the internal network, and the other interfaces to the external network.

For those who do not intend to compile or modify Filter, an executable version of Filter is supplied with the source code. The make files (called *project files* in Borland C++) are included for those who want to examine, modify, or compile the code themselves. Except for two warnings about an unused parameter in a function definition, which can be safely ignored, the compilation process should go smoothly.

The two Ethernet boards must be set as follows:

Ethernet Card 1 [Inside (Internal) Network]

Memory Base Address: 0xD000

I/O port address: 0x280

Ethernet Card 2 [Outside (External) Network]

Memory Base Address: 0xCC00

I/O port address: 0x300

Other considerations in setting up the hardware are as follows:

- Configure the cards properly for either AUI, Thin Wire, or 10BASE-T cabling.
- Interrupt settings are not so critical because the Filter program always polls.
- Depending on the model of 8013 card, you may need to set these configurations with jumpers, with the ezsetup program, or both.

For a system to be used in a production environment, you should "burn-in" a PC configuration for a week before you start using it. Other considerations are to ensure proper cooling and power regulation and protection from high humidity and dusty environments.

The Filter takes one argument, which determines which Ethernet interface is used for listening for remote Filter Manager commands. You can use the following syntax for invoking the Filter program:

```
filter [-l (inside¦outside¦both)]
```

The "inside" should be used if the commands are to be received on the internal network Ethernet interface. The "outside" should be used if the commands are to be received on the external network Ethernet interface. The "both" should be used if the commands are to be received on either internal or external network Ethernet interfaces. If no arguments are specified, the Filter does not respond to any Filter Manager commands.

A typical sample AUTOEXEC.BAT file includes the following:

```
rem Change to the "filter" directory
cd \FILTER
filter -1 inside
```

In the preceding example, the FILTER.EXE has been copied to a special directory called FILTER. You can, of course, use any other directory name.

After the Filter program is loaded, all management activity occurs on the Filter Manager host. If you do not want to use the management software, you need to take the output files from Filter Compiler (using the byte reversal switch), and then copy them to the same directory on the PC on which FILTER.EXE is located.

To halt a running Filter program, type the **$** character at the PC Filter's keyboard. This causes Filter to exit and return you to the DOS prompt. If you never intend to access the Filter PC physically after it is installed, you can remove the monitor, monitor card, and keyboard, provided that you can disable the use of these devices from the BIOS so that your system does not generate errors on rebooting. If you have a keyboard attached to the system, your security policy should consider locking the keyboard or physically securing the Filter PC against accidental or intentional damage.

If you are using the Filter in conjunction with DES and lose your DES key on the Filter Manger host, you must disable the existing key. You can do this by halting the Filter program and then deleting (or replacing) the DES.KEY file in the Filter directory. This puts the Filter in a non-secure mode. You can then install a new DES key.

Drawbridge Filter Manager

The Filter Manager is the tool used for managing a PC running the Filter program. The Filter Manager enables you to load new tables into the Filter and inspect them.

Currently, the Filter Manager can only run on Sun4 systems running SunOS 4.1.*x* with a kernel that has the Network Interface Tap (NIT) driver properly configured. Attempts to compile on other systems using the make files that come with the distribution produce error messages about missing critical "include" files.

If you do not have access to the hardware platform required for the Filter Manager, you can use the "sneaker-net" for distributing the filter tables to the Filter PC. You can do this by using a floppy disk to copy the output files of the Filter Compilation rules to the Filter PC.

You can build the Filter Manager by running the make program in the fm distribution directory. A binary image of the program also exists in the fm distributing directory, so you can copy it to the Filter Manager platform. The Filter Manager is written using ANSI C. The GNU C compiler is recommended.

The executable can be installed anywhere and does not need any support files. However, the Filter Manager program must be installed with SUID set for root. The Filter Manager uses root privilege only long enough to open the network interface and then changes its SUID to the real user.

The Filter Manager is an interactive program modeled after lpc. It provides more feedback and help than your typical Unix tool. The following is the syntax for using the Filter Manager:

```
fm [-i interface ]
```

If the network interface is not specified, the Filter Manager picks a likely device. You can use the following command to determine which interface to use.

```
netstat -i
```

To manage the Filter PC, you must select an interface on the same physical network to which the PC running Filter is attached. If Filter Manager's standard input is not a tty device, it does not print out any information on standard output. Instead, it sends an error message to the standard error (stderr) device.

After the Filter Manager starts, you can use commands such as help or ? to get additional help information. The commands supported include the following:

```
set (verbose¦target¦key) <args>
load (network¦classes¦allow¦reject) <filename>
show (host¦class¦allow¦reject¦target¦verbose¦key) [<args>]
query (host¦class¦allow¦reject¦stats) [<args>]
upload (networks¦classes¦allow¦reject)
write
release (classes¦allow¦reject¦network) [<args>]
ping
reboot
clear
reset
newkey <name>
genkey <name>
quit
```

A "#" at the beginning of a line comments the entire line. You can use the "!" to "escape" to the shell. The following command gives you additional information on using a specified command:

```
help command
```

To communicate with Filter PC, you need to tell the Filter Manager the Ethernet address of the Ethernet card on the PC that is connected to the same physical network. This is done using the set target command. You must specify the correct hardware address, or you cannot communicate with Filter. Filter Manager uses a nonroutable protocol to communicate with the Filter PC, so that it is not susceptible to remote attacks on routed, gated, or routing tables.

The Filter program must be started with the correct switches to enable communication with the Filter Manager. The following are examples of commands that can be used to start the Filter and enable communication with the Filter Manager:

```
filter -l inside
filter -l outside
filter -l both
```

When Filter is first started, it does not have any tables loaded (unless you used a floppy to copy the tables to the Filter PC). In this case, the default rules apply for all packet filtering. You must build a filter configuration file and use Filter Compiler to generate the filtering tables.

By default, Filter also does not use DES. You can use the newkey command in the Filter Manager to install a key in Filter to enable DES. To enable DES support, you must add additional files from a separate DES-enabled version of Drawbridge and then generate an executable using make.

When you run the newkey command, a ~/.fmkey.[name] file is created. This file holds your DES key. In this first case, the key is transmitted to the Filter PC on the network in an unencrypted form. All subsequent newkey generated keys are encrypted with the previous key. If you are extremely concerned about security, you can use the genkey command and copy the first DES key to the Filter PC using a floppy disk.

DES is only used for authentication, and it is not used to encrypt every packet. The authentication mechanism is used to prevent spoofing of the Filter Manager host or the Filter. The filter data goes across the network unencrypted, except for sequence numbers used for the authentication. Changes made to the DES key are sent with the new key encrypted with the current DES key.

The filtering tables created by the Filter Compiler are loaded into Filter Manager, using the Filter Manager load commands. Each table is loaded with separate load commands. These tables can be inspected with the show command. After the tables are loaded into the Filter Manager, you can load the information into the Filter PC using the upload command.

For example, the following command:

```
upload networks
```

uploads all the networks. When the uploaded information is successfully transferred, it is used immediately by the Filter PC. However, the uploaded information is made permanent until you issue a write command that tells Filter to write its currently loaded tables to disk.

You can inspect the currently loaded tables in the Filter PC with the query command. The use of the query command is similar to the show command for examining the tables loaded at the Filter Manager. The query command, however, queries the loaded tables in Filter PC and not the Filter Manager.

If you want to delete tables loaded into the Filter PC, you can use the release command. This command takes effect immediately and also deletes the tables from disk. The Filter PC reverts to default behavior for the deleted tables until new ones are loaded.

When the Filter Manager is started, it reads the file ~/.fmrc on startup and executes all commands in this file. Typically, the .fmrc file contains a set target command so that the Filter Manager can communicate with the Filter PC on startup, as in the following example:

```
set target 00:00:C0:04:AC:89
```

If you are using DES for authentication, you can add a set key command to the .fmrc file to avoid manually entering this command every time the Filter Manager is started.

The ~/.fmkey.* files contain DES keys so that you can manage Filter PC in a secure manner. The machine running the Filter Manager must be secured against attacks, otherwise, the key can be stolen from this machine and the Filter PC accessed by intruders. The ~/.fmrc and ~/.fmkey.* files must be set with the permission mode of 400 or the Filter Manager will complain about excessive permissions granted for these files.

You can use the Filter Manager ping command to determine if you can communicate with the Filter PC. The reboot command can be used to cause Filter PC to cold boot the PC. Therefore, the AUTOEXEC.BAT file on the PC must be configured to correctly restart the Filter PC on rebooting.

The reset and clear commands can be used to reset the Filter Manager. The reset command completely resets the Filter Manager and causes the .fmrc file to be reread. The clear command only causes the currently loaded tables to be unloaded.

Summary

This chapter discussed examples of PC-based packet filters. Packet filtering enables the rejection of “suspect” packets based upon protocol information corresponding to the network and transport layers of the OSI model. The “suspect” packets are those in violation of the network security policy.

By controlling the type of network traffic that can exist on a network segment, the PC-based packet filters can control the type of services that can exist on a network segment. Services that can compromise the network security can be, therefore, restricted.

This chapter also discussed some practical tips in designing and configuring the KarlBridge and Drawbridge filters. A number of examples of packet filter specifications for Drawbridge were presented. Because screening routers operate with limited information from the network and transport layers, they are not as effective as firewall gateway solutions for controlling network traffic. However, they can be combined with firewall gateway solutions to provide a first line of defense. These solutions are discussed in the next chapter.

CHAPTER 8

FIREWALL ARCHITECTURE AND THEORY

IN CHAPTER 5, "An Introduction to Screening Routers," you learned about screening routers and how you can use them to secure a network against intrusions. Screening routers are often used as the first level of defense against an untrusted network.

The packet-filtering technologies that are used in screening routers provide an efficient and general way to control network traffic. They have the advantage that no changes are required to client and host applications because they operate at the IP and TCP layers, and these layers are independent of application-level issues. On the other hand, packet-filtering approaches have not addressed many security requirements because they have incomplete information with which to work. Only network and transport layer (OSI model) information, such as IP addresses, port numbers, and TCP flags, is available for filtering decisions. In many packet filter implementations, the number of rules may be limited. Additionally, as the number of rules increases, there is a high performance penalty because of the extra processing needed for the additional rules.

Because of lack of context information, certain protocols such as UDP and RPC are more difficult to filter effectively. Also, in many packet-filtering implementations, auditing and alerting mechanisms are missing. To ensure their successful implementation, the implemention of packet filters often requires a high level of understanding of communication protocols and their behavior when used by different applications.

Packet-filtering devices such as screening routers are often augmented by other devices called firewalls. *Firewalls*, because they operate at the upper layers of the OSI model, have complete information on the application functions on which to base their decisions. In this chapter, you learn the different types of firewalls that can be deployed.

There are several approaches to building a firewall. Organizations that have programming talent and financial resources often prefer to use a "roll your own" approach. This involves building custom firewall solutions to protect the organization's network. If implemented properly, this is perhaps the most effective (and also the more expensive) approach.

Other organizations prefer to use existing off-the-shelf products, and customize and configure them to meet the organization's network security policy. To make the discussion on firewalls more practical, several commercial and noncommercial firewall products are discussed in the next chapter.

Examining Firewall Components

The major objective of a firewall is to protect one network from another. Usually, the network that is being protected belongs to you (or is your responsibility), and the network that you are protecting against is an external network that cannot be trusted and from which security intrusions can originate. Protecting your network involves preventing unauthorized users from having access to sensitive data, while allowing legitimate users to have unencumbered access to the network resources.

Chapter 3, "Designing a Network Policy," introduced the OSI model as a means of making the distinction between screening routers and firewalls. Though the OSI model is a time-honored way of making distinctions between communication architectures and capabilities, not everyone is aware of it or makes use of it. The term firewall is used by many as a generic term that describes a wide range of functions and the architecture of devices that protect the network. Some, in fact, use the term firewall to describe almost any network security device, such as a hardware encryption device, a screening router, or an application-level gateway.

In general, a firewall is placed between the internal trusted network and the external untrusted network. The firewall acts as a choke-point that monitors and rejects application-level network traffic (see fig. 8.1). Firewalls also can operate at the network and transport layers, in which case they examine the IP and TCP headers of incoming and outgoing packets, and reject or pass packets based on the programmed packet filter rules.

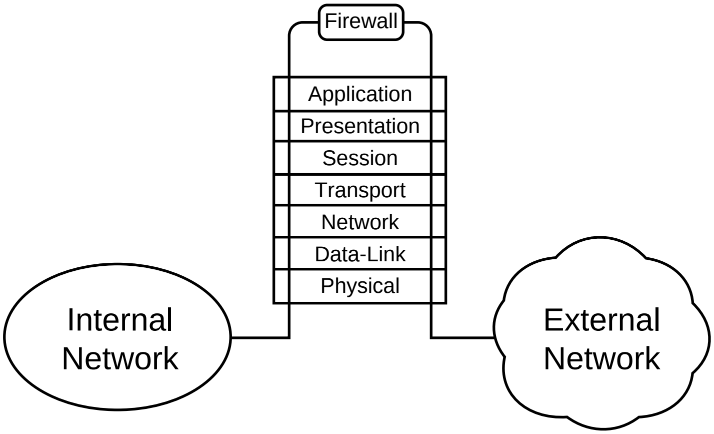


FIGURE 8.1

Firewall operation.

NOTE

> The firewall is the chief instrument used to implement an organization's network security policy. In many cases, authentication, security, and privacy enhancement techniques are needed to enhance the network security or implement other aspects of the network security policy.

The following sections describe the different types of firewalls that can be used.

DUAL-HOMED HOST

In TCP/IP networks, the term *multi-homed host* describes a host that has multiple network interface boards (see fig. 8.2). Usually, each network interface board is connected to a network. Historically, this multi-homed host also could route traffic between the network segments. The term *gateway* was used to describe the routing function performed by these multi-homed hosts. Today, the term *router* is used to describe this routing function, whereas the term gateway is reserved for those functions that correspond to the upper layers of the OSI model.

If the routing function in the multi-homed host is disabled, the host can provide network traffic isolation between the networks it connects to, and yet each network will be able to process applications on the multi-homed hosts. Furthermore, if the applications permit, the networks can also share data. A dual-homed host is a special example of a multi-homed host that has two network interfaces and has the routing functions disabled. Figure 8.3 shows an example of a dual-homed host with the routing functions disabled. Host A on Network 1 can access Application A on the dual-homed host. Similarly, Host B can access Application B on the dual-homed host. The two applications on the dual-homed hosts can even share data. It is possible for the hosts A and B to exchange information through the shared data on the dual-homed hosts, and yet there is no exchange of network traffic between the two network segments connected to the dual-homed host.

FIGURE 8.2

A classic multi-homed host.

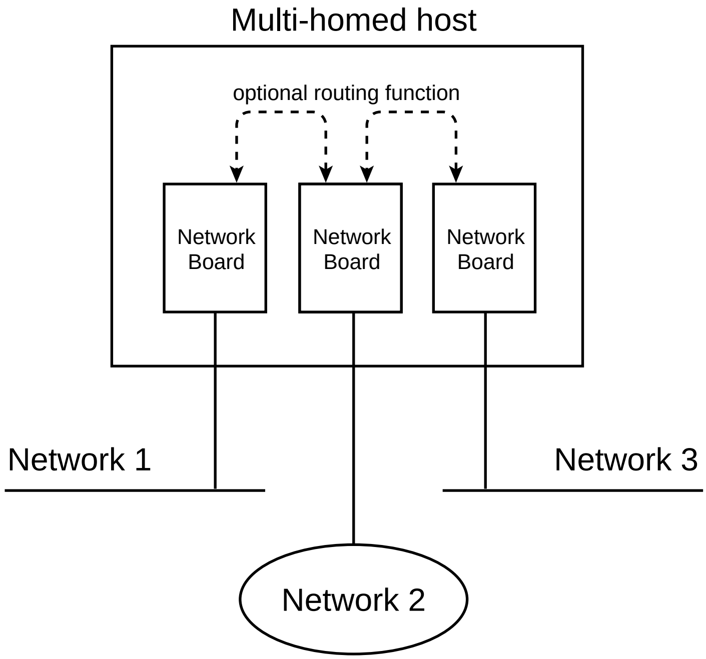
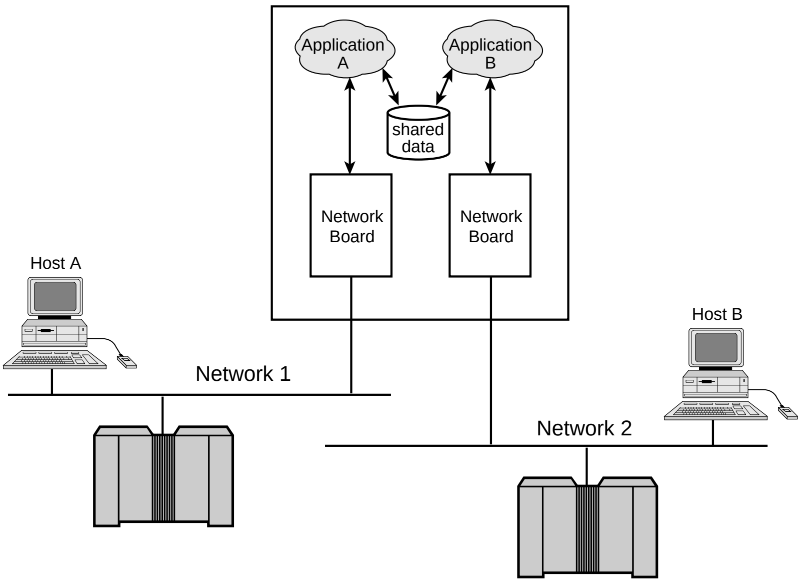


FIGURE 8.3

A dual-homed host.



Dual-homed Host As a Firewall

The dual-homed host can be used to isolate an internal network from an external untrusted network (see fig. 8.4). Because the dual-homed host does not forward any TCP/IP traffic, it completely blocks any IP traffic between the internal and external untrusted network.

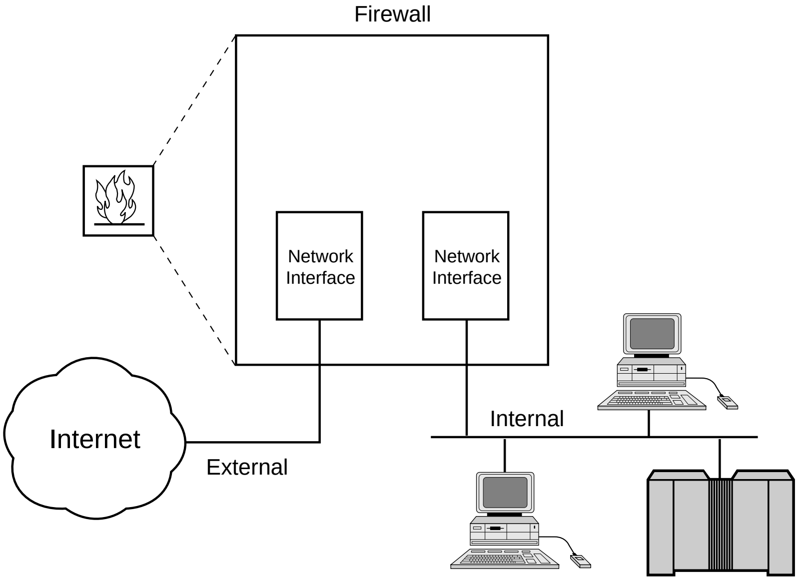


Figure 8.4

A dual-homed host as a firewall.

Internet services such as mail and news are essentially store-and-forward services. The World Wide Web also can be considered store and forward, but the terms "caching" and "proxy" are more commonly used in Web vocabulary. If these services run on the dual-homed host, they can be configured to transmit application services from one network to the other. If application data must cross the firewall, application forwarder agents can be set up to run on the dual-homed host (see fig. 8.5). Application forwarder agents are special software used to forward application requests between two connected networks. Another approach is to allow the users to log in to the dual-homed host, and then access external services from the external network interface of the dual-homed host (see fig. 8.6).

If application forwarders are used, the application traffic cannot cross the dual-homed firewall unless the application forwarder is running and configured on the firewall machine. This is an implementation of the policy "That which is not expressly permitted is prohibited." If users are allowed to log in to the firewall directly (see fig. 8.6), the firewall security can be compromised. This is because the dual-homed firewall is a central point of connection between the external network and the internal network. By definition, the dual-homed firewall is in the zone of risk. If the user selects a weak password, or allows his user account to be compromised (such as by giving away passwords), the zone of risk could extend to the internal network, thus defeating the purpose of the dual-homed firewall.

Figure 8.5

A dual-homed host with application forwarders.

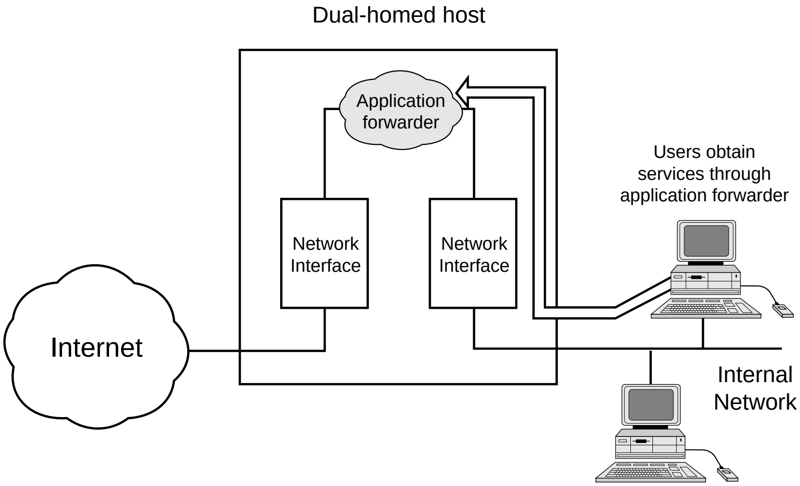
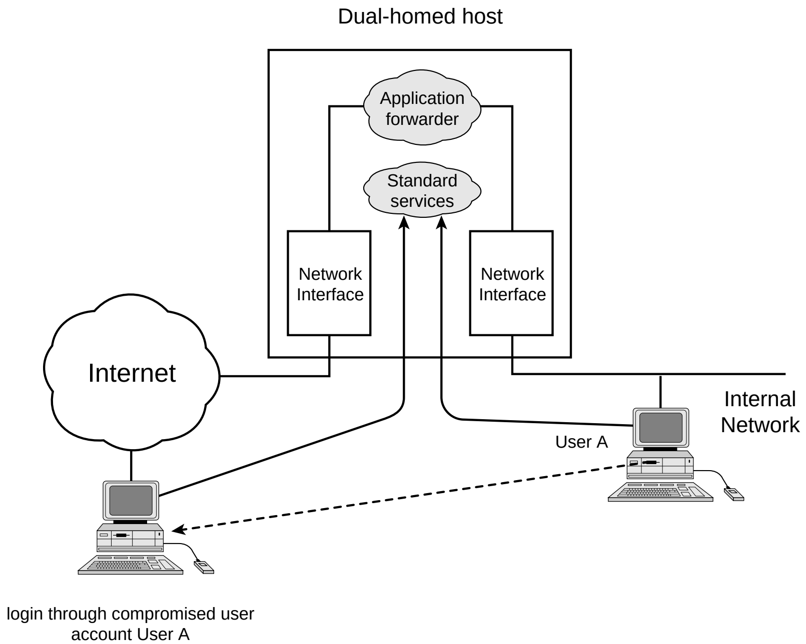


Figure 8.6

Insecurity introduced with standard user login to dual-homed host.



Warning

The wise security manager will forbid the creation of user accounts to access the firewall. The firewall should be used only to authenticate users to allow their sessions to pass through the firewall.

If proper logs are kept of user logins, it is possible to trace unauthorized logins to the firewall when a security breach is discovered. If users are not permitted direct login to the dual-homed firewall, however, any attempt at a direct user login will be registered as a noteworthy event and a potential security breach.

Examples of store-and-forward services are SMTP (mail) and NNTP (news). Figure 8.7 shows a situation where the dual-homed host is configured to provide discretionary forwarding of mail messages between an external untrusted network and an internal network. Figure 8.8 shows a situation where the dual-homed host is configured to provide discretionary forwarding of news messages between news servers on the external untrusted network and the internal network.

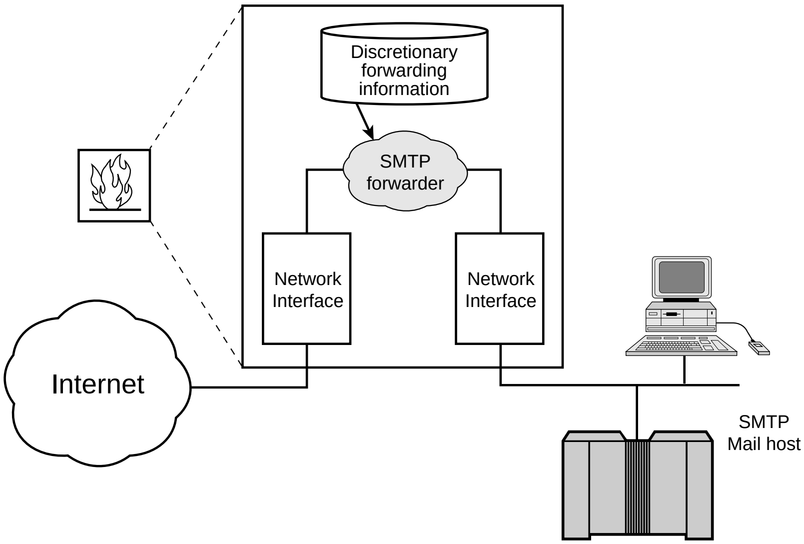


FIGURE 8.7

A dual-homed host as a mail forwarder.

The dual-homed host is the basic configuration used in firewalls. The critical aspect of dual-homed firewall hosts is that routing is disabled, and that the only path between the network segments is through an application layer function. If the routing is accidentally (or by design) misconfigured so IP forwarding is enabled, it is possible that the application layer functions of the dual-homed firewalls are bypassed (see fig. 8.9).

Most firewalls are built around Unix machines. On some Unix implementations, the routing functions are enabled by default. It is therefore important to verify that the routing functions in the dual-homed firewall are disabled, and if they are not, you should know how to disable them.

FIGURE 8.8

A dual-homed host as a news forwarder.

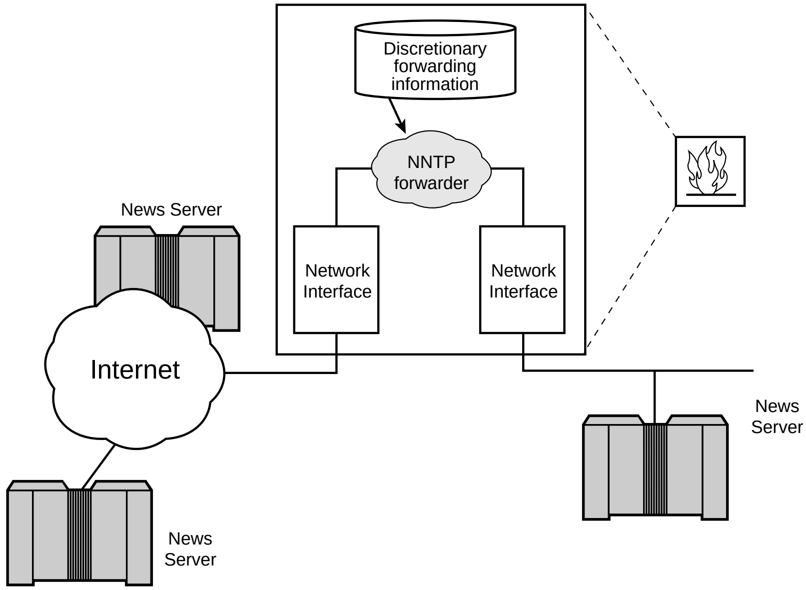
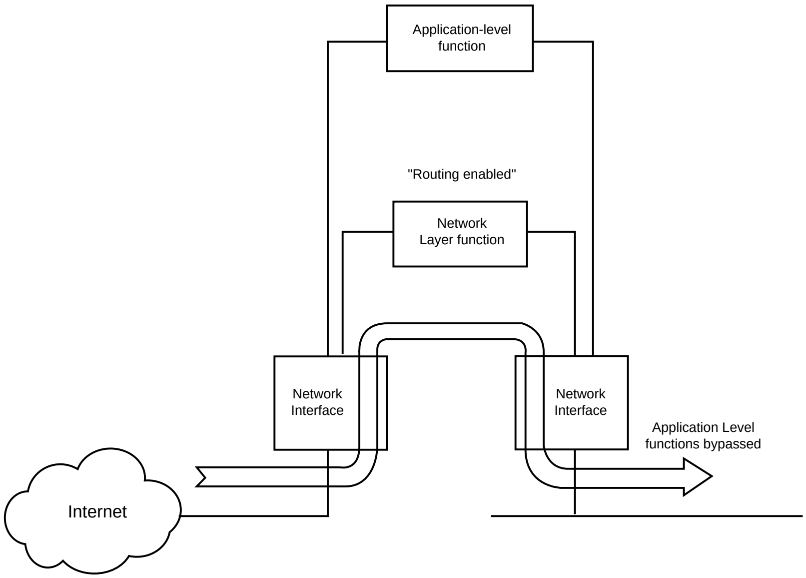


FIGURE 8.9

A misconfigured dual-homed firewall.



DISABLING ROUTING IN A DUAL-HOMED FIREWALL

Disabling routing functions in a dual-homed host based on Unix requires reconfiguring and rebuilding the kernel. This section describes that process for BSD Unix.

The Unix kernel is compiled using the make command. A command called config is used to read the kernel configuring file and generate the files that are needed to build the kernel. The kernel configuration file is in /usr/sys/conf or /usr/src/sys. In BSDI (Berkeley Systems Design, Inc.) Unix for the Intel platform, the configuration file is in /usr/src/sys/i386/conf directory.

To check to see which kernel configuration you are using, you can use the strings command on the kernel image file and look for the name of your operating system. The following shows an example of how you can check the kernel version.

```
% strings  /bsd ¦ grep BSD
BSDI $Id: if_pe.c,v 1.4 1993/02/21 20:35:01 karels Exp $
BSDI $Id: if_petbl.c,v 1.2 1993/02/21 20:36:09 karels Exp $
BSD/386
@(#)BSDI BSD/386 1.0 Kernel #0: Wed Mar 24 17:23:44 MST 1993
    polk@hilltop.BSDI.COM:/home/hilltop/polk/sys.clean/compile/GENERIC
```

The last line reveals that the current configuration is GENERIC.

You should change to the configuration directory (/usr/src/sys/i386/conf) and copy this GENERIC file into a new file suggestive of the new configuration. You can call this new file FIREWALL or LOCAL, for example.

```
cd /usr/src/sys/i386/conf
cp GENERIC FIREWALL
```

Next, edit the options parameter IPFORWARDING in the file FIREWALL and change it to –1 to represent "never forward IP datagrams." This variable has the affect of setting the *ipforwarding* kernel variable so that IP forwarding is disabled.

```
options IPFORWARDING=-1
```

On some systems, instead of the IPFORWARDING parameter, you might see the following statement in the configuration file.

```
options  GATEWAY
```

To disable forwarding of IP packets, comment this line out by placing a # as the first character in the line.

```
#options  GATEWAY
```

Also, verify that the following kernel configuration statements for TCP/IP exist.

```
options     INET                # Internet Protocol support is to be included
pseudo-device   loop            # The loop back device is to be defined
                                  (127.0.0.1)
pseudo-device   ether           # Generic Ethernet support such as ARP
                                  functions
pseudo-device   pty             # Pseudo teletypes for telnet/rlogin access
device we0 at isa? port 0×280  # Could be different for your Ethernet
                                  interface
```

Run the config command to build the LOCAL directory and go to the directory shown in the following example:

```
config LOCAL
cd ../../compile/LOCAL
```

Next, run the make commands to create the necessary dependencies and build the kernel.

```
make depend
make
```

Copy the kernel image to the root directory, and then reboot.

```
cp /bsd /bsd.old
cp bsd /bsd
reboot
```

After rebooting, the machine can be set up as a dual-homed firewall.

Compromising the Security of a Dual-Homed Firewall

You should understand how the integrity of a dual-homed firewall can be compromised. With this understanding, then, you can take steps to prevent such an occurrence.

The biggest threat is if an intruder obtains direct login access to the dual-homed host. Login should always occur through an application proxy on the dual-homed host. Logins from external untrusted networks should require a strong authentication.

Warning

> The only access to the firewall itself should be either through the console or secure remote access. To prevent the circumvention of the firewall, no user accounts should be permitted on the system.

If the user obtains login access to the dual-homed host, the internal network is subject to intrusions. These intrusions can come through any of the following sources:

- Weak permissions on the file system
- Internal network NFS-mounted volumes
- Permissions granted to Berkeley r*-utilities through host equivalent files, such as .rhosts, in users home directories for user accounts that have been compromised
- Network backup programs that could restore excessive permissions
- The use of administrative shell scripts that have not been properly secured
- Learning about the system from older software revision levels and release notes that have not been properly secured
- Installing older operating system kernels that have IP forwarding enabled, or installing versions of older operating system kernels with known security problems

- The use of sniffing programs such as tcpdump or etherfind to "sniff" the internal network looking for user name and password information.

If the dual-homed host fails, the internal network is wide-open for future intruders, unless the problem is detected and corrected quickly.

As mentioned previously, the Unix kernel variable *ipforwarding* controls whether IP routing is performed. If the intruder gains sufficient system privileges, the intruder can change the value of this kernel variable and enable IP forwarding. With IP forwarding enabled, the firewall mechanism is bypassed.

Services on a Dual-Homed Firewall

Besides disabling IP forwarding, you should remove from the dual-homed firewall all programs, utilities, and services that could be dangerous in the hands of an intruder. The following is a partial list of some useful checkpoints for Unix dual-homed firewalls:

- Remove programming tools: compilers, linkers, and so forth.
- Remove programs with SUID and SGID permissions that you do not need or do not understand (see Chapter 2, "Security"). If things do not work, you can always put back essential programs. If you have the experience, build a disk space monitor that will shut down the dual-homed host should a critical disk partition become full.
- Use disk partitions so that an intrusion to fill all disk space on the partition will be confined to that partition.
- Remove unneeded system and special accounts.
- Delete network services that are not needed. Use the netstat -a command to verify that you only have network services that you need. Edit the /etc/inetd.conf and /etc/services files and remove unneeded service definitions.
- Alter the system startup scripts to prevent the initialization of unneeded programs such as routed/gated and any routing support programs.

Bastion Hosts

A *bastion host* is any firewall host that is critical to the network security. The bastion host is the central host in an organization's network security. Because the bastion host is critical to network security, it must be well-fortified. This means that the bastion host is closely monitored by the network administrators. The bastion host software and system security should undergo regular audits. The access logs should be examined for any potential security breaches and any attempts to assault the bastion host.

The dual-homed host discussed earlier is an example of a bastion host because it is critical to the security of the network.

Simplest Deployment of a Bastion Host

Because bastion hosts act as an interface point to an external untrusted network, they are often subject to intrusion. The simplest deployment of a bastion host is as the first and only point of entry for external network traffic (see fig. 8.10).

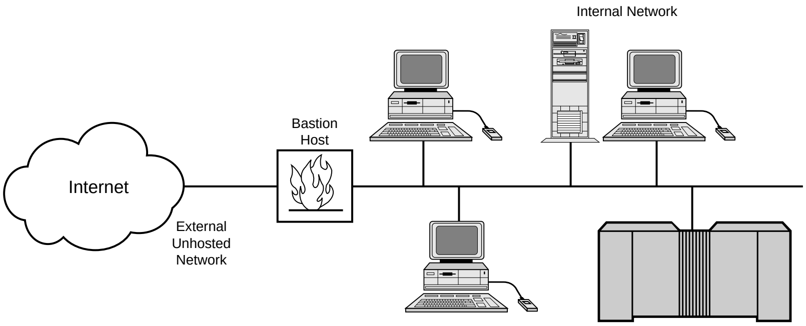


FIGURE 8.10

The simplest deployment of a bastion host (B2 configuration).

Notation Used to Describe Bastion Configurations

The author has devised a notation that can simplify the description of a firewall configuration. In this notation, symbols are used to have the following meanings:

Symbol	*Description*
S	Screening router
R	Ordinary router
F1	Firewall with single network connection to the network
F2	Firewall with two network connections
B1	Bastion host with single network connection to the network
B2	Bastion host with two network connections

Using these symbols, you can describe the network configuration in figure 8.10. You can follow the path of the network traffic from the external to the internal networks:

B2

So the network in figure 8.10 is a B2 configuration.

Screened Host Gateway

Because the bastion host is critical to the security of the internal network, another first line of defense is often introduced between the external untrusted network and the internal network. This first line of defense is usually provided by a screening router. Figure 8.11 shows the use of a bastion host with a screening router as the first line of defense. In this example, only the network interface of the bastion host is configured, and this network interface is connected to the internal network. One of the ports of the screening router is connected to the internal network, and the other port is connected to the Internet. This type of configuration is called the *screened host gateway*.

Using the notation defined in this chapter, the screened host gateway configuration shown in figure 8.11 can be described as the S-B1 configuration or just "SB1."

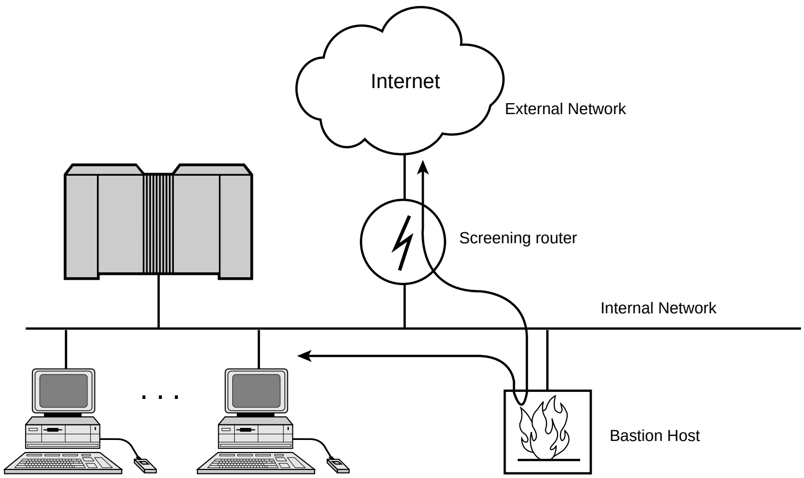


FIGURE 8.11

A bastion host with single network interface and screening router as the first line of defense (SB1 configuration).

You must configure the screening router so that it sends all traffic received from the external networks for the internal network to the bastion host first. Before it forwards traffic to the bastion host, the screening router will apply its filter rules to the packet traffic. Only network traffic that passes the filter rules is diverted to the bastion host; all other network traffic is rejected. This architecture gives a level of confidence in the network security that is missing in figure 8.10. An intruder must first penetrate the screening router. If the intruder manages to penetrate the screening router, he must contend with the bastion host.

The bastion host uses application-level functions to determine if requests to and from the external network are permitted or denied. If the request passes the scrutiny of the bastion host, it is forwarded to the internal network for incoming traffic. For outgoing traffic (traffic to the external network), the requests are forwarded to the screening router. Figure 8.12 shows the path of network traffic between the external and internal networks.

OFF-LOADING PACKET FILTERING TO THE IAP

Some organizations prefer to have their Internet Access Provider (IAP) provide packet filter rules for network traffic sent to the organization's network (see fig. 8.13). The packet filter still acts as the first line of defense, but you have to rely on your IAP for the correct maintenance of the packet filter rules.

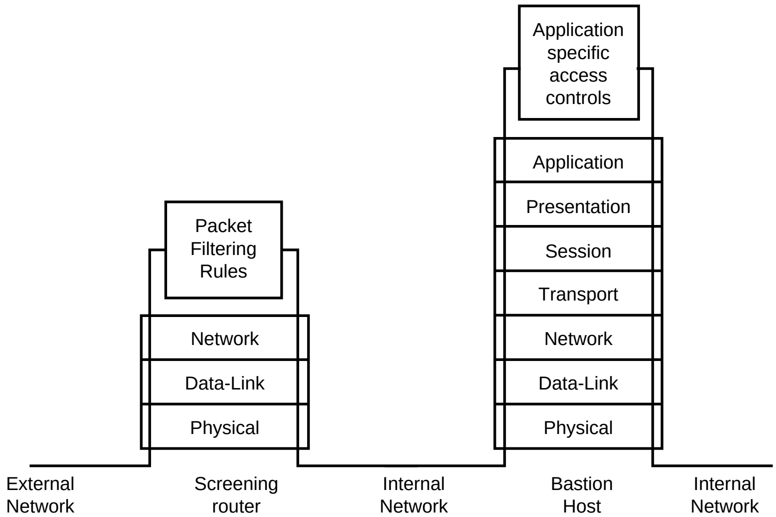


FIGURE 8.12

The path of network traffic for network with screening router and bastion host.

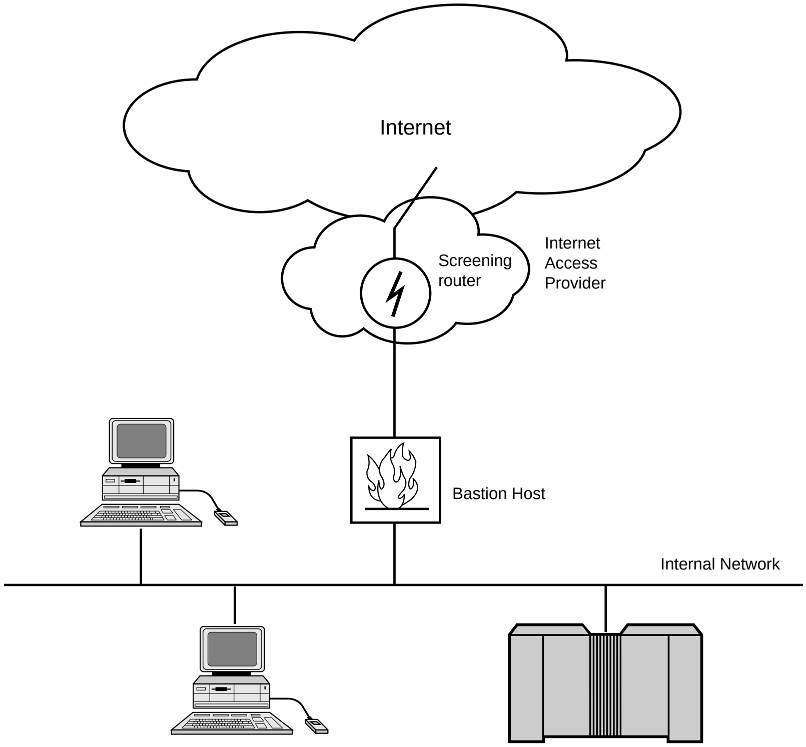


FIGURE 8.13

The IAP provides packet filtering to the internal network.

ROUTING CONFIGURATION FOR SCREENED HOST GATEWAY NETWORKS

The routing tables of the screening routers must be configured so that external traffic is forwarded to the bastion host. The routing tables in the screening router should be protected from intrusion and unauthorized change. If the routing table entry is changed so that the traffic is not forwarded to the bastion host but sent directly to the locally connected network, the bastion host is bypassed.

Figure 8.14 shows a situation where the screening router's routing table points to the bastion host. The internal network number is 199.245.180.0, and the bastion hosts IP address is 199.245.180.10. The screening router has the following entry in its routing table.

Destination = 199.245.180.0

Forward to = 199.245.180.10

All network traffic for network 199.245.180.0 is forwarded to the bastion host's IP address of 199.245.180.10.

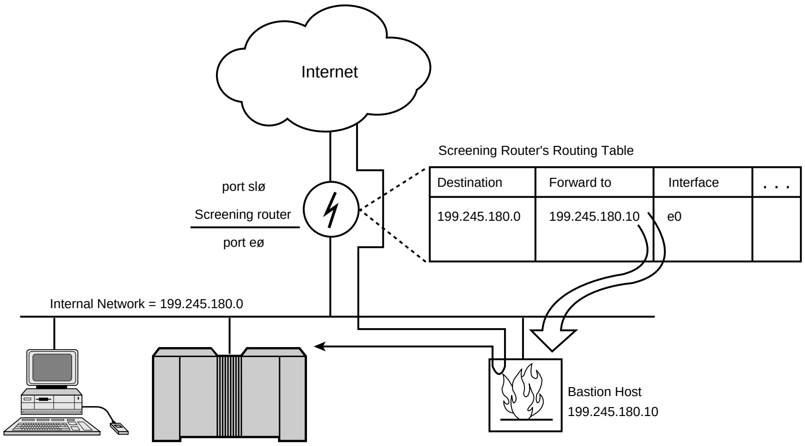


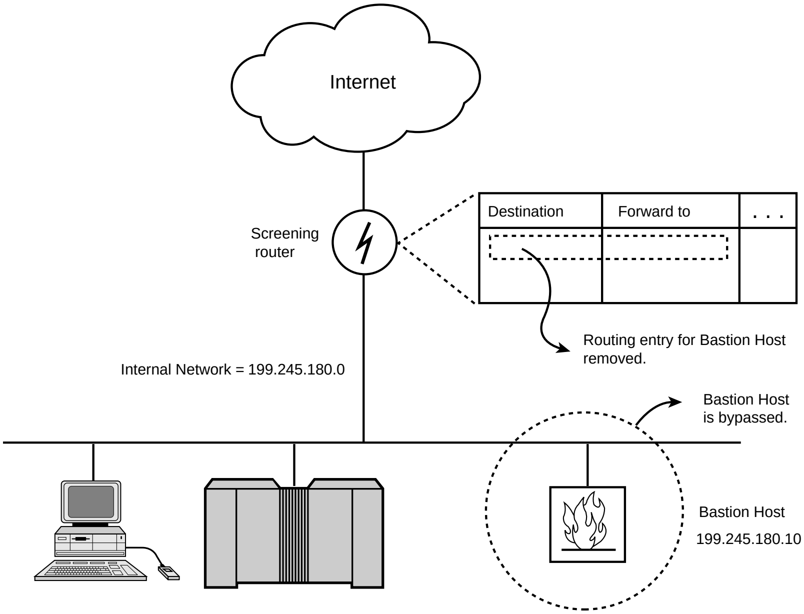
FIGURE 8.14

The normal setting of the screening router's routing table.

Figure 8.15 shows a situation where the screening router's routing tables have been subverted, and the entry for the destination network 199.245.180.0 has been removed. External traffic received by the screening router for the network 199.245.180.0 is not sent to the bastion host, but is sent directly through the local interface on the internal network. The bastion host is bypassed and the screening router is the only line of defense. Chances are that if the screening router has been subverted, other functions of the router could also be subverted, and the zone of risk will encompass the internal network.

FIGURE 8.15

The screening router's routing table is subverted.



If the screening router responds to ICMP (Internet Control Message Protocol) redirect messages, it is vulnerable to false ICMP messages sent by the intruder. For this reason, response to ICMP redirect messages should be disabled.

NOTE

> If the screening router is based on Unix-derived software, you should configure the kernel to disable directed broadcasts so that broadcast packets from the internal network are not forwarded to the external network. Otherwise, information on the internal network can leak to the outside world. You can configure the kernel to disable directed broadcasts by setting the following in the configuration file:
>
> ```
> options DIRECTED_BROADCAST=0
> ```
>
> Next, you must run config on the configuration file. After this, you must run the make commands as follows:
>
> config *nameoffile*
>
> make depend
>
> make

> You should also remove unneeded network services and use static routing. Especially ensure that "routed" or "gated" daemons are not running, otherwise routes will be advertised to the external world.
>
> Also, make permanent entries in the ARP cache table to point to the bastion host. Use the following command to make permanent entries in the ARP cache.
>
> ```
> arp -s ipaddress hardware-address
> ```
>
> For example, to specify that the hardware address of the bastion host 199.245.180.10 is 0000C005564A, use the following command:
>
> ```
> arp -s 199.245.180.10 0000C004564A
> ```

You should configure the screening routers to use static routing. Static routes are not expired and changed by routing protocols. This protects the static routes from false route advertisements.

You also should disable the processing of the following at the router: ARP, ICMP redirects, proxy ARP, and ICMP unreachable messages. You can use the following configuration statements, for example, on a Cisco router.

```
no ip redirects
no ip route-cache
no ip proxy-arp
no ip unreachables
no service finger
```

If your screening router supports TELNET access, you should disable it. On a Cisco router, you can set access control lists on virtual terminals to prevent remote access through TELNET.

In normal ARP operation, ARP table entries are built dynamically and are expired after a predetermined time interval. You should manually initialize the ARP cache table for the router and the bastion host. ARP entries that are made manually are never expired and act as "static" entries. With ARP processing disabled at the router, the router will not give out its hardware address.

NOTE

In BSD Unix, static routes are set up using the route command. The general syntax of this command is shown in the following example.

```
route [-n] [-q] [-v] command [-net ¦ -host] destination forward_to
[count]
```

The flags in this example have the following meanings:

-n	Prevents attempts to print symbolic names when reporting actions
-v	(verbose) Prints additional details
-q	Suppresses all output

The *command* can have the following values:

add	Adds a route
flush	Removes all routes
delete	Deletes a specific route
change	Changes aspects of a route (such as its gateway)
get	Looks up and displays the route for a destination
monitor	Continuously reports any changes to the routing information base, routing lookup misses, or suspected network partionings

To forward network traffic for the network 199.245.180.0 to the bastion host 199.245.180.10, for example, you would use the following command.

```
route add -net 199.245.180.0  199.245.180.10
```

It is best to specify all four numeric values of the dotted decimal IP address, unless you understand how partial IP addresses are interpreted. Here are some examples:

Using 130.33 without the -host or -net is interpreted as:

-host 130.0.0.33

The address 130.33.5 is interpreted as:

-host 130.33.0.5

The address -net 130.33 is interpreted as:

130.33.0.0

The address -net 130.33.5 is interpreted as:

130.33..5

The *count* keyword, which is optional on some systems and required on others, is used to indicate whether the host is local or remote. If the count is 1 or more, then the forward to is a remote system. If the value is 0, then the forward to is another interface on the same system.

BASTION HOST WITH BOTH NETWORK INTERFACES CONFIGURED

Figure 8.16 shows the use of a bastion host with a screening router, but both network interfaces of the bastion host are configured. One network interface is connected to the "outside" network and the other network interface is connected to the "inside" network. One of the ports of the screening router is connected to the "inside" network and the other port is connected to the Internet. Using the notation defined in this chapter, figure 8.16 can be described as the S-B2 configuration or just as "SB2."

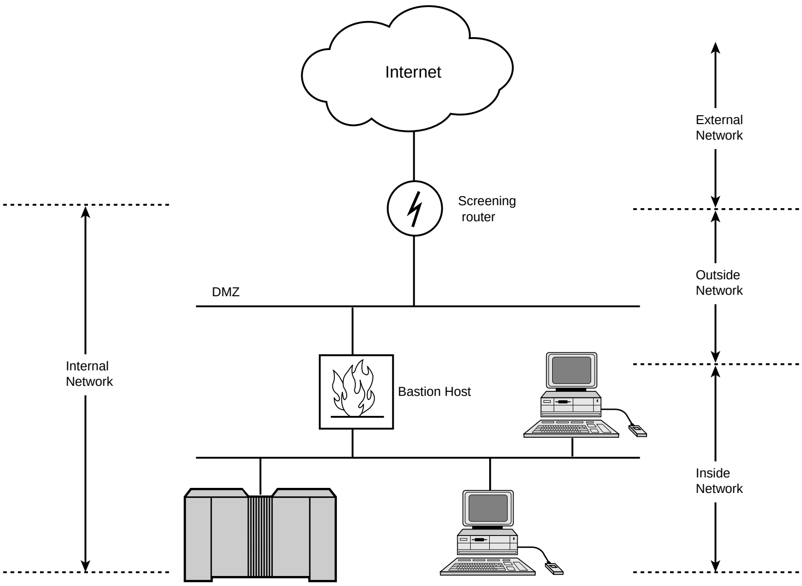


FIGURE 8.16

A bastion host with two network interfaces and a screening router as the first line of defense (SB2 configuration).

The screening router must be configured so that it sends all traffic received from the external networks for the internal network to the "inside" network interface of the bastion host. Before it forwards traffic to the bastion host, the screening router will apply its filter rules to the packet traffic. Only network traffic that passes the filter rules is diverted to the bastion host; all other network traffic is rejected. An intruder must first penetrate the screening router. If the intruder manages to penetrate the screening router, he must contend with the bastion host.

There are no hosts on the outside network, other than the screening router and one of the network interfaces of the bastion host. The outside network forms a Demilitarized Zone (DMZ). Because the DMZ has only two network connections, it can be replaced by a dedicated point-to-point link. This makes it more difficult to tap into this link using protocol analyzers. If an Ethernet or token ring network is used for the DMZ, a workstation that places its network interface in the promiscuous mode can capture network traffic and access sensitive data. Normally, network interfaces only read the packet directly addressed to it. In the promiscuous mode, however, network interfaces read all packets seen by the network interface. All the organization's hosts (except for the bastion host) are connected to the inside network.

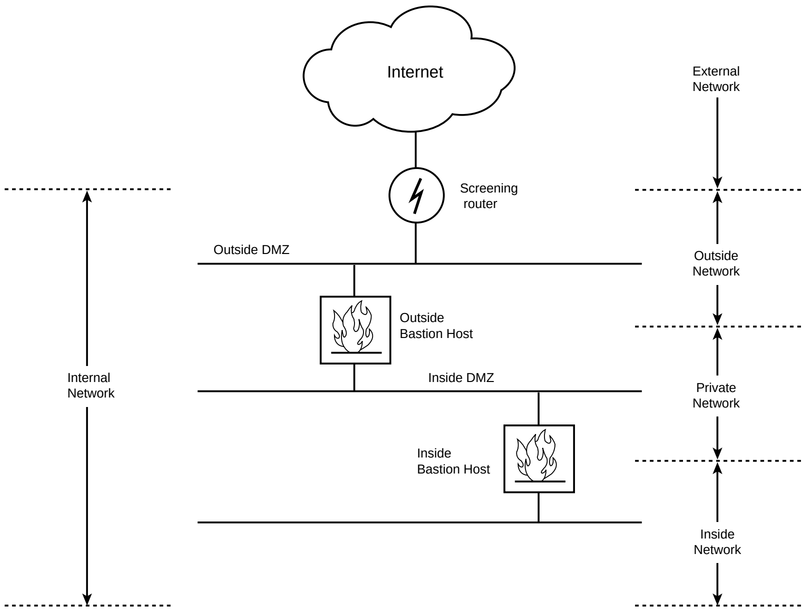
The network configuration in figure 8.16 has another advantage over the network configuration in which only one network interface of the bastion host was used (refer to fig. 8.11). This advantage is that the bastion host cannot be bypassed by attacking the screening routers routing tables. Network traffic must pass through the bastion host to reach the inside network.

Uses of Two Bastion Hosts and Two DMZs

Figure 8.17 shows the use of two bastion hosts with a screening router. Both network interfaces of the two bastion hosts are configured. Three network zones are formed in the internal network: the outside network, the private network, and the inside network. Using the notation defined in this chapter, figure 8.17 can be described as the S-B2-B2 configuration, or just "SB2B2."

Figure 8.17

Two bastion hosts with both network interface cards configured (SB2B2 configuration).



The screening router and the outside bastion host are the only two network interfaces on the outside network. The outside network forms the outside DMZ.

A private network exists between the inside and outside bastions. The private network provides a level of protection similar to that in figure 8.18. An organization could place some of its hosts on the private network and keep the more sensitive hosts behind the inside bastion host. Alternatively, an organization may want maximum security and use the private network as a second buffer zone or inside DMZ, and keep all of the hosts on the inside network.

If an organization wants to provide full access to a wide array of services, such as anonymous FTP (File Transfer Protocol), Gopher, and WWW (World Wide Web) services, it can provide certain sacrificial hosts on the outside DMZ (see fig. 8.18). The bastion hosts should not trust any traffic originating from these sacrificial hosts.

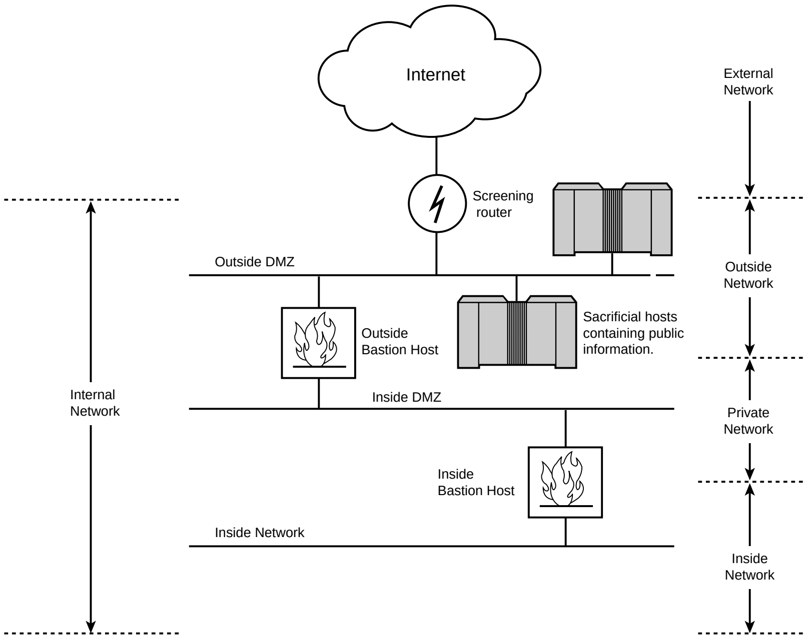


Figure 8.18

Sacrificial hosts on the outside DMZ.

The screening router must be configured so that it sends all traffic received from the external networks for the internal network to the inside bastion host. Before it forwards traffic to the bastion host, the screening router will apply its filter rules to the packet traffic. Only network traffic that passes the filter rules is diverted to the outside bastion host; all other network traffic is rejected. An intruder must first penetrate the screening router. If the intruder manages to penetrate the screening router, he must contend with the outside bastion host.

Even if the defenses of the outside network are breached, the intruder must penetrate the inside bastion host. If resources permit, you might want to make each bastion host the responsibility of a different administrative group. This ensures that the mistakes committed by one set of administrators will not be repeated by the other administrators. You also should ensure that the two groups share information about discovered weaknesses in the bastion hosts. Figure 8.19 shows the path that the network traffic has to take between the external and internal networks.

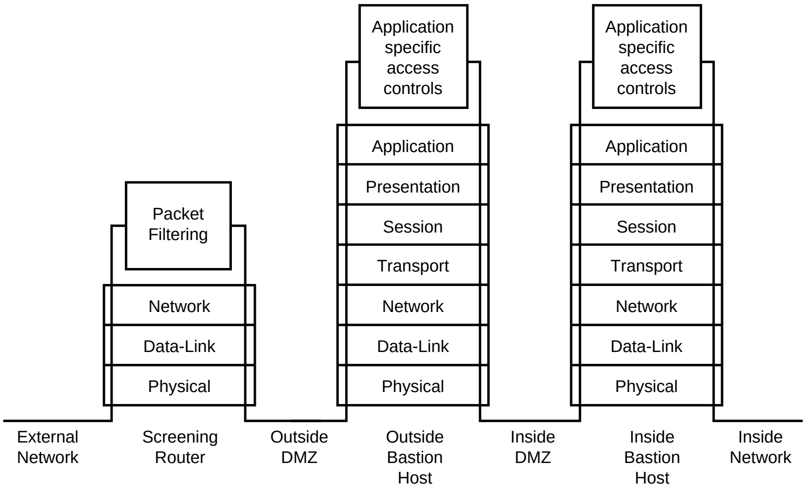


Figure 8.19

The path of network traffic for the network in figure 8.17.

Another type of network configuration is to use two bastion hosts but only one network interface of each bastion host (see fig. 8.20). A second router called the *choke* is added between the DMZ and the inside networks. Figure 8.21 shows the path taken by network traffic between the external and internal networks.

With the network configuration of figure 8.20, you should ensure that the bastion hosts are not bypassed. You should ensure that the screening routers are using static routes. Using the notation defined in this chapter, figure 8.20 can be described as the S-B1-S-B1 configuration, or just "SB1SB1."

Other possible combinations are shown in figures 8.22 and 8.23. Whenever only a single network interface of the bastion is used, you should use static routes at the routers and properly configure the routing table entries to ensure that the bastion hosts are not bypassed. Using the notation defined in this chapter, figure 8.22 can be described as the S-B2-B1 configuration, or just "SB2B1;" figure 8.23 can be described as the S-B1-B2, or just "SB1B2."

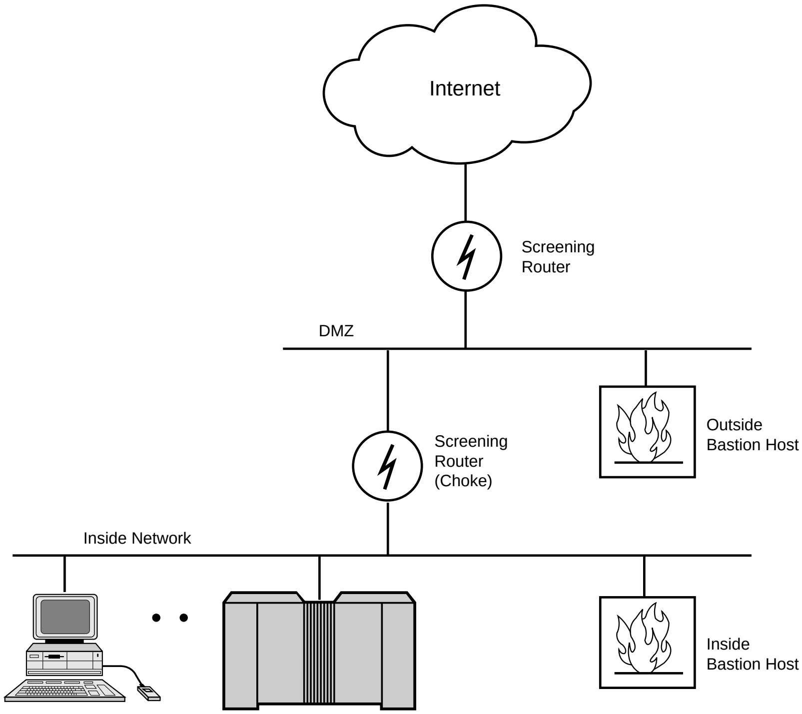


Figure 8.20

Bastion hosts using single network interface (SB1SB1 configuration).

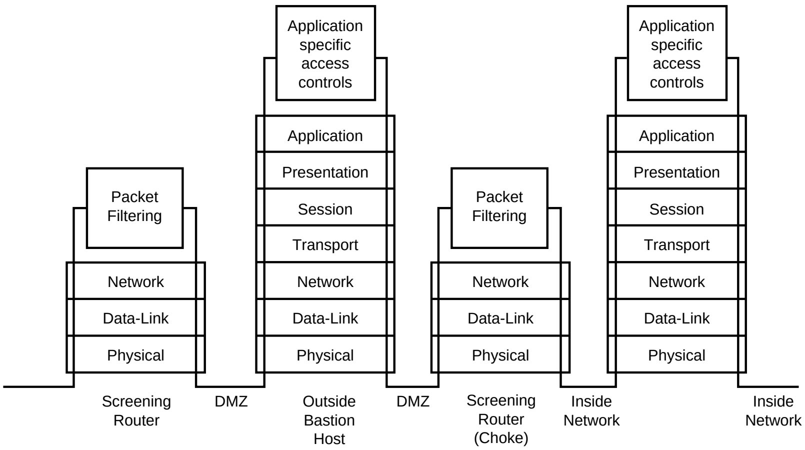


Figure 8.21

The path of network traffic for the network in figure 8.20.

FIGURE 8.22

A double-ended/ single-ended bastion configuration (SB2B1 configuration).

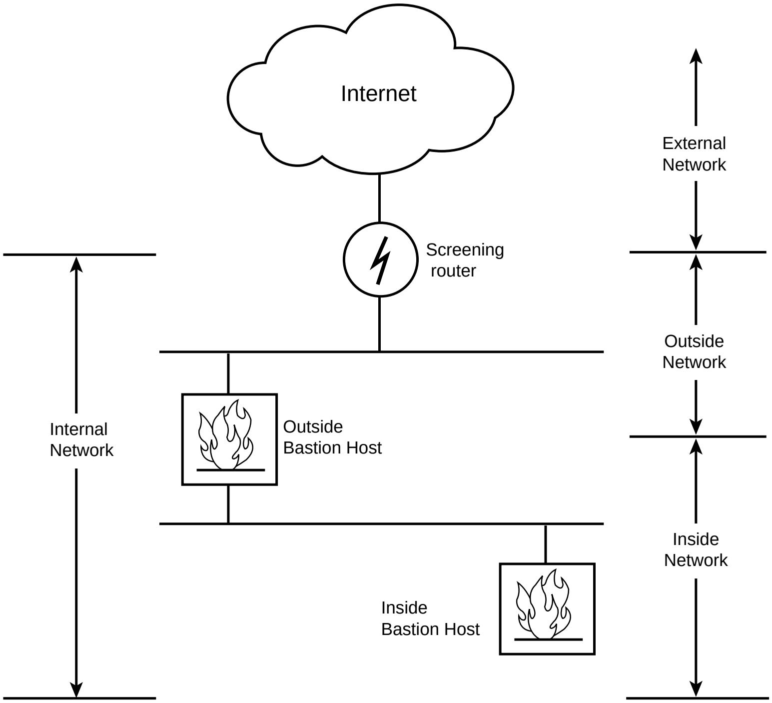
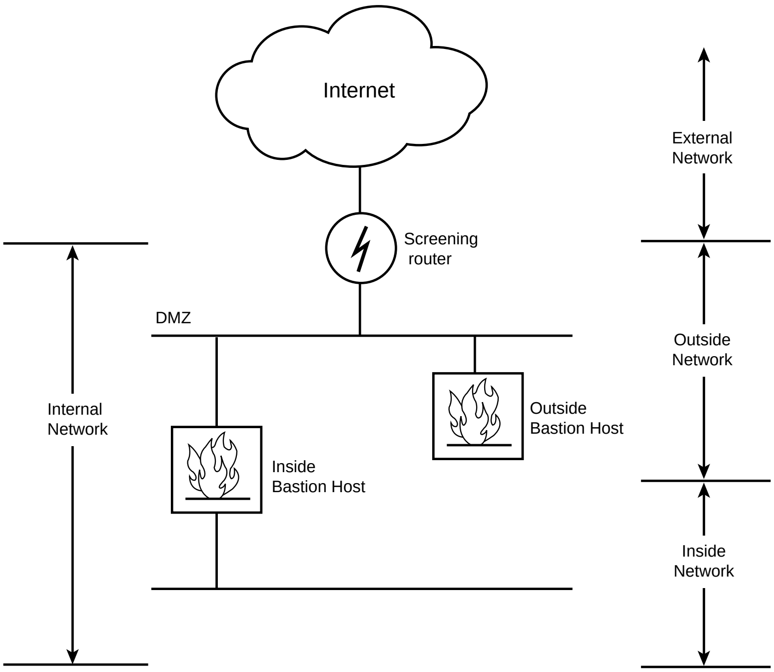


FIGURE 8.23

A single-ended/ double-ended bastion configuration (SB1B2 configuration).



Screened Subnets

In some firewall configurations, an isolated network of the type shown in figure 8.24 can be created. In this network, both the untrusted external network and the internal network can access the isolated network. However, no network traffic can flow between the untrusted external network and the internal network *through the isolated network.* The isolation of the network is performed using a combination of screening routers that are properly configured (see fig. 8.25). Such an isolated network is called a screened subnet.

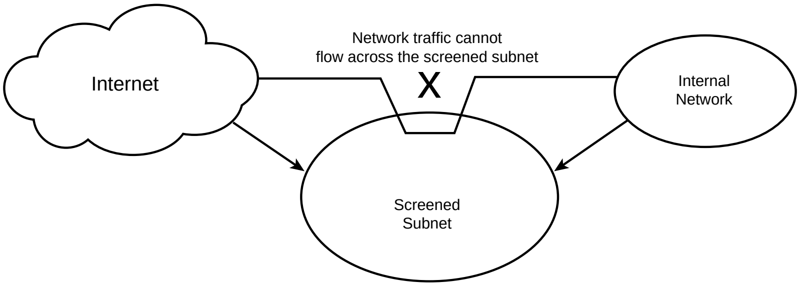


Figure 8.24

A screened subnet.

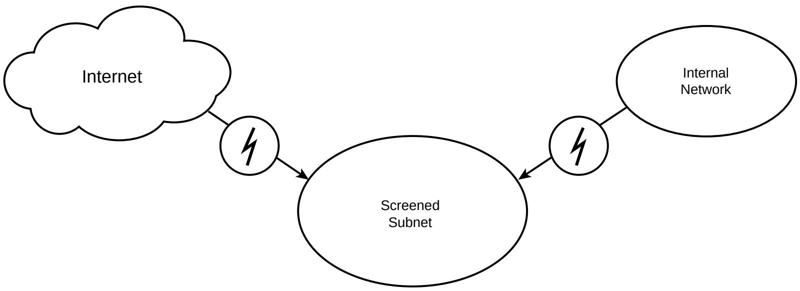


Figure 8.25

A screened subnet implemented using screening routers.

Some screened subnets may have application-level gateways on them that act as bastion hosts and provide interactive access to outside services (see fig. 8.26).

Figure 8.27 shows a screened subnet configured with a bastion host as the central access point to the screened subnet. This type of configuration is the S-B1-S, or "SB1S" configuration. Screening routers are used to connect the Internet and the internal network. The bastion host is an application gateway and denies all traffic that is not expressly permitted.

Because the only access to the screened subnet is through the bastion host, it is very difficult for an intruder to breach the screened subnet. If the intrusion comes through the Internet, the intruder would have to reconfigure the routing on the Internet, the screened subnet, and the internal network to have free access (which is made difficult if the screening routers allow access to only specific hosts). Even if the bastion host is breached, the intruder would have to break into one of the hosts on the internal network and then back into the screening router to access the screened subnet. This type of island-hopping intrusion is difficult to do without disconnecting oneself or tripping over an alarm.

Figure 8.26

Screened subnets with bastion hosts on the screened subnet.

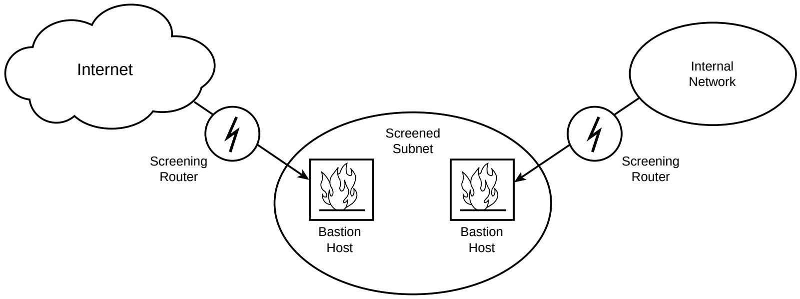
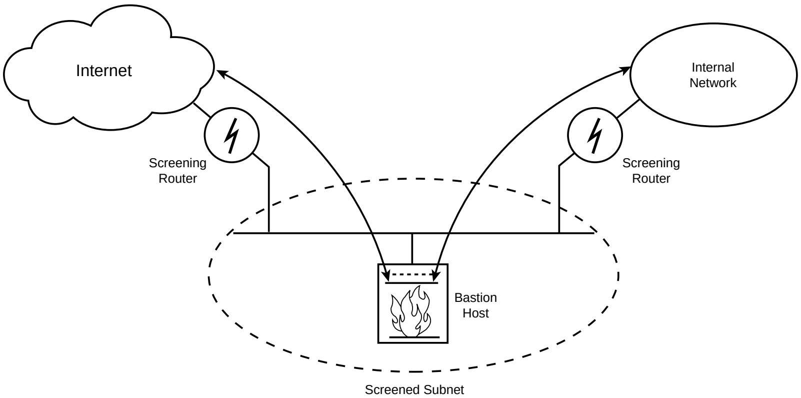


Figure 8.27

A screened subnet using the "SB1S" configuration.



Because screened subnets do not allow network traffic to flow between the Internet and internal network, the IP addresses of the hosts on these networks are hidden from each other. This allows an organization that has not converted over to officially assigned network numbers from the NIC (Network Information Center) to access the Internet through application gateway services provided by the bastion host on the screened subnet. If these services through the application gateway are restrictive, these restrictions can act as a spur to convert the internal network to officially assigned network numbers.

Application-Level Gateways

Application-level gateways can handle store-and-forward traffic as well as some interactive traffic (see fig. 8.28). Application-level gateways are programmed to understand the traffic at the user application level (layer 7 of the OSI model). They can therefore provide access controls at a user level and application protocol level. Moreover, they can be used to maintain an intelligent log of all usage of the applications. The ability to log and control all incoming and outgoing traffic is one of the main advantages of having an application-level gateway. The gateways themselves can have additional security built into them as needed.

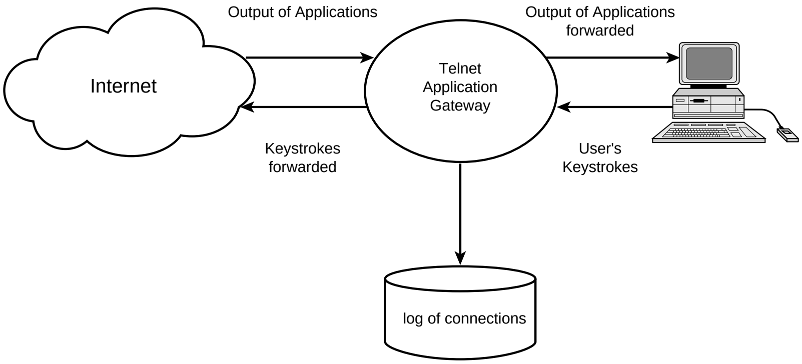


FIGURE 8.28

Application-level gateways.

For each application that is relayed, application-level gateways use a special-purpose code. Because of this special-purpose code, application gateways provide a high level of security. For each new type of application that is added to the network and that requires protection, new special-purpose code has to be written. Therefore, most application-level gateways provide a limited subset of basic applications and services.

To use application-level gateways, users must log in to the application gateway machine, or implement a specific client application service on every host that will utilize this service. Each application-specific gateway module can have its own set of management tools and command language.

NOTE

Some applications will have trouble interacting with the application gateway. If your application, such as a WWW browser, is capable of handling the challenge that will be presented to you, then your connection will succeed. If your application does not have support to deal with the challenge, then you will need to either use a special client application or authenticate your session through alternate means.

A disadvantage of application-level gateways is that a custom program often has to be written for each application. This fact is also an advantage from a security view point, though, because you cannot go through the firewall unless an explicit application-level gateway has been provided. This is an implementation of the philosophy, *That which is not expressly permitted is prohibited.*

The custom application program acts a "proxy" that accepts incoming calls and checks them against an access list of what types of requests are permitted. The proxy in this case is an application server proxy. On receiving the call—and after verifying that the call is permitted—the proxy forwards the request to the requested server. The proxy, therefore, acts as both a server and a client (see fig 8.29). It acts as a server to receive the incoming request and as a client when forwarding the request. After the session is established, the application proxy acts as a relay and copies data between the client that initiated the application and the server. Because all data between the client and server is intercepted by the application proxy, it has full control over the session and can perform as detailed a log as needed. In figure 8.29, the proxy is shown as a client and server for the purposes of explaining its behavior. In most implementations, this is implemented by a single application module.

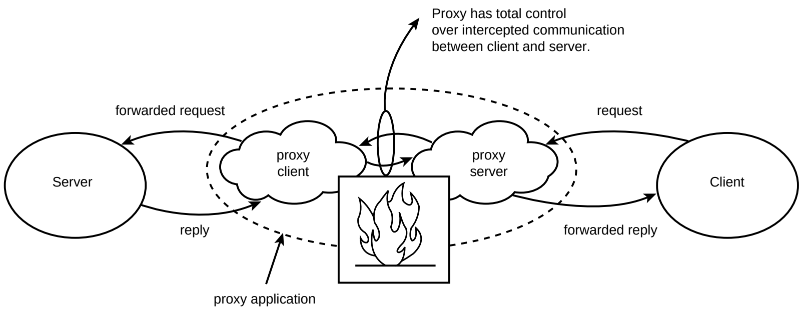


FIGURE 8.29

An application proxy as a client and server.

To connect to a proxy application, many application-level gateways require that you run a custom client application on your internal machines. Alternatively, you can use the telnet command to specify the port at which the proxy application service is available. If the proxy application was on host gatekeeper.kinetics.com and on port 63, for example, you would use the following command.

```
telnet gatekeeper.kinetics.com  63
```

After you connect to the port at which the proxy service runs, you should see a special prompt that identifies the proxy application. You have to run custom commands to specify the destination server. Regardless of which approach is used, the user interface to the standard application changes. If a custom client is used, the client is usually modified so that it always connects to your proxy machine and tells the proxy machine where to connect. The proxy machine then connects to the ultimate destination and passes the data through.

Some proxy application services are written so that they behave like the standard application. When the user specifies a connection target that is in a different network, the proxy application is invoked.

If you are using a custom client with the proxy application, you must install the custom client on all your internal machines that access a network through the application-level gateway. Depending on the size of the network, this can be a difficult task. If some of your users use DOS/Windows and Macintosh clients, it is usually the case that proxy versions of your client programs are not available. If you do not have the source code to these client applications (which is usually the case for Macintosh and PC client programs), you cannot modify them.

If the proxy client knows about only one application gateway server, and if this server is down, then you are vulnerable to a single point of failure. If the proxy client can be changed by the administrator to point to an alternate application gateway, the single point of failure problem can be avoided.

Because of the problems of configuring proxy clients, some sites prefer to use packet filtering for those applications such as FTP and TELNET that can be secured by proper filter rules. These sites use the proxy client approach for more complex applications, such as DNS, SMTP, NFS, HTTP, Gopher, and so forth.

If a custom client application is needed to communicate with the proxy server, some standard system calls such as connect() have to be replaced by a proxy version of these system calls. You must then compile and link the client application with the proxy versions of the system calls. A freely available library called *socks* contains nearly compatible replacements of the standard system calls such as socket(), bind(), connect(), and so on. This is available at the URL (Uniform Resource Locator) of ftp://ftp.inoc.dl.nec.com/pub/security/socks.cstc.

The proxy servers should be written in such a way as to provide a 'fail-safe' mode of operation if the properly modified client is not used. If a standard client is used to contact the proxy server, for example, then the communication should be prohibited and not cause undesirable and unpredictable behavior of the firewall and the screening routers.

Another type of application-level gateway is called the *circuit-gateway*. In circuit-level gateways, the packets are addressed to a user-application level process. A circuit-gateway is used to relay packets between the two communication end-points. The circuit-gateway simply copies the bytes back and forth between the two end points (see fig. 8.30).

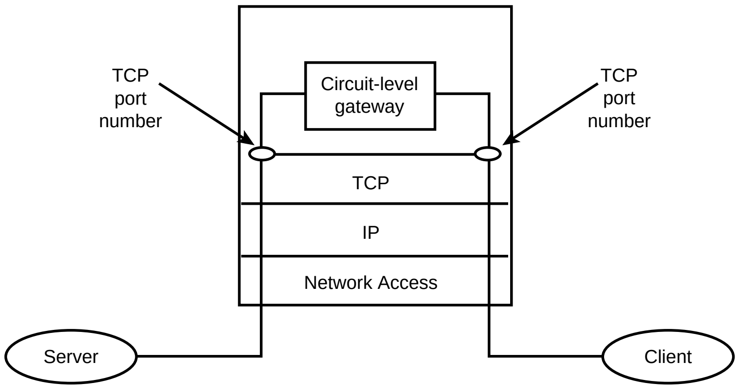


FIGURE 8.30

Circuit-level application gateways.

Circuit-level gateways are a more flexible and general approach to build application gateways. Although they may include code to support some specific TCP/IP application, this is usually limited. If they support an application, it is likely to be a TCP/IP application.

In circuit-level gateways, special client software might have to be installed, and the users might have to interact with an altered user interface or change their work habits. Installing and configuring special applications on each internal host can be time consuming and error prone for large heterogeneous networks because of the difference in hardware platforms and operating systems.

Because each packet is processed by software running at the application layer, the host performance is affected. Each packet is processed twice by all the communication layers and requires user-level processing and context switching. The application-level gateway (either a bastion-host or dual-homed host) remains exposed to the network. Other means such as packet filtering can be used to protect the application gateway host.

Summary

In this chapter, you learned about different network configurations that use firewalls and screening routers to provide enhanced network security. The advantages and disadvantages of the different network configurations were discussed.

Chapter 9, "Firewall Implementations," discusses practical implementations of firewall solutions. Some of these examples are taken from commercial firewall products and others are freely available software.

Chapter 9

FIREWALL IMPLEMENTATIONS

The previous chapter addressed firewall architecture and theory. This chapter completes the section by showing how those principles are implemented in the real world.

Several commercially available packages are discussed, as well as a shareware toolkit which allows you to construct your own firewall from scratch.

THE TCP WRAPPER

The TCP Wrapper is a freely available access control software for Unix systems. It is not a program that is run on a firewall. Rather it is used to protect the systems in the public network outside the firewall. The TCP Wrapper performs the following basic functions:

- Logs request for internet service made through the /etc/inetd.conf file
- Provides an access control mechanism to control access to the services

Both of these capabilities can be used to construct a simple firewall solution. The solution is simple because the TCP Wrapper does not provide an application or circuit-level gateway. It does provide, however, a series of tools intended to improve the level of auditing and security for the commonly available network utilities.

The current version of the TCP Wrapper program is 7.4 and is available at the following URL:

```
ftp://cert.sei.cmu.edu/pub/network_tools/tcp_wrapper*.shar
```

or

```
ftp://ftp.win.tue.nl/pub/network_tools/tcp_wrapper*.shar.Z
```

The file contains a shell archive. You must first uncompress and unarchive the files:

```
gunzip tcp_wrappers_6.3.shar.Z    # You can also use the Unix uncompress
sh tcp_wrappers_6.3.shar
make                              # You will see instructions on what to do
                                  # for building the binaries
```

The TCP Wrapper program is created in a file called *tcpd*. Place this file in a directory such as /usr/local/bin. Next, edit the /etc/inetd.conf file and place under the control of tcpd the services that you want to log and control. Suppose that you want to log and control telnet sessions to the host. The entry for telnet in the inetd.conf file should be similar to the following:

```
telnet    stream    tcp    nowait    root    /usr/libexec/telnetd    telnetd
```

To monitor access to telnet, replace the sixth field by the /usr/local/bin/tcpd program, as shown:

```
telnet    stream    tcp    nowait    root    /usr/local/bin/tcpd    telnetd
```

You can make similar changes to monitor any other services such as ftpd, tftpd, rlogind, rexecd, rshd, fingerd, and so forth.

After making the changes, restart inetd:

```
ucs# ps -aux ¦ grep inetd
root        82  0.0  0.0   652  124 ??  Is    3:17AM    0:00.27 inetd
ucs# kill -HUP 82
```

When the inetd program receives a request for the telnet service, the program tcpd is started, instead of the normal program telnetd. The tcpd program logs the request and checks the access control information in two files:

/etc/hosts.allow

/etc/hosts.deny

The hosts.allow file contains a list of hosts that are allowed access to the service. The hosts.deny file contains a list of hosts that are denied access. If these files do not exist, tcpd logs the request and allows all hosts access to the service. The tcpd program first checks the file hosts.allow, then the file hosts.deny. If a match is found for a service, it stops further examination of these files. This means that access granted through the hosts.allow cannot be overridden by the hosts.deny file.

The format of the entries for each file is as follows:

```
service-list: host-list [:shell-cmd]
```

The *service-list* is a comma-separated list of services defined in the /etc/services file.

The *host-list* is a comma-separated list of host names, IP addresses, network names, network numbers, domain names or NIS netgroups.

If a match occurs in the hosts.allow file, the listed systems are allowed access, and if a match occurs in the hosts.deny file, the listed systems are denied access.

Consider the following examples:

Example 1

Take following entry in the hosts.allow file:

```
fingerd, telnetd: 144.19.74.1, 144.20
```

This means that hosts 144.19.74.1 and all hosts on class B network 144.20.0.0 are allowed access to telnet and finger services.

Example 2

In the following entry in the hosts.deny file, all hosts in the domain HACKER.ORG are denied access to telnet and finger services:

```
fingerd, telnetd: .hacker.org
```

Example 3

The keyword ALL can be used to match any service or any host. The keyword LOCAL can be used to match any host on the locally connected network:

Note the following entry in the hosts.allow file:

```
ALL: LOCAL, .KINETICS.COM
```

This means that all hosts on the locally connected network that are in the domain KINETICS.COM are allowed access to *all* services.

Example 4

Consider the following entry in the hosts.deny file:

```
ALL: ALL
```

This means that all hosts on the network are denied access to all services, if there are no matching entries in the hosts.allow file. Remember that hosts.allows is processed before hosts.deny.

The definition of ALL is limited to those services that are invoked through inetd and are under the control of the tcpd wrapper program.

The FireWall-1 Gateway

The FireWall-1 is a commercial gateway product, available from Internet Security Corporation. The product currently runs on SUN SparcStations.

The FireWall-1 uses the following two methods to establish network security.

- Application gateway
- Packet filtering

The configuration of the FireWall-1 gateway is done using graphical interfaces such as OpenLook for SunOS operating systems. The FireWall-1 gateway provides the following features:

- Secure packet filtering
- Adding context information to stateless connections
- Auditing and alerting
- Ability to define and add new protocols and services

- Authenticated telnet and FTP sessions
- Creating encrypted channels

A unique aspect of FireWall-1 is that, although it uses packet filtering as its basic mechanism, the packet filtering is done at layers 2 to 7 of the OSI model. Protocols that lack context information, such as UDP, are handled by building context information within the FireWall-1 gateway.

Resource Requirements for FireWall-1

Before investing resources in evaluating a product to see if it is suitable for your network environment, it is helpful to know details such as the hardware and software and training resources needed to implement the product. This information is provided for your reference. The FireWall-1 can currently run on the following platform/operating systems:

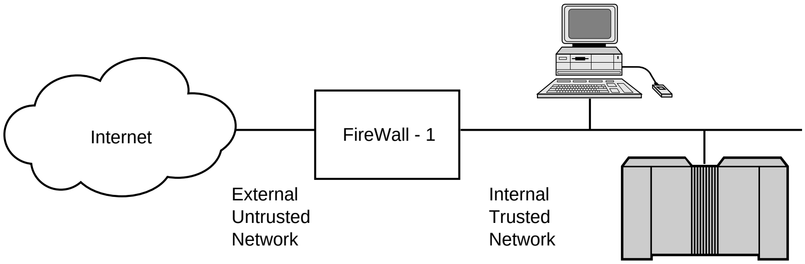
Hardware platform:	Sun Sparc-based systems
	Intel running Solaris 2.4 or higher
Operating System:	SunOS 4.1.3 or Solaris
	NetWare server platforms
Graphical Interface:	X11R5/Open Look
	(Open Windows 3)
Disk space requirements:	5 MB
Memory requirements:	16 MB control module
Network Interface:	Standard SUN workstation interfaces
	Access List support for Cisco routers (version 9.1 or higher) and Wellfleet routers (version 8.0 or higher)

Overview of FireWall-1 Architecture

The FireWall-1 gateway acts as a secure router between an organization's internal network and an external network (see fig. 9.1). All the network traffic between the organization's network and an external, untrusted network is sent through the FireWall-1 gateway. Each packet exchanged between the external network and internal networks thus can be verified to comply with the organization's internal security.

Figure 9.1

The FireWall-1 gateway.



The FireWall-1 is composed of two major components, as follows:

- Packet-filter modules
- Control modules

A single control module can be used to control and monitor multiple packet filter modules. The packet filter module operates independently of the control module and can be placed on additional Internet gateways and servers to provide compartmentalized zones of risk. The control module is placed in the control workstation. The control module and packet filter module can reside on the same or different hosts. If they are placed on different hosts, communication between the control and packet filter modules are authenticated with a one-time password authentication.

The packet filter module implements the secure router functions between networks and is situated between the data link and network layers of the OSI model. Figures 9.2 and 9.3 show the packet filter operations implemented for incoming and outgoing packets, respectively. The data link layer is implemented by the network board, and the network layer is the IP protocol layer of the TCP/IP stack in the gateway.

For both incoming and outgoing packets, the packet headers are checked to see if they match a packet filtering rule. If no match takes place, the next rule is tried. For matched packets, a log/alert of the packet can optionally be done. After the log, a decision is made to pass the packet or drop it as specified in the packet filter rule.

If the packet cannot be matched, it is dropped in accordance with the security policy *That which is not expressly permitted is prohibited.*

A unique feature of FireWall-1's packet filtering is that it provides an effective way for filtering UDP and RPC traffic. This packet filtering is done by building context information for packets that are not part of a virtual circuit.

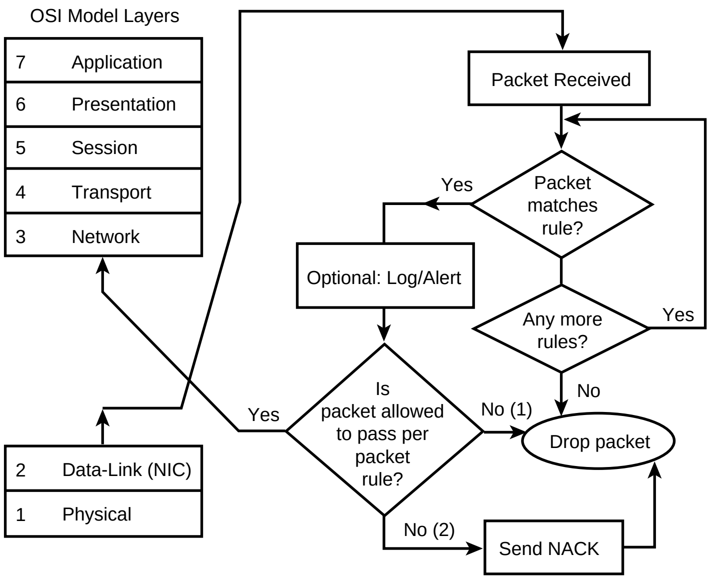


FIGURE 9.2

A FireWall-1 packet filter module operation on incoming packets.

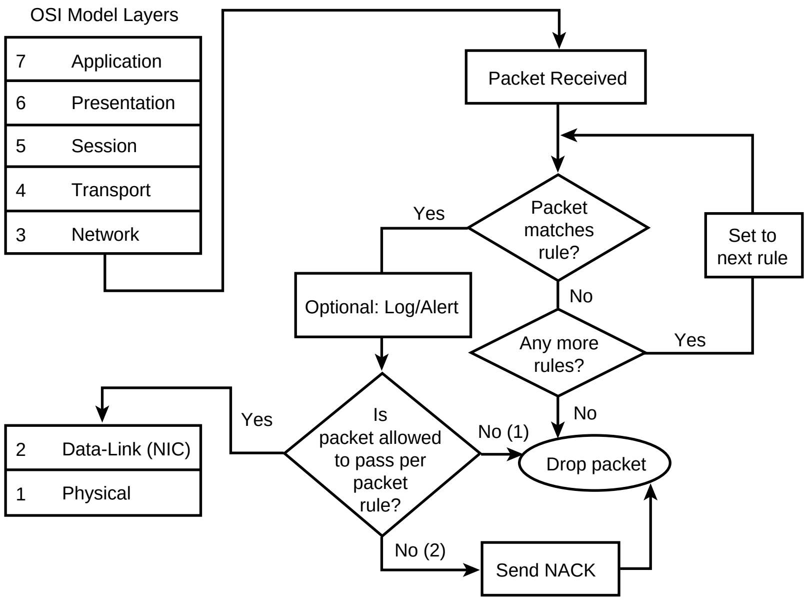


FIGURE 9.3

A FireWall-1 packet filter module operation on outgoing packets.

Other capabilities of FireWall-1's packet filtering follow:

- **Lists and objects.** Networks and services are represented by objects, which can be grouped together and referenced in access lists, allowing filter rules to be specified at a higher level than most packet filters.
- **Full data access.** The packet filtering rules can filter on addresses and header information for any of the layers 2 to 7 of the OSI model. Filtering on upper layer protocol information allows the specification of more intelligent rules based on more complete information. This allows the implementation of security policies based on application knowledge.
- **Protocol Independence.** High-level definitions can be used to define new protocols and applications, making the packet filter module more generic and flexible.
- **Auditing and Alerting.** Packets that are matched can be audited in a log or used to generate alerts. The log and alert formats and actions are user-configurable. The standard formats contain source and destination addresses, the protocol used, the service attempted, time and date, and the action carried out. Alerts can be programmed to run user-defined scripts to perform actions, such as triggering alarms and pager alerts, opening windows, and sending e-mail. Alerts can also be sent using SNMP to a SNMP Manager. The status report can be viewed using SNMP or through the Status System Monitor software.
- **Status reporting.** A summary of status information on network traffic is available in the System Status Monitor. This information is also available through an SNMP manager.
- **Kernel Module.** The packet filter module is a dynamically loadable kernel module, meaning that upgrades are simple to implement. Because the packet filter operated inside the kernel, it is not hampered by context switching overheads, which would exist if the module existed as a user process. In general, running inside the kernel has the advantage of negligible processing overheads, and is more efficient.
- **Optimization.** Packet filter optimizers are used to reduce the time spent processing packet filter rules. Techniques such as caching and hashing tables are used to unify multiple instances of objects and to access the data efficiently.

FIREWALL-1 CONTROL MODULE

The Control Module is used to implement the network security policy and the control packet filter modules (also called communication gateways). The Control Module can also be used as a central facility to view logging and control information.

The control workstation uses the OpenLook X11R5 GUI. Alternatively, a set of command line utilities can be used that permit a simpler computer terminal to perform management functions. The command line tools also enable the writing of script files to perform specialized control functions.

The overall architecture of the control module is shown in figure 9.4.

You can use FireWall-1 to define a global security policy for the network. Figure 9.5 shows the user interface for setting the security policy parameters.

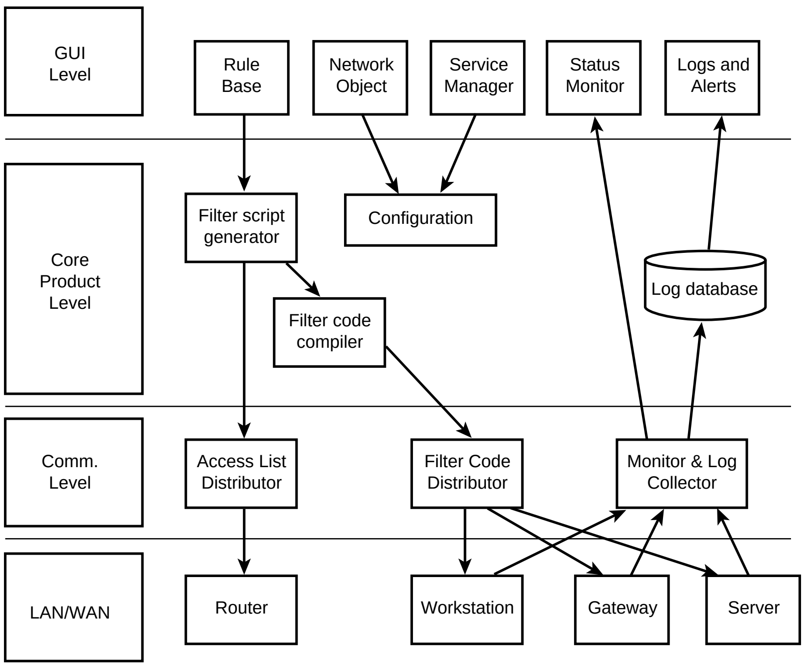
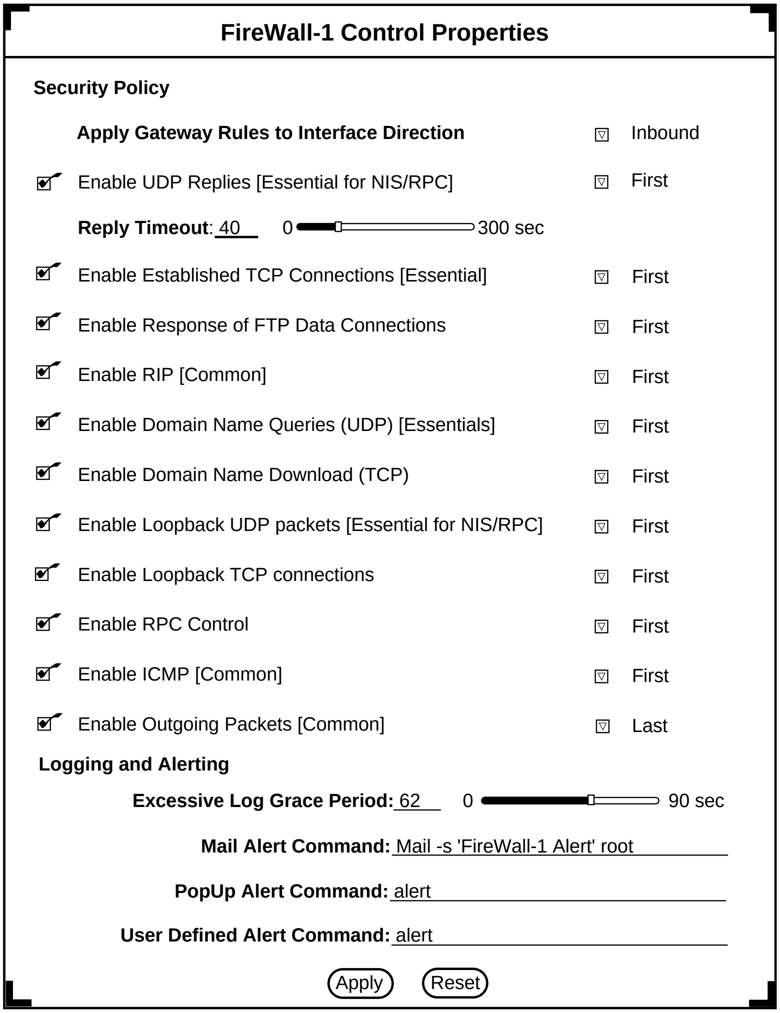


FIGURE 9.4
The FireWall-1 Control Module.

Figure 9.5

The FireWall-1 control properties.



Network Objects Manager

The Network Objects Manager is used to define objects that are specified in the security policy. These include the following object types:

- Networks and sub-networks
- Servers and workstations
- FireWall-1 hosts and gateways

- Routers
- Internet daemon

Figure 9.6 shows the Network Object Manager screen that can be used to define objects of the type discussed in this section. You can view the networks by types (internal, external) and type of objects (host, network, router, gateway, domain, group). All the object types in this figure are selected.

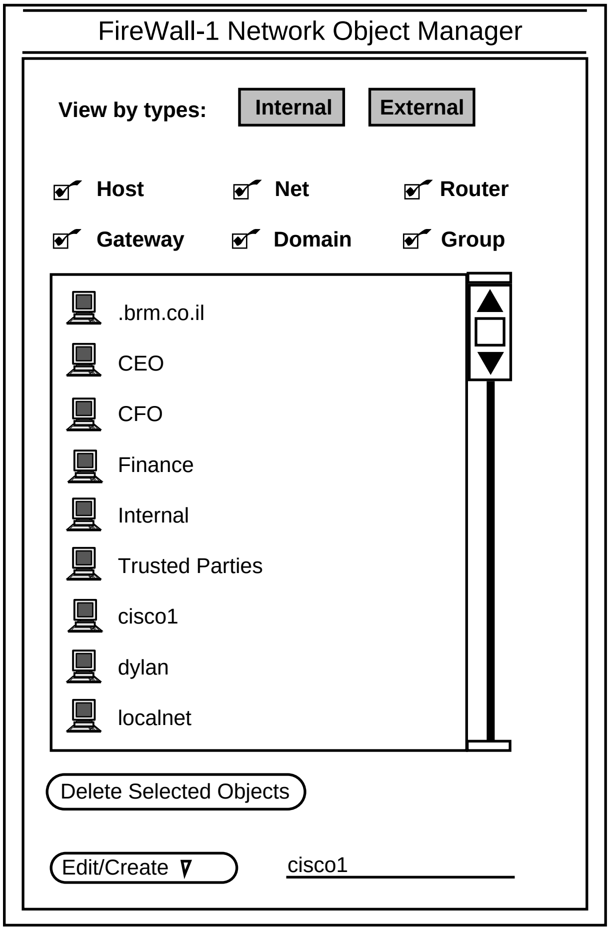


Figure 9.6

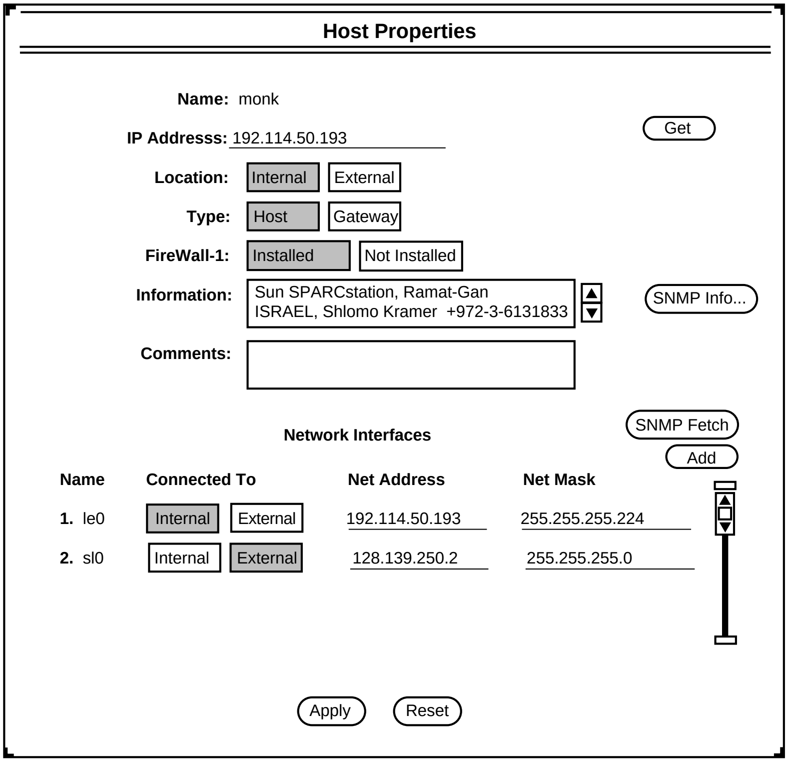
The FireWall-1 Network Object Manager screen.

Every network object has attributes, such as IP addressees and subnet masks, that define the characteristics of that object. These object attribute values are initialized from information stored in network databases. These network databases include files such as the /etc/hosts, /etc/networks, /etc/services, /etc/gateways. SUN-base networks may have the *Network Information Services* (NIS) databases. The information about various network objects can be extracted by polling SNMP (Simple Network Management) agents running on these devices. Objects can be combined in groups to create higher level object abstractions and hierarchies.

Figure 9.7 shows the attributes of a Host Object. This figure shows that some of the host properties are its IP address, location (internal or external), whether it is a host or gateway, whether FireWall-1 is installed on the host, and information on its network interface. The SNMP MIB (Management Information Base) variables for this host can be set, and you can define parameters such as the SNMP community name used to access the MIB variables in this host.

FIGURE 9.7

The FireWall-1 host properties.



Services Manager

The Services Manager defines the services that are known to the system and that are specified in the network security policy. All network services are screened and controlled. FireWall-1 comes pre-loaded with definitions for over 40 TCP/IP and Internet services that include the following:

- Standard services: Telnet, FTP, SMTP, and so on
- Berkeley r* services: rlogin, rsh, rcp, rexec, rwho, ruptime, and so on
- SunRPC services: NIS, NFS
- Internet search tools: HTTP (Hyper Text Transfer Protocol), Gopher, Archie, WAIS (Wide Area Information Services)
- IP services: ICMP, RIP
- Management services: SNMP

The Services Manager can be used to define a new service by selecting the service type. The service types include the following choices:

- TCP
- UDP
- RPC
- Others: This enables definition of other services and protocols that are not standard.

Services can be grouped into families and hierarchies, such as the NFS group that includes NFS-server, Lock Manager, and Mount Program. Another example is Mosaic (WWW) that includes HTTP, Archie, Gopher, and others.

Figure 9.8 shows the Services Manager. You can use this screen to define new services or edit existing ones. Figure 9.9 shows the service properties of a standard service such as Telnet.

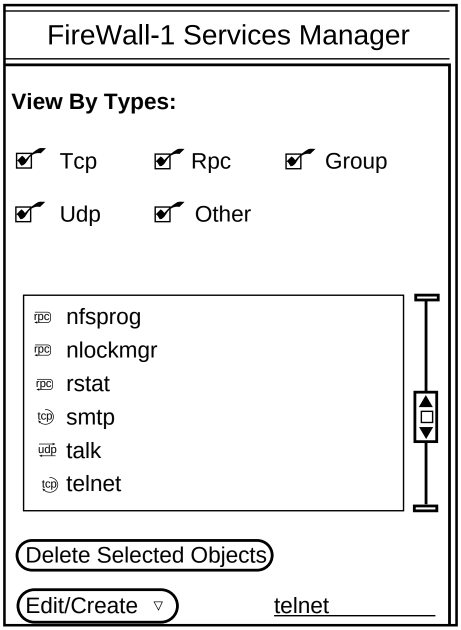


FIGURE 9.8

The FireWall-1 Services Manager.

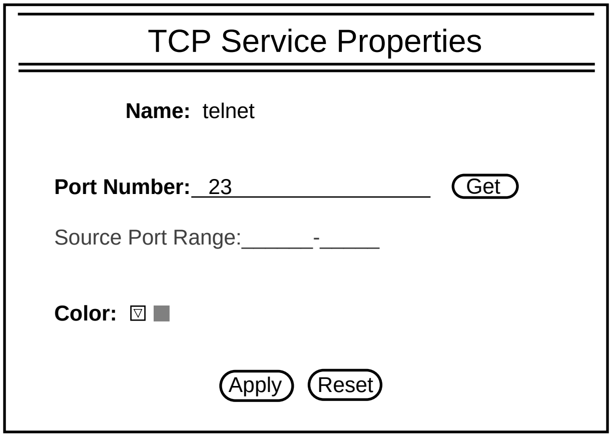


FIGURE 9.9

The FireWall-1 TCP Services Properties.

RULES-BASE MANAGER

The Rules-Base Manager is a software component that is used for managing the packet filter rules for FireWall-1. After you define the network objects and services, you can define the rules that make these services for implementing the security policy. When the rules are compiled and installed, they act as a packet filter. The Rules-Base editor is used for making changes to the packet filter rules. Figure 9.10 shows the Rules-Base editor for FireWall-1. This figure shows that rules can be entered in a table that has columns for source, destination, services, action, track, and "install on."

In figure 9.10, Rule 1 accepts an SMTP service packet from any source to a group of hosts defined by the group "mailservers." The action is to accept these SMTP services on the installed gateways.

Rule 2 allows any host on the network to access any service on any network. This rule is installed on the gateway. Additionally, a short log entry is created for matching packets.

Rule 3 blocks any attempt by any external source to the local network. This rule is implemented on the gateway. Additionally, alert messages are sent with the occurrence of such attempts.

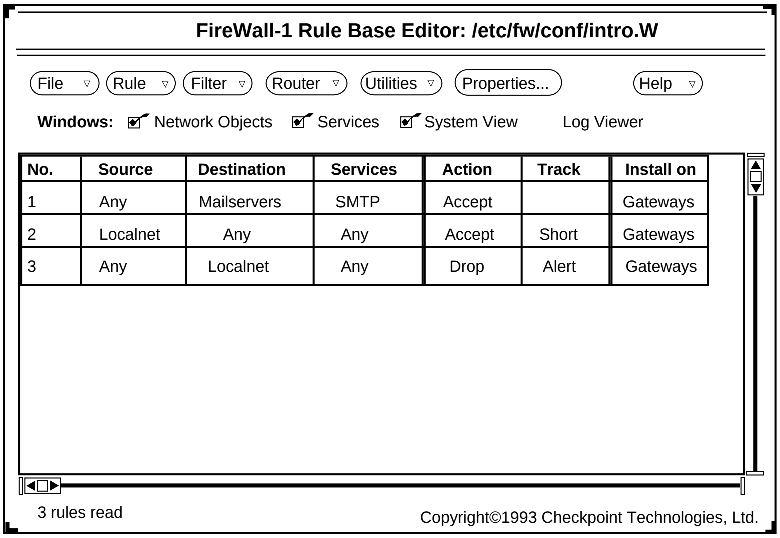


Figure 9.10

The FireWall-1 Rule Base Editor.

These rules are a high-level definition that consists of four parts: match, action, track, target. These parts are explained as follows:

- ✣ **Match.** Specifies the communication attempt in the rule. It includes the source and destination of the services involved in the communication.
- ✣ **Action.** Specifies how the communication attempt should be handled. Communication attempts can be accepted or rejected, and a negative-acknowledgment generated or simply dropped.

- **Track.** Specifies the type of auditing or alerting that should be performed, including the following:

  No logging

  Short format record

  Long format record

  Send e-mail message

  Generate a SNMP trap

  Activate a user defined procedure

  Generate Alert message, such as a pop-up window on the system manager's workstation

- **Target.** Used to specify which packet filter module on the network should implement the filter rule. This mechanism can be used to implement enterprise-wide security rules. Rules can be enforced on all gateways, destination servers and gateways, and on source gateways and hosts.

Every communication attempt is matched against the Rules-Base. Rules are matched in the order they are specified, enabling predictable behavior of the packet filtering system. If no rules are matched, the packet is rejected, thus implementing the *That which is not expressly permitted is prohibited* security policy.

Rules are verified for correctness. This verification mechanism includes:

- Heuristic tests
- Definition inconsistencies
- Redundancy and order checking

The complete Rules-Base is used to generate a filter script. The *filter script* describes the objects and rules using the CheckPoint's filter definition language. These scripts can be used to implement sophisticated security policies.

You can view the filter scripts that are generated. Figure 9.11 shows an example of viewing a filter script generated from a Rules-Base.

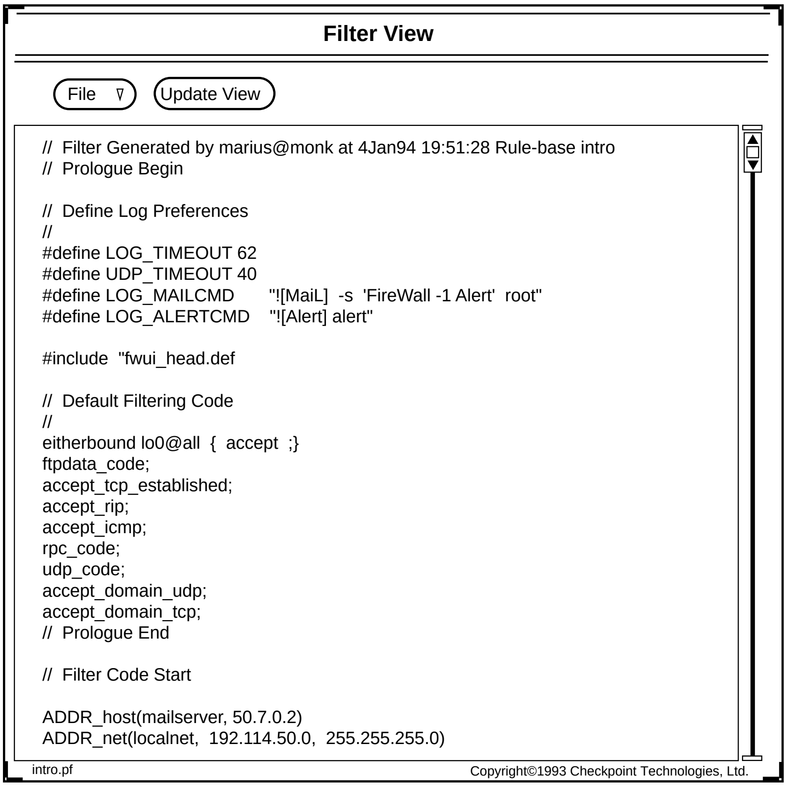


FIGURE 9.11

The FireWall-1 Filter View.

ROUTER ACCESS LIST

The packet filter rules that are defined for routers are sometimes called router access lists by router vendors. The FireWall-1 supports both Cisco and Welfleet router access lists.

If Cisco routers with release level 9.1 (or higher) and Welfleet routers with release 8.0 (or higher) are used, they can be programmed with access lists that are generated from the Rules-Base. The rules for the routers are implemented using the router object definitions. The rules are verified and sent to the Access List Distributor module, which distributes it to the different routers.

Figure 9.12 shows an access list created from the filter script for a Cisco router. You might recognize these commands from the discussion of the access list commands for Cisco router in Chapter 6, "Packet Filters."

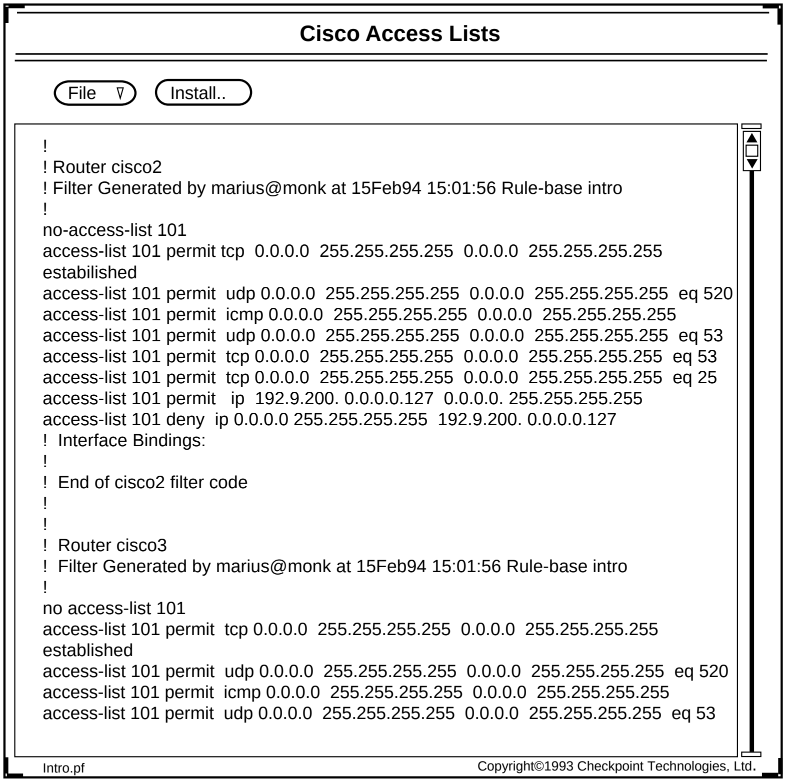


FIGURE 9.12
The FireWall-1 access list for Cisco routers.

Most routers do not implement many of the capabilities required for secure packet filtering because they look at TCP/UDP/IP headers only. A protocol such as UDP is stateless, and the routers do not keep track of the context of previous packets.

STATUS MONITOR

The Status Monitor is a software component of FireWall-1 that is used to monitor the status of the FireWall-1 filter module. A system status window can be activated to display a snapshot of all the FireWall-1 filter modules at any time interval (see fig. 9.13). The status includes packet statistics such as the number of packets dropped, passed, logged, and so forth. The status of the filter modules is displayed as well.

The packet filtering modules support an SNMP agent that can be used for exporting information to other SNMP managers.

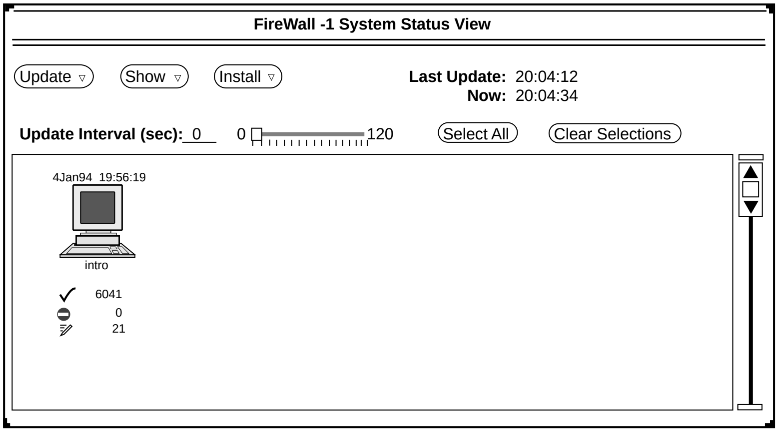


FIGURE 9.13

The FireWall-1 System Status View.

LOG VIEWER

The *Log Viewer* can display the events that have been logged, such as communication attempts, system shutdowns, firewall installations, and so on (see fig. 9.14). The event information includes date and time of event, originating machine, source and destination of communication, services attempted, action taken, and log alert types.

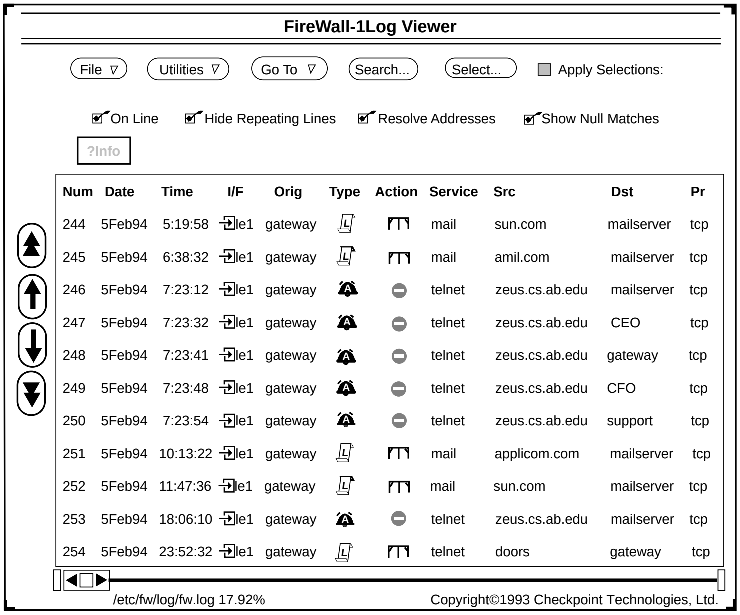


FIGURE 9.14

The FireWall-1 Log Viewer.

The Log Viewer can hide or display any of the information that has been logged (see fig. 9.15). A search facility enables you to locate rapidly any event of interest to you (see fig. 9.16). Reports are generated by applying selected criteria to the log fields. Reports can be viewed and exported to ASCII or PostScript. You can monitor communication activities and alerts in real-time. Nodes that appear in the log and that have SNMP agents can be probed from within the Log Viewer for SNMP information.

FIGURE 9.15

The FireWall-1 Log Viewer Selection Manager.

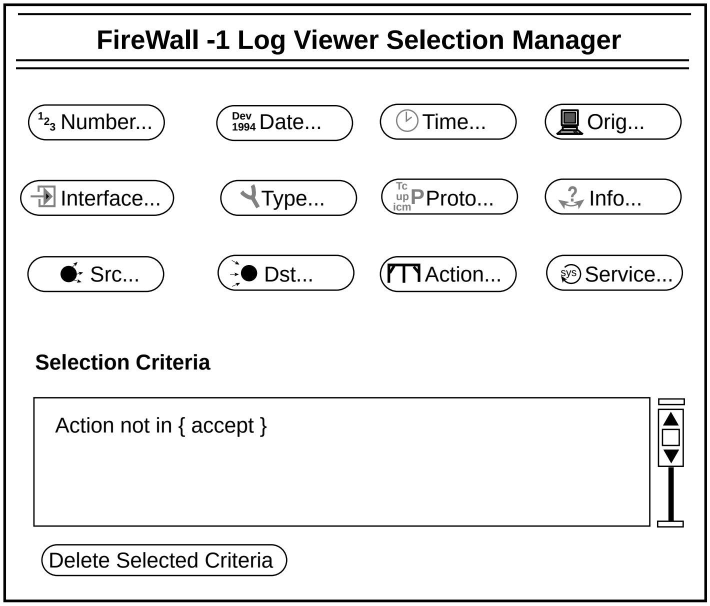
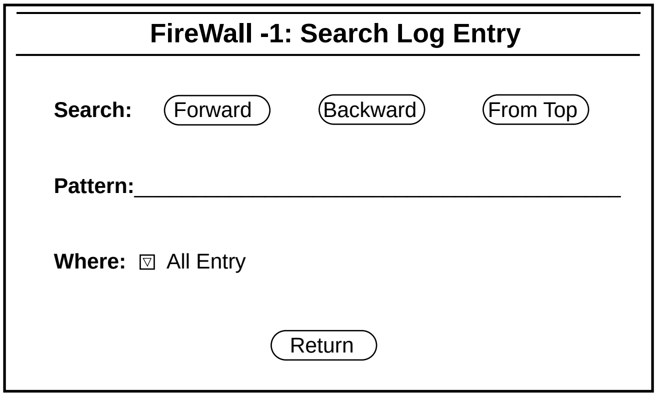


FIGURE 9.16

The FireWall-1 Search Log Entry.



EXAMPLES OF FIREWALL-1 APPLICATIONS

This section explains how FireWall-1 can be used to handle Internet Protocols and applications by performing secure packet filtering.

UDP APPLICATIONS

The problem of filtering UDP applications was discussed in detail in Chapter 6. Briefly, the problem with UDP, as it relates to packet filtering, is that it is a connection-less protocol. Unlike connections-oriented protocols such as TCP, there is no inherent distinction between the originator and request of the response. Because of this UDP source, port numbers can be easily "spoofed." UDP-based applications such as TFTP, DNS, and WAIS are difficult to filter in a secure manner. Some packet filtering solutions avoid this problem by not doing anything about it, thereby exposing the internal hosts to attacks through UDP ports, or by completely eliminating access to UDP ports.

FireWall-1 packet filtering solves the UDP packet filtering problem by maintaining context information on top of the UDP connection. You might think of this additional context information as representing an internal virtual-circuit in the firewall on top of the UDP connection. Every UDP request packet that is permitted to cross the firewall is recorded. Each incoming UDP packet is checked against a list of pending connections. Only when the packet is a response to a pending request is it delivered. Forged UDP packets pretend to be a response to a previous request. Because no such request is pending, the forged UDP packet is rejected.

OUTBOUND FTP CONNECTIONS

Although the FTP is a common application on the Internet, it still provides a level of complexity that can be difficult to handle. Briefly, the problem is that after the FTP client initiates a connection, the FTP server establishes a new back-connection (also called *call-back connection*) to the FTP client. The connection request originates from the FTP server, which may be outside the firewall boundary, to a dynamically allocated port on the FTP client. The dynamically allocated port is not known in advance. Many FTP clients specify new port numbers for each data transfer connection. Some older FTP clients open up the entire range of high-numbered ports, greater than 1,023, to back-connection requests. Unless a protection mechanism is implemented, the high-numbered ports are exposed to intruders.

> **NOTE**
>
> The call-back connection initiated by the FTP server exposes the entire range of high-numbered ports greater that 1,023 to "spoofed" back-connection requests.

FireWall-1 handles FTP data connections by examining application level data. When the client generates a request for a back-connection (through the FTP PORT command), the firewall records this request. When the back-connection is attempted, it is checked against the pending FTP PORT command. Only the back-connection that specifies the port number in the pending FTP port command is allowed. All other attempts to connect from the FTP_DATA port number 20 are rejected. The list of pending connections is maintained dynamically, so only the required FTP ports are opened during an FTP session.

RPC-Based Problems

RPC-based services, such as NFS, use dynamic port number assignments. The client program contacts, with the name of a service, the portmapper program that listens on UDP port 111. Other services register with the portmapper so that the portmapper has a record of all the services registered with it. The portmapper, on being contacted for an RPC-based service by a client, maps the service name to the registered port number and sends this information back to the client. Because of dynamic port assignments, you do not know which port numbers are in use, making it difficult to protect the port numbers.

FireWall-1 keeps track of RPC port numbers, and extracts application specific information from the packet. This information is used to identify the program and source using the service. It also keeps track of the UDP packets and builds context information for these circuits. By using a combination of these approaches, FireWall-1 protects the RPC-based services from attack.

Mosaic, WWW, and Gopher

Clients, such as Gopher clients and Mosaic front-ends to the World Wide Web (WWW), pose a new set of problems for securing networks. Mosaic front-ends, in particular, can access a number of application services, such as HTTP, FTP, WAIS, Archie, and Gopher, through a unified front-end. These expose the network to the security risks of the underlying protocols, such as UDP, TCP, and FTP.

Because FireWall-1 can handle the underlying protocols, such as TCP, UDP, and FTP, it can be used for securing the network when used with Mosaic.

Performance of FireWall-1

Because FireWall-1 employs several optimization techniques, the performance degradation for most applications is negligible. These optimization techniques include the following:

- Running inside the operating system kernel reduces processing overhead and avoids context switching overheads.

- Packet filter optimization reduces the time spent performing filtering actions.
- Memory management techniques such as caching and hashing provide rapid access to network resource objects. These techniques can unify multiple instances of objects, which allows for more efficient sharing of memory.

The Internet Security Corporation reports that tests done on low-end Sparcstations showed negligible degradation over Ethernet at 10 Mbps. Performance degradation on lower speed links will be even less because WAN links operating at 56 Kbps or 1.544 Mbps (T1) are slower, and the bottleneck is the speed of the WAN link and not the packet filtering overhead.

FireWall-1 Rules Language

You can use a text editor to write filter rules directly, using theFireWall-1 rules language. You might want to do this to have greater control over the packet filtering specifications. The following examples are presented to give you an overview of the FireWall-1 rules language.

The file std.def must be included in each rules set. This file contains definitions for standard macros, aliases, log formats and TCP packet structure.

Example 1

The following rule checks inbound packets at all interfaces on ucs.xyz.com and allows any host to send smtp mail to ucs.xyz.com.

```
#include "std.def"
inbound all@ucs.xyz.com accept dst in ucs.xyz.com, smtp;
```

Example 2

The following rule checks inbound packets at all interfaces on ucs.xyz.com and allows all hosts on the local net to do ftp or telnet sessions to ucs.xyz.com.

```
#include "std.def"
inbound all@ucs.xyz.com accept src in local_net, dst in ucs.xyz.com, (ftp or
telnet);
```

Example 3

The following rule allows all hosts to access any other local host for services that are defined in the list `locally_allowed`. The list `locally_allowed` is user defined.

```
#include "std.def"
eitherbound all@all accept src in local_net, dst in local_net, service in
locally_allowed;
```

Example 4

The following rule checks packets coming in through the sl0 serial interface of gateway GW1 and enables clients in TRUST.COM to send smtp mail to the local network.

```
#include "std.def"
inbound sl0@GW1 accept src in .trust.com, dst in local_net, smtp;
```

Example 5

This rule checks outgoing packets on serial interface sl0 of the gateway GW1 and allows the local network to access services on hosts in the university BERKELEY.EDU.

```
#include "std.def"
outbound sl0@GW1 accept src in local_net, dst in .berkeley.edu;
```

Example 6

The following rule checks incoming and outgoing packets on serial interface sl0 of the gateway GW1 and rejects all packets coming from `.hacker.org` to the local network. All such attempts that match the rule are logged and reported immediately to the systems manager.

```
#include "std.def"
eitherbound sl0@GW1 reject log <"! alert"> short src in .hacker.org and dst
in local_net
```

Example 7

The first rule that follows logs the first packet of every attempt to establish a telnet connection through serial interface sl0 of gateway GW1, whether it succeeds or not. Clients must be in the user-defined list `trusted_hosts` to make the connection.

The second rule allows packets to travel in either direction once a TCP connection is established.

```
#include "std.def"
inbound sl0@GW1 accept src in trusted_hosts,  dst in local_net, telnet,
first. log short;
eitherbound sl0@GW1 accept tcp, established;
```

Example 8

This rule checks incoming packets on serial interface sl0 of the gateway GW1 and blocks all packets coming from `.hack.org` to the hosts defined in the list `protect_list`. All such attempts that match the rule are logged and reported immediately to the systems manager though e-mail.

```
#include "std.def"
inbound sl0@GW1 drop src in .hack.org dst in protect_list log <"!mail -s FW-1
alert root"> short;
```

Obtaining Information on FireWall-1

Additional information on FireWall-1 can be obtained from the following:

Internet Security Corporation
Phone: (617)863-6400
Fax: (617)863-6464
Email: info@security.com

or

CheckPoint Software Technologies
Phone: (617)859-9051
Fax: (617)863-0524

ANS InterLock

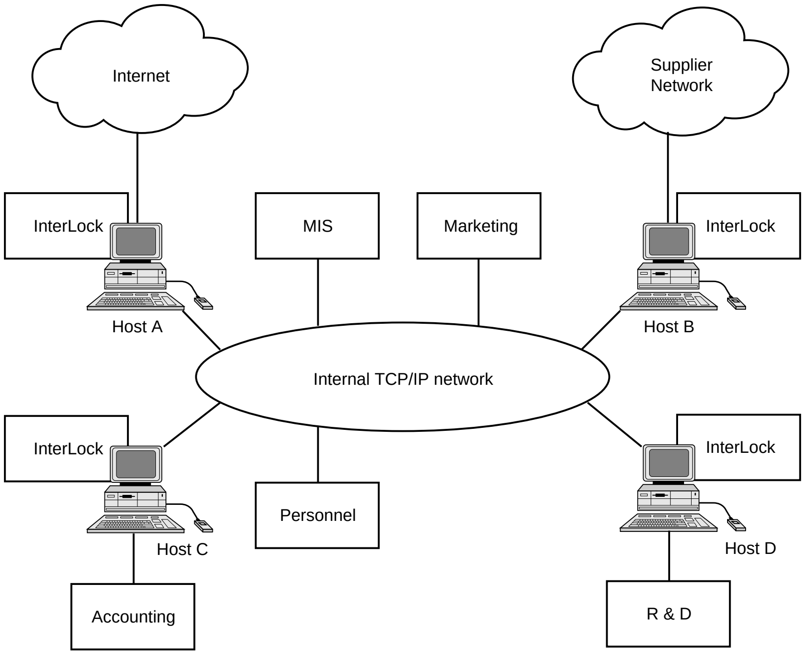
The FireWall-1 is a commercial gateway product, available from Advanced Network Services (ANS). The product currently runs on IBM AIX 3.2 (or higher) workstations or SparcStations (Solaris 2.3 or higher).

The ANS InterLock provides an application-level firewall gateway between an organization's internal network and an untrusted external network. The ANS InterLock can also be used to control access between segments of an internal TCP/IP network. This situation is shown in figure 9.17 where the InterLock service is deployed at the boundaries of the internal network. In this figure, the InterLock services running on host A is used to secure access to the Internet. The InterLock service running on host B limits access from a supplier (vendor) network. The InterLock service running on host C limits access from users of the Accounting department network, and the InterLock service running on host D limits access from users of the R&D department network.

The ANS InterLock gateway provides the following capabilities:

- Access control mechanisms
- Auditing and logging information
- Information hiding
- Mail Gateway services

Figure 9.17
Multiple ANS InterLock connections.



The Access control mechanism provides the ability to control access between any two networks based on a flexible criteria. Access control can be specified in terms of type of service, source, destination, and time of day the services are allowed. For example, one can specify that FTP access from external sites will be allowed from 9 a.m. to 5 p.m., Monday through Friday. You can also impose a requirement that users must authenticate themselves using a SecurID card (made by Security Dynamics Technologies, Inc.). This type of authentication can be used instead of the Unix user login, or to augment the Unix login.

The InterLock can be used to generate detailed logs and reports on authorized and unauthorized network connections. The logs accumulate usage statistics on a per user, per IP address, or per service basis. The data includes information on the duration of connection, bytes transferred, file names and sizes of files. Run-time data reduction tools are used to specify the level of detail and the amount of information logged for each service. Post run-time reporting tools can be used to generate usage statistics.

The InterLock does not reveal information about the internal network to the external network. The InterLock runs on a host with routing functions and IP forwarding disabled. As a result of this, information, such as host names, IP addresses, and network structure, is hidden from the external network. Although not recommended, it is possible to use non-NIC-assigned IP addresses to the internal network, because these addresses are hidden from the outside world.

Resource Requirements for InterLock

Before investing resources in evaluating a product to see if it is suitable for your network environment, it is helpful to know details such as the hardware and software and training resources needed to implement the product. This information is provided for your reference.

The ANS InterLock runs on a dedicated platform based on R6000 IBM workstations or SUN SparcStations. The R6000 workstations run a modified version of the AIX operating system, and the SUN Sparcstations run a modified version Solaris. The modified workstations are pre-configured and can be purchased as a leased "turn-key." Alternatively, a leased software version (running on customer-provided hardware) and a software-only version can be purchased with an annual support contract.

The operating system modifications are done to prevent security holes that can exist because of improper configuration. Instead of turning off operating systems options that can be hazardous to security, ANS has completely removed these capabilities from the operating system to eliminate the possibility of an intruder gaining access to the system and enabling features that would bypass the firewall security.

The following operating system capabilities are removed from the operating system:

- No IP forwarding
- ICMP redirects are rejected
- No strict or loose source routing

With IP forwarding disabled, all connection requests are handled by application proxy daemons.

ICMP redirects can be used to create false entries in router tables, which can lead to denial of service or to network traffic being diverted to an unsecured host.

Source routing is removed to prevent packets from bypassing the firewall. On many systems, source routed packets are forwarded even if IP forwarding is disabled. By removing source routing and IP forwarding, packets can never be routed through the network layer.

Overview of InterLock

Typical tasks that need to be performed to set up the InterLock firewall are the following:

- Adding, deleting, and modifying user and groups
- Defining a security policy in the Access Control Rule Base
- Monitoring logs and generating reporting statistics

- Customizing prompts and system messages
- Executing the system backup scripts
- Editing mail mapping databases
- Establishing and managing password controls

All of the previously listed tasks can be performed by a single administrator or by a number of administrators. To ease system administration tasks, InterLock offers a flexible way of assigning different privileges to administrator accounts. A user account on the InterLock gateway can be assigned a combination of the following privileges:

- **Mail.** Allows configuration and control over the mail system.
- **Security.** Maintenance of security policies implemented by the Rules-Base.
- **Admin.** Creation and maintenance of user accounts.
- **Audit.** Monitoring and data reduction of log information.
- **System.** Miscellaneous privileged operations.

Separation of the security privileges enables the organization to assign specific individuals to the different security functions. A specific privilege is assigned by making a user account a member of a group. This provides the organization with a flexible and familiar way of assigning privileges.

The access controls that govern the behavior of each application gateway are implemented in the *Access Control Rule Base* (ACRB). Rules to the ACRB can be specified using a menu-driven interface or a command interface. The organization's security policy is expressed as a set of rules. Explicit rules need to be provided to enable a service. Rules can be established for individual users, groups of users, or to an entire community of users using a wildcard entry.

TIP

> Because users have similar security needs, it is easier to define groups and establish rules for the groups. The individual users, then, inherit the rules because of membership in groups.

InterLock requires that all connections should be handled by application proxy daemons running on the InterLock gateway machine. Each application-layer daemon is responsible for reading its system configuration information and rules from the ACRB. If a proxy daemon is not provided, that service will be denied. InterLock provides proxy daemons for the following services:

- TELNET
- Login service

- FTP
- SMTP
- X Windows systems
- NNTP
- InterLock encryption service
- InterLock Generic Protocol Daemon (GPD)
- Network Time Protocol

Configuring InterLock

Before the InterLock gateway can be used, it must be configured with some site-specific information, which is defined in the file with the following path name:

/interlock/config/ILconfig

A sample ILconfig file follows:

```
#
# Interface names
#
     Public_Interface               en0
#    Private_Address                145.122.12.22
#    Public_Address                 193.34.33.53
#Counters
#
    Max_TELNET_login failures       3
    Max_FTP_login failures          3
#
#Timers
#
    StartX_timeout_minutes                3
    X_Inactive_timeout_minutes            10
    Telnet_Inactivity_Timeout_minutes     20
    Telnet_Inactivity_Warning_minutes     5
#
#NNTP Configuration options
#
    NNTP_Public                    server.pubcorp.net
    NNTP_Private                   news.xyz.com
    NNTP_DomainName                disguise.com
    NNTP Post_Header_Mapping       On
    NNTP_Use_Mailmaps              Off
#
#Authentication options
#
    Authentication_List            unixpassword
```

The `Public_Interface` field is used to define which network interface is attached to the public network. The public network is the untrusted external network.

The `Private_Address` and `Public_Address` are commented out and are not configurable by the InterLock administrator.

The `Max_TELNET_Login_Failures` is the number of times a user will be allowed to attempt an unsuccessful TELNET login attempt before InterLock will close the connection.

The `Max_FTP_Login_Failures` is the number of times a user will be allowed to attempt an unsuccessful FTP login attempt before InterLock will close the connection.

There is currently no provision to lock-out an intruder for a period of time. Also, if specific TELNET and FTP connections are to be allowed or blocked, these must be specified by the Access Control Rule Base (ACRB).

The `StartX_timeout_minutes` defines the number of minutes the InterLock proxy X-daemon will listen for an X connection on the TCP port associated with the pseudo-terminal display number. If a connection does not occur in the specified time period, the InterLock X-daemon will close the connection and not accept any connections to the display number. When the value is set to zero, timeouts are disabled.

The `X_Inactive_timeout_minutes` defines the number of minutes that a X session can be inactive before the connection is closed. When the value is set to zero, timeouts are disabled.

The `Telnet_Inactivity_Timeout_minutes` is the number of minutes that a Telnet session can be inactive before the connection is closed. When the value is set to zero, timeouts are disabled.

The `Telnet_Inactivity_Warning_minutes` is the number of minutes that a Telnet session can be inactive before a warning message is issued. When the value is set to zero, warnings are not sent.

The `NNTP_Public` is used to define the host on the public (external) network that acts as a news server. If you have an internal news server that exchanges messages with the public news server, you must define this field.

The `NNTP_Private` is used to define the host on the private (internal) network that acts as a news server. This is the destination to which NNTP news originating from the external network will be directed.

The `NNTP Post_Header_Mapping` can be set to On or Off. When set to On, the NNTP proxy daemon will attempt to map the header portion of postings from internal users with the domain name specified in the `NNTP_DomainName` field. Remapping involves replacing the lines in the news header that contain references to internal hosts.

The `NNTP Post_Use_Mailmaps` can be set to On or Off. This field is defined only when the `NNTP Post_Header_Mapping` is On. When set to On, the NNTP proxy daemon will use the mail mapping database (/interlock/config/maild.mf derived dbm file) when changing the user part of the post headers. If Off, the user will query the database built from the file derived from /interlock/config/nntpd.mf.

`Authentication_List` is the list of available authentication types that are supported in a specific InterLock gateway. The default authentication type is Unix passwords (unixpasswd). You can use values of "securid" or "pinpad" to specify support for Security Dynamics' SecurID and Pinpad cards (optional services).

TIP

> The ILconfig file can be edited to change any of the parameter values just discussed. After editing this file, you can run the makeconfig command, which reads the entries in the configuration file and stores them in the binary database /interlock/config/config.bin.

The InterLock ACRB

The rules that govern the actions to be performed on packets, by an InterLock proxy application, are specified in the ACRB.

The first portion describes the situations in which the rule is to be enforced. Rules that do not match a particular situation (such as being outside the valid time range for which a service can be accessed) can be configured by the administrator to deny access or remain inactive.

The second part of the rule specifies what rule constraints should be enforced. You can define different levels of logging with each rule. For example, the logging levels could be low, medium, high, or debug, for each rule.

The rules can specify combinations of any of the following:

- **User name or group.** This is the site-defined user name or group name. A wildcard (*) can represent any user. A special user name called "trust" can be used when the internal user's identity is not required from a specific host or network. This enables users on the internal network to use authenticated services TELNET and FTP without having to enter their user name and password.
- **Protocol name.** This is the name of a service, such as LOGIN, TELNET, FTP, FTP-DATA SMTP, NNTP, X, or a customer-defined service. The port number can be used instead of the service name.

- **PRIVATE NETWORK OR HOST NAME.** This is the network or host on the inside (protected) network. The host's IP address or the network IP address (example: 144.19.0.0) can be used. A wildcard (*) matches all hosts on the internal network.
- **PRIVATE NETMASK.** This is the range of host IP addresses on the inside (protected) network. This mask value is not the subnet mask value.
- **PUBLIC NETWORK OR HOST NAME.** This is the network or host on the external (untrusted) network. The host's IP address or the network IP address (example: 144.19.0.0) can be used. A wildcard (*) matches all hosts on the external network.
- **PUBLIC NETMASK.** This is the range of host IP addresses on the external (untrusted) network. This mask value is not the subnet mask value.
- **START TIME.** This specifies the time when the rule will take effect, in a 24-hour format.
- **STOP TIME.** This specifies the time when the rule stops being in effect, in a 24-hour format. Existing connections are not shut down at the stop time; however, new connections are not allowed.
- **DAY-OF-THE-WEEK.** This specifies the day or range of days that the rule is in effect.
- **DIRECTION.** This defines in which directions the connections are allowed to flow.
- **ENCRYPTION.** This is defined only if encryption is provided as part of the package.
- **AUTHENTICATION TYPE.** This specifies the type of authentication mechanisms that are used. Valid authentication types are described in the ILconfig file described in the previous section.

The Username or group field and the Protocol name field cannot both be wildcards (*) in the same rule. The security policy of *That which is not expressly permitted is prohibited* is used. Therefore, at a minimum, there will be *N* rules if *N* applications are to be supported across the InterLock gateway.

To define the rules, a program called *rulebase* (located in the /interlock/security directory) is run. The user interface is simple to use, as shown in the following list. The subsequent list shows the choices (in bold) for adding a new rule. This rule defines that user ksiyan is allowed to telnet from any host on the private network to the host maclean.spy.com on the public (external) network. It also allows ksiyan to telnet from host MACLEAN.SPY.COM to any host on the private network. The user ksiyan is required to use the Unix password authentication. Telnet access is allowed only from 9 a.m. to 5 p.m., from Monday through Friday. Telnet access for this user is denied at all other times.

The following is the InterLock ACRB for Version 1.2:

```
<A>dd a new rule
<D>elete an existing rule
<M>odify an existing rule
<T>oggle remote login access for root
<S>how which rules apply
<C>heck user permissions
<W>rite text verson of rules base to a file
<Q>uit
```

This list shows the Add New Access Control Rule settings:

```
Group or username: ksiyan
Protcol name: telnet

Private network or host name: *

Public network or host name: maclean.spy.com
Is MACLEAN.SPY.COM a host name (y,n)? y

What direction is access permitted?
From MACLEAN.SPY.COM to ANY private host [y,n] ? y
From ANY private host to MACLEAN.SPY.COM [y,n] ? y

What type of user authentication is required on this connection?
Authentication type (unixpasswd,securid,pinpad,*): u

What type of encryption is required on this connection?
Encryption type (software,hardware,*,both,none): n

Should this rule be enforced all of the time (y,n) ? n
Days of week (su,m,tu,w,th,f,sa,*) [*]: m-f
Starting at time (00:00 - 23:59): 0900
Ending at time (00:00 - 23:59): 1700
Should KSIYANs TELNET access be DENIED all other times (y,n) ? y
```

InterLock Proxy Application Gateway Services

This section discusses some of the application gateway proxy services provided by the InterLock gateway. Examples are shown using the command line interface to these services.

InterLock Telnet Proxy Services

Figure 9.18 shows a typical deployment of the InterLock gateway. Users on either side of the InterLock gateway must first connect to the InterLock gateway and authenticate themselves.

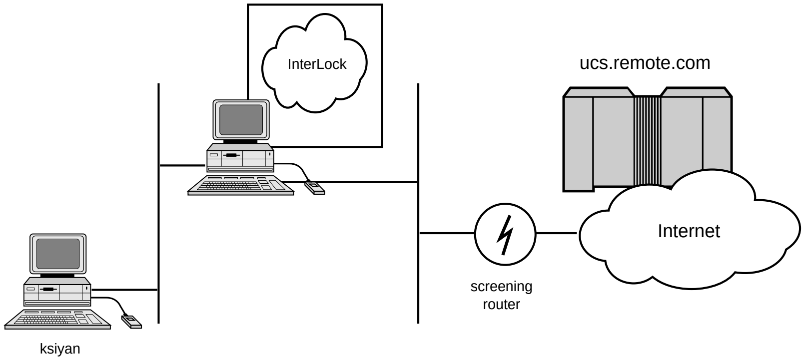


FIGURE 9.18

Typical InterLock gateway deployment.

The InterLock gateway runs a telnet daemon that listens to well-known ports associated with these services. If desired, the organization can change the port associated with these services.

In this example, assume that the user ksiyan wants to establish a Telnet session from an internal host to a remote host on the Internet. The Telnet session must go through the InterLock gateway. The user must first initiate a telnet session to the InterLock gateway. The Telnet daemon on the InterLock receives this request and queries its ACRB to see if a Telnet session is allowed at the current time and day of week. If a connection is allowed, the user is prompted for a user name and password. If the InterLock service was configured with a SecurID option, prompts similar to that follow appear:

```
telnet interlock.kinetics.com
Trying...
Connected to interlock.kinetics.com
Escape character is ^[
InterLock login: ksiyan
ksiyans PIN: PIN entered is not displayed
ksiyans CARDCODE: cardcode entered is not displayed
```

After a successful login, you can see the InterLock gateway message. You must enter the remote destination to connect to.

```
InterLock Telnet Gateway N.0 (interlock.kinetics.com)
Destination: ucs.remote.com
```

The ACRB is queried again to determine if Telnet access to the specified destination is permitted. If access is allowed, connection is established by the proxy Telnet daemon to the remote host, and the user sees the login prompt for the remote host. If the ACRB checks fail (rules do not match), Telnet access is denied.

InterLock FTP Proxy Services

FTP works in a manner similar to Telnet. First an FTP connection to the InterLock gateway is established. The ACRB is consulted to see if a connection is allowed. The InterLock FTP server prompts the user for login name and authentication information. The user then issues the USER command to specify the user and host name for the desired destination. The InterLock FTP consults the ACRB to see if a connection is allowed.

If a connection is allowed, an FTP connection is established by InterLock FTP to the remote. The user is prompted for a password on the destination host. The following list shows a sample FTP session.

```
% ftp interlock.kinetics.com
Connected to interlock.kinetics.com
220 interlock.kinetics.com InterLock FTP Gateway N.0 ready at Wed Nov 16
12:34:33 1994
Name (interlock.kinetics.com:ksiyan): ksiyan
331 Password required for ksiyan
Password: password is not displayed
230- You are authorized as InterLock user ksiyan. Specify your remote user
230  and destination with the USER command (for example: USER
remoteuser@remotehost)
ftp> user anonymous@ucs.remote.com
331- ucs.remote.com  FTP server (Version 5.60) ready
331  Guest login ok, send ident as password
Password: password is not displayed
```

InterLock SMTP Gateway Services

The InterLock can act as a store-and-forward SMTP gateway. There is no password authentication of a user's SMTP session. The InterLock mailer, by default, consults a mapping database to determine where to deliver incoming mail from public hosts. This database also determines if the address in the FROM: line in the message header is allowed to send mail. An example mapping entry from this database follows:

```
ksiyan: karanjit@rama.kinetics.com, karan@shiv.kinetics.com
```

The entry before the colon defines the mailbox known to the outside network for this user. This is the user's external e-mail address. Mail to this user can be sent from the outside world to the address `ksiyan@kinetics.com`. The entries after the colon are valid sending internal e-mail addresses for the user. Mail sent to user `ksiyan@kinetics.com` is mapped to `karanjit@rama.kinetics.com`. Mail which arrives from `karanjit@rama.kinetics.com` or `karan@shiv.kinetics.com` is mapped as if it originated from the user `ksiyan@kinetics.com`. Therefore, the internal e-mail addresses are not known to the outside world. This is important for security reasons because internal e-mail addresses indicate the names of the hosts on the network.

An attempt made to send e-mail to the user at `karanjit@rama.kinetics.com` or `karan@shiv.kinetics.com` will fail because there is no such entry in the SMTP mapping database. A side benefit of this approach is that the remapping provides a common e-mail naming convention for the entire organization.

Rules can be specified to allow or block SMTP traffic between hosts or networks. Mail can also be configured to pass through the InterLock without requiring mapping. But in this case, replies will reveal the internal host names on the outbound headers.

InterLock X Services

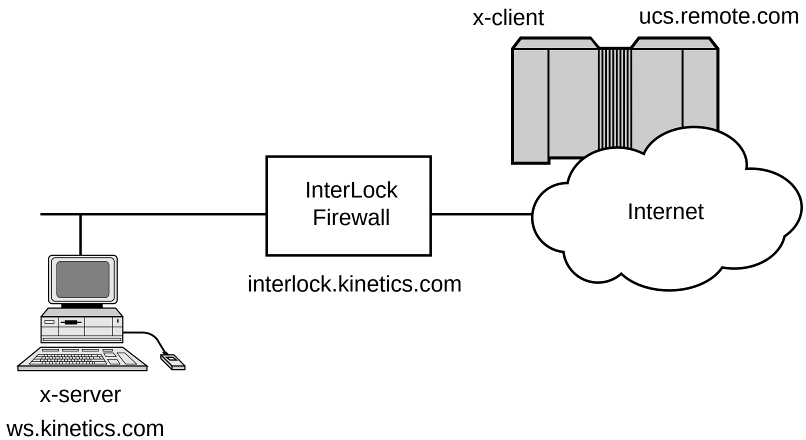
The InterLock X daemon acts as a combination of a pseudo-X server and X-client. Suppose that an internal user wants to run an X-application that resides on a remote host (ucs.remote.com) on a local workstation (ws.kinetics.com). This situation is shown in figure 9.19. The user must telnet to the InterLock system where the user will receive the standard destination prompt after authentication. The user then runs the startX command and specifies the display number:

```
!startX ucs.kinetics.com:0
```

The exclamation mark (!) before the startX command tells the telnet daemon to run the command instead of specifying a destination host.

Figure 9.19

InterLock X services.



The following shows a typical X-session:

```
Destination [interlock.kinetics.com]: !startX ws.kinetics.com:0
interlock.kinetics.com will now accept X connections for screen 108.
```

```
Telnet to the machine which has your X client and use this host
and screen number in the display argument of your X client. e.g.:

         xterm -display interlock.kinetics.com:108

The server will shutdown if not used within the next 3 minutes.

The server will shutdown after 20 minutes of inactivity.
```

The user must then Telnet to the remote host on which the X client application resides by specifying the host name at the destination prompt. The user must then execute the xterm command at the remote host within the next three minutes, as follows:

```
Destination [interlock.kinetics.com]: ucs.remote.com
ucs% xterm -display interlock.kinetics.com:108
```

The InterLock X daemon then reads the X-packets from the host ucs.remote.com on port 6108 (X uses 6000 + *pseudo_screen_ number*), and display the graphics on ws.kinetics.com:0. The user must also give the InterLock gateway access permissions to his X-server, using the following command:

```
xhost +interlock.kinetics.com
```

Before establishing the X-sessions, the InterLock gateway checks its ACRB to see if the user has appropriate permissions for using Telnet and X-sessions to the remote host.

For security reasons, the InterLock gateway monitors the telnet session for inactivity. If there is no activity for 20 minutes, the session and the X-server shuts down. The time-out is configured in the ILconfig file discussed earlier.

InterLock NNTP (Network News Transfer Protocol) Services

Under normal circumstances, when there is no InterLock firewall connecting to the external network, the news readers in the local network point to the local news servers, as shown in figure 9.20. On Unix systems, this is done by setting the NNTPSERVER environment variable to the news server host. A typical setting to run the news reader might be the following:

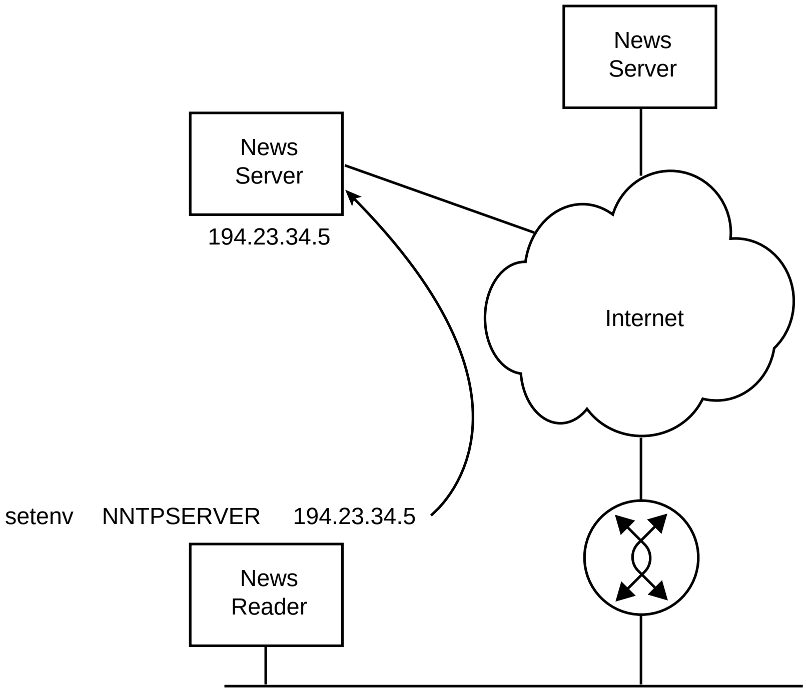
```
% setenv NNTPSERVER 194.23.34.5    # set the news environment variable
% xrn                              # Run the newsreader
```

With the InterLock gateway positioned between the internal hosts and the Internet, the news server must be contacted through the firewall (see fig. 9.21). The InterLock firewall does not allow direct news traffic between the internal and external networks. The internal users must, therefore, specify the InterLock gateway as their news server:

```
% setenv NNTPSERVER interlock.kinetics.com    # set the news environment
variable to point to firewall
% xrn                                         # Run the newsreader
```

FIGURE 9.20

New Reader configuration with no InterLock firewalls.



The InterLock gateway has a local application (/interlock/bin/nntpd) that listens for NNTP connection requests. When it gets a request, *nntpd* performs the following actions:

- ✣ It determines if the connection originated from the internal or external network.
- ✣ Checks the ACRB to see if the connection request is authorized. Unauthorized connection requests are logged in the file nntpd.log.
- ✣ If connection is authorized, nntpd opens a second connection request to the appropriate NNTP server. If the connection originated from the external network, and if an internal news server is defined, it will establish a connection to the internal news server defined by the field `NNTP_Private` in the ILconfig file. Similarly, if the connection originated from the internal network, nntpd will establish a connection to the external news server defined by the field `NNTP_Public` in the ILconfig file.

- nntpd reads data from one interface and copies it to the other interface, thus, acting as a circuit-gateway. If the `Post_Heading_Remapping` in the ILconfig file is set to On, it performs mapping operations in the posting header fields of the news messages. As a result of these mapping operations, it appears as if the news originated from the firewall, and all references to internal host names are remapped to the firewall host name. To perform this mapping, the nntpd can use a separate database (files nntpd.mf.dir, nntpd.mf.pag) or the same database used by SMTP mail (files smtpd.mf.dir, smtpd.mf.pag).

The InterLock gateway does not require password identification for users accessing the NNTP service because the NNTP protocol currently does not support any authentication. However, the InterLock can deny or allow access to services based on the value of the From: field in the news message post header.

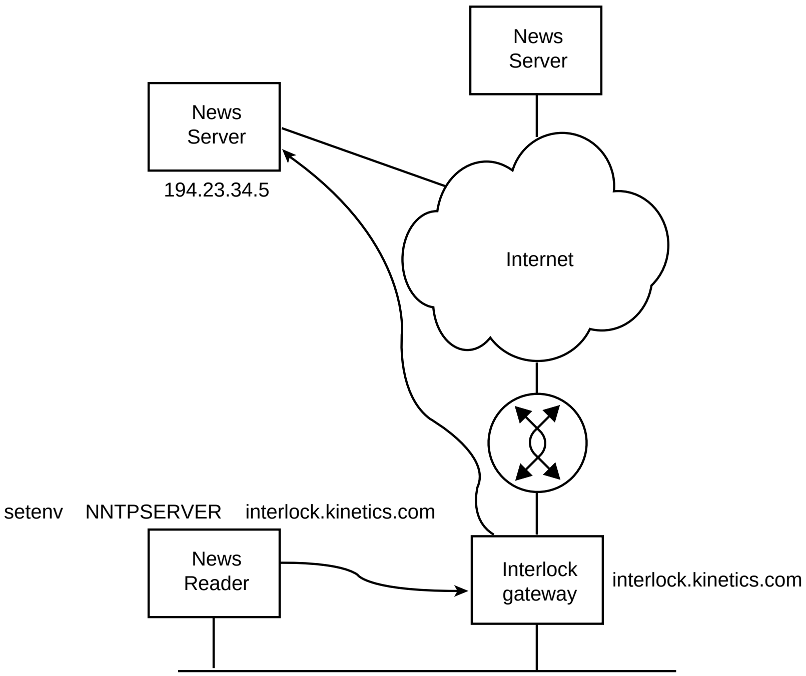


FIGURE 9.21
News Reader configuration through InterLock firewalls.

INTERLOCK HTTP (HYPER TEXT TRANSPORT PROTOCOL) SERVICES

The InterLock gateway can act as a HTTP forwarder by providing WWW proxy agents on the gateway. The InterLock forwarder can also work with WWW clients that support the

CERN proxy model (such as NCSA's Mosaic and University of Kansas' Lynx). By setting the `http_proxy` environment variable in the WWW clients, HTTP requests are directed to the InterLock gateway for resolution.

InterLock Gopher Services

To access gopher services through the InterLock firewall, users must point their gopher clients to the InterLock gateway. The InterLock gateway can act as a data-less gopher server with menu entries but no associated data, or point to a real gopher server. The gopher protocol involves the exchange of selector information between the gopher client and server. The InterLock gateway forwarder replaces the returned selector with a *Uniform Resource Locator* (URL) containing the type, selector, and port associated with the real gopher server. The host field containing internal host names are replaced with the appropriate InterLock name. These translations occur transparently.

The user needs to know how to convert the URL advertised by external gopher servers to an appropriate InterLock format; otherwise, the InterLock gateway rejects the user's requests. Most gopher clients have mechanisms to specify information on the gopher servers to connect to. The gopher protocol sends information in four fields separated by the tab character and ending in a carriage-return/line-feed combination character (see RFC 1436). An example of the entries in these four fields that are needed to contact a remote gopher server is shown as follows:

```
Type = 0            # Gopher item is a file
Name = Remote Corp Gopher
Path = 0/gopher/recipes
Host = gopher.remote.com
Port = 70
```

Because the InterLock gateway hides knowledge of public hosts from the internal network, the gopher clients cannot directly connect to the host gopher.remote.com. This host name must be replaced by the InterLock gateway (interlock.kinetics.com in this example). The gopher client must access the external gopher using the following information:

```
Type = 0            # Gopher item is a file
Name = Remote Corp Gopher
Path = gopher://gopher.remote.com:70/0/gopher/recipes
Host = interlock.kinetics.com
Port = 70
```

For sites using the data-less model, a gopher root menu is constructed on the InterLock gateway for user connection from the internal network. A different root menu can be created for users attempting connections from the external network.

InterLock Generic Protocol Daemon Services

The *Generic Protocol Daemon* (GPD) implements a TCP and UDP forwarder and is meant for those applications that are not directly supported as InterLock application gateways. The TCP and UDP forwarders query the ACRB to determine if connection requests should be granted. Because no authentication is required, no user-level access controls, such as time-of-day, day-of-week, encryption required, and so on, can be specified.

To illustrate how the GPD works, consider the following example. The security policy requires that that organization's clients on the external networks access a special application on an internal host at TCP port 81. The TCP forwarder can specify a rule that consists of the IP address of the server, and the TCP port. Users on the external network can be informed to connect to port 81 on the InterLock gateway. When a request is received for port 81, the InterLock gateway queries the ACRB and checks if this connection is allowed. If the connection is authorized, it establishes a second connection from the InterLock gateway to port 81 on the internal host.

The GPD can be used to support services such as lpr, archie, nslookup, whois, traceroute, and dig.

Additional Source of Information on ANS InterLock

For additional information on the ANS InterLock product and related services contact:

Advanced Network & Services, Inc.
1875 Campus Commons Drive
Suite 220
Reston, VA 22091
Phone: (703)758-7700
Fax: (703)758-7717
E-Mail: `info@ans.com`

Trusted Information Systems Gauntlet

Gauntlet is a firewall product from Trusted Information Systems, Inc. (TIS). The product can run on the Intel-made *Classic-R* system. The base system consists of a board VGA adapter connected to a WYSE monochrome VGA monitor, two serial ports, and one parallel port.

An Adaptec SCSI controller is used to connect to a 440-MB SCSI drive. Additional SCSI devices can be added through the standard Centronics "D" connector on the Adaptec SCSI card. The CPU is an Intel 486 with a 66-MHz clock, 256 KB of cache memory, and 16 MB of main memory. The system board comes with a *zero insertion force* (ZIF) socket that can be used to upgrade to the Pentium processor.

The base operating system used for the Gauntlet product is BSD Unix from Berkeley Software Design, Inc (BSDI). The Gauntlet product comes pre-installed with BSD Unix. This pre-installed version is a subset of the binaries and utilities that normally come with BSD Unix from BSDI.

In the Gauntlet system, the kernel of BSD Unix has been modified to identify and log certain network-based threats. End users can reconfigure and rebuild the kernel, but care should be exercised to ensure that the custom networking utilities are linked into the kernel.

The Gauntlet system includes a single embedded license from BSDI. This is a special low-cost license restricted for using BSDI's BSD Unix for firewall processing. The license does not extend to general use of the system. The pre-installed BSD Unix system has, in itself, the capability to be used as a general-purpose computing platform, but this type of use is in violation of the licensing policy.

The Gauntlet system provides firewall-based proxy services for the following:

- TELNET
- rlogin
- FTP
- SMTP
- NNTP
- Gopher+
- HTTP
- X-Windows

For each service, there is a separate secure forwarding server that performs protocol-specific access control and auditing.

Configuration for the different application-level services is done through a screen-based systems interface. Operational tools include those for installing binary upgrades, checking digital signatures of system files against a database of cryptographic checksum (to detect alteration in these files), tools for audit log reduction and reporting, system alerts, and trouble monitoring.

Gauntlet's base policy is that all systems, other than Gauntlet itself, are initially untrusted. Configuration, therefore, involves explicitly adding trusted systems. In its initial default configuration mode, Gauntlet does not permit any data traffic through the firewall.

CONFIGURATION EXAMPLES USING GAUNTLET

The Gauntlet system has two Ethernet interfaces. Typically, these are Intel Express boards with both Thin Wire and 10BASE-T connectors. Figure 9.22 shows a typical use of the Gauntlet firewall. The Gauntlet network interfaces are labeled as ex0 and ex1. The *ex0* interface is connected to the outside network, and the *ex1* interface is connected to the inside network. The Gauntlet acts as a firewall between the inside and outside networks.

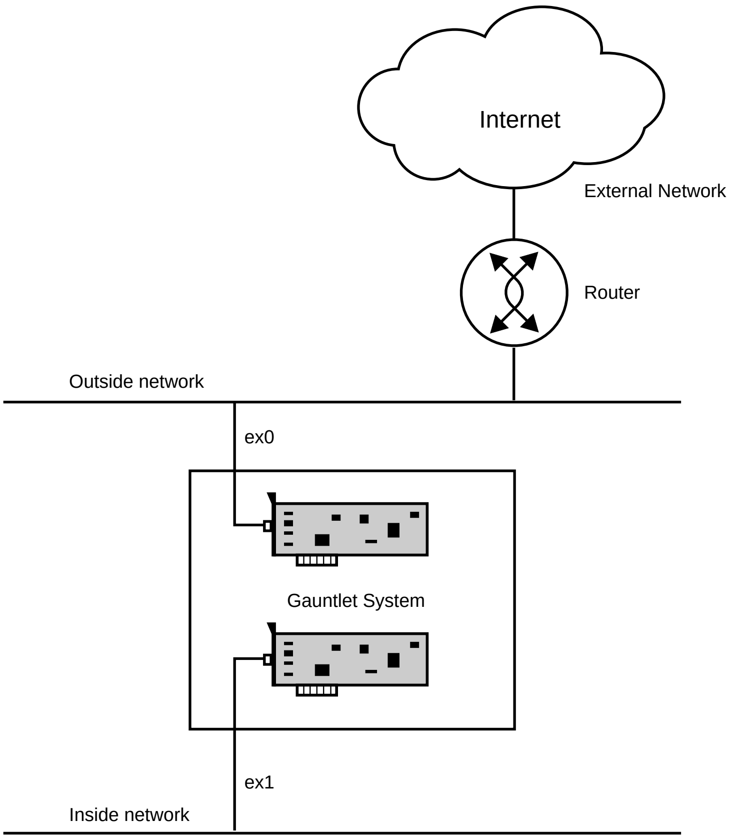


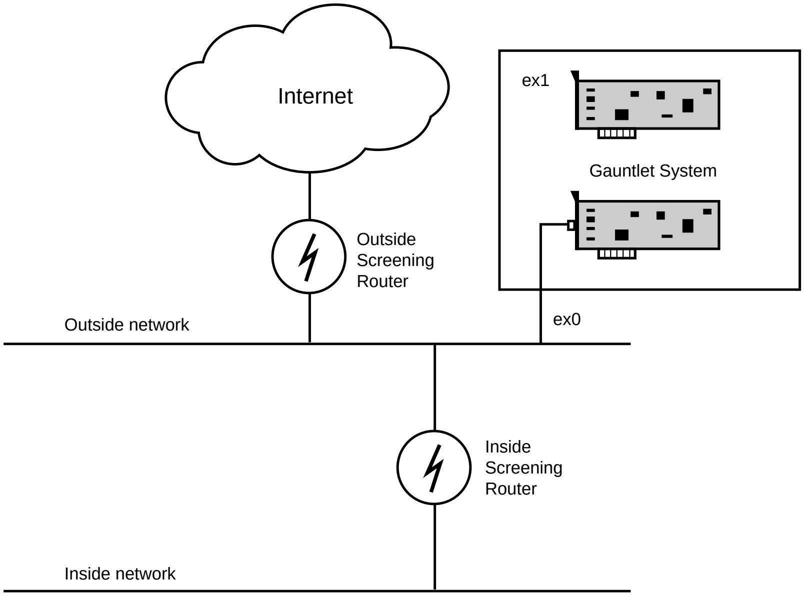
FIGURE 9.22

Typical Gauntlet firewall use.

Another possible use of the Gauntlet system is illustrated in figure 9.23. In this configuration, the Gauntlet system connects to the outside network through its network interface ex0 only. The network interface ex1 is not connected. A screening router is used to connect the outside network with the external network (Internet in this case), and another screening router is used to connect the outside and internal networks. The security of the network depends on a combined use of the screening routers and the Gauntlet system. If the screening routers are not set up correctly, the Gauntlet system could be completely bypassed.

Figure 9.23

The Gauntlet system using single network interface.



When setting up the routing tables for the screening routers, and the default route for the Gauntlet system, it is best to specify static routing to avoid accidental changes, and foil intruder attempts to change the routing tables. If your network policy requires that dynamic routes be used, the system can use the routed or gated daemons for providing dynamic routing.

Configuring Gauntlet

Configuration of the Gauntlet system can be done either through the menu interface or a command-line interface. A primary step is to configure the connected Ethernet interfaces with appropriate values for IP address, subnet mask, and broadcast address.

If the inside network interface, ex1, is not to be connected (refer to figure 9.23), it should not be configured. Additionally, you can specify the default route, which typically is the router used to connect to the Internet. A default DNS server can be specified initially to simplify testing while the system is being installed.

The following shows Gauntlet's main management menu. This list shows the type of administration options that are available through the menu interface. These options are performed using shell scripts that invoke the Gauntlet-specific utilities and commands.

```
               Gauntlet(tm) Main Management Menu
               --------------------------------

Select option: System Backups
               System Integrity Checks
               Install Software Upgrade
               System Event/Reporting Configuration
               Trusted Network Configuration
               System Access Configuration
               User Authentication Management
               IP Address Configuration
               DNS Configuration
               Electronic Mail Configuration
               NNTP Forwarder Configuration

                   Exit Configuration Menus
                            Help
                      Menuing System Help

             To navigate, use arrow keys and Tab
          To select a highlighted item, press Enter
```

The different Gauntlet options in the previous list are briefly explained so you can form an idea of the typical administration tasks needed to configure Gauntlet.

The *System Backup* option is used to configure and perform backups of the Gauntlet system. System backups are performed by using a SCSI tape drive. The BSD Unix *dump* utility is used for backups. The dump levels that are used are levels 0, 5, and 9. Level 0 represents the full-system backup. Level 5 represents backups of all data modified since dump level 5 or lower. Level 9 represents backups of all data modified since dump level 9 or lower. If a backup device is not available for the Gauntlet system, the backup can be performed on a network tape server. The configuration for this is also completed with this option.

The *System Integrity Checks* option is used to detect files that have been modified, deleted, or replaced. It can be used to alert the system administrator if accidental or deliberate changes (to lower the security, perhaps) have been done to the Gauntlet system. After making legitimate changes, this option can be used to update the integrity of the database. The System Integrity Checks' database can be copied to different devices to protect them for later use. Options exist to copy the systems integrity database to and from a floppy disk. You can also check the system integrity against databases stored on a floppy disk or network server. You can check the system release levels against an online database, a floppy disk, or a network server.

The *Install Software Upgrade* option is used to implement software patches and upgrades. Floppy disk upgrades are issued in the tar archive that contains multiple archives and the file "upgrade.cat," which lists the upgrade on the floppy disk.

The *System Event/Reporting Configuration* option is used to specify Gauntlet's report configuration and event reporting options. Reports can be generated on a daily or weekly basis. You can specify the e-mail address to which reports should be mailed. Unless you configure otherwise, the reports are mailed to the alias named "firewalladmin," defined in the file /etc/aliases. Even the daily and weekly reports are disabled; they are stored in the files /var/log/daily.out and /var/log/weekly.out so you can access them at a later time. The daily and weekly reports are run from the standard daily and weekly processing scripts /etc/daily and /etc/weekly. An alarm reporting option can be enabled, which allows you to specify the times at which the logs will be scanned for "noteworthy" events.

NOTE

> Gauntlet uses an interesting approach to define that which is noteworthy. It does not attempt to list events that are noteworthy because this could leave out unexpected events that have not been defined. Instead, Gauntlet enables the administrator to define which events are ordinary. Noteworthy events are defined as those that are not ordinary. A file called /usr/local/etc/frequentcheck.ignore contains regular expression patterns defining ordinary events.

The *Trusted Network Configuration* enables you to define a list of trusted and untrusted networks, which are stored in /usr/local/etc/trusted-networks and /usr/local/etc/untrusted-networks. By default, the only network that Gauntlet trusts is its loopback interface (127.0.0.1)—that is, Gauntlet trusts only itself. The default untrusted network value is "*" (all other networks). The trusted and untrusted network files are used to build a copy of the firewall proxy permissions table file /usr/local/etc/netperm-table.

The *System Access Configuration* option is used to determine whether the administrator should be allowed to telnet or rlogin to the system. By default, these options are disabled, which means that the administrator must have physical access to the console.

The *User Authentication Management* is used to add, delete, or edit user configuration information. This screen can be used to specify the authentication mechanism that will be used for the user's login. The following authentication methods can be used with Gauntlet:

- **Plain passwords.** This is not recommended when logging in over an untrusted network.

- **SNK.** This is Digital Pathways Secure Net Key. Gauntlet supports the SNK authentication calculator.
- **S/KEY.** This is the freely available S/Key authentication system developed by Bellcore. Software for this is included, but TIS does not provide support for it.

The *IP Address Configuration* option is used to configure the network interfaces for the Gauntlet system. This option can be used to change the host name, domain name, IP address, subnet mask, broadcast address, default DNS server, and default router. If the internal network is subnetted and additional routes need to be defined for the Gauntlet system, the *route add* command can be used to add these routes in the /etc/netstart file. If the DNS software is to run on the Gauntlet system, the /etc/resolv.conf, which is used to list the DNS servers to contact for name resolution, must contain an entry for the Gauntlet system. This can be done by placing the following in the /etc/resolv.conf file:

```
nameserver 127.0.0.1
```

The *DNS Configuration* option is used to edit host tables (/etc/hosts), reload the DNS configuration from the host table, and configure DNS options. It can be used as a quick method for setting up a DNS server. For more complex configurations, you will have to edit and configure the DNS software directly, which would involve configuring the /etc/named.boot and the zone files referenced by this file.

The *Electronic Mail Configuration* option is used to specify if e-mail is to be forwarded to an internal mail hub, the postmaster for the Gauntlet system's e-mail address, or if mail headers should be modified to indicate that mail originated from the domain itself. This hides knowledge of hosts from the outside world. If there is no internal mail hub, you should not modify the mail headers; otherwise, return addresses might not be replyable.

The *NNTP Forwarder Configuration* is used to specify the internal and external NNTP servers between which news feed flows. Both internal and external news server should be configured to feed news to the Gauntlet system. The Gauntlet system uses proxy agents to forward the news.

USER'S VIEW OF USING THE GAUNTLET FIREWALL

Using a firewall such as Gauntlet changes the user's interaction for some standard services, such as Telnet, ftp, and rlogin. The user must first connect to the proxy server running on the firewall, then specify the destination to connect to. The application proxy on the firewall performs the requested action on behalf of the user. Sending e-mail does not require any change in the user interface, as long as the firewall is configured properly.

USING FTP WITH THE GAUNTLET FIREWALL

Assume that the user wants to do an anonymous FTP session to DS.INTERNIC.NET. The user must first connect to the FTP proxy server running in the firewall, as follows:

```
% ftp gauntlet.kinetics.com
```

Once connected to the server, the user is prompted for the user name and host name to connect to the remote FTP server. You have to format this in the following RFC-822 address syntax:

username@hostname

For the example in this section, you must specify this as the following:

anonymous@ds.internic.net

A typical session through the Gauntlet system from a host on the internal network follows:

```
ucs% ftp gauntlet.kinetics.com
Connected to gauntlet.kinetics.com
220 gauntlet FTP proxy (Version N stable) ready.
Name (gauntlet:you): anonymous@ds.internic.net
331-(----GATEWAY CONNECTED TO ds.internic.net----)
331-(220 ds.internic.net FTP server (Version 5.60 ready.)
331 Guest login ok, send ident as password.
Password: password not echoed
230 Guest login ok, access restrictions apply.
ftp> dir
200 PORT command successful.
```

When you see the message GATEWAY CONNECTED TO, it means that the FTP proxy server has established connection to the remote host. From then on, all commands you enter are forwarded to the remote host by the proxy. The proxy also returns all status messages for you.

The following shows the situation when you are using the FTP from an untrusted network and want to use an FTP server on an internal host. This sequence assumes that the SNK method of authentication has been enabled at the gauntlet firewall. Assume that the DNS name of the firewall from the outside is gatekeeper.kinetics.com.

```
% ftp gatekeeper.kinetics.com
Connected to gatekeeper.kinetics.com.
220-Before using the proxy you must first authenticate
220 gatekeeper FTP proxy (Version N stable) ready.
Name (gatekeeper.kinetics.com:you):
331 SNK Challenge "81416":
Password: password not echoed
230 User authenticated to proxy
ftp> user kss@ucs.kinetics.com
331-(----GATEWAY CONNECTED TO ucs.kinetics.com----)
```

```
331-(220 ucs.kinetics.com FTP server (Version 5.60 ready.)
331 Password required for you.
Password: password not echoed
ftp>
```

Using Telnet with the Gauntlet Firewall

The user must first connect to the Telnet proxy server running in the firewall, using the following line:

```
% telnet gauntlet.kinetics.com
```

After a short pause, the command prompt for the proxy server appears. Use **connect *hostname*** of the c *hostname* command to connect to the remote host. If the system is unreachable, the proxy informs you. You can then use the quit command to break the connection to the gateway. The following commands illustrate this sequence:

```
% telnet gatekeeper.kinetics.com
Trying 199.245.180.15 ...
Connected to gatekeeper.kinetics.com
Escape character is ^].
gatekeeper telnet proxy (Version N) ready:
tn-gw-> c ucs.siyan.com
ucs BSD 386/ v1.0 (ttys1)

login: enter loginname
Password: entered password not displayed
You have mail.
```

Using rlogin with the Gauntlet Firewall

The user must first connect to the rlogin proxy server running in the firewall, using the following line:

```
% rlogin  gauntlet.kinetics.com
```

After a short pause, the command prompt for the proxy server appears. Use the connect *hostname* or c *hostname* command to connect to the remote host. If the system is unreachable, the proxy informs you. Unlike Telnet, you do not have to authenticate the user twice.

Consider the following example where the rlogin server on ucs.siyan.com prompted for a password, because this firewall host was not in the user's .rhosts file.

```
% rlogin  gatekeeper.kinetics.com
rlogin-gw-> ?
Valid commands are: (unique abbreviations must be used)
   connect hostname
   help/?
   quit/exit
   password
```

```
rlogin-gw> c ucs.siyan.com
Password: entered password not displayed
Trying you@199.245.180.10
Last login: Wed Nov 16 17:45:23 from world.std.com
ucs BSD 386/ v1.0 (ttys1)

You have mail.
ucs%
```

If a user name and host name are specified, the rlogin proxy automatically reconnects to the specified remote system, as the following lines show:

```
% rlogin  gatekeeper.kinetics.com -l kss@rams.siyan.com
Trying you@199.245.180.12
Last login: Wed Nov 16 18:34:12 from world.std.com
ucs BSD 386/ v1.0 (ttys1)

You have mail.
ucs%
```

If you are logging in from an external site and the SNK authentication is used, you will see something similar to the following:

```
% rlogin gatekeeper.kinetics.com -l kss@si.kscs.com
Username: kss
SNK Challenge "23451": enter numeric code
Login Accepted
Trying you@193.25.10.3
Last login: Wed Nov 16 18:34:12 from world.std.com
ucs BSD 386/ v1.0 (ttys1)

You have mail.
ucs%
```

THE TIS FIREWALL TOOLKIT

Trusted Information Systems, Inc., the makers of the commercial Gauntlet product, also make available a free firewall package called the TIS Firewall Toolkit. Much of the functionality of the commercial Gauntlet product is built on top of this toolkit.

The TIS Firewall Toolkit is a collection of tools, from which you can pick and choose to implement the services that you need. The TIS Firewall Toolkit can be obtained by anonymous FTP from the following source:

FTP server: FTP.TIS.COM

Path: /pub/firewall/toolkit

> For a thorough presentation on the construction of a firewall using the TIS Firewall Toolkit, see Chapter 10 of this book.

NOTE

BUILDING THE TIS FIREWALL TOOLKIT

After you have downloaded the software package from the source mentioned in the previous section, you must compute and build the firewall toolkit on your system.

The firewall toolkit is a Unix compressed tar file. In this discussion, the name of this file shall be fwtk.tar.Z.

Copy this file into a working directory on your system and use the uncompress or gunzip commands to uncompress the file. You then must extract the files from the archive. An example of these commands for BSD Unix follows:

```
uncompress fwtk.tar.Z
tar -xvBf fwtk.tar
```

When the files are extracted, change the UID and GID on the extracted files to avoid any accidental access to these files. On BSD Unix systems, you can perform the following to assign the UID and GID for root and wheel.

```
chown -R root:wheel fwtk
```

Go to the fwtk directory under which the source files have been extracted. The file Makefile references the file Makefile.config. You should change some target definitions for this file. A listing of the Makefile.config follows:

```
#
#  Copyright (c) 1993, Trusted Information Systems, Incorporated
#  All rights reserved.
#
#  Redistribution and use are governed by the terms detailed in the
#  license document ("LICENSE") included with the toolkit.
#


#
#    Author: Marcus J. Ranum, Trusted Information Systems, Inc.
#
# RcsId: "$Header: Makefile.config,v 1.3 94/11/01 12:04:59 mjr rel $"
```

```
# Your C compiler (eg, "cc" or "gcc")
CC=    cc


# program to use for installation -- this may or may not preserve
# old versions (or whatever). assumes that it takes parameters:
# copy source dest
CP=    cp


# Options for your compiler (eg, "-g" for debugging, "-O" for
# optimizing, or "-g -O" for both under GCC)
#COPT=    -g -traditional
COPT=    -g
#COPT=    -O


# Version of "make" you want to use
#MAKE=    gnumake
MAKE=    make


# Your ranlib utility (use "touch" if you don't have ranlib)
RANLIB=    ranlib
#RANLIB=    touch


# Destination directory for installation of binaries
DEST=    /usr/local/etc


# Destination directory for installation of man pages
#DESTMAN=    $(DEST)/../man


# Names of any auxiliary libraries your system may require (e.g., -lsocket)
# If you want to link against a resolver library, specify it here.
AUXLIB= -lresolv
#AUXLIB= -lsocket


# DBM library should be specified if it is an external library or
# you with to use a different one than what is included in libc
#DBMLIB=    -lndbm
DBMLIB=
```

```
# Flags to pass to the linker (eg, -static for static binaries under GCC,
# or -Bstatic for static binaries under SunOS 4.1.x)
#LDFL=    -Bstatic
#LDFL=
LDFL= -g


# Location of the fwtk sources [For #include by any external tools needing
it]
FWTKSRCDIR=/u/b/mjr/firewall/fwtk
#FWTKSRCDIR=/usr/local/src/fwtk


# Location of X libraries for X-gw
#XLIBDIR=/usr/X11/lib
XLIBDIR=/usr/local/X11R5/lib


# Location of X include files
#XINCLUDE=/usr/X11/include
XINCLUDE=/usr/local/X11R5/include
```

You can change any of the variables defined in these configuration files, but pay special attention to the directories in which the files are referenced and installed. You should think about the settings of at least the following variables:

DEST, DESTMAN, AUXLIB, FWTKSRC, XLIBDIR, XINCLUDE

On BSDI Unix, you should change XLIBDIR to /usr/X11/lib and XINCLUDE to /usr/X11/include.

After making changes to the Makefile.config, execute the make command:

```
make
```

If you see compilation errors, make the appropriate fixes.

On BSD Unix, the author encountered the following error messages when trying to build the tools:

```
"Makefile", line 16: Need an operator
Fatal errors encountered -- cannot continue
```

This error message is caused by the fact that the "include" statements in the Makefiles in the fwtk directory and the subdirectories where the separate tools reside use a form of include that is incompatible with make on BSDI Unix. If you encounter the previous error, it is because the include statement in the Makefiles has the following syntax:

```
include name_of_file
```

You should change this to an include that uses the following syntax:

```
.include "name_of_file"
```

Another problem the author encountered was the conflict in getenv() and setenv() routines in the files in the fwtk/x-gw directory and the routines defined in the C-library. You can resolve this conflict by suitably renaming the functions that cause conflicts.

After you successfully compile and link the tools, you should run the following command from the fwtk directory:

```
make install
```

This command installs the tools in the directory specified in the DEST variable in the Makefile.config. To access these tools more conveniently, place the fwtk directory on your search path.

Configuring the Bastion Host with Minimal Services

You should configure the host on which you will be running the TIS Firewall Toolkit with only those services that are needed. This requires system-specific knowledge. The following is a partial list:

- Edit the system startup files, such as the /etc/rc, /etc/netstart, and /etc/rc.local (BSD Unix).
- Edit the operating system configuration and eliminate any undesirable kernel-based services, such as NFS, and rebuild the kernel.
- Edit the /etc/inetd.conf file, which specifies the internet daemon controlled services. Remove services that are not needed such as TFTP.

At each step, use the ps command to verify the services that are running. On a firewall, a reduced configuration should typically have the following processes:

```
% ps -aux
USER       PID %CPU %MEM   VSZ  RSS TT  STAT STARTED      TIME COMMAND
root         0  0.0  0.0     0    0 ??  DLs   2:35AM   0:00.00 (swapper)
root         1  0.0  0.0   652  116 ??  Is    2:35AM   0:00.11 init --
root         2  0.0  0.0     0   12 ??  DL    2:35AM   0:00.00 (pagedaemon)
```

```
root         29  0.0  0.0   608  112 ??  Ss     2:36AM    0:00.30 syslogd
root         46  0.0  0.0   520   28 ??  Ss     2:36AM    0:00.11 update
root         48  0.0  0.0   676  152 ??  I      2:36AM    0:00.38 cron
root         74  0.0  0.0   652  104 ??  Is     2:36AM    0:00.14 inetd
root         87  0.0  0.0   972  412 co  Is+    2:36AM    0:00.66 -tcsh (tcsh)
root        488  0.0  0.0   692  172 p0  R+     5:06PM    0:00.07 ps -aux
%
```

You can also use the netstat -a command to see a status of the network services that are running, as follows:

```
% netstat -a
Active Internet connections (including servers)
Proto Recv-Q Send-Q  Local Address          Foreign Address        (state)
tcp        0      0  *.time                 *.*                    LISTEN
tcp        0      0  *.daytime              *.*                    LISTEN
tcp        0      0  *.chargen              *.*                    LISTEN
tcp        0      0  *.discard              *.*                    LISTEN
tcp        0      0  *.echo                 *.*                    LISTEN
tcp        0      0  *.finger               *.*                    LISTEN
tcp        0      0  *.telnet               *.*                    LISTEN
tcp        0      0  *.ftp                  *.*                    LISTEN
tcp        0      0  *.smtp                 *.*                    LISTEN
tcp        0      0  *.sunrpc               *.*                    LISTEN
udp        0      0  *.time                 *.*
udp        0      0  *.daytime              *.*
udp        0      0  *.chargen              *.*
udp        0      0  *.discard              *.*
udp        0      0  *.echo                 *.*
udp        0      0  *.659                  *.*
udp        0      0  *.syslog               *.*
Active Unix domain sockets
Address  Type   Recv-Q Send-Q    Inode     Conn     Refs  Nextref Addr
fe46c400 dgram       0      0        0 fe396a94        0 fe3eea14
fe46c800 dgram       0      0        0 fe396a94        0 fe453994
%
```

You also can use the *portscan* tool that is in the fwtk directory under tools/admin/portscan to scan the ports on a host to see if any ports are active. You first will have to run make on the Makefile in the tools directory. Figure 9.24 shows the results of scanning ports 500 to 515 on host ucs.kinetics.com. The ports that are active on the host are displayed.

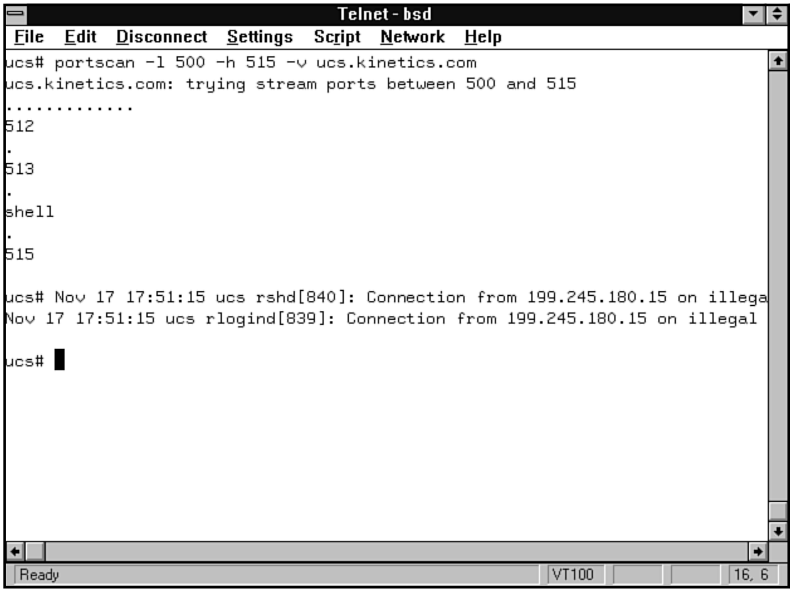


FIGURE 9.24
Using the FWTK portscan tool.

INSTALLING THE TOOLKIT COMPONENTS

When you run the make install command after successfully compiling the individual components, the binaries are installed in the directory specified by the DEST variable in the fwtk/Makefile.config file. The default is to install the tools in the /usr/local/etc directory.

Most of the toolkit components are invoked by the inetd daemon. The configuration for inetd is specified in the /etc/inetd.conf file. You must edit this file to invoke the firewall proxy components. The access control configuration information for the toolkit components is kept in the netperm-table file. By default, this file is found in the /usr/local/etc directory.

The following listing shows the partial contents of the /etc/inetd.conf file configured for the firewall.

```
#
# Internet server configuration database
#
#    @(#)inetd.conf    5.4 (Berkeley) 6/30/90
#
ftp       stream    tcp    nowait    root      /usr/local/etc/netacl
ftpd
ftp-gw    stream    tcp    nowait    root      /usr/local/etc/ftp-gw
ftp-gw
```

```
telnet    stream    tcp    nowait    root      /usr/local/etc/tn-gw
tn-gw
tn-admin  stream    tcp    nowait    root      /usr/local/etc/netacl
telnetd
login     stream    tcp    nowait    root      /usr/local/etc/rlogin-gw
rlogin-gw
finger    stream    tcp    nowait    nobody    /usr/local/etc/netacl
fingerd
smtp      stream    tcp    nowait    root      /usr/local/etc/smap
smap
echo      stream    tcp    nowait    root      internal
discard   stream    tcp    nowait    root      internal
chargen   stream    tcp    nowait    root      internal
daytime   stream    tcp    nowait    root      internal
time      stream    tcp    nowait    root      internal
echo       dgram    udp    wait      root      internal
discard    dgram    udp    wait      root      internal
chargen    dgram    udp    wait      root      internal
daytime    dgram    udp    wait      root      internal
time       dgram    udp    wait      root      internal
```

The netacl is a TIS firewall toolkit program that is invoked for standard ftp and administrative telnet sessions. It provides network access control for TCP-based services based on the network permissions table (/usr/local/etc/netperm-table). Usually, the netacl is configured to allow hosts on the internal network limited access to the gateway. The netacl is similar to the TCP Wrapper program discussed earlier in this chapter. This netacl program is designed to work with the network permissions table used by the TIS firewall toolkit.

If a telnet proxy gateway service has to be set up, it must be assigned the telnet standard port of 23 because the telnet program often disables options processing when connecting to a port that is different from this standard port. The telnet proxy service requires options processing so it must be connected to the standard telnet port. In the /etc/inetd.conf listing shown previously, this is done by the following entry:

```
telnet    stream    tcp    nowait    root    /usr/local/etc/tn-gw    tn-gw
```

The telnet gateway program can be found in /usr/local/etc/tn-gw . The tn-gw in the last column is the name of the service. When the /usr/local/etc/tn-gw program runs, it consults the permissions table to see if this access should be permitted or denied.

The meaning of the other fields are as follow: The *telnet* in the first column is the name of the service defined in the /etc/services file. The field *streams*, in the second column, refers to the TCP byte stream service. The field *tcp* in the third column refers to the protocol service which is TCP. The field *nowait* in the fourth column refers to the fact that when inetd receives a connection request, it does not have to wait, but can easily begin listening for additional connection requests. The field *root* in the fifth column is the user name under whose permissions the service runs.

If standard port 23 is used by the telnet gateway service and administrative access to the standard telnet service is required, the standard telnet service must be defined on another port. For example, you can define the standard telnet service on port 24, and make the following entry in the /etc/services file:

```
tn-admin        24/tcp
```

Then, in the /etc/inetd.conf file, you can make the following entry:

```
tn-admin stream tcp    nowait    root    /usr/local/etc/netacl    telnetd
```

When a connection attempt is made on port 24 of the firewall, the program /usr/local/etc/netacl executes, consulting the network permissions table file /usr/local/etc/netperm-table to determine if access is allowed or not.

The FTP gateway can be configured on any port. You might want to configure the FTP gateway on a different port, if you want internal users to have access to the standard FTP server. In any case, the standard FTP services should be under the control of the netacl wrapper program which can control access to this service.

```
ftp    stream    tcp    nowait    root    /usr/local/etc/netacl    ftpd
```

The FTP gateway service can be configured, as follows, in the inetd.conf file:

```
ftp-gw    stream    tcp    nowait    root    /usr/local/etc/ftp-gw    ftp-gw
```

The following are partial contents of a sample /etc/services file that define the firewall proxy services:

```
#
# Network services, Internet style
#
#    @(#)services    5.8 (Berkeley) 5/9/91
#
echo            7/tcp
echo            7/udp
discard         9/tcp        sink null
discard         9/udp        sink null
systat         11/tcp        users
daytime        13/tcp
daytime        13/udp
chargen        19/tcp        ttytst source
chargen        19/udp        ttytst source
ftp            21/tcp
ftp-gw         22/tcp
telnet         23/tcp
tn-admin       24/tcp
smtp           25/tcp        mail
time           37/tcp        timserver
```

```
time          37/udp        timserver
rlp           39/udp        resource      # resource location
nameserver    42/tcp        name          # IEN 116
whois         43/tcp        nicname
domain        53/tcp        nameserver    # name-domain server
domain        53/udp        nameserver
```

NOTE

When debugging the firewall toolkit configuration, you must restart inetd if you make changes to the /etc/inetd.conf file. You can restart inetd by using the ps command to determine its *process identifier* (PID) and then using kill -HUP on inetd's PID. For example:

```
% ps -aux ¦ grep inetd
root        76  0.0  0.0   652  128 ??  Is    5:27AM    0:00.99
➥inetd
% kill -HUP 76
```

If rlogin proxy is to be installed, it must be installed on the firewall's rlogin service port (512). Rlogin requires binding to a privileged port, an operation that requires system permissions. Because of the permissive nature of rlogin security, you can set the rlogin proxy's directory option to ensure that it runs under chroot.

NOTE

The *chroot* is a mechanism under Unix where a process is confined to a single branch of the file system. After the chroot is performed, the root of the restricted branch of the file system is treated as the root directory, preventing the process from accessing critical parts of the file system, such as the /etc directory that contains the password files and other critical information, and the /dev directory that contains the device files.

THE NETWORK PERMISSIONS TABLE

The netacl and the firewall proxies read the network permissions table file to determine if access is permitted. A sample network permissions table file (/usr/local/etc/netperm-table) follows:

```
#
# Sample netperm configuration table
#
```

```
# To get a good sample working netperm-table, just globally
# substitute YOURNET for your network address (e.g.; 666.777.888)
#

# Example netacl rules:
# --------------------
# if the next 2 lines are uncommented, people can get a login prompt
# on the firewall machine through the telnet proxy
netacl-telnetd: permit-hosts 127.0.0.1 -exec /usr/libexec/telnetd
netacl-telnetd: permit-hosts 199.245.180.9 199.245.180.10 -exec /usr/libexec/
telnetd
#
netacl-ftpd: permit-hosts 199.245.180.9 199.245.180.10 -exec /usr/libexec/
ftpd
netacl-ftpd: permit-hosts unknown -exec /bin/cat /usr/local/etc/noftp.txt
netacl-ftpd: permit-hosts * -chroot /var/anon-ftp  -exec /usr/libexec/ftpd

#
# if the next line is uncommented, the telnet proxy is available
netacl-telnetd: permit-hosts * -exec /usr/local/etc/tn-gw
#
# if the next 2 lines are uncommented, people can get a login prompt
# on the firewall machine through the rlogin proxy
netacl-rlogind: permit-hosts 127.0.0.1 -exec /usr/libexec/rlogind -a
netacl-rlogind: permit-hosts 199.245.180.* -exec /usr/libexec/rlogind -a
#
# if the next line is uncommented, the rlogin proxy is available
#netacl-rlogind: permit-hosts * -exec /usr/local/etc/rlogin-gw

#
# to enable finger service uncomment these 2 lines
netacl-fingerd: permit-hosts 199.245.180.* -exec /usr/libexec/fingerd
netacl-fingerd: permit-hosts * -exec /bin/cat /usr/local/etc/finger.txt
# Example smap rules
# -----------------
smap, smapd:     userid
smap, smapd:     directory /var/spool/sma
smapd:         executable /usr/local/etc/smap
smapd:         sendmail /usr/sbin/sendmai
smap:         timeout 360

# Example ftp gateway rules
# -------------------------
ftp-gw:    denial-msg    /usr/local/etc/ftp-deny.tx
ftp-gw:    welcome-msg    /usr/local/etc/ftp-welcome.tx
ftp-gw:    help-msg    /usr/local/etc/ftp-help.tx
ftp-gw:         timeout 360

# uncomment the following line if you want internal users to be
# able to do FTP with the internet
ftp-gw:        permit-hosts 199.245.180.* -log {retr stor}
```

```
# uncomment the following line if you want external users to be
# able to do FTP with the internal network using authentication
ftp-gw:        permit-hosts * -authall -log { retr stor }

# Example telnet gateway rules:
# ----------------------------
tn-gw:         denial-msg    /usr/local/etc/tn-deny.txt
tn-gw:         welcome-msg    /usr/local/etc/tn-welcome.txt
#tn-gw:         help-msg    /usr/local/etc/tn-help.txt
tn-gw:         timeout 3600
tn-gw:         permit-hosts 127.0.0.1 199.245.180.* -passok -xok
# if this line is uncommented incoming traffic is permitted WITH
# authentication required
#tn-gw:        permit-hosts * -auth
# Example rlogin gateway rules
# -----------------------------
rlogin-gw:    denial-msg     /usr/local/etc/rlogin-deny.tx
rlogin-gw:    welcome-msg     /usr/local/etc/rlogin-welcome.tx
rlogin-gw:    help-msg    /usr/local/etc/rlogin-help.tx
rlogin-gw:    timeout 360
rlogin-gw:    permit-hosts 199.245.180.* -passok -xo
# if this line is uncommented incoming traffic is permitted WIT
# authentication require
rlogin-gw:    permit-hosts * -auth -xo


# Example auth server and client rule
# -------------------------------------
authsrv:     hosts 127.0.0.
authsrv:     database /usr/local/etc/fw-authd

authsrv:    badsleep 1200
authsrv:    nobogus true
# clients using the auth serve
*:        authserver 127.0.0.1 777

# X-forwarder rule
tn-gw, rlogin-gw:    xforwarder /usr/local/etc/x-g
```

Consider the following rules from this sample configuration file for the FTP proxy

```
ftp-gw:    denial-msg    /usr/local/etc/ftp-deny.tx
ftp-gw:    welcome-msg    /usr/local/etc/ftp-welcome.tx
ftp-gw:    help-msg    /usr/local/etc/ftp-help.tx
ftp-gw:        timeout 360
ftp-gw:        permit-hosts 199.245.180.* -log {retr stor
# uncomment the following line if you want external users to b
# able to do FTP with the internal network using authenticatio

ftp-gw:        permit-hosts * -authall -log { retr stor }
```

The meaning of the previous configurations statements is as follows: The hosts on the network 199.245.180.0 are permitted to use the FTP proxy. The "-log {retr stor}" indicates that logging actions for retrieval (GET) and storage (PUT) are enabled. The FTP proxy will timeout and shut the connection for a period of inactivity lasting 3,600 seconds. Also, external users can access the firewall FTP proxy if they are using authentication.

TELNET AND RLOGIN ACCESS

Normal telnet access to the firewall should generally be available to administrators only. To run the telnet/rlogin proxies, normal telnet/rlogin access cannot use the same port. You can use the following approaches for this problem:

- Permit administrative logins through the console of the firewall only.
- Run telnet/rlogin proxies on the standard service port. Run standard telnet/rlogin proxies on alternate ports, and protect them with netacl.
- Use netacl to switch services, depending on the origin of the request.

The first solution might be very restrictive to the administrators and might not always be possible.

The second solution might cause a problem with some telnet/rlogin implementations that do not work properly when connecting to a non-standard port.

The last solution is more convenient because users do not have to specify non-standard ports for standard telnet service. This solution, however, requires proper use of access controls. An example of the rules that provide this service follow:

```
#Telnet access
netacl-telnetd:    permit-hosts    127.0.0.1    -exec    /usr/libexec/telnetd
netacl-telnetd:    permit-hosts    199.245.180.10    -exec    /usr/libexec/
telnetd
netacl-telnetd:    permit-hosts    *        -exec    /usr/local/etc/tn-gw

#Rlogin access
netacl-rlogind:    permit-hosts    127.0.0.1    -exec    /usr/libexec/rlogind
netacl-rlogind:    permit-hosts    199.245.180.10    -exec    /usr/libexec/
rlogind
netacl-rlogind:    permit-hosts    *        -exec    /usr/local/etc/rlogin-gw
```

SETTING UP ANONYMOUS FTP ACCESS

Anonymous FTP user accounts have had their share of security problems, such as bugs in the FTP server implementation or misconfigurations. One approach to making Anonymous FTP more secure is to use the netacl program to "chroot" to a directory dedicated for anonymous FTP, before the FTP server is started. For example, the following lines from the sample network permissions table can be used for this purpose:

```
netacl-ftpd: permit-hosts 199.245.180.9 199.245.180.10 -exec /usr/libexec/
ftpd
netacl-ftpd: permit-hosts unknown -exec /bin/cat /usr/local/etc/noftp.txt
netacl-ftpd: permit-hosts * -chroot /var/anon-ftp  -exec /usr/libexec/ftpd
```

The first line permits only hosts 199.245.180.9 and 199.245.180.10 to connect to the FTP service. In the second line, hosts that are not known to DNS will have displayed the contents of the file /usr/local/etc/noftp.txt. Presumably, this text will have appropriate messages indicating that FTP service is not available to them. The third line permits access to any host, but it "chroots" to the /var/anon-ftp directory before executing the FTP service. When the FTP service starts, it will read the /etc/passwd file for user authentication. Because it has been chrooted to the FTP area, it will read the file /var/anon-ftp/etc/passwd, and not the main /etc/passwd file. This allows user accounts to be created and assigned that are for FTP only and that are independent of user accounts kept in the main /etc/passwd file.

Supporting the Proxy FTP Server and Standard FTP Server on the Same Host

The proxy FTP gateway server can be invoked directly from the dedicated port 21 used for FTP service, or from inside a modified FTP daemon. If the proxy server is run on the standard FTP port, the standard FTP daemon cannot also run on this port, which means that standard FTP services are not available on the host that runs the FTP proxy. However, it is sometimes desirable to have both standard FTP and proxy FTP on the same host. For this reason, the toolkit supports a version of the FTP daemon that has knowledge about the FTP proxy, and invokes it when it recognizes an address in the form *user@host*. The source code for the modified daemon is in the fwtk/tools/server/ftpd directory. The Makefile in this directory defines a `PROXY_PASSTHROUGH` variable. The sample Makefile in this directory follows:

```
.include "../../../Makefile.config"

SRCS=    ftpd.c ftpcmd.c glob.c logwtmp.c popen.c vers.c
OBJS=    ftpd.o ftpcmd.o glob.o logwtmp.o popen.o vers.o
LIBS=     ../../../libauth.a ../../../libfwall.


# SETPROCTITLE sets the process title to something readabl
# PARANOID disables many internal FTP operation
# NOEXPORT enables TIS internal export control routine
# PROXY_PASSTHROUGH enables toolkit aware proxy switchin


#CFLAGS= $(COPT) -I../../.. -DNOEXPORT -DSETPROCTITLE
#CFLAGS= $(COPT) -I../../.. -DNOEXPORT -DSETPROCTITLE
CFLAGS= $(COPT) -I../../.. -DPARANOID -DSETPROCTITLE -DPROXY_PASSTHRU=\"/bin/
ftp-gw\"
```

```
all: ftpd
ftpd:  $(OBJS
    $(CC) $(LDFL) -o $@ $(OBJS) $(LIBS) $(AUXLIB

install:  al
    $(CP) ftpd $(DEST

clean
    rm -f ftpd $(OBJS
```

The `PROXY_PASSTHROUGH` should be set to the path name of the proxy executable, before you run make on the Makefile.

NOTE

> If the modified FTP daemon is invoked after it has been chrooted to the FTP area, the PROXY_PASSTHROUGH must be relative to the FTP area, and not the root of the file system. A duplicate of the ftp-gw- and ftpd-specific rules from the netperm-table must also be installed in the ~anon-ftp/usr/local/etc/netperm-table file.

Authentication Mechanisms in the Toolkit

The TIS Firewall Toolkit has an optional authentication server, authsrv, that can be used to implement different types of authentication mechanisms in a mechanism-independent manner.

The authsrv program includes support for the following authentication mechanisms:

- Plaintext passwords
- Bellcore's S/Key
- Security Dynamics SecurID
- Enigma Logics Silver Card
- Digital Pathways SNK (Secure Net Key)

Plaintext passwords should not be used from untrusted networks. The authsrv should be run on a secure firewall. An administrative shell called the authmgr can be used from a remote location to manipulate the database information maintained by authsrv. The authmgr can optionally use encryption to protect its communication with the firewall.

When you compile the authsrv program, you must edit the authentication protocol bindings in the auth.h file for the authentication mechanisms specified in your network security policy.

The authsrv program supports authentication for all the interactive proxy servers used in the toolkit. Authentication can be done for both incoming and outgoing requests, and it consists of a simple challenge/response protocol that can be easily integrated in the software.

The authsrv maintains an internal user database that has a record for each user. This record contains the authentication mechanism to use for the user, user group, user full name, last successful authentication, and so on.

The users in the authsrv database can be organized in groups. Each group can have a separate administrator, who has administrative control of the group, including the ability to add and delete users to the group. Use of groups and a group administrator allows administrative responsibilities to be delegated in situations when a single firewall is shared by multiple departments in an organization.

The authsrv program is invoked for each authentication request. To configure authsrv, you must assign a free TCP port number and create entries in the /etc/services and /etc/inetd.conf files:

Use the following in the /etc/services file:

```
authsrv    7557/tcp    # Port 7557 was selected for authsrv
```

Use the following line in the /etc/inetd.conf file:

```
authsrv stream  tcp  nowait root /usr/local/etc/authsrv authsrv
```

It is not necessary to authenticate all services. You can authenticate selectively, based upon where the request originated and the type of requested operation. Consider the following example which shows the firewall gateway rules:

```
#
# FTP Firewall rules
#
ftp-gw:        authserver    127.0.0.1    7557
ftp-gw:        denial-msg    /usr/local/etc/ftp-deny.txt
ftp-gw:        welcome-msg    /usr/local/etc/ftp-welcome.txt
ftp-gw:        help-msg    /usr/local/etc/ftp-help.txt
ftp-gw:        permit-hosts    199.245.180.*    -log { retr stor } -auth
{stor}
ftp-gw:        permit-hosts    *    -authall
```

The first line identifies that the authsrv is running on the firewall (127.0.0.1 is the local host or loopback address).

The first permit-hosts statement identifies that hosts on the network 199.245.180.0 can access the FTP proxy. The file retrieval and storage options are logged. Of these, only file store operations are required to be authenticated. This is represented by the option:

```
-auth {stor}
```

The second permit-hosts statement says that all operations from external hosts will require authentication. This is represented by the following option:

```
-authall
```

To prevent intruders from probing the authentication server, the authsrv can be configured to specify which clients can connect to it. Typically, the authsrv and the proxy clients run on the same central firewall (bastion host). Because the clients and authsrv run on the same machine, it is only necessary to specify that the localhost (127.0.0.1) should be allowed to contact the authsrv. If administration is to be done from a remote workstation, the workstation's IP address should be listed. Following are some example rules for authsrv taken from the netperm-table configuration file:

```
#
# Example authsrv rules
#
authsrv:    database /var/adm/fwtk/authsrv.db
authsrv:    permit-hosts    127.0.0.1
authsrv:    permit-hosts    199.245.180.9 cipherkey
```

The database statement specifies the location of the authsrv database. This location should be secured from access by intruders. The clients on the host on which the authsrv runs are allowed access. The host at 199.245.180.9 is allowed access, presumably because this is a workstation used by the administrator. The keyword "cipherkey" indicates that communication between the workstation and authsrv is protected using DES encryption. The configuration file containing the cipher key should be protected from unauthorized access. To support encryption, two source code modules need to be replaced.

To import and export data from the authsrv database, two tools, authload and authdump, are included in the toolkit. The *authload* can be used to bulk-load the database from an ASCII text file. The *authdump* can be used to create an ASCII backup copy of the records in the database.

Summary

In this chapter firewall solutions for several commercial products were discussed. These included Internet Security Corporation's FireWall-1, ANS's InterLock, and TIS's Gauntlet. The chapter also discussed how a firewall could be built using freely available toolkits such as the Trusted Information Systems (TIS), and the TCP Wrapper program tcpd.

Chapter 10

THE TIS FIREWALL TOOLKIT

The Firewall Toolkit produced by Trusted Information Systems, also known as TIS, is not a single integrated package, but a set of tools that are used to build a firewall. For this reason, it is not for everyone who intends to construct and operate a firewall. Consequently, it is difficult to produce documentation that can be used in all situations.

Remember that a firewall is intended to be the security policy your organization has chosen to develop and support. In this chapter, you will examine how to compile the TIS Toolkit, and configure the various components that make up the kit. By the end of the chapter, you will know the techniques and issues concerned with the construction of a firewall using this Toolkit.

Understanding TIS

The TIS Firewall Toolkit is a collection of applications that, when properly assembled with a security policy, forms the basis of a firewall. This Toolkit is available as freeware to the Internet user community. As such, the Toolkit has gained a wide following and is in use worldwide.

The Toolkit is not a single integrated package like most commercial packages. Rather, it is a set of tools for building a number of different types of firewalls. Because of its inherent flexibility, a wide variety of combinations are possible regarding the installation and configuration of the TIS Toolkit. As such, this chapter explains what the Toolkit is and how the underlying technology works. With this knowledge in hand, and a copy of the Toolkit in another, you will be able to configure the Toolkit for your protection.

Where to Get TIS Toolkit

The TIS Toolkit is available from the site `ftp.tis.com`, in the directory /pub/firewalls/toolkit. The filename is fwtk.tar.Z.

After you retrieve the file, it must be uncompressed and extracted from the tar archive. While you're at the TIS anonymous FTP site, you may want to examine its collection of firewall documentation and information. After uncompressing and extracting the archive, the directory structure illustrated in figure 10.1 is created.

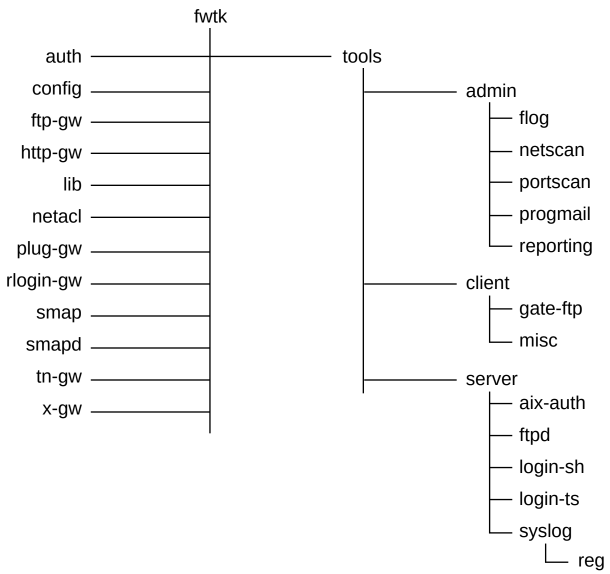


Figure 10.1

The TIS Toolkit directory structure.

When the files are extracted from the tar archive, the next task is to compile them. Before compiling, any site-specific changes should be made to firewall.h and the Makefile.config files. Major issues that you need to consider are the installation location of the Toolkit—defaults to /usr/lcoal/etc—and how the library and compiler are to be configured.

> **Note**
>
> Most users experience difficulties compiling the X-gw proxy. The reason for this is the program's dependencies on the X Window System Athena Widget set. If you do not have this widget set, you will experience problems in getting this application to compile.

Compiling under SunOS 4.1.3 and 4.1.4

There should be little difficulty in compiling the TIS Toolkit under the SunOS 4.1.3 and 4.1.4 operating systems. There are no changes required from the base configuration to achieve a successful compile. After the archive is extracted, a successful compile can be achieved even without modifying the Toolkit configuration.

Compiling under BSDI

No significant surprises occur when you compile the Toolkit under BSD/OS Version 2.0 from BSD, Inc. A few changes do need to be made to ensure the compile is successful, however. First, the Makefiles are not in the correct format for the make command. In TIS, the Makefiles use the syntax:

```
include Makefile.config
```

This syntax is not understood by the make command that is shipped with BSD/OS. To resolve the problem you can edit each of the Makefiles by hand, or use the program fixmake. The include statement also requires a small change. The required format looks like this:

```
.include        <Makefile.config>
```

If you edit the Makefiles by hand, this is what the change looks like. However, you can also use the fixmake command to correct the syntax of the Makefile by removing the include statement and including all of the required instructions in one Makefile.

While you are tweaking, it is a good idea to make the following additional changes. No other changes are necessary.

```
CC=     gcc
COPT=   -g -traditional -DBSDI
```

Code Changes

Several issues need to be considered when you compile the Toolkit components. These issues revolve primarily around the definition of sys_errlist. To resolve the problem, you must change the declaration of sys_errlist in all places where it is declared. For example, sys_errlist is defined in the code as:

```
extern  char    *sys_errlist[];
```

Commenting out the line using the C comment symbols (/* */) results in a successful compile of the source code:

```
/* extern       char    *sys_errlist[]; */
```

Installing the Toolkit

After the compile process is completed successfully, you must install the files in the appropriate place. The easiest way to install these files is to use the command:

```
make install
```

This command uses information in the Makefile to place the objects in the correct place. The process is shown in the following command sequence:

```
pc# make install
if [ ! -d /usr/local/etc ]; then  mkdir /usr/local/etc;  fi
for a in config lib auth smap smapd netacl plug-gw ftp-gw tn-gw rlogin-gw
➥http-g
w; do  ( cd $a; echo install: 'pwd'; make install );  done
install: /usr/tis/fwtk/config
if [ ! -f /usr/local/etc/netperm-table ]; then  cp netperm-table /usr/local
/etc;  chmod 644 /usr/local/etc/netperm-table;  fi
install: /usr/tis/fwtk/lib
install: /usr/tis/fwtk/auth
if [ -f /usr/local/etc/authsrv ]; then  mv /usr/local/etc/authsrv /u
sr/local/etc/authsrv.old;  fi
cp authsrv /usr/local/etc
chmod 755 /usr/local/etc/authsrv
if [ -f /usr/local/etc/authmgr ]; then  mv /usr/local/etc/authmgr /u
```

```
sr/local/etc/authmgr.old;  fi
cp authmgr /usr/local/etc
chmod 755 /usr/local/etc/authmgr
if [ -f /usr/local/etc/authload ]; then  mv /usr/local/etc/authload
/usr/local/etc/authload.old;  fi
cp authload /usr/local/etc
chmod 755 /usr/local/etc/authload
if [ -f /usr/local/etc/authdump ]; then  mv /usr/local/etc/authdump
/usr/local/etc/authdump.old;  fi
cp authdump /usr/local/etc
chmod 755 /usr/local/etc/authdump
install: /usr/tis/fwtk/smap
if [ -f /usr/local/etc/smap ]; then  mv /usr/local/etc/smap /usr/local/etc/
➥smap.old;  fi
cp smap /usr/local/etc
chmod 755 /usr/local/etc/smap
install: /usr/tis/fwtk/smapd
if [ -f /usr/local/etc/smapd ]; then  mv /usr/local/etc/smapd /usr/local/etc/
➥smapd.old;  fi
cp smapd /usr/local/etc
chmod 755 /usr/local/etc/smapd
install: /usr/tis/fwtk/netacl
if [ -f /usr/local/etc/netacl ]; then  mv /usr/local/etc/netacl /usr
/local/etc/netacl.old;  fi
cp netacl /usr/local/etc
chmod 755 /usr/local/etc/netacl
install: /usr/tis/fwtk/plug-gw
if [ -f /usr/local/etc/plug-gw ]; then  mv /usr/local/etc/plug-gw /u
sr/local/etc/plug-gw.old;  fi
cp plug-gw /usr/local/etc
chmod 755 /usr/local/etc/plug-gw
install: /usr/tis/fwtk/ftp-gw
if [ -f /usr/local/etc/ftp-gw ]; then  mv /usr/local/etc/ftp-gw /usr
/local/etc/ftp-gw.old;  fi
cp ftp-gw /usr/local/etc
chmod 755 /usr/local/etc/ftp-gw
install: /usr/tis/fwtk/tn-gw
if [ -f /usr/local/etc/tn-gw ]; then  mv /usr/local/etc/tn-gw /usr/local/etc/
➥tn-gw.old;  fi
cp tn-gw /usr/local/etc
chmod 755 /usr/local/etc/tn-gw
install: /usr/tis/fwtk/rlogin-gw
if [ -f /usr/local/etc/rlogin-gw ]; then  mv /usr/local/etc/rlogin-g
w /usr/local/etc/rlogin-gw.old;  fi
cp rlogin-gw /usr/local/etc
chmod 755 /usr/local/etc/rlogin-gw
install: /usr/tis/fwtk/http-gw
if [ -f /usr/local/etc/http-gw ]; then  mv /usr/local/etc/http-gw /usr/local/
➥etc
```

```
/http-gw.old;  fi
cp http-gw /usr/local/etc
chmod 755 /usr/local/etc/http-gw
```

With the Toolkit successfully installed and compiled, the next step is the security policy and the configuration of the Toolkit.

Preparing for Configuration

When configuring the Toolkit, the first step is to turn off all unnecessary services that are running on the system that will affect your firewall. This requires that you have some level of Unix knowledge regarding the system startup procedure and services for your system. For example, you may have to:

- Edit the /etc/inetd.conf file
- Edit the system startup scripts such as /etc/rc /etc/rc2.d/* and others
- Edit the operating system configuration to disable unnecessary kernel-based services

You can use the ps command to see that a number of services are in operation. The following output shows such services on a sample system:

```
pc# ps -aux
USER       PID %CPU %MEM   VSZ  RSS  TT  STAT  STARTED  TIME    COMMAND
root       442  0.0  1.7   144  240  p0  R+    3:34AM   0:00.04 ps -aux
root         1  0.0  1.7   124  244  ??  Is    3:02AM   0:00.08 /sbin/init --
root         2  0.0  0.1     0   12  ??  DL    3:02AM   0:00.01 (pagedaemon)
root        15  0.0  6.0   816  888  ??  Is    3:03AM   0:00.47 mfs -o rw -s 1
root        36  0.0  1.5   124  220  ??  Ss    3:03AM   0:00.21 syslogd
root        40  0.0  1.2   116  176  ??  Ss    3:03AM   0:00.06 routed -q
root        77  0.0  0.5    72   72  ??  Ss    3:03AM   0:00.34 update
root        79  0.0  1.6   284  232  ??  Is    3:03AM   0:00.08 cron
root        85  0.0  0.3    72   36  ??  I     3:03AM   0:00.01 nfsiod 4
root        86  0.0  0.3    72   36  ??  I     3:03AM   0:00.01 nfsiod 4
root        87  0.0  0.3    72   36  ??  I     3:03AM   0:00.01 nfsiod 4
root        88  0.0  0.3    72   36  ??  I     3:03AM   0:00.01 nfsiod 4
root        91  0.0  1.0    96  144  ??  Is    3:03AM   0:00.07 rwhod
root        93  0.0  1.3   112  180  co- I     3:03AM   0:00.05 rstatd
root        95  0.0  1.3   128  192  ??  Is    3:03AM   0:00.07 lpd
root        97  0.0  1.3   104  184  ??  Ss    3:03AM   0:00.13 portmap
root       102  0.0  1.6   332  224  ??  Is    3:03AM   0:00.05 (sendmail)
root       108  0.0  1.4   144  200  ??  Is    3:03AM   0:00.11 inetd
root       117  0.0  2.1   228  300  co  Is+   3:03AM   0:00.90 -csh (csh)
root       425  0.0  2.0   156  292  ??  S     3:33AM   0:00.15 telnetd
chrish     426  0.0  2.1   280  304  p0  Ss    3:33AM   0:00.26 -ksh (ksh)
root       440  0.4  1.9   220  280  p0  S     3:34AM   0:00.17 -su (csh)
root         0  0.0  0.1     0    0  ??  DLs   3:02AM   0:00.01 (swapper)
pc#
```

By editing the /etc/inetd.conf file so that it resembles the following output, you can reduce the number of active processes. This reduces the load on the system and, more importantly, does not accept TCP connections on unnecessary ports.

```
#
# Internet server configuration database
#
#    BSDI    $Id: inetd.conf,v 2.1 1995/02/03 05:54:01 polk Exp $
#    @(#)inetd.conf  8.2 (Berkeley) 3/18/94
#
# ftp     stream    tcp    nowait    root    /usr/libexec/tcpd    ftpd -l -A
# telnet  stream    tcp    nowait    root    /usr/libexec/tcpd    telnetd
# shell   stream    tcp    nowait    root    /usr/libexec/tcpd    rshd
# login   stream    tcp    nowait    root    /usr/libexec/tcpd    rlogind -a
# exec    stream    tcp    nowait    root    /usr/libexec/tcpd    rexecd
# uucpd   stream    tcp    nowait    root    /usr/libexec/tcpd    uucpd
# finger  stream    tcp    nowait    nobody  /usr/libexec/tcpd    fingerd
# tftp    dgram     udp    wait      nobody  /usr/libexec/tcpd    tftpd
# comsat  dgram     udp    wait      root    /usr/libexec/tcpd    comsat
# ntalk   dgram     udp    wait      root    /usr/libexec/tcpd    ntalkd
# pop     stream    tcp    nowait    root    /usr/libexec/tcpd    popper
# ident   stream    tcp    nowait    sys     /usr/libexec/identd  identd -l
# #bootp  dgram     udp    wait      root    /usr/libexec/tcpd    bootpd -t 1
# echo    stream    tcp    nowait    root    internal
# discard stream    tcp    nowait    root    internal
# chargen stream    tcp    nowait    root    internal
# daytime stream    tcp    nowait    root    internal
# tcpmux  stream    tcp    nowait    root    internal
# time    stream    tcp    nowait    root    internal
# echo    dgram     udp    wait      root    internal
# discard dgram     udp    wait      root    internal
# chargen dgram     udp    wait      root    internal
# daytime dgram     udp    wait      root    internal
# time    dgram     udp    wait      root    internal
# Kerberos authenticated services
#klogin   stream    tcp    nowait    root    /usr/libexec/rlogind    rlogind
➥-k
#eklogin  stream    tcp    nowait    root    /usr/libexec/rlogind    rlogind
➥-k -x
#kshell   stream    tcp    nowait    root    /usr/libexec/rshd       rshd -k
# Services run ONLY on the Kerberos server
#krbupdate stream tcp      nowait    root    /usr/libexec/registerd
➥registerd
#kpasswd  stream    tcp    nowait    root    /usr/libexec/kpasswdd
➥kpasswdd
```

The reason for turning off all these services is to reduce the likelihood that your system will be compromised while the firewall is being installed and configured. You should also use the console to perform the initial setup and configuration of the firewall. With the /.etc/inetd.conf file updated, inetd must be signaled to know that some changes have been made. This signal is generated using the command:

```
kill -1 inetd.pid
```

The process identifier (PID) can be procured, and inetd restarted by using this command sequence:

```
pc# ps -aux ¦ grep inetd
root       108  0.0  1.4   144  200  ??  Is    3:03AM    0:00.11 inetd
pc# kill -1 108
```

To ensure that the services are turned off, you can attempt to connect to a service offered by inetd:

```
pc# telnet pc ftp
Trying 204.191.3.150...
telnet: Unable to connect to remote host: Connection refused
pc#
```

Now that the inetd services are disabled, disable other services that are part of the system startup files and the kernel. Some of these services are system-specific, which might require some exploration. Nevertheless, try to find the following services and turn them off:

gated, cgd	pcnfsd	rwhod
mountd	portmap	sendmail
named	printer	timed
nfsd	rstatd	xntpd
nfsiod		

TIP

While timed, which is when the NTP time server process is turned off, you should configure your firewall to get time updates via an NTP server. This allows your firewall clock to have accurate time, which may prove invaluable should you take legal action.

After turning off these daemons, the process table on the sample system now looks like this:

```
pc.unilabs.org$ ps -aux
USER       PID %CPU %MEM   VSZ  RSS  TT  STAT  STARTED   TIME    COMMAND
chrish      89  2.3  2.1   280  304  p0  Ss    4:24AM    0:00.25 -ksh (ksh)
root         1  0.0  1.7   124  244  ??  Is    4:18AM    0:00.07 /sbin/init --
root         2  0.0  0.1     0   12  ??  DL    4:18AM    0:00.01 (pagedaemon)
root        15  0.0  3.2   816  464  ??  Is    4:19AM    0:00.08 mfs -o rw -s 1
root        36  0.0  1.5   124  220  ??  Ss    4:19AM    0:00.17 syslogd
root        71  0.0  0.5    72   72  ??  Ss    4:19AM    0:00.05 update
root        73  0.0  1.8   284  256  ??  Is    4:19AM    0:00.05 cron
root        75  0.0  1.3   140  192  ??  Ss    4:19AM    0:00.04 inetd
root        84  0.0  2.0   220  292  co  Is+   4:19AM    0:00.26 -csh (csh)
```

```
root        88  0.1  2.0   156  292  ??  S     4:24AM    0:00.13 telnetd
root         0  0.0  0.1     0    0  ??  DLs   4:18AM    0:00.00 (swapper)
chrish      95  0.0  1.6   136  232  p0  R+    4:24AM    0:00.02 ps -aux
pc.unilabs.org$
```

The ps command output shown now represents a quiet system. For clarification, the mfs command in the ps output is for a memory-based temporary file system on the BSDI Version 2.0 Operating System. However, this does not really list the actual services that are provided on this system. In the sample inetd.cof file presented earlier, virtually all the available network services were disabled. This is illustrated in the output of the netstat command:

```
pc# netstat -a
Active Internet connections (including servers)
Proto Recv-Q Send-Q  Local Address          Foreign Address        (state)
tcp        0      0  pc.telnet              stargazer.1037         ESTABLISHED
tcp        0      0  *.telnet               *.*                    LISTEN
udp        0      0  *.syslog               *.*
Active Unix domain sockets
Address  Type   Recv-Q Send-Q    Inode     Conn     Refs  Nextref Addr
f0764400 dgram       0      0        0 f0665c94        0 f0665214
f074e480 dgram       0      0        0 f0665c94        0        0
f0665c00 dgram       0      0 f0665780        0 f06d6194        0 /dev/log
pc#
```

The tools directory in the Toolkit distribution includes a utility called portscan, which probes a system to determine what TCP services are currently being offered. This program probes the ports on a system and prints a list of available port numbers, or service names. The output of the command is shown here:

```
pc# ./portscan pc
7
9
13
19
21
23
25
...
512
513
shell
1053
1054
1055
1056
1057
pc#
```

This command shows what ports were available prior to reducing the available services. After reducing those services by shutting off the entries in inetd.conf and the startup files, the system now offers the following ports:

```
pc# ./portscan pc
21
23
pc#
```

With the host almost completely shut down from the network, the next step is to configure TIS Toolkit components.

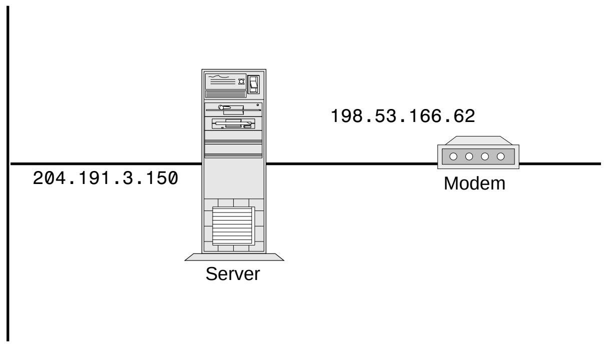
Configuring TCP/IP

For TIS to be effective as a firewall, the system on which it is running must not perform routing. A system that has two or more network interfaces must be configured so that it does not automatically route packets from one interface to another. If this occurs, services that are being constructed with the TIS Toolkit will not be used.

IP Forwarding

To receive any real benefits from a firewall installation, you need to make sure IP forwarding has been disabled. IP forwarding causes the packets received on one interface to be retransmitted on all other applicable interfaces. To help illustrate IP forwarding, suppose you are considering setting up a firewall on the system in figure 10.2.

Figure 10.2
Multihomed machines.



This machine has two interfaces: one is for the local area network, which has an IP address of 204.191.3.150. The other interface is for the wide area network, and is a PPP link using an IP address of 198.53.166.62. When IP forwarding is enabled, any packets received on the LAN interface of this machine that are destined for a different network are automatically forwarded to the PPP link. The same is true for packets on the PPP link. If the packets received on the PPP link are for the rnet, they will be transmitted on the Ethernet interface in the machine.

This type of arrangement is unsuitable for a firewall. The reason is that the firewall will still pass unlogged and unauthenticated traffic from either direction. Consequently, there is little or no point in going through this exercise if you leave IP forwarding enabled.

Disabling IP forwarding usually requires that a new kernel be configured. The reason for this is that the process of IP disabling involves changing some kernel parameters. Table 10.1 lists parameters that must be changed for the identified operating systems.

Table 10.1
Disabling IP Forwarding

Operating System	*Parameter*
BSDI Version 2.0	Make sure GATEWAY is commented out in the kernel configuration files.
SunOS 4.1.x	Run adb on the kernel to set IP_forwarding to -1, and save the modified kernel image. Alternatively, modify /usr/kvm/sys/netinet/in_proto.c) to set the variable to -1 by default and rebuild the kernel.

After making the required changes to the kernel parameters, you need to build a new kernel, install it, and reboot. This removes any configured IP forwarding, and enables you to maximize the capabilities of the Toolkit. After IP forwarding is removed, all traffic requests either into or out of the private network need to be made through the proxy servers on the firewall.

The netperm Table

The netperm table, found in /usr/local/etc/netperm-table, is the master configuration file for all the components in the Trusted Firewall Toolkit (netacl, smap, smapd, ftp-gw, tn-gw, and plug-gw). When an application in the Toolkit starts, it reads its configuration and permissions information from netperm-table and stores it in an in-memory database. Saving the information in an in-memory database allows the information to be preserved, even after a chroot system call is used to reset the directory structure.

The permissions/configuration file is organized into rules. Each rule is the name of the application that rule applies to, followed by a colon. Multiple applications can be targeted by a single rule by separating the names with commas, or wildcarding them with an asterisk. When an application extracts its configuration information, it only extracts the rules that apply to it, preserving the order in which they appeared in the file. The following sequence lists a sample set of rules for the smap and smapd application.

```
# sample rules for smap
       smap, smapd:  userid 4
       smap, smapd:  directory /mail/inspool
       smap:         timeout 3600
```

Note

> Comments regarding the rules can be inserted in the configuration file by starting the line with "#" as the first character. As with any configuration file or program, the more comments that are used, the easier it is later to maintain the rules.

When an application has matched a rule, the rule is translated into whitespace-delimited strings for later use. Typically, the application retrieves matching rules based on the first word in the rule; the remaining words serve as parameters for that particular clause. For the smap client and smapd server in the preceding example, the rules specify the userid to use when the application executes, the directory clause identifies the location of files, and the timeout clause indicates how long the server or client will wait before assuming that the remote end is "hung."

Special modifiers are available for each clause. For example, if the clause begins with a permit- or deny- modifier, the rule is internally flagged as granting or revoking permission for that clause. This means that if an application retrieves all of its configuration clauses for "hosts," the following will be returned:

```
netacl-in.ftpd: permit-hosts 192.33.112.117 -exec /usr/etc/in.ftpd
netacl-in.ftpd: permit-hosts 198.137.240.101 -exec /usr/etc/in.ftpd
netacl-in.ftpd: deny-hosts unknown
netacl-in.ftpd: deny-hosts *
```

Although this example may not seem clear, keep in mind that each application within the Toolkit has its own unique set of clauses. The default configuration for each of the application's clauses and examples are presented with the application's description.

When assembling your netperm-table file, you might want to consider a few conventions. These conventions promote consistency in the file, and help produce a more readable and maintainable rules list. When a hostname or host IP address is specified in the rule, matching is performed based on whether the pattern to which the address will be matched is all digits and decimal points, or other characters.

To better explain this process, consider this configuration rule:

```
netacl-in.ftpd: permit-hosts 192.33.112.117 -exec /usr/etc/in.ftpd
```

When a connection is received and this rule is applied, the IP address of the remote machine will be used to match this rule. If the pattern to match consists entirely of digits and decimals, matching is performed against the IP address; otherwise, it is performed against the hostname.

If the rule specifies a host or domain name, as in the following rule,

```
netacl-in.ftpd: permit-hosts *.istar.net -exec /usr/etc/in.ftpd
```

then the remote system's name is used to validate against the rule, not the IP address. To prevent any vulnerability from DNS spoofing, it is highly recommended that the configuration rules be bound to IP addresses. When matching, asterisk wildcards are supported, with syntax similar to the shell's, matching as many characters as possible.

When the application attempts to resolve an IP address to the domain name and the reverse lookup fails, the hostname is set to "unknown." Otherwise the real hostname of the remote system is returned. When the Domain Name resolution is performed by the firewall, a check is made to ensure that the IP address for the DNS name returned by the reverse lookup is the same.

This setup prevents DNS spoofing. If a hostname for this IP address cannot be located in the DNS system, the hostname is set to "unknown" and a warning is logged. This permits rules to operate on hosts that didn't have valid DNS mappings. This means that it is possible to allow any host in the Internet to pass through your firewall, or access certain services (or both) as long as reverse DNS, or IN-ADDR.ARPA addressing is properly configured.

CONFIGURING NETACL

netacl is a network access control program; it provides a degree of access control for various TCP-based services available on the server. For example, you may want to have Telnet access to the firewall for authorized users. The netacl program and the appropriate rules enable you to create this setup. The same capabilities are possible for any of the available services, including FTP and rlogin.

The netacl program is started through inetd; after inetd performs some checks, netacl allows or denies the request for service from the remote user/system. When configuring the inetd.conf file for netacl, it is important to know that netacl accepts only one argument: the name of the service to be started. Any other arguments that are intended for the service do not go in the inetd.conf file. Consider this example:

```
ftp     stream  tcp     nowait  root    /usr/local/etc/netacl   ftpd
```

In this situation, when a connection request is accepted by inetd for an FTP service, the netacl program is started with an argument of ftpd. Before the ftpd daemon is started, the request is validated using the rules found in the netperm-table. The rule name for netacl consists of the keyword netacl- followed by the name of the service. For example, if the named service is ftpd, the rule name consists of netacl-ftpd, as in the following:

```
netacl-ftpd: permit-hosts 204.191.3.147 -exec /usr/libexec/ftpd -A -l
```

When you examine these two lines—the first from inetd.conf and the second from netperm-table—you can see that the command-line arguments and other information required for the daemon are found in netperm-table.

As with all the TIS Toolkit components, arguments and descriptive keywords are permitted in the authentication clause. As seen in the preceding command output, only the host 204.191.3.147 is permitted access on the firewall to run the ftpd command. It does, however, mean that FTP requests can be sent through the firewall. Table 10.2 lists various keywords that are understood by the netacl program.

Table 10.2
The netacl Rules and Clauses

Service	*Keyword*	*Description*
netacl	permit-hosts IP Address or hostname	Specifies a permission rule to allow the named hosts. This is a list of IP addresses or hostnames.
	deny-hosts IP Address or hostname	Specifies a permission rule to deny the named hosts. This is a list of IP addresses or hostnames. The denial of service is logged via syslogd.
	-exec executable [args]	Specifies a program to invoke to handle the service. This option must be the final option in the rule. An -exec option must be present in every rule.

Service	Keyword	Description
	-user userid	userid is the numeric UID or the name from a login in /etc/passwd that the program should use when it is started.
	-chroot rootdir	Specifies a directory to which netacl should chroot(2) prior to invoking the service program. This requires that the service program be present, and the pathname for the executable be relative to the new root.

Acceptance or rejection of the service is logged by the syslog facility. The messages printed in the syslog files resemble those shown here:

```
Oct  4 00:56:12 pc netacl[339]: deny host=stargazer.unilabs.org/204.191.3.147
service=ftpd
Oct  4 01:00:20 pc netacl[354]: permit host=stargazer.unilabs.org/
➥204.191.3.147
service=ftpd execute=/usr/libexec/ftpd
```

The first line in the log report indicates that the host stargazer.unilabs.org was denied access to the FTP service through the netacl program. The second line of output indicates that the FTP request was accepted and allowed. Notice that the logging information only specifies the service that was originated, and from where it originated. It does not show who the user connected to. The sample netacl rules that follow illustrate the use of some of the parameters and clauses for netacl.

```
netacl-in.telnetd: permit-hosts 198.53.64.*-exec /usr/etc/in.telnetd
netacl-in.ftpd: permit-hosts unknown -exec /bin/cat /usr/local/etc/noftp.txt
netacl-in.ftpd: permit-hosts 204.191.3.* -exec /usr/etc/in.ftpd
netacl-in.ftpd: permit-hosts * -chroot /home/ftp -exec /bin/ftpd -f
```

In this example, netacl is configured to permit Telnet only for hosts in a particular subnet. netacl is configured to accept all FTP connections from systems that do not have a valid DNS name ("unknown") and to invoke cat to display a file when a connection is made. This provides an easy and flexible means of politely informing someone that they are not permitted to use a service. Hosts in the specified subnet are connected to the real FTP server in /usr/etc/in.ftpd but all connections from other networks are connected to a version of the FTP server that is already chrooted to the FTP area, effectively making all FTP activity "captive."

Connecting with netacl

When netacl is configured for the service that you want to provide, you should test it to ensure that it is working. Testing requires verifying rules configured for that service to ensure that they are in fact operating as they should. Consider the following rules:

```
netacl-ftpd: permit-hosts 204.191.3.147 -exec /usr/libexec/ftpd -A -l
```

This rule says that FTP connections will be accepted only from the host 204.191.3.147. When this connection is received, the ftpd server with the appropriate arguments will be started. This can be evaluated by connecting to the FTP server from the authorized host, as illustrated here:

```
C:\ >ftp pc
Connected to pc.unilabs.org.
220 pc.unilabs.org FTP server (Version wu-2.4(1) Fri Feb 3 11:30:22 MST 1995)
➥ready.
User (pc.unilabs.org:(none)): chrish
331 Password required for chrish.
Password:
230 User chrish logged in.
ftp>
```

As you can see from this output, the connection from the authorized machine to the target system did in fact work. This could further be validated by examining the syslog records for the target system where any transfers may in fact be logged. The availability of this feature depends on the implementation of the ftpd that is in use at your site.

Another security breach you want to avoid is granting a non-authorized system a connection. To illustrate, consider the exchange:

```
pc# ftp pc
Connected to pc.unilabs.org.
421 Service not available, remote server has closed connection
ftp>
```

The connection is initially established, but after netacl has performed verification of the rules, it finds that the host is not permitted access, and the connection is closed. On the target system, a deny informational message is written to the syslog and to the console:

```
Oct  4 02:53:12 pc netacl[1775]: deny host=pc.unilabs.org/204.191.3.150
➥service=ftpd
```

In this case, the remote system received no information other than the connection has been closed. Meanwhile, the system administrator knows that the remote has been attempting to gain access. If this occurs enough, some other action may be required against the remote user.

Such a blunt response to an unauthorized attempt to gain access might not be the most appreciated. For this reason, you might be wise to consider a rule like the one that follows:

```
netacl-ftpd: permit-hosts 204.191.3.147 -exec /bin/cat /usr/local/etc/
➥noftp.txt
```

In this case, a user who attempts to connect from the site 204.191.3.147 will not be refused a connection; he or she will just not get what they want. With this configuration, you can log the connection, and tell the user that he or she is not permitted access to the requested service. For example, when you attempt to connect to your server, the /usr/local/etc/noftp.txt file displays this response:

```
C:\ >ftp pc
Connected to pc.unilabs.org.

**** ATTENTION ****

Your attempt to use this server's FTP facility is not permitted due to
organizational security policies.  Your connection attempt has been logged
and recorded.

Use of the FTP Services on this machine is restricted to specific sites.

If you believe that you are an authorized site, please contact Jon Smith
at 555-1212 ext 502, or e-mail to ftpadmin@org.com.

Connection closed by remote host.

C:\ >
```

Any type of message can be displayed here instead of allowing access to the requested service. This "denial" can be for system administration purposes, for example, or because of maintenance.

RESTARTING INETD

Remember that after each reconfiguration of the inetd.conf file, inetd must be restarted. To do this, you must find the Process ID or PID number for inetd and send a SIGHUP to it. The following commands are used in this process:

```
Signalling inetd

pc# ps -aux ¦ grep inetd
root      1898  0.0  0.2   120   28  p3  R+   10:46AM    0:00.02 grep inetd
root        75  0.0  1.5   140  220  ??  Is   11:19AM    0:00.25 inetd
pc# kill -1 75
pc#
```

When inetd has been signaled with the -1, or SIGHUP, it rereads the /etc/inetd.conf file and applies the new configuration immediately.

NOTE

> You might have to send a second SIGHUP signal to inetd to make the changes permanent. Specific systems are IRIX and some versions of SunOS.

This is the most common problem that system administrators have when changing the configuration file. They make the change, but forget to restart inetd.

Configuring the Telnet Proxy

The Telnet proxy, tn-gw, provides passthrough Telnet services. In many circumstances, a system administrator may not want to allow Telnet access through the firewall and either into or out of the private network. The Telnet proxy does not provide the same type of access to the firewall host as the netacl program. The intent behind using Telnet with netacl is to allow access to the firewall host. With the proxy, the intent is to provide passthrough Telnet with logging control.

Because of the dilemma of allowing remote administrative access and establishing a proxy Telnet, it is common for the firewall administrator to run the real telnetd on a TCP port other than the default, and to place the proxy on the standard TCP port. This is accomplished by editing the /etc/services file and changing it to be something similar to the following:

```
telnet          23/tcp
telnet-a        2023/tcp
```

These changes are only effective after /etc/inetd.conf has been changed to reflect the configuration shown here:

```
telnet          stream  tcp     nowait  root    /usr/local/etc/tn-gw    tn-gw
telnet-a        stream  tcp     nowait  root    /usr/local/etc/netacl   telnetd
```

When an incoming connection is received on the Telnet port with this configuration, the tn-gw application is started. When tn-gw receives a request, it first verifies that the requesting host is permitted to connect to the proxy. Access to the proxy is determined by the rules established in the netperm-table. These rules resemble those seen previously for the netacl application. However, there are application-specific parameters. The rule clauses for tn-gw are listed in table 10.3.

Table 10.3
tn-gw Rules and Clauses

Option	*Description*
userid user	Specifies a numeric userid or the name of a password file entry. If this value is specified, tn-gw will set its userid before providing service.
directory pathname	Specifies a directory to which tn-gw will chroot(2) prior to providing service.
prompt string	Specifies a prompt for tn-gw to use while it is in command mode.
denial-msg filename	Specifies the name of a file to display to the remote user if he or she is denied permission to use the proxy. If this option is not set, a default message is generated.
timeout seconds	Specifies the number of seconds of idleness after which the proxy should disconnect. Default is no timeout.
welcome-msg filename	Specifies the name of a file to display as a welcome banner upon successful connection. If this option is not set, a default message is generated.
help-msg filename	Specifies the name of a file to display if the "help" command is issued. If this option is not set, a list of the internal commands is printed.
denydest-msg filename	Specifies the name of a file to display if a user attempts to connect to a remote server for which he or she is not authorized. If this option is not set, a default message is generated.
authserver hostname [portnumber [cipherkey]]	Specifies the name or address of a system to use for network authentication. If tn-gw is built with a compiled-in value for the server and port, these values will be used as defaults but can be overridden if specified in the authserver rule. If support for DES-encryption of traffic is present in the server, an optional cipherkey can be provided to secure communications with the server.

continues

Table 10.3, Continued
tn-gw Rules and Clauses

Option	*Description*
hosts host-pattern [host-pattern2...] [options]	Specifies host and access permissions.

The initial configuration for the tn-gw application is shown here.

```
tn-gw:          denial-msg      /usr/local/etc/tn-deny.txt
tn-gw:          welcome-msg     /usr/local/etc/tn-welcome.txt
tn-gw:          help-msg        /usr/local/etc/tn-help.txt
tn-gw:          timeout 3600
tn-gw:          permit-hosts 204.191.3.* -dest *.fonorola.net -dest !* -
➥passok -xok
```

NOTE

> If any of the files identified in the denial-msg, welcome-msg, help-msg, or denydest-msg clauses are missing, the connection will be dropped as soon as a request is made for that file.

This configuration informs users when they are or are not allowed to connect to the proxy server, and when connections are denied due to their destination. The timeout line indicates how long the Telnet connection can be idle before the firewall will terminate it. The last line establishes an access rule to the tn-gw application. This rule and the optional parameters are discussed shortly. A sample connection showing the host denial message is shown as follows:

```
$ telnet pc
Connecting to pc ...

**** ATTENTION ****

Your attempt to use this server's telnet proxy is not permitted due to
organizational security policies.  Your connection attempt has been logged
and recorded.

Use of the telnet proxy Service on this machine is restricted to specific
sites.

If you believe that you are an authorized site, please contact Jon Smith
at 555-1212 ext 502, or e-mail to ftpadmin@org.com.

Connection closed by foreign host
$
```

If the host is permitted to converse with the tn-gw application, tn-gw enters a command loop where it accepts commands to connect to remote hosts. The commands available within the tn-gw shell are listed in table 10.4.

Table 10.4
tn-gw Commands

Command	*Description*
c[onnect] hostname [port] telnet hostname [port] open	Connects to a remote host. Access to the remote host may be denied based on a host destination rule.
x[-gw] [display/hostname]	This command invokes the X Windows gateway for a connection to the user's display. By default, the display name is the connecting machine followed by :0.0, as in pc.myorg.com:0.0. The x-gw command is discussed later in this chapter.
help ?	Displays a user-definable help file.
quit exit close	Exits the gateway.

Connecting through the Telnet Proxy

When a permitted host connects to the proxy, it is greeted by the contents of the welcome file—configured in the tn-gw options—and by a prompt. At the prompt, tn-gw expects to receive one of the commands listed in table 10.4. When the connect request is made, the access rules are applied to the destination host to confirm that a connection to that host is permitted. If the connection is permitted, the connection is made. A successful connection is shown as follows:

```
Welcome to the URG Firewall Telnet Proxy

Supported commands are
        c[onnect] hostname [port]
        x-gw
        help
        exit
```

```
To report problems, please contact Network Security Services at 555-1212 or
by e-mail at security@org.com

Enter Command>c sco.sco.com
Not permitted to connect to sco.sco.com
Enter Command>c nds.fonorola.net
Trying 204.191.124.252 port 23...


SunOS Unix (nds.fonorola.net)

login:
```

In this output you can see that a Telnet connection is established to the firewall, from which the tn-gw application is started. The user first attempts to contact sco.sco.com, which is denied. A second connection request to nds.fonorola.net is then permitted. This sequence begs the question, "What's the difference?" The answer is that host destination rules are in force. This means that a given system may be blocked through options on the host command in the tn-gw rules.

Host Access Rules

The host rules that permit and deny access to the Telnet proxy can be modified by a number of additional options, or rules that have other host access permissions. As seen in table 10.3, the host rules are stated:

```
tn-gw:    deny-hosts unknown
tn-gw:    hosts 192.33.112.* 192.94.214.*
```

These statements indicate that hosts that cannot be found in the DNS in-addr.arpa domain are unknown, and therefore denied, or that hosts connecting from the network 192.33.112 and 192.94.214 are allowed to connect to the proxy. Optional parameters, which begin with a hyphen, further restrict the hosts that can connect to the proxy, or where the remote host can connect to behind the firewall.

Earlier output showed that the connect request to sco.scolcom was denied by the proxy because the user was not permitted to connect to that host. This was configured by using the rule:

```
tn-gw:          permit-hosts 204.191.3.* -dest *.fonorola.net -dest !* -passok
➥-xok
```

This rule states that any host from the 204.191.3 network is allowed to contact any machine in the fonorola.net domain, but no others. This example illustrates the -dest option, which restricts which hosts can be connected. The -dest parameter, described in table 10.5 with the other optional parameters, is used to specify a list of valid destinations. If no list is specified, then the user is not restricted to connecting to any host.

Table 10.5
Host Access Rules

Rule	*Description*
-dest pattern -dest { pattern1 pattern2 ... }	Specifies a list of valid destinations. If no list is specified, all destinations are considered valid. The -dest list is processed in order as it appears on the options line. -dest entries preceded with a "!" character are treated as negation entries.
-auth	Specifies that the proxy should require a user to authenticate with a valid userid prior to being permitted to use the gateway.
-passok	Specifies that the proxy should permit users to change their passwords if they are connected from the designated host. Only hosts on a trusted network should be permitted to change passwords, unless token-type authenticators are distributed to all users.

The -dest options are applied in the order that they appear in the line. Consequently, in the example used so far in this chapter, if the machine you are connecting to is sco.sco.com, then the first option describing a machine in the fonorola.net domain is not matched. This means that the second destination specification is matched, which is a denial. The "!" is a negation operator, and indicates that this is not permitted. The end result is that users on the 204.191.3 network can only connect to systems in the fonorola.net domain, and no others.

The use of an IP address instead of a domain name does not alter the rule. Before the connection is permitted, the tn-gw application attempts to validate the IP address. If the returned host matches one of the rules, then the rule is applied. Otherwise, the connection is dropped.

Verifying the Telnet Proxy

The operation of the proxy rules can be determined by attempting a connection through each of the rules, and verifying whether the correct files are displayed when information is requested. For example, if a user connects to tn-gw and enters the help command, does the user get the requested information? Are the restricted sites in fact restricted?

This verification is accomplished by exercising each of the rules. For example, consider the following rule:

```
tn-gw:          permit-hosts 204.191.3.* -dest *.fonorola.net -dest !*
```

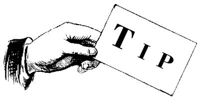
The operation of this rule can be easily verified, once it is clear what is being controlled. This rule says: "Permit any host in the 204.191.3 network to connect to any machine in the fonorola.net domain. All connections to machines outside that domain are denied."

This can be easily verified by using Telnet to contact tn-gw and attempting to connect to a site within the fonorola.net domain space, and then attempting to connect to any other site. If the fonorla.net site is accessible, but no other site is, then it is safe to say that the Telnet is working as it should.

For example, consider the following rules:

```
tn-gw:          permit-hosts 204.191.3.* -dest *.fonorola.net -dest !* -passok
➥-xok
tn-gw:          deny-hosts * -dest 204.191.3.150
```

If the connecting host is from the 204.191.3 network, access is granted to the proxy, but the user can only connect to the sites in the fonorola.net domain. The second line says that any host attempting to access 204.191.3.150 will be denied. Should the second line be first in the file, access to the proxy server itself would not be permitted.



When entering the rules in the netperm-table, remember to write them from least to most specific. Or, write them in order of use, after conducting some traffic analysis to determine where the traffic is going. This can be difficult and time-consuming.

This type of configuration is advantageous because it ensures that the firewall cannot be accessed through the proxy, and leaves the Telnet server available through the netacl program, which has been configured to listen on a different port.

Even though the firewall host is not available through the proxy, it can still be accessed through the netacl program and the Telnet server running on the alternate port.

Configuring the rlogin Gateway

The rlogin proxy provides a service similar to the Telnet proxy with the exception of access being provided through the rlogin service rather than Telnet. Typically, access to the firewall using rlogin would not be allowed because of the large number of problems that can occur. Consequently, the only access to the firewall host is through Telnet.

Regardless, there are requirements that justify the need for an rlogin proxy service. For example, the rlogin service provides rules for additional authentication that allow the connection to be granted without the user logging in like Telnet. The process of configuring the rlogin-gw rules is similar to the tn-gw application; they both support the same options. The rules that are available for the rlogin-gw service are listed and explained in table 10.6.

Table 10.6
rlogin-gw Rules and Clauses

Option	*Description*
userid user	Specifies a numeric userid or the name of a password file entry. If this value is specified, tn-gw will set its userid before providing service.
directory pathname	Specifies a directory to which tn-gw will chroot(2) prior to providing service.
prompt *string*	Specifies a prompt for tn-gw to use while it is in command mode.
denial-msg *filename*	Specifies the name of a file to display to the remote user if he or she is denied permission to use the proxy. If this option is not set, a default message is generated.
timeout seconds	Specifies the number of seconds the system remains idle before the proxy disconnects. Default is no timeout.
welcome-msg *filename*	Specifies the name of a file to display as a welcome banner after the system successfully connects. If this option is not set, a default message is generated.
help-msg *filename*	Specifies the name of a file to display if the "help" command is issued. If this option is not set, a list of the internal commands is printed.

continues

Table 10.6, Continued
rlogin-gw Rules and Clauses

Option	*Description*
denydest-msg *filename*	Specifies the name of a file to display if a user attempts to connect to a remote server from which he or she is restricted. If this option is not set, a default message is generated.
authserver hostname [*portnumber* [*cipherkey*]]	Specifies the name or address of a system to use for network authentication. If tn-gw is built with a compiled-in value for the server and port, these will be used as defaults but can be overridden if specified on this line. If support exists for DES-encryption of traffic in the server, an optional cipherkey can be provided to secure communication with the server.
hosts host-pattern [*host*-pattern2...] [*options*]	Specifies host and access permissions.

To illustrate the use of these rules to configure the rlogin-gw service, examine these sample rules from the netperm-table file:

```
rlogin-gw:    denial-msg      /usr/local/etc/rlogin-deny.txt
rlogin-gw:    welcome-msg     /usr/local/etc/rlogin-welcome.txt
rlogin-gw:    help-msg        /usr/local/etc/rlogin-help.txt
rlogin-gw:    denydest-msg    /usr/local/etc/rlogin-dest.txt
rlogin-gw:    timeout 3600
rlogin-gw:    prompt "Enter Command>"
rlogin-gw:    permit-hosts 204.191.3.* -dest *.fonorola.net -dest !* -passok
➥-xok
rlogin-gw:    deny-hosts * -dest 204.191.3.150
```

Note

> If any of the files identified in the denial-msg, welcome-msg, help-msg, or denydest-msg clauses are missing, the connection will be dropped as soon as a request is made for that file.

These rules are virtually identical to the rules used to configure the tn-gw. One exception is that the rlogin-gw is configured to display a different message when a connection request is made for a restricted host. The following output shows the different message for rlogin:

```
pc# rlogin pc
Welcome to the URG Firewall Rlogin Proxy

Supported commands are
        c[onnect] hostname [port]
        x-gw
        help
        password
        exit

To report problems, please contact Network Security Services at 555-1212 or
by e-mail at security@org.com

Enter Command>c fox.nstn.ca

*** ATTENTION ***

You have attempted to contact a restricted host from this rlogin proxy.  Your
attempt has been recorded.

To report problems, please contact Network Security Services at 555-1212 or
by e-mail at security@org.com

Enter Command>
```

Now that the proxy configuration is finished, you can move on to establishing a connection.

Connecting through the rlogin Proxy

Connecting through the rlogin proxy requires a process similar to the Telnet proxy. A connection is first established with the firewall host, and then the user requests a connection to the remote host. The commands supported by the rlogin proxy are the same as for the Telnet proxy. The following output illustrates a successful connection to a remote host using the rlogin proxy:

```
pc.unilabs.org$ rlogin pc
Welcome to the URG Firewall Rlogin Proxy

Supported commands are
        c[onnect] hostname [port]
        x-gw
        help
        password
        exit
```

```
To report problems, please contact Network Security Services at 555-1212 or
by e-mail at security@org.com

Enter Command>c nds.fonorola.net
Trying chrish@204.191.124.252...
Password:
Last login: Sun Oct  8 20:33:26 from pc.unilabs.org
SunOS Release 4.1.4 (GENERIC) #1: Wed Sep 13 19:50:02 EDT 1995
You have mail.
bash$
```

The user enters the name of the host he or she wants to connect to by using the c[onnect] command followed by the hostname. Before the connection request is made, the local username is added to the left of the requested hostname. Consequently,

```
nds.fonorola.net
```

becomes

```
chrish@nds.fonorola.net.
```

The establishment of the rlogin session to the remote host is then a matter of how the service is configured on that host. Remember that the name or IP address of the gateway must be in the .rhosts file because that is the machine where the connection is coming from, not the real originating host.

Host Access Rules

Host rules that permit and deny access to the rlogin proxy can be modified by a number of additional options, or rules. The host rules use the following format:

```
rlogin-gw:    deny-hosts unknown
rlogin-gw:    hosts 192.33.112.* 192.94.214.*
```

In this example, hosts that cannot be found in the DNS in-addr.arpa domain are unknown, and therefore denied; hosts connecting from the networks 192.33.112 and 192.94.214 are allowed to connect to the proxy. The optional parameters—each beginning with a hyphen—further restrict the hosts that can connect to the proxy by limiting where they can connect.

Verifying the rlogin Proxy

Operation of the rlogin proxy is verified by attempting to circumvent the established rules, and checking to see that the text from each of the configured files displays when it should display. For example, if your security policy states that only certain hosts can connect to the rlogin proxy, you must test this from each of the permitted hosts, and also test the connection from a few hosts that are not permitted.

Each rule for rlogin-gw must be carefully evaluated to ensure that it is operating as it should.

Configuring the FTP Gateway

The FTP proxy allows FTP traffic through the firewall to either private or public networks. The FTP proxy executes when a connection is made to the FTP port on the firewall. From there a connection could be made to the firewall, although it is not a good idea to allow FTP traffic to the firewall on the default port. It is better to have an additional FTP server system running elsewhere. A more secure setup would be to run the FTP server processes when a connection is made to a different port. By not publishing this port number, it is harder to have an FTP session established directly on the firewall.

Remember that the FTP service is found on port 21 as stated in the /etc/services file. To change this, edit the /etc/services file and add a second ftp entry called ftp-a—like the telnet-a that was added earlier. Establish this ftp-a service to run on a different port, such as 2021. The new /etc/services file will look like this:

```
ftp        21/tcp
ftp-a      2021/tcp
```

This new ftp-a entry only addresses part of the problem. The /etc/inetd.conf file is where the actual specification is made regarding which service is executed when a connection is made. The trick here is to configure the inetd.conf file so that when a connection is made to the FTP port, the ftp-gw application is started. When a connection is made to the ftp-a port, the real FTP server is started through the netacl application:

```
# ftp   stream  tcp    nowait  root   /usr/libexec/tcpd       ftpd -l -A
ftp     stream  tcp    nowait  root   /usr/local/etc/ftp-gw   ftp-gw
ftp-a   stream  tcp    nowait  root   /usr/local/etc/netacl   ftpd
```

Three entries for the FTP service are included here to illustrate a point. The first entry is uncommented out and is provided to show you how the FTP service was originally started. The second entry establishes a connection to the FTP proxy. The third line allows FTP connections to the firewall itself. Examine the configuration of the ftp-gw proxy application first.

The ftp-gw proxy, like the other Toolkit applications, reads the lines in the netperm-table file that start with the application name, ftp-gw. Table 10.7 lists clauses that are understood by ftp-gw.

TABLE 10.7
The ftp-gw Program Rules

Rule	*Description*
userid user	Specifies a numeric userid or the name of a password file entry. If this value is specified, ftp-gw will set its userid before providing service.
directory pathname	Specifies a directory to which ftp-gw will chroot(2) prior to providing service.
denial-msg filename	Specifies the name of a file to display to the remote user if he or she is denied permission to use the proxy. If this option is not set, a default message is generated. When the denial-msg file is displayed to the remote user, each line is prefixed with the FTP codes for permission denied.
welcome-msg filename	Specifies the name of a file to display as a welcome banner upon successful connection. If this option is not set, a default message is generated.
help-msg filename	Specifies the name of a file to display if the help command is issued. If this option is not set, a list of the internal commands is printed.
denydest-msg filename	Specifies the name of a file to display if a user attempts to connect to a remote server from which he or she is restricted. If this option is not set, a default message is generated.
timeout secondsvalue	Specifies the idle timeout value in seconds. When the specified number of seconds elapses with no activity through the proxy server, it will disconnect. If this value is not set, no timeout is enforced.

If these options are not used, default values are used instead. When these options are used, however, the ftp-gw rules look like this:

```
ftp-gw: denial-msg      /usr/local/etc/ftp-deny.txt
ftp-gw: welcome-msg     /usr/local/etc/ftp-welcome.txt
ftp-gw: help-msg        /usr/local/etc/ftp-help.txt
ftp-gw:         timeout 3600
ftp-gw: denydest-msg    /usr/local/etc/ftp-badest.txt
```

By using the Host Access rules, you can control who has access to your private network using FTP, or to whom your internal users can connect.

Host Access Rules

The host rules that permit and deny access to the FTP proxy can be modified by a number of additional options. The host rules use the format:

```
ftp-gw:    deny-hosts unknown
ftp-gw:    hosts 192.33.112.* 192.94.214.*
```

In this example, hosts that cannot be found in the DNS in-addr.arpa domain are unknown, and therefore denied; hosts connecting from the network 192.33.112 and 192.94.214 are allowed to connect to the proxy. The optional parameters—each beginning with a hyphen—further restrict the hosts that can connect to the proxy by limiting where they can connect.

Like the other proxy agents, a number of options, listed in table 10.8, are available for controlling the proxy.

Table 10.8
Host Access Options

Option	*Description*
-dest pattern -dest { pattern1 pattern2 ... }	Specifies a list of valid destinations. If no list is specified, all -dest destinations are considered valid. The -dest list is processed in the order it appears on the options line. -dest entries preceded with a "!" character are treated as negation entries.
-auth	Specifies that the proxy should require a user to authenticate with a valid userid prior to being permitted to use the gateway.
-passok	Specifies that the proxy should permit users to change their passwords if they are connected from the designated host. Only hosts on a trusted network should be permitted to change passwords, unless token-type authenticators are distributed to all users.

The use of an IP address instead of a domain name does not alter the rule. Before the connection is permitted, the tn-gw application attempts to validate the IP address. If the returned host matches one of the rules, then the rule is applied. Otherwise, the connection is dropped.

Verifying the FTP Proxy

Verifying the operation of the FTP proxy involves testing each of the rules and connection points. For example, if you are allowing FTP sessions to originate from the private network, but deny FTP access to hosts outside the private network, then the ftp-gw rules would look like this:

```
ftp-gw: permit-hosts    206.116.65.*    -log { retr stor }
```

This can only be verified by attempting to establish an FTP session from a host on the LAN and going out to the public network. To prove the proper operation of the proxy, a connection from the public network to a machine on the private network must be attempted. The following command sequence illustrates the use of Telnet to access the firewall from a host on the internal network:

```
C:\WINDOWS>ftp pc.unilabs.org
Connected to pc.unilabs.org.
220-Welcome to the URG Firewall FTP Proxy
220-
220-To report problems, please contact Network Security Services at 555-1212
or 220-by e-mail at security@org.com
220
User (pc.unilabs.org:(none)): chrish@nds.fonorola.net
331-(----GATEWAY CONNECTED TO nds.fonorola.net----)
331-(220 nds.fonorola.net FTP server (Version A) ready.)
331 Password required for chrish.
Password:
230 User chrish logged in.
ftp>
```

Notice that the user was allowed access to the FTP proxy, and an FTP session was established to the machine nds.fonorola.net. The converse for this rule then must also be true: Any host outside the private network is not permitted access to the FTP proxy. The following output illustrates this restriction:

```
bash$ ftp pc.unilabs.org
Connected to pc.unilabs.org.
500-
500-**** ATTENTION ****
500-
500-Your attempt to use this server's ftp proxy is not permitted due to
500-organizational security policies.  Your connection attempt has been
➥logged
```

```
500-and recorded.
500-
500-If you believe that you are an authorized site, please contact Jon Smith
500-at 555-1212 ext 502, or e-mail to ftpadmin@org.com.
500
ftp>
```

In this situation, the user on the system nds.fonorola.net attempted to connect to the firewall, but because its IP address [204.191.124.252] is not within the address space specified on the ftp-gw rule, the connection is denied, and the message shown here appears. Remember that this message is from the denial-msg rule in the configuration file.

Connecting through the FTP Proxy

Establishing a connection through the proxy involves connecting to the FTP port and then specifying the host to connect to. The target specification, however, is not quite what you might expect:

```
$ ftp 204.191.3.150
Connected to 204.191.3.150.
220 pc.unilabs.org FTP proxy (Version V1.3) ready.
User (204.191.3.150:(none)): anonymous@ftp.fonorola.net
331-(----GATEWAY CONNECTED TO ftp.fonorola.net----)
331-(220 net FTP server (Version wu-2.4(1) Fri Apr 21 22:42:18 EDT 1995) ready.)

331 Guest login ok, send your complete e-mail address as password.
Password:
230-
230-                    Welcome to i*internet Inc.
230-                      Anonymous FTP Server
230-
230-We are currently in the process of deploying the Washington
230-University Anonymous FTP Server.
230-
230 Guest login ok, access restrictions apply.
ftp>
```

When establishing a connection through the proxy, you first run the ftp command and connect to the firewall, which serves as the host. After you are connected, you must specify the user name and the site to connect to. This is done using the syntax:

```
user@site
```

After validating that the site is indeed one that is allowed, the proxy connects to the FTP server on the remote system and starts to log in using the supplied username. The remote server then prompts for the user's password, and if it is correct, allows the connection.

ALLOWING FTP WITH NETACL

It is fairly common to restrict the proxy from connecting to the firewall for FTP services, but occasionally you may need to upgrade software or change text files and messages. For this reason, you may need to enable FTP access. This can be done using the services of netacl. With netacl, you can restrict what machines can connect to the firewall to specific machines within the local network. Consider the sample configuration entries in the following command:

```
netacl-ftpd: permit-hosts 204.191.3.* -exec /usr/libexec/ftpd -A -l
```

This entry for netacl allows systems on the 204.191.3 network to connect to the FTP server through netacl. The entry also locks out all other systems, as you can see when one of them tries to access the FTP server:

```
ftp> open 198.53.166.62 2021
Connected to 198.53.166.62.
421 Service not available, remote server has closed connection
ftp>
```

From this message it appears that there is no server listening on port 2021, when in fact there is. netacl does not allow the request because the IP address where the request originated does not match the rule established previously.

If you're not sure whether you will ever need access for FTP services to the firewall, the safest thing to do is to not allow this type of access except when absolutely necessary. This means that netacl can be set up in the netperm-table file, but commented out, thereby making it unavailable. Furthermore, the proxy must be configured to prevent connections to the firewall on the FTP port.

CONFIGURING THE SENDMAIL PROXY

SMAP AND SMAPD

Two components are used for the successful delivery of mail through the firewall: smap and smapd. The smap agent is a client that implements a minimal version of SMTP. The smap program accepts messages from the network and writes them to disk for future delivery by smapd. smap is designed to run under chroot as a non-privileged process; this setup overcomes potential security risks from privileged mailers that can be accessed from over a network.

The smapd daemon periodically scans the mail spool area maintained by smap and delivers any messages that have been gathered and stored. Mail is delivered by sendmail, and the spool file is deleted. If the mail cannot be delivered normally, smapd can be configured to store spooled files to an area for later examination.

These two applications can share configuration information in the netperm-table file if desired. Some of the operations are different, so different steps need to be taken when configuring the two applications.

Installing the smap Client

The smap client runs whenever a connection request is received on the smtp port of the firewall. This is done by adding an entry for smtp to the /etc/inetd.conf file:

```
smtp    stream  tcp    nowait  root    /usr/local/etc/smap     smap
```

After /etc/inetd.conf has been updated, the inetd process must be restarted so that smap accepts connections. This can be checked by connecting manually to the smtp port:

```
pc# telnet pc 25
Trying 206.116.65.3...
Connected to pc.unilabs.org.
Escape character is '^]'.
220 pc.unilabs.org SMTP/smap Ready.
helo
250 Charmed, Im sure.
help
214-Commands
214-HELO    MAIL    RCPT    DATA    RSET
214 NOOP    QUIT    HELP    VRFY    EXPN
quit
221 Closing connection
Connection closed by foreign host.
pc#
```

As you can see, smap implements a minimal SMTP implementation, and spools the mail into the specified spool area. In the spool directory, it may be required that an etc directory with system specific configuration files be installed. A recommended setup is to build smap so that it is completely stand-alone—it does not depend on other libraries and will run without fail.

Configuring the smap Client

The smap client reads its configuration from the netperm-table file by looking for the lines beginning with smap. If the line applies to both smap and smapd, the two programs can be listed on the same line by separating them with a comma:

```
smap, smapd:    userid 6
```

The rules for smap are listed in table 10.9.

Table 10.9
smap Rules

Rule	*Description*
userid name	Specify the userid under which smap should run. The name can be either a name from the password database, or a numeric userid. This userid should be the same as that under which smapd runs, and should have write permission to the spool directory.
directory pathname	Specifies the spool directory where smap should store incoming messages. A chroot system call is used to irrevocably make the specified directory the root file system for the remainder of the process.
maxbytes value	Specifies the maximum size of messages to gather, in bytes. If no value is set, message sizes are limited by the amount of disk space in the spool area.
maxrecip value	Specifies the maximum number of recipients allowed for any message. This option is only for administrators who are worried about the more esoteric denial of service attacks.
timeout value	Specifies a timeout, after which smap should exit if it has not collected a message. If no timeout value is specified, smap will never time out a connection.

As you can see in table 10.9, some items are common between the smap and smapd applications. These similarities will be discussed later. For now, develop a configuration section for the smap application.

The userid, directory, and timeout values are self-explanatory. However, unlike the directory clauses for the other applications, the smap client also uses the directory to save incoming messages. Consequently, these form the basis of your configuration:

```
smap:    userid 6
smap:    directory /var/spool/smap
smap:    timeout 3600
```

The maxbytes value specifies the size of the largest email message. If the message is larger than the maxbytes value, the message size is truncated. If maxbytes is not included in the configuration information, then the maximum message size is the size of the available space in the spool area. The final clause specifies the maximum number of recipients that can be attached to the mail message. This is not a commonly used option. The completed entry for the netperm-table file looks like this:

```
smap:    userid 6
smap:    directory /var/spool/smap
smap:            timeout 3600
smap:            maxbytes        10000
smap:            maxrecip        20
```

If you set the value of maxbytes too small, users may not be able to receive some messages because of the message's size. This type of problem reveals itself in the log files. Lines that resemble the following indicate the incoming mail message is too large to process:

```
Oct 29 12:09:52 pc smap[868]: connect host=unknown/198.53.64.9
Oct 29 12:09:59 pc smap[868]: exiting too much data
```

No other warnings of this problem occur. This is the only way the firewall operator can check to see if large messages are the reason why mail isn't being sent.

At this point, you have installed and configured the smap application. It is not very difficult to complete its setup.

Installing the smapd Application

Unlike smap, which is started from inetd on a connection by connection basis, smapd is started from the /etc/rc.local script and runs the entire time the system is running. The daemon startup is added to the file /etc/rc.local and then the system is rebooted. The following shows the addition of the command to the rc.local file:

```
echo "Starting Firewall Mail Processor ..."
/usr/local/etc/smapd
```

Because sendmail is not running in daemon mode, messages that cannot be delivered and are queued must be delivered by periodically invoking sendmail to process the queue. To do this, add a line similar to the following to the crontab file:

```
0,30 * * * * /usr/sbin/sendmail -q > /dev/null 2>&1
```

This ensures that any messages that cannot be successfully delivered by the smapd application will be properly handled.

Configuring the smapd Application

The configuration of the smapd application is no more difficult than configuring smap. They generally run without a problem. Like smap, smapd reads its configuration from the netperm-table file; it accepts no command-line arguments. The smap application reads the mail queue on a periodic basis and delivers mail to the remote system. Rules that are available to build the smapd configuration file are listed in table 10.10.

Table 10.10
smapd Rules

Rule	*Description*
executable pathname	Specifies the pathname of the smapd executable. For historical reasons, smapd forks and execs copies of itself to handle delivering each individual message. THIS ENTRY IS MANDATORY.
sendmail pathname	Specifies an alternate pathname for the sendmail executable. smapd assumes the use of sendmail but does not require it. An alternate mail delivery system can replace sendmail, but it should be able to accept arguments in the form of executable -f fromname recip1 [recip2 ... recipN]. The exit code from the mailer is used to determine the status of delivery; for this reason, replacements for sendmail should use similar exit codes.
baddir pathname	Specifies a directory where smapd should move any spooled mail that cannot be delivered normally. This directory must be on the same device as the spool directory because the rename(2) system call is employed. The pathname specified should not contain a trailing "/".
userid name	Specifies the userid that smapd should run under. The name can be either a name from the password database, or a numeric userid. This userid should be the same as that under which smap runs, and should have write permission to the spool directory.
directory pathname	Specifies the spool directory in which smapd should search for files. smapd should have write permission to this directory.
wakeup value	Specifies the number of seconds smapd should sleep between scans of the spool directory. The default is 60 seconds.

Some options are common for smap and smapd. Nevertheless, you can build a separate configuration for smapd, such as the one shown here:

```
smapd:          executable /usr/local/etc/smapd
smapd:          sendmail /usr/sbin/sendmail
smapd:          userid 6
smapd:          directory /var/spool/smap
smapd:          baddir /var/spool/smap/bad
smapd:          wakeup 900
```

This configuration defines the operating parameters for smapd. The executable rule identifies the location of the smapd program. This rule is mandatory. The sendmail option specifies where the sendmail program is found. Alternate programs such as zmailer or smail can be used in place of sendmail, as long as they conform to the exit codes used within sendmail.

The userid and directory rules specify the user under which the smapd binary executes, and the home directory used for that configuration. The baddir value is related to directory. The value assigned to directory provides the name of the directory where the in transit mail messages are stored; a bad directory will be created there to save any undelivered or questionable messages.

The last value for smapd specifies how long the delay is between the processing of the queue. The default is 60 seconds; this example uses a 15 minute window.

Configuring DNS for smap

For mail to be successfully and correctly routed through the firewall, MX records need to be published in the zone's DNS files to identify where SMTP mail is to be sent. This is done by adding MX, or mail exchanger, records to the DNS providers for the network domain, or zone. The zone information shown here provides some information regarding how this is configured.

```
Server:  nic.fonorola.net
Address:  198.53.64.7

unilabs.org     nameserver = nic.fonorola.net
unilabs.org     nameserver = fonsrv00.fonorola.com
unilabs.org     preference = 10, mail exchanger = mail.fonorola.net
unilabs.org     preference = 1, mail exchanger = pc2.unilabs.org
unilabs.org     preference = 5, mail exchanger = nis.fonorola.net
unilabs.org
        origin = nic.fonorola.net
        mail addr = chrish.fonorola.net
        serial = 95102902
        refresh = 10800 (3 hours)
        retry   = 1800 (30 mins)
        expire  = 3600000 (41 days 16 hours)
        minimum ttl = 86400 (1 day)
```

```
unilabs.org     nameserver = nic.fonorola.net
unilabs.org     nameserver = fonsrv00.fonorola.com
nic.fonorola.net        internet address = 198.53.64.7
fonsrv00.fonorola.com   internet address = 149.99.1.3
mail.fonorola.net       internet address = 198.53.64.8
pc2.unilabs.org         internet address = 198.53.166.62
nis.fonorola.net        internet address = 198.53.64.14
>
```

This output is from the nslookup command. Despite how this looks, you are in fact looking for the lines that contain the description mail exchanger, which are as follows:

```
unilabs.org     preference = 1, mail exchanger = pc2.unilabs.org
unilabs.org     preference = 5, mail exchanger = nis.fonorola.net
unilabs.org     preference = 10, mail exchanger = mail.fonorola.net
```

When mail for the domain unilabs.org is to be sent from a host, that host will first try to locate the unilabs.org domain itself. The rule determining which host will be contacted first is simple: The host that has the lowest preference value is the first to be contacted. In the sample setup you've watched develop throughout this chapter, the host pc2.unilabs.org, which is the firewall, will be contacted first to see if it can in fact accept the e-mail. A recommended setup is to give the firewall the lowest priority on the system, so that no other machines can be directly contacted by the outside world.

If the machine with the lowest preference value is not available, then the next system is contacted—in this case, nis.fonorola.net. If the mail is delivered to nis.fonorola.net, then the sendmail daemon on nis will now take responsibility for attempting to deliver it to the lowest preference value machine, pc2.unilabs.org. The same is true should the second mail system not be available and the mail server must then contact the third system. The behavior described here may not be what happens in all situations. For example, the system nis.fonorola.net could simply decide to attempt delivery itself and not use the next MX record. The operation of sendmail is controlled by the sendmail.cf file on the remote machine. Remember that when you make changes to your DNS, you must restart or reload the DNS so that the new information is integrated into the DNS.

CONFIGURING THE HTTP PROXY

The HTTP proxy, http-gw, does more than simply provide a mechanism for HTTP requests to be sent through the firewall. It also provides support for Gopher clients, so that Gopher, Gopher+, and FTP requests can originate from a Gopher client, and for HTPP, Gopher, Gopher+, and FTP requests to be passed through from a WWW client.

The HTTP proxy also supports "proxy-aware" clients, and supports clients that are not designed to work with these daemons. Before examining how to enable these services, first examine the steps required to place the proxy into operation, and also look at the configuration rules for this proxy.

By default, an HTTP or Gopher server usually runs on TCP/IP ports 80 and 70, respectively. These will not be running on the firewall, so it is necessary to configure inetd to accept connections on these ports and start the proxy agent. This is done by adding the following line to the /etc/services file:

```
gopher          70/tcp
httpd           80/tcp
```

With these lines added, inetd now knows on what ports to listen. inetd must then have the appropriate lines added to its configuration file, inetd.conf:

```
httpd    stream  tcp     nowait  root    /usr/local/etc/http-gw  http-gw
gopher   stream  tcp     nowait  root    /usr/local/etc/http-gw  http-gw
```

With the inetd configuration file now updated, inetd must be restarted, or instructed to read its configuration file using the kill -1 command. When these steps are completed, the http-gw proxy is ready to configure.

http-gw reads its configuration rules and permissions information from the firewall configuration table netperm-table, retrieving all rules specified for "http-gw." The "ftp-gw" rules are also retrieved and are evaluated when looking for host rules after all the http-gw rules have been applied. Table 10.11 lists configuration rules applicable to this proxy.

Table 10.11
http-gw Proxy Rules

Option	*Description*
userid user	Allows the system administrator to specify a numeric userid or the name of a password file entry. If this value is specified, http-gw will set its userid before providing service. Note that this option is included mostly for completeness; http-gw performs no local operations that are likely to introduce a security hole.
directory pathname	Specifies a directory to which http-gw will chroot prior to providing service.
timeout secondsvalue	Used as a dead-watch timer when the proxy is reading data from the Internet. Defaults to 60 minutes.
default-gopher server	Defines a Gopher server to which requests can be handed off.

continues

TABLE 10.11, CONTINUED
http-gw Proxy Rules

Option	*Description*
default-httpd server	Defines an HTTP server to which requests can be handed off if they came from a WWW client using the HTTP protocol.
ftp-proxy server	This defines an ftp-gw that should be used to access FTP servers. If not specified, the proxy will do the FTP transaction with the FTP server. The ftp-gw rules will be used if there are no relevant http-gw rules, so this is not a major problem.

The userid, directory, and timeout values serve the same functions as the other proxy agents in the Toolkit. However, you need to examine the rules that the default-httpd server, default-gopher server, and default-ftp server play. To understand their impact, you need to examine how a non–proxy-aware and a proxy-aware WWW client operate.

NON–PROXY-AWARE HTTP CLIENTS

A non–proxy -aware HTTP client, such as the Internet Explorer Version 1.0 from Microsoft, cannot communicate with a proxy. The user must configure the client to connect first to the firewall, and then to go to the desired site. To do this, the user must specify the URL in the format:

```
http://firewall_system/http://destination
```

as in

```
http://pc.unilabs.org/http://www.unilabs.org
```

The client will pass the request for `http://www.unilabs.org` to the firewall. The firewall then establishes the connections required to bring the requested information to the client.

Although a proxy-aware client can still use this format, this is the only format that can be used with non-proxy HTTP clients. World Wide Web clients are also capable of accessing FTP and Gopher services. Table 10.12 lists the URL formats used for each of these services.

Table 10.12
Supported URL Formats

Service	*URL*
HTTP	http://firewall_name/http://www_server
Gopher	http://firewall_name/gopher://gopher_server
FTP	http://firewall_name/ftp://FTP_server

Internet users who work with non–proxy-aware clients need to make changes to their WWW client if a firewall is installed after the users have developed and built their hotlists. In these situations, their WWW client hotlists will have to be edited to include the firewall in the URL.

Using a Proxy-Aware HTTP Client

A proxy-aware HTTP client such as Netscape Navigator or NCSA Mosaic does not have these problems. However, some application-specific configuration is required to make it work. Although nothing additional must be done on the HTTP proxy side, the client must be configured with the appropriate proxy information.

Aside from this application-specific customization, there are no other difficulties in using the proxy-aware client. When these WWW clients have been configured, they are much easier for the end user to handle because there is less confusion in accessing sites.

All World Wide Web clients can access Gopher (and FTP) sites. As you have seen, if the client is aware of the proxy, access to these different types of Internet sites is much simpler to set up. Accessing a Gopher server with a World Wide Web browser is much easier than with many Gopher clients, if the World Wide Web browser is proxy-aware. Connecting to the Gopher server is as simple as specifying an URL:

```
http://firewall_host_name/gopher://gopher_server_name
```

This syntax allows the connection to the external Gopher server through the firewall.

Host Access Rules

Up to this point in the chapter, you have seen how the user interacts with the proxy. Now examine how you can alter the operation of the proxy by applying some host access rules. Some of these rules have been examined already, and are important enough to mention again. The host access rules may include optional parameters to further control the session. Some of these parameters include restricting the allowable functions. The rules and their parameters are included in table 10.13.

TABLE 10.13
Host Access Rules

Option	*Descriptions*
Hosts host-pattern [host-pattern ...] [options] Permit-hosts host-pattern [host-pattern ...] options] Deny-hosts host-pattern [host-pattern ...]	Rules specify host and access permissions. Typically, a host rule will be in the form of: http-gw: deny-hosts unknown http-gw: hosts 192.33.112.* 192.94.214.*
-permit function -permit { function [function ...] }	Only the specified functions are permitted. Other functions will be denied. If this option is not specified, then all functions are initially permitted.
-deny function -deny { function [function ...] }	Specifies a list of Gopher/HTTP functions to deny.
-gopher server	Makes server the default server for this transaction.
-httpd server	Makes server the default HTTP server for this transaction. This will be used if the request came in through the HTTP protocol.
-filter function -filter { function [function ...] }	Removes the specified functions when rewriting selectors and URLs. This rule does not stop the user from entering selectors that the client will execute locally, but this rule can be used to remove them from retrieved documents.

Several host patterns may follow the "hosts" keyword; the first optional parameter after these patterns begins with "-". Optional parameters permit the selective enabling or disabling of logging information.

Some basic configuration rules are shown here to help you understand how the options for host rules are used:

```
http-gw:        userid www
# http-gw:      directory /usr/local/secure/www
http-gw:        timeout 1800
http-gw:        default-httpd www.fonorola.net
http-gw:        default-gopher gopher.fonorola.net
http-gw:        permit-hosts 206.116.65.*
```

The permit-hosts line establishes what hosts or networks are allowed to pass through the firewall using the proxy. To deny access to specific hosts or networks, use a line similar to:

```
http-gw:    deny-hosts 206.116.65.2
```

When this type of setup is in operation, a user who is trying to use the proxy from this machine receives a `Sorry, access denied` error message.

The permit-host rules can include function definitions that are permitted or denied depending on the established criteria in the rule. The proxy characterizes each transaction as one of a number of functions. For the deny options the request is used; for filter options the returned selectors are used. These functions are listed in table 10.14.

TABLE 10.14
Function Definitions

Function	*Description*
dir	Fetching Gopher menus. Getting a directory listing via FTP. Fetching an HTML document.
read	Fetching a file of any type. HTML files are treated as read, even though they are also dir.
write	Putting a file of any type. Needs Gopher+ since only available to Gopher+ and HTTP/1.x.
ftp	Accessing an FTP server.
plus	Gopher+ operations. HTTP methods other than GET.
wais	WAIS index operations.
exec	Operations that require a program to be run; that is, Telnet.

Function controls enable the firewall administrator to specifically set up what will and will not be allowed to pass through the proxy. If no deny or permit functions are specified, every function is permitted. Consider, for example, a setup that would not allow file transfers using the -deny ftp command:

```
http-gw:        userid www
# http-gw:      directory /usr/local/secure/www
http-gw:        timeout 1800
http-gw:        default-httpd www.fonorola.net
http-gw:        default-gopher gopher.fonorola.net
http-gw:        permit-hosts 206.116.65.* -deny ftp
# http-gw:      deny-hosts 206.116.65.2
http-gw:        deny-hosts unknown
```

By using this deny request to restrict the use of the ftp command, users can no longer request an FTP session through the http-gw proxy. A sample error message would look like:

```
use file fig11.pcx
```

In this configuration, any attempt to establish an FTP session using either the following syntax or a WWW page will result in failure:

```
ftp://ftp.somewhere.com
```

Note

> If you are concerned about FTP transfers, and you have disabled the ftp-gw proxy to prevent FTP transfers, you need to carefully consider the value of disabling the ftp commands in the HTTP protocol set. Closing one door but leaving a related one open is not wise.

Few of the current Gopher clients are capable of interacting as well as proxy-aware WWW clients. To use a Gopher client, you must configure the default Gopher server that is used to establish the connection to the firewall. From here you will have to configure jumping off points to different Gophers.

Because of the looming difficulty associated with Gopher clients, the use of Gopher via the World Wide Web interface is popular and widely accepted. Clearly, this capability indicates that there is more flexibility within the HTTP architecture.

Configuring the X Windows Proxy

The x-gw X Windows proxy is provided to allow a user-level X Windows interface that operates under the tn-gw and rlogin-gw access control. Recall from the earlier discussion of the tn-gw command that this command enables an X session through the gateway.

The proxy operates by allowing clients to be started on arbitrary hosts outside the firewall, and then requesting a connection to the specified display. When the X connection request by the client is made, the x-gw proxy displays a window that is running on a virtual display on the firewall. Upon receiving the connection request, x-gw displays the window on the user's real display. This display prompts for confirmation before proceeding with the connection. If the user agrees to accept the connection, x-gw passes the data from the virtual display to the user's real display.

The x-gw proxy can be started from a telnet or rlogin sequence, as shown by this output:

```
% telnet pc
Trying 206.116.65.3...
Connected to pc.unilabs.org.
Escape character is '^]'.
pc.unilabs.org telnet proxy (Version V1.3) ready:
tn-gw-> x
tn-gw-> exit
Disconnecting...
Connection closed by foreign host.
```

At this point a window pops up on the user's display that shows the port number of the proxy to use; the window also serves as the control window. Choosing the Exit button will close all multiple X connections.

Although the x-gw proxy is advanced and user-friendly, some issues concerning this proxy need to be mentioned. The major issue is that this proxy relies on the X11 Athena Widget set. If your system does not have the X11 libraries or the Athena Widget set, this proxy will not compile, and you will be forced to live without it. Fortunately, very few people allow the use of X windows applications through their firewall.

Understanding the Authentication Server

The TIS Firewall Toolkit includes extensive authentication mechanisms. The TIS authentication server consists of two components: the actual server itself, and a user authentication manager, which is used to interact with and configure the server.

The authentication server, known as authsrv, is designed to support multiple authentication processes independently. This server maintains an internal user database that contains a record for each user. The information stored for each user consists of:

- The user's name
- The user's group
- The user's full name
- The last successful authentication

Passwords may be plaintext for local users, or encrypted for all others. The only time plaintext passwords would be used is when the administrator wants to control access to firewall services by users on the protected network.

WARNING

> Plaintext passwords should never be used for authentication by users on nonsecure networks.

Users in the authsrv database can belong to different groups; a group administrator can be named who can only manage the users in that group. authsrv also contains support for multiple forms of authentication, including:

- Internal plaintext passwords
- Bellcore's S/Key
- Security Dynamics SecurID
- Enigma Logics Silver Card
- Digital Pathways SNK004 Secure Net Key

NOTE

> The Bellcore S/Key mechanism that is included with the Toolkit does not include the complete software. The entire S/Key distribution can be downloaded via FTP from thumper.bellcore.com.

When compiling authsrv, the administrator needs to decide which authentication forms will be supported locally. It is typical to find multiple forms in use by a single company depending on cost and availability. For each proxy in the Toolkit, authentication can be enabled or disabled, or fit certain criteria, such as incoming must authenticate, and outgoing requires no authentication.

Authsrv should be run on as secure a host as possible, which is generally the firewall itself. To configure the authentication server, you must find an unused TCP/IP port number and add it to /etc/services. For example, if you use port 7777 as the TCP port, the following line would be added to the /etc/services file.

```
authsrv         7777/tcp                # TIS Toolkit Authentication
```

Authsrv is not a daemon. It runs whenever a connection request is made on the specified TCP port. Consequently, it is necessary to add an entry to the /etc/inetd.conf file, such as this example:

```
authsrv stream  tcp     nowait  root    /usr/local/etc/authsrv  authsrv
```

After the required entries are placed in the /etc/services and /etc/inetd.conf files, inetd must be reloaded or restarted using the kill command. At this point, individual clients must be configured to use the authentication server when required. Keep in mind that not all operations need to require authentication.

To configure a given proxy, you must use the port number and the authserver keyword specifying the host to connect to for the authentication server. To see this in action, consider adding authentication to the FTP proxy. For the FTP proxy to be able to use the authentication server, you must tell it to use authserver rule:

```
# Use the following lines to use the authentication server
ftp-gw: authserver      localhost       7777
```

When the FTP proxy is activated, requests must be authenticated. The permit-hosts entry, however, has the flexibility to take advantage of the authentication system. For example, consider the permit-hosts entry in the following:

```
ftp-gw: permit-hosts    206.116.65.*    -log { retr stor } -auth { stor }
```

The permit-hosts entry says that all retrieve and store file requests to the FTP proxy are logged, and all store file requests are blocked until the user has authenticated. This process will be demonstrated later in this chapter after you learn how to configure the users in the authentication database.

THE AUTHENTICATION DATABASE

The authentication server must also be configured to accept connections from specific clients. This prevents unwanted attempts to probe the authentication server from hosts running software that needs no authentication. The authentication server reads its rules from the netperm-table, which can include rules listed in table 10.15.

Table 10.15
Authentication Server Rules

Rule	*Description*
database pathname	Specifies the pathname of the authsrv database. The database is stored as a dbm(3) file with a third file used for locking. If the software is built with a compiled-in database name, this option need not be set; otherwise, it is mandatory.
nobogus true	Indicates that authsrv should return "user-friendly" error messages when users attempt to authenticate and fail. The default message is to simply respond, "Permission Denied" or to return a bogus challenge. If nobogus is set, attempts to log on will return more explicit error messages. Sites that are concerned about attempts to probe the authentication server should leave this option disabled.
badsleep seconds	Establishes a "sleep time" for repeated bad logins. If a user attempts to authenticate five times and fails, his user record is marked as suspicious, and he cannot log on again. If the badsleep value is set, the user may attempt to log in again after the set number of seconds has expired. If the badsleep value is 0, users can attempt to log in as many times as they would like. The default value is to effectively disable the account until an administrator re-enables it manually.
userid name	Specifies the userid under which authsrv should run. The name can be either a name from the password database, or a numeric userid.
hosts host-pattern [key]	Specifies that authsrv should permit the named host or addresses to use the service. Hosts that do not have a matching entry are denied use of the service. If the optional key is specified, and the software is compiled with DES-encrypted communications, all traffic with that client will be encrypted and decrypted with the specified key.

Rule	*Description*
operation userid telnet-gw host	Operation rules are stored in operation userid ftp-gw host put netperm-table. For each user/group the name is specified followed by the service destination [optional tokens] [time start end]. The user/group field indicates whether the record is for a user or a group. The name is either the username or the group name. The service can be a service specified by the proxy (usually ftp-gw, tn-gw, or rlogin-gw). The destination can be any valid domain name. The optional tokens are checked for a match, permitting a proxy to send a specific operation check to the authentication server. The time field is optional and must be specified time start_time end_time; start_time and end_time can be in the range 00:00 to 23:59.

If no other systems on the private network require access to the authsrv, then clients and the server should be configured to accept connections only using the localhost name or IP address 127.0.0.1. The authentication server configuration rules shown earlier illustrate a sample configuration for the server.

The example shown here establishes the following rules for the authentication server:

```
authsrv:        hosts 127.0.0.1
authsrv:        database /usr/local/etc/fw-authdb
authsrv:        badsleep 1200
authsrv:        nobogus true
```

- Identifies that the localhost is allowed to access the server
- Specifies that the authentication database is found in /usr/local/etc/fw-authdb
- The user cannot attempt to authenticate after five bad logins until 1,200 seconds have expired
- Prints more verbose messages about authentication failures

The operation rule is essential to administrators who want to restrict the commands that can be executed by certain users at certain times. This is done by adding configuration rules consisting of the user, the operation, and the time restrictions to the netperm-table. These rules apply to the authsrv command and not to the individual proxies themselves. Consider the following example:

```
authsrv  permit-operation  user  chrish telnet-gw relay.cdnnet.ca time 08:00
➥17:00
authsrv deny-operation  user  paulp  telnet-gw  mailserver.comewhere.com
➥time 17:01 07:59
authsrv permit-operation group admin telnet-gw * time 08:00 17:00
```

You can see that through careful consideration, the availability of various services can be tightly controlled depending on the environment and the organization's security policy. With the authentication server configured and ready, users must now be added so that they can be authenticated whenever necessary.

Adding Users

Before a user can be authenticated by the server, the user must be added to the database. This can be done by using the authsrv command. When invoking authsrv on the firewall with a userid of zero, authsrv grants administrative privileges for the database.

The authentication server has a number of commands, listed in table 10.16, for user administration.

Table 10.16
Administrator Commands for Authentication Setup

Command	*Description*
adduser username [longname]	Adds a user to the authentication database. Before the authentication server permits the use of this command, the administrator must first be authenticated to the server as an administrator or a group administrator. If the user is a group administrator, the newly created user is automatically initialized as a member of that group. When a user is added, the user is initially disabled. If a long name is provided, it will be stored in the database. Long names should be quoted if they contain whitespace.
deluser username	Deletes the specified user from the authentication database. Before an administrator can use this command, he or she must first be authenticated to the server as the administrator or group administrator of the group to which the user belongs.

Command	*Description*
display username	Displays the status, authentication protocol, and last login of the specified user. Before the authentication server permits the use of this command, the administrator must first be authenticated to the server as the administrator or as the group administrator of the group to which the user belongs.
enable username or disable username	Enables or disabled the specified user's account for login. Before this command can be used, the administrator must first be authenticated to the server as the administrator or group administrator of the group to which the user belongs.
group user groupname	Sets the specified user's group. To use this command, the administrator must first be authenticated to the server as the administrator. Group administrators do not have the power to "adopt" members.
list [group]	Lists all users that are known to the system, or the members of the specified group. Group administrators may list their own groups, but not the entire database. The list displays several fields, including: ✣ user. The login ID of the user. ✣ group. The group membership of the user. If none is listed, the user is in no group. ✣ longname. The user's full name. This may be left blank. ✣ status. Contains codes indicating the user's status.
password [username] text	Sets the password for the current user. If an optional user name is given and the authenticated user is the administrator or group administrator, the password for the specified user is changed. The password command is polymorphic depending on the user's specified authentication protocol. For example, if the user's authentication protocol is plaintext passwords, it will update the plaintext password. If the authentication protocol is SecurID with PINs, it will update the PIN.

continues

Table 10.16, Continued
Administrator Commands for Authentication Setup

Command	*Description*
proto user protoname	Sets the authentication protocol for the specified user to the named protocol. Available protocols depend on the compiled-in support within authsrv. To change a user's authentication protocol, the administrator must be authenticated to the server either as the administrator or group administrator of the user's group.
quit or exit	Disconnects from the authentication server.
superwiz user	Sets the specified user as a global administrator. This command should only be used with deliberation; global administrative privileges are seldom used because the group mechanism is powerful enough.
wiz user	Sets or turns off the group or unwiz user administrator flag on the specified user. To issue this command, the administrator must be authenticated to the server as the administrator.
? or help	Lists a short synopsis of available commands.

To illustrate the use of these administrator commands, suppose you want to add a new user to the database. To do this, make sure you are logged in as root on the firewall, and run the authsrv command:

```
pc# pwd
/usr/local/etc
pc# ./authsrv
authsrv#
```

At this point, you can run any command shown in table 10.16. To add a user, use both the user name and the long name with the command:

```
authsrv# adduser chrish "Chris Hare"
ok - user added initially disabled
authsrv#
```

Notice that the user, although added, is initially disabled. No password is associated with the user. At this point, you need to set a password for the user, and specify the group to which the user belongs.

```
authsrv# password chrish whisper
Password for chrish changed.
authsrv# group chrish production
set group
authsrv#
```

Now that the password and group membership are changed, identify the authentication protocol that will be used for this user. Available protocols depend on the protocols that were compiled when authsrv was built.

```
authsrv# proto chrish plaintext
Unknown protocol "plaintext", use one of: none password
authsrv# proto chrish password
changed
authsrv# enable chrish
enabled
authsrv#
```

If an unknown protocol is used when you set the protocol type, authsrv lists the available authentication protocols. In this instance, the only options available are none and password. After the authentication protocol is set, the user chrish is enabled. At this point, the user chrish can authenticate him- or herself using the authentication server.

Before you give the user free rein, however, establish for this user the wizard for group administrator privileges, and superwiz, which grants global administrator privileges. Normally this wouldn't be done because global administrative privileges supersede the privileges of the group administrator.

```
authsrv# wiz chrish
set group-wizard
authsrv# superwiz chrish
set wizard
```

With these additional privileges set, you can list the information from the authsrv database using the list command.

```
authsrv# list
Report for users in database
user       group      longname      status proto      last
----       -----      --------      ------ -----      ----
chrish     production Chris Hare      y G  passw      never
authsrv#
```

This output shows the user name, the group that the user belongs to, the long name, the status flags, authentication protocol, and when the user last authenticated. The status field includes the following information:

Letter	*Description*
b	Account locked due to too many failed logins
n	Account disabled
y	Account enabled
G	Group Wizard flag set
W	Global Wizard flag set

The list command displays information for all the users; the display command shows more information for a given user.

```
authsrv# display chrish
Report for user chrish, group production (Chris Hare)
Authentication protocol: password
Flags: WIZARD GROUP-WIZARD
authsrv#
```

As you can see, this command provides information similar to the list command, but includes a text explanation of the flags set for this user.

As many users as needed can be added in this manner, although you can see that this is a tedious job for even a small organization.

The Authentication Shell—authmgr

The authsrv command enables a local user access to the firewall host to manipulate the database; the authmgr program also allows users to manipulate the database such access, but from a trusted host on the network or through the local host. Unlike the authsrv command, the authmgr program requires that the user log in to authenticate him- or herself to the database. If the user is not enabled or in the database, the connection is refused. Here is a short authmgr session.

```
pc# ./authmgr
Connected to server
authmgr-> login
Username: admin
Password:
Logged in
authmgr-> list
```

```
Report for users in database
user       group      longname      status  proto      last
----       -----      --------      ------  -----      ----
paulp      copy                       n G   passw      never
chrish     production Chris Hare      y W   passw      never
admin      manager    Auth DBA        y W   passw      Fri Oct 27 23:47:04 1995
authmgr-> quit
pc#
```

All the commands and functionality that are part of the authsrv command are also part of authmgr. This may be apparent, but keep in mind that the authmgr command actually established a TCP session to the authsrv program.

Database Management

Two more commands are available for manipulating the authentication database: authload and authdump. The authload command manipulates individual records in the database; it does not truncate an existing database. It is useful when you need to add several new entries to the database, or when you need to share databases between sites. If you have users who share similar information between sites, the existing records will be overwritten with newer information when this information is loaded by the authload command.

The authdump command creates an ASCII backup copy of the information in the database. This ASCII copy contains all the information regarding the user account. The passwords however, are encrypted, so that they cannot be read and used to circumvent the security provided by the Toolkit.

The authdump command reads the contents of the authentication database and writes the ASCII text. A sample execution of the command is here:

```
pc# ./authdump

user=chrish
longname=Chris Hare
group=production
pass=cY8IDuONJDQRA
flags=2
bad_count=0
proto=p
last=0

user=admin
longname=Auth DBA
group=manager
pass=tx6mxx/lUy2Mw
flags=2
```

If the command is executed and the output is redirected to a file, the program prints a dot for each record dumped, along with a report of the total records processed:

```
pc# ./authdump > /tmp/auth
...
3 records dumped
pc#
```

If you have this information stored somewhere else in a human-readable form (except for the passwords), you can re-create the user database if the firewall ever needs to be rebuilt.

The authload program can take the output of the authdump program and reload the database. The authload command is valuable if the user database was destroyed, or you have a large number of users to add at once. In this manner, new records can be added to the ASCII file and only the new records will be loaded into the authentication database. Consider the new entry added to this ASCII dump file:

```
user=terrih
longname=Terri Hare
group=production
pass=
flags=0
bad_count=0
proto=p
last=
```

Now you can load the records into the database, using input redirection because the information is in the ASCII dump file:

```
pc# ./authload < /tmp/auth
....
4 records loaded
pc#
```

This results in a report showing the number of records that have been loaded. You can then verify the status of the additional records using the authmgr "list" command:

```
pc# ./authmgr
Connected to server
authmgr-> login
Username: admin
Password:
Logged in
authmgr-> list
Report for users in database
user       group      longname      status proto      last
----       -----      --------      ------ -----      ----
paulp      copy                       n G  passw      never
terrih     production Terri Hare      y    passw      never
chrish     production Chris Hare      y W  passw      never
admin      manager    Auth DBA        y W  passw      Sat Oct 28 01:45:32 1995
authmgr->
```

At this point, it is important to note that the new account terrih is enabled, but there is no password. A password should be assigned as quickly as possible to prevent fraudulent use of the firewall, and potential loss of security of the network.

As an added measure of safety, it is advised to add a line to root's crontab to make "backups" of the authentication database. The following shows a sample entry:

```
0    1    *    *    *    /usr/local/etc/authdump > /usr/local/etc/auth.backup
```

The cron command will run the authdump command at 1:00 AM, every morning. This ensures a reliable backup of your database in ASCII format. If the information on your server does not change very often, you probably should adjust the timing of the cron execution of authdump.

Authentication at Work

You might now be interested in seeing how the authentication server operates. Each of the proxies has the option of being configured to operate with the authentication server. The example shown here focuses on the FTP proxy. The FTP proxy's configuration can be found in the earlier section "Configuring the FTP Gateway."

```
ftp-gw: denial-msg      /usr/local/etc/ftp-deny.txt
ftp-gw: welcome-msg     /usr/local/etc/ftp-welcome.txt
ftp-gw: help-msg        /usr/local/etc/ftp-help.txt
ftp-gw: authserver      localhost       7777
ftp-gw: timeout         3600
ftp-gw: permit-hosts    206.116.65.*    -log { retr stor } -auth { stor }
```

Recall from earlier discussions that the last line of this configuration is actually what causes the authentication to be performed. In fact, it is fairly specific in that any request to retrieve a file from the remote, or to store a file on the remote results in that operation being logged by the proxy. In addition, the store command to save a file on the remote system is not permitted until the user authenticates him- or herself to the proxy. This process is illustrated here:

```
pc# ftp pc
Connected to pc.unilabs.org.
220-Welcome to the URG Firewall FTP Proxy
220-
220-To report problems, please contact Network Security Services at 555-1212
or 220-by e-mail at security@org.com
220
Name (pc.unilabs.org:chrish): chrish@nds.fonorola.net
331-(----GATEWAY CONNECTED TO nds.fonorola.net----)
331-(220 nds.fonorola.net FTP server (Version A) ready.)
331 Password required for chrish.
```

```
Password:
230 User chrish logged in.
Remote system type is Unix.
Using binary mode to transfer files.
ftp> put /tmp/trace
local: /tmp/trace remote: /tmp/trace
200 PORT command successful.
500 command requires user authentication
ftp> quote authorize chrish
331 Enter authentication password for chrish
ftp> quote response whisper
230 User authenticated to proxy
ftp> put /tmp/trace
local: /tmp/trace remote: /tmp/trace
200 PORT command successful.
150 Opening BINARY mode data connection for /tmp/trace.
226 Transfer complete.
2181 bytes sent in 0.0061 seconds (3.5e+02 Kbytes/s)
ftp> quit
221 Goodbye.
```

For FTP clients that do not know which proxy is used for authentication, the ftp quote command must be used to "speak" with the authentication server on the firewall. During this process, the password that is submitted by the user is echoed on-screen, and is therefore visible to anyone in the immediate vicinity.

This is just one example of authentication use with proxies; countless more examples could be used. Hopefully, the information and examples you have seen so far on proxies and the authentication server should help you design a secure firewall.

USING PLUG-GW FOR OTHER SERVICES

The applications you have read about so far cover about 80 percent of the network traffic. What about TIS Toolkit support for the Network News Transport Protocol (NNTP) or even the Post Office Protocol (POP)? Both of these services, and many others, are available through the plug-gw application. This application provides plugboard type connections; that is, it connects a TCP/IP port on the firewall to another host using the same or a different TCP port number. This functionality makes it easy to provide other services through the firewall. The next few sections examine the operation and configuration of plug-gw by looking specifically at their services.

Configuring plug-gw

plug-gw reads the configuration lines that start with plug-gw: from the netperm-table file—just like the other Toolkit applications. The clauses listed in table 10.17 are used with the plug-gw application.

Table 10.17
plug-gw Rules and Clauses

Rule	*Description*
timeout seconds	Specifies a timeout value, after which inactive connections are disconnected. If no timeout is specified, the default is to remain connected until one side or the other closes its connection.
port portid host pattern [options]	Specifies a connection rule. When a connection is made, a match is searched for on the portid and calling host. The portid may be either a numeric value (such as 119) or a value from /etc/services (such as "nntp"). If the calling port matches, then the host pattern is checked for a match following the standard address matching rules employed by the firewall. If the rule matches, the connection will be made based on the remaining options in the rule, all of which begin with "-".
-plug-to host	Specifies the name or address of the host to connect to. This option is mandatory.
-privport	Indicates that a reserved port number should be used when connecting. Reserved port numbers must be specified for protocols, such as rlogin, which rely on them for "security."
-port portid	Specifies a different port. The default port is the same as the port used by the incoming connection.

The purpose of plug-gw is to allow for other services to be passed through the firewall with additional logging to track the use of these services. The availability of this service means that additional service-specific applications do not need to be created unless required. Some applications do not have extended authentication mechanisms in them; plug-gw makes their use with firewalls much less of a bother.

The rules available for plug-gw, when used on a POP connection, look like this:

```
plug-gw:        port 110 206.116.65.* -plug-to 198.53.64.14
```

This line indicates that any connection received on port 110 (Post Office Protocol) from the 206.116.65 network is to be connected to 198.53.64.14. Additional options for the rule allow for the specification of a priveleged port number. Few services actually require these. The final option allows for the specification of an alternate port number should the same service be running on a different port number at the remote end.

As with the other services, the host pattern that is specified with the port command allows for both the allowed and non-allowed network or host IP addresses to be specified.

plug-gw and NNTP

The NNTP news protocol is used for reading Internet newsgroups. This protocol also performs news feeds and is often used to provide news reading services at the workstation level. The configuration of the plug-gw proxy for an Internet news feed is essentially the same as the configuration for a news reader.

In both cases, the NNTP port is defined in the etc/services file as 119. You must configure the plug-gw line as follows:

```
plug-gw:        port 119 206.116.65.* -plug-to 198.53.64.1
```

This means that any connections received on port 119 from the local LAN will be directed to the same port on the system at 198.53.64.1.The two major reasons for handling NNTP with plug-gw are to allow NNTP client access through the firewall, and to allow for a newsfeed.

For the firewall to accept news connections, inetd must be configured to start the plug-gw application whenever a connection request is made for the NNTP port. This is done by adding the following line to the /etc/inetd.conf file and restarting inetd:

```
nntp    stream  tcp     nowait  root    /usr/local/etc/plug-gw  plug-gw 119
```

If you configure plug-gw but forget this step, the TIS firewall Toolkit will seem to not operate—no log messages will print to the files or to the console.

To configure an NNTP client, such as WinVN for the PC-based architecture, you must set up WinVN so that it knows where to connect. Normally, this would be the actual NNTP server that you want to access, but in this case, it is the name or IP address of the firewall. On the firewall, the appropriate line in the netperm-table file must be included to specify where the NNTP client requests are to go. If several NNTP servers are available for reading news, you may want to separate them onto different network ports on the firewall, so that traffic can be sent to the different sites. Consider this sample part of the netperm-table file:

```
plug-gw:     port 2119 206.116.65.* -plug-to 198.53.64.1 -port 119
plug-gw:     port 2120 206.116.65.* -plug-to 198.53.64.5 -port 119
```

In this scenario, when users want to read news from the 198.53.64.5 server, they must connect to the firewall on port 2120. Figure 10.3 illustrates the configuration of the WinVN client for access to news through the firewall.

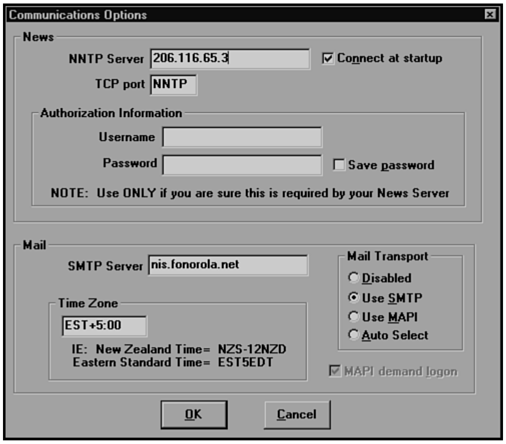


FIGURE 10.3

Configuring WinVN to use the NNTP proxy.

Regardless of the news reader client software that you use, it needs to be configured to use the firewall as the connection point or news host.

What if different news servers are available that your hosts are permitted to connect to? How does the system administrator configure multiple hosts at the same TCP/IP service port? The answer is to specify a different port on the firewall, and let plug-gw redirect to the correct port on the remote system. This is done by using a rule in the nbetperm-table file:

```
plug-gw:     port 2120 206.116.65.* -plug-to 198.53.64.5 -port 119
```

According to this command, if a connection on port 2120 is requested, redirect that request on port 119 or the host at 198.53.64.5. This is only part of the solution. The /etc/services file should also be edited to add a news NNTP service entry to show the new service port for this connection. For example, the following line specifies that the service nntp-a is on port 2120:

```
nntp-a     2120/tcp              readnews untp   # USENET News Transfer Protocol
```

The next step is to tell inetd that connections on this port are to be sent to the plug-gw application. This is done by adding the following line to the /etc/inetd.conf file and restarting inetd.

```
nntp-a  stream  tcp    nowait  root    /usr/local/etc/plug-gw  plug-gw 2120
```

When the user wants to use this alternate server, he or she must reconfigure the news client software, as shown in figure 10.4, to point to the new services port.

FIGURE 10.4

Configuring WinVN and NNTP.

Although you can set up your firewall so that NNTP clients can read news, this is generally not a popular setup. A much more realistic configuration would be for the clients to interact with a local news server. This configuration requires the firewall to allow for a news feed to be passed through to the internal news server.

To do this, the external news server and the internal news client must be set up so that they pass their information through the firewall. The trick is understanding what configuration information must be placed in the news server configuration files on both ends. For the purpose of this discussion, assume that the news server software in use is INN 1.4. The file hosts.nntp provides information regarding what hosts are permitted to connect to the INN NNTP server. Consider the news server and firewall configuration shown in figure 10.5.

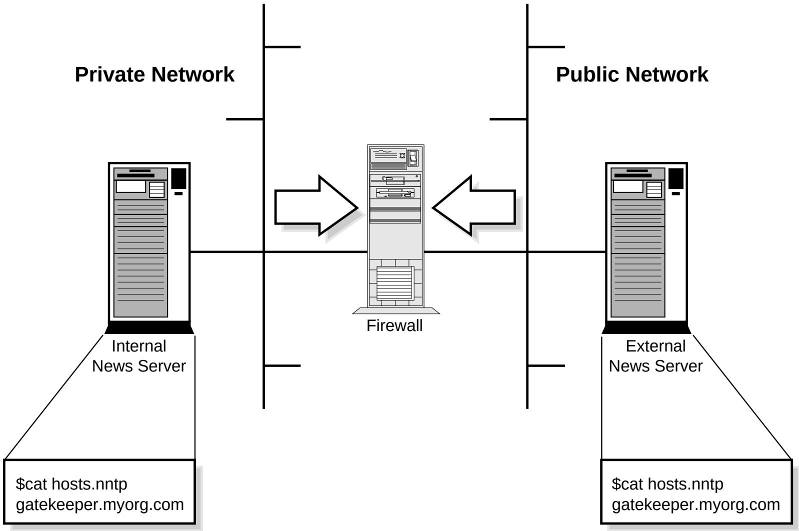


Figure 10.5
News client and server.

Normally, the hosts.nntp file on each news server contains the name or IP address of the other news server that is allowed to connect to it. In this case, the name of the machine that goes in both hosts.nntp files is in fact the name or IP address of the firewall. This is because the firewall actually establishes a connection from one network to the other, and from one server to the other using the correct service port. With the hosts.nntp file correctly configured, there will be no problems passing news through the firewall.

plug-gw and POP

When you first think about using plug-gw with the TIS plug-gw application, the obvious question that comes to mind is "How do I configure things for authentication?" The trick is to remember which machine is actually performing the authentication. The firewall using plug-gw does no authentication. It merely accepts the incoming connection on the named port, and establishes a connection from itself to the named system on the same or different port.

To see this in operation, you can establish a Telnet connection to the POP port. Consider the sample output shown here:

```
$ telnet 206.116.65.3 110
+OK UCB Pop server (version 2.1.2-R3) at 198.53.64.14 starting.
USER chrish
+OK Password required for chrish.
PASS agdfer
```

```
+OK chrish has 0 message(s) (0 octets).
QUIT
Connection closed by foreign host.
$
```

Notice that the connection to the firewall was established at 206.116.65.3. The remote system [198.53.64.14] does not normally list its IP address in the output; a modified version of the POP server was used to show the IP instead of the name.

Unfortunately, simply adding the entries to the netpwrm-table file is not enough. Like NNTP, inetd must be configured to accept connections on the POP service port, 110. This is done by adding the following line to the /etc/inetd.conf file and restarting inetd:

```
pop     stream  tcp     nowait  root    /usr/local/etc/plug-gw  plug-gw 110
```

With the firewall now accepting POP service requests, plug-gw must be configured to redirect those POP requests to the appropriate server. This is done by adding this next line to the netperm-table file:

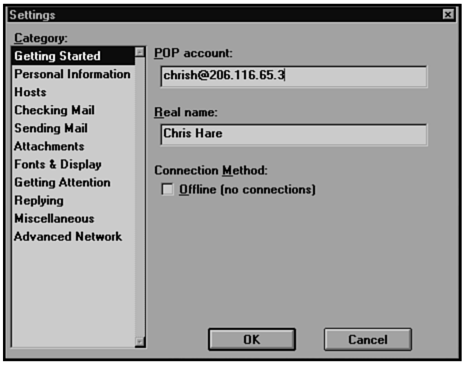
```
plug-gw:        port 110 206.116.65.* -plug-to 198.53.64.14
```

After it is added, POP service requests received by the firewall are redirected to the specified server.

The preceding example shows the process of establishing a POP session using Telnet, but how do you configure a workstation that relies on POP to pass traffic through the firewall? Figure 10.6 shows a configuration screen from the Eudora 1.52 shareware e-mail package.

FIGURE 10.6

Setup for a POP e-mail package.



In this example, the user@hostname specification for the POP server identifies the real user name, but specifies the IP address for the firewall. The IP or name of the firewall can be used interchangeably in this field. The only reason for using the IP address rather than the name is if you have a DNS reliability problem, or to ensure that you connect to the correct host.

Consequently, when the incoming connection is received on port 110, plug-gw starts a session to the remote host specified in the plug-gw rule. This results in the mail being transferred from the remote end through the firewall to the workstation.

Incidentally, the POP mail client in use is irrelevant. The plug-gw configuration has been tested with Eudora, Microsoft Exchange, and Pegasus Mail; every package tested functions properly.

THE COMPANION ADMINISTRATIVE TOOLS

A set of support tools are included with the TIS Toolkit to assist in the setup and ongoing administration of the firewall. These include a port scanner, a network subnet ping manager, and log analysis and reporting tools.

> Depending upon the version and completeness of the Toolkit you downloaded, some services and programs may not be installed or compiled automatically. It is strongly suggested that you retrieve the lastest version and patches directly from the the TIS FTP site.

NOTE

PORTSCAN

The portscan program attempts to connect to every TCP port on a given machine. The default operation is to connect to each port in sequence on the named host/. The portscan program's scan of the machine pc.unilabs.org, for example, was answered by the following ports:

```
pc# ./portscan pc.unilabs.org
ftp
telnet
gopher
httpd
pop
nntp
who
2021
2023
2120
7777
pc#
```

You can see from the output of portscan that very few ports are in fact in operation on the machine that was contacted.

NETSCAN

This is a network ping program. It accepts as an argument a network address and starts to ping each address on the network. Its default output is a list of each of the addresses that responded to the ping, along with the host's name. The use of netscan in default mode is shown in this example:

```
pc# ./netscan 198.53.32
198.53.32.5
Vaxxine-GW.Toronto.fonorola.net (198.53.32.6)
198.53.32.9
Harte-Lyne-gw.Toronto.fonorola.net (198.53.32.10)
198.53.32.13
Globe-n-Mail-GW.Toronto.fonorola.net (198.53.32.14)
^C
pc#
```

This output shows that the first host that responded to a ping was 198.53.32.5. Notice that even though the program pings each address in turn, there is not always a response. This indicates that either no device exists, or netscan attempted to contact a device that does not respond to pings.

A verbose mode is also available with netscan. In verbose mode, addresses that respond to a ping are placed with their name or address flush left; addresses that did not respond are indented one tab space. This mode is enabled by using the -v option on the command line:

```
pc# ./netscan -v 198.53.32
trying subnet 198.53.32
    198.53.32.1
    198.53.32.2
    198.53.32.3
    198.53.32.4
198.53.32.5
Vaxxine-GW.Toronto.fonorola.net (198.53.32.6)
    198.53.32.7
    198.53.32.8
198.53.32.9
Harte-Lyne-gw.Toronto.fonorola.net (198.53.32.10)
    198.53.32.11
    198.53.32.12
198.53.32.13
^C
pc#
```

This tool helps determine what hosts are on a network, which may affect how you specify the configuration rules for your network.

Reporting Tools

The TIS Toolkit, configured as a firewall, logs transactions and requests processed by Toolkit applications, and records the outcome of these requests. The log file messages are recorded through the syslog daemon. The files used to save the details are listed in /etc/syslog.conf, and vary from system to system. The TIS Toolkit applications all interact with the syslog service and send logging information and status messages for the lifetime of the connection.

You can periodically peruse the log files, or use the reporting programs included with the Toolkit to search out and report usage of the firewall. Because the logging is performed using the syslogd service, the log messages observe the standard format:

```
Date Time hostname program[PID]: message
```

This format appears in the log file looking like this:

```
Oct  4 02:42:14 pc ftp-gw[1763]: permit host=stargazer.unilabs.org/
➥204.191.3.147 use of gateway
```

A wide variety of log messages can be displayed in the syslog file. Some of these are illustrated in the following output:

```
cannot connect to server 198.53.64.14/110: No route to host
cannot connect to server 198.53.64.14/110: Operation timed out
cannot connect to server nis.fonorola.net/110: Connection refused
cannot connect to server nis.fonorola.net/110: Operation timed out
cannot get our port
connect host=stargazer.unilabs.org/206.116.65.2 destination=198.53.64.14/110
connect host=unknown/206.116.65.2 destination=198.53.64.14/110
connected host=pc.unilabs.org/204.191.3.150 to nds.fonorola.net
content-type= multipart/x-mixed-replace;boundary=ThisRandomString
content-type= text/html
deny host=204.191.3.150/pc.unilabs.org connect to fox.nstn.ca
deny host=pc.unilabs.org/204.191.3.150 service=ftpd
deny host=stargazer.unilabs.org/204.191.3.147 destination=sco.sco.com
deny host=unknown/206.116.65.2 service=110
disconnect host=stargazer.unilabs.org/206.116.65.2 destination=198.53.64.14/
➥110 in=3512 out=92 duration=8
disconnect host=unknown/206.116.65.2 destination=198.53.64.14/110 in=0 out=0
➥duration=75
exit host=pc.unilabs.org/204.191.3.150 dest= in=0 out=0
exit host=pc.unilabs.org/204.191.3.150 dest= in=0 out=0 user=unauth
➥duration=2
exit host=pc.unilabs.org/204.191.3.150 dest=nds.fonorola.net in=35 out=21
➥user=unauth duration=37
exit host=pc.unilabs.org/204.191.3.150 dest=none in=0 out=0 user=unauth
➥duration=14
```

```
exit host=stargazer.unilabs.org/204.191.3.147 cmds=1 in=0 out=0 user=unauth
➥duration=2
exit host=stargazer.unilabs.org/204.191.3.147 no auth
failed to append to file (null)
failed to connect to http server iback.gif (80)
fwtksyserr: cannot display denial-msg /usr/local/etc/tn-deny.txt: No such
➥file or directory
fwtksyserr: cannot display help file /usr/local/etc/tn-help.txt: No such file
➥or directory
fwtksyserr: cannot display help message /usr/local/etc/rlogin-help.txt: No
➥such file or directory
fwtksyserr: cannot display welcome /usr/local/etc/rlogin-welcome.txt: No such
➥file or directory
fwtksyserr: cannot display welcome /usr/local/etc/tn-welcome.txt: No such
➥file or directory
log host=stargazer.unilabs.org/206.116.65.2 protocol=HTTP cmd=dir
➥dest=www.istar.ca path=/
log host=stargazer.unilabs.org/206.116.65.2 protocol=HTTP cmd=dir
➥dest=iback.gif path=/
log host=stargazer.unilabs.org/206.116.65.2 protocol=HTTP cmd=get dest=
➥www.nstn.ca path=/cgi-bin/test/tide.cgi
Network connection closed during write
permit host=pc.unilabs.org/204.191.3.150 connect to 204.191.124.252
permit host=pc.unilabs.org/204.191.3.150 connect to chrish@nds.fonorola.net
permit host=pc.unilabs.org/204.191.3.150 use of gateway
permit host=stargazer.unilabs.org/204.191.3.147 connect to mail.fonorola.net
permit host=stargazer.unilabs.org/204.191.3.147 destination=204.191.3.150
permit host=stargazer.unilabs.org/204.191.3.147 service=ftpd execute=/usr/
➥libexec/ftpd
permit host=stargazer.unilabs.org/204.191.3.147 service=ftpd execute=/bin/cat
permit host=stargazer.unilabs.org/204.191.3.147 service=telnetdexecute=/usr/
libexec/telnetd
permit host=stargazer.unilabs.org/204.191.3.147 use of gateway
permit host=stargazer.unilabs.org/206.116.65.2 use of gateway (Ver p1.4 /  1)
```

These log messages do not represent a complete list. The only way to see a complete list of possible log messages and their exact meanings is to perform a line-by-line review of the TIS Toolkit code, and then document each item individually.

The Toolkit includes a number of reporting tools that can be used to analyze the log records saved by the syslog service. These shell scripts, listed in table 10.18, are in the fwtk/tool/admin/reporting directory.

Table 10.18
syslog Report Generating Scripts

Script Name	*Description*
authsrv-summ.sh	Summarizes auth server reports
daily-report.sh	Runs the report scripts on a daily basis
deny-sum.sh	Reports on denial of services
ftp-summ.sh	Summarizes ftp-gw traffic
http-summ.sh	Summarizes the http-gw traffic
netacl-summ.sh	Summarizes netacl accesses
smap-summ.sh	Summarizes smap email records
tn-gw-summ.sh	Summarizes tn-gw and rlogin-gw traffic
weekly-report.sh	Top-level driver that calls each summary report generator

The reporting tools included in the TIS Toolkit are not installed automatically when the Toolkit applications are compiled and installed. They must be installed later by changing to the directory tools.admin.reporting and running the make install command. This copies all the files to the same directory in which the Toolkit applications were copied.

The Authentication Server Report

The authentication server report identifies various authentication operations that are carried out on the server. A typical report of authsrv-summ.sh looks like this:

```
pc# ./authsrv-summ.sh < /var/log/messages.0

Top 100 permitted user authentications (total: 6)
Logins        User ID
------        -------
4             admin
2             chrish

Top 100 failed user authentications (total: 2)
Attempts      Username
--------      --------
1             paulp
1             chrish
```

```
Authentication Management Operations
---------------------------------------
administrator ADDED admin
administrator ADDED admin
administrator ADDED chrish
administrator ADDED chrish
administrator ADDED paulp
administrator DELETED admin
administrator DELETED chrish
administrator ENABLED admin
administrator ENABLED chrish
administrator GROUP admin manager
administrator GROUP chrish production
administrator GROUP paulp copy
administrator GWIZ chrish
administrator GWIZ chrish
administrator GWIZ paulp
administrator PASSWORD admin
administrator PASSWORD chrish
administrator PROTOCOL admin
administrator PROTOCOL chrish
administrator UN-GWIZ chrish
administrator WIZ admin
administrator WIZ chrish
```

Notice that this and all the other reporting tools expect to read their data from the standard input stream. These reporting tools can do this by using the cat command with a pipe, or by redirecting the input stream from the log file.

The authsrv summary report lists the total authentication requests made and by whom, the denied authentication, and the authentication database management operations. If you run this report after a heavy period of user administration, it will be quite verbose.

The Service Denial Report

The purpose of the service denial report is to identify hosts that attempted to connect through the firewall and were not permitted. The report reads through the specified log file and reports on:

- The top 100 network service users
- The top 100 denied service users
- The total service requests by service

A sample execution of deny-summ.sh looks like this:

```
pc# ./deny-summ.sh < /var/log/messages.0
```

```
Authentication Failures
Failures      Proxy: Host - ID
--------      ----------------
1             s: disable - paulp
1             ftp-gw: pc.unilabs.org/206.116.65.3 - chrish

Top 100 network service users (total: 152)
Connects      Host/Address
--------      ------------
120           stargazer.unilabs.org/206.116.65.2:
11            pc.unilabs.org/206.116.65.3:ftp
5             stargazer.unilabs.org/206.116.65.2:telnet
3             stargazer.unilabs.org/206.116.65.2:telnetd
3             stargazer.unilabs.org/206.116.65.2:ftpd
3             pc.unilabs.org/206.116.65.3:telnet
2             stargazer.unilabs.org/206.116.65.2:ftp
2             pc.unilabs.org/206.116.65.3:
1             unknown/206.116.65.2:
1             pc.unilabs.org/206.116.65.3:telnetd
1             pc.unilabs.org/206.116.65.3:ftpd

Top 100 Denied network service users (total: 12)
Connects      Host/Address
--------      ------------
2             stargazer.unilabs.org/206.116.65.2:telnet
2             pc.unilabs.org/206.116.65.3:ftp
1             unknown/206.116.65.2:110
1             stargazer.unilabs.org/206.116.65.2:telnetd
1             stargazer.unilabs.org/206.116.65.2:110
1             stargazer.unilabs.org/206.116.65.2:
1             pc.unilabs.org/206.116.65.3:2120
1             pc.unilabs.org/206.116.65.3:119
1             pc.unilabs.org/206.116.65.3:110
1             pc.unilabs.org/206.116.65.3:

Service Requests
Requests      Service
--------      -------
125
15            ftp
10            telnet
5             telnetd
4             ftpd
3             110
1             2120
1             119
```

The report can be used to highlight sites that have attempted unauthorized connections to the firewall; the report also highlights sites that are authorized to connect, but whose users do not know how, or have forgotten their passwords. All of these examples may be legitimate problems or potential security breaches.

The FTP Usage Report

The FTP usage report identifies sites that are connected to FTP services through the firewall. It identifies the number of connections, the origin of the connection, and the amount of data transferred. A sample execution of ftp-summ.sh looks like this:

```
pc# cat /var/log/messages* ¦ ./ftp-summ.sh
FTP service users (total: 23)
Connects      Host/Address
--------      ------------
13            stargazer. unilabs.org/204.191.3.147
5             pc.unilabs.org/206.116.65.3
3             pc.unilabs.org/204.191.3.150
2             stargazer.unilabs.org/206.116.65.2

Denied FTP service users (total: 4)
Connects      Host/Address
--------      ------------
2             pc.unilabs.org/206.116.65.3
2             nds.fonorola.net/204.191.124.252

FTP service output thruput (total Kbytes: 6)
KBytes        Host/Address
------        ------------
6             pc.unilabs.org/206.116.65.3

FTP service input thruput (total Kbytes: 4)
KBytes        Host/Address
------        ------------
3             pc.unilabs.org/206.116.65.3
0             stargazer.unilabs.org/206.116.65.2
0             stargazer.unilabs.org/204.191.3.147
pc#
```

As you can see in this report, several service denials occurred on this firewall. A couple came from an external site, but also an internal host attempted to access the site. Many sites choose to not allow FTP at all because of the potential problems associated with pirated software or virus-infected software.

The HTTP Usage Report

The HTTP usage report identifies traffic that has been passed through the http-gw application. The report covers connection requests, denied service requests, and input and output through the proxy. A sample HTTP usage report looks like this:

```
pc#   cat /var/log/messages* ¦ ./http-summ.sh
HTTP service users (total: 130)
Connects      Host/Address
--------      -----------
127           stargazer.unilabs.org/206.116.65.2
2             pc.unilabs.org/206.116.65.3
1             unknown/206.116.65.2
Denied HTTP service users (total: 1)
Connects      Host/Address
--------      -----------
1             stargazer.unilabs.org/206.116.65.2

HTTP service output thruput (total Kbytes: 1)
KBytes        Host/Address
------        -----------
1             stargazer.unilabs.org/206.116.65.2

HTTP service input thruput (total Kbytes: 315)
KBytes        Host/Address
------        -----------
315           stargazer.unilabs.org/206.116.65.2
pc#
```

A few requests out through the firewall may result in a much higher rate of information input to the firewall. You can see this in list 4; 1 KB of data out through the firewall resulted in 315 KB from the remote end.

THE NETACL REPORT

Recall that netacl is a method of allowing access to the services on the firewall itself, such as Telnet. This program enables administrators and other users to operate directly on the firewall without the need to be on the console.

The netacl report identifies the connects that have been made to the firewall and on what services, as well as the origin of the requests. A sample execution of the netacl-summ.sh command is shown here:

```
pc# cat /var/log/messages* ¦ ./netacl-summ.sh
Top 100 network service users (total: 40)
Connects      Host/Address
--------      -----------
19            stargazer.unilabs.org/204.191.3.147
13            stargazer.unilabs.org/206.116.65.2
4             unknown/206.116.65.2
2             unknown/204.191.3.147
2             pc.unilabs.org/206.116.65.3
```

```
Top 100 Denied network service users (total: 11)
Connects      Host/Address
--------      ------------
6             pc.unilabs.org/204.191.3.150
2             stargazer.unilabs.org/204.191.3.147
1             stargazer.unilabs.org/206.116.65.2
1             nds.fonorola.net/204.191.124.252
1             mail.fonorola.net/198.53.64.8

Service Requests
Requests      Service
--------      ------
32            ftpd
18            telnetd
```

In a previous section in this chapter, only Telnet and FTP services were configured to be available with netacl. This setup was chosen so that you, the network administrator, could update files and interact with the firewall from places other than the console. The denied requests result from other hosts attempting to connect to your netacl ports (Telnet was 2023, and FTP was 2021).

This report identifies sites that are attempting to log in or ftp directly to the firewall itself, rather than log in to a site behind the firewall.

The Mail Usage Report

Another important piece of information for the administrator is knowing how much mail is flowing through the firewall. Many sites do not allow any traffic other than mail through the firewall; for this reason, knowledge of the amount of information available helps determine whether the chosen hardware platform is in fact doing the job. The mail usage report generator identifies for the administrator the number of messages received per user, and how many bytes in mail traffic were handled by the firewall.

The following sample execution of the mail report, smap-summ.sh, illustrates this script's importance:

```
pc# cat /var/log/messages* ¦ ./smap-summ.sh
Total messages: 10  (22 Kb)

Top 100 mail recipients (in messages)
Messages
 Count      Kb    Address
 -----      --    -------
     2      7.6   skhan@compmore.net
     2      7.6   chrish
     2      2.9   74507.3713@compuserve.com
```

```
     1      1.5   chrish@fonorola.net
     1      1.1   chrish@unilabs.org
     1      0.9   denny@nstn.ca
     1      0.9   chrish@nds.fonorola.net

Top 100 mail senders (in messages)
Messages
 Count      Kb    Address
 ----       --    -------
     9      21.4  chrish@unilabs.org
     1      1.1   news@news.compmore.net

Top 100 mail recipients (in kilobytes)
Messages
 Count      Kb    Address
 -----      --    -------
     2      7.6   skhan@compmore.net
     2      7.6   chrish
     2      2.9   74507.3713@compuserve.com
     1      1.5   chrish@fonorola.net
     1      1.1   chrish@unilabs.org
     1      0.9   denny@nstn.ca
     1      0.9   chrish@nds.fonorola.net

Top 100 mail senders (in kilobytes)
Messages
 Count      Kb    Address
 -----      --    -------
     9     21.4   chrish@unilabs.org
     1      1.1   news@news.compmore.net
```

The Telnet and rlogin Usage Report

The Telnet and rlogin usage report (tn-gw-summ.sh) combines activity through the firewall of the Telnet and rlogin services. This report identifies the following:

- The number of connections
- The connecting host
- Characters input to the firewall for transmission to the public network
- Characters received by the firewall for the private network
- Denied connections

The following report provides a sample execution of tn-gw-summ.sh:

```
Top 100 telnet gateway clients (total: 43)
Connects      Host/Address            Input    Output     Total
--------      ------------            -----    ------     -----
    17        stargazer.unilabs.or      924       177      1101
    16        pc.unilabs.org/204.1    97325      1243     98568
     3        stargazer.unilabs.or      274         6       280
     3        mailhost.unilabs.org    26771       717     27488
     2        unknown/204.191.3.14    27271       710     27981
     1        unknown/206.116.65.4    10493       701     11194
     1        pc.unilabs.org/206.1        0         0         0

Top 100 telnet gateway clients in terms of traffic
Connects      Host/Address            Input    Output     Total
--------      ------------            -----    ------     -----
    16        pc.unilabs.org/204.1    97325      1243     98568
     3        mailhost.unilabs.org    26771       717     27488
     2        unknown/204.191.3.14    27271       710     27981
     1        unknown/206.116.65.4    10493       701     11194
    17        stargazer.unilabs.or      924       177      1101
     3        stargazer.unilabs.or      274         6       280
     1        pc.unilabs.org/206.1        0         0         0

Top 100 Denied telnet gateway clients (total: 20)
Connects      Host/Address
--------      ------------
    14        stargazer.unilabs.or
     2        stargazer.unilabs.or
     2        204.191.3.150/pc.uni
     1        unknown/204.191.3.14
     1        mail.fonorola.net/19
```

This report provides details on who is connecting through the firewall, how much traffic is being generated, and who is being denied. You can see, for example, that stargazer.unilabs.org is in both the connections and denied lists. This may indicate that at one point the site was denied, and then later authorized to use the Telnet or rlogin gateways.

Where to Go for Help

Help with the TIS Toolkit is easy to find. Discussions on general Internet security-related topics can be found in the Usenet newsgroups:

```
alt.2600

alt.security

comp.security
```

You can also find help by joining the mailing list concerned with a general discussion of firewalls and security technology:

```
firewalls@greatcircle.com
```

To subscribe to the mailing list, send a message to

```
majordomo@greatcircle.com
```

with the text

```
subscribe firewalls
```

in the body of the message.

To reach users familiar with the TIS Toolkit applications and their configuration, contact this mailing list:

```
fwall-users-request@tis.com
```

In addition, the TIS Toolkit includes a large amount of documentation on firewalls. If you plan to make significant use of the Toolkit you should join the TIS discussion lists first. Before you commit to an operating system and hardware platform, ask questions on this mailing list; probably many of the list's readers have had similar questions and experiences.

CHAPTER 11

BLACK HOLE

FIREWALL IMPLEMENTATIONS are available today from a wide array of vendors. With the ever-increasing awareness of network security and the costs of lost information, many new firewall implementations continue to emerge. This chapter discusses Black Hole, which is the firewall produced by Milkyway Networks Corporation in Ottawa, Canada. This firewall is currently the only implementation certified at the AL-1 level (refer to Chapter 2, "Security") from the Canadian government's Department of National Defense Communications Security Establishment (DND-CSE).

Black Hole was developed to address the challenge of securing the private network from the public network, more commonly known as the Internet. Black Hole does this by providing the following:

- Full authentication for both incoming and outgoing traffic
- Mail relay services to eliminate the need for sendmail on the firewall for mail delivery
- Real-time alert messaging for faster administrative response

- Statistical traffic processing that logs intruder attempts
- Full network address translation to reduce administrative time and costs
- One-time password schemes that increase user and password security
- A flexible hardware platform and full GUI interface to provide ease of use and customization

This chapter discusses how to protect the systems and information on your private network using Black Hole as your firewall.

Understanding Black Hole

A black hole is by definition a void. Dr. Stephen Hawking describes a black hole in his book *A Brief History of Time* as a compact star with sufficient gravitational force as to prevent any light from escaping, and would therefore not be visible to us (paraphrased here for this book's purposes). This is in fact what Black Hole firewall does. It completely isolates the internal network from the external, blocking all knowledge of the internal network from the external network.

When this firewall is initially installed, it defaults to a totally secure policy of "that which is not explicitly permitted is prohibited." The security manager must determine how to configure Black Hole to satisfy the organization's security policy.

Black Hole is a secure application and circuit-level gateway that is installed between the public and private networks. It utilizes rule-based proxy servers and does not require nor use packet-filtering mechanisms of any kind. Black Hole also can be installed inside an organization if internal networks exist that must be protected against from the remainder of the organization. Black Hole's most common application is that of an Internet firewall; it is capable of offering Virtual Private Network services between multiple Internet-connected sites.

A Virtual Private Network can be constructed between two or more sites with Black Hole and an Internet connection. The Internet is used to provide the network connection, and the VPN capabilities of Black Hole are used to provide an encrypted and secure communications channel between the sites.

Figure 11.1 illustrates several different connection scenarios.

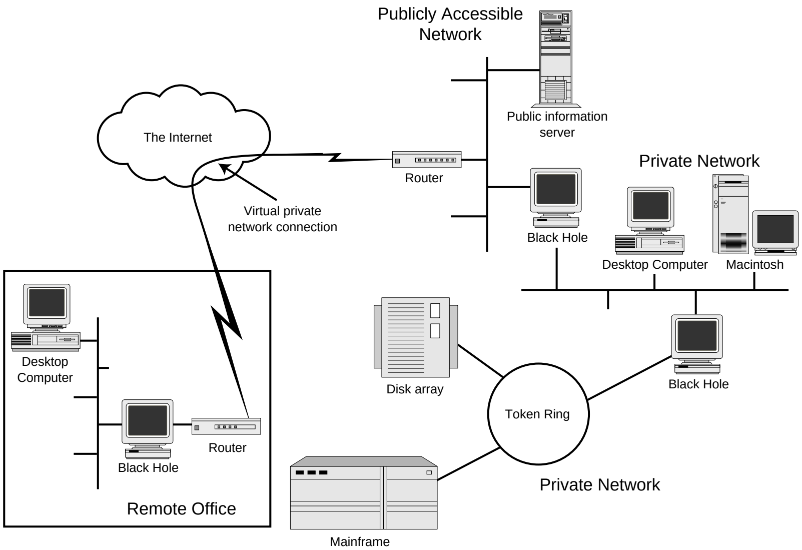


FIGURE 11.1

Sample Black Hole installation points.

Figure 11.1 presents Black Hole in several different configurations, such as providing services for a remote office using a Virtual Private Network scenario, protecting the corporate LAN from the public, and isolating the corporate accounting network from the remainder of the LAN. This variety of configurations demonstrates that Black Hole can be used in an interdepartmental role as well as the more common Internet security role. As an application-level firewall, Black Hole enables the security manager to restrict both inbound and outbound traffic based upon the following:

- Source and destination IP address
- Protocol or application
- User ID
- Time of day

In addition, for the allowed services, Black Hole's operation can be completely transparent to the user. This transparency distinguishes Black Hole from some other firewalls, such as the freely available TIS Firewall Toolkit. Having a transparent operation means that the firewall does not require the user to have special training to access the typical services, which decreases the training time and initial operational problems often associated with the installation of a firewall.

Transparency need not be applied in all cases, however. For example, it may be reasonable to allow certain users or types of traffic through the firewall without authentication. In this case, other types of traffic are restricted to users who can authenticate and are permitted for that service. The extensibility of Black Hole can rely upon the requirement for user authentication; this chapter also addresses the configuration of user accounts in the later section "Services, Users, and Rules."

The implementation of a security policy that makes it harder for the users to do their jobs and complete their assignments is an undesirable situation. Black Hole was designed to protect the systems, information, and users—not to inhibit them.

System Requirements

Black Hole comes in either a SPARC- or PC-based configuration. The SPARC-based configuration consists of a SPARC CPU such as a SPARC-2, SPARC-5, SPARC-10, or SPARC-20. The determining factor on the CPU size is the anticipated amount of traffic that will be running through the firewall. The configuration for the SPARC-based system is as follows:

- SPARC-based CPU (SPARC-2, SPARC-5, SPARC-10, SPARC-20)
- 16 MB of RAM (minimum)
- 500 MB of disk space (minimum)
- Two network interface cards, either Token-Ring or 10-MB Ethernet
- 17-inch or 20-inch high-resolution monitor

Sun Microsystems and other vendors produce other SPARC systems. At this time, however, only systems that are compliant with the previously mentioned SPARC architectures are supported. Although other forms of network interface cards are provided by manufacturers, Token-Ring and 10-MB Ethernet NICs are proven and do not cause network performance bottlenecks. Milkyway is working to support other network interfaces.

The PC-based implementation requires either a 486/66 MHz as a minimum, or a Pentium processor. The configuration for an Intel-based system is as follows:

- Intel 486/66 MHz or Pentium processor
- 16 MB of RAM (minimum)
- 500 MB of disk space (minimum)
- Two network interface cards, Token-Ring or 10-MB Ethernet
- 15-inch or 17-inch high-resolution monitor

If you have chosen to provide your PC-based hardware yourself, you need to remember that the system will be running a modified version of the BDSI implementation of BSD Unix. Consequently, any hardware selections should be verified through Milkyway as supported under the modified BSDI kernel before purchasing.

Black Hole is available in several different sizes depending on the needs of the organization. The determining factors as to which version of Black Hole you should consider are the following:

- The number of licensed users
- The hardware platform
- The number of simultaneous sessions

Table 11.1 illustrates the different configurations of Black Hole.

Table 11.1
Black Hole Software Configurations

Product Number	*Product Name*	*Product Description*	*Simultaneous Sessions*
M-BH-2001	Black Hole	Security-enhanced SUN OS	Unlimited
M-BH-2002	Mid-size Black Hole	Security-enhanced SUN OS	100
M-BH-2003	Entry level Black Hole	Security-enhanced SUN OS	40
M-BH-2006	Sub Entry Black Hole	Security-enhanced SUN OS	5
M-BH-2004	Mini Black Hole	Security-enhanced BSDI OS	Unlimited
M-BH-2005	Macro Black Hole	Security-enhanced BSDI OS	40
M-BH-2007	Sub Micro Black Hole	Security-enhanced BSDI OS	5

In table 11.1, the number of simultaneous sessions is defined as the number of currently active proxy processes. The current number of active proxies is determined by running a `ps` command on a running Black Hole implementation. Consequently, the selection of the product to use becomes a question of how many users are in the organization and how much external network traffic there will be. The more users, or the greater the amount of network traffic, the larger the number of simultaneous sessions that will be required for Black Hole to provide the requested services.

Black Hole Core Modules

Both the SPARC- and PC-based implementations are based on secured versions of SunOS or BSDI Unix respectively. These kernel modifications include the following:

- Disabled source routing to ensure that unauthenticated source-routed packets are not forwarded
- Disabled ICMP redirect functions to ensure that the routing table cannot be modified
- Disabled IP forwarding so that Black Hole cannot be made to function as a router
- Forcing all IP packets arriving on the physical interface to be processed through the TCP/UDP layer for delivery to the proxies
- Monitoring all 64-KB TCP/UDP ports for connections
- Verifying IP packet direction to eliminate the possibility of IP spoofing of the internal network addresses

NOTE

> IP address spoofing occurs when the source address of the packet is based on the internal network address protected by Black Hole. When Black Hole receives a packet on its external network interface with a source address of the internal network, the packet is discarded as being spoofed.

- Disabling X Window, Open Look, and syslog ports on Black Hole to prevent a vandal from using these network ports to make contact with Black Hole

The base Unix system also has been modified to remove as many of the potential threats as possible while still leaving the system in an operable state. The operating system modifications include the following:

- Removal of all redundant applications
- Establishment of static routing
- Removal of route listeners (routed and so on)
- Removal of all unnecessary user accounts
- Removal of all compilers and loaders, with the exception of PERL, which is used for application support

> **Note:** One might question the existence of the PERL interpreter on a firewall with the capabilities of this language. However, the version of PERL provided has been secured and does not provide network sockets support.

- Configuration of services to run in a chroot environment

Black Hole is composed of a collection of management programs that operate the system and apply the rules for each incoming or outgoing packet. The Guardian is the module that offers services similar to the standard inetd super-server. The Guardian performs some rudimentary checking of the incoming requests, examining the source and destination addresses and ports, at which point it either accepts or denies the requested connection.

The Oracle, or Authentication Server, is responsible for implementing the policies and rules configured by the security manager. The result of the Oracle's processing can be one of several scenarios:

- The client is properly authenticated and the connection is allowed to proceed.
- The client has not been properly authenticated.
- The client is not permitted to connect, in which case the connection is terminated.

Black Hole includes a series of proxy agents, or processes that relay information from one side of Black Hole to the other. The proxies are listed in table 11.2.

Table 11.2
Available Proxy Applications

Proxy Name	*Description*
Proxy-telnet	Handles all telnet requests and connections
Proxy-FTP	Handles FTP requests and connections

continues

Table 11.2, Continued
Available Proxy Applications

Proxy Name	*Description*
Proxy-Gopher	Handles Gopher requests and connections
Proxy-HTTP	Handles all HTTP requests and connections
Proxy-TCP	A generic proxy handler that processes TCP traffic not handled by a specific proxy
UDP Relay	A generic proxy handler that establishes a virtual UDP session to the destination and waits for a response before relaying the answer to the requester
NetACL	Permits controlled access to Unix services, such as telnet and FTP on ports 23 and 21 respectively. These services can be used to provide for remote administration of Black Hole itself.

Mail on Black Hole is handled through a specialized mail handler known as DuhMail. DuhMail consists of two components: a mail receiver and a mail sender. The mail receiver is started by Guardian whenever a connection on port 25 is received. The message is collected and saved in a secure area on disk and then transmitted to the remote site by the mail sender.

Black Hole does not run sendmail; therefore, it is not prone to the typical sendmail attacks. Black Hole requires that an SMTP server running sendmail or another SMTP Message Transfer Agent be on the private network to handle the message delivery.

The Black Hole Alarm system constantly reviews the log messages being recorded and watches for patterns specified by the security manager. When a match is found, a predetermined action is taken, which can include sending e-mail messages, ringing the console bell, and calling a pager.

The final core module is the report generator. This module is used to construct statistical reports based on the log file and traffic flow analysis. Once generated, these reports are typically sent via e-mail to the security manager. A printer can be attached directly to the system to generate paper copies of the report if required. The reports available include the following:

- Traffic summary by hour
- Incoming traffic
- Outgoing traffic
- Connections per hour

- Traffic by port
- Top 10 destinations

The report generator is discussed later in this chapter in the section "Generating Reports."

Black Hole Extension Modules

Black Hole supports Virtual Private Networks through additional software plug-in modules. These modules enable two more Black Hole systems to send information in an encrypted data stream over a public network such as the Internet. As of version 3.0, Black Hole supports Virtual Private Networking with public-key encryption. Although the previous version of Black Hole supported VPN using DES encryption, version 3.0 uses public-key encryption.

Public-key encryption is the most common form of encryption on the Internet today. This form of encryption uses a pair of mathematically related numeric keys. One key is private; the other can be known by anyone. When one key is used to encrypt data, only the other key can be used to decrypt the data. Because public-key encryption and decryption are slow (when compared to private-key encryption), Milkyway's VPN solution uses a mixture of public-key and private-key encryption. In this scheme, each Black Hole has a public key and a private key. Each participating Black Hole has the public keys of each of the other Black Holes in the VPN. When an initial connection is made between two Black Holes in the VPN, a private key (or session key) is created by Black Hole accepting the connection.

Black Hole 3.0 uses a VPN mechanism that will interoperate within a Nortel Entrust environment (Entrust encapsulates key management, encryption, and digital signature software).

> **Note**
>
> Entrust is a public-key encryption system developed by Northern Telecom (Nortel) Secure Networks division. This client/server application supports digital signatures and public-key encryption in an application-independent fashion.

With the 3.0 release, Black Hole became the only firewall gateway with strong key management using technology that meets the existing and emerging security standards, such as X.509 certificates, X.500 corporate directory, digital signatures, Diffie-Hellman key exchange, and public-key cryptography.

NETWORK DESIGN WITH BLACK HOLE

The design or redesign of your network is not complicated by Black Hole. Black Hole's easy integration is due to the transparent operation of which Black Hole is capable. With a transparent configuration, users do not even know that Black Hole is there. The principle of Black Hole is to divide the network into two portions: the external network and the private network.

WARNING

> In an Internet situation, the external or public network has unrestricted access to only the servers that are on the public network, as shown previously in figure 11.1. This unrestricted access means that any machine outside Black Hole is visible and still open to attack. Consequently, extra caution and care should be taken when you're configuring these systems to ensure they are not compromised.
>
> When a publicly accessible system may be compromised, it must be removed from the network immediately until an investigation is completed. The process of conducting an investigation must be designed into the security policy and not left until the time when you need it.
>
> A system that has been compromised must be reinstalled. Do not attempt to surgically remove the affected components; you may miss something. Consider a compromised machine a cancer—you can't be sure that you got it all without radical action.

To Black Hole, the public information servers on the locally attached network have no preferred status; they are simply machines on the Internet. To protect these publicly accessible servers, the system administrator may want to place them on a third network known as the service network, which is illustrated in figure 11.2.

This service network configuration requires a third network interface on Black Hole. External users are allowed to access only the authorized services on the publicly available servers (FTP service on FTP server; HTTP service on HTTP server; and so on) found on the service network. Should an intruder gain privileged access to one of the servers, that intruder will have access only to the systems on the service network, which is simply a protected version of an external network.

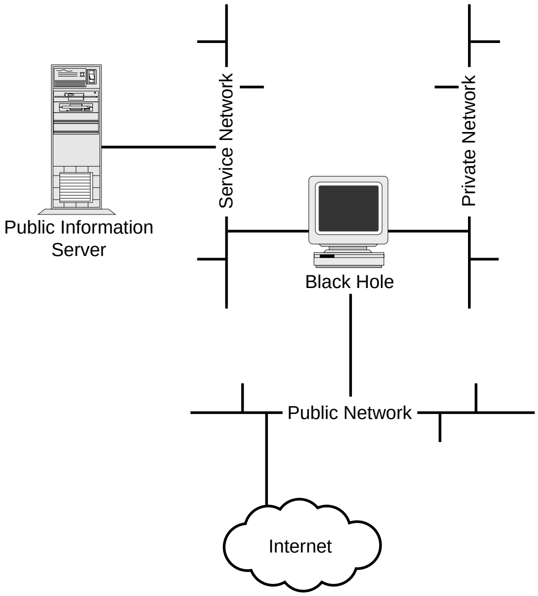


FIGURE 11.2

A Service Network configuration.

Because the service network must be visible to the Internet or external networks, you should configure this service network in a "White Hole" configuration. As was previously mentioned, Black Hole typically is configured in an environment where RFC 1597 private network addresses are used on the network behind Black Hole. In a White Hole configuration, the addresses behind the White Hole are registered and routable network addresses.

NOTE

> White Hole uses the same software and configuration services of Black Hole, but does not render the internal network invisible. White Hole is used to provide external access to the internal network with the extended logging and alarm features of Black Hole.

The White Hole configuration can be used in the Service Network approach or in the situation in which the organization primarily wants better logging capability, with the added features that Black Hole brings in this configuration.

By placing the public servers on a service network, the security manager prohibits an intruder from using a penetrated machine on the external network to monitor traffic between the Internet and the private network to collect passwords and to penetrate the internal private network.

Remembering the Security Policy

Black Hole is a tool used to enforce the security policy that the organization has developed. Attempting to use Black Hole without a security policy leads to inconsistent behavior among users and a poor understanding of the benefits within the organization. The organization must remember that as many threats come from within the organization as without. The security manager must therefore be aware of the organization's network design and how Black Hole is configured. If Black Hole is configured as a White Hole, an internal accomplice could assist the external intruder in accessing systems within the White Hole subnet, providing a way for the intruder to then move on through Black Hole using tunneling or a back door. The security policy must clearly delineate that this assistance is an unacceptable procedure and describe ways of discouraging its use.

Using the Black Hole Interface

The Black Hole interface is built around the X Windows System using X11 Release 6 technology. When you start Black Hole, you see a login screen, and you must log in through the X display Manager (XDM) interface. Along with the XDM login screen, nonmanaged windows showing the Console and Alarm Windows appear. When the screen is logging, it changes to that shown in figure 11.3, which displays the Log and Alarm Windows, and an xterm session, as managed by an X window display manager.

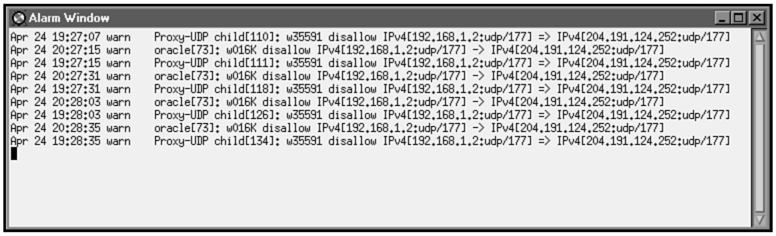


Figure 11.3
The Black Hole Alarm Window.

As mentioned previously, the FTP and telnet services are also available for remote administration. The X Windows interface is intended for use on the console or from an X Windows display server somewhere on the private network. The FTP and telnet mechanism is used when administration must be performed without X Windows, such as over a modem.

To start a remote X windows session for managing the server, a telnet session to Black Hole must be started, and then the desired X windows clients can be initiated. The windows started are an xterm session, an alarm window, and a log window. The Alarm Window, as shown in figure 11.3, is a continuous listing of the contents of the file /var/log/swatch. Each record displays the time of the alarm, the alarm level (as listed in the upcoming minitable), the application initiating the alarm, and the message text.

Worth noting is that the alarm mechanism used in Black Hole is based on the publicly available tool, swatch. The Alarm Window displays the entries from the log file that are matches for the criteria established in the Alarms section of Black Hole.

The second window shown contains the actual Black Hole log file (see figure. 11.4). This file is /var/log/messages, and each record written to that file is also displayed in this window.

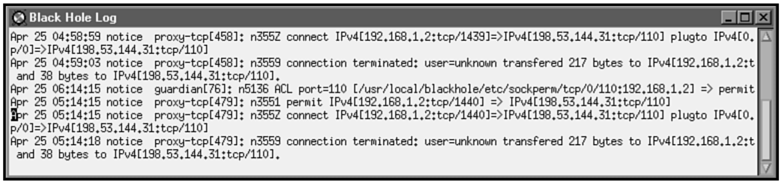


Figure 11.4
The Black Hole Log Window.

Like the Alarm Window, the Black Hole Log Window displays the date and time of the record, the level of the message, the service that generated the message, and any information for the record, such as source and destination IP addresses. Because DNS names can be spoofed, and Black Hole uses the IP source and destination IP addresses, no IP address to host name translation is performed. Several message levels define the severity of the message, as listed in the following minitable:

Reported Name	*Expanded Name*	*Description*
crit	Critical	Any message that could indicate a potential system failure.
Error	Error	An error has occurred, but a system failure is not imminent.
warn	Warning	A potential problem has occurred.
notice	Notice	Something out of the ordinary has occurred.

Actions that Black Hole takes may not always be logged. These situations are known by the security manager; they must be consciously configured. This configuration is achieved by changing the service flags for the rule applicable to the action. This is discussed further in the section entitled "Services, Users, and Rules" later in this chapter.

The main GUI interface for Black Hole is shown in figure 11.5. This screen allows the user to select the different configuration areas of the application: Users, Services, Alarms, and Reports.

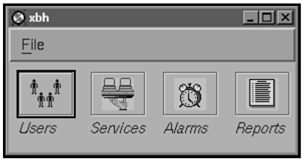


Figure 11.5

The Black Hole main interface.

The main interface allows the security manager to configure the users who are permitted to access services through Black Hole, the services or applications that are permitted to interact with Black Hole, and the Alarms that should be reported. The main interface also is used to generate reports.

As shown in figure 11.6, the File menu contains the selections Commit and Revert. The Revert selection cancels changes that have been made and reverts to the previous version of the policy database. If several changes have been made, and you want to cancel all those changes before Commit has been selected, you may select Revert. If you choose to revert to the previous version of the database and lose your changes, Black Hole prompts you to confirm that you want to revert to the previously saved version of the policy database.

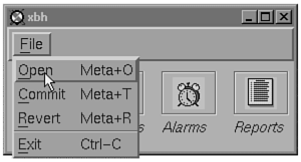


Figure 11.6

The File menu.

If you select Yes, then the changes you have made are lost. Conversely, if you choose Commit, then all changes you have made are written to the policy database and are placed immediately into operation.

The balance of the interfaces and windows are presented in the following discussion, but focus mainly on the services, users, and rules that enable Black Hole to enforce the corporate security policy.

Understanding the Policy Database

The policy database can be considered the heart and soul of Black Hole. The policy database contains the rule sets used to determine what action is taken when a connection is received by Black Hole. The security manager uses Black Hole policy database to implement the organizational security policy by developing rules for services and users.

The policy database is comprised of several tables that are assembled by a relational engine to produce rules. These tables are listed in table 11.3.

TABLE 11.3
Rule Tables

Table	*Contents*
Time	Start times/dates and duration
Service	Applications offered to the users
Address	IP addresses and address patterns
User	Black Hole user IDs, passwords, and minimum password clauses
Rule	Relationships to the four aforementioned tables

Black Hole comes with some predefined rules that allow it to be placed into operation with minimal effort on the part of the security manager. The advantage is that the security manager can immediately work to secure the organization and protect its assets in the short term.

NOTE

To protect the security manager and the organization, Black Hole makes a copy of the policy database in ASCII format every hour. This copy is made to allow for easy recovery should some attempted action leave Black Hole in an inoperable or questionable state. The ASCII database can then be reloaded, and the previous database recovered.

The ASCII copies are stored in the directory /usr/local/blackhole/oracledb/dumps. If Black Hole crashes, and either the Oracle or Guardian processes stop functioning, no connections will be accepted from either the internal or external networks, making it extremely difficult for the hacker to violate the system.

As mentioned, the rule table defines a series of relationships that are used to determine what action is taken when a connection is received. Each record in the rule table, also known as a *rule*, consists of the components in table 11.4.

Table 11.4
Rule Components

Component	*Description*
Interval	A reference to an entry in the Time table.
Service	A reference to a service type from the Service table.
Source	A reference to the Address table. In this case the address supplied may act as the source address for the rule.
Destination	A reference to the Address table, supplying an address that may act as the destination address.
User	A reference to the User table. A user name is required if the user must authenticate before being permitted to use the service.
Transfer	The action performed when the rule is applied. There are four options, which are explained later in this section.
Password	Defines the minimum password class that can be used when authenticating for this rule. The options are discussed in the text.
Service Flag	Additional options specific to the individual service.
Connect	Additional options that modify the behavior of the rule as a connection is being established. These options are discussed later in this section.

As mentioned in table 11.4, the Transfer entry in the rule has four options. These options are the following:

- **Allow.** Connect to the request service transparently, meaning that no user authentication is required.
- **Challenge.** Connect only after proper authentication has been performed.
- **Deny.** Do not connect, and provide a "deny" message to the user.
- **Drop.** Do not connect and do not provide any feedback to the user.

The rule entry contains two password-related options. One is Fixed, which means that a fixed password in the Unix style is the minimum password type permitted for this rule. An alternative method of describing this is a password that is set once and changed only when the user requests it. The second option is to use a one-time password, such as S/Key. This then becomes the minimum password level. Using this level means that using a fixed password for authentication is not permitted.

The Service flag options modify the FTP and UDP proxies provided with Black Hole. The UDP proxy operates in the same fashion as the other connection-based, or TCP, proxies by receiving the UDP packets, applying the rules for the service, and then relaying them to the internal or external host. Remember, if the service is not permitted, it is denied. The Service flag has four FTP options and three UDP options as follows:

Service	*Option*	*Description*
FTP	Put Permission	Allows the FTP put command
FTP	Get Permission	Allows the FTP get command
FTP	Delete/Rename Permission	Allows the FTP destructive administration commands
FTP	CD/DIR permission	Allows the FTP non-destructive administration commands
UDP	Multiple UDP Packets	Enables the receipt of multiple response packets, such as seen with Archie
UDP	Do not log	Prevents logging of certain activity (for example, DNS)
UDP	Use privileged port	Uses a port at 1024 or lower

The Connect options modify the behavior of the rule as a connection is being established. The modifications include the following:

- **Transparent Promote.** If the Transfer option previously discussed is set to Challenge, and transparent mode was previously enabled on the source address, then the connection is prompted to transparent for this session only.
- **Plug To.** This is the IP address of the system to which connection requests should be sent. This IP address is used in the destination field when the packet is transmitted on the other network.
- **Source Firewall.** This is the default configuration, and it means that the originating address is always Black Hole. This configuration causes the internal network and related routing to be invisible.

With these changes, the security manager can determine where the request will be sent. For example, incoming requests on port 80 would be sent to a specific system in the network. A specific example is discussed in this chapter in the later section “Configuring Application Services.”

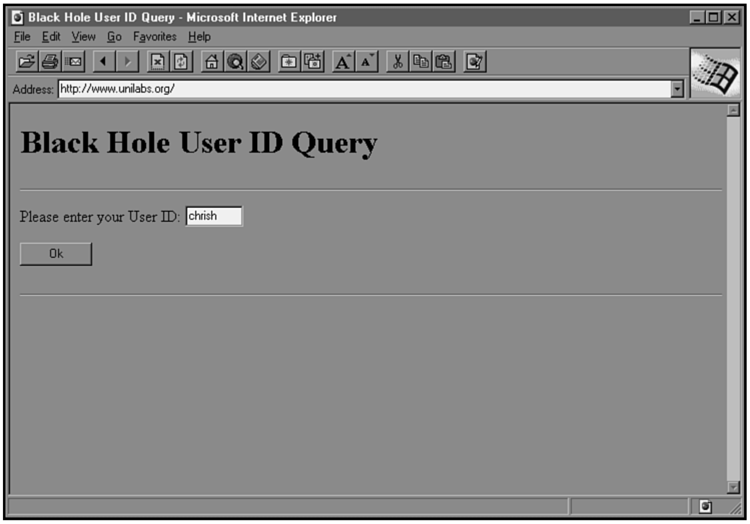
Transparent mode is a way for users to interact with the other network as if Black Hole were not even there. This seeming absence can be a great advantage sometimes. For example, all FTP requests for systems behind Black Hole from the external network can be denied. However, FTP requests originating from the internal network are permitted. Transparent mode can be enabled for individual addresses or for entire networks. Even if transparent mode is enabled for a specific address, the individual service rule must also be configured to Transparent-Allow; otherwise the connection will fail.

Transparent mode access is granted only if the connection rule uses a transfer type of Allow, in which case the connection proceeds, or a value of Challenge, in which case the user must authenticate first. For the Challenge transfer rule to succeed, the Transparent Promote flag in the Transfer options must also be selected.

When the Transfer type is set to Challenge, the user must authenticate before being able to access the requested service through the firewall. The challenge presented to the user is illustrated in figure 11.7.

FIGURE 11.7

The user authentication challenge.



As shown in figure 11.7, the user enters his or her user name and chooses OK. If the user enters a valid user name, then the dialog box shown in figure 11.8 is displayed.

The user then enters his or her password and selects Enable Transparent mode. With this mode enabled, the user does not have to authenticate for services until the session ends.

Figure 11.8

The user authentication response.

> **Note**
>
> With the multitude of client software available, some implementations might not work when Black Hole prompts the user to authenticate. For example, if the FTP client does not implement the FTP protocol specification as documented, problems may occur during authentication. If the application uses extensions specific to its implementation, which is common for Windows programs, problems will likely occur.
>
> If your client cannot authenticate and is not proxy-aware, then you must authenticate through the WWW and enable transparent access. This will enable your client to operate properly through the firewall.

Policy and Rule Resolution

As mentioned earlier, the rule table provides the information needed when a connection request is received. For each new request, a two-step process is carried out. The first step involves a comparison of the first request packet, the rules in the rule table. The comparison looks for matches between the following items:

- The current system time and the time in the rule
- The packet source address and the rule source address

- The packet destination address and the rule destination address
- The packet destination port and the rule service port

If all of these fields match within a certain range, then the rule is considered for the next step in the process. If no matches exist, the rule is dropped. If all the rules have been applied, and there is no corresponding match, then the connection is dropped.

Note

> The default action is that no data is transferred through Black Hole unless specifically configured.

Any and all rules that match are then processed a second time to determine which rule is priority. This phase is called the *priority and challenge phase*. The values in each field of the packet and the rule are considered according to the following priorities:

- Service ID or port. A unique service port, like telnet, has a higher priority than the TCP wildcard rule.
- A specific user ID has a higher priority than the default (all users).
- The 32-bit source address in the packet is compared with the rule. The rule having the most matched bits from Most to Least Significant (MSB to LSB) has priority over the other rules.
- The 32-bit destination address in the packet is compared with the rule. The rule having the most matched bits from Most to Least Significant (MSB to LSB) has priority over the other rules.
- The transfer field is checked: a more restrictive action has priority over a less restrictive action in the Drop->Disallow->Challenge->Allow order.

  This means that if you choose to disallow a service to all users except one, the service can be denied even for that user because of the transfer restrictions.

With some careful thought to how you will apply rules to implement your security policy, developing solutions to very complex access problems becomes possible.

Services, Users, and Rules

Black Hole distinguishes between users and services, each of which has its own distinct rules. A *user* corresponds to a human who is accessing some resource and must be allowed access through the firewall to reach that resource. A *service* is equivalent to an application, such as SMTP, which must be configured in order to allow packets for that protocol, or application, to be handled and, if appropriate, passed through Black Hole.

Rules

The rules are the underlying infrastructure for Black Hole. Without them, all that would be left is a fancy interface. They provide the function and operational definition of what is permitted and denied through Black Hole. Rules are saved in the policy database and are applicable to either a user or a service. This section focuses on the rules' windows and how to interact with them. Specific issues relating to user or service rules are presented in the following section.

Users and User Maintenance

Users are equivalent to Unix accounts on a typical Unix machine. It is recommended that each user within the organization be provided with an account, unless you intend to provide unrestricted access to the commonly used services (such as the World Wide Web) and restrict other applications such as telnet and FTP to certain users. However, providing each user with his or her own distinct account makes identifying the user associated with an action easier because, if no authentication is done, log records do not include the name of the user who performed the action, as illustrated here:

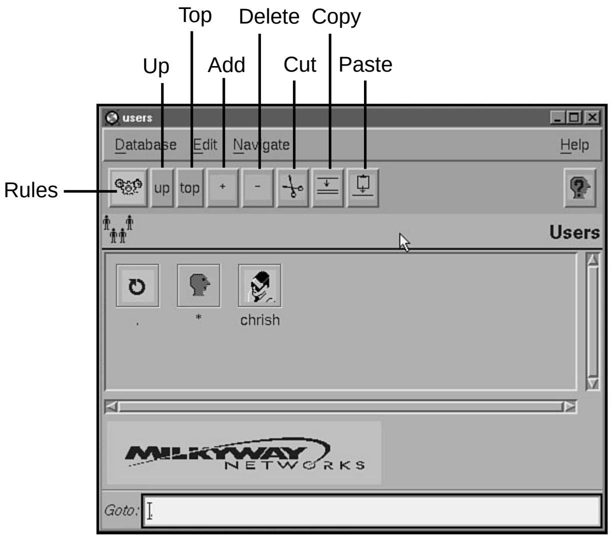
```
Apr  8 20:15:09 notice  proxy-tcp[22706]: n3559 connection terminated:
➥user=unknown
transfered 0 bytes to IPv4[192.168.1.3:tcp/1146] and 0 bytes to
➥IPv4[198.53.144.31:tcp/110].
```

User Account Management

To create users' accounts, the security manager double-clicks on the Users icon, or selects it and chooses Open from the File menu. The user-management window is then displayed, with a selection of buttons and menu items (see fig 11.9). The buttons and menu choices enable the security manager to manipulate and manage the user population.

FIGURE 11.9

The user-management window.



The buttons are described in table 11.5.

TABLE 11.5
Icon Descriptions

Button Name	*Action*
Rules	Opens the Rules window for the selected user or group
Up	Moves view of users/groups up in the group hierarchy
Top	Moves view to the root group
Add	Adds a new user
Delete	Deletes a user
Cut	Removes the selected user or group to paste it in elsewhere
Copy	Copies the selected user or group
Paste	Pastes the information in the selected group

The buttons listed in table 11.5 are found in several places in Black Hole. The controls available to administer the users and groups are identified in table 11.6.

TABLE 11.6
User-Management Commands

To	*Action*
Add a new user	Select Add from the Edit menu, or press the Add (+) button
Delete a user	Select Delete from the Edit menu, or press the Delete (-) button
Modify a user	Select a user icon and choose modify from the Edit menu, or double-click on the User icon
Disable a user	Modify the user account and, in the Update User window, select the Disable user button
Open a group in the same window	Select the group and choose Follow from the Navigate menu
Open a group in another window	Select the group icon and choose Open from the Navigate menu
Add a group	Choose New from the Navigate menu
Modify a group	Select the group and choose Modify from the Edit menu
Move up a group	Choose Up from the Navigate menu or press the Up button
Go to the top group (root group)	Select the top button, or choose Top from the Navigate menu
Go to a specific group	Enter the group name in the Goto field and press Enter
Add or modify rules	Select a user icon and choose Rules from the Edit menu, or press the Rules button
Save changes	Select Commit from the File menu

CREATING USERS AND GROUPS

Before adding users, considering how to structure the users in your organization is wise. For example, are there specific departments in which people work? If so, arranging by department may be one method of organizing the user accounts. For example, say that Company

ABC Corporation makes widgets. It has administration, accounting, sales, operations, technical, and management groups. The security manager chooses to create the user groups according to the same structure. To do this, select New from the Navigate menu to display the Update Group window shown in figure 11.10.

FIGURE 11.10

Adding a new user group.

In this window, the security manager enters the name of the group, which is limited to eight characters despite the size of the text field in the window, and selects the password method.

USER AUTHENTICATION METHODS

The authentication types are used to determine how the users in this group will gain access to the firewall. The available options are the following:

- SecurID
- Safe Word
- Unix-like Password
- S/Key

The possible authentication methods are selected from the list and added as possible methods for users in this group. When users must authenticate a specific service, they may have to authenticate with something stronger than their user account. For example, users may require only Unix-like passwords when they are authenticating for FTP usage, but may require

SecurID authentication when they are using the telnet proxy. This is controlled not only by the user account but by the authentication established for the service.

The Memo field is simply a note for the security manager that describes the group. To save the group information, choose OK to save it temporarily and return to the Users window, with the new group created. To save these current changes permanently, select Commit from the File menu.

After establishing the possible password authentication methods for each user group, you can add users to the groups. To create a user in the admin group, first choose the group by selecting the admin group from the users window and open it by choosing Open from the Navigate menu. This creates another Users window with no entries in it. Then select the Add option from the Edit menu to open the Update User window (see fig. 11.11). The Update User window contains a series of buttons and fields to be completed before the user is added.

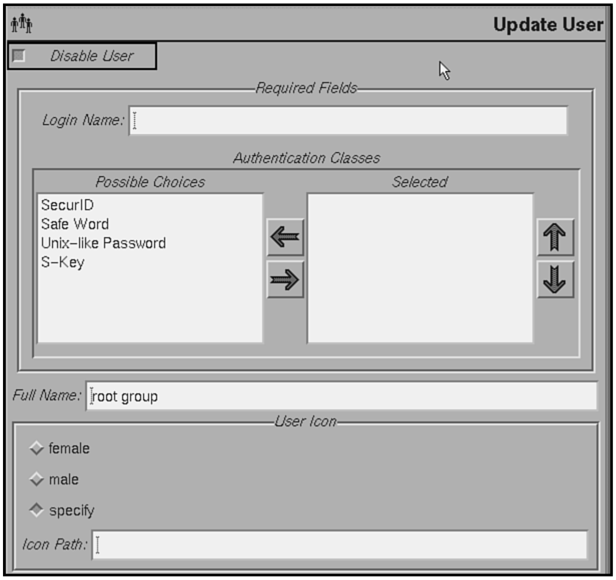


FIGURE 11.11

The Update User window.

The buttons found at the top of the Update User window (which are not visible in figure 11.11) are described in table 11.7.

TABLE 11.7
Update User Window Icons

Icon	*Name*	*Description*
	Password	Enables password management
	Icon Browser	Allows for the selection of a picture to identify the user
	Rules	Opens the rule management window

The Password button provides the security manager with the opportunity of establishing an initial password or changing a forgotten password. This button is used for the S/Key and Unix-like password schemes. Those users that utilize a hardware-type device such as SecurID and Safe Word do not require a password to be provided. When you click on the Password button, a message indicating that the password is not required, or a dialog box allowing the user to enter a password, is displayed. If the dialog box is displayed, then the user must correctly type the password twice before it is accepted.

NOTE

Users can change the password for their accounts by establishing a telnet session to Black Hole. The telnet proxy enables them to connect to a remote host or to change their password. At this point they must provide their current password, and then if validated, can supply a new password.

WARNING

Unfortunately, no mechanism is currently in place to verify that the password provided by the user is in fact a good one. Although a single carriage return may appear to be sufficient, in fact it is not. You should note that there is some basic password checking on the admin (root) password used to secure Black Hole.

THE USER ICON

The Icon browser is a tool that allows the security manager to select an icon for a user, group, or service. The browser displays the available icons and their names for inclusion in the Icon Path field. The User Icon window contains default icons for male and female users. The icon is simply a method of graphically depicting the user in the users window and has no significance beyond that.

> TIP
>
> To add new icons for use in Black Hole, add them to the directory /usr/X11R6/lib/X11/pixmaps. The only file format supported is XPM, which is the X Windows Pixmap.

The Memo section at the bottom of the Update User window is used for saving information about the user such as his or her full name, workstation location, phone number, and position.

THE USER RULES

The final issue is the rules for this user. Rules, as mentioned previously, are the mechanisms by which Black Hole actually operates. A set of predefined rules exists; making changes, however, to a Black Hole configuration—such as adding users—requires that rules be added.

The rules for users affect the following:

- What services they can use
- Source and destination addresses
- Whether users are allowed to use services, must authenticate in order to use services, or are denied services
- The type of password that users must use to authenticate
- Any connection-specific options
- When users are permitted to use services

Using the rules enables the security administrator to limit who has access to what, and when. For example, if certain Internet sites are receiving too much traffic during work hours for non–work-related material, then access to those destinations could be denied during work hours. This denial could apply to a specific user or to all users.

The rules window, which is the same for the administration of user or service rules, is discussed in the earlier section, “Understanding the Policy Database.”

Configuring Black Hole

The preceding section discussed how to add a user to Black Hole. If you have a large number of users, this can be tedious work. However, Black Hole must also be configured to offer the applications that you require as part of your operations. This section discusses the available services and how to configure them. It also presents how to add support for a currently nonsupported application.

The initial configuration of Black Hole is not very difficult, but some thought must be given before you embark on the activation of Black Hole. For example, Black Hole relies on DNS for name lookups. The DNS configurations available include the following:

- Internal and external DNS
- Black Hole as internal DNS
- Black Hole as external DNS

The solution that offers the maximum protection is to use an internal and external DNS that is not operating on Black Hole. This is the recommended practice because the versions of BIND change very quickly due to new protocol enhancements and security-related problems. By adding these unknowns to the firewall, you may be creating a problem that is not obvious until the security of your firewall has been circumvented.

Configuring an Internal and External DNS

As you saw earlier, Black Hole is often used to completely hide the internal network from the Internet, or public network. When using an internal and external DNS, you can use the Private Network Number Addressing scheme presented in RFC 1597. To use this scheme, however, an external and internal nameserver must be in operation. The external DNS is used to point systems at the private network, and the internal DNS is used to resolve hostnames in the internal network. However, you must take some care in configuring both services. Figure 11.12 illustrates the internal and external DNS configuration.

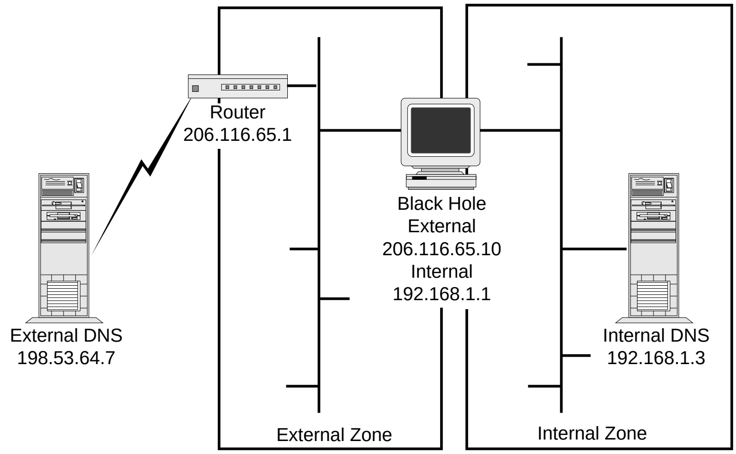


FIGURE 11.12
The internal and external DNS configuration.

The internal DNS is used to resolve hostnames within the private network. When a host outside the local enterprise is named, the internal nameserver sends the query to an external nameserver for resolution. If you have a valid root.cache file for your internal nameserver, providing a forwarder with Black Hole 2.0p3 is not necessary. Here is the named.boot file from the internal nameserver:

```
;
;    boot file for name server
;
directory /etc
cache      .                                                   root.cache
primary    unilabs.org          named.hosts
primary    0.0.127.IN-ADDR.ARPA                                named.local
primary    1.168.192.IN-ADDR.ARPA                             named.rev
```

In this file, the nameserver loads local names from the listed files. However, nonlocal names are sent to the nameservers listed in the root.cache file via Black Hole. This in itself does not mean that Black Hole is running a nameserver. Black Hole then retransmits the request on to the external nameserver for resolution and the answer.

The internal nameserver must be configured with the correct hostnames and IP addresses for the internal network, as follows:

```
ns:/etc# cat named.hosts
@       IN SOA  ns.unilabs.org. chrish.unilabs.org. (
                1996041001      ; Seria
                18000           ; Refresh
                3600            ; retry
                84600           ; expire
                84600           ; minimum
                )
IN      NS      ns.unilabs.org.
```

```
                IN      MX      20 mail.unilabs.org.
internet        IN      A       192.168.1.1
blackhole       IN      A       192.168.1.1
stargazer       IN      A       192.168.1.2
ns              IN      A       192.168.1.3
mail            IN      A       192.168.1.4
ns:/etc#
```

The named.hosts file contains only addresses for the internal systems. With this minimal configuration, the internal nameserver is operational. However, to allow for external addresses to be resolved, including a valid root.cache file on your internal nameserver is essential. This means that the internal nameserver will be able to resolve unknown addresses by contacting the root nameservers listed in the cache file to provide the requested answer.

The external nameserver is configured to show that the external machines, or those in the public network space, are listed with their real IP Addresses. This means that no private hosts are listed in the external nameserver, as illustrated in the following zone file:

```
unilabs.org.    IN SOA  nic.fonorola.net.       chrish.fonorola.net. (
                         96040901        ; Serial
                        10800   ; Refresh
                        1800    ; Retry
                        3600000 ; Expire
                        86400 ) ; Minimum
@               IN NS   nic.fonorola.net.
                IN NS   fonsrv00.fonorola.com.
                MX      10 owl.nstn.ca.
nis.fonorola.net.       IN      A       198.53.64.14
www.unilabs.org.        IN      A       204.191.126.10
cisco.unilabs.org.      IN      A       206.116.65.1
stargazer.unilabs.org.  IN      A       206.116.65.10
reliant.unilabs.org.    IN      A       206.116.65.3
mailhost.unilabs.org.   IN      A       206.116.65.4
pc2.unilabs.org.        IN      A       198.53.166.62
stealth.unilabs.org.    IN      A       206.116.65.5
blackhole.unilabs.org.  IN      A       206.116.65.10
```

Those systems that are hidden behind Black Hole and must be visible to the public network, however, are configured with Black Hole external address. This allows Black Hole to accept the connection and forward it to the correct system on the internal network. This type of configuration is done on a case-by-case basis depending on the application and host involved. A specific example is illustrated in the next section, "Configuring Application Services."

The systems on the private network should be able to resolve internal and external hosts. This capability to resolve the hosts can be tested using the `nslookup` command. The operation and view of the DNS from Black Hole must be tested until it operates as expected before traffic handling commences.

CONFIGURING APPLICATION SERVICES

A series of specific proxy applications exist that govern how Black Hole accepts and handles connections for those services. Aside from the specific proxies, a generic proxy is available to handle applications for which no proxy is supplied.

The configuration for all these service is handled through the Services window in xbh. To add or modify a service, open the Services window and select Add from the Edit menu. This opens the Services window, where you specify the information required for the proxy. This window is illustrated in figure 11.13.

FIGURE 11.13
The Black Hole Services window.

From the Services window, you select the type of proxy, TCP or UDP; the port on which the service will listen; and the class of proxy. The class of proxy determines which of the proxy agents will be used to handle the traffic. If the specific service has no proxy agent, then the best idea is to try the generic class proxy first.

The following sections illustrate how to add services that are not initially defined on Black Hole, and what issues you must be aware of when configuring the other services for operation.

POP

The use of the Post Office Protocol (POP) through Black Hole is rather unusual; therefore, a POP proxy does not come preconfigured. It is, however, part of the rules database and can be easily added. This service makes use of the generic class proxy with no special rule configuration required. No special configuration is required because the host making the POP request will be attempting to contact a specific host on either side of Black Hole.

FTP

The FTP protocol is configured by default as part of Black Hole. When Black Hole initially arrives at your site, use of the FTP service will be denied. Allowing FTP services to transfer from the internal network to the external is accomplished by changing the rules to indicate the transfer. An example is shown in figure 11.14.

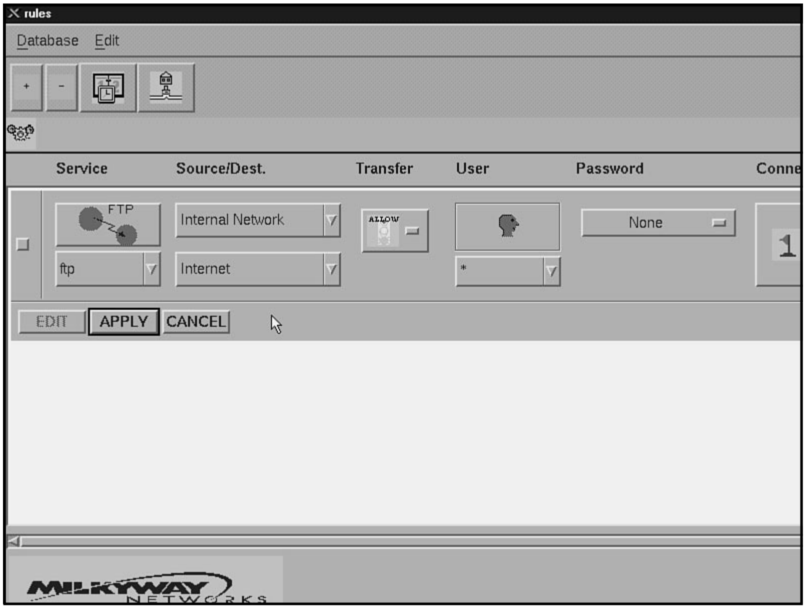


Figure 11.14

Configuring an FTP rule.

Figure 11.14 shows the rule allowing FTP transfers originating from the internal network. There is a corresponding rule that denies external FTP attempts. When configuring outgoing FTP services, it is important to configure the rule with a TCP service name of "FTP," and to use a class of FTP so that the FTP proxy is used. By doing so, the proper authentication will be used, and you will be able to configure which FTP commands are allowable for the connection.

When you are considering the need for incoming FTP services, determine whether you will have an FTP server outside the firewall as shown in the sample network in figure 11.15.

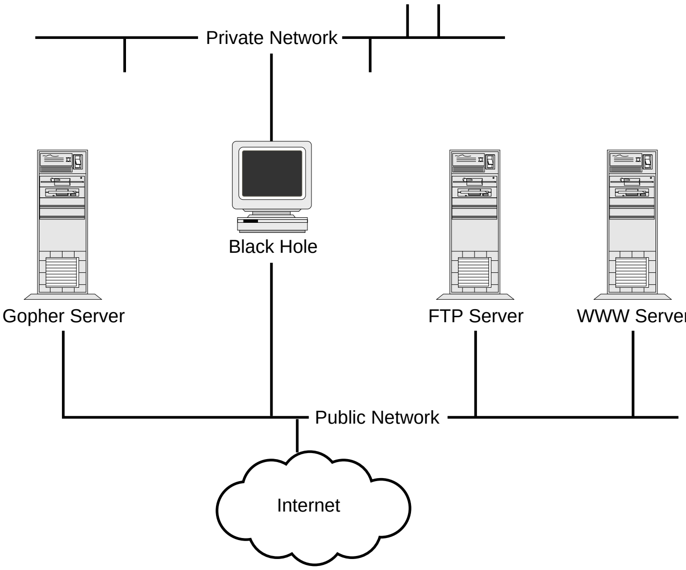


Figure 11.15

An information server network.

If you will have an FTP server outside the firewall, probably no one will ever need to download a file through Black Hole to your internal network. Not to worry, however: Even though outgoing FTP is permitted, incoming FTP from the Internet to the Private network is already denied; nothing is permitted unless specifically configured.

Gopher

Although the services database has an entry for the Gopher service, no rules are defined in the rule table, so Gopher access will be denied. When you are considering a rule for outgoing Gopher access, you should note that even if the Gopher connection request is made from a WWW browser, the connection is made through the Gopher proxy, not the HTML proxy. The connection will therefore fail, as shown in figure 11.16.

The denial of this Gopher attempt also is supported by the log files:

```
May 15 22:12:28 notice  guardian[81]: n5136 ACL port=70
[/usr/local/blackhole/etc/sockperm/tcp/0/70:192.168.1.2] => disallow
May 15 22:12:28 warn  guardian[81]: w5132 ACL disallow IPv4[192.168.1.2:1480]=>
IPv4[204.191.126.2:70]
```

When you are configuring your Gopher proxy, you must use a TCP service named "Gopher," with a class of Gopher, using port 70. You can also set up another rule using the Gopher type and class with a port of "Any." This allows other services such as Veronica to operate properly.

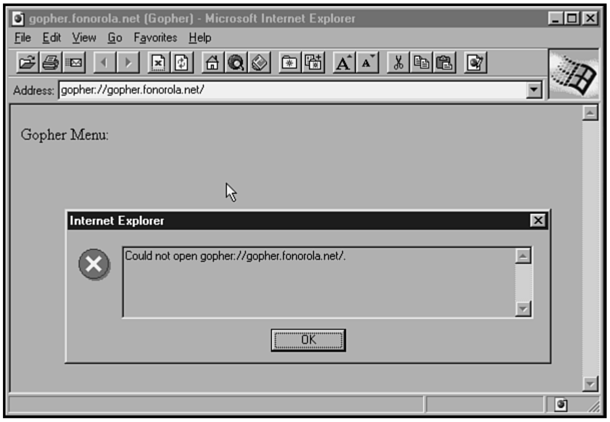


Figure 11.16
A failed Gopher via HTML.

When you are considering the need for incoming Gopher, you must determine whether you have a Gopher server on the network, as shown in figure 11.15. If so, you should disable incoming Gopher access through Black Hole using a rule similar to that shown in figure 11.16 for FTP.

HTTP

When you are configuring HTTP rules, you should use a service type of WWW, with a class of HTTP on port 80, or the other commonly used WWW port, 8080. To make sure that other facilities that can operate from a WWW browser are included, you may also create a rule that uses a port of "Any." If you want to restrict sites with material that the organization does not want to be available on its systems (such as pornography), you can restrict access to those sites by applying specific outgoing rules, as shown in figure 11.17, in which a rule to prevent access to `www1.playboy.com` is illustrated.

When you examine incoming HTTP, you must decide whether you will have a Web server on the outside of your firewall, as shown in figure 11.15. If so, consider blocking all incoming HTTP requests. If, however, you plan to run your HTTP server from behind your firewall, you should configure a rule to use the connection-specific plug-to address in order to indicate exactly where the HTTP will be sent, as shown in figure 11.18.

In figure 11.18, any attempts to contact Black Hole on port 80 are automatically sent to the address named in the plug-to field. To configure this rule, establish the source address as the Internet and the destination address as the public side of Black Hole in the rules window. For example, if the public address of Black Hole is 206.116.65.10, then the destination address for the rule is 206.116.65.10. Any connection requests on port 80 at this address will be redirected to the system identified in the Plug To field of the connection rules.

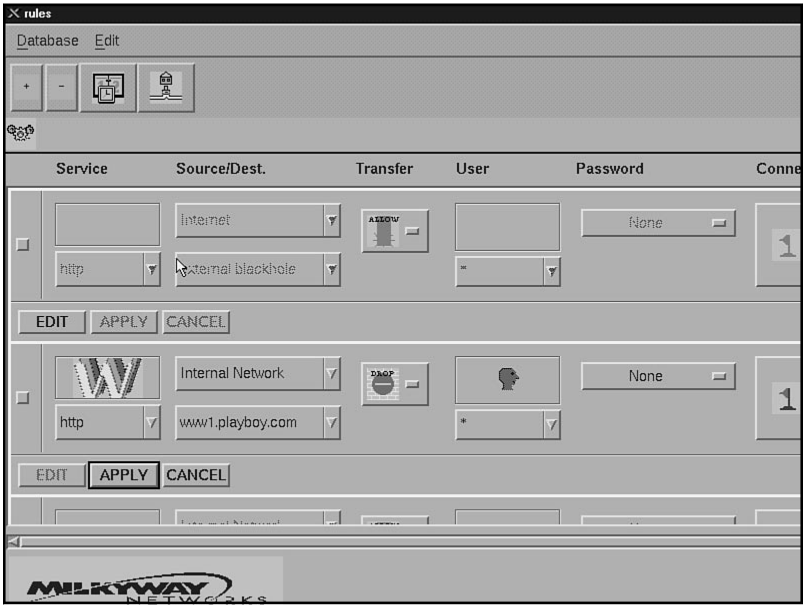


Figure 11.17

Denying host access.

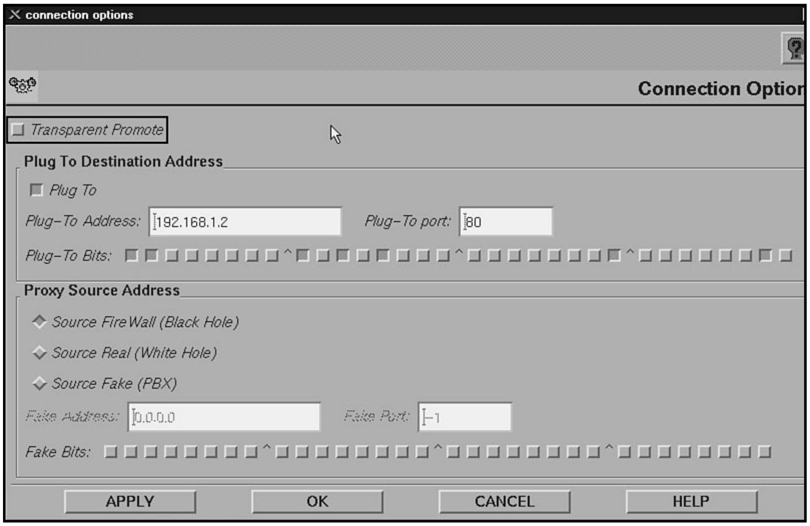


Figure 11.18

Using the plug-to address.

NNTP

Configuring Black Hole for Network News using the NNT protocol is done by configuring the external news server to transfer with an internal news server, as shown in figure 11.19.

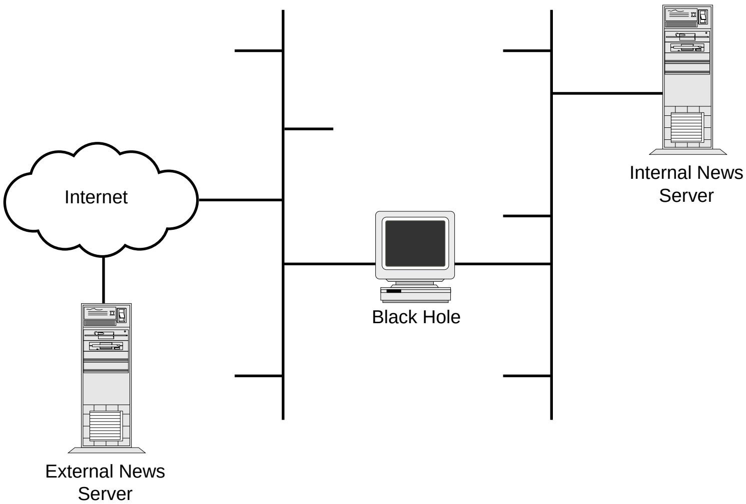


FIGURE 11.19
The NNTP relay news configuration.

Figure 11.19 illustrates a relay news server configuration in which Black Hole accepts the connection from the external news server and then forwards the packets without processing for assembly on the internal news server. From there, the news clients in the organization access news through the internal news server. The benefit of this method is that it is easier to control what administrative commands can be done through the news protocols, which have been exploited from time to time.

When configuring Black Hole for NNTP support, remember to use a service name of "NNTP," the generic proxy, and a port of 119. The generic proxy is used because no proxy specific to NNTP services exists. To establish a trusted relationship between the internal and relay news server, the source address in the rule is configured as

- The internal news server address for outgoing connections
- The relay server for incoming connections

The destination address is configured as

- The relay news server for outgoing connections
- The Black Hole address for the incoming connections

In the last case, in which the incoming address on Black Hole is used for incoming news requests, the connection-specific options are used to specify a plug-to field to specify where the packet is to be routed within your LAN.

When you are completing the configuration for this service, remember that this service will be active 24 hours a day, so transfer access must be set to Allow, and the interval must be set to Daily. These settings mean that your NNTP news provider will use a connection address equal to the public address of Black Hole. Black Hole will then forward any received packets on the NNTP port to the internal news system. News packets for the external server are first sent to the NNTP port on Black Hole, which then redirects them to the external news server.

SMTP

When the initial installation of Black Hole is completed, the mail handler is configured, making Black Hole ready to receive mail from the external sites on port 25 and deliver it to the internal mail processing server. What remains to be completed, if it hasn't already been done, is an MX entry in the DNS for your domain indicating the name of the server IP address of Black Hole, which receives the mail and then forwards it internally. This DNS entry is to be made in the external server so that Internet sites know how and where to send their messages. This entry is made by adding the following lines to the nameserver records for your domain and reloading the nameserver:

```
IN MX blackhole.unilabs.org.

blackhole.unilabs.org IN A 192.168.1.1
```

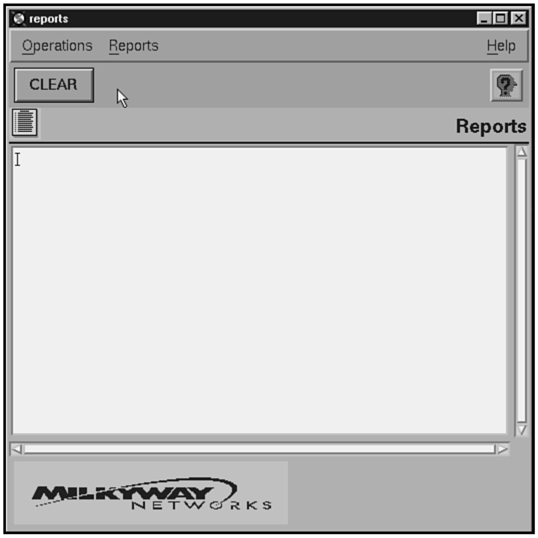
GENERATING REPORTS

Black Hole also provides for a series of reports to be generated so that you can view the traffic through your Black Hole. The reports that are currently available are the following:

- Transactions Per Hour
- Total Bytes In
- Total Bytes Out
- Connections Per Host
- Bytes per Port
- Top Ten Destinations

To access the report generator, select Reports from the xbh menu. This displays the Reports window as shown in figure 11.20.

FIGURE 11.20
The Black Hole report generator.



The Report window has two menu options, Operations and Reports, and one push-button. The push-button clears the Report window. The only menu choice under Operations is to close the Report window. The reports previously listed are found in the Report menu. The reports are based on the records found in the /var/log/messages file. Depending on the traffic on your system and how often this file is "rolled over," the file may include all information since the system was started.

The Transactions Per Hour report lists the transactions per hour over the previous 24 hours. A sample report follows:

```
TRAFFIC REPORT on Sun May  5 23:05:34 EDT 1996
   Traffic Summary by Hour
-----------------------------------------------------------------
Time Interval                       Connections
     0:00 to  1:00                        83
     1:00 to  2:00                        84
     2:00 to  3:00                        83
     3:00 to  4:00                        79
     4:00 to  5:00                        85
     5:00 to  6:00                       169
     6:00 to  7:00                       134
     7:00 to  8:00                       104
     8:00 to  9:00                       198
     9:00 to 10:00                        97
    10:00 to 11:00                        86
    11:00 to 12:00                       164
    12:00 to 13:00                       178
    13:00 to 14:00                       123
```

```
    14:00 to 15:00                        119
    15:00 to 16:00                        212
    16:00 to 17:00                        135
    17:00 to 18:00                        175
    18:00 to 19:00                        125
    19:00 to 20:00                        230
    20:00 to 21:00                        186
    21:00 to 22:00                        238
    22:00 to 23:00                        211
    23:00 to 24:00                         90
Done.
```

This report lists the transactions made per hour through Black Hole. It does not identify the protocol or user, or the direction of the connection.

The Total Bytes In and Total Bytes Out reports identify the amount of traffic received from the internal and external hosts, respectively. The report identifies the total traffic received by each unique IP address in the internal network. The same is true for outgoing, where the total traffic to each external address is listed. A Total Bytes In report is shown here:

```
TRAFFIC REPORT on Sun May  5 23:10:25 EDT 1996
   Traffic Statistics by direction (Incoming)
----------------------------------------------------------------
station IP                           Incoming Bytes
192.168.1.2                                24603459
192.168.1.3                                 1231037
204.101.157.18                               136882
204.101.157.20                                80597
128.100.177.220                               70673
204.191.39.9                                  68528
198.53.212.63                                 65829
205.150.42.41                                 65681
199.185.131.63                                63063
206.172.244.95                                56835
204.191.141.216                               50266
198.53.212.104                                46751
206.172.228.113                               46751
142.78.43.65                                  46751
Total incoming bytes = 26919911
Done.
```

The Connections Per Host report lists the number of connections made to each host identified in the report. A sample report resembles the following:

```
TRAFFIC REPORT on Sun May  5 23:16:37 EDT 1996
   Traffic Statistics by Station
----------------------------------------------------------------
Station IP                                  Connections
192.168.1.2                                     3016
204.101.157.18                                    39
198.53.167.27                                     34
206.172.244.95                                    31
128.100.177.220                                   26
```

```
199.45.70.109                                       23
204.191.39.9                                        22
Done.
```

The Bytes Per Port report lists the amount of traffic though each of a predetermined set of ports. The amount of traffic includes connections made from both sides of the firewall. The ports reported cannot be changed. You may notice that the report does not include the SMTP port. It is not included because the report is based on the log messages found in /var/log/messages. SMTP log messages are not written in that file and therefore are not reported. Because the reports are written in PERL, however, they can be extended and modified to suit your needs. The following is a sample Bytes Per Hour report:

```
TRAFFIC REPORT on Sun May  5 23:18:03 EDT 1996
   Packet Count by Port (Service)
-----------------------------------------------------------------
Port                                        Number of Bytes
    21                                      14856854
    80                                       8234632
   110                                       3151985
   119                                        662296
    23                                         17048
    53                                            31
Done.
```

The last report available lists the top ten destinations for internal users, meaning the ten sites that internal users visit most often. A sample report is shown here:

```
TRAFFIC REPORT on Sun May  5 23:23:32 EDT 1996
   Most Popular Destinations
-----------------------------------------------------------------
Rank  Station IP                                     Connections
  1   198.53.144.31                                        2077
  2   192.168.1.2                                           358
  3   206.61.207.115                                        104
  4   206.29.210.100                                         94
  5   204.191.126.13                                         78
  6   199.85.71.65                                           61
  7   199.0.62.2                                             40
  8   198.73.137.2                                           39
  9   140.142.50.1                                           37
 10   193.129.186.2                                          31
Done.
```

Although no method currently exists for a user to easily generate on-the-fly reports, Black Hole reports are written in the PERL programming language. An experienced programmer could therefore easily develop some additional reports. However, adding them to the Reports menu is impossible at this time. According to Milkyway Networks, a new SQL-based report generator that will support on-the-fly and ad-hoc reporting is coming in Black Hole Version 3.0.

For More Information

For more Information on Black Hole, you can reach Milkyway Networks Corporation as follows:

Milkyway Networks Corp.
255-2650 Queensview Drive
Ottawa, Ontario K2B 8H6

Voice +1 (613) 596-5549
Fax: +1 (613) 596-5615

URL : `http://www.milkyway.com`

Summary

The selection of the firewall that will be used to protect your organization is a choice that will follow you for many years. The decision must be based upon several questions:

- What features do I need within my firewall?
- Do I need an application gateway?
- How much effort will it take to maintain?

This chapter has presented the operation of the Black Hole firewall/application gateway from Milkyway Networks. It has covered how to design a network with Black Hole and how to operate and manage the system to implement the organizational security policy. Black Hole is a capable, application-level secure gateway that has many of the features that most security managers want. The success of Black Hole in the commercial marketplace is testimony to the value placed upon the product.