CHAPTER 9

Firewalls and Intrusion Prevention Systems

LEARNING OBJECTIVES

After studying this chapter, you should be able to:

- Explain the role of firewalls as part of a computer and network security strategy.
- List the key characteristics of firewalls.
- Discuss the various basing options for firewalls.
- Understand the relative merits of various choices for firewall location and configurations.
- Distinguish between firewalls and intrusion prevention systems.

Firewalls can be an effective means of protecting a local system or network of systems from network-based security threats while at the same time affording access to the outside world via wide area networks and the Internet.

9.1 THE NEED FOR FIREWALLS

Information systems in corporations, government agencies, and other organizations have undergone a steady evolution. The following are notable developments:

- Centralized data processing system, with a central mainframe supporting a number of directly connected terminals.
- Local area networks (LANs) interconnecting PCs and terminals to each other and the mainframe.
- Premises network, consisting of a number of LANs, interconnecting PCs, servers, and perhaps a mainframe or two.
- Enterprise-wide network, consisting of multiple, geographically distributed premises networks interconnected by a private wide area network (WAN).
- Internet connectivity, in which the various premises networks all hook into the Internet and may or may not also be connected by a private WAN.
- Enterprise cloud computing, which we will describe further in Chapter 13, with virtualized servers located in one or more data centers that can provide both internal organizational and external Internet accessible services.

Internet connectivity is no longer optional for most organizations. The information and services available are essential to the organization. Moreover, individual users within the organization want and need Internet access, and if this is not provided via their LAN, they could use a wireless broadband capability from their PC to an Internet service provider (ISP). However, while Internet access provides benefits to the organization, it enables the outside world to reach and interact with local network assets. This creates a threat to the organization. While it is possible to equip each workstation and server on the premises network with strong security features, such as intrusion protection, this may not be sufficient, and in some cases is not cost-effective.

Consider a network with hundreds or even thousands of systems, running various operating systems, such as different versions of Windows, MacOS, and Linux. When a security flaw is discovered, each potentially affected system must be upgraded to fix that flaw. This requires scaleable configuration management and aggressive patching to function effectively. While difficult, this is possible and is necessary if only host-based security is used. A widely accepted alternative or at least complement to host-based security services is the firewall. The firewall is inserted between the premises network and the Internet to establish a controlled link and to erect an outer security wall or perimeter. The aim of this perimeter is to protect the premises network from Internet-based attacks and to provide a single choke point where security and auditing can be imposed. The firewall may be a single computer system or a set of two or more systems that cooperate to perform the firewall function.

The firewall, then, provides an additional layer of defense, insulating the internal systems from external networks. This follows the classic military doctrine of "defense in depth," which is just as applicable to IT security.

9.2 FIREWALL CHARACTERISTICS AND ACCESS POLICY

[BELL94] lists the following design goals for a firewall:

1. All traffic from inside to outside, and vice versa, must pass through the firewall. This is achieved by physically blocking all access to the local network except via the firewall. Various configurations are possible, as explained later in this chapter.
2. Only authorized traffic, as defined by the local security policy, will be allowed to pass. Various types of firewalls are used, which implement various types of security policies, as explained later in this chapter.
3. The firewall itself is immune to penetration. This implies the use of a hardened system with a secured operating system, as we will describe in Chapter 12.

A critical component in the planning and implementation of a firewall is specifying a suitable access policy. This lists the types of traffic authorized to pass through the firewall, including address ranges, protocols, applications, and content types. This policy should be developed from the organization's information security risk assessment and policy, that we will discuss in Chapters 14 and 15. This policy should be developed from a broad specification of which traffic types the organization needs to support. It is then refined to detail the filter elements we will discuss next, which can then be implemented within an appropriate firewall topology.

NIST SP 800-41 (*Guidelines on Firewalls and Firewall Policy*, September 2009) lists a range of characteristics that a firewall access policy could use to filter traffic, including:

- **IP Address and Protocol Values:** Controls access based on the source or destination addresses and port numbers, direction of flow being inbound or outbound, and other network and transport layer characteristics. This type of filtering is used by packet filter and stateful inspection firewalls. It is typically used to limit access to specific services.

- **Application Protocol:** Controls access on the basis of authorized application protocol data. This type of filtering is used by an application-level gateway that relays and monitors the exchange of information for specific application protocols, for example, checking Simple Mail Transfer Protocol (SMTP) e-mail for spam, or HTTP Web requests to authorized sites only.
- **User Identity:** Controls access based on the users identity, typically for inside users who identify themselves using some form of secure authentication technology, such as IPSec (see Chapter 22).
- **Network Activity:** Controls access based on considerations such as the time or request, for example, only in business hours; rate of requests, for example, to detect scanning attempts; or other activity patterns.

Before proceeding to the details of firewall types and configurations, it is best to summarize what one can expect from a firewall. The following capabilities are within the scope of a firewall:

1. A firewall defines a single choke point that attempts to keep unauthorized users out of the protected network, prohibit potentially vulnerable services from entering or leaving the network, and provide protection from various kinds of IP spoofing and routing attacks. The use of a single choke point simplifies security management because security capabilities are consolidated on a single system or set of systems.
2. A firewall provides a location for monitoring security-related events. Audits and alarms can be implemented on the firewall system.
3. A firewall is a convenient platform for several Internet functions that are not security related. These include a network address translator, which maps local addresses to Internet addresses, and a network management function that audits or logs Internet usage.
4. A firewall can serve as the platform for IPSec. Using the tunnel mode capability described in Chapter 22, the firewall can be used to implement virtual private networks.

Firewalls have their limitations, including the following:

1. The firewall cannot protect against attacks that bypass the firewall. Internal systems may have wired or mobile broadband capability to connect to an ISP. An internal LAN may have direct connections to peer organizations that bypass the firewall.
2. The firewall may not protect fully against internal threats, such as a disgruntled employee or an employee who unwittingly cooperates with an external attacker.
3. An improperly secured wireless LAN may be accessed from outside the organization. An internal firewall that separates portions of an enterprise network cannot guard against wireless communications between local systems on different sides of the internal firewall.
4. A laptop, PDA, or portable storage device may be used and infected outside the corporate network, then attached and used internally.

9.3 TYPES OF FIREWALLS

A firewall can monitor network traffic at a number of levels, from low-level network packets, either individually or as part of a flow, to all traffic within a transport connection, up to inspecting details of application protocols. The choice of which level is appropriate is determined by the desired firewall access policy. It can operate as a positive filter, allowing to pass only packets that meet specific criteria, or as a negative

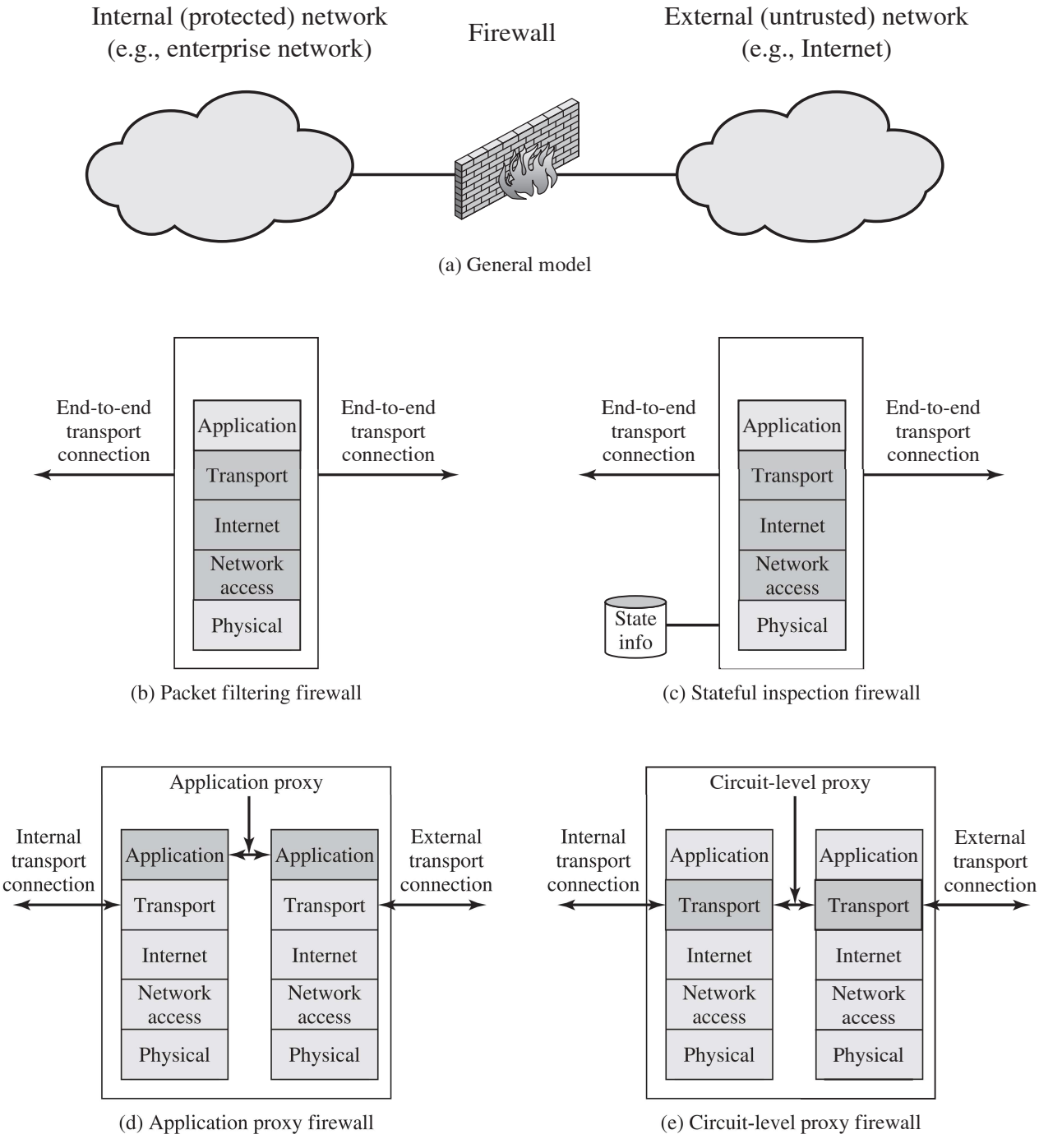

Figure 9.1 Types of Firewalls

filter, rejecting any packet that meets certain criteria. The criteria implement the access policy for the firewall that we discussed in the previous section. Depending on the type of firewall, it may examine one or more protocol headers in each packet, the payload of each packet, or the pattern generated by a sequence of packets. In this section, we look at the principal types of firewalls.

Packet Filtering Firewall

A **packet filtering firewall** applies a set of rules to each incoming and outgoing IP packet and then forward or discards the packet (see Figure 9.1b). The firewall is typically configured to filter packets going in both directions (from and to the internal network). Filtering rules are based on information contained in a network packet:

- **Source IP address:** The IP address of the system that originated the IP packet (e.g., 192.178.1.1).
- **Destination IP address:** The IP address of the system the IP packet is trying to reach (e.g., 192.168.1.2).
- **Source and destination transport-level address:** The transport-level (e.g., TCP or UDP) port number, which defines applications such as SNMP or HTTP.
- **IP protocol field:** Defines the transport protocol.
- **Interface:** For a firewall with three or more ports, which interface of the firewall the packet came from or for which interface of the firewall the packet is destined.

The packet filter is typically set up as a list of rules based on matches to fields in the IP or TCP header. If there is a match to one of the rules, that rule is invoked to determine whether to forward or discard the packet. If there is no match to any rule, then a default action is taken. Two default policies are possible:

- **Default = discard:** That which is not expressly permitted is prohibited.
- **Default = forward:** That which is not expressly prohibited is permitted.

The default discard policy is more conservative. Initially, everything is blocked, and services must be added on a case-by-case basis. This policy is more visible to users, who are more likely to see the firewall as a hindrance. However, this is the policy likely to be preferred by businesses and government organizations. Further, visibility to users diminishes as rules are created. The default forward policy increases ease of use for end users but provides reduced security; the security administrator must, in essence, react to each new security threat as it becomes known. This policy may be used by generally more open organizations, such as universities.

Table 9.1 is a simplified example of a rule set for SMTP traffic. The goal is to allow inbound and outbound e-mail traffic but to block all other traffic. The rules are applied top to bottom to each packet. The intent of each rule is:

1. Inbound mail from an external source is allowed (port 25 is for SMTP incoming).
2. This rule is intended to allow a response to an inbound SMTP connection.
3. Outbound mail to an external source is allowed.

Table 9.1 Packet-Filtering Examples

Rule	Direction	Src address	Dest addresss	Protocol	Dest port	Action
1	In	External	Internal	TCP	25	Permit
2	Out	Internal	External	TCP	>1023	Permit
3	Out	Internal	External	TCP	25	Permit
4	In	External	Internal	TCP	>1023	Permit
5	Either	Any	Any	Any	Any	Deny

4. This rule is intended to allow a response to an outbound SMTP connection.
5. This is an explicit statement of the default policy. All rule sets include this rule implicitly as the last rule.

There are several problems with this rule set. Rule 4 allows external traffic to any destination port above 1023. As an example of an exploit of this rule, an external attacker can open a connection from the attacker's port 5150 to an internal Web proxy server on port 8080. This is supposed to be forbidden and could allow an attack on the server. To counter this attack, the firewall rule set can be configured with a source port field for each row. For rules 2 and 4, the source port is set to 25; for rules 1 and 3, the source port is set to >1023.

But a vulnerability remains. Rules 3 and 4 are intended to specify that any inside host can send mail to the outside. A TCP packet with a destination port of 25 is routed to the SMTP server on the destination machine. The problem with this rule is that the use of port 25 for SMTP receipt is only a default; an outside machine could be configured to have some other application linked to port 25. As the revised rule 4 is written, an attacker could gain access to internal machines by sending packets with a TCP source port number of 25. To counter this threat, we can add an ACK flag field to each row. For rule 4, the field would indicate that the ACK flag must be set on the incoming packet. Rule 4 would now look like this:

Rule	Direction	Src address	Src port	Dest address	Protocol	Dest port	Flag	Action
4	In	External	25	Internal	TCP	>1023	ACK	Permit

The rule takes advantage of a feature of TCP connections. Once a connection is set up, the ACK flag of a TCP segment is set to acknowledge segments sent from the other side. Thus, this rule allows incoming packets with a source port number of 25 that include the ACK flag in the TCP segment.

One advantage of a packet filtering firewall is its simplicity. In addition, packet filters typically are transparent to users and are very fast. NIST SP 800-41 lists the following weaknesses of packet filter firewalls:

- Because packet filter firewalls do not examine upper-layer data, they cannot prevent attacks that employ application-specific vulnerabilities or functions. For example, a packet filter firewall cannot block specific application commands; if

a packet filter firewall allows a given application, all functions available within that application will be permitted.

- Because of the limited information available to the firewall, the logging functionality present in packet filter firewalls is limited. Packet filter logs normally contain the same information used to make access control decisions (source address, destination address, and traffic type).
- Most packet filter firewalls do not support advanced user authentication schemes. Once again, this limitation is mostly due to the lack of upper-layer functionality by the firewall.
- Packet filter firewalls are generally vulnerable to attacks and exploits that take advantage of problems within the TCP/IP specification and protocol stack, such as *network layer address spoofing*. Many packet filter firewalls cannot detect a network packet in which the OSI Layer 3 addressing information has been altered. Spoofing attacks are generally employed by intruders to bypass the security controls implemented in a firewall platform.
- Finally, due to the small number of variables used in access control decisions, packet filter firewalls are susceptible to security breaches caused by improper configurations. In other words, it is easy to accidentally configure a packet filter firewall to allow traffic types, sources, and destinations that should be denied based on an organization's information security policy.

Some of the attacks that can be made on packet filtering firewalls and the appropriate countermeasures are the following:

- **IP address spoofing:** The intruder transmits packets from the outside with a source IP address field containing an address of an internal host. The attacker hopes that the use of a spoofed address will allow penetration of systems that employ simple source address security, in which packets from specific trusted internal hosts are accepted. The countermeasure is to discard packets with an inside source address if the packet arrives on an external interface. In fact, this countermeasure is often implemented at the router external to the firewall.
- **Source routing attacks:** The source station specifies the route that a packet should take as it crosses the Internet, in the hopes that this will bypass security measures that do not analyze the source routing information. A countermeasure is to discard all packets that use this option.
- **Tiny fragment attacks:** The intruder uses the IP fragmentation option to create extremely small fragments and force the TCP header information into a separate packet fragment. This attack is designed to circumvent filtering rules that depend on TCP header information. Typically, a packet filter will make a filtering decision on the first fragment of a packet. All subsequent fragments of that packet are filtered out solely on the basis that they are part of the packet whose first fragment was rejected. The attacker hopes the filtering firewall examines only the first fragment and the remaining fragments are passed through. A tiny fragment attack can be defeated by enforcing a rule that the first fragment of a packet must contain a predefined minimum amount of the transport header. If the first fragment is rejected, the filter can remember the packet and discard all subsequent fragments.

Stateful Inspection Firewalls

A traditional packet filter makes filtering decisions on an individual packet basis and does not take into consideration any higher-layer context. To understand what is meant by *context* and why a traditional packet filter is limited with regard to context, a little background is needed. Most standardized applications that run on top of TCP follow a client/server model. For example, for the SMTP, e-mail is transmitted from a client system to a server system. The client system generates new e-mail messages, typically from user input. The server system accepts incoming e-mail messages and places them in the appropriate user mailboxes. SMTP operates by setting up a TCP connection between client and server, in which the TCP server port number, which identifies the SMTP server application, is 25. The TCP port number for the SMTP client is a number between 1024 and 65535 that is generated by the SMTP client.

In general, when an application that uses TCP creates a session with a remote host, it creates a TCP connection in which the TCP port number for the remote (server) application is a number less than 1024 and the TCP port number for the local (client) application is a number between 1024 and 65535. The numbers less than 1024 are the "well-known" port numbers and are assigned permanently to particular applications (e.g., 25 for server SMTP). The numbers between 1024 and 65535 are generated dynamically and have temporary significance only for the lifetime of a TCP connection.

A simple packet filtering firewall must permit inbound network traffic on all these high-numbered ports for TCP-based traffic to occur. This creates a vulnerability that can be exploited by unauthorized users.

A **stateful packet inspection firewall** tightens up the rules for TCP traffic by creating a directory of outbound TCP connections, as shown in Table 9.2. There is an entry for each currently established connection. The packet filter will now allow incoming traffic to high-numbered ports only for those packets that fit the profile of one of the entries in this directory.

Table 9.2 Example Stateful Firewall Connection State Table

Source Address	Source Port	Destination Address	Destination Port	Connection State
192.168.1.100	1030	210.9.88.29	80	Established
192.168.1.102	1031	216.32.42.123	80	Established
192.168.1.101	1033	173.66.32.122	25	Established
192.168.1.106	1035	177.231.32.12	79	Established
223.43.21.231	1990	192.168.1.6	80	Established
219.22.123.32	2112	192.168.1.6	80	Established
210.99.212.18	3321	192.168.1.6	80	Established
24.102.32.23	1025	192.168.1.6	80	Established
223.21.22.12	1046	192.168.1.6	80	Established

A stateful packet inspection firewall reviews the same packet information as a packet filtering firewall, but also records information about TCP connections (see Figure 9.1c). Some stateful firewalls also keep track of TCP sequence numbers to prevent attacks that depend on the sequence number, such as session hijacking. Some even inspect limited amounts of application data for some well-known protocols such as FTP, IM, and SIPS commands, in order to identify and track related connections.

Application-Level Gateway

An **application-level gateway**, also called an application proxy, acts as a relay of application-level traffic (see Figure 9.1d). The user contacts the gateway using a TCP/IP application, such as Telnet or FTP, and the gateway asks the user for the name of the remote host to be accessed. When the user responds and provides a valid user ID and authentication information, the gateway contacts the application on the remote host and relays TCP segments containing the application data between the two endpoints. If the gateway does not implement the proxy code for a specific application, the service is not supported and cannot be forwarded across the firewall. Further, the gateway can be configured to support only specific features of an application that the network administrator considers acceptable while denying all other features.

Application-level gateways tend to be more secure than packet filters. Rather than trying to deal with the numerous possible combinations that are to be allowed and forbidden at the TCP and IP level, the application-level gateway need only scrutinize a few allowable applications. In addition, it is easy to log and audit all incoming traffic at the application level.

A prime disadvantage of this type of gateway is the additional processing overhead on each connection. In effect, there are two spliced connections between the end users, with the gateway at the splice point, and the gateway must examine and forward all traffic in both directions.

Circuit-Level Gateway

A fourth type of firewall is the **circuit-level gateway** or circuit-level proxy (see Figure 9.1e). This can be a stand-alone system or it can be a specialized function performed by an application-level gateway for certain applications. As with an application gateway, a circuit-level gateway does not permit an end-to-end TCP connection; rather, the gateway sets up two TCP connections, one between itself and a TCP user on an inner host and one between itself and a TCP user on an outside host. Once the two connections are established, the gateway typically relays TCP segments from one connection to the other without examining the contents. The security function consists of determining which connections will be allowed.

A typical use of circuit-level gateways is a situation in which the system administrator trusts the internal users. The gateway can be configured to support application-level or proxy service on inbound connections and circuit-level functions for outbound connections. In this configuration, the gateway can incur the processing overhead of examining incoming application data for forbidden functions, but does not incur that overhead on outgoing data.

An example of a circuit-level gateway implementation is the SOCKS package [KOBL92]; version 5 of SOCKS is specified in RFC 1928. The RFC defines SOCKS in the following fashion:

> The protocol described here is designed to provide a framework for client–server applications in both the TCP and UDP domains to conveniently and securely use the services of a network firewall. The protocol is conceptually a "shim-layer" between the application layer and the transport layer, and as such does not provide network-layer gateway services, such as forwarding of ICMP messages.

SOCKS consists of the following components:

- The SOCKS server, which often runs on a UNIX-based firewall. SOCKS is also implemented on Windows systems.
- The SOCKS client library, which runs on internal hosts protected by the firewall.
- SOCKS-ified versions of several standard client programs such as FTP and TELNET. The implementation of the SOCKS protocol typically involves either the recompilation or relinking of TCP-based client applications, or the use of alternate dynamically loaded libraries, to use the appropriate encapsulation routines in the SOCKS library.

When a TCP-based client wishes to establish a connection to an object that is reachable only via a firewall (such determination is left up to the implementation), it must open a TCP connection to the appropriate SOCKS port on the SOCKS server system. The SOCKS service is located on TCP port 1080. If the connection request succeeds, the client enters a negotiation for the authentication method to be used, authenticates with the chosen method, then sends a relay request. The SOCKS server evaluates the request and either establishes the appropriate connection or denies it. UDP exchanges are handled in a similar fashion. In essence, a TCP connection is opened to authenticate a user to send and receive UDP segments, and the UDP segments are forwarded as long as the TCP connection is open.

9.4 FIREWALL BASING

It is common to base a firewall on a stand-alone machine running a common operating system, such as UNIX or Linux, that may be supplied as a pre-configured security appliance. Firewall functionality can also be implemented as a software module in a router or LAN switch, or in a server. In this section, we look at some additional firewall basing considerations.

Bastion Host

A bastion host is a system identified by the firewall administrator as a critical strong point in the network's security. Typically, the bastion host serves as a platform for application-level or circuit-level gateways, or to support other services such as IPSec. Common characteristics of a bastion host are as follows:

- The bastion host hardware platform executes a secure version of its operating system, making it a hardened system.

Summary of Firewall Locations and Topologies

We can now summarize the discussion from Sections 9.4 and 9.5 to define a spectrum of firewall locations and topologies. The following alternatives can be identified:

- **Host-resident firewall:** This category includes personal firewall software and firewall software on servers, both physical and virtual. Such firewalls can be used alone or as part of an in-depth firewall deployment.
- **Screening router:** A single router between internal and external networks with stateless or full packet filtering. This arrangement is typical for small office/home office (SOHO) applications.
- **Single bastion inline:** A single firewall physical or virtual device located between an internal and external router (e.g., Figure 9.1a). The firewall may implement stateful filters and/or application proxies. This is the typical firewall appliance configuration for small to medium-sized organizations.
- **Single bastion T:** Similar to single bastion inline, but has a third network interface on bastion to a DMZ where externally visible servers are placed. Again, this is a common appliance configuration for medium to large organizations.
- **Double bastion inline:** Figure 9.2 illustrates this configuration, where the DMZ is sandwiched between bastion firewalls. This configuration is common for large businesses and government organizations.
- **Double bastion T:** Figure 8.5 illustrates this configuration. The DMZ is on a separate network interface on the bastion firewall. This configuration is also common for large businesses and government organizations and may be required.
- **Distributed firewall configuration:** Illustrated in Figure 9.4. This configuration is used by some large businesses and government organizations.

9.6 INTRUSION PREVENTION SYSTEMS

A further addition to the range of security products is the intrusion prevention system (IPS), also known as intrusion detection and prevention system (IDPS). It is an extension of an IDS that includes the capability to attempt to block or prevent detected malicious activity. Like an IDS, an IPS can be host-based, network-based, or distributed/hybrid, as we discussed in Chapter 8. Similarly, it can use anomaly detection to identify behavior that is not that of legitimate users, or signature/heuristic detection to identify known malicious behavior.

Once an IDS has detected malicious activity, it can respond by modifying or blocking network packets across a perimeter or into a host, or by modifying or blocking system calls by programs running on a host. Thus, a network IPS can block traffic, as a firewall does, but makes use of the types of algorithms developed for IDSs to determine when to do so. It is a matter of terminology whether a network IPS is considered a separate, new type of product, or simply another form of firewall.

Host-Based IPS

A host-based IPS (HIPS) can make use of either signature/heuristic or anomaly detection techniques to identify attacks. In the former case, the focus is on the specific

content of application network traffic, or of sequences of system calls, looking for patterns that have been identified as malicious. In the case of anomaly detection, the IPS is looking for behavior patterns that indicate malware. Examples of the types of malicious behavior addressed by a HIPS include the following:

- **Modification of system resources:** Rootkits, Trojan horses, and backdoors operate by changing system resources, such as libraries, directories, registry settings, and user accounts.
- **Privilege-escalation exploits:** These attacks attempt to give ordinary users root access.
- **Buffer-overflow exploits:** These attacks will be described in Chapter 10.
- **Access to e-mail contact list:** Many worms spread by mailing a copy of themselves to addresses in the local system's e-mail address book.
- **Directory traversal:** A directory traversal vulnerability in a Web server allows the hacker to access files outside the range of what a server application user would normally need to access.

Attacks such as these result in behaviors that can be analyzed by a HIPS. The HIPS capability can be tailored to the specific platform. A set of general-purpose tools may be used for a desktop or server system. Some HIPS packages are designed to protect specific types of servers, such as Web servers and database servers. In this case, the HIPS looks for particular application attacks.

In addition to signature and anomaly-detection techniques, a HIPS can use a sandbox approach. Sandboxes are especially suited to mobile code, such as Java applets and scripting languages. The HIPS quarantines such code in an isolated system area, then runs the code and monitors its behavior. If the code violates predefined policies or matches predefined behavior signatures, it is halted and prevented from executing in the normal system environment.

[ROBB06a] lists the following as areas for which a HIPS typically offers desktop protection:

- **System calls:** The kernel controls access to system resources such as memory, I/O devices, and processor. To use these resources, user applications invoke system calls to the kernel. Any exploit code will execute at least one system call. The HIPS can be configured to examine each system call for malicious characteristics.
- **File system access:** The HIPS can ensure that file access system calls are not malicious and meet established policy.
- **System registry settings:** The registry maintains persistent configuration information about programs and is often maliciously modified to extend the life of an exploit. The HIPS can ensure that the system registry maintains its integrity.
- **Host input/output:** I/O communications, whether local or network-based, can propagate exploit code and malware. The HIPS can examine and enforce proper client interaction with the network and its interaction with other devices.

The Role of Hips Many industry observers see the enterprise endpoint, including desktop and laptop systems, as now the main target for hackers and criminals, more so than network devices [ROBB06b]. Thus, security vendors are focusing more on

developing endpoint security products. Traditionally, endpoint security has been provided by a collection of distinct products, such as antivirus, antispyware, antispam, and personal firewalls. The HIPS approach is an effort to provide an integrated, single-product suite of functions. The advantages of the integrated HIPS approach are that the various tools work closely together, threat prevention is more comprehensive, and management is easier.

It may be tempting to think that endpoint security products such as HIPS, if sophisticated enough, eliminate or at least reduce the need for network-level devices. For example, the San Diego Supercomputer Center reports that over a four-year period, there were no intrusions on any of its managed machines, in a configuration with no firewalls and just endpoint security protection [SING03]. Nevertheless, a more prudent approach is to use HIPS as one element in a defense-in-depth strategy that involves network-level devices, such as either firewalls or network-based IPSs.

Network-Based IPS

A network-based IPS (NIPS) is in essence an inline NIDS with the authority to modify or discard packets and tear down TCP connections. As with a NIDS, a NIPS makes use of techniques such as signature/heuristic detection and anomaly detection.

Among the techniques used in a NIPS but not commonly found in a firewall is flow data protection. This requires that the application payload in a sequence of packets be reassembled. The IPS device applies filters to the full content of the flow every time a new packet for the flow arrives. When a flow is determined to be malicious, the latest and all subsequent packets belonging to the suspect flow are dropped.

In terms of the general methods used by a NIPS device to identify malicious packets, the following are typical:

- **Pattern matching:** Scans incoming packets for specific byte sequences (the signature) stored in a database of known attacks.
- **Stateful matching:** Scans for attack signatures in the context of a traffic stream rather than individual packets.
- **Protocol anomaly:** Looks for deviation from standards set forth in RFCs.
- **Traffic anomaly:** Watches for unusual traffic activities, such as a flood of UDP packets or a new service appearing on the network.
- **Statistical anomaly:** Develops baselines of normal traffic activity and throughput, and alerts on deviations from those baselines.

Distributed or Hybrid IPS

The final category of IPS is in a distributed or hybrid approach. This gathers data from a large number of host and network-based sensors, relays this intelligence to a central analysis system able to correlate, and analyze the data, which can then return updated signatures and behavior patterns to enable all of the coordinated systems to respond and defend against malicious behavior. A number of such systems have been proposed. One of the best known is the digital immune system.

DIGITAL IMMUNE SYSTEM The digital immune system is a comprehensive defense against malicious behavior caused by malware, developed by IBM

[KEPH97a, KEPH97b, WHIT99], and subsequently refined by Symantec [SYMA01] and incorporated into its Central Quarantine produce [SYMA05]. The motivation for this development includes the rising threat of Internet-based malware, the increasing speed of its propagation provided by the Internet, and the need to acquire a global view of the situation.

In response to the threat posed by these Internet-based capabilities, IBM developed the original prototype digital immune system. This system expands on the use of sandbox analysis discussed in Section 6.10 and provides a general-purpose emulation and malware detection system. The objective of this system is to provide rapid response time so malware can be stamped out almost as soon as they are introduced. When new malware enters an organization, the immune system automatically captures it, analyzes it, adds detection and shielding for it, removes it, and passes information about it to client systems, so the malware can be detected before it is allowed to run elsewhere.

The success of the digital immune system depends on the ability of the malware analysis system to detect new and innovative malware strains. By constantly analyzing and monitoring malware found in the wild, it should be possible to continually update the digital immune software to keep up with the threat.

Figure 9.5 shows an example of a hybrid architecture designed originally to detect worms [SIDI05]. The system works as follows (numbers in figure refer to numbers in the following list):

1. Sensors deployed at various network and host locations detect potential malware scanning, infection, or execution. The sensor logic can also be incorporated in IDS sensors.
2. The sensors send alerts and copies of detected malware to a central server, which correlates and analyzes this information. The correlation server determines the likelihood that malware is being observed and its key characteristics.
3. The server forwards its information to a protected environment, where the potential malware may be sandboxed for analysis and testing.
4. The protected system tests the suspicious software against an appropriately instrumented version of the targeted application to identify the vulnerability.
5. The protected system generates one or more software patches and tests these.
6. If the patch is not susceptible to the infection and does not compromise the application's functionality, the system sends the patch to the application host to update the targeted application.

Snort Inline

We introduced Snort in Section 8.9 as a lightweight intrusion detection system. A modified version of Snort, known as Snort Inline [KURU12], enhances Snort to function as an intrusion prevention system. Snort Inline adds three new rule types that provide intrusion prevention features:

- **Drop:** Snort rejects a packet based on the options defined in the rule and logs the result.
- **Reject:** Snort rejects a packet and logs the result. In addition, an error message is returned. In the case of TCP, this is a TCP reset message, which resets the TCP

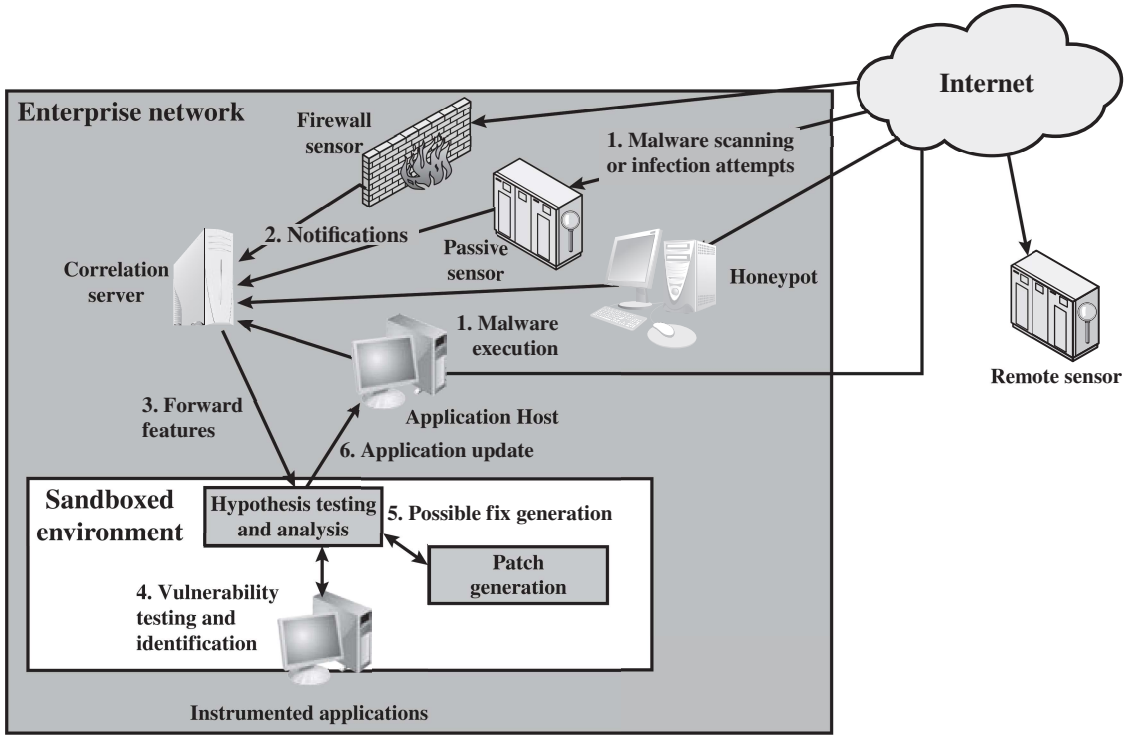

Figure 9.5 Placement of Malware Monitors
Source: Based on [SIDI05]. Sidiroglou, S., and Keromytis, A. "Countering Network Worms Through Automatic Patch Generation.", Columbia University, Figure 1, page 3, November-December 2005. http://www1.cs.columbia.edu/~angelos/Papers/2005/j6ker3.pdf IEEE.

connection. In the case of UDP, an ICMP port unreachable message is sent to the originator of the UDP packet.

- **Sdrop:** Snort rejects a packet but does not log the packet.

Snort Inline also includes a replace option, which allows the Snort user to modify packets rather than drop them. This feature is useful for a honeypot implementation [SPIT03]. Instead of blocking detected attacks, the honeypot modifies and disables them by modifying packet content. Attackers launch their exploits, which travel the Internet and hit their intended targets, but Snort Inline disables the attacks, which ultimately fail. The attackers see the failure but cannot figure out why it occurred. The honeypot can continue to monitor the attackers while reducing the risk of harming remote systems.

9.7 EXAMPLE: UNIFIED THREAT MANAGEMENT PRODUCTS

In the past few chapters, we have reviewed a number of approaches to countering malicious software and network-based attacks, including antivirus and antiworm products, IPS and IDS, and firewalls. The implementation of all of these systems can provide an organization with a defense in depth using multiple layers of filters and defense mechanisms to thwart attacks. The downside of such a piecemeal implementation is the need to configure, deploy, and manage a range of devices and software packages. In addition, deploying a number of devices in sequence can reduce performance.