CHAPTER 21

NETWORK ENDPOINT SECURITY

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.
- Understand the relative merits of various choices for firewall location and configurations.
- Understand the basic principles of and requirements for intrusion detection.
- Discuss the key features of intrusion detection systems.
- Describe some of the main categories of malicious software.
- Present an overview of the key elements of malware defense.
- Discuss the nature of a distributed denial of service attack.

This chapter focuses on security threats directed at endpoints, such as servers, workstations, and mobile devices, that are attached to an enterprise network or the Internet. Detailed discussion of the countermeasures implemented on the endpoints, such as antivirus software, is beyond our scope. Instead, this chapter looks at endpoint security from a network perspective.

The chapter begins with a discussion of firewalls. 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.

Section 21.2 deals with intrusion detection systems, while Section 21.3 provides an overview of malicious software. The last section discusses the important topic of distributed denial of service.

21.1 FIREWALLS

The firewall is an important complement to host-based security services such as intrusion detection systems. Typically, a 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. Firewalls are also deployed internal to the enterprise network to segregate portions of the network.

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

Firewall Characteristics

[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 section.
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 (OS). Trusted computer systems are suitable for hosting a firewall and are often required in government applications.

In general terms, there are four techniques that firewalls use to control access and enforce the site's security policy. Originally, firewalls focused primarily on service control, but they have since evolved to provide all four:

- **Service control:** Determines the types of Internet services that can be accessed, inbound or outbound. The firewall may filter traffic on the basis of IP address, protocol, or port number; may provide proxy software that receives and interprets each service request before passing it on; or may host the server software itself, such as a Web or mail service.
- **Direction control:** Determines the direction in which particular service requests may be initiated and allowed to flow through the firewall.
- **User control:** Controls access to a service according to which user is attempting to access it. This feature is typically applied to users inside the firewall perimeter (local users). It may also be applied to incoming traffic from external users; the latter requires some form of secure authentication technology, such as the one provided in IPsec.
- **Behavior control:** Controls how particular services are used. For example, the firewall may filter email to eliminate spam, or it may enable external access to only a portion of the information on a local Web server.

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 keeps unauthorized users out of the protected network, prohibits potentially vulnerable services from entering or leaving the network, and provides 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 implementing virtual private networks. This is discussed in the following section.

Firewalls have their limitations, including the following:

1. The firewall cannot protect against attacks that bypass the firewall. Internal systems may have dial-out capability to connect to an ISP. An internal LAN may support a modem pool that provides dial-in capability for traveling employees and telecommuters.
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, smartphone, or portable storage device may be used and infected outside the corporate network, and then connected and used internally.

Types of Firewalls

A firewall may act as a packet filter. It can operate as a positive filter, allowing to pass only packets that meet specific criteria, or as a negative filter, rejecting any packet that meets certain criteria. 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 forwards or discards the packet (Figure 21.1b). 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 TELNET
- **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 which interface of the firewall the packet is destined for

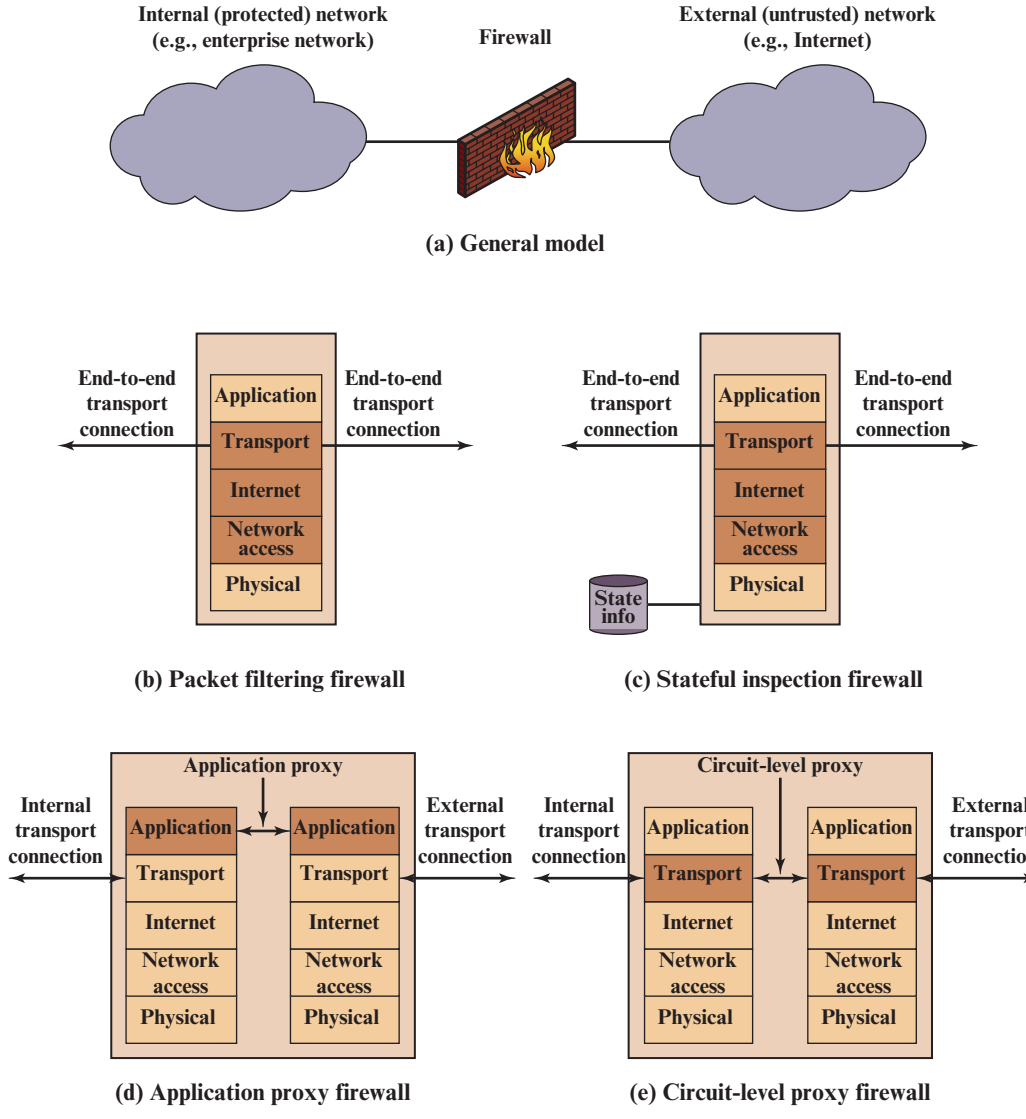


Figure 21.1 Types of Firewalls

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.

Figure 21.2 gives some examples of packet filtering rule sets. In each set, the rules are applied top to bottom. The "*" in a field is a wildcard designator that matches everything. We assume that the default = discard policy is in force. The rule sets can be described as follows:

A. Inbound mail is allowed (port 25 is for SMTP incoming), but only to a gateway host. However, packets from a particular external host, SPIGOT, are blocked because that host has a history of sending massive files in e-mail messages.

B. This is an explicit statement of the default policy. All rule sets include this rule implicitly as the last rule.

Rule Set A

action	Ourhost	port	theirhost	port	comment
block	*	*	SPIGOT	*	we don't trust these people
allow	OUR-GW	25	*	*	connection to our SMTP port

Rule Set B

action	Ourhost	port	theirhost	port	comment
block	*	*	*	*	default

Rule Set C

action	Ourhost	port	theirhost	port	comment
allow	*	*	*	25	connection to their SMTP port

Rule Set D

action	Src	port	dest	port	flags	comment
allow	{our hosts}	*	*	25		our packets to their SMTP port
allow	*	25	*	*	ACK	their replies

Rule Set E

action	Src	port	dest	port	flags	comment
allow	{our hosts}	*	*	*		our outgoing calls
allow	*	*	*	*	ACK	replies to our calls
allow	*	*	*	>1024		traffic to nonservers

Figure 21.2 Packet-Filtering Example

C. This rule set is 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 this rule is written, an attacker could gain access to internal machines by sending packets with a TCP source port number of 25.

D. This rule set achieves the intended result that was not achieved in C. The rules take 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 set states that it allows IP packets where the source IP address is one of a list of designated internal hosts and the destination TCP port number is 25. It also allows incoming packets with a source port number of 25 that include the ACK flag in the TCP segment. Note that we explicitly designate source and destination systems to define these rules explicitly.

E. This rule set is one approach to handling FTP connections. With FTP, two TCP connections are used: a control connection to set up the file transfer and a data connection for the actual file transfer. The data connection uses a different port number that is dynamically assigned for the transfer. Most servers, and hence most attack targets, use low-numbered ports; most outgoing calls tend to use a higher-numbered port, typically above 1023. Thus, this rule set allows

—Packets that originate internally

—Reply packets to a connection initiated by an internal machine

—Packets destined for a high-numbered port on an internal machine

This scheme requires that the systems be configured so that only the appropriate port numbers are in use.

Rule set E points out the difficulty in dealing with applications at the packet filtering level. Another way to deal with FTP and similar applications is either stateful filters or an application-level gateway, both described subsequently in this section.

One advantage of a packet filtering firewall is its simplicity. Also, packet filters typically are transparent to users and are very fast. However, packet filters have the following weaknesses:

- Because packet filter firewalls do not examine upper-layer data, they cannot prevent attacks that employ application-specific vulnerabilities or functions. For example, if a packet filter firewall cannot block specific application commands and 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. The 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 that the filtering firewall examines only the first fragment and that 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 Simple Mail Transfer Protocol (SMTP), email is transmitted from a client system to a server system. The client system generates new email messages, typically from user input. The server system accepts incoming email 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 inspection packet firewall tightens up the rules for TCP traffic by creating a directory of outbound TCP connections, as shown in Table 21.1. 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.

A stateful packet inspection firewall reviews the same packet information as a packet filtering firewall, but also records information about TCP connections (Figure 21.1c). 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 like 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 (Figure 21.1d). 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

Table 21.1 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

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** (Figure 21.1e). 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.

DMZ Networks

Figure 21.3 suggests the most common distinction, that between an internal and an external firewall. An external firewall is placed at the edge of a local or enterprise network, just inside the boundary router that connects to the Internet or some wide area network (WAN). One or more internal firewalls protect the bulk of the enterprise network. Between these two types of firewalls are one or more networked devices in a region referred to as a demilitarized zone (DMZ) network. Systems that are externally accessible but need some protections are usually located on DMZ networks. Typically, the systems in the DMZ require or foster external connectivity, such as a corporate Web site, an email server, or a domain name system (DNS) server.

The external firewall provides a measure of access control and protection for the DMZ systems consistent with their need for external connectivity. The external firewall also provides a basic level of protection for the remainder of the enterprise network. In this type of configuration, internal firewalls serve three purposes:

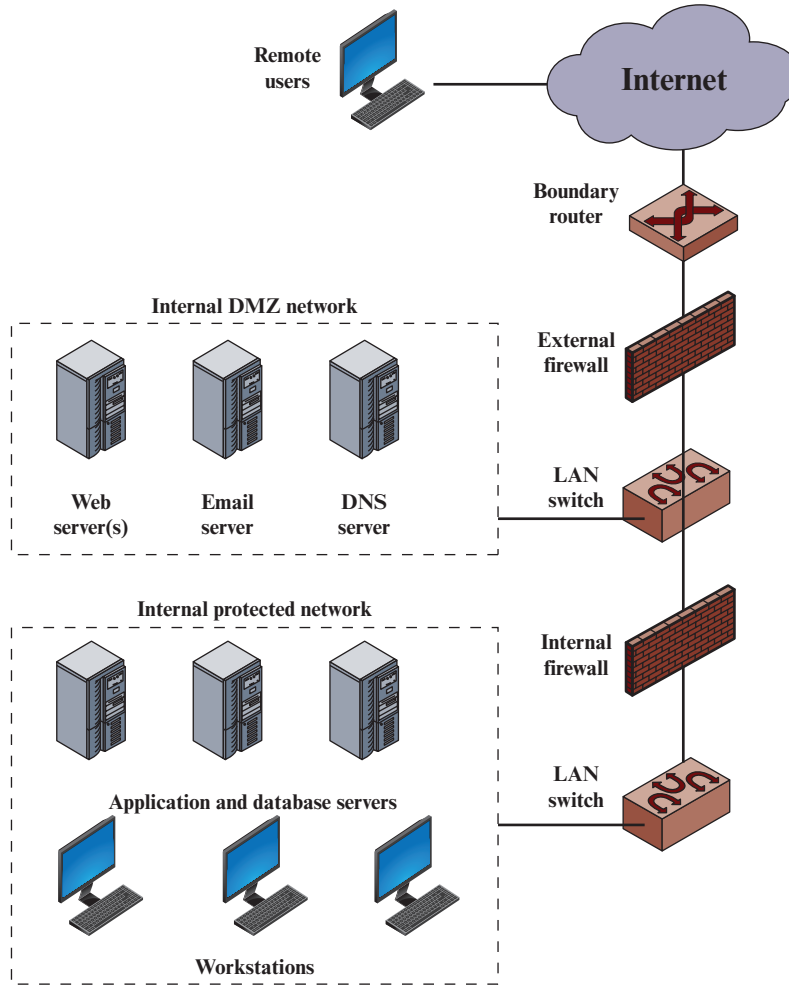


Figure 21.3 Example Firewall Configuration

1. The internal firewall adds more stringent filtering capability, compared to the external firewall, in order to protect enterprise servers and workstations from external attack.
2. The internal firewall provides two-way protection with respect to the DMZ. First, the internal firewall protects the remainder of the network from attacks launched from DMZ systems. Such attacks might originate from worms, rootkits, bots, or other malware lodged in a DMZ system. Second, an internal firewall can protect the DMZ systems from attack from the internal protected network.
3. Multiple internal firewalls can be used to protect portions of the internal network from each other. For example, firewalls can be configured so that internal servers are protected from internal workstations and vice versa. A common practice is to place the DMZ on a different network interface on the external firewall from that used to access the internal networks.

21.2 INTRUSION DETECTION SYSTEMS

It is useful to begin this section by defining the following terms:

- **Intrusion:** Violations of security policy, usually characterized as attempts to affect the confidentiality, integrity, or availability of a computer or network. These violations can come from attackers accessing systems from the Internet or from authorized users of the systems who attempt to overstep their legitimate authorization levels or who use their legitimate access to the system to conduct unauthorized activity.
- **Intrusion detection:** The process of collecting information about events occurring in a computer system or network and analyzing them for signs of intrusions.
- **Intrusion detection system:** Hardware or software products that gather and analyze information from various areas within a computer or a network for the purpose of finding, and providing real-time or near-real-time warning of, attempts to access system resources in an unauthorized manner.

Intrusion detection systems (IDSs) can be classified as follows:

- **Host-based IDS:** Monitors the characteristics of a single host and the events occurring within that host for suspicious activity. This vantage point allows host-based IDSs to determine exactly which processes and user accounts are involved in a particular attack on the OS. Furthermore, unlike network-based IDSs, host-based IDSs can more readily see the intended outcome of an attempted attack, because they can directly access and monitor the data files and system processes usually targeted by attacks.
- **Network-based IDS:** Monitors network traffic for particular network segments or devices and analyzes network, transport, and application protocols to identify suspicious activity.

An IDS comprises three logical components:

- **Sensors:** Sensors are responsible for collecting data. The input for a sensor may be any part of a system that could contain evidence of an intrusion. Types of input to a sensor include network packets, log files, and system call traces. Sensors collect and forward this information to the analyzer.
- **Analyzers:** Analyzers receive input from one or more sensors or from other analyzers. The analyzer is responsible for determining if an intrusion has occurred. The output of this component is an indication that an intrusion has occurred. The output may include evidence supporting the conclusion that an intrusion occurred. The analyzer may provide guidance about what actions to take as a result of the intrusion.
- **User interface:** The user interface to an IDS enables a user to view output from the system or control the behavior of the system. In some systems, the user interface may equate to a manager, director, or console component.

Basic Principles

Authentication facilities, access control facilities, and firewalls all play a role in countering intrusions. Another line of defense is intrusion detection, and this has been the focus of much research in recent years. This interest is motivated by a number of considerations, including the following:

1. If an intrusion is detected quickly enough, the intruder can be identified and ejected from the system before any damage is done or any data are compromised. Even if the detection is not sufficiently timely to preempt the intruder, the sooner that the intrusion is detected, the less the amount of damage and the more quickly that recovery can be achieved.
2. An effective IDS can serve as a deterrent, thus acting to prevent intrusions.
3. Intrusion detection enables the collection of information about intrusion techniques that can be used to strengthen intrusion prevention measures.

Approaches to Intrusion Detection

Intrusion detection assumes that the behavior of the intruder differs from that of a legitimate user in ways that can be quantified. Of course, we cannot expect that there will be a crisp, exact distinction between an attack by an intruder and the normal use of resources by an authorized user. Rather, we must expect that there will be some overlap.

There are two general approaches to intrusion detection: misuse detection and anomaly detection (Figure 21.4).

Misuse detection is based on rules that specify system events, sequences of events, or observable properties of a system that are believed to be symptomatic of security incidents. Misuse detectors use various pattern-matching algorithms, operating on large databases of attack patterns, or *signatures*. An advantage of misuse detection is that it is accurate and generates few false alarms. A disadvantage is that it cannot detect novel or unknown attacks.

Anomaly detection searches for activity that is different from the normal behavior of system entities and system resources. An advantage of anomaly detection is that it is able to detect previously unknown attacks based on an audit of activity. A disadvantage is that there is a significant trade-off between false positives and false negatives. Figure 21.5 suggests, in abstract terms, the nature of the task confronting the designer of an anomaly detection system. Although the typical behavior of an intruder differs from the typical behavior of an authorized user, there is an overlap

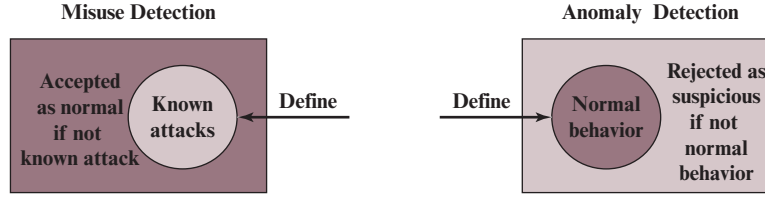


Figure 21.4 Approaches to Intrusion Detection

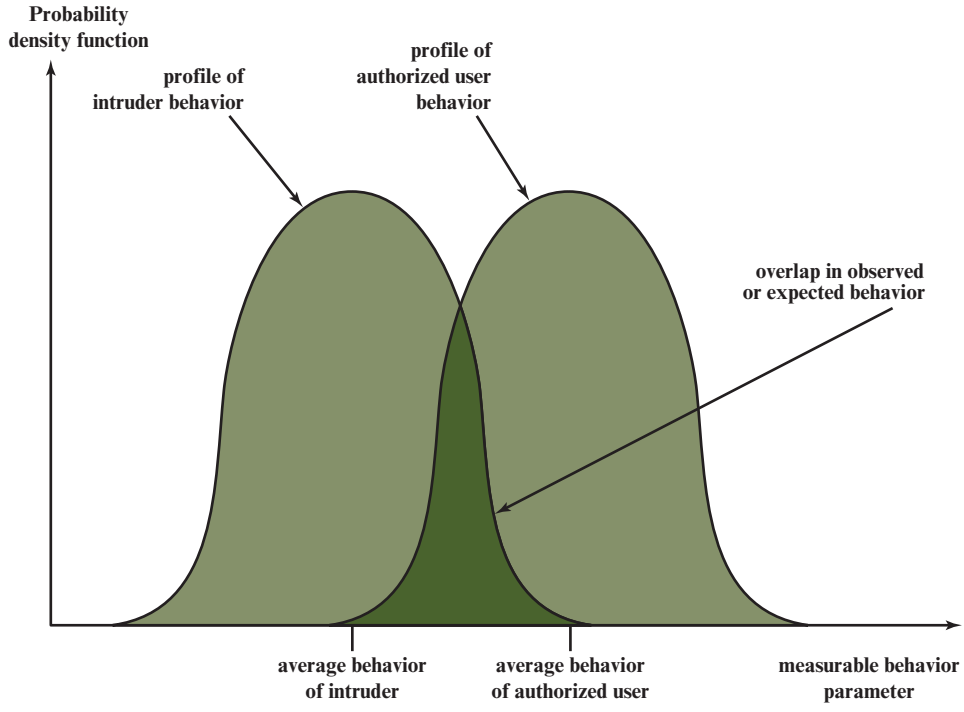


Figure 21.5 Profiles of Behavior of Intruders and Authorized Users

Table 21.2 Test Outcomes

Test Result	Condition A Occurs	Condition A Does Not Occur
Test says "A"	True positive	False positive
Test says "NOT A"	False negative	True negative

in these behaviors. Thus, a loose interpretation of intruder behavior, which will catch more intruders, will also lead to a number of **false positives**, or authorized users identified as intruders. On the other hand, an attempt to limit false positives by a tight interpretation of intruder behavior will lead to an increase in **false negatives**, or intruders not identified as intruders. Thus, there is an element of compromise and art in the practice of anomaly detection.

Table 21.2 clarifies the relationship between the terms false positive, true positive, false negative, and true negative.

Host-Based Intrusion Detection Techniques

Host-based IDSs add a specialized layer of security software to vulnerable or sensitive systems; examples include database servers and administrative systems. The host-based IDS monitors activity on the system in a variety of ways to detect suspicious behavior. In some cases, an IDS can halt an attack before any damage

is done, but its primary purpose is to detect intrusions, log suspicious events, and send alerts.

The primary benefit of a host-based IDS is that it can detect both external and internal intrusions, something that is not possible either with network-based IDSs or firewalls.

Host-based IDSs use one or a combination of anomaly and misuse protection. For anomaly detection, two common strategies are:

- **Threshold detection:** This approach involves defining thresholds, independent of user, for the frequency of occurrence of various events.
- **Profile based:** A profile of the activity of each user is developed and used to detect changes in the behavior of individual accounts.

Network-Based Intrusion Detection Systems

A network-based ID system (NIDS) monitors the traffic on its network segment as a data source. This is generally accomplished by placing the network interface card in promiscuous mode to capture all network traffic that crosses its network segment. Network traffic on other segments, and traffic on other means of communication (like phone lines), can't be monitored by a single NIDS.

NIDS Function Network-based ID involves looking at the packets on the network as they pass by some sensor. Packets are considered to be of interest if they match a signature. Three primary types of signatures are string signatures, port signatures, and header condition signatures.

String signatures look for a text string that indicates a possible attack. An example string signature for UNIX might be "cat "+ +" >/.rhosts", which if successful, might cause a UNIX system to become extremely vulnerable to network attack. To refine the string signature to reduce the number of false positives, it may be necessary to use a compound string signature. A compound string signature for a common Web server attack might be "cgi-bin" AND "aglimpse" AND "IFS".

Port signatures simply watch for connection attempts to well known, frequently attacked ports. Examples of these ports include telnet (TCP port 23), FTP (TCP port 21/20), SUNRPC (TCP/UDP port 111), and IMAP (TCP port 143). If any of these ports aren't used by the site, then incoming packets to these ports are suspicious.

Header signatures watch for dangerous or illogical combinations in packet headers. The most famous example is WinNuke, where a packet is destined for a NetBIOS port and the Urgent pointer, or Out Of Band pointer is set. This resulted in the "blue screen of death" for Windows systems. Another well-known header signature is a TCP packet with both the SYN and FIN flags set, signifying that the requestor wishes to start and stop a connection at the same time.

NIDS Placement An NIDS sensor can only see the packets that happen to be carried on the network segment to which it is attached. Accordingly, a NIDS deployment is typically set up as a number of sensors distributed on key network points to passively gather traffic data and feed information on potential threats to a central

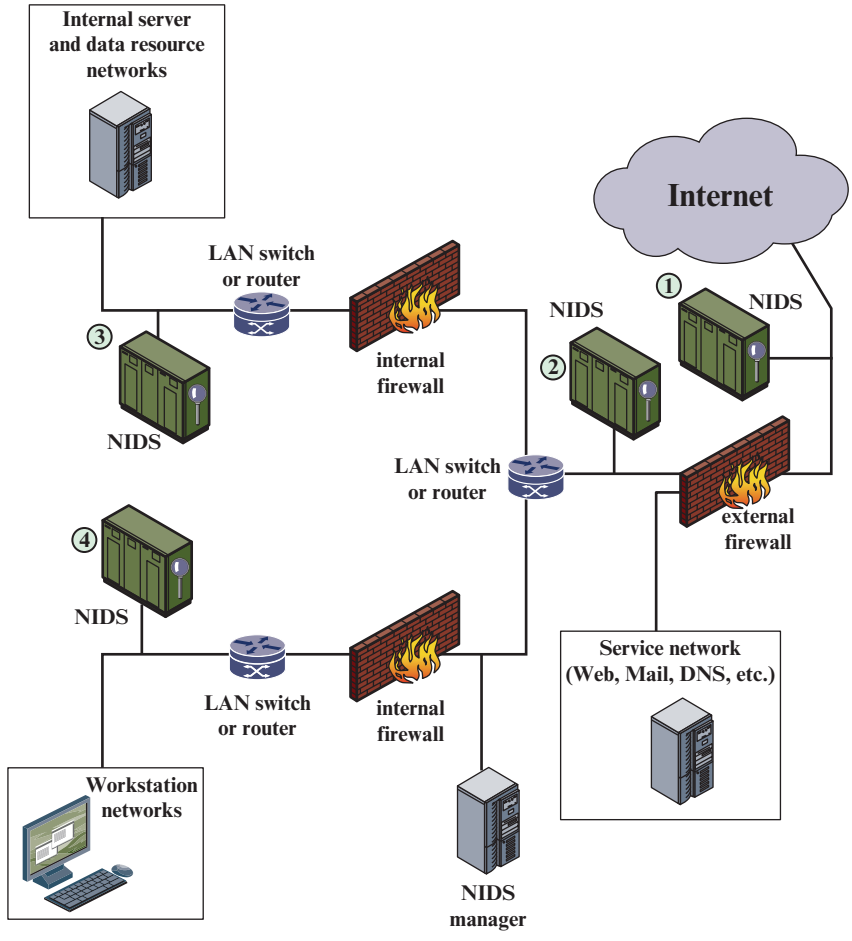


Figure 21.6 Example of NIDS Sensor Deployment

NIDS manager. Figure 21.6 gives examples of NIDS sensor placement. There are four types of locations for the sensors:

1. Outside the main enterprise firewall. Useful for establishing the level of threat for a given enterprise network. Those responsible for winning management support for security efforts can find this placement valuable.
2. In the network DMZ (inside the main firewall but outside internal firewalls). This location can monitor for penetration attempts that target Web and other services generally open to outsiders.
3. Behind internal firewalls, positioned to monitor major backbone networks, such as those that support internal servers and database resources.
4. Behind internal firewalls, positioned to monitor LANs that support user workstations and servers specific to a single department. Locations 3 and 4 in Figure 21.6 can monitor for more specific attacks at network segments, as well as attacks originating from inside the organization.

21.3 MALICIOUS SOFTWARE

Malicious software, commonly called **malware**, is perhaps the most significant security threat to organizations. NIST SP 800-83 (*Guide to Malware Incident Prevention and Handling for Desktops and Laptops*) defines malware as "a program that is covertly inserted into another program with the intent to destroy data, run destructive or intrusive programs, or otherwise compromise the confidentiality, integrity, or availability of the victim's data, applications, or operating system." Hence, malware can pose a threat to application programs, to utility programs, such as editors and compilers, and to kernel-level programs. Malware can also be used on compromised or malicious Web sites and servers, or in especially crafted spam emails or other messages, which aim to trick users into revealing sensitive personal information.

Types of Malware

There is a growing variety of types of malware, most of which fits into one of the following broad categories:

- **Virus:** A computer program that can copy itself and infect a computer without permission or knowledge of the user. A virus might corrupt or delete data on a computer, use email programs to spread itself to other computers, or even erase everything on a hard disk. It can replicate itself and can attach to another program. The program to which the virus attaches itself is known as host.
- **Worm:** A self-replicating, self-propagating, self-contained program that uses networking mechanisms to spread itself. The main differences between viruses and worms is that the worms can self-replicate and propagate without human interaction and that the worm does not integrate into existing code. Worms target systems and applications that have known vulnerabilities.
- **Trojan Horse:** A computer program that appears to have a useful function, but also has a hidden and potentially malicious function that evades security mechanisms, sometimes by exploiting legitimate authorizations of a system entity that invokes the program. As the name suggests, the purpose of a Trojan horse is to make a malicious program appear like a legitimate program. Trojan horse can monitor users' action, steal users' data, and can open a backdoor for the attackers.
- **Spyware:** Software that is secretly or surreptitiously installed into an information system to gather information on individuals or organizations without their knowledge.
- **Rootkit:** A set of tools used by an attacker after gaining root-level access to a host to conceal the attacker's activities on the host and permit the attacker to maintain root-level access to the host through covert means.
- **Backdoor:** An undocumented way of gaining access to a computer system. Typically, a backdoor is a program that has the ability to bypass a system's security control, allowing an attacker to access the system stealthily. Backdoors are usually installed by the attackers or by a malware program.

- **Mobile code:** Software (e.g., script, macro, or other portable instruction) that can be shipped unchanged to a heterogeneous collection of platforms and execute with identical semantics.
- **Bot:** Also known as a zombie. Program that is installed on a system to launch attacks on other machines. For example, a distributed denial-of-service (DDoS) attack involves traffic from a number of infected bot machines to a single target, to overwhelm the resources of the target machine. A collection of bots that act in concert is referred to as a **botnet**.

Malware Defense

Approaches to malware defense are commonly categorized along two dimensions, as shown in Figure 21.7. In terms of time scale, there are two categories:

- **Real-time and Near-real-time:** Approaches in this category involve monitoring and, if possible, blocking malware-related attacks as they are happening or very soon thereafter. These approaches typically also involve remedial action, such as removing malware and reporting the incident.
- **Post-compromise:** Approaches in this category involve analysis of incident reports and traffic patterns to aid in improving security controls.

The remainder of this section provides an overview of the approaches shown in Figure 21.7.

Network Traffic Analysis Network traffic analysis involves monitoring traffic flows to detect potentially malicious activity. Such monitors are often placed at the boundary of the enterprise network to the outside world, such as the Internet or private networks. Monitors can also be placed on internal network devices or near server endpoints.

As with intrusion detection, traffic analysis can involve misuse detection (signature detection) or anomaly detection. As an example of misuse detection, a dramatic surge in traffic at any point likely indicates that a DDoS attack is underway. For anomaly detection, network security software needs to collect and maintain profiles of typical network traffic patterns, and then monitor current traffic for

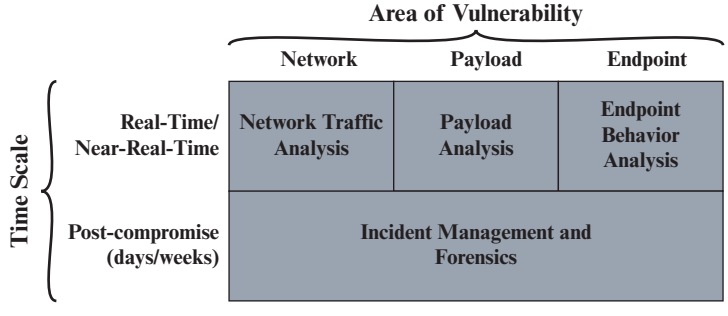


Figure 21.7 Five Elements of Malware Defense

significant deviation from normal behavior. For example, anomalous DNS (Domain Name System) traffic is a good indicator of botnet activity.

Payload Analysis The term payload refers to the data encapsulated within packets that has meaning to endpoint applications. As with traffic analysis, payload analysis is a real-time or near-real-time activity. It involves looking for known malicious payloads (signature detection) or looking for payload patterns that are anomalous. One useful technique for payload analysis is the use of a sandbox environment, which quarantines the payload until the analysis is done. This enables a payload analysis system to observe the behavior of payloads in motion, such as when they cross the network perimeter, and to either flag suspicious payloads or block them outright.

Endpoint Behavior Analysis This category involves a wide variety of tools and approaches implemented at the endpoint. Antivirus software uses signature and anomaly detection techniques to identify malware and prevent it from executing on the host system. Application whitelisting, which restricts application execution to only known good applications is also employed. At the system software level, application containers can isolate applications and files in virtual containers to prevent damage.

Incident Management Information security incident management as consisting of processes for detecting, reporting, assessing, responding to, dealing with, and learning from information security incidents.

Key elements of incident management include:

- **Data collection:** In a typical use case, an incident management system must be able to touch any number of different systems: firewalls, proxy servers, data bases, intrusion detection and prevention systems, OSs, routers, switches, access control systems, etc. Some of these may share similar logging and alert functions, but frequently there is significant variation in the format, protocol and information provided.
- **Data aggregation:** The aggregator serves as a consolidating resource before data is sent to be correlated or retained.
- **Data normalization:** Normalization is the process of resolving different representations of the same types of data into a similar format in a common database.
- **Correlation:** Event correlation is the function of linking multiple security events or alerts, typically within a given time window and across multiple systems, to identify anomalous activity that would not be evident from any singular event.
- **Alerting:** When data is gathered or identified that trigger certain responses, such as alerts or potential security problems, tools can activate certain protocols to alert users, like notifications sent to the dashboard, an automated email or text message.
- **Reporting/Compliance:** Protocols can be established that automatically collect data necessary for compliance with company, organizational, and government policies.

The goal is to analyze the security incidents both for purposes of improving system security and for updating signatures and anomaly profiles used for detection. This process applies both to malware-related attacks and to intrusions.

FORENSICS NIST SP 800-96 (*Guide to Integrating Forensic Techniques into Incident Response*) defines computer forensics, or digital forensics, as the identification, collection, examination, and analysis of data while preserving the integrity of the information and maintaining a strict chain of custody for the data. Computer forensics seeks to answer a number of questions including the following:

- What happened?
- When did the events occur?
- In what order did the events occur?
- What was the cause of these events?
- Who caused these events to occur?
- What enabled these events to take place?
- What was affected? How much was it affected?

Most security incidents do not require a forensic investigation but can be dealt with by the ordinary incident management process. But more serious incidents may warrant the more in-depth analysis of a forensic investigation.

21.4 DISTRIBUTED DENIAL OF SERVICE ATTACKS

A denial-of-service (DoS) attack is an attempt to prevent legitimate users of a service from using that service. When this attack comes from a single host or network node, then it is simply referred to as a DoS attack. A more serious threat is posed by a DDoS attack. DDoS attacks make computer systems inaccessible by flooding servers, networks, or even end-user systems with useless traffic so that legitimate users can no longer gain access to those resources. In a typical DDoS attack, a large number of compromised hosts are amassed to send useless packets.

This section is concerned with DDoS attacks. First, we look at the nature and types of attacks. Next, we examine methods by which an attacker is able to recruit a network of hosts for attack launch. Finally, this section looks at countermeasures.

DDoS Attack Description

A DDoS attack attempts to consume the target's resources so that it cannot provide service. One way to classify DDoS attacks is in terms of the type of resource that is consumed. Broadly speaking, the resource consumed is either an internal host resource on the target system or data transmission capacity in the local network to which the target is attacked.

A simple example of an internal resource attack is the SYN flood attack. Figure 21.8a shows the steps involved:

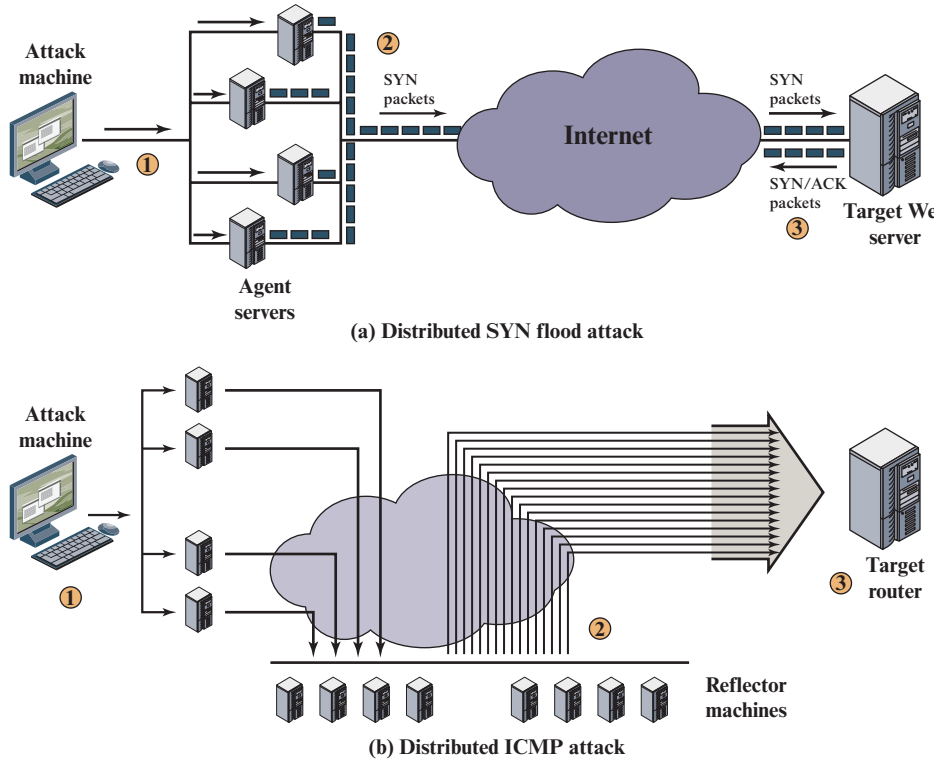


Figure 21.8 Examples of Simple DDoS Attacks

1. The attacker takes control of multiple hosts over the Internet, instructing them to contact the target Web server.
2. The agent hosts begin sending TCP/IP SYN (synchronize/initialization) packets, with erroneous return IP address information, to the target.
3. Each SYN packet is a request to open a TCP connection. For each such packet, the Web server responds with a SYN/ACK (synchronize/acknowledge) packet, trying to establish a TCP connection with a TCP entity at a spurious IP address. The Web server maintains a data structure for each SYN request waiting for a response back and becomes bogged down as more traffic floods in. The result is that legitimate connections are denied while the victim machine is waiting to complete bogus "half-open" connections.

The TCP state data structure is a popular internal resource target but by no means the only one. Other possibilities include the following:

1. An intruder may attempt to use up available data structures that are used by the OS to manage processes, such as process table entries and process control information entries. The attack can be quite simple, such as a program that forks new processes repeatedly.

2. An intruder may attempt to allocate to itself large amounts of disk space by a variety of straightforward means. These include generating numerous emails, forcing errors that trigger audit trails, and placing files in shareable areas.

Figure 21.8b illustrates an example of an attack that consumes data transmission resources. The following steps are involved:

1. The attacker takes control of multiple hosts over the Internet, instructing them to send ICMP ECHO packets[1] with the target's spoofed IP address to a group of hosts that act as reflectors, as described subsequently.
2. Nodes at the bounce site receive multiple spoofed requests and respond by sending echo reply packets to the target site.
3. The target's router is flooded with packets from the bounce site, leaving no data transmission capacity for legitimate traffic.

Another way to classify DDoS attacks is as either direct or reflector DDoS attacks. In a direct DDoS attack (Figure 21.9a), the attacker is able to implant zombie software on a number of sites distributed throughout the Internet. Often, the DDoS attack involves two levels of zombie machines: primary zombies and agent zombies. The hosts of both machines have been infected with malicious code. The attacker coordinates and triggers the primary zombies, which in turn coordinate and trigger the agent zombies. The use of two levels of zombies makes it more difficult to trace the attack back to its source and provides for a more resilient network of attackers.

A reflector DDoS attack adds another layer of machines (Figure 21.9b). In this type of attack, the agent zombies construct packets requiring a response that contain the target's IP address as the source IP address in the packet's IP header. These packets are sent to uninfected machines known as reflectors. The uninfected machines respond with packets directed at the target machine. A reflector DDoS attack can easily involve more machines and more traffic than a direct DDoS attack and hence be more damaging. Further, tracing back the attack or filtering out the attack packets is more difficult because the attack comes from widely dispersed uninfected machines.

Constructing the Attack Network

The first step in a DDoS attack is for the attacker to infect a number of machines with zombie software that will ultimately be used to carry out the attack. The essential ingredients in this phase of the attack are the following:

1. Software that can carry out the DDoS attack. The software must be able to run on a large number of machines, must be able to conceal its existence, must be able to communicate with the attacker or have some sort of time-triggered mechanism, and must be able to launch the intended attack toward the target.

[1]The Internet Control Message Protocol (ICMP) is an IP-level protocol for the exchange of control packets between a router and a host or between hosts. The ECHO packet requires the recipient to respond with an echo reply to check that communication is possible between entities.

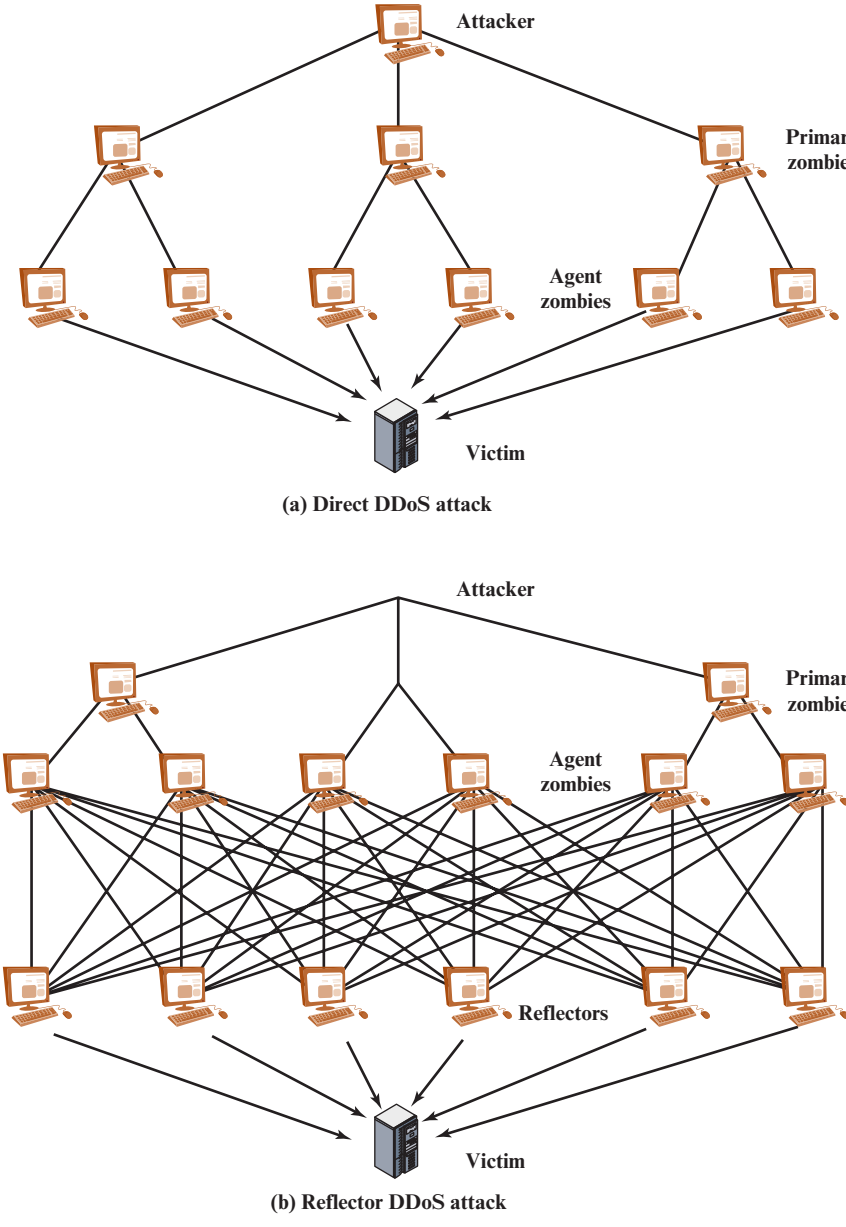


Figure 21.9 Types of Flooding-Based DDoS Attacks

2. A vulnerability in a large number of systems. The attacker must become aware of a vulnerability that many system administrators and individual users have failed to patch and that enables the attacker to install the zombie software.
3. A strategy for locating vulnerable machines, a process known as scanning.

In the scanning process, the attacker first seeks out a number of vulnerable machines and infects them. Then, typically, the zombie software that is installed in the infected machines repeats the same scanning process, until a large distributed network of infected machines is created. [MIRK04] lists the following types of scanning strategies:

- **Random:** Each compromised host probes random addresses in the IP address space, using a different seed. This technique produces a high volume of Internet traffic, which may cause generalized disruption even before the actual attack is launched.
- **Hit list:** The attacker first compiles a long list of potential vulnerable machines. This can be a slow process done over a long period to avoid detection that an attack is underway. Once the list is compiled, the attacker begins infecting machines on the list. Each infected machine is provided with a portion of the list to scan. This strategy results in a very short scanning period, which may make it difficult to detect that infection is taking place.
- **Topological:** This method uses information contained on an infected victim machine to find more hosts to scan.
- **Local subnet:** If a host can be infected behind a firewall, that host then looks for targets in its own local network. The host uses the subnet address structure to find other hosts that would otherwise be protected by the firewall.

DDoS Countermeasures

In general, there are three lines of defense against DDoS attacks:

- **Attack prevention and preemption (before the attack):** These mechanisms enable the victim to endure attack attempts without denying service to legitimate clients. Techniques include enforcing policies for resource consumption and providing backup resources available on demand. In addition, prevention mechanisms modify systems and protocols on the Internet to reduce the possibility of DDoS attacks.
- **Attack detection and filtering (during the attack):** These mechanisms attempt to detect the attack as it begins and respond immediately. This minimizes the impact of the attack on the target. Detection involves looking for suspicious patterns of behavior. Response involves filtering out packets likely to be part of the attack.
- **Attack source traceback and identification (during and after the attack):** This is an attempt to identify the source of the attack as a first step in preventing future attacks. However, this method typically does not yield results fast enough, if at all, to mitigate an ongoing attack.

The challenge in coping with DDoS attacks is the sheer number of ways in which they can operate. Thus, DDoS countermeasures must evolve with the threat.

21.5 KEY TERMS, REVIEW QUESTIONS, AND PROBLEMS

Key Terms

anomaly detection	circuit-level proxy	malware
application proxy	false negatives	misuse detection
botnet	false positives	

Review Questions

21.1 List three design goals for a firewall.

21.2 List four techniques used by firewalls to control access and enforce a security policy.

21.3 What information is used by a typical packet filtering firewall?

21.4 What are some weaknesses of a packet filtering firewall?

21.5 What is the difference between a packet filtering firewall and a stateful inspection firewall?

21.6 What is an application-level gateway?

21.7 What is a circuit-level gateway?

21.8 What is a DMZ network and what types of systems would you expect to find on such networks?

21.9 What is the difference between an internal and an external firewall?

21.10 Explain the difference between host-based and network-based intrusion detection systems.

21.11 What are the main logical components of an IDS?

21.12 What are the two main approaches to intrusion detection?

21.13 List the main categories of malicious software.

21.14 Explain the difference between network traffic analysis, payload analysis, and endpoint behavior analysis.

21.15 What is a distributed denial-of-service system?

Problems

21.1 As was mentioned in Section 21.1, one approach to defeating the tiny fragment attack is to enforce a minimum length of the transport header that must be contained in the first fragment of an IP packet. If the first fragment is rejected, all subsequent fragments can be rejected. However, the nature of IP is such that fragments may arrive out of order. Thus, an intermediate fragment may pass through the filter before the initial fragment is rejected. How can this situation be handled?

21.2 In an IPv4 packet, the size of the payload in the first fragment, in octets, is equal to Total Length $-$ (4 $\times$ IHL). If this value is less than the required minimum (8 octets for TCP), then this fragment and the entire packet are rejected. Suggest an alternative method of achieving the same result using only the Fragment Offset field.

21.3 RFC 791, the IPv4 protocol specification, describes a reassembly algorithm that results in new fragments overwriting any overlapped portions of previously received fragments. Given such a reassembly implementation, an attacker could construct a series of packets in which the lowest (zero-offset) fragment would contain innocuous data (and thereby be passed by administrative packet filters), and in which some subsequent packet having a nonzero offset would overlap TCP header information (destination

Table 21.3 Sample Packet Filter Firewall Ruleset

	Source Address	Source Port	Destination Address	Destination Port	Action
1	Any	Any	192.168.1.0	> 1023	Allow
2	192.168.1.1	Any	Any	Any	Deny
3	Any	Any	192.168.1.1	Any	Deny
4	192.168.1.0	Any	Any	Any	Allow
5	Any	Any	192.168.1.2	SMTP	Allow
6	Any	Any	192.168.1.3	HTTP	Allow
7	Any	Any	Any	Any	Deny

port, for instance) and cause it to be modified. The second packet would be passed through most filter implementations because it does not have a zero fragment offset. Suggest a method that could be used by a packet filter to counter this attack.

21.4 Table 21.3 shows a sample of a packet filter firewall ruleset for an imaginary network of IP address that range from 192.168.1.0 to 192.168.1.254. Describe the effect of each rule.

21.5 SMTP (Simple Mail Transfer Protocol) is the standard protocol for transferring mail between hosts over TCP. A TCP connection is set up between a user agent and a server program. The server listens on TCP port 25 for incoming connection requests. The user end of the connection is on a TCP port number above 1023. Suppose you wish to build a packet filter rule set allowing inbound and outbound SMTP traffic. You generate the following rule set:

Rule	Direction	Src Addr	Dest Addr	Protocol	Dest Port	Action
A	In	External	Internal	TCP	25	Permit
B	Out	Internal	External	TCP	>1023	Permit
C	Out	Internal	External	TCP	25	Permit
D	In	External	Internal	TCP	>1023	Permit
E	Either	Any	Any	Any	Any	Deny

a. Describe the effect of each rule.

b. Your host in this example has IP address 172.16.1.1. Someone tries to send email from a remote host with IP address 192.168.3.4. If successful, this generates an SMTP dialogue between the remote user and the SMTP server on your host consisting of SMTP commands and mail. Additionally, assume that a user on your host tries to send email to the SMTP server on the remote system. Four typical packets for this scenario are as shown:

Packet	Direction	Src Addr	Dest Addr	Protocol	Dest Port	Action
1	In	192.168.3.4	172.16.1.1	TCP	25	?
2	Out	172.16.1.1	192.168.3.4	TCP	1234	?
3	Out	172.16.1.1	192.168.3.4	TCP	25	?
4	In	192.168.3.4	172.16.1.1	TCP	1357	?

Indicate which packets are permitted or denied and which rule is used in each case.

c. Someone from the outside world (10.1.2.3) attempts to open a connection from port 5150 on a remote host to the Web proxy server on port 8080 on one of your

local hosts (172.16.3.4), in order to carry out an attack. Typical packets are as follows:

Packet	Direction	Src Addr	Dest Addr	Protocol	Dest Port	Action
5	In	10.1.2.3	172.16.3.4	TCP	8080	?
6	Out	172.16.3.4	10.1.2.3	TCP	5150	?

Will the attack succeed? Give details.

21.6 To provide more protection, the rule set from the preceding problem is modified as follows:

Rule	Direction	Src Addr	Dest Addr	Protocol	Src Port	Dest Port	Action
A	In	External	Internal	TCP	>1023	25	Permit
B	Out	Internal	External	TCP	25	>1023	Permit
C	Out	Internal	External	TCP	>1023	25	Permit
D	In	External	Internal	TCP	25	>1023	Permit
E	Either	Any	Any	Any	Any	Any	Deny

a. Describe the change.

b. Apply this new rule set to the same six packets of the preceding problem. Indicate which packets are permitted or denied and which rule is used in each case.

21.7 A hacker uses port 25 as the client port on his or her end to attempt to open a connection to your Web proxy server.

a. The following packets might be generated:

Packet	Direction	Src Addr	Dest Addr	Protocol	Src Port	Dest Port	Action
7	In	10.1.2.3	172.16.3.4	TCP	25	8080	?
8	Out	172.16.3.4	10.1.2.3	TCP	8080	25	?

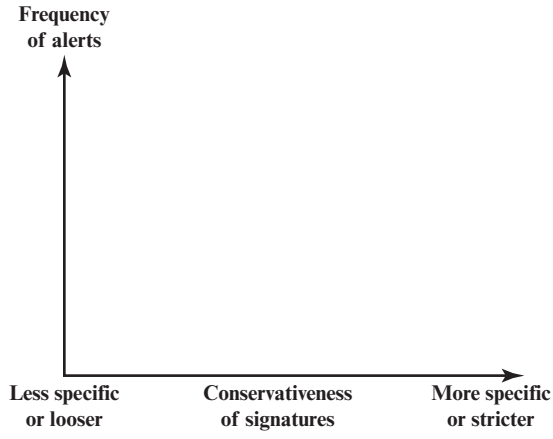
Explain why this attack will succeed, using the rule set of the preceding problem.

b. When a TCP connection is initiated, the ACK bit in the TCP header is not set. Subsequently, all TCP headers sent over the TCP connection have the ACK bit set. Use this information to modify the rule set of the preceding problem to prevent the attack just described.

21.8 A common management requirement is that "all external Web traffic must flow via the organization's Web proxy." However, that requirement is easier stated than implemented. Discuss the various problems and issues, possible solutions, and limitations supporting this requirement. In particular, consider issues such as identifying exactly what constitutes "Web traffic" and how it may be monitored, given the large range of ports and various protocols used by Web browsers and servers.

21.9 Consider the threat of "theft/breach of proprietary or confidential information held in key data files on the system." One method by which such a breach might occur is the accidental/deliberate emailing of information to a user outside to the organization. A possible countermeasure to this is to require all external email to be given a sensitivity tag (classification if you like) in its subject and for external e-mail to have the lowest sensitivity tag. Discuss how this measure could be implemented in a firewall and what components and architecture would be needed to do this.

21.10 In the context of an IDS, we define a false positive to be an alarm generated by an IDS in which the IDS alerts to a condition that is actually benign. A false negative occurs when an IDS fails to generate an alarm when an alert-worthy condition is in effect.



Using the above diagram, depict two curves that roughly indicate false positives and false negatives, respectively.

21.11 The overlapping area of the two probability density functions of Figure 21.5 represents the region in which there is the potential for false positives and false negatives. Further, Figure 21.5 is an idealized and not necessarily representative depiction of the relative shapes of the two density functions. Suppose there is 1 actual intrusion for every 1000 authorized users, and the overlapping area covers 1% of the authorized users and 50% of the intruders.

a. Sketch such a set of density functions and argue that this is not an unreasonable depiction.

b. What is the probability that an event that occurs in this region is that of an authorized user? Keep in mind that 50% of all intrusions fall in this region.

21.12 An example of a host-based intrusion detection tool is the tripwire program. This is a file integrity checking tool that scans files and directories on the system on a regular basis and notifies the administrator of any changes. It uses a protected database of cryptographic checksums for each file checked and compares this value with that recomputed on each file as it is scanned. It must be configured with a list of files and directories to check, and what changes, if any, are permissible to each. It can allow, for example, log files to have new entries appended, but not for existing entries to be changed. What are the advantages and disadvantages of using such a tool? Consider the problem of determining which files should only change rarely, which files may change more often and how, and which change frequently and hence cannot be checked. Hence consider the amount of work in both the configuration of the program and on the system administrator monitoring the responses generated.

21.13 A taxicab was involved in a fatal hit-and-run accident at night. Two cab companies, the Green and the Blue, operate in the city. You are told that:

- 85% of the cabs in the city are Green and 15% are Blue.
- A witness identified the cab as Blue.

The court tested the reliability of the witness under the same circumstances that existed on the night of the accident and concluded that the witness was correct in identifying the color of the cab 80% of the time. What is the probability that the cab involved in the incident was Blue rather than Green?

21.14 The question arises as to whether it is possible to develop a program that can analyze a piece of software to determine if it is a virus. Consider that we have a program D that is supposed to be able to do that. That is, for any program P, if we run D(P), the result returned is TRUE (P is a virus) or FALSE (P is not a virus). Now consider the following program:

```
Program CV:=
{…
main-program:=
{if D(CV) then goto next:
else infect-executable;
}
next:
}
```

In the preceding program, infect-executable is a module that scans memory for executable programs and replicates itself in those programs. Determine if D can correctly decide whether CV is a virus.