UNIT-3

CO2, CO3

7

Security in Networks

In this chapter:

- Networks vs. stand-alone applications and environments: differences and similarities
- Threats against networked applications, including denial of service, web site defacements, malicious mobile code, and protocol attacks
- Controls against network attacks: physical security, policies and procedures, and a range of technical controls
- Firewalls: design, capabilities, limitations
- Intrusion detection systems
- Private e-mail: PGP and S/MIME

Networks—their design, development, and usage—are critical to our style of computing. We interact with networks daily, when we perform banking transactions, make telephone calls, or ride trains and planes. The utility companies use networks to track electricity or water usage and bill for it. When we pay for groceries or gasoline, networks enable our credit or debit card transactions and billing. Life without networks would be considerably less convenient, and many activities would be impossible. Not surprisingly, then, computing networks are attackers' targets of choice. Because of their actual and potential impact, network attacks attract the attention of journalists, managers, auditors, and the general public. For example, when you read the daily newspapers, you are likely to find a story about a network-based attack at least every month. The coverage itself evokes a sense of evil, using terms such as hijacking, distributed denial of service, and our familiar friends viruses, worms, and Trojan horses. Because any large-scale attack is likely to put thousands of computing systems at risk, with potential losses well into the millions of dollars, network attacks make good copy.

The media coverage is more than hype; network attacks are critical problems. Fortunately, your bank, your utility company, and even your Internet service provider

take network security very seriously. Because they do, they are vigilant about applying the most current and most effective controls to their systems. Of equal importance, these organizations continually assess their risks and learn about the latest attack types and defense mechanisms so that they can maintain the protection of their networks.

In this chapter we describe what makes a network similar to and different from an application program or an operating system, which you have studied in earlier chapters. In investigating networks, you will learn how the concepts of confidentiality, integrity, and availability apply in networked settings. At the same time, you will see that the basic notions of identification and authentication, access control, accountability, and assurance are the basis for network security, just as they have been in other settings.

Networking is growing and changing perhaps even faster than other computing disciplines. Consequently, this chapter is unlikely to present you with the most current technology, the latest attack, or the newest defense mechanism; you can read about those in daily newspapers and at web sites. But the novelty and change build on what we know today: the fundamental concepts, threats, and controls for networks. By developing an understanding of the basics, you can absorb the most current news quickly and easily. More importantly, your understanding can assist you in building, protecting, and using networks.

7.1 NETWORK CONCEPTS

To study network threats and controls, we first must review some of the relevant networking terms and concepts. This review does not attempt to provide the depth of a classic networking reference, such as [COM04, STE02, GAL99, or TAN03]. In earlier chapters, our study of security focused on the individual pieces of a computing system, such as a single application, an operating system, or a database. Networks involve not only the pieces but also—importantly—the connections among them.

Networks are both fragile and strong. To see why, think about the power, cable television, telephone, or water network that serves your home. If a falling tree branch breaks the power line to your home, you are without electricity until that line is repaired; you are vulnerable to what is called a **single point of failure**, because one failure to the network destroys electrical functionality for your entire home. Similarly, there may be one telephone trunk line or water main that serves your home and those nearby; a failure can leave your building, street, or neighborhood without service. But we have ways to keep the entire network from failing. If we trace back through the network from your home to the source of what flows through it, we are likely to see that several main distribution lines support an entire city or campus. That is, there is more than one way to get from the source to your neighborhood, enabling engineers to redirect the flow along alternative paths. Redundancy makes it uncommon for an entire city to lose service from a single failure. For this reason, we say that such a network has **resilience** or **fault tolerance**.

Complex routing algorithms reroute the flow not just around failures but also around overloaded segments. The routing is usually done automatically; the com-

program is often supplemented by human supervision or intervention. Many types of networks have very high reliability by design, not by accident. But because there often is less redundancy near a network's endpoints than elsewhere, we say that the network has great strength in the middle and fragility at the perimeter.

From the user's perspective, a network is sometimes designed so that it looks like two endpoints with a single connection in the middle. For example, the municipal water supply may appear to be little more than a reservoir (the source), the pipes (the transmission or communication medium), and your water faucet (the destination). Although this simplistic view is functionally correct, it ignores the complex design, implementation, and management of the "pipes." In a similar way, we describe computer networks in this chapter in ways that focus on the security concepts but present the networks themselves in a simplistic way, to highlight the role of security and prevent the complexity of the networks from distracting our attention. Please keep in mind that our network descriptions are often abstractions of a more complex actuality.

The Network

Figure 7-1 shows a network in its simplest form, as two devices connected across some medium by hardware and software that enable the communication. In some cases, one device is a computer (sometimes called a "**server**") and the other is a simpler device (sometimes called a "**client**") enabled only with some means of input (such as a keyboard) and some means of output (such as a screen). For example, a powerful computer can be a server, but a handheld personal digital assistant (PDA) or a cell phone might be a network client. In fact, because more consumer devices are becoming network-enabled, network security issues will continue to grow.

Although this model defines a basic network, the actual situation is frequently significantly more complicated.

- the simpler client device, employed for user-to-computer communication, is often a PC or workstation, so the client has considerable storage and processing capability.
- a network can be configured as just a single client connected to a single server. But more typically, many clients interact with many servers.
- the network's services are often provided by many computers. As a single user's communication travels back and forth from client to server, it may merely pass through some computers but pause at others for significant interactions.

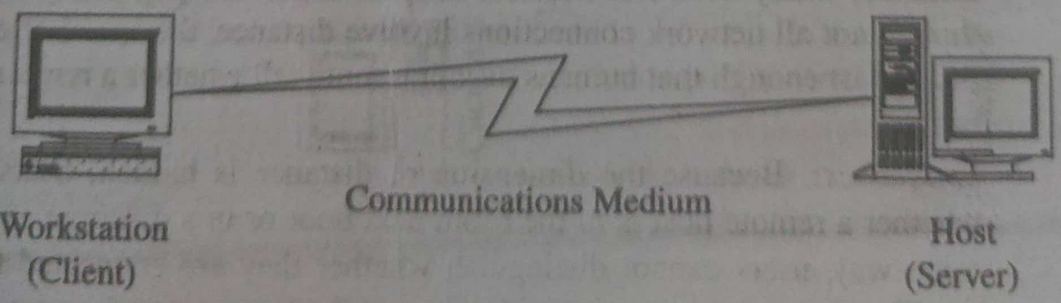

FIGURE 7-1 Simple View of Network.

- the end user is usually unaware of many of the communications and computations taking place in the network on the user's behalf.

Most real-world situations are more like Figure 7-2. In this second view, the user at one of the lettered client machines may send a message to System 3, unaware that communication is actually passing through the active Systems 1 and 2. In fact, the user may be unaware that System 3 sometimes passes work to System 4.

A single computing system in a network is often called a **node**, and its processor (computer) is called a **host**. A connection between two hosts is known as a **link**. Network computing consists of users, communications media, visible hosts, and systems not generally visible to end users. In Figure 7-2, Systems 1 through 4 are nodes. In our figure the users are at the lettered client machines, perhaps interacting with Server F.

Users communicate with networked systems by interacting directly with terminals, workstations, and computers. A **workstation** is an end-user computing device, usually designed for a single user at a time. Workstations often have powerful processors and good-sized memory and storage so that they can do sophisticated data manipulation (such as converting coded data to a graphical format and displaying the picture). A system is a collection of processors, perhaps including a mixture of workstations and independent processors, typically with more processing power and more storage capacity than a workstation.

Environment of Use

The biggest difference between a network and a stand-alone device is the environment in which each operates. Although some networks are located in protected spaces (for example, a local area network in a single laboratory or office), at least some portion of most networks is exposed, often to total strangers. The relatively simple network in Figure 7-2 is a good example. Systems 2, 3, and 4 are remote from System 1, and they may be under different ownership or control.

Networks can be described by several typical characteristics:

- *anonymity*. You may have seen the cartoon image that shows a dog typing at a workstation, and saying to another dog, "On the Internet, nobody knows you're a dog." A network removes most of the clues, such as appearance, voice, context, by which we recognize acquaintances.
- *automation*. In some networks, one or both endpoints, as well as all intermediate points, involved in a given communication may be machines with only minimal human supervision.
- *distance*. Many networks connect endpoints that are physically far apart. Although not all network connections involve distance, the speed of communication is fast enough that humans usually cannot tell whether a remote site is near or far.
- *opaqueness*. Because the dimension of distance is hidden, users cannot tell whether a remote host is in the room next door or in a different country. In the same way, users cannot distinguish whether they are connected to a node in an office, school, home, or warehouse, or whether the node's computing system is large or small, modest or powerful. In fact, users cannot tell if the cu

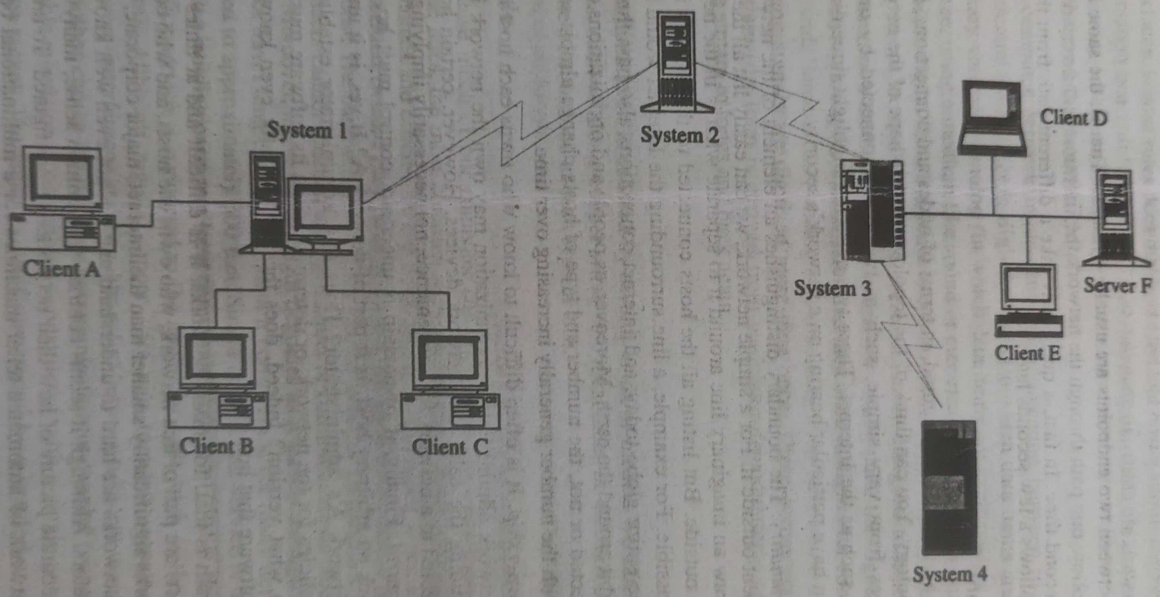

FIGURE 7-2 More Complex but More Typical View of Networks.

communication involves the same host with which they communicated the last time.

- *routing diversity*. To maintain or improve reliability and performance, routings between two endpoints are usually dynamic. That is, the same interaction may follow one path through the network the first time and a very different path the second time. In fact, a query may take a different path from the response that follows a few seconds later.

Shape and Size

The way a network is configured, in terms of nodes and connections, is called the network **topology**. You can think of the topology as the shape of the network. The topology ranges from very simple, such as two hosts connected by one path, to very complex, such as the Internet. These two extremes highlight three dimensions of networks that have particular bearing on a network's security.

- *boundary*. The boundary distinguishes an element of the network from an element outside it. For a simple network, we can easily list all the components and draw an imaginary line around it to separate what is in the network from what is outside. But listing all the hosts connected to the Internet is practically impossible. For example, a line surrounding the Internet would have to surround the entire globe today, and Internet connections also pass through satellites in orbit around the earth. Moreover, as people and organizations choose to be connected or not, the number and type of hosts change almost second by second, with the number generally increasing over time.
- *ownership*. It is often difficult to know who owns each host in a network. The network administrator's organization may own the network infrastructure, including the cable and network devices. However, certain hosts may be connected to a network for convenience, not necessarily implying ownership.
- *control*. Finally, if ownership is uncertain, control must be, too. To see how, pick an arbitrary host. Is it part of network A? If yes, is it under the control of network A's administrator? Does that administrator establish access control policies for the network, or determine when its software must be upgraded and to what version? Indeed, does the administrator even know what version of software that host runs?

The truth is that, for many networks, it is difficult and at times impossible to tell which hosts are part of that network, who owns the hosts, and who controls them. Even for networks significantly smaller than the Internet, major corporate, university, or government networks are hard to understand and are not even well known by their system administrators. Although it seems contrary to common sense, many corporations today have no accurate picture of how their networks are configured. To understand why, consider a network of automated teller machines for a multinational bank. The bank may have agreements with other banks to enable customers to withdraw money anywhere in the world. The multinational bank may understand its own bank's network, but it may have no conception of how the connecting banks' networks are configured; no "big picture" shows how the combined networks look or operate. Similarly, a given host may be

part of more than one network. In such a situation, suppose a host has two network interfaces. Whose rules does that host (and that host's administrator) have to follow?

Depicting, configuring, and administering networks are not easy tasks.

Mode of Communication

A computer network implements communication between two endpoints. Data are communicated either in **digital** format (in which data items are expressed as discrete binary values) or **analog** (in which data items are expressed as points in a continuous range, using a medium like sound or electrical voltage). Computers typically store and process digital data, but some telephone and similar cable communications are in analog form (because telephones were originally designed to transmit voice). When the transmission medium expects to transfer analog data, the digital signals must be converted to analog for transmission and then back to digital for computation at the receiving end. Some mostly analog networks may even have some digital segments, so the analog signals are digitized more than once. These conversions are performed by a **modem** (the term is derived from *mo*dulator-*demo*dulator), which converts a digital data stream to tones and back again.

Media

Communication is enabled by several kinds of media. We can choose among several types, such as along copper wires or optical fiber or through the air, as with cellular phones. Let us look at each type in turn.

Cable

Because much of our computer communication has historically been done over telephone lines, the most common network communication medium today is **wire**. Inside our homes and offices, we use a pair of insulated copper wires, called a **twisted pair** or **unshielded twisted pair (UTP)**. Copper has good transmission properties at a relatively low cost. The bandwidth of UTP is limited to under 10 megabits per second (Mbps),[1] so engineers cannot transmit a large number of communications simultaneously on a single line. Moreover, the signal strength degrades as it travels through the copper wire, and it cannot travel long distances without a boost. Thus, for many networks, line lengths are limited to approximately 300 feet. Single twisted pair service is most often used locally, within a building or up to a local communications drop (that is, the point where the home or office service is connected to the larger network, such as the commercial telephone system). Although regular copper wire can transmit signals, the twisting reduces crossover (interference and signal transfer) between adjacent wires.

Another choice for network communication is **coaxial (coax) cable**, the kind used for cable television. Coax cable is constructed with a single wire surrounded by an insulation jacket. The jacket is itself surrounded by a braided or spiral-wound wire. The inner wire carries the signal, and the outer braid acts as a ground. The most widely

[1] The figures in this section were accurate when they were written, but technology is constantly changing. However, as speeds or capacities change, the basic ranking of two technologies tends to remain the same.

used computer communication coax cable is **Ethernet**, carrying up to 100 Mbps over distances of up to 1500 feet.

Coax cable also suffers from degradation of signal quality over distance. **Repeaters** (for digital signals) or **amplifiers** (for analog signals) can be spaced periodically along the cable to pick up the signal, amplify it, remove spurious signals called "noise," and retransmit it.

Optical Fiber

A newer form of cable is made of very thin strands of glass. Instead of carrying electrical energy, these fibers carry pulses of light. The bandwidth of optical fiber is up to 1000 Mbps, and the signal degrades less over fiber than over wire or coax; the fiber is good for a run of approximately 2.5 miles. Optical fiber involves less interference, less crossover between adjacent media, lower cost, and less weight than copper. Thus, optical fiber is generally a much better transmission medium than copper. Consequently, as copper ages, it is being replaced by optical fiber in most communication systems. In particular, most long distance communication lines are now fiber.

Wireless

Radio signals can also carry communications. Similar to pagers, wireless microphones, garage door openers, and portable telephones, wireless radio can be used in networks, following a protocol developed for short-range telecommunications, designated the 802.11 family of standards. The wireless medium is used for short distances; it is especially useful for networks in which the nodes are physically close together, such as in an office building or at home. Many 802.11 devices are becoming available for home and office wireless networks.

Microwave

Microwave is a form of radio transmission especially well suited for outdoor communication. Microwave has a channel capacity similar to coax cable; that is, it carries similar amounts of data. Its principal advantage is that the signal is strong from point of transmission to point of receipt. Therefore, microwave signals do not need to be regenerated with repeaters, as do signals on cable.

However, a microwave signal travels in a straight line, presenting a problem because the earth curves. Microwave signals travel by line of sight: The transmitter and receiver must be in a straight line with one another, with no intervening obstacles, such as mountains. As shown in Figure 7-3, a straight microwave signal transmitted between towers of reasonable height can travel a distance of only about 30 miles because of the earth's curvature. Thus, microwave signals are "bounced" from receiver to receiver, spaced less than 30 miles apart, to cover a longer distance.

Infrared

Infrared communication carries signals for short distances (up to 9 miles) and also requires a clear line of sight. Because it does not require cabling, it is convenient for portable objects, such as laptop computers and connections to peripherals. An infrared

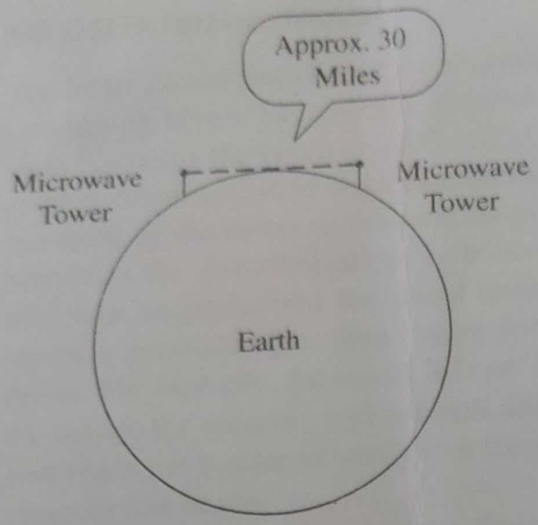

FIGURE 7-3 Microwave Transmission.

signal is difficult to intercept because it is a point-to-point signal. However, it is subject to "in the middle" attacks in which the interceptor functions like a repeater, receiving the signal, extracting any desired data, and retransmitting to the original destination the original signal or a modified version. Because of line-of-sight requirements and limited distance, infrared is typically used in a protected space, such as an office, in which in-the-middle attacks would be difficult to conceal.

Satellite

Many communications, such as international telephone calls, must travel around the earth. In the early days of telephone technology, telephone companies ran huge cables along the ocean's bottom, enabling calls to travel from one continent to another. Today, we have other alternatives. The communication companies place satellites in orbits that are synchronized with the rotation of the earth (called **geosynchronous orbits**), so the satellite appears to hover in a fixed position 22,300 miles above the earth. Although the satellite can be expensive to launch, once in space it is essentially maintenance free. Furthermore, the quality of a satellite communication link is often better than an earth-bound wire cable.

Satellites act as naïve transponders: Whatever they receive they broadcast out again. Thus, satellites are really sophisticated receivers, in that their sole function is to receive and repeat signals. From the user's point of view, the signal essentially "bounces" off the satellite and back to earth. For example, a signal from North America travels 22,300 miles into the sky and the same distance back to a point in Europe. The process of bouncing a signal off a satellite is shown in Figure 7-4.

We can project a signal to a satellite with reasonable accuracy, but the satellite is not expected to have the same level of accuracy when it sends the signal back to earth. To reduce complexity and eliminate beam focusing, satellites typically spread their transmissions over a very wide area. A rather narrow angle of dispersion from the satellite's transmitter produces a fairly broad pattern (called the **footprint**) on the surface of the earth because of the 22,300-mile distance from the satellite to earth. Thus, a typical

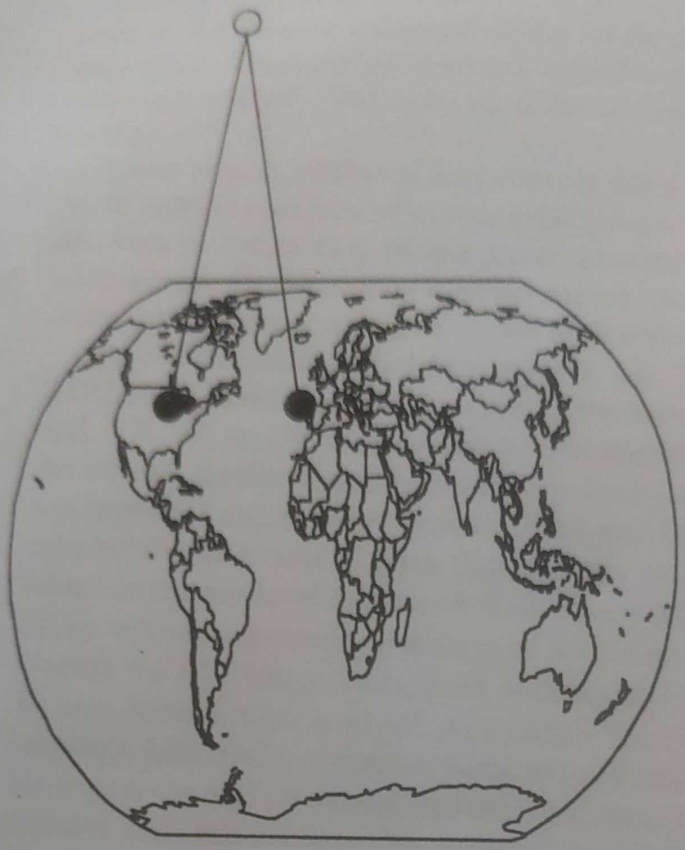

FIGURE 7-4 Satellite Communication.

satellite transmission can be received over a path several hundred miles wide; some cover the width of the entire continental United States in a single transmission. For some applications, such as satellite television, a broad footprint is desirable. But for secure communications, the smaller the footprint, the less the risk of interception.

Protocols

When we use a network, the communication media are usually transparent to us. That is, most of us do not know whether our communication is carried over copper wire, optical fiber, satellite, microwave, or some combination. In fact, the communication medium may change from one transmission to the next. This ambiguity is actually a positive feature of a network: its *independence*. That is, the communication is separated from the actual medium of communication. Independence is possible because we have defined **protocols** that allow a user to view the network at a high, abstract level of communication (viewing it in terms of user and data); the details of *how* the communication is accomplished are hidden within software and hardware at both ends. The software and hardware enable us to implement a network according to a **protocol stack**, a layered architecture for communications. Each layer in the stack is much like a language for communicating information relevant at that layer.

Two popular protocol stacks are used frequently for implementing networks: the Open Systems Interconnection (OSI) and the Transmission Control Protocol and Internet Protocol (TCP/IP) architecture. We examine each one in turn.

ISO OSI Reference Model

The International Standards Organization (ISO) Open Systems Interconnection model consists of layers by which a network communication occurs. The OSI reference model contains the seven layers listed in Table 7-1.

How communication works across the different layers is depicted in Figure 7-5. We can think of the layers as creating an assembly line, in which each layer adds its own service to the communication. In concert, the layers represent the different activities that must be performed for actual transmission of a message. Separately, each layer serves a purpose; equivalent layers perform similar functions for the sender and receiver. For example, the sender's layer four affixes a header to a message, designating the sender, the receiver, and relevant sequence information. On the receiving end, layer four reads the header to verify that the message is for the intended recipient, and then removes this header.

Each layer passes data in three directions: *above* with a layer communicating more abstractly, *parallel* or *across* to the same layer in another host, and *below* with a layer handling less abstract (that is, more fundamental) data items. The communications above and below are actual interactions, while the parallel one is a virtual communication path. Parallel layers are called "**peers**."

Let us look at a simple example of protocol transmission. Suppose that, to send e-mail to a friend, you run an application such as Eudora, Outlook, or Unix mail. You type a message, using the application's editor, and the application formats the message into two parts: a header that shows to whom the message is intended (as well as other things, such as sender and time sent), and a body that contains the text of your message. The application reformats your message into a standard format so that even if you and your friend use different mail applications, you can still exchange e-mail. This transformation is shown in Figure 7-6.

However, the message is not transmitted exactly as you typed it, as raw text. Raw text is a very inefficient coding, because an alphabet uses relatively few of the 255 possible

TABLE 7-1 OSI Protocol Layer Levels.

Layer	Name	Activity
7	Application	User-level data
6	Presentation	Standardized data appearance, blocking, text compression
5	Session	Sessions or logical connections between parts of an application; message sequencing, recovery
4	Transport	Flow control, end-to-end error detection and correction, priority service
3	Network	Routing, message blocking into uniformly sized packets
2	Data Link	Reliable data delivery over physical medium; transmission error recovery, separating packets into uniformly sized frames
1	Physical	Actual communication across physical medium; individual bit transmission

Starting at the local area network, each node has a unique address, defined in hardware on the network connector device (such as a network interface card) or its software driver. A network administrator may choose network addresses to be easy to work with, such as 1001, 1002, 1003 for nodes on one LAN, and 2001, 2002, and so forth on another.

A host on a TCP/IP wide area network has a 32-bit address,[2] called an **IP address**. An IP address is expressed as four 8-bit groups in decimal notation, separated by periods, such as 100.24.48.6. People prefer speaking in words or pseudowords, so network addresses are also known by **domain names**, such as ATT.COM or CAM.AC.UK. Addressing tables convert domain names to IP addresses.

A domain name is parsed from right to left. The rightmost portion, such as .COM, .EDU, .NET, .ORG, or .GOV, or one of the two-letter country specific codes, such as .UK, .FR, .JP, or .DE, is called a **top-level domain**. A small set of organizations called the Internet Registrars controls these top-level domains; the registrars also control the registration of second-level domains, such as ATT in ATT.COM. Essentially, the registrars publish addresses of hosts that maintain tables of the second-level domains contained in the top-level domain. A host connected to the Internet queries one of these tables to find the numeric IP address of ATT in the .COM domain. AT&T, the company owning the ATT Internet site, must maintain its own host to resolve addresses within its own domain, such as MAIL.ATT.COM. You may find that the first time you try to resolve a fully qualified domain name to its IP address, your system performs a lookup starting at the top; for subsequent attempts, your system maintains a cache of domain name records that lets it resolve addresses locally. Finally, a domain name is translated into a 32-bit, four-octet address, and that address is included in the IP packets destined for that address. (We return to name resolution later in this chapter because it can be used in network attacks.)

Routing Concepts

A host needs to know how to direct a packet from its own IP address. Each host knows to what other hosts it is directly connected, and hosts communicate their connections to their neighbors. For the example network of Figure 7-2, System 1 would inform System 2 that it was one hop away from Clients A, B, and C. In turn, System 2 would inform its other neighbor, System 3, that it (System 2) was two hops away from Clients A, B, and C. From System 3, System 2 would learn that System 3 was one hop away from Clients D and E, Server F, and System 4, which System 2 would then pass to System 1 as being a distance of two hops. The routing protocols are actually more complex than this description, but the concepts are the same; hosts advertise to their neighbors to describe to which hosts (addresses) they can route traffic and at what cost (number of hops). Each host routes traffic to a neighbor that offers a path at the cheapest cost.

[2]The world's networks are running out of unique addresses. This 32-bit standard address is being increased to 128 bits in a scheme called IPv6. But because 32-bit addresses will remain for some time, we focus on the older version.

Types of Networks

A network is a collection of communicating hosts. But to understand the network and how it works, we have several key questions to ask, such as How many hosts? Communicating by what means? To answer these questions, we are helped by an understanding of several types of subclasses of networks, since they commonly combine into larger networks. The subclasses are general notions, not definitive distinctions. But since the terms are commonly used, we present several common network subclasses that have significant security properties.

Local Area Networks

As the name implies, a **local area network** (or **LAN**) covers a small distance, typically within a single building. Usually a LAN connects several small computers, such as personal computers, as well as printers and perhaps some dedicated file storage devices. Figure 7-10 shows the arrangement of a typical LAN. The primary advantage of a LAN is the opportunity for its users to share data and programs and to share access to devices such as printers.

Most LANs have the following characteristics.

- *small*. Typically, fewer than 100 users share a single LAN, within a distance less than 3 kilometers, or 2 miles. More commonly, a LAN is much smaller, stretching less than 1 kilometer inside a single building.
- *locally controlled*. The equipment is owned and managed by a single organization. The users all are affiliated with a single organization, such as a company, a department, a workgroup, or a physical proximity.
- *physically protected*. The LAN is on the premises of a company or other organization, so malicious outsiders usually cannot readily get to the LAN equipment.
- *limited scope*. Many LANs support a single group, department, floor, activity, or other geographical or administrative unit. As a result, each has a narrowly scoped set of functions it performs.

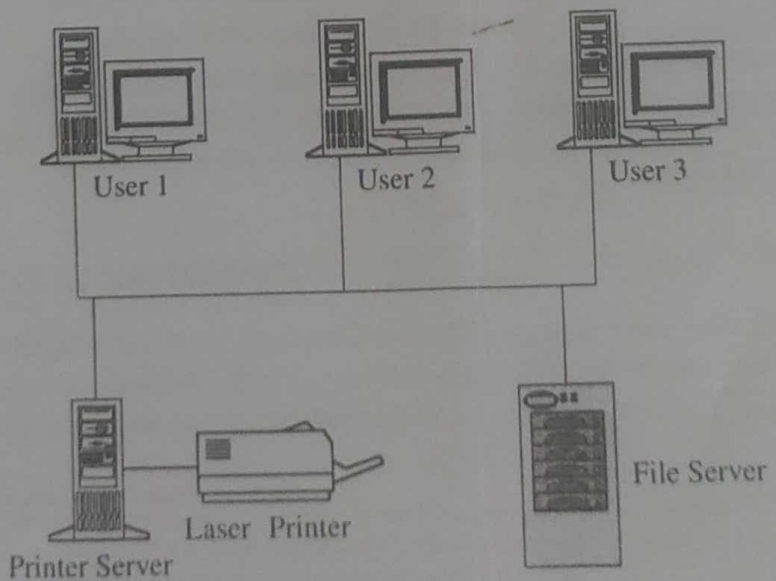

FIGURE 7-10 Typical LAN.

Wide Area Networks

A **wide area network**, or **WAN**, differs from a local area network in terms of both size or distance (as its name implies, it covers a wider geographic area than does a LAN) and control or ownership (it is more likely *not* to be owned or controlled by a single body). Still, there tends to be some unifying principle to a WAN. The hosts on a WAN may all belong to a company with many offices, perhaps even in different cities or countries, or they may be a cluster of independent organizations within a few miles of each other, who share the cost of networking hardware. These examples also show how WANs themselves differ. Some are under close control and maintain a high degree of logical and physical isolation (typically, these are WANs controlled by one organization), while others are only marriages of convenience. Typical characteristics of WANs are these.

- *single control*. Typically, a single organization is responsible for and controls a wide area network. Even if a network is shared by several unrelated subscribers, one organization usually determines who may join the network.
- *covers a significant distance*. A WAN generally serves a distance greater than a LAN can cover, typically from a few miles to the entire globe.
- *physically exposed* (often, but not always). Most wide area networks use publicly available communications media, which are relatively exposed. However, the fact that many subscribers share those media helps protect the privacy of any one subscriber.

Other network types include campus area networks (CANs) and metropolitan area networks (MANs). A CAN is usually under the control of a single organization, such as a university or company, and covers the adjacent buildings of one site of that organization. A MAN often covers a city, with the communication offering of one provider in that area. CANs, MANs, and WANs cover a wide range of possibilities; they loosely characterize everything between LANs and Internets, the two extremes of the networking spectrum.

Internetworks (Internets)

Networks of networks, or internetwork networks, are sometimes called **internets**. An internet is a connection of two or more separate networks, in that they are separately managed and controlled. The most significant internetwork is known as the **Internet**, because it connects so many of the other public networks.

The Internet is, in fact, a federation of networks, loosely controlled by the Internet Society (ISOC) and the Internet Corporation for Assigned Names and Numbers (ICANN). These organizations enforce certain minimal rules of fair play to ensure that all users are treated equitably, and they support standard protocols so that users can communicate. These are the characteristics of the Internet.

- *federation*. Almost no general statements can be made about Internet users or even network service providers. Some may access the network through businesses or government organizations whose memberships are very restrictive, while others may obtain access simply by paying a small monthly fee.
- *enormous*. No one really knows how large the Internet is. Our knowledge is incomplete in part because new hosts are added daily, in part because one Internet

Sidebar 7-1 Traffic at a Typical Web Site

Many sites record network traffic data; some publicize the data and many more use the data internally to monitor performance, manage resources, or demonstrate usage. For example, the site worldpress.org provides readers with an international view of important news: what stories are reported and how stories are covered throughout the world. During summer 2006, they averaged monthly 2 million visits by 800,000 visitors with over 3 million pages viewed.

The median length of a visit was approximately two minutes, which would correspond to a user who read several headlines and perhaps one news story. Over 60 percent of visits came from the United States, with approximately 20 percent from other countries and 20 percent of undeterminable origin. Full statistics are at http://www.worldpress.org/traffic.cfm.

But these statistics count all traffic, not just the security-relevant activity. The security company ISS (Internet Security Systems) tracks the status of actual Internet security risk. Its four-point scale goes from 1 (normal risk from random malicious attacks experienced by all site administrators) to 4 (actual or potential catastrophic security event requiring immediate defense). During a period from April to June 2002, ISS reported 56 days at level 1, 22 at level 2, and 7 at level 3 [ISS02].

access point can support hundreds or thousands of machines connected through that single access point, and in part because nobody has laid the basis for an accurate census. The Internet connects many thousands of networks. In 2006, according to isc.org, there were almost 400 million Internet hosts and well over 700 million users.[3] Based on past history, we can expect the size of the Internet to double each year. Sidebar 7-1 describes the large number of outside accesses just to one pubic news web site.

- *heterogeneous*. Probably at least one of every kind of commercially available hardware and software is connected to the Internet. Unix is popular as the operating system at the Internet connection point, although most other multiuser operating systems could support access.
- *physically and logically exposed*. Since there is no global access control, practically any attacker can access the Internet and, because of its complex connectivity, reach practically any resource on the net.

7.2 THREATS IN NETWORKS

Up to now, we have reviewed network concepts with very little discussion of their security implications. But our earlier discussion of threats and vulnerabilities, as well as outside articles and your own experiences, probably have you thinking about the many

[3]Counting the number of hosts or users is obviously difficult. But from a security perspective, even if these numbers are too high and if only a small percentage of hosts and users are malicious, the number of possible attacks is still large enough to be worth attention.

possible attacks against networks. This section describes some of the threats you have already hypothesized and perhaps presents you with some new ones. But the general thrust is the same: threats aimed to compromise confidentiality, integrity, or availability, applied against data, software, and hardware by nature, accidents, nonmalicious humans, and malicious attackers.

hat Makes a Network Vulnerable?

An isolated home user or a stand-alone office with a few employees is an unlikely target for many attacks. But add a network to the mix and the risk rises sharply. Consider how a network differs from a stand-alone environment:

- *anonymity*. An attacker can mount an attack from thousands of miles away and never come into direct contact with the system, its administrators, or users. The potential attacker is thus safe behind an electronic shield. The attack can be passed through many other hosts in an effort to disguise the attack's origin. And computer-to-computer authentication is not the same for computers as it is for humans; as illustrated by Sidebar 7-2, secure distributed authentication requires thought and attention to detail.
- *many points of attack—both targets and origins*. A simple computing system is a self-contained unit. Access controls on one machine preserve the confidentiality of data on that processor. However, when a file is stored in a network host remote from the user, the data or the file itself may pass through many hosts to get to the user. One host's administrator may enforce rigorous security policies, but that administrator has no control over other hosts in the network. Thus, the user must depend on the access control mechanisms in each of these systems. An attack can come from any host to any host, so that a large network offers many points of vulnerability.
- *sharing*. Because networks enable resource and workload sharing, more users have the potential to access networked systems than on single computers. Perhaps worse, access is afforded to *more systems*, so that access controls for single systems may be inadequate in networks.
- *complexity of system*. In Chapter 4 we saw that an operating system is a complicated piece of software. Reliable security is difficult, if not impossible, on a large operating system, especially one not designed specifically for security. A network combines two or more possibly dissimilar operating systems. Therefore, a network operating/control system is likely to be more complex than an operating system for a single computing system. Furthermore, the ordinary desktop computer today has greater computing power than did many office computers in the last two decades. The attacker can use this power to advantage by causing the victim's computer to perform part of the attack's computation. And because an average computer is so powerful, most users do not know what their computers are really doing at any moment: What processes are active in the background while you are playing *Invaders from Mars*? This complexity diminishes confidence in the network's security.

Sidebar 7-2 Distributed Authentication in Windows NT and 2000

Authentication must be handled carefully and correctly in a network because a network involves authentication not just of people but of processes, servers, and services only loosely associated with a person. And for a network, the authentication process and database are often distributed for performance and reliability. Consider Microsoft's authentication scheme for its Windows operating systems. In Windows NT 4.0, the authentication database is distributed among several domain controllers. Each domain controller is designated as a primary or backup controller. All changes to the authentication database must be made to the (single) primary domain controller; then the changes are replicated from the primary to the backup domain controllers.

In Windows 2000, the concept of primary and backup domain controllers has been abandoned. Instead, the network views the controllers as equal trees in a forest, in which any domain controller can update the authentication database. This scheme reflects Microsoft's notion that the system is "multimaster": Only one controller can be master at a given time, but any controller can be a master. Once changes are made to a master, they are automatically replicated to the remaining domain controllers in the forest.

This approach is more flexible and robust than the primary-secondary approach because it allows any controller to take charge—especially useful if one or more controllers have failed or are out of service for some reason. But the multimaster approach introduces a new problem. Because any domain controller can initiate changes to the authentication database, any hacker able to dominate a domain controller can alter the authentication database. And, what's worse, the changes are then replicated throughout the remaining forest. Theoretically, the hacker could access anything in the forest that relies on Windows 2000 for authentication.

When we think of attackers, we usually think of threats from outside the system. But in fact the multimaster approach can tempt people inside the system, too. A domain administrator in any domain in the forest can access domain controllers within that domain. Thanks to multimaster, the domain administrator can also modify the authentication database to access anything else in the forest.

For this reason, system administrators must consider how they define domains and their separation in a network. Otherwise, we can conjure up scary but possible scenarios. For instance, suppose one domain administrator is a bad apple. She works out a way to modify the authentication database to make herself an administrator for the entire forest. Then she can access any data in the forest, turn on services for some users, and turn off services for other users.

- *unknown perimeter.* A network's expandability also implies uncertainty about the network boundary. One host may be a node on two different networks, so resources on one network are accessible to the users of the other network as well. Although wide accessibility is an advantage, this unknown or uncontrolled group of possibly malicious users is a security disadvantage. A similar problem occurs when new hosts can be added to the network. Every network node must be able to react to the possible presence of new, untrustable hosts.

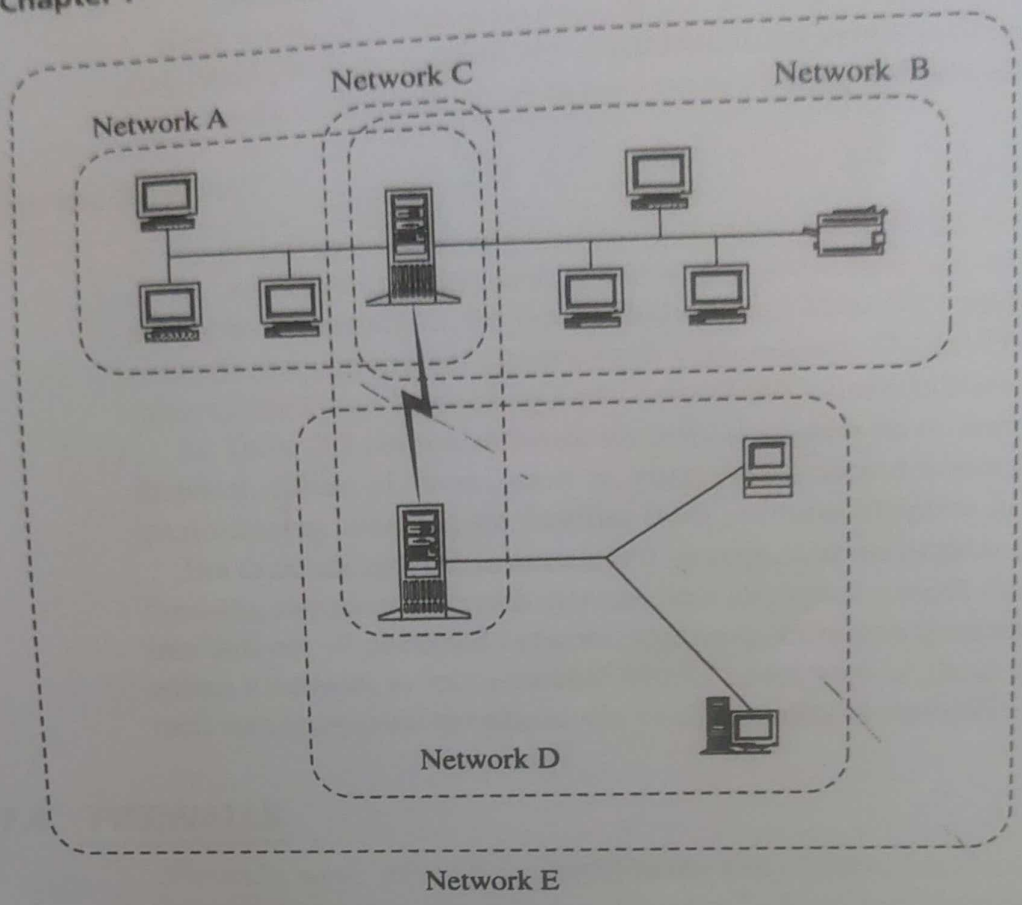

FIGURE 7-11 Unclear Network Boundaries.

Figure 7-11 points out the problems in defining the boundaries of a network. Notice, for example, that a user on a host in network D may be unaware of the potential connections from users of networks A and B. And the host in the middle of networks A and B in fact belongs to A, B, C, and E. If there are different security rules for these networks, to what rules is that host subject?

- *unknown path.* Figure 7-12 illustrates that there may be many paths from one host to another. Suppose that a user on host A1 wants to send a message to a user on host B3. That message might be routed through hosts C or D before arriving at host B3. Host C may provide acceptable security, but not D. Network users seldom have control over the routing of their messages.

Thus, a network differs significantly from a stand-alone, local environment. Network characteristics significantly increase the security risk.

Categories of Attack

Active v/s Passive Attack

Active attack attempts to alter system resources or affect the operation, that is, Attacks that violate the integrity of the system or message. An example in this category is an attack on the availability of a system or services, a so-called *denial-of-service* attack. Active attacks can affect the availability, integrity and authenticity of the system. Passive

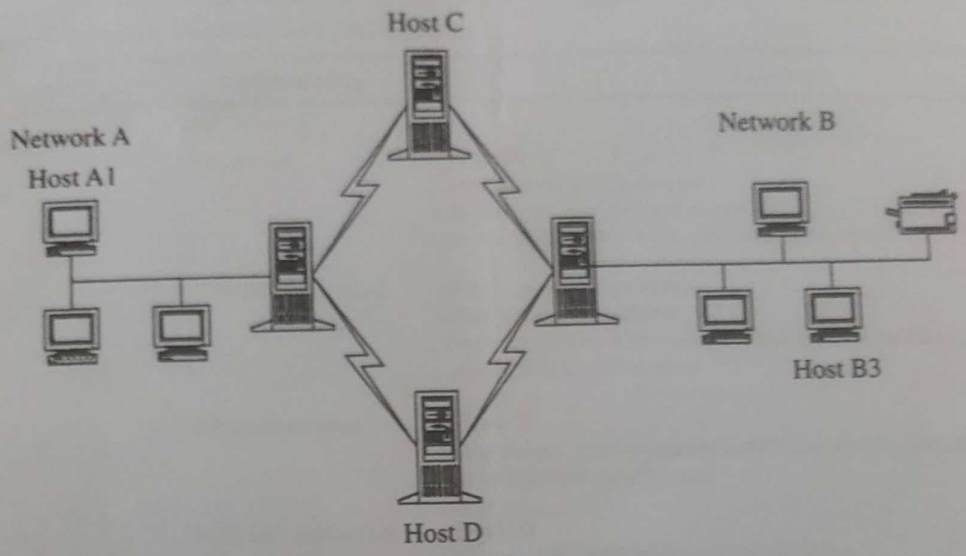

FIGURE 7-12 Uncertain Message Routing in a Network.

attacks are those that violate the confidentiality without affecting the state of the system. An example is the electronic eavesdropping on network transmissions to release message contents or to gather unprotected passwords. The key word here is 'confidentiality' and relates to preventing the disclosure of information to unauthorized persons.

Insider attack v/s Outsider attack

An '*insider attack*' is an attack initiated by an entity inside the security perimeter (within an organization), that is, an entity that is authorized to access system resources but uses them in a way not approved by the authority. An '*outside attack*' is initiated from outside the perimeter, by an unauthorized or illegitimate user of the system. For example International terrorists, hostile governments, organized criminals etc.

Who Attacks Networks?

Who are the attackers? We cannot list their names, just as we cannot know who are all the criminals in our city, country, or the world. Even if we knew who they were, we do not know if we could stop their behavior. (See Sidebar 7-3 for a first, tenuous link between psychological traits and hacking.) To have some idea of who the attackers might be, we return to concepts introduced in Chapter 1, where we described the three necessary components of an attack: method, opportunity, and motive.

In the next sections we explore method: tools and techniques the attackers use. Here we consider first the motives of attackers. Focusing on motive may give us some idea of who might attack a networked host or user. Four important motives are challenge or power, fame, money, and ideology.

Challenge

Why do people do dangerous or daunting things, like climb mountains or swim the English Channel or engage in extreme sports? Because of the challenge. The situation

Both forms of security testing can be of value to an organization. A black-box test may highlight how supposedly confidential information is leaked, while a white-box test is likely to dedicate much more time to probing for vulnerabilities.

Controls Review

At the end of our earlier discussion on threats in networks, we listed in Table 7-4 many of the vulnerabilities present in networks. Now that we have surveyed the controls available for networks, we repeat that table as Table 7-7, adding a column to show the controls that can protect against each vulnerability. (Note: This table is not exhaustive; other controls can be used against some of the vulnerabilities.)

As Table 7-7 shows, network security designers have many successful tools at their disposal. Some of these, such as encryption, access control and authentication, and programming controls, are familiar from previous chapters in this book.

But three are specific to networked settings, and we explore them now in greater depth: firewalls, intrusion detection systems, and encrypted e-mail. Firewalls control traffic flow into and out of protected network segments. Intrusion detection systems monitor traffic within a network to spot potential attacks under way or about to occur. And encrypted e-mail uses encryption to enhance the confidentiality or authenticity of e-mail messages.

7.4 FIREWALLS

Firewalls were officially invented in the early 1990s, but the concept really reflects the reference monitor (described in Chapter 5) from two decades earlier. The first reference to a firewall by that name may be [RAN92]; other early references to firewalls are the Trusted Information Systems firewall toolkit [RAN94] and the book by Cheswick and Bellovin [updated as CHE02].

What Is a Firewall?

A firewall is a device that filters all traffic between a protected or "inside" network and a less trustworthy or "outside" network. Usually a firewall runs on a dedicated device; because it is a single point through which traffic is channeled, performance is important, which means nonfirewall functions should not be done on the same machine. Because a firewall is executable code, an attacker could compromise that code and execute from the firewall's device. Thus, the fewer pieces of code on the device, the fewer tools the attacker would have by compromising the firewall. Firewall code usually runs on a proprietary or carefully minimized operating system.

The purpose of a firewall is to keep "bad" things outside a protected environment. To accomplish that, firewalls implement a security policy that is specifically designed to address what bad things might happen. For example, the policy might be to prevent any access from outside (while still allowing traffic to pass *from* the inside *to* the outside). Alternatively, the policy might permit accesses only from certain places, from certain users, or for certain activities. Part of the challenge of protecting a network with a firewall is determining which security policy meets the needs of the installation.

People in the firewall community (users, developers, and security experts) disagree about how a firewall should work. In particular, the community is divided about a firewall's default behavior. We can describe the two schools of thought as "that which is

TABLE 7-1 Network Vulnerabilities and Controls.

Target	Vulnerability	Control
Precursors to attack	• Port scan	• Firewall • Intrusion detection system • Running as few services as possible • Services that reply with only what is necessary
	• Social engineering	• Education, user awareness • Policies and procedures • Systems in which two people must agree to perform certain *security-critical functions*
	• Reconnaissance	• Firewall • "Hardened" (self-defensive) operating system and applications • Intrusion detection system
	• OS and application fingerprinting	• Firewall • "Hardened" (self-defensive) applications • Programs that reply with only what is necessary • Intrusion detection system
Authentication failures	• Impersonation	• Strong, one-time authentication
	• Guessing	• Strong, one-time authentication • Education, user awareness
	• Eavesdropping	• Strong, one-time authentication • Encrypted authentication channel
	• Spoofing	• Strong, one-time authentication
	• Session hijacking	• Strong, one-time authentication • Encrypted authentication channel • Virtual private network
	• Man-in-the-middle attack	• Strong, one-time authentication • Virtual private network • Protocol analysis
Programming flaws	• Buffer overflow	• Programming controls • Intrusion detection system • Controlled execution environment • Personal firewall
	• Addressing errors	• Programming controls • Intrusion detection system • Controlled execution environment • Personal firewall • Two-way authentication
	• Parameter modification, time-of-check to time-of-use errors	• Programming controls • Intrusion detection system • Controlled execution environment • Intrusion detection system • Personal firewall

TABLE 7-7 Network Vulnerabilities and Controls. (*Continued*)

Target	Vulnerability	Control
	• Server-side include	• Programming controls • Personal firewall • Controlled execution environment • Intrusion detection system
	• Cookie	• Firewall • Intrusion detection system • Controlled execution environment • Personal firewall
	• Malicious active code: Java, ActiveX	• Intrusion detection system • Programming controls • Signed code
	• Malicious code: virus, worm, Trojan horse	• Intrusion detection system • Signed code • Controlled execution environment • Intrusion detection system
	• Malicious typed code	• Signed code • Intrusion detection system • Controlled execution environment
Confidentiality		
	• Protocol flaw	• Programming controls • Controlled execution environment
	• Eavesdropping	• Encryption
	• Passive wiretap	• Encryption
	• Misdelivery	• Encryption
	• Exposure within the network	• End-to-end encryption
	• Traffic flow analysis	• Encryption • Traffic padding • Onion routing
	• Cookie	• Firewall • Intrusion detection system • Controlled execution environment
Integrity		
	• Protocol flaw	• Firewall • Controlled execution environment • Intrusion detection system • Protocol analysis • Audit
	• Active wiretap	• Encryption • Error detection code • Audit

TABLE 7-7 (*Continued*)

Target	Vulnerability	Control
	• Impersonation	• Firewall • Strong, one-time authentication • Encryption • Error detection code • Audit
	• Falsification of message	• Firewall • Encryption • Strong authentication • Error detection code • Audit
	• Noise	• Error detection code
	• Web site defacement	• Error detection code • Intrusion detection system • Controlled execution environment • Hardened host • Honeypot • Audit
	• DNS attack	• Firewall • Intrusion detection system • Strong authentication for DNS changes • Audit
Availability	• Protocol flaw	• Firewall • Redundant architecture
	• Transmission or component failure	• Architecture
	• Connection flooding, for example, echo-chargen, ping of death, smurf, syn flood	• Firewall • Intrusion detection system • ACL on border router • Honeypot
	• DNS attack	• Firewall • Intrusion detection system • ACL on border router • Honeypot
	• Traffic redirection	• Encryption • Audit
	• Distributed denial of service	• Firewall • Intrusion detection system • ACL on border router • Honeypot

not expressly forbidden is permitted" (default permit) and "that which is not expressly permitted is forbidden" (default deny). Users, always interested in new features, prefer the former. Security experts, relying on several decades of experience, strongly counsel the latter. An administrator implementing or configuring a firewall must choose one of the two approaches, although the administrator can often broaden the policy by setting the firewall's parameters.

Design of Firewalls

Remember from Chapter 5 that a reference monitor must be

- always invoked
- tamperproof
- small and simple enough for rigorous analysis

A firewall is a special form of reference monitor. By carefully positioning a firewall within a network, we can ensure that all network accesses that we want to control must pass through it. This restriction meets the "always invoked" condition. A firewall is typically well isolated, making it highly immune to modification. Usually a firewall is implemented on a separate computer, with direct connections only to the outside and inside networks. This isolation is expected to meet the "tamperproof" requirement. And firewall designers strongly recommend keeping the functionality of the firewall simple.

Types of Firewalls

Firewalls have a wide range of capabilities. Types of firewalls include

- packet filtering gateways or screening routers
- stateful inspection firewalls
- application proxies
- guards
- personal firewalls

Each type does different things; no one is necessarily "right" and the others "wrong." In this section, we examine each type to see what it is, how it works, and what its strengths and weaknesses are. In general, screening routers tend to implement rather simplistic security policies, whereas guards and proxy gateways have a richer set of choices for security policy. Simplicity in a security policy is not a bad thing; the important question to ask when choosing a type of firewall is what threats an installation needs to counter.

Because a firewall is a type of host, it often is as programmable as a good-quality workstation. While a screening router *can* be fairly primitive, the tendency is to host even routers on complete computers with operating systems because editors and other programming tools assist in configuring and maintaining the router. However, firewall developers are minimalists: They try to eliminate from the firewall all that is not strictly necessary for the firewall's functionality. There is a good reason for this minimal constraint: to give as little assistance as possible to a successful attacker. Thus, firewalls tend not to have user accounts so that, for example, they have no password file to conceal. Indeed, the most desirable firewall is one that runs contentedly in a back room; except for periodic scanning of its audit logs, there is seldom reason to touch it.

Packet Filtering Gateway

A packet filtering gateway or screening router is the simplest, and in some situations, the most effective type of firewall. A packet filtering gateway controls access to packets on the basis of packet address (source or destination) or specific transport protocol type (such as HTTP web traffic). As described earlier in this chapter, putting ACLs on routers may severely impede their performance. But a separate firewall behind (on the local side) of the router can screen traffic before it gets to the protected network. Figure 7-38 shows a packet filter that blocks access from (or to) addresses in one network; the filter allows HTTP traffic but blocks traffic using the Telnet protocol.

For example, suppose an international company has three LANs at three locations throughout the world, as shown in Figure 7-39. In this example, the router has two sides: inside and outside. We say that the local LAN is on the inside of the router, and the two connections to distant LANs through wide area networks are on the outside. The company might want communication *only* among the three LANs of the corporate network. It could use a screening router on the LAN at 100.24.4.0 to allow *in* only communications destined to the host at 100.24.4.0 and to allow *out* only communications addressed either to address 144.27.5.3 or 192.19.33.0.

Packet filters do not "see inside" a packet; they block or accept packets solely on the basis of the IP addresses and ports. Thus, any details in the packet's data field (for example, allowing certain Telnet commands while blocking other services) is beyond the capability of a packet filter.

Packet filters can perform the very important service of ensuring the validity of inside addresses. Inside hosts typically trust other inside hosts for all the reasons described as characteristics of LANs. But the only way an inside host can distinguish another inside host is by the address shown in the source field of a message. Source addresses in packets can be forged, so an inside application might think it was communicating with another host on the inside instead of an outside forger. A packet filter sits between the inside network and the outside net, so it can know if a packet from the outside is forging an inside

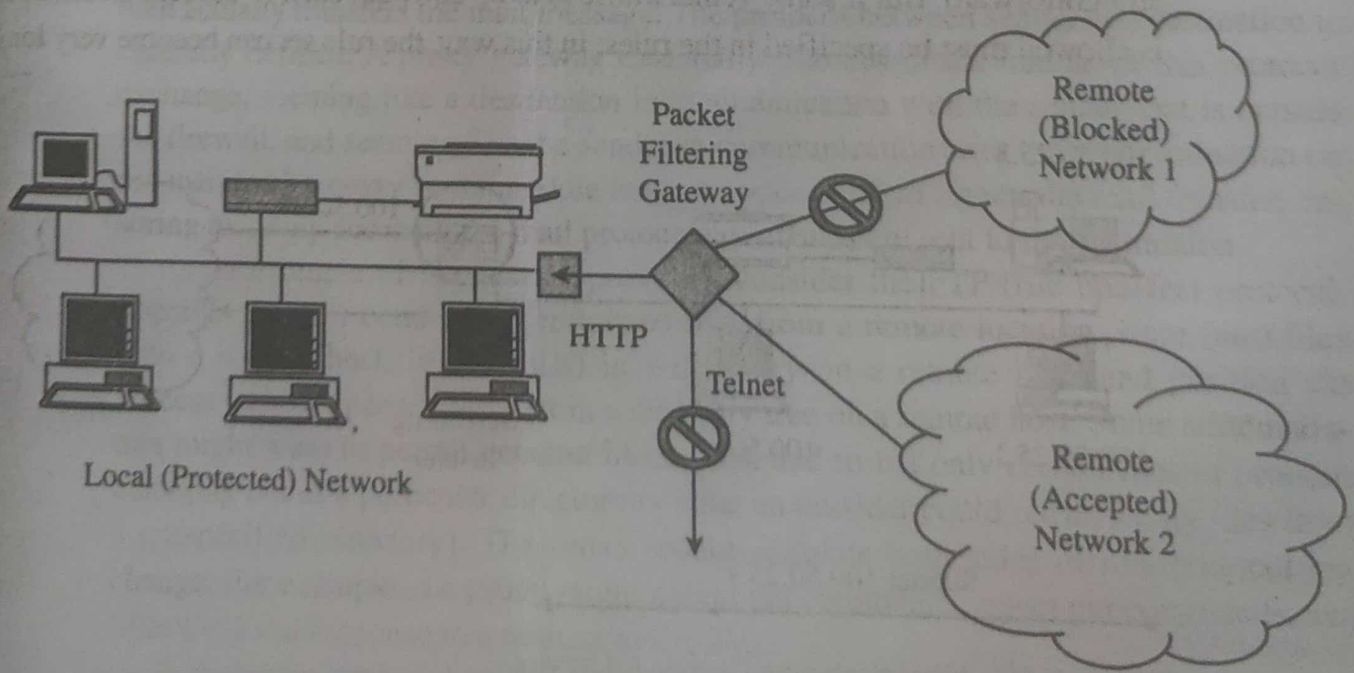

FIGURE 7-38 Packet Filter Blocking Addresses and Protocols.

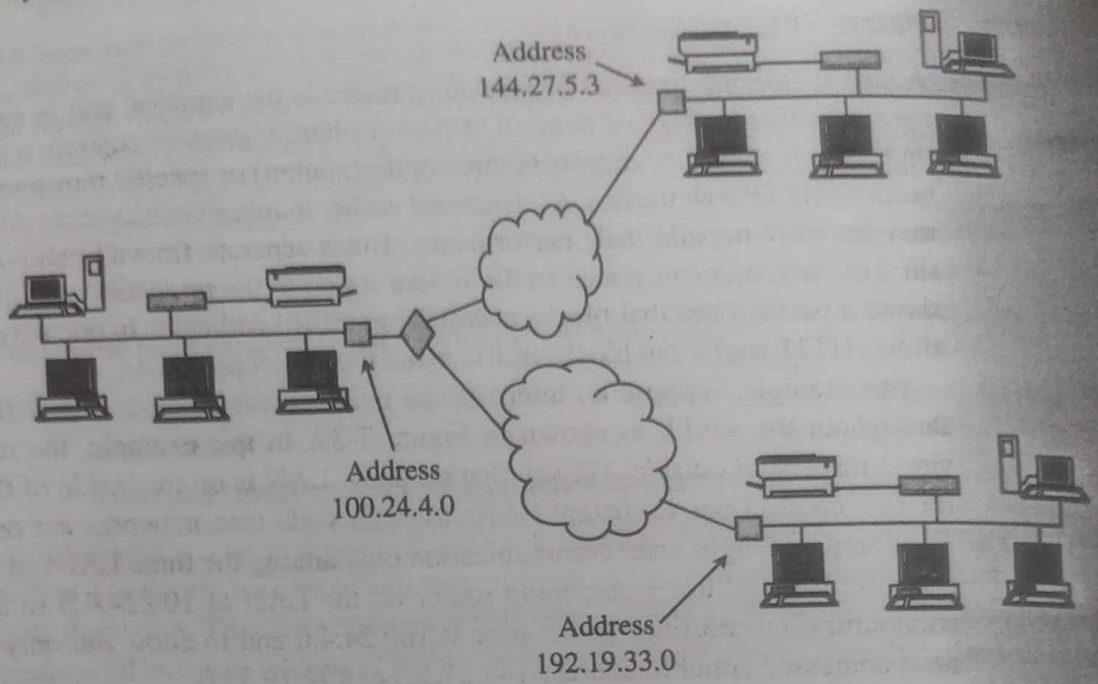

FIGURE 7-39 Three Connected LANs.

address, as shown in Figure 7-40. A screening packet filter might be configured to block all packets from the *outside* that claimed their source address was an *inside* address. In this example, the packet filter blocks all packets claiming to come from any address of the form 100.50.25.*x* (but, of course, it permits in any packets with *destination* 100.50.25.*x*).

The primary disadvantage of packet filtering routers is a combination of simplicity and complexity. The router's inspection is simplistic; to perform sophisticated filtering, the filtering rules set needs to be very detailed. A detailed rules set will be complex and therefore prone to error. For example, blocking all port 23 traffic (Telnet) is simple and straightforward. But if some Telnet traffic is to be allowed, each IP address from which it is allowed must be specified in the rules; in this way, the rule set can become very long.

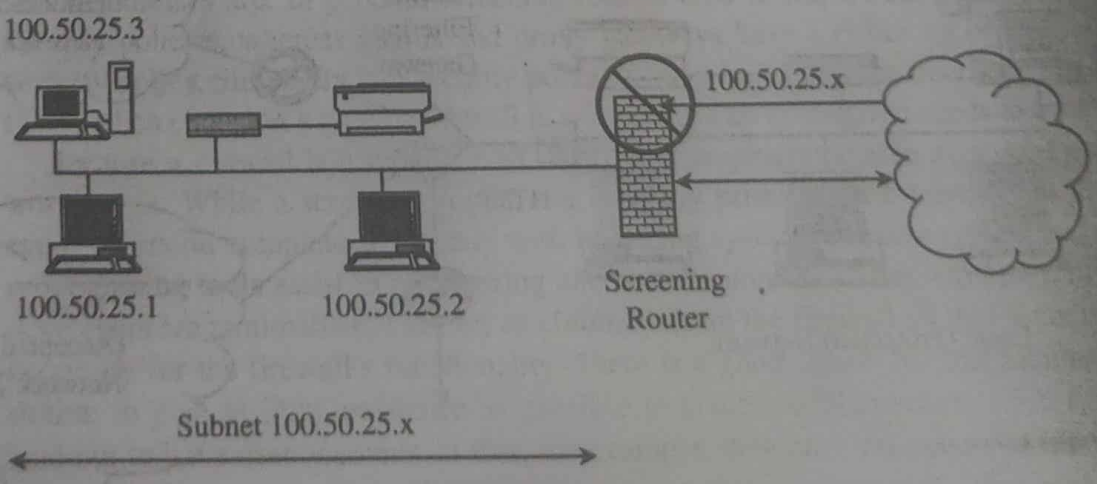

FIGURE 7-40 Filter Screening Outside Addresses.

Stateful Inspection Firewall

Filtering firewalls work on packets one at a time, accepting or rejecting each packet and moving on to the next. They have no concept of "state" or "context" from one packet to the next. A **stateful inspection firewall** maintains state information from one packet to another in the input stream.

One classic approach used by attackers is to break an attack into multiple packets by forcing some packets to have very short lengths so that a firewall cannot detect the signature of an attack split across two or more packets. (Remember that with the TCP protocols, packets can arrive in any order, and the protocol suite is responsible for reassembling the packet stream in proper order before passing it along to the application.) A stateful inspection firewall would track the sequence of packets and conditions from one packet to another to thwart such an attack.

Application Proxy

Packet filters look only at the headers of packets, not at the data *inside* the packets. Therefore, a packet filter would pass anything to port 25, assuming its screening rules allow inbound connections to that port. But applications are complex and sometimes contain errors. Worse, applications (such as the e-mail delivery agent) often act on behalf of all users, so they require privileges of all users (for example, to store incoming mail messages so that inside users can read them). A flawed application, running with all users' privileges, can cause much damage.

An **application proxy gateway,** also called a **bastion host,** is a firewall that simulates the (proper) effects of an application so that the application receives only requests to act properly. A proxy gateway is a two-headed device: It looks to the inside as if it is the outside (destination) connection, while to the outside it responds just as the insider would.

An application proxy runs pseudoapplications. For instance, when electronic mail is transferred to a location, a sending process at one site and a receiving process at the destination communicate by a protocol that establishes the legitimacy of a mail transfer and then actually transfers the mail message. The protocol between sender and destination is carefully defined. A proxy gateway essentially intrudes in the middle of this protocol exchange, seeming like a destination in communication with the sender that is outside the firewall, and seeming like the sender in communication with the real destination on the inside. The proxy in the middle has the opportunity to screen the mail transfer, ensuring that only acceptable e-mail protocol commands are sent to the destination.

As an example of application proxying, consider the FTP (file transfer) protocol. Specific protocol commands fetch (*get*) files from a remote location, store (*put*) files onto a remote host, list files (*ls*) in a directory on a remote host, and position the process (*cd*) at a particular point in a directory tree on a remote host. Some administrators might want to permit gets but block puts, and to list only certain files or prohibit changing out of a particular directory (so that an outsider could retrieve only files from a prespecified directory). The proxy would simulate both sides of this protocol exchange. For example, the proxy might accept get commands, reject put commands, and filter the local response to a request to list files.

To understand the real purpose of a proxy gateway, let us consider several examples.

- a company wants to set up an online price list so that outsiders can see the products and prices offered. It wants to be sure that (a) no outsider can change the prices or product list and (b) outsiders can access only the price list, not any of the more sensitive files stored inside.
- a school wants to allow its students to retrieve any information from World Wide Web resources on the Internet. To help provide efficient service, the school wants to know what sites have been visited and what files from those sites have been fetched; particularly popular files will be cached locally.
- a government agency wants to respond to queries through a database management system. However, because of inference attacks against databases, the agency wants to restrict queries that return the mean of a set of fewer than five values.
- a company with multiple offices wants to encrypt the data portion of all e-mail to addresses at its other offices. (A corresponding proxy at the remote end will remove the encryption.)
- a company wants to allow dial-in access by its employees, without exposing its company resources to login attacks from remote nonemployees.

Each of these requirements can be met with a proxy. In the first case, the proxy would monitor the file transfer protocol *data* to ensure that only the price list file was accessed, and that file could only be read, not modified. The school's requirement could be met by a logging procedure as part of the web browser. The agency's need could be satisfied by a special-purpose proxy that interacted with the database management system, performing queries but also obtaining the number of values from which the response was computed and adding a random minor error term to results from small sample sizes. The requirement

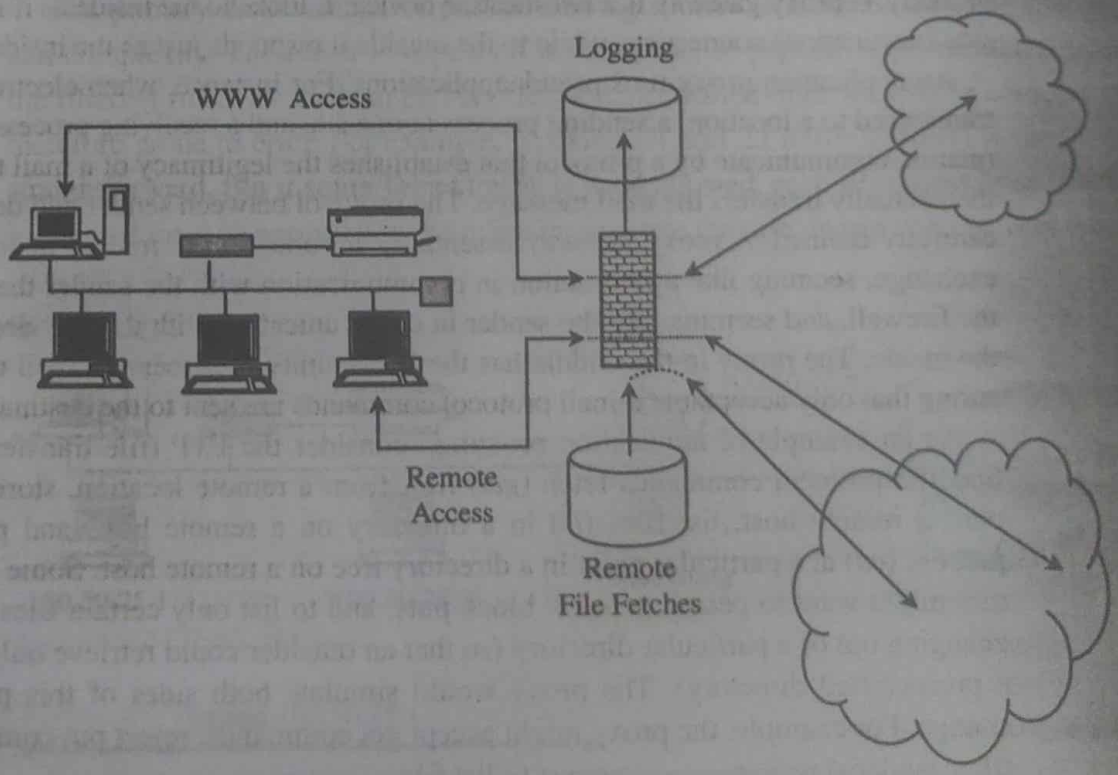

FIGURE 7-41 Actions of Firewall Proxies.

for limited login could be handled by a specially written proxy that required strong user authentication (such as a challenge–response system), which many operating systems do not require. These functions are shown in Figure 7-41.

The proxies on the firewall can be tailored to specific requirements, such as logging details about accesses. They can even present a common user interface to what may be dissimilar internal functions. Suppose the internal network has a mixture of operating system types, none of which support strong authentication through a challenge–response token. The proxy can demand strong authentication (name, password, and challenge–response), validate the challenge–response itself, and then pass on only simple name and password authentication details in the form required by a specific internal host's operating system.

The distinction between a proxy and a screening router is that the proxy interprets the protocol stream to an application, to control actions through the firewall on the basis of things visible *within* the protocol, not just on external header data.

Guard

A guard is a sophisticated firewall. Like a proxy firewall, it receives protocol data units, interprets them, and passes through the same or different protocol data units that achieve either the same result or a modified result. The guard decides what services to perform on the user's behalf in accordance with its available knowledge, such as whatever it can reliably know of the (outside) user's identity, previous interactions, and so forth. The degree of control a guard can provide is limited only by what is computable. But guards and proxy firewalls are similar enough that the distinction between them is sometimes fuzzy. That is, we can add functionality to a proxy firewall until it starts to look a lot like a guard.

Guard activities can be quite sophisticated, as illustrated in the following examples:

- a university wants to allow its students to use e-mail up to a limit of so many messages or so many characters of e-mail in the last so many days. Although this result could be achieved by modifying e-mail handlers, it is more easily done by monitoring the common point through which all e-mail flows, the mail transfer protocol.
- a school wants its students to be able to access the World Wide Web but, because of the slow speed of its connection to the web, it will allow only so many characters per downloaded image (that is, allowing text mode and simple graphics, but disallowing complex graphics, animation, music, or the like).
- a library wants to make available certain documents but, to support fair use of copyrighted matter, it will allow a user to retrieve only the first so many characters of a document. After that amount, the library will require the user to pay a fee that will be forwarded to the author.
- a company wants to allow its employees to fetch files via *ftp*. However, to prevent introduction of viruses, it will first pass all incoming files through a virus scanner. Even though many of these files will be nonexecutable text or graphics, the company administrator thinks that the expense of scanning them (which should pass) will be negligible.

Each of these scenarios can be implemented as a modified proxy. Because the proxy decision is based on some quality of the communication data, we call the proxy a guard. Since the security policy implemented by the guard is somewhat more complex

than the action of a proxy, the guard's code is also more complex and therefore more exposed to error. Simpler firewalls have fewer possible ways to fail or be subverted.

Personal Firewalls

Firewalls typically protect a (sub)network of multiple hosts. University students and employees in offices are behind a real firewall. Increasingly, home users, individual workers, and small businesses use cable modems or DSL connections with unlimited, always-on access. These people need a firewall, but a separate firewall computer to protect a single workstation can seem too complex and expensive. These people need a firewall's capabilities at a lower price.

A **personal firewall** is an application program that runs on a workstation to block unwanted traffic, usually from the network. A personal firewall can complement the work of a conventional firewall by screening the kind of data a single host will accept, or it can compensate for the lack of a regular firewall, as in a private DSL or cable modem connection.

Just as a network firewall screens incoming and outgoing traffic for that network, a personal firewall screens traffic on a single workstation. A workstation could be vulnerable to malicious code or malicious active agents (ActiveX controls or Java applets), leakage of personal data stored on the workstation, and vulnerability scans to identify potential weaknesses. Commercial implementations of personal firewalls include Norton Personal Firewall from Symantec, McAfee Personal Firewall, and Zone Alarm from Zone Labs (now owned by CheckPoint).

The personal firewall is configured to enforce some policy. For example, the user may decide that certain sites, such as computers on the company network, are highly trustworthy, but most other sites are not. The user defines a policy permitting download of code, unrestricted data sharing, and management access from the corporate segment, but not from other sites. Personal firewalls can also generate logs of accesses, which can be useful to examine in case something harmful does slip through the firewall.

Combining a virus scanner with a personal firewall is both effective and efficient. Typically, users forget to run virus scanners daily, but they do remember to run them occasionally, such as sometime during the week. However, leaving the virus scanner execution to the user's memory means that the scanner detects a problem only after the fact—such as when a virus has been downloaded in an e-mail attachment. With the combination of a virus scanner and a personal firewall, the firewall directs all incoming e-mail to the virus scanner, which examines every attachment the moment it reaches the target host and before it is opened.

A personal firewall runs on the very computer it is trying to protect. Thus, a clever attacker is likely to attempt an undetected attack that would disable or reconfigure the firewall for the future. Still, especially for cable modem, DSL, and other "always on" connections, the static workstation is a visible and vulnerable target for an ever-present attack community. A personal firewall can provide reasonable protection to clients that are not behind a network firewall.

Comparison of Firewall Types

We can summarize the differences among the several types of firewalls we have studied in depth. The comparisons are shown in Table 7-8.

TABLE 7-8 Comparison of Firewall Types.

Packet Filtering	Stateful Inspection	Application Proxy	Guard	Personal Firewall
Simplest	More complex	Even more complex	Most complex	Similar to packet filtering firewall
Sees only addresses and service protocol type	Can see either addresses or data	Sees full data portion of packet	Sees full text of communication	Can see full data portion of packet
Auditing difficult	Auditing possible	Can audit activity	Can audit activity	Can—and usually does—audit activity
Screens based on connection rules	Screens based on information across packets—in either header or data field	Screens based on behavior of proxies	Screens based on interpretation of message content	Typically, screens based on information in a single packet, using header or data
Complex addressing rules can make configuration tricky	Usually preconfigured to detect certain attack signatures	Simple proxies can substitute for complex addressing rules	Complex guard functionality can limit assurance	Usually starts in "deny all inbound" mode, to which user adds trusted addresses as they appear

Example Firewall Configurations

Let us look at several examples to understand how to use firewalls. We present situations designed to show how a firewall complements a sensible security policy and architecture.

The simplest use of a firewall is shown in Figure 7-42. This environment has a screening router positioned between the internal LAN and the outside network connection. In many cases, this installation is adequate when we need only screen the address of a router.

However, to use a proxy machine, this organization is not ideal. Similarly, configuring a router for a complex set of approved or rejected addresses is difficult. If the firewall router is successfully attacked, then all traffic on the LAN to which the firewall is connected is visible. To reduce this exposure, a proxy firewall is often installed on its own LAN, as shown in Figure 7-43. In this way the only traffic visible on that LAN is the traffic going into and out of the firewall.

For even more protection, we can add a screening router to this configuration, as shown in Figure 7-44. Here, the screening router ensures address correctness to the

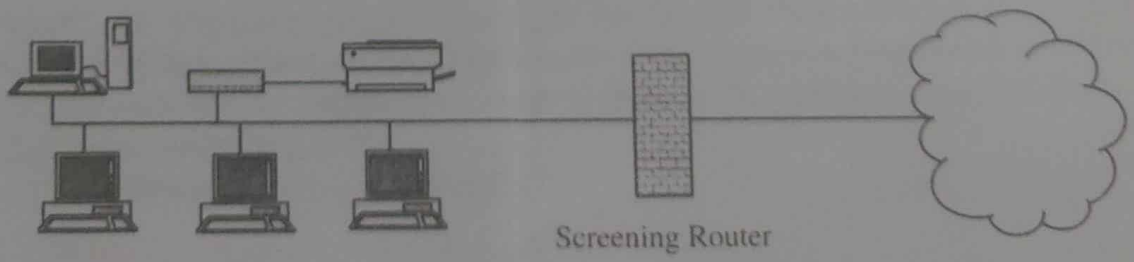

FIGURE 7-42 Firewall with Screening Router.

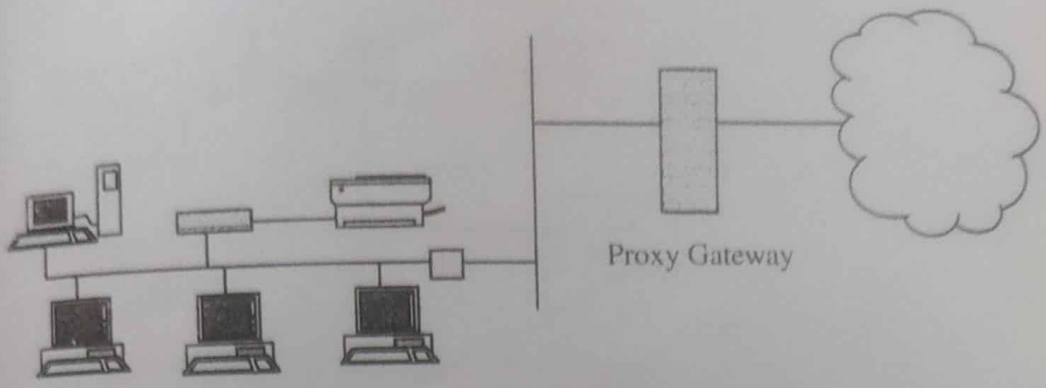

FIGURE 7-43 Firewall on Separate LAN.

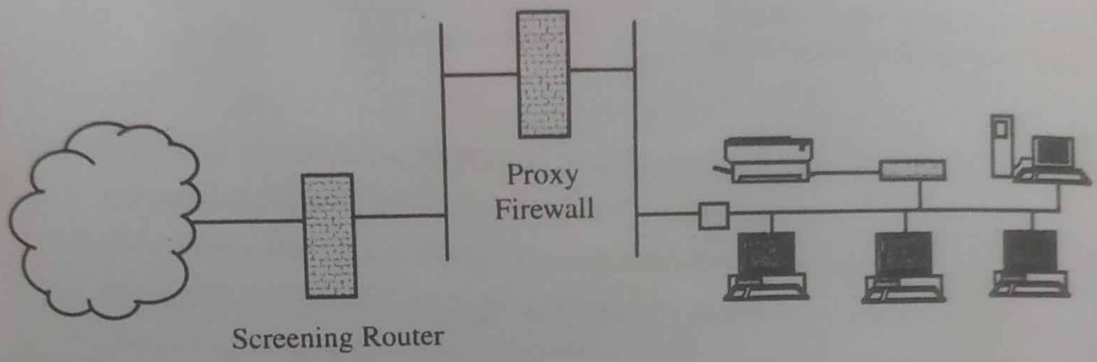

FIGURE 7-44 Firewall with Proxy and Screening Router.

proxy firewall (so that the proxy firewall cannot be fooled by an outside attacker forging an address from an inside host); the proxy firewall filters traffic according to its proxy rules. Also, if the screening router is subverted, only the traffic to the proxy firewall is visible—not any of the sensitive information on the internal protected LAN.

Although these examples are simplifications, they show the kinds of configurations firewalls protect. Next, we review the kinds of attacks against which firewalls can and cannot protect.

What Firewalls Can—and Cannot—Block

As we have seen, firewalls are not complete solutions to all computer security problems. A firewall protects only the perimeter of its environment against attacks from outsiders who want to execute code or access data on the machines in the protected environment. Keep in mind these points about firewalls.

- firewalls can protect an environment only if the firewalls control the entire perimeter. That is, firewalls are effective only if no unmediated connections breach the perimeter. If even one inside host connects to an outside address, by a modem for example, the entire inside net is vulnerable through the modem and its host.
- firewalls do not protect data outside the perimeter; data that have properly passed (outbound) through the firewall are just as exposed as if there were no firewall.

- firewalls are the most visible part of an installation to the outside, so they are the most attractive target for attack. For this reason, several different layers of protection, called **defense in depth,** are better than relying on the strength of just a single firewall.
- firewalls must be correctly configured, that configuration must be updated as the internal and external environment changes, and firewall activity reports must be reviewed periodically for evidence of attempted or successful intrusion.
- firewalls are targets for penetrators. While a firewall is designed to withstand attack, it is not impenetrable. Designers intentionally keep a firewall small and simple so that even if a penetrator breaks it, the firewall does not have further tools, such as compilers, linkers, loaders, and the like, to continue an attack.
- firewalls exercise only minor control over the content admitted to the inside, meaning that inaccurate data or malicious code must be controlled by other means inside the perimeter.

Firewalls are important tools in protecting an environment connected to a network. However, the environment must be viewed as a whole, all possible exposures must be considered, and the firewall must fit into a larger, comprehensive security strategy. Firewalls alone cannot secure an environment.

7.5 INTRUSION DETECTION SYSTEMS

After the perimeter controls, firewall, and authentication and access controls block certain actions, some users are admitted to use a computing system. Most of these controls are preventive: They block known bad things from happening. Many studies (for example, see [DUR99]) have shown that most computer security incidents are caused by insiders, people who would not be blocked by a firewall. And insiders require access with significant privileges to do their daily jobs. The vast majority of harm from insiders is not malicious; it is honest people making honest mistakes. Then, too, there are the potential malicious outsiders who have somehow passed the screens of firewalls and access controls. Prevention, although necessary, is not a complete computer security control; detection during an incident copes with harm that cannot be prevented in advance. Halme and Bauer [HAL95] survey the range of controls to address intrusions.

Intrusion detection systems complement these preventive controls as the next line of defense. An **intrusion detection system (IDS)** is a device, typically another separate computer, that monitors activity to identify malicious or suspicious events. Kemmerer and Vigna [KEM02] survey the history of IDSs. An IDS is a sensor, like a smoke detector, that raises an alarm if specific things occur. A model of an IDS is shown in Figure 7-45. The components in the figure are the four basic elements of an intrusion detection system, based on the Common Intrusion Detection Framework of [STA96]. An IDS receives raw inputs from sensors. It saves those inputs, analyzes them, and takes some controlling action.

IDSs perform a variety of functions:

- monitoring users and system activity
- auditing system configuration for vulnerabilities and misconfigurations

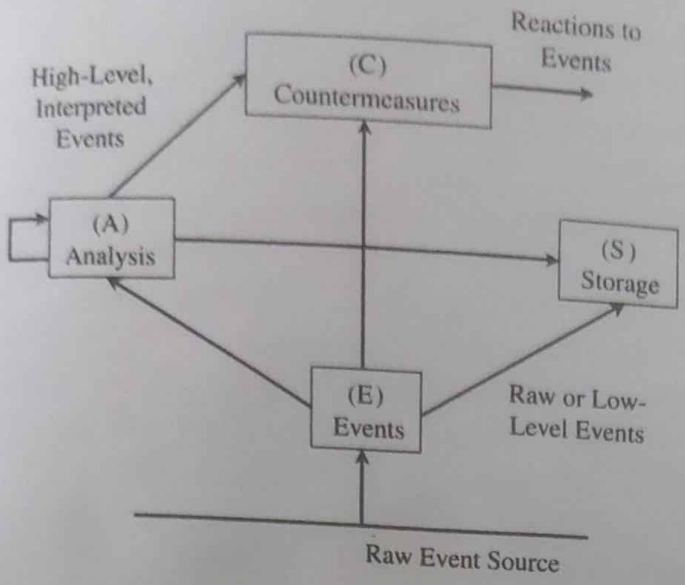

FIGURE 7-45 Common Components of an Intrusion Detection Framework.

- assessing the integrity of critical system and data files
- recognizing known attack patterns in system activity
- identifying abnormal activity through statistical analysis
- managing audit trails and highlighting user violation of policy or normal activity
- correcting system configuration errors
- installing and operating traps to record information about intruders

No one IDS performs all of these functions. Let us look more closely at the kinds of IDSs and their use in providing security.

Types of IDSs

The two general types of intrusion detection systems are signature based and heuristic. **Signature-based** intrusion detection systems perform simple pattern-matching and report situations that match a pattern corresponding to a known attack type. **Heuristic** intrusion detection systems, also known as **anomaly based,** build a model of acceptable behavior and flag exceptions to that model; for the future, the administrator can mark a flagged behavior as acceptable so that the heuristic IDS will now treat that previously unclassified behavior as acceptable.

Intrusion detection devices can be network based or host based. A **network-based** IDS is a stand-alone device attached to the network to monitor traffic throughout that network; a **host-based** IDS runs on a single workstation or client or host, to protect that one host.

Early intrusion detection systems (for example, [DEN87b, LUN90a, FOX90, LIE89]) worked after the fact, by reviewing logs of system activity to spot potential misuses that had occurred. The administrator could review the results of the IDS to find and fix weaknesses in the system. Now, however, intrusion detection systems operate in real time (or near real time), watching activity and raising alarms in time for the administrator to take protective action.

Signature-Based Intrusion Detection

A simple signature for a known attack type might describe a series of TCP SYN packets sent to many different ports in succession and at times close to one another, as would be the case for a port scan. An intrusion detection system would probably find nothing unusual in the first SYN, say, to port 80, and then another (from the same source address) to port 25. But as more and more ports receive SYN packets, especially ports that are not open, this pattern reflects a possible port scan. Similarly, some implementations of the protocol stack fail if they receive an ICMP packet with a data length of 65535 bytes, so such a packet would be a pattern for which to watch.

The problem with signature-based detection is the signatures themselves. An attacker will try to modify a basic attack in such a way that it will not match the known signature of that attack. For example, the attacker may convert lowercase to uppercase letters or convert a symbol such as "blank space" to its character code equivalent %20. The IDS must necessarily work from a canonical form of the data stream in order to recognize that %20 matches a pattern with a blank space. The attacker may insert malformed packets that the IDS will see, to intentionally cause a pattern mismatch; the protocol handler stack will discard the packets because of the malformation. Each of these variations could be detected by an IDS, but more signatures require additional work for the IDS, which reduces performance.

Of course, signature-based IDSs cannot detect a new attack for which a signature is not yet installed in the database. Every attack type starts as a new pattern at some time, and the IDS is helpless to warn of its existence.

Signature-based intrusion detection systems tend to use **statistical analysis.** This approach uses statistical tools both to obtain sample measurements of key indicators (such as amount of external activity, number of active processes, number of transactions) and to determine whether the collected measurements fit the predetermined attack signatures.

Ideally, signatures should match every instance of an attack, match subtle variations of the attack, but not match traffic that is not part of an attack. However, this goal is grand but unreachable.

Heuristic Intrusion Detection

Because signatures are limited to specific, known attack patterns, another form of intrusion detection becomes useful. Instead of looking for matches, heuristic intrusion detection looks for behavior that is out of the ordinary. The original work in this area (for example, [TEN90]) focused on the individual, trying to find characteristics of that person that might be helpful in understanding normal and abnormal behavior. For example, one user might always start the day by reading e-mail, write many documents using a word processor, and occasionally back up files. These actions would be normal. This user does not seem to use many administrator utilities. If that person tried to access sensitive system management utilities, this new behavior might be a clue that someone else was acting under the user's identity.

If we think of a compromised system in use, it starts clean, with no intrusion, and it ends dirty, fully compromised. There may be no point in the trace of use in which the

system changed from clean to dirty; it was more likely that little dirty events occurred occasionally at first and then increasing as the system became more deeply compromised. Any one of those events might be acceptable by itself, but the accumulation of them and the order and speed at which they occurred could have been signals that something unacceptable was happening. The inference engine of an intrusion detection system performs continuous analysis of the system, raising an alert when the system's dirtiness exceeds a threshold.

Inference engines work in two ways. Some, called **state-based** intrusion detection systems, see the system going through changes of overall state or configuration. They try to detect when the system has veered into unsafe modes. Others try to map current activity onto a model of unacceptable activity and raise an alarm when the activity resembles the model. These are called **model-based** intrusion detection systems. This approach has been extended to networks in [MUK94]. Later work (for example, [FOR96, LIN99]) sought to build a dynamic model of behavior, to accommodate variation and evolution in a person's actions over time. The technique compares real activity with a known representation of normality.

Alternatively, intrusion detection can work from a model of known bad activity. For example, except for a few utilities (login, change password, create user), any other attempt to access a password file is suspect. This form of intrusion detection is known as **misuse intrusion detection.** In this work, the real activity is compared against a known suspicious area.

All heuristic intrusion detection activity is classified in one of three categories: good/benign, suspicious, or unknown. Over time, specific kinds of actions can move from one of these categories to another, corresponding to the IDS's learning whether certain actions are acceptable or not.

As with pattern-matching, heuristic intrusion detection is limited by the amount of information the system has seen (to classify actions into the right category) and how well the current actions fit into one of these categories.

Stealth Mode

An IDS is a network device (or, in the case of a host-based IDS, a program running on a network device). Any network device is potentially vulnerable to network attacks. How useful would an IDS be if it itself were deluged with a denial-of-service attack? If an attacker succeeded in logging in to a system within the protected network, wouldn't trying to disable the IDS be the next step?

To counter those problems, most IDSs run in **stealth mode,** whereby an IDS has two network interfaces: one for the network (or network segment) being monitored and the other to generate alerts and perhaps other administrative needs. The IDS uses the monitored interface as input only; it *never* sends packets out through that interface. Often, the interface is configured so that the device has no published address through the monitored interface; that is, a router cannot route anything to that address directly, because the router does not know such a device exists. It is the perfect passive wiretap. If the IDS needs to generate an alert, it uses only the alarm interface on a completely separate control network. Such an architecture is shown in Figure 7-46.

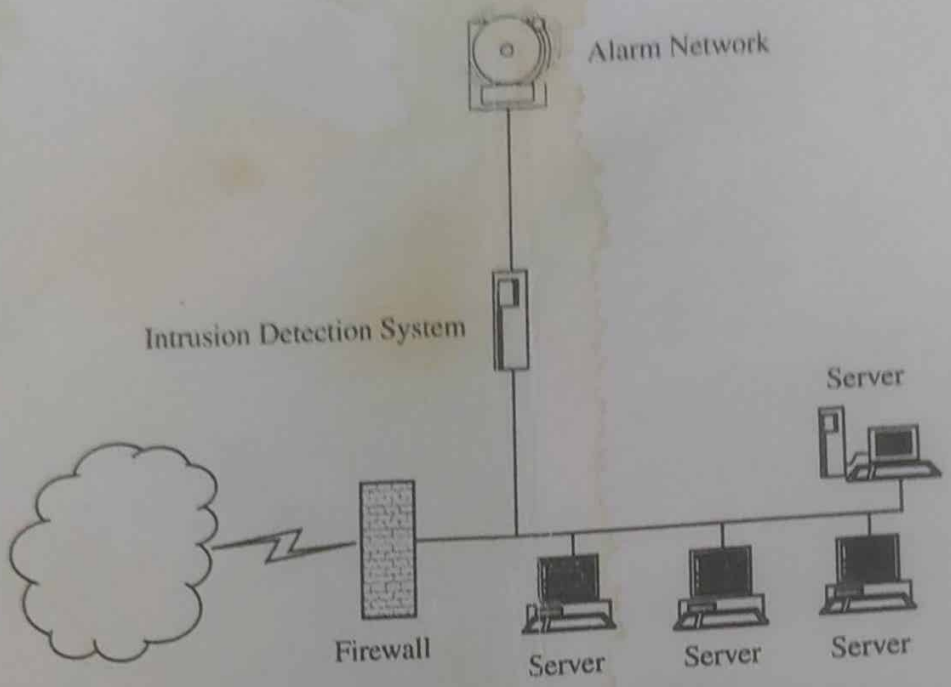

FIGURE 7-46 Stealth Mode IDS Connected to Two Networks.

Other IDS Types

Some security engineers consider other devices to be IDSs as well. For instance, to detect unacceptable code modification, programs can compare the active version of a software code with a saved version of a digest of that code. The *tripwire* program [KIM98] is the most well known software (or static data) comparison program. You run *tripwire* on a new system, and it generates a hash value for each file; then you save these hash values in a secure place (offline, so that no intruder can modify them while modifying a system file). If you later suspect your system may have been compromised, you rerun *tripwire*, providing it the saved hash values. It recomputes the hash values and reports any mismatches, which would indicate files that were changed.

System vulnerability scanners, such as *ISS Scanner* or *Nessus*, can be run against a network. They check for known vulnerabilities and report flaws found.

As we have seen, a honeypot is a faux environment intended to lure an attacker. It can be considered an IDS, in the sense that the honeypot may record an intruder's actions and even attempt to trace who the attacker is from actions, packet data, or connections.

Goals for Intrusion Detection Systems

The two styles of intrusion detection—pattern matching and heuristic—represent different approaches, each of which has advantages and disadvantages. Actual IDS products often blend the two approaches.

Ideally, an IDS should be fast, simple, and accurate, while at the same time being complete. It should detect all attacks with little performance penalty. An IDS could use some—or all—of the following design approaches: