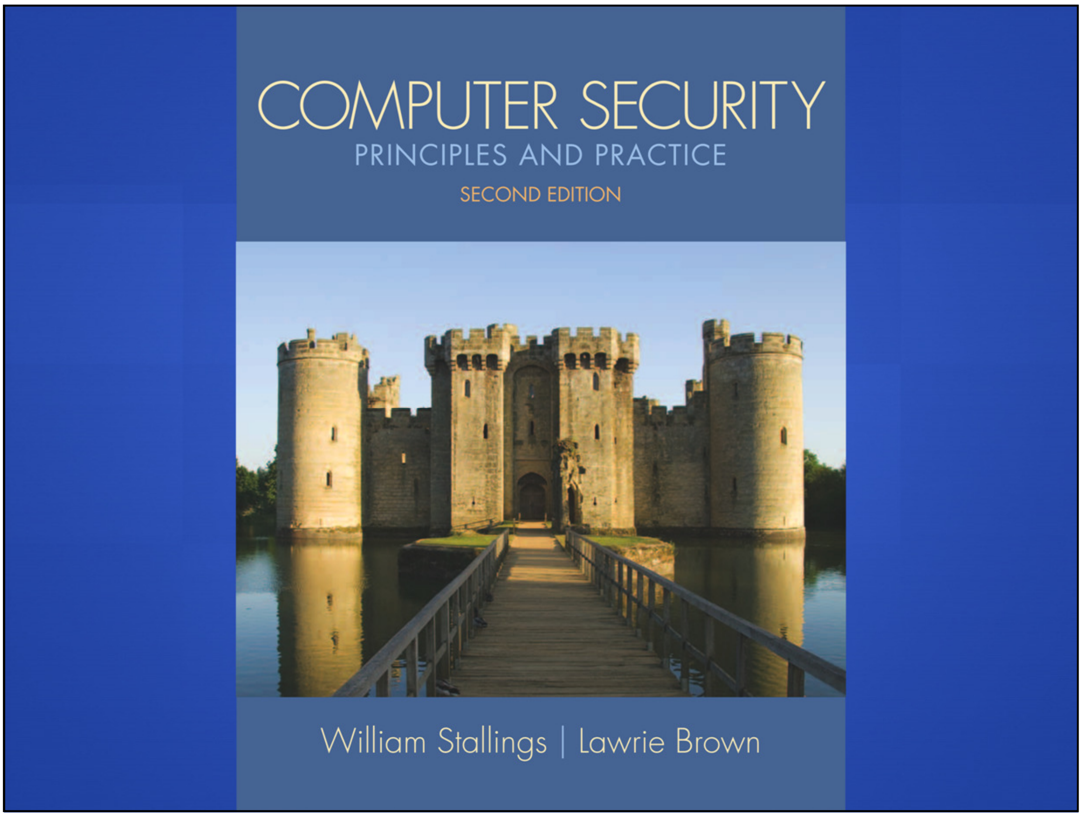
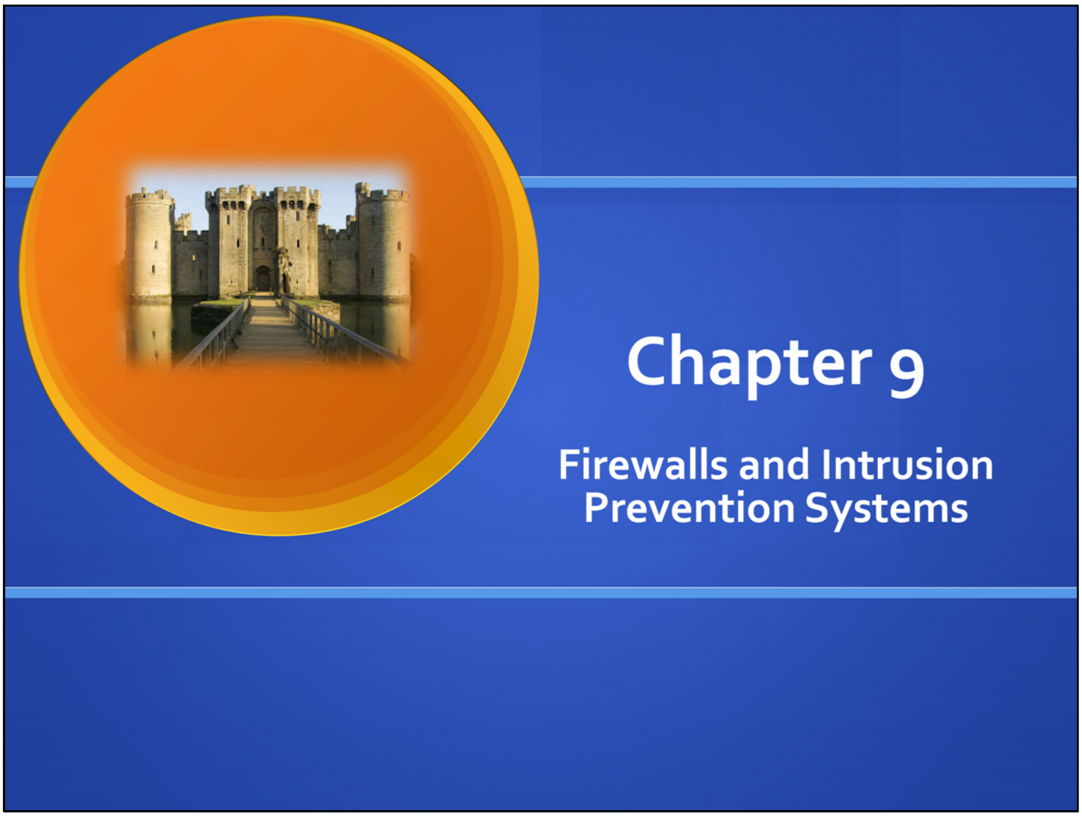


Lecture slides for "Computer Security: Principles and Practice", 2/e, by William Stallings and Lawrie Brown, Chapter 9 "Firewalls and Intrusion Prevention Systems".



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

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

• Centralized data processing system, with a central mainframe supporting a number of directly connected terminals

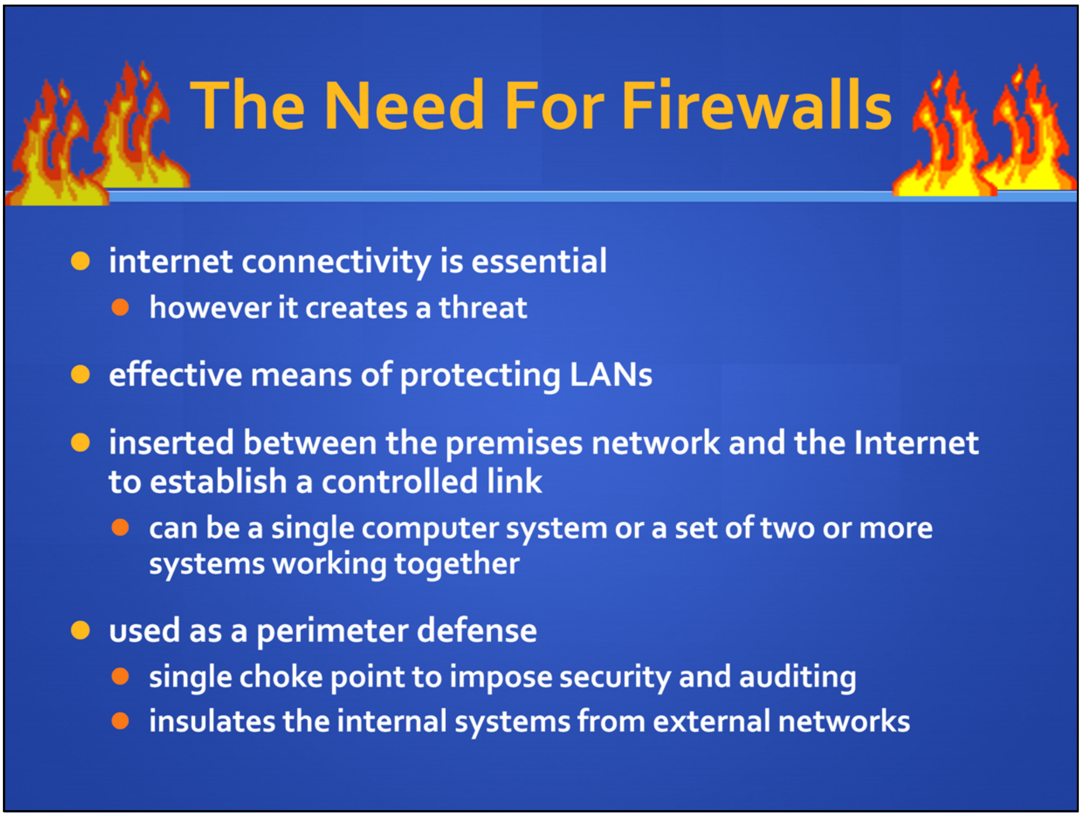
• Local area networks (LANs) interconnecting PCs and terminals to each other and the mainframe

• Premises network, consisting of a number of LANs, interconnecting PCs, servers, and perhaps a mainframe or two

• Enterprise-wide network, consisting of multiple, geographically distributed premises networks interconnected by a private wide area network (WAN)

• Internet connectivity, in which the various premises networks all hook into the

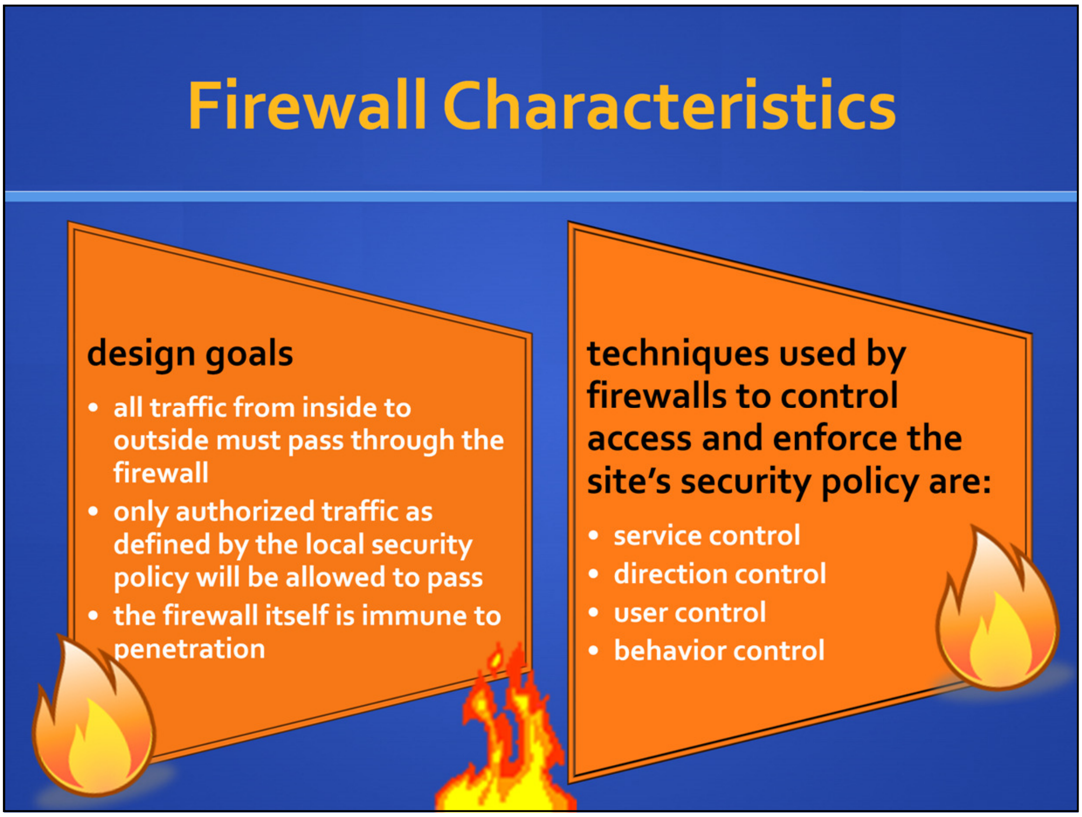
Internet and may or may not also be connected by a private WAN



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

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

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



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

1. All traffic from inside to outside, and vice versa, must pass through the firewall.

This is achieved by physically blocking all access to the local network except via the firewall. Various configurations are possible, as explained later in this chapter.

2. Only authorized traffic, as defined by the local security policy, will be allowed

to pass. Various types of firewalls are used, which implement various types of security policies, as explained later in this chapter.

3. The firewall itself is immune to penetration. This implies the use of a hardened

system with a secured operating system. Trusted computer systems are suitable for hosting a firewall and often required in government applications. This topic is discussed in Chapter 13 .

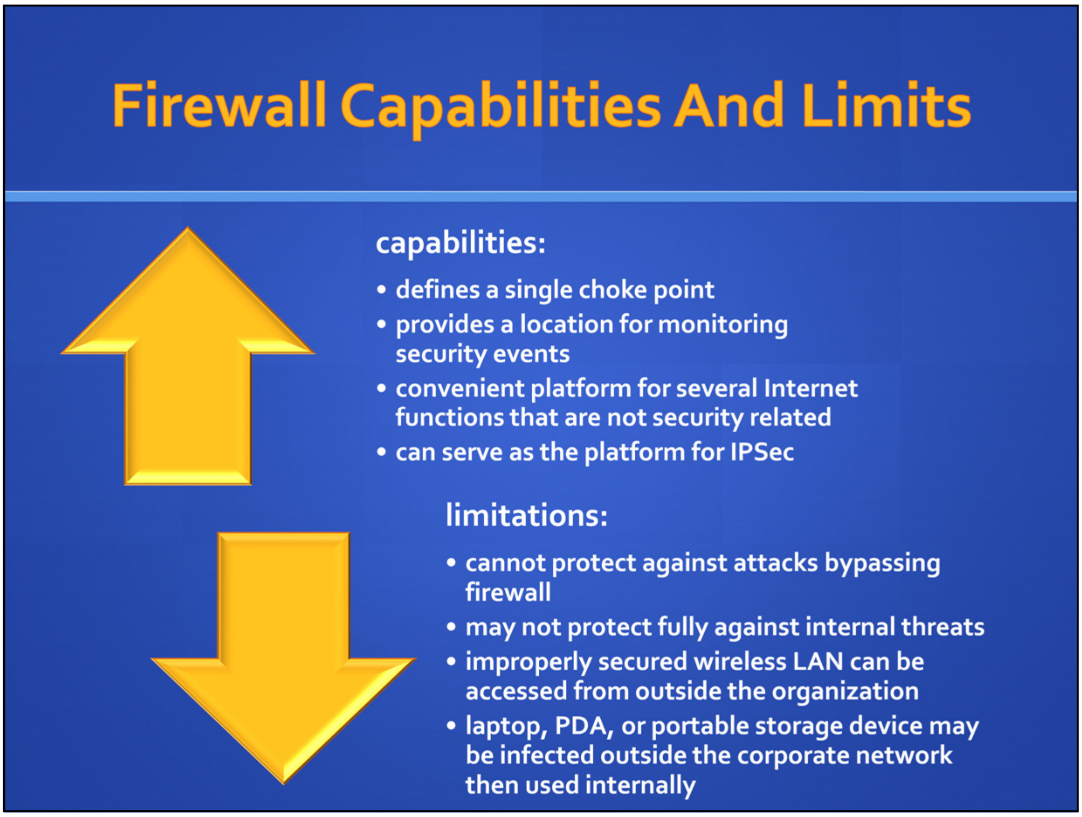
[SMIT97] lists four general techniques that firewalls use to control access and enforce the site's security policy. Originally, firewalls focused primarily on service control, but they have since evolved to provide all four:

• **Service control: Determines the types of Internet services that can be accessed,**
inbound or outbound. The firewall may filter traffic on the basis of IP address, protocol, or port number; may provide proxy software that receives and interprets each service request before passing it on; or may host the server software itself, such as a Web or mail service.

• **Direction control: Determines the direction in which particular service**
requests may be initiated and allowed to flow through the firewall.

• **User control: Controls access to a service according to which user is attempting**
to access it. This feature is typically applied to users inside the firewall perimeter (local users). It may also be applied to incoming traffic from external users; the latter requires some form of secure authentication technology, such as is provided in IPSec (Chapter 22).

Behavior control: Controls how particular services are used. For example, the
firewall may filter e-mail to eliminate spam, or it may enable external access to only a portion of the information on a local Web server.



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

1. A firewall defines a single choke point that attempts to keep unauthorized
users out of the protected network, prohibit potentially vulnerable services from entering or leaving the network, and provide protection from various kinds of IP spoofing and routing attacks. The use of a single choke point simplifies security management because security capabilities are consolidated on a single system or set of systems.

2. A firewall provides a location for monitoring security-related events. Audits
and alarms can be implemented on the firewall system.

3. A firewall is a convenient platform for several Internet functions that are not
security related. These include a network address translator, which maps local

addresses to Internet addresses, and a network management function that audits or logs Internet usage.

4. A firewall can serve as the platform for IPSec. Using the tunnel mode capability
described in Chapter 22 , the firewall can be used to implement virtual private networks.

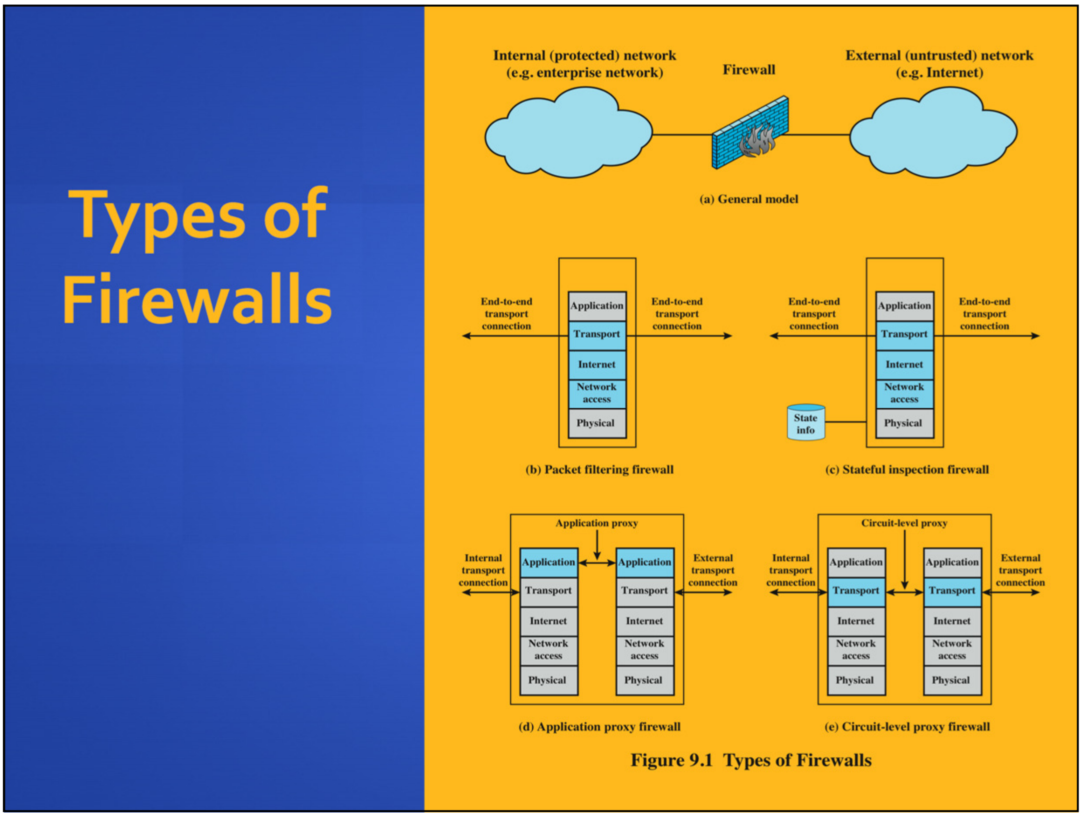
Firewalls have their limitations, including the following:

1. The firewall cannot protect against attacks that bypass the firewall. Internal
systems may have dial-out or mobile broadband capability to connect to an ISP. An internal LAN may support a modem pool that provides dial-in capability for traveling employees and telecommuters.

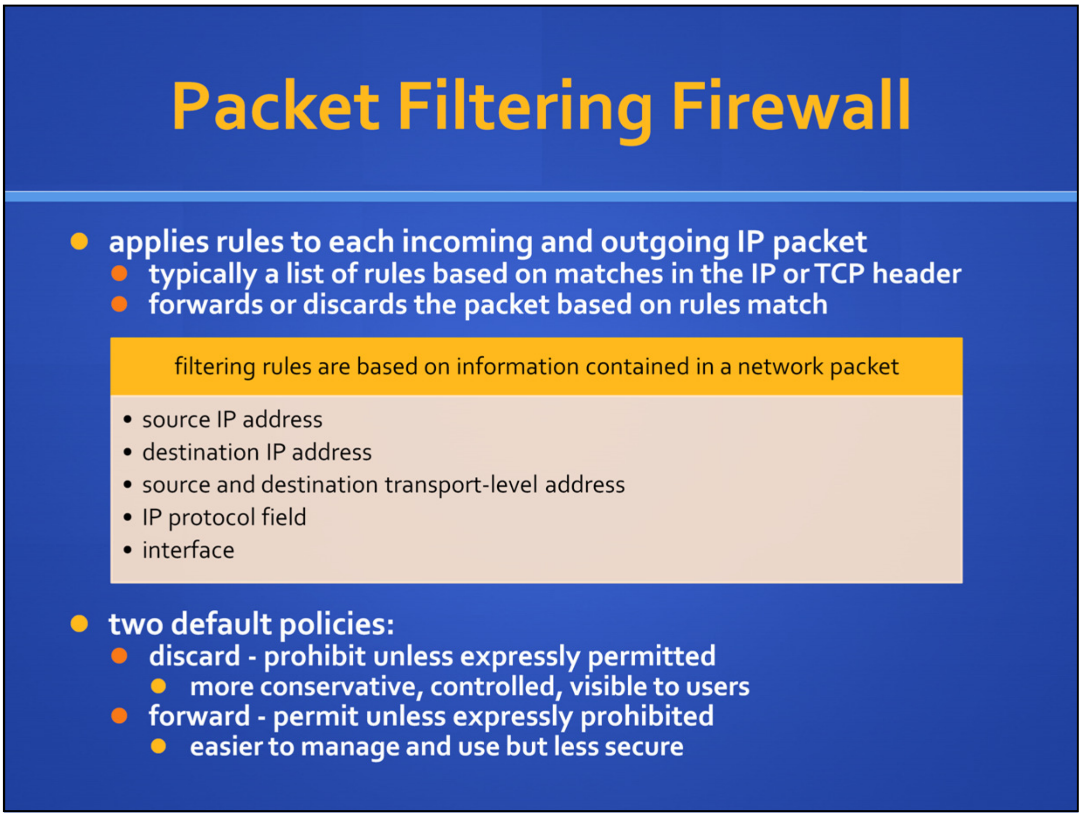
2. The firewall may not protect fully against internal threats, such as a disgruntled
employee or an employee who unwittingly cooperates with an external attacker.

3. An improperly secured wireless LAN may be accessed from outside the
organization. An internal firewall that separates portions of an enterprise network cannot guard against wireless communications between local systems on different sides of the internal firewall.

4. A laptop, PDA, or portable storage device may be used and infected outside
the corporate network and then attached and used internally.



A firewall may act as a packet filter. It can operate as a positive filter, allowing to pass only packets that meet specific criteria, or as a negative filter, rejecting any packet that meets certain criteria. Depending on the type of firewall, it may examine one or more protocol headers in each packet, the payload of each packet, or the pattern generated by a sequence of packets. In this section, we look at the principal types of firewalls.



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

• **Source IP address: The IP address of the system that originated the IP packet**
(e.g., 192.178.1.1)

• **Destination IP address: The IP address of the system the IP packet is trying to**
reach (e.g., 192.168.1.2)

• **Source and destination transport-level address: The transport-level (e.g., TCP**
or UDP) port number, which defines applications such as SNMP or TELNET

• **IP protocol field: Defines the transport protocol**

• **Interface: For a firewall with three or more ports, which interface of the firewall**
the packet came from or which interface of the firewall the packet is destined for

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

• **Default discard: That which is not expressly permitted is prohibited.**

• **Default forward: That which is not expressly prohibited is permitted.**

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

Table 9.1 Packet Filter Rules

Rule Set A

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

Rule Set B

action	ourhost	port	theirhost	port	comment
block	*	*	*	*	default

Rule Set C

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

Rule Set D

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

Rule Set E

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

Table 9.1 , from [BELL94], gives some examples of packet filtering rule sets. In each set, the rules are applied top to bottom. The "*" in a field is a wildcard designator that matches everything. We assume that the default discard policy is in force.

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

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

C. This rule set is intended to specify that any inside host can send mail to the
outside. A TCP packet with a destination port of 25 is routed to the SMTP server on the destination machine. The problem with this rule is that the use of port 25 for SMTP receipt is only a default; an outside machine could be configured

to have some other application linked to port 25. As this rule is written, an attacker could gain access to internal machines by sending packets with a TCP source port number of 25.

D. This rule set achieves the intended result that was not achieved in C. The rules

take advantage of a feature of TCP connections. Once a connection is set up, the ACK flag of a TCP segment is set to acknowledge segments sent from the other side. Thus, this rule set states that it allows IP packets where the source IP address is one of a list of designated internal hosts and the destination TCP port number is 25. It also allows incoming packets with a source port number of 25 that include the ACK flag in the TCP segment. Note that we explicitly designate source and destination systems to define these rules explicitly.

E. This rule set is one approach to handling FTP connections. With FTP, two TCP

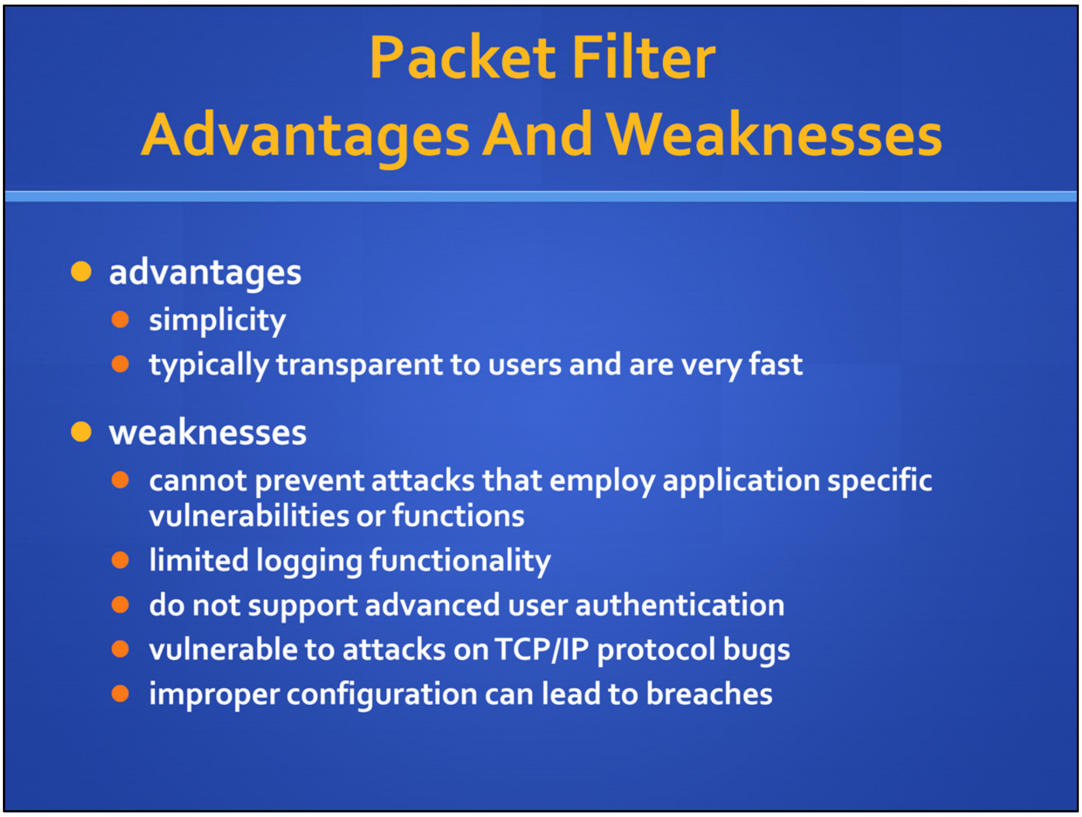
connections are used: a control connection to set up the file transfer and a data connection for the actual file transfer. The data connection uses a different port number that is dynamically assigned for the transfer. Most servers, and hence most attack targets, use low-numbered ports; most outgoing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

• Packets that originate internally

• Reply packets to a connection initiated by an internal machine

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

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



One advantage of a packet filtering firewall is its simplicity. Also, packet filters typically are transparent to users and are very fast. [SCAR09b] lists the following weaknesses of packet filter firewalls:

• Because packet filter firewalls do not examine upper-layer data, they cannot prevent attacks that employ application-specific vulnerabilities or functions. For example, a packet filter firewall cannot block specific application commands; if a packet filter firewall allows a given application, all functions available within that application will be permitted.

• Because of the limited information available to the firewall, the logging functionality present in packet filter firewalls is limited. Packet filter logs normally contain the same information used to make access control decisions (source address, destination address, and traffic type).

• Most packet filter firewalls do not support advanced user authentication schemes. Once again, this limitation is mostly due to the lack of upper-layer functionality by the firewall.

• Packet filter firewalls are generally vulnerable to attacks and exploits that take advantage of problems within the TCP/IP specification and protocol stack, such as *network layer address spoofing . Many packet filter firewalls cannot* detect a network packet in which the OSI Layer 3 addressing information has been altered. Spoofing attacks are generally employed by intruders to bypass the security controls implemented in a firewall platform.

• Finally, due to the small number of variables used in access control decisions, packet filter firewalls are susceptible to security breaches caused by improper configurations. In other words, it is easy to accidentally configure a packet filter firewall to allow traffic types, sources, and destinations that should be denied based on an organization's information security policy.

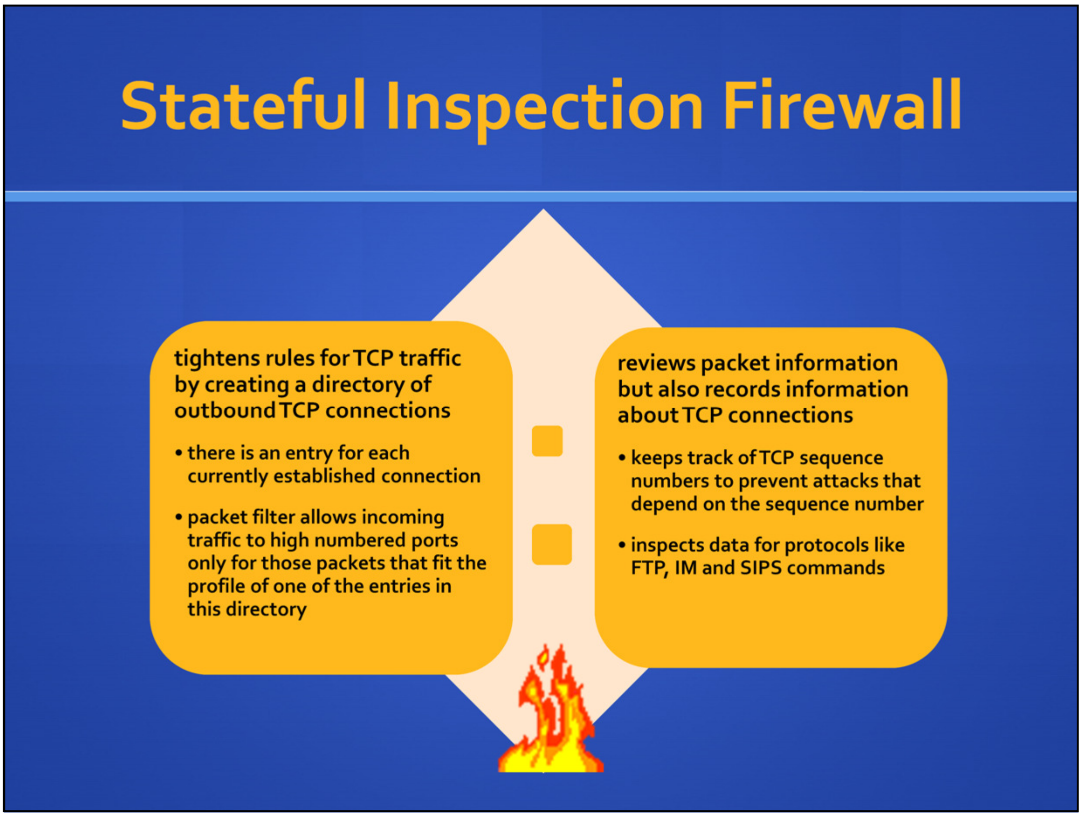
Some of the attacks that can be made on packet filtering firewalls and the appropriate countermeasures are the following:

• **IP address spoofing : The intruder transmits packets from the outside with a**
source IP address field containing an address of an internal host. The attacker hopes that the use of a spoofed address will allow penetration of systems that employ simple source address security, in which packets from specific trusted internal hosts are accepted. The countermeasure is to discard packets with an inside source address if the packet arrives on an external interface. In fact, this countermeasure is often implemented at the router external to the firewall.

• **Source routing attacks: The source station specifies the route that a packet**
should take as it crosses the Internet, in the hopes that this will bypass security measures that do not analyze the source routing information. A countermeasure is to discard all packets that use this option.

• **Tiny fragment attacks : The intruder uses the IP fragmentation option to create**
extremely small fragments and force the TCP header information into a separate packet fragment. This attack is designed to circumvent filtering rules that depend on TCP header information. Typically, a packet filter will make a filtering decision on the first fragment of a packet. All subsequent

fragments of that packet are filtered out solely on the basis that they are part of the pac-ket whose first fragment was rejected. The attacker hopes that the filtering firewall examines only the first fragment and that the remaining fragments are passed through. A tiny fragment attack can be defeated by enforcing a rule that the first fragment of a packet must contain a predefined minimum amount of the transport header. If the first fragment is rejected, the filter can remember the packet and discard all subsequent fragments.



A traditional packet filter makes filtering decisions on an individual packet basis and does not take into consideration any higher-layer context. To understand what

is meant by *context and why a traditional packet filter is limited with regard to context,*

a little background is needed. Most standardized applications that run on top of TCP follow a client/server model. For example, for the Simple Mail Transfer Protocol (SMTP), e-mail is transmitted from a client system to a server system. The client system generates new e-mail messages, typically from user input. The server system accepts incoming e-mail messages and places them in the appropriate

user mailboxes. SMTP operates by setting up a TCP connection between client and server, in which the TCP server port number, which identifies the SMTP server

application, is 25. The TCP port number for the SMTP client is a number between 1024 and 65535 that is generated by the SMTP client.

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

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

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

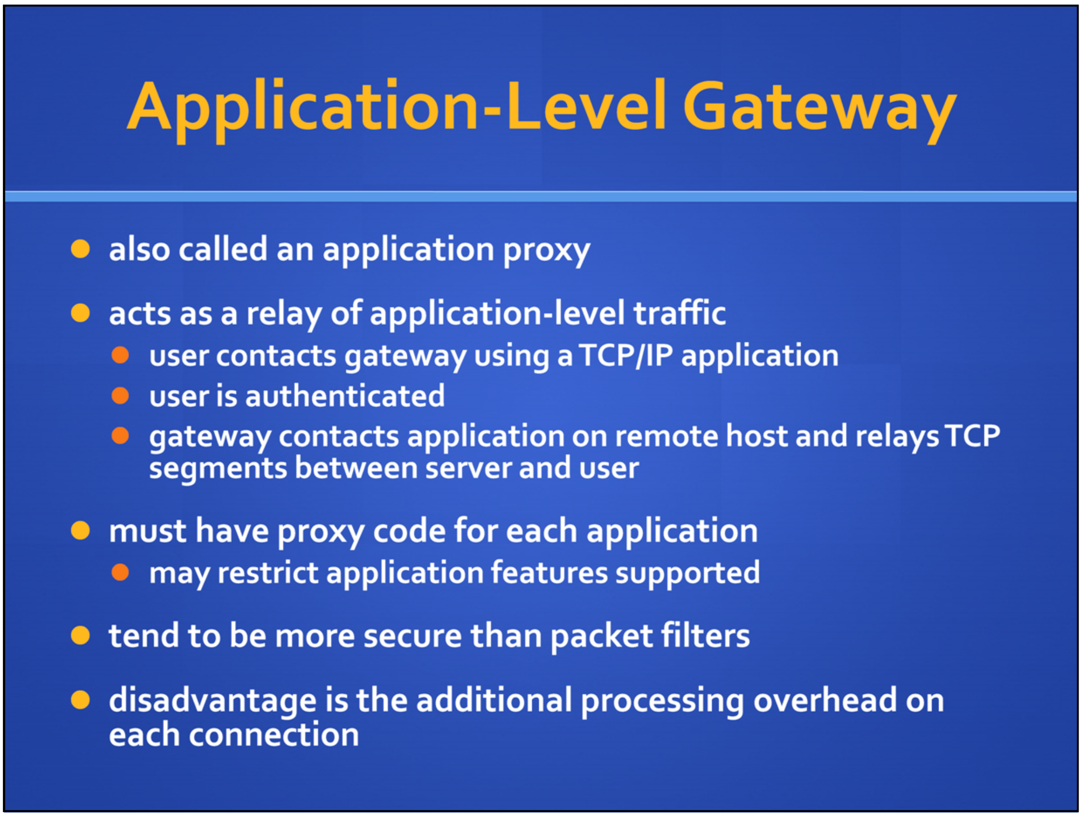
Example

Stateful Firewall Connection State Table

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

Table 9.2 Example Stateful Firewall Connection State Table [WACK02]

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

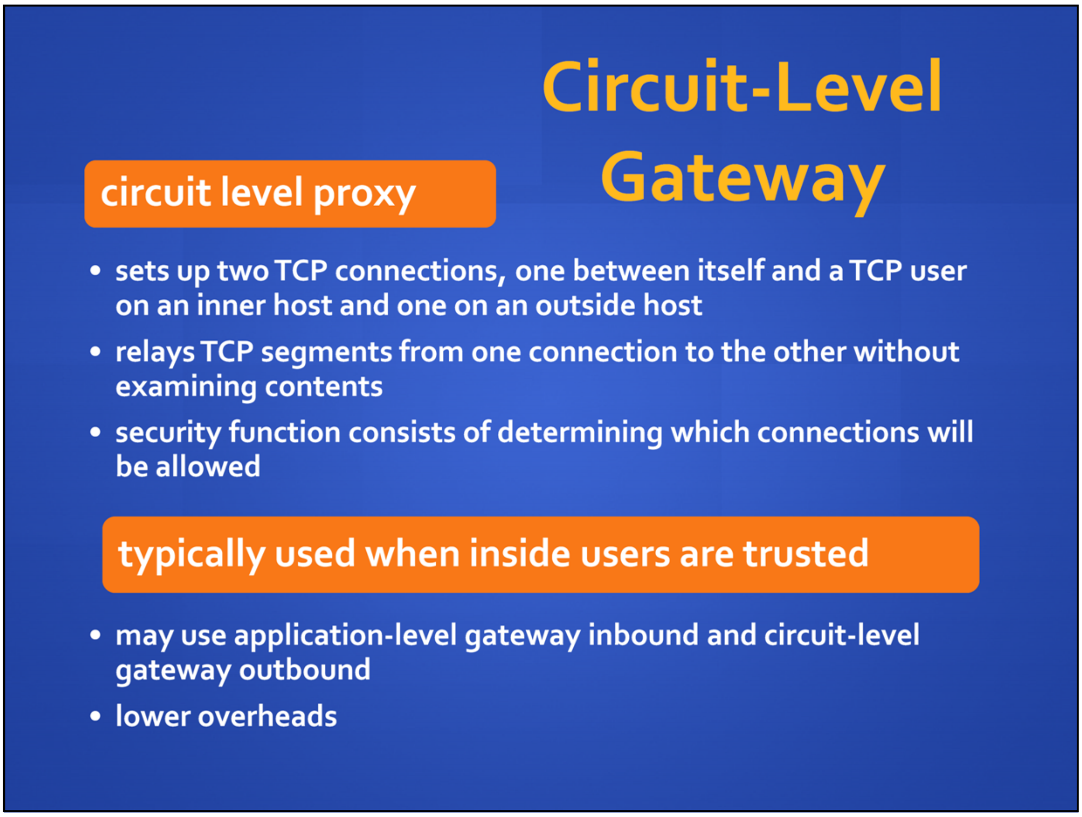


An application-level gateway, also called an **application proxy , acts as a relay of**
application-level traffic (Figure 9.1d). The user contacts the gateway using a TCP/
IP application, such as Telnet or FTP, and the gateway asks the user for the name
of the remote host to be accessed. When the user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

Application-level gateways tend to be more secure than packet filters. Rather
than trying to deal with the numerous possible combinations that are to be

allowed
and forbidden at the TCP and IP level, the application-level gateway need only scrutinize a few allowable applications. In addition, it is easy to log and audit all incoming traffic at the application level.

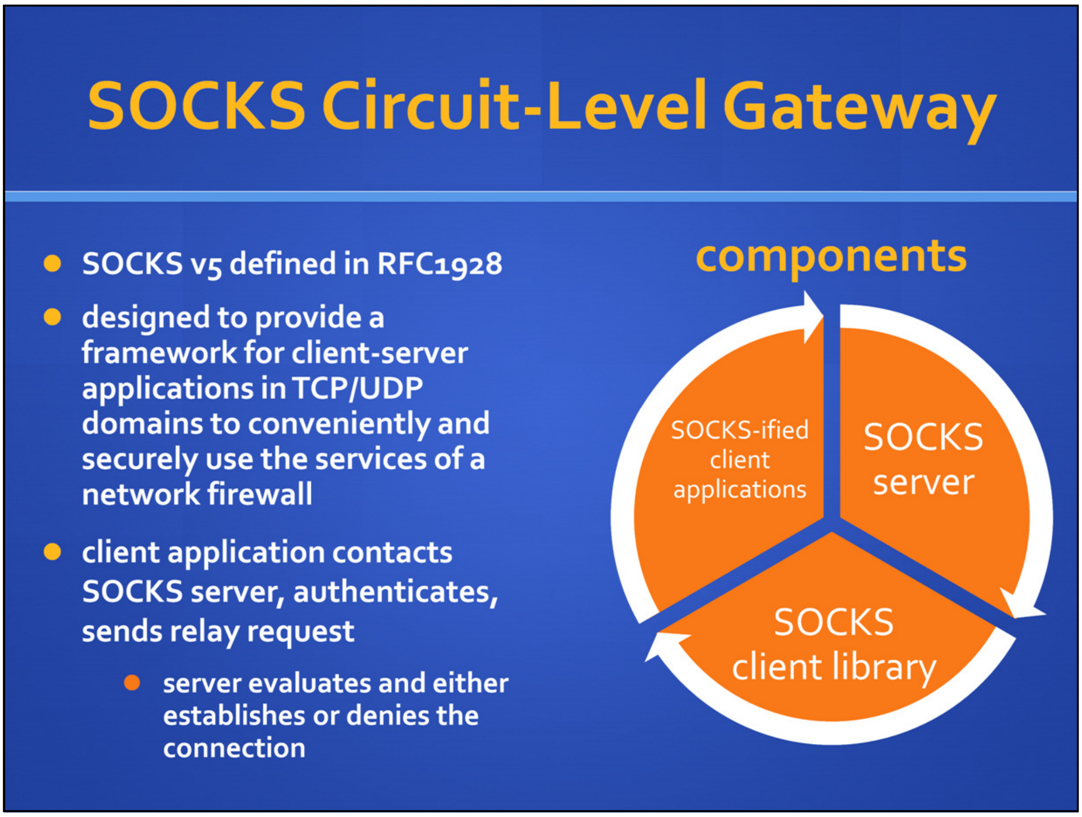
A prime disadvantage of this type of gateway is the additional processing overhead on each connection. In effect, there are two spliced connections between the end users, with the gateway at the splice point, and the gateway must examine and forward all traffic in both directions.



A fourth type of firewall is the circuit-level gateway or **circuit-level proxy (Figure 9.1e).**

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

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



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

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

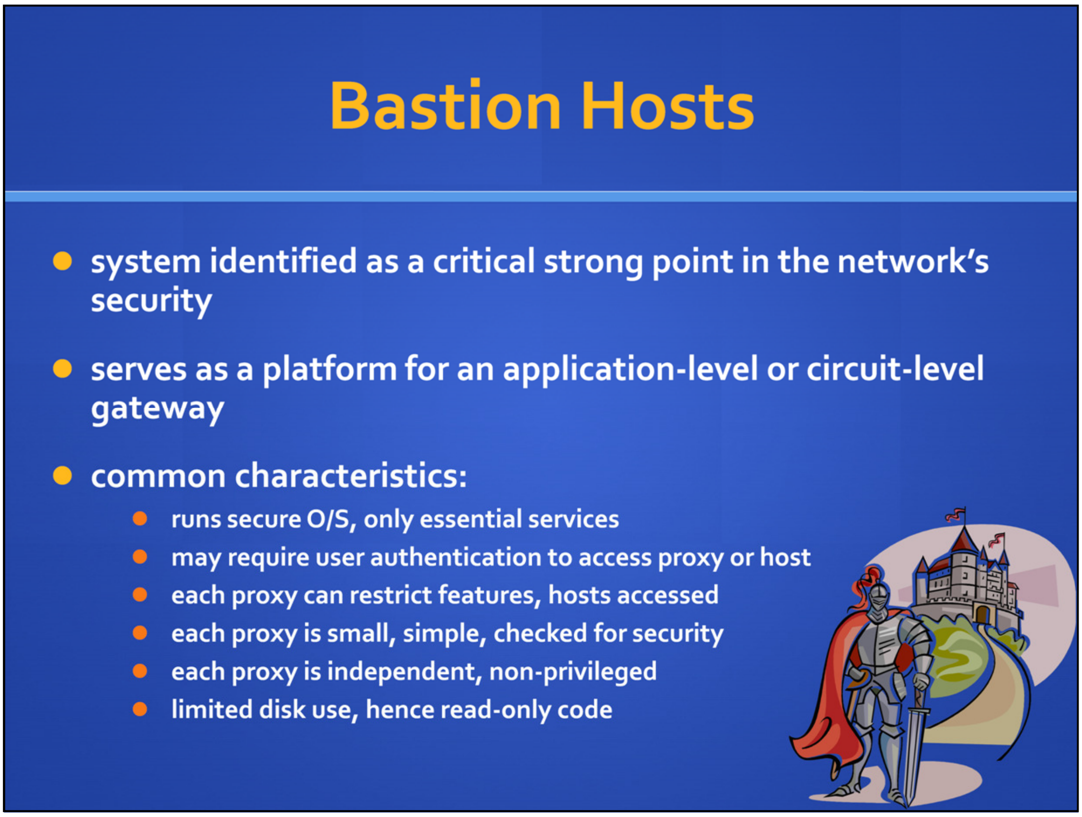
SOCKS consists of the following components:

• The SOCKS server, which often runs on a UNIX-based firewall. SOCKS is also implemented on Windows systems.

• The SOCKS client library, which runs on internal hosts protected by the firewall.

• SOCKS-ified versions of several standard client programs such as FTP and TELNET. The implementation of the SOCKS protocol typically involves either the recompilation or relinking of TCP-based client applications, or the use of alternate dynamically loaded libraries, to use the appropriate encapsulation routines in the SOCKS library.

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



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

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

• Only the services that the network administrator considers essential are installed on the bastion host. These could include proxy applications for DNS, FTP, HTTP, and SMTP.

• The bastion host may require additional authentication before a user is allowed access to the proxy services. In addition, each proxy service may require its own authentication before granting user access.

• Each proxy is configured to support only a subset of the standard application's command set.

• Each proxy is configured to allow access only to specific host systems. This means that the limited command/feature set may be applied only to a subset of systems on the protected network.

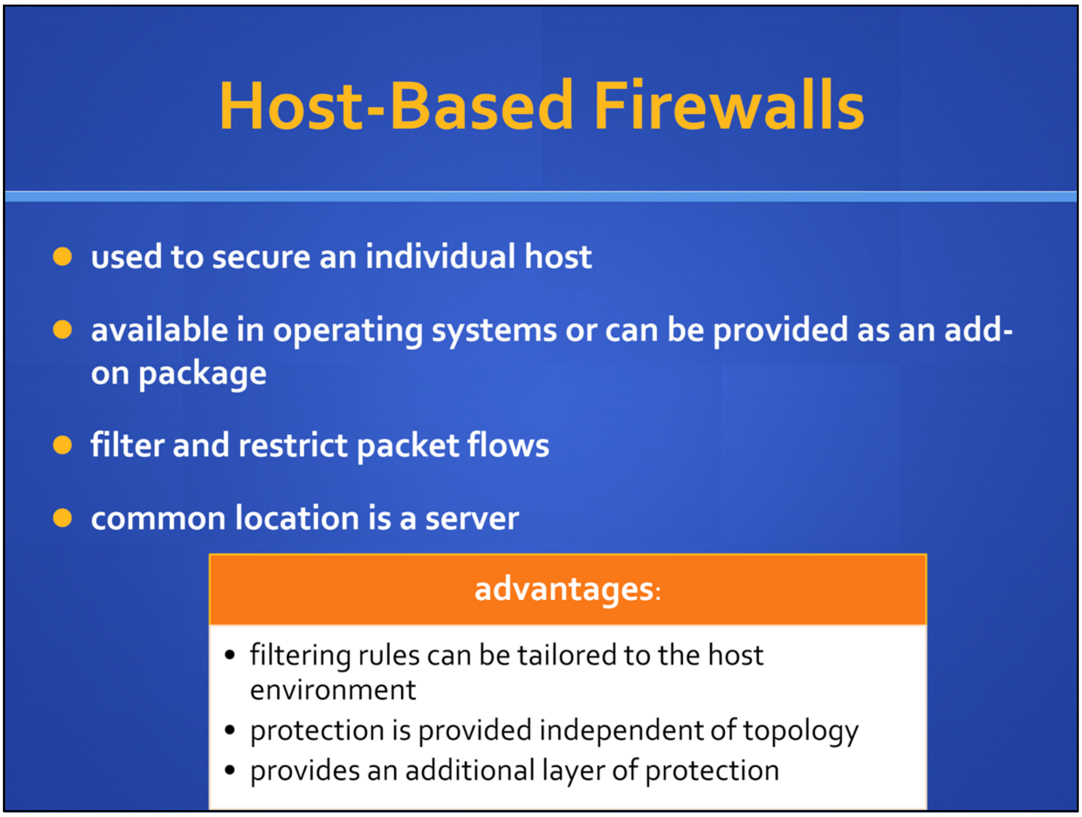
• Each proxy maintains detailed audit information by logging all traffic, each connection, and the duration of each connection. The audit log is an essential tool for discovering and terminating intruder attacks.

• Each proxy module is a very small software package specifically designed for network
security. Because of its relative simplicity, it is easier to check such modules for security flaws. For example, a typical UNIX mail application may contain over 20,000 lines of code, while a mail proxy may contain fewer than 1000.

• Each proxy is independent of other proxies on the bastion host. If there is a problem with the operation of any proxy, or if a future vulnerability is discovered, it can be uninstalled without affecting the operation of the other proxy applications. Also, if the user population requires support for a new service, the network administrator can easily install the required proxy on the bastion host.

• A proxy generally performs no disk access other than to read its initial configuration
file. Hence, the portions of the file system containing executable code can be made read only. This makes it difficult for an intruder to install Trojan horse sniffers or other dangerous files on the bastion host.

• Each proxy runs as a non-privileged user in a private and secured directory on the bastion host.

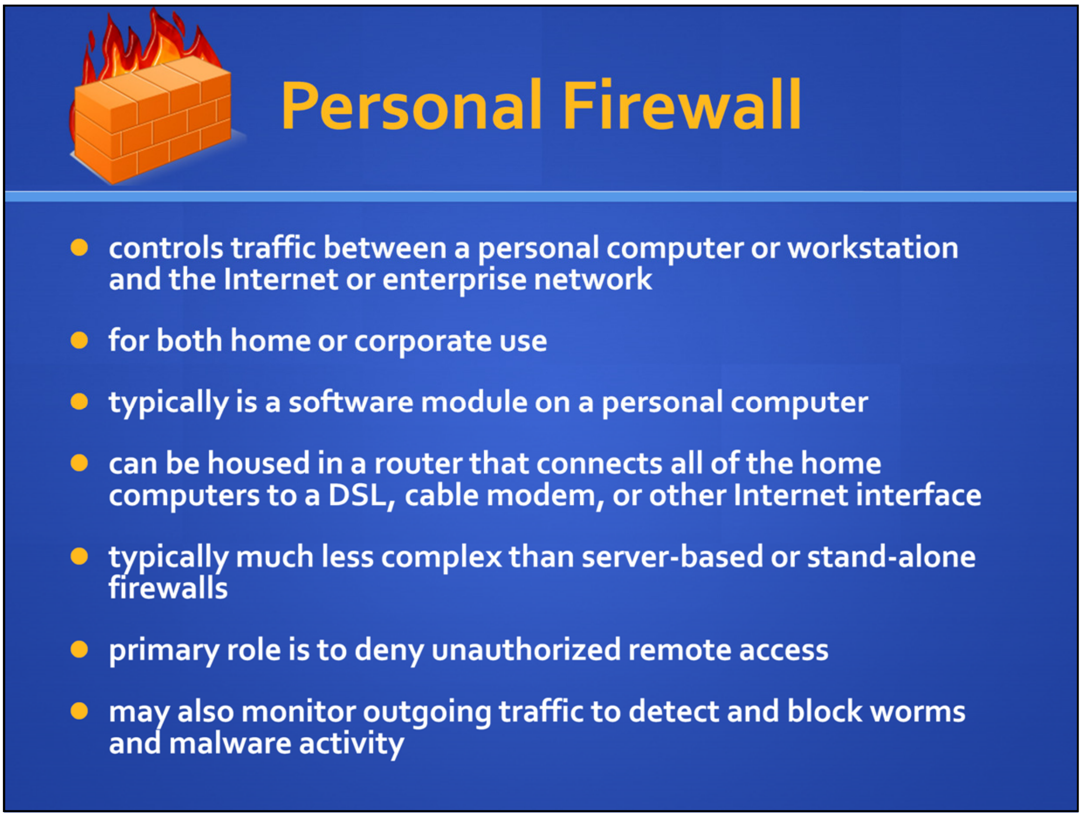


A host-based firewall is a software module used to secure an individual host. Such modules are available in many operating systems or can be provided as an add-on package. Like conventional stand-alone firewalls, host-resident firewalls filter and restrict the flow of packets. A common location for such firewalls is a server. There are several advantages to the use of a server-based or workstation-based firewall:

• Filtering rules can be tailored to the host environment. Specific corporate security policies for servers can be implemented, with different filters for servers used for different application.

• Protection is provided independent of topology. Thus both internal and external attacks must pass through the firewall.

• Used in conjunction with stand-alone firewalls, the host-based firewall provides an additional layer of protection. A new type of server can be added to the network, with its own firewall, without the necessity of altering the network firewall configuration.



A personal firewall controls the traffic between a personal computer or workstation
on one side and the Internet or enterprise network on the other side. Personal firewall
functionality can be used in the home environment and on corporate intranets.
Typically, the personal firewall is a software module on the personal computer. In
a home environment with multiple computers connected to the Internet, firewall
functionality can also be housed in a router that connects all of the home computers
to a DSL, cable modem, or other Internet interface.

Personal firewalls are typically much less complex than either server-based firewalls or stand-alone firewalls. The primary role of the personal firewall is to deny unauthorized remote access to the computer. The firewall can also monitor outgoing activity in an attempt to detect and block worms and other malware.

An example of a personal firewall is the capability built in to the Mac OS X operating system. When the user enables the personal firewall in Mac OS X, all inbound connections are denied except for those the user explicitly permits. The list of inbound services that can be selectively reenabled, with their port numbers, includes the following:

• Personal file sharing (548, 427)

• Windows sharing (139)

• Personal Web sharing (80, 427)

• Remote login—SSH (22)

• FTP access (20-21, 1024-65535 from 20-21)

• Remote Apple events (3031)

• Printer sharing (631, 515)

• IChat Rendezvous (5297, 5298)

• ITunes Music Sharing (3869)

CVS (2401)

• Gnutella/Limewire (6346)

• ICQ (4000)

• IRC (194)

• MSN Messenger (6891-6900)

• Network Time (123)

• Retrospect (497)

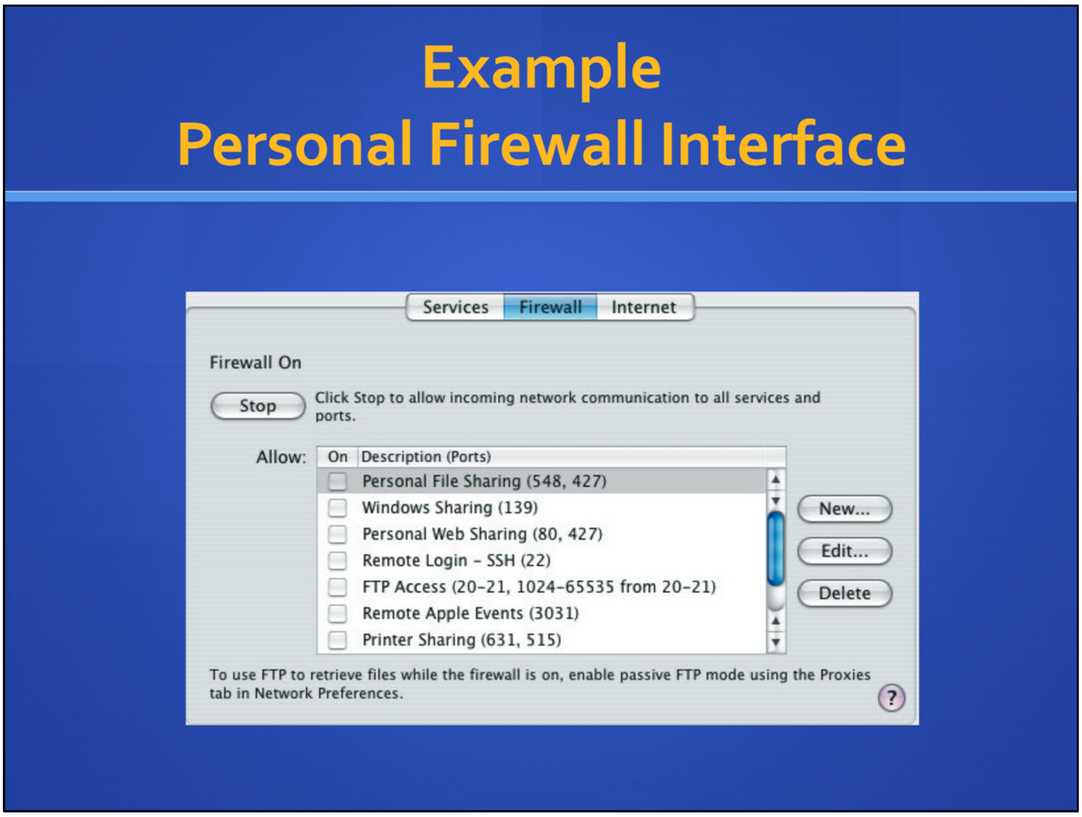
• SMB (without netbios–445)

• VNC (5900-5902)

• WebSTAR Admin (1080, 1443)

When FTP access is enabled, ports 20 and 21 on the local machine are opened for FTP; if others connect to this computer from ports 20 or 21, the ports 1024 through 65535 are open.

For increased protection, advanced firewall features are available through easy-to-configure checkboxes. Stealth mode hides the Mac on the Internet by dropping unsolicited communication packets, making it appear as though no Mac is present. UDP packets can be blocked, restricting network traffic to TCP packets only for open ports. The firewall also supports logging, an important tool for checking on unwanted activity. The firewall also allows the user to enable a feature that allows software signed by a valid certificate authority to provide services accessed from the network.



Example Personal Firewall Interface

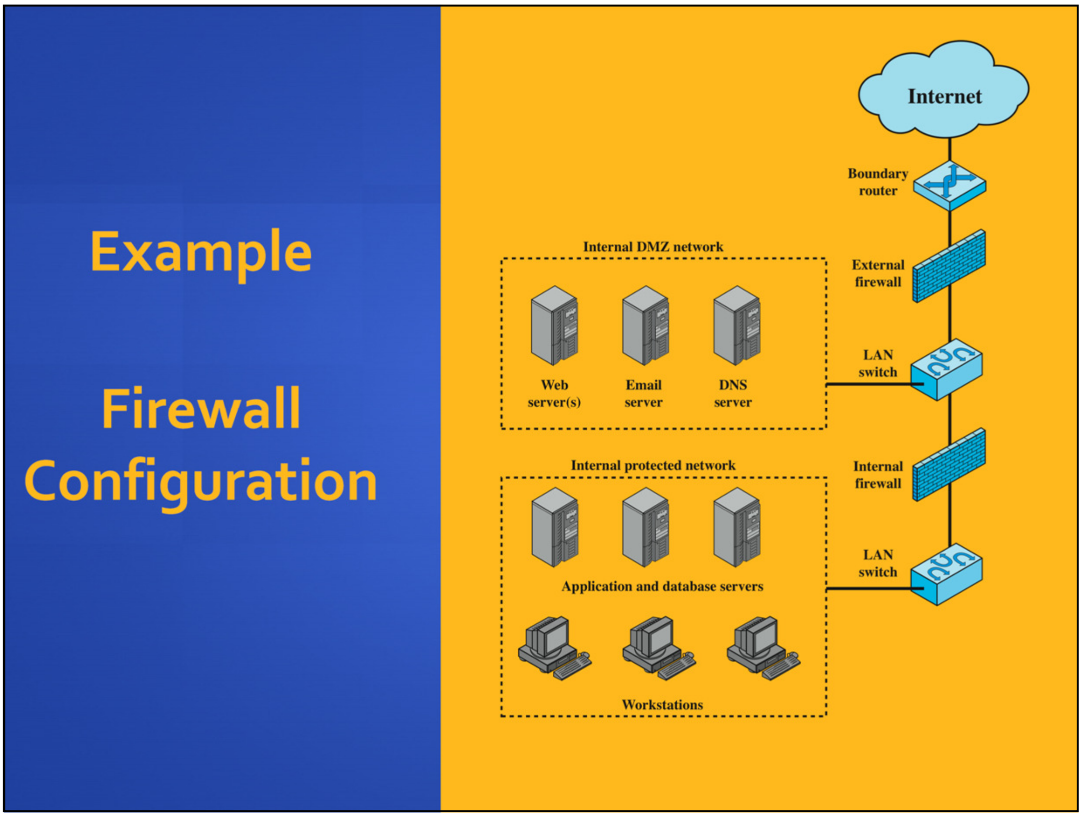


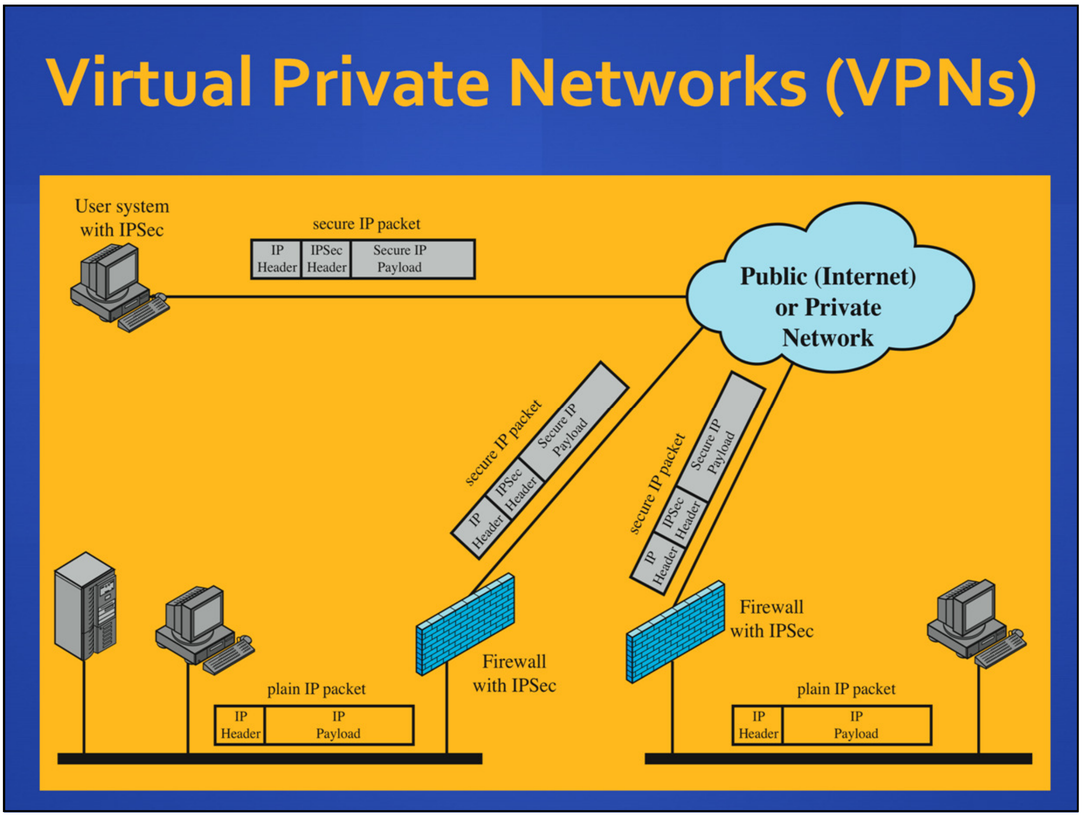
Figure 9.2 suggests the most common distinction, that between an internal and an
external firewall (see also Figure 8.5). An external firewall is placed at the edge of
a local or enterprise network, just inside the boundary router that connects to the
Internet or some wide area network (WAN). One or more internal firewalls protect
the bulk of the enterprise network. Between these two types of firewalls are one
or more networked devices in a region referred to as a DMZ (demilitarized zone)
network. Systems that are externally accessible but need some protections are usually
located on DMZ networks. Typically, the systems in the DMZ require or foster
external connectivity, such as a corporate Web site, an e-mail server, or a DNS
(domain name system) server.

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

1. The internal firewall adds more stringent filtering capability, compared to the
external firewall, in order to protect enterprise servers and workstations from external attack.

2. The internal firewall provides two-way protection with respect to the DMZ. First, the internal firewall protects the remainder of the network from attacks launched from DMZ systems. Such attacks might originate from worms, rootkits, bots, or other malware lodged in a DMZ system. Second, an internal firewall can protect the DMZ systems from attack from the internal protected network.

3. Multiple internal firewalls can be used to protect portions of the internal network from each other. Figure 8.5 (network intrusion detection system) shows a configuration in which the internal servers are protected from internal workstations and vice versa. It also illustrates the common practice of placing the DMZ on a different network interface on the external firewall from that used to access the internal networks.

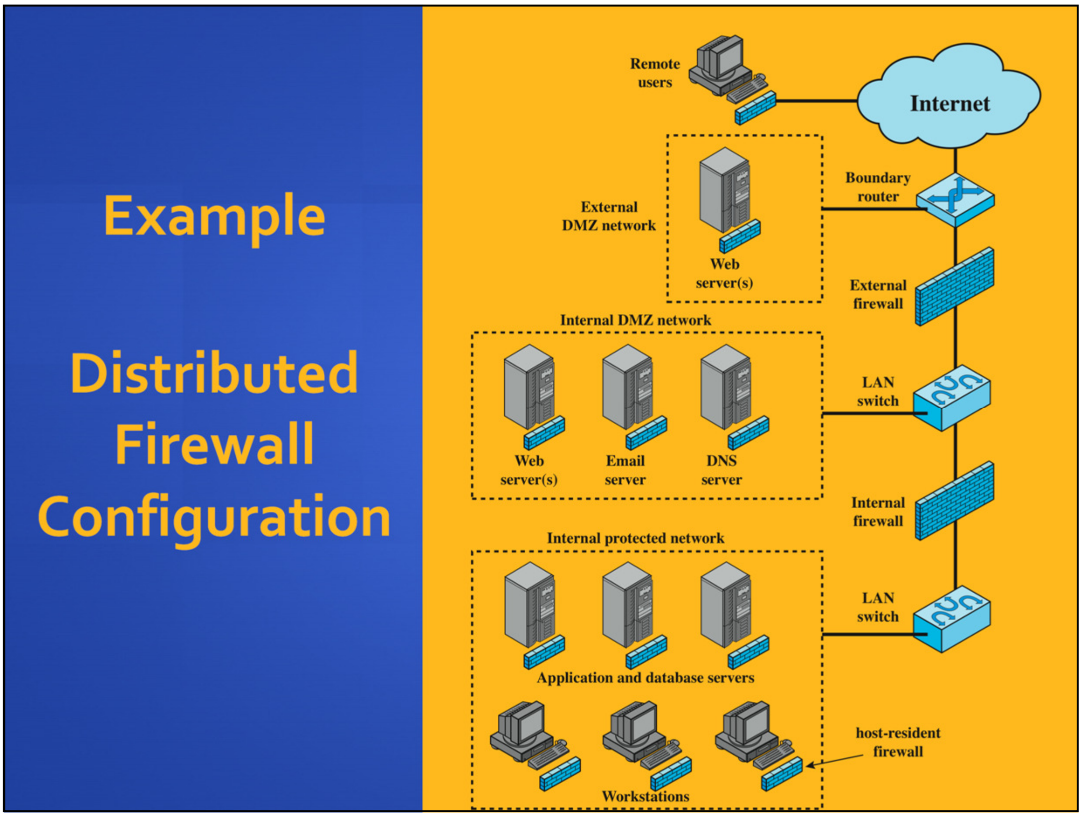


In today's distributed computing environment, the **virtual private network (VPN)** offers an attractive solution to network managers. In essence, a VPN consists of a set of computers that interconnect by means of a relatively unsecure network and that make use of encryption and special protocols to provide security. At each corporate site, workstations, servers, and databases are linked by one or more local area networks (LANs). The Internet or some other public network can be used to interconnect sites, providing a cost savings over the use of a private network and offloading the wide area network management task to the public network provider.
That same public network provides an access path for telecommuters and other mobile employees to log on to corporate systems from remote sites.

But the manager faces a fundamental requirement: security. Use of a public network exposes corporate traffic to eavesdropping and provides an entry point for unauthorized users. To counter this problem, a VPN is needed. In essence, a VPN uses encryption and authentication in the lower protocol layers to provide a

secure
connection through an otherwise insecure network, typically the Internet. VPNs are
generally cheaper than real private networks using private lines but rely on having the same encryption and authentication system at both ends. The encryption may be performed by firewall software or possibly by routers. The most common protocol
mechanism used for this purpose is at the IP level and is known as IPSec.

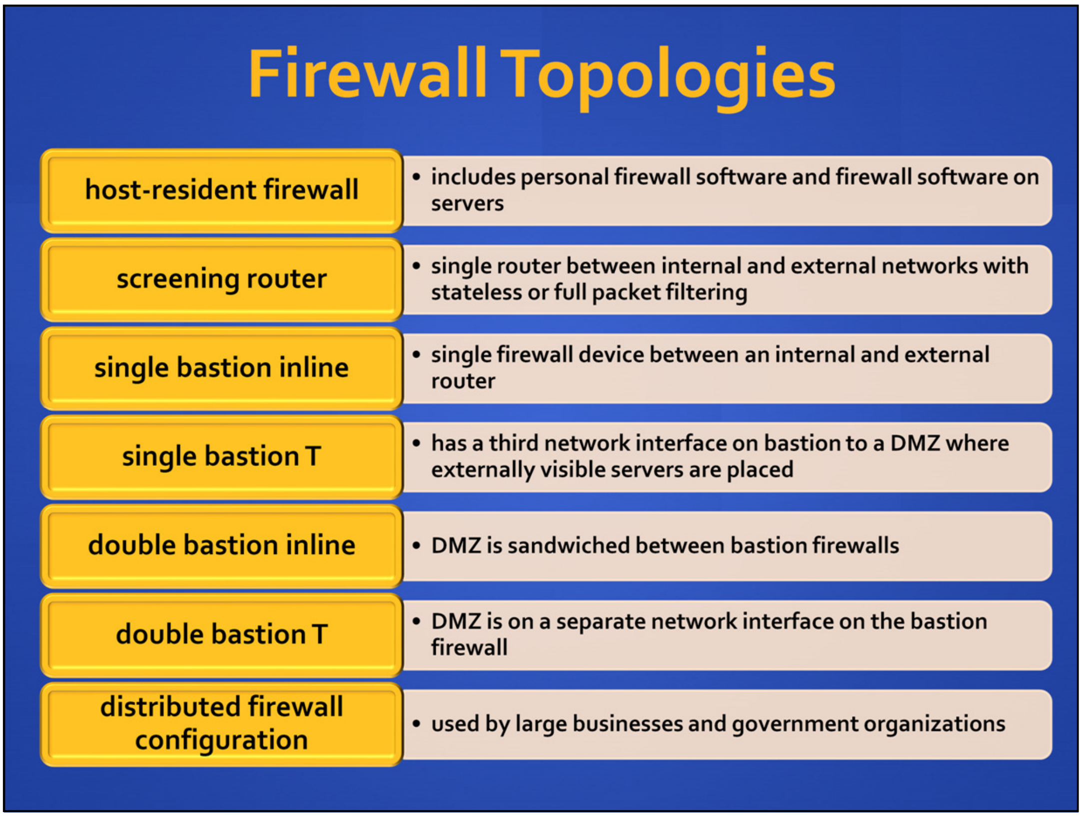
Figure 9.3 is a typical scenario of IPSec usage. An organization maintains LANs at dispersed locations. Nonsecure IP traffic is conducted on each LAN. For traffic off site, through some sort of private or public WAN, IPSec protocols are used. These protocols operate in networking devices, such as a router or firewall, that connect each LAN to the outside world. The IPSec networking device will typically encrypt and compress all traffic going into the WAN and decrypt and uncompress traffic coming from the WAN; authentication may also be provided. These operations are transparent to workstations and servers on the LAN. Secure transmission is also possible with individual users who dial into the WAN. Such user workstations must implement the IPSec protocols to provide security. They must also implement high levels of host security, as they are directly connected to



A distributed firewall configuration involves stand-alone firewall devices plus host-based firewalls working together under a central administrative control. Figure 9.4 suggests a distributed firewall configuration. Administrators can configure host resident firewalls on hundreds of servers and workstation as well as configure personal firewalls on local and remote user systems. Tools let the network administrator set policies and monitor security across the entire network. These firewalls protect against internal attacks and provide protection tailored to specific machines and applications. Stand-alone firewalls provide global protection, including internal firewalls and an external firewall, as discussed previously.

With distributed firewalls, it may make sense to establish both an internal and an external DMZ. Web servers that need less protection because they have less critical information on them could be placed in an external DMZ, outside the external firewall. What protection is needed is provided by host-based firewalls on these servers.

An important aspect of a distributed firewall configuration is security monitoring. Such monitoring typically includes log aggregation and analysis, firewall statistics, and fine-grained remote monitoring of individual hosts if needed.



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

• **Host-resident firewall: This category includes personal firewall software and** firewall software on servers. Such firewalls can be used alone or as part of an in-depth firewall deployment.

• **Screening router: A single router between internal and external networks with** stateless or full packet filtering. This arrangement is typical for small office/ home office (SOHO) applications.

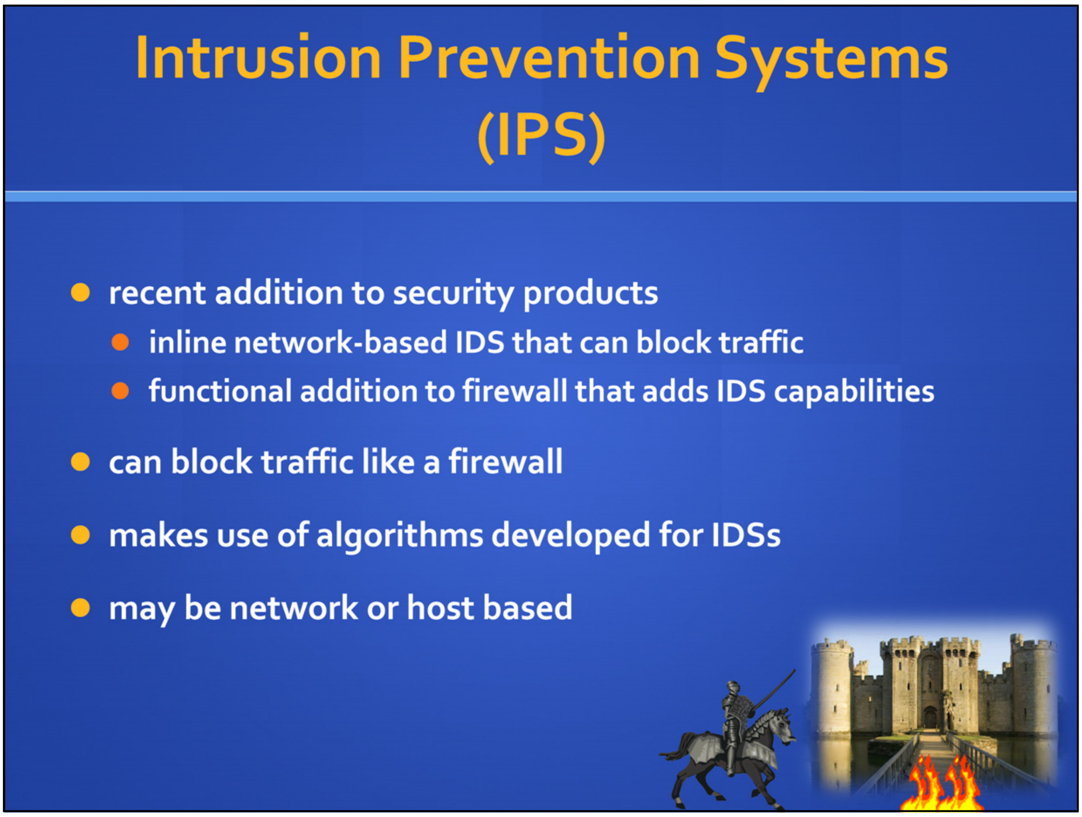
• **Single bastion inline: A single firewall device between an internal and external** router (e.g., Figure 9.1a). The firewall may implement stateful filters and/ or application proxies. This is the typical firewall appliance configuration for small to medium-sized organizations.

• **Single bastion T: Similar to single bastion inline but has a third network interface**
on bastion to a DMZ where externally visible servers are placed. Again, this is a common appliance configuration for medium to large organizations.

• **Double bastion inline: Figure 9.2 illustrates this configuration, where the** DMZ is sandwiched between bastion firewalls. This configuration is common for large businesses and government organizations.

• **Double bastion T: Figure 8.5 illustrates this configuration. The DMZ is on** a separate network interface on the bastion firewall. This configuration is also common for large businesses and government organizations and may be required. For example, this configuration is often required for Australian government use (Australian Government Information Technology Security Manual - ACSI33).

• **Distributed firewall configuration: Illustrated in Figure 9.4 . This configuration**
is used by some large businesses and government organizations.

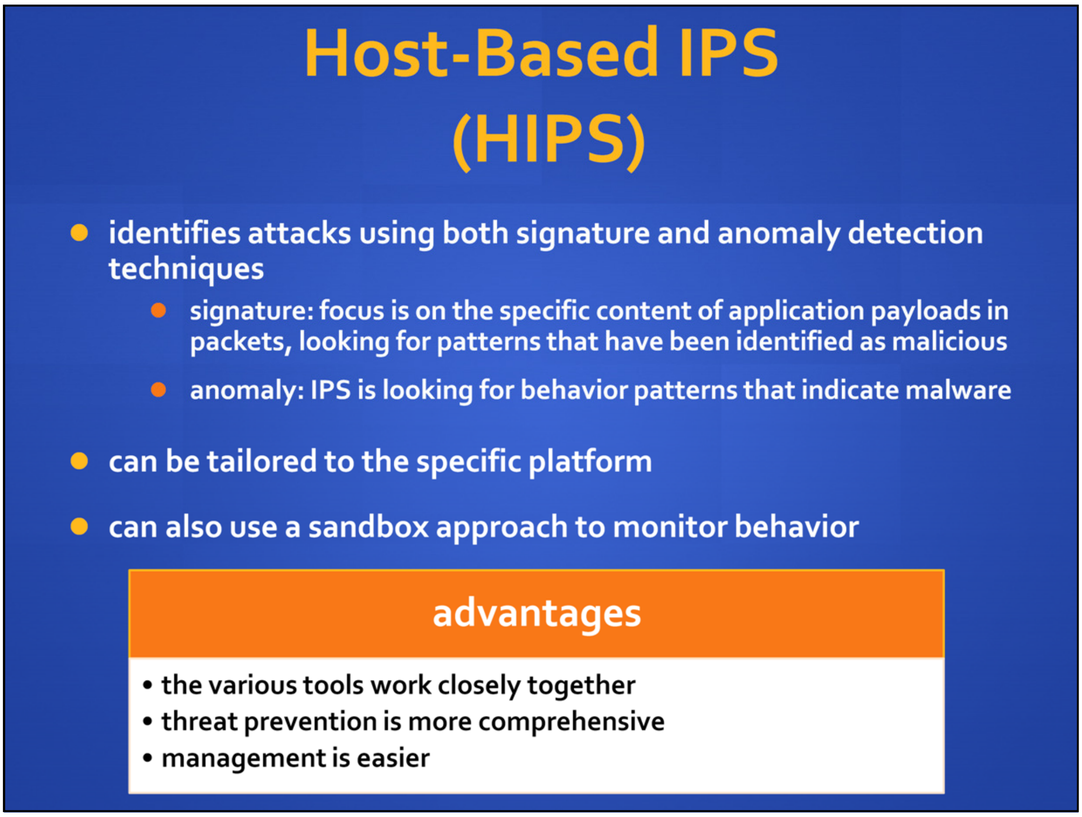


A relatively recent addition to the terminology of security products is the intrusion prevention system (IPS). There are two complementary ways of looking at an IPS:

1. An IPS is an inline network-based IDS (NIDS) that has the capability to block
traffic by discarding packets as well as simply detecting suspicious traffic. Alternatively, the IPS can monitor ports on a switch that receives all traffic and then send the appropriate commands to a router or firewall to block traffic. For host-based systems, an IPS is a host-based IDS that can discard incoming traffic.

2. An IPS is a functional addition to a firewall that adds IDS types of algorithms
to the repertoire of the firewall.

Thus, an IPS blocks traffic, as a firewall does, but makes use of the types of algorithms developed for IDSs. It is a matter of terminology whether an IPS is considered a separate, new type of product or simply another form of firewall.



As with an IDS, an IPS can be either host based or network based. A host-based IPS (HIPS) makes use of both signature and anomaly detection techniques to identify attacks. In the former case, the focus is on the specific content of application payloads in packets, looking for patterns that have been identified as malicious. In the case of anomaly detection, the IPS is looking for behavior patterns that indicate malware. Examples of the types of malicious behavior addressed by a HIPS include the following:

• **Modification of system resources: Rootkits, Trojan horses, and backdoors** operate by changing system resources, such as libraries, directories, registry settings, and user accounts.

• **Privilege-escalation exploits: These attacks attempt to give ordinary users** root access.

• **Buffer-overflow exploits: These attacks are described in Chapter 10 .**

• **Access to e-mail contact list: Many worms spread by mailing a copy of themselves**
to addresses in the local system's e-mail address book.

• **Directory traversal: A directory traversal vulnerability in a Web server allows**
the hacker to access files outside the range of what a server application user would normally need to access.

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

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

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

• **System calls: The kernel controls access to system resources such as memory,**
I/O devices, and processor. To use these resources, user applications invoke system calls to the kernel. Any exploit code will execute at least one system call. The HIPS can be configured to examine each system call for malicious characteristics.

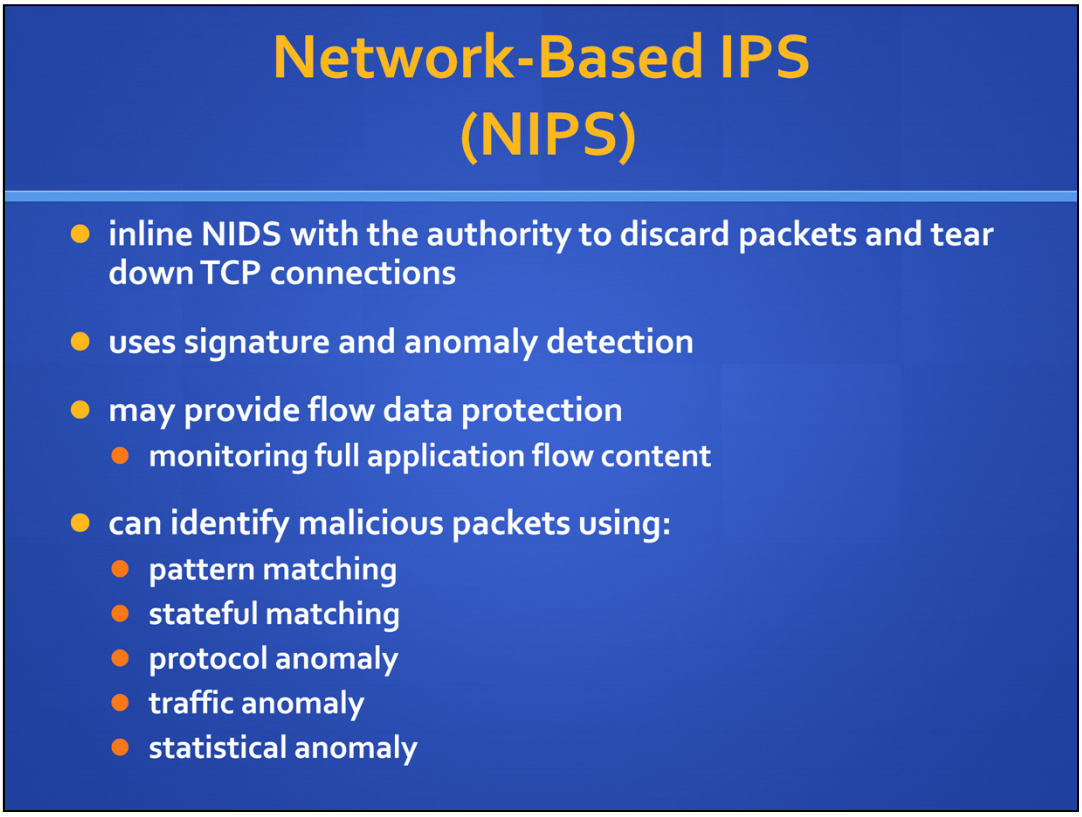
• **File system access: The HIPS can ensure that file access system calls are not** malicious and meet established policy.

System registry settings: The registry maintains persistent configuration information about programs and is often maliciously modified to extend the life of an exploit. The HIPS can ensure that the system registry maintains its integrity.

• **Host input/output: I/O communications, whether local or network based, can** propagate exploit code and malware. The HIPS can examine and enforce proper client interaction with the network and its interaction with other devices.

Many industry observers see the enterprise endpoint, including desktop and laptop systems, as now the main target for hackers and criminals, more so than network devices [ROBB06b]. Thus, security vendors are focusing more on developing endpoint security products. Traditionally, endpoint security has been provided by a collection of distinct products, such as antivirus, antispyware, antispam, and personal firewalls. The HIPS approach is an effort to provide an integrated, single-product suite of functions. The advantages of the integrated HIPS approach are that the various tools work closely together, threat prevention is more comprehensive, and management is easier.

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



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

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

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

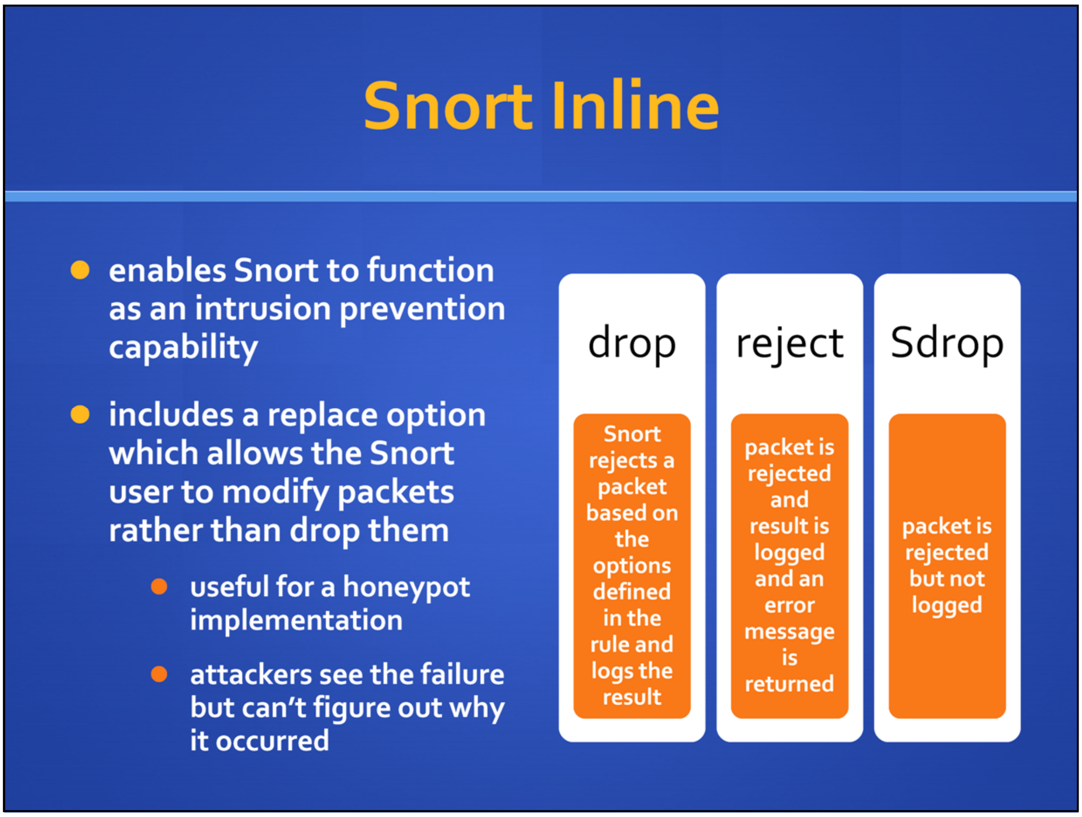
• **Pattern matching: Scans incoming packets for specific byte sequences (the** signature) stored in a database of known attacks

• **Stateful matching: Scans for attack signatures in the context of a traffic stream**
rather than individual packets

• **Protocol anomaly: Looks for deviation from standards set forth in RFCs**

Traffic anomaly: Watches for unusual traffic activities, such as a flood of UDP
packets or a new service appearing on the network

• **Statistical anomaly: Develops baselines of normal traffic activity and throughput,**
and alerts on deviations from those baselines



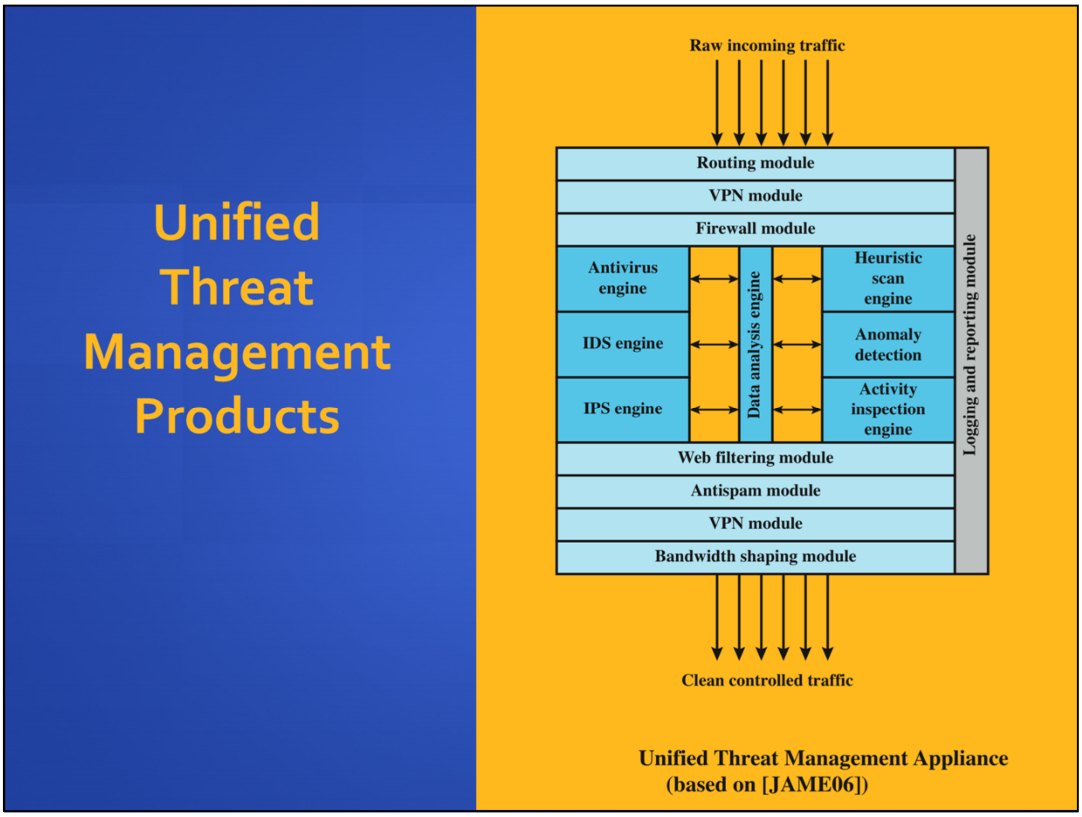
We introduced Snort in Chapter 8 as a lightweight intrusion detection capability. A modified version of Snort, known as Snort Inline, enables Snort to function as an intrusion prevention capability. Snort Inline adds three new rule types and provide intrusion prevention features:

• **Drop: Snort rejects a packet based on the options defined in the rule and logs** the result.

• **Reject: Snort rejects a packet and logs the result. In addition, an error message** is returned. In the case of TCP, this is a TCP reset message, which resets the TCP connection. In the case of UDP, an ICMP port unreachable message is sent to the originator of the UDP packet.

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

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



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

One approach to reducing the administrative and performance burden is to
replace all inline network products (firewall, IPS, IDS, VPN, antispam, antisypware,
and so on) with a single device that integrates a variety of approaches to dealing
with network-based attacks. The market analyst firm IDC refers to such a device as
a unified threat management (UTM) system and defines UTM as follows: "Products

that include multiple security features integrated into one box. To be included in this category, [an appliance] must be able to perform network firewalling, network intrusion detection and prevention and gateway anti-virus. All of the capabilities in the appliance need not be used concurrently, but the functions must exist inherently
in the appliance."

A significant issue with a UTM device is performance, both throughput and latency. [MESS06] reports that typical throughput losses for current commercial devices is 50% Thus, customers are advised to get very high-performance, high-throughput devices to minimize the apparent performance degradation.

Figure 9.5 is a typical UTM appliance architecture. The following functions are noteworthy:

1. Inbound traffic is decrypted if necessary before its initial inspection. If the device functions as a VPN boundary node, then IPSec decryption would take place here.

An initial firewall module filters traffic, discarding packets that violate rules and/or passing packets that conform to rules set in the firewall policy.

3. Beyond this point, a number of modules process individual packets and flows
of packets at various protocols levels. In this particular configuration, a data analysis engine is responsible for keeping track of packet flows and coordinating the work of antivirus, IDS, and IPS engines.

4. The data analysis engine also reassembles multipacket payloads for content
analysis by the antivirus engine and the Web filtering and antispam modules.

5. Some incoming traffic may need to be reencrypted to maintain security of the
flow within the enterprise network.

6. All detected threats are reported to the logging and reporting module, which

is used to issue alerts for specified conditions and for forensic analysis.

7. The bandwidth-shaping module can use various priority and quality-of-service

(QoS) algorithms to optimize performance.

Table 9.3

Sidewinder G2 Security Appliance Attack Protections Summary - Transport Level Examples

Attacks and Internet Threats		Protections	
TCP			
•Invalid port numbers •Invalid sequence •numbers •SYN floods •XMAS tree attacks •Invalid CRC values •Zero length •Random data as TCP •header	•TCP hijack attempts •TCP spoofing attacks •Small PMTU attacks •SYN attack •Script Kiddie attacks •Packet crafting: different TCP options set	•Enforce correct TCP flags •Enforce TCP header length •Ensures a proper 3-way handshake •Closes TCP session correctly •2 sessions, one on the inside and one on the outside •Enforce correct TCP flag usage •Manages TCP session timeouts •Blocks SYN attacks	•Reassembly of packets ensuring correctness •Properly handles TCP timeouts and retransmits timers •All TCP proxies are protected •Traffic Control through access lists •Drop TCP packets on ports not open •Proxies block packet crafting
UDP			
•Invalid UDP packets •Random UDP data to bypass rules	•Connection prediction •UDP port scanning	•Verify correct UDP packet •Drop UDP packets on ports not open	

As an example of the scope of a UTM appliance, Tables 9.3 and 9.4 . lists some of the attacks that the UTM device marketed by Secure Computing is designed to counter.

Table 9.4

Sidewinder G2 Security Appliance Attack Protections Summary - Application Level Examples (page 1 of 2)

Attacks and Internet Threats	Protections
DNS	
Incorrect NXDOMAIN responses from AAAA queries could cause denial-of-service conditions.	•Does not allow negative caching •Prevents DNS Cache Poisoning
ISC BIND 9 before 9.2.1 allows remote attackers to cause a denial of service (shutdown) via a malformed DNS packet that triggers an error condition that is not properly handled when the rdataset parameter to the dns_message_findtype() function in message.c is not NULL.	•Sidewinder G2 prevents malicious use of improperly formed DNS messages to affect firewall operations. •Prevents DNS query attacks •Prevents DNS answer attacks
DNS information prevention and other DNS abuses.	•Prevent zone transfers and queries •True split DNS protect by Type Enforcement technology to allow public and private DNS zones. •Ability to turn off recursion
FTP	
•FTP bounce attack •PASS attack •FTP Port injection attacks •TCP segmentation attack	•Sidewinder G2 has the ability to filter FTP commands to prevent these attacks. •True network separation prevents segmentation attacks.
SQL	
SQL Net man in the middle attacks	•Smart proxy protected by Type Enforcement Technology •Hide Internal DB through nontransparent connections
Real-Time Streaming Protocol (RTSP)	
•Buffer overflow •Denial of service	•Smart proxy protected by Type Enforcement technology •Protocol validation •Denies multicast traffic •Checks setup and teardown methods •Verifies PNG and RTSP protocol, discards all others •Auxiliary port monitoring
SNMP	
•SNMP flood attacks •Default community attack •Brute force attack •SNMP put attack	•Filter SNMP version traffic 1, 2c •Filter Read, Write, and Notify messages •Filter OIDs •Filter PDU (Protocol Data Unit)

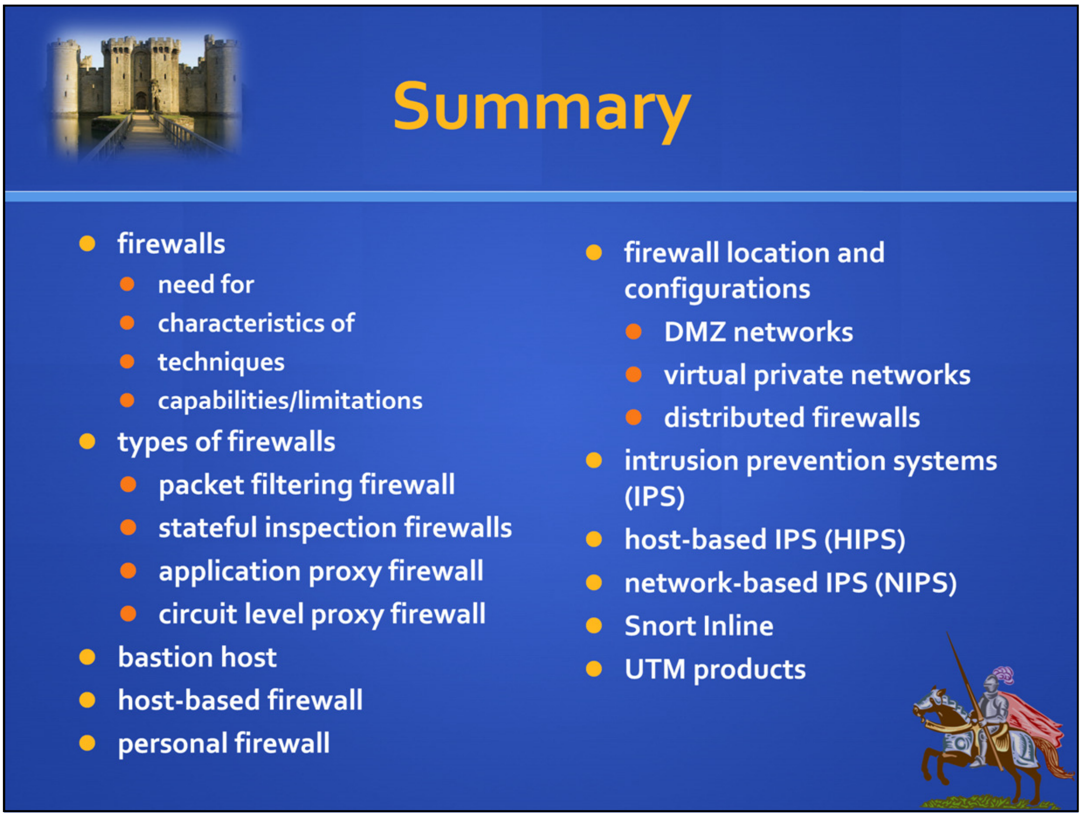
Table 9.4 page 1

Table 9.4

Sidewinder G2 Security Appliance Attack Protections Summary - Application Level Examples (page 2 of 2)

SSH			
•Challenge-Response buffer overflows •SSHD allows users to override "Allowed Authentications" •OpenSSH buffer_append_space buffer overflow •OpenSSH/PAM challenge Response buffer overflow •OpenSSH channel code offer-by-one		Sidewinder G2 v6.x's embedded Type Enforcement technology strictly limits the capabilities of Secure Computing's modified versions of the OpenSSH daemon code.	
SMTP			
•Sendmail buffer overflows •Sendmail denial of service attacks •Remote buffer overflow in sendmail	•Sendmail address parsing buffer overflow •SMTP protocol anomalies	•Split Sendmail architecture protected by Type Enforcement technology •Sendmail customized for controls	•Prevents buffer overflows through Type Enforcement technology •Sendmail checks SMTP protocol anomalies
•SMTP worm attacks •SMTP mail flooding •Relay attacks •Viruses, Trojans, worms	•E-mail Addressing spoofing •MIME attacks •Phishing e-mails	•Protocol validation •Anti-spam filter •Mail filters – size, keyword •Signature antivirus	•Anti-relay •MIME/Antivirus filter •Firewall antivirus •Anti-phishing through virus scanning
Spyware Applications			
•Adware used for collecting information for marketing purposes •Stalking horses •Trojan horses	•Malware •Backdoor Santas	•SmartFilter® URL filtering capability built in with Sidewinder G2 can be configured to filter Spyware URLs, preventing downloads.	

Table 9.4 page 2



Chapter 9 summary.