

Chapter 76: Firewalls

Errin W. Fulp Wake Forest University, Winston-Salem, NC, United States

Abstract

The purpose of the firewall and its location is to have network connections traverse the firewall, which can then stop any unauthorized packets. A simple firewall will filter packets based on IP addresses and ports. A useful analogy is filtering your postal mail based only on the information on the envelope. You typically accept any letter addressed to you and return any letter addressed to someone else. This act of filtering is essentially the same for firewalls. This chapter refers to the secure network as the internal network; the insecure network is the external network. The remainder of this chapter provides an overview of firewall policies, designs, features, and configurations. Of course, technology is always changing, and network firewalls are no exception. However, the intent of this chapter is to describe aspects of network firewalls that tend to endure over time.

Keywords

Default accept policy; Firewall security; Firewalls; Network firewalls; Packet filter and stateful packet firewalls; Policy optimization; Policy reordering; Rule-match firewall policies; Rules

1. Introduction

Providing a secure computing environment continues to be an important and challenging goal of any computer administrator. The difficulty is in part due to the increasing interconnectivity of computers via networks, which includes the Internet. Such interconnectivity brings great economies of scale in terms of resources, services, and knowledge, but it has

also introduced new security risks. For example, interconnectivity gives illegitimate users much easier access to vital data and resources from almost anywhere in the world.

In a secure environment it is important to maintain the privacy, integrity, and availability of data and resources. *Privacy* refers to limiting information access and disclosures to authorized users and preventing access by or disclosure to illegitimate users. In the United States, a range of state and federal laws—for example, FERPA, FSMA, and HIPAA—define the legal terms of privacy. *Integrity* is the trustworthiness of information. It includes the idea of *data integrity*, which means data has not been changed inappropriately. It can also include *source integrity*, which means the source of the data is who it claims to be. *Availability* is the accessibility of resources. Of course these security definitions can also form the basis of *reputation*, which is vital to businesses.

2. Network Firewalls

Network firewalls (see An Agenda For Action For Network Firewalls section) are a vital component for maintaining a secure environment and are often the first line of defense against attack. Simply stated, a firewall is responsible for controlling access among devices, such as computers, networks, and servers. Therefore the most common deployment is between a secure and an insecure network (for example, between the computers you control and the Internet), as shown in [Fig. 76.1](#).

However, in response to the richer services provided over modern networks (such as multimedia and encrypted connections), the role of the firewall has grown over time. Advanced firewalls may also perform NAT that allows multiple computers to share a limited number of network addresses (explained later in this chapter). Firewalls may provide service differentiation, giving certain traffic priority to ensure that data is received in a timely fashion. Voice over IP (VoIP) is one type of application that needs differentiation to ensure proper operation. This idea is discussed several times in this chapter, since the use of multimedia services will only continue to increase. Assuming that email and VoIP packets ar-

rive at the firewall at the same time, VoIP packets should be processed first because the application is more susceptible to delays.

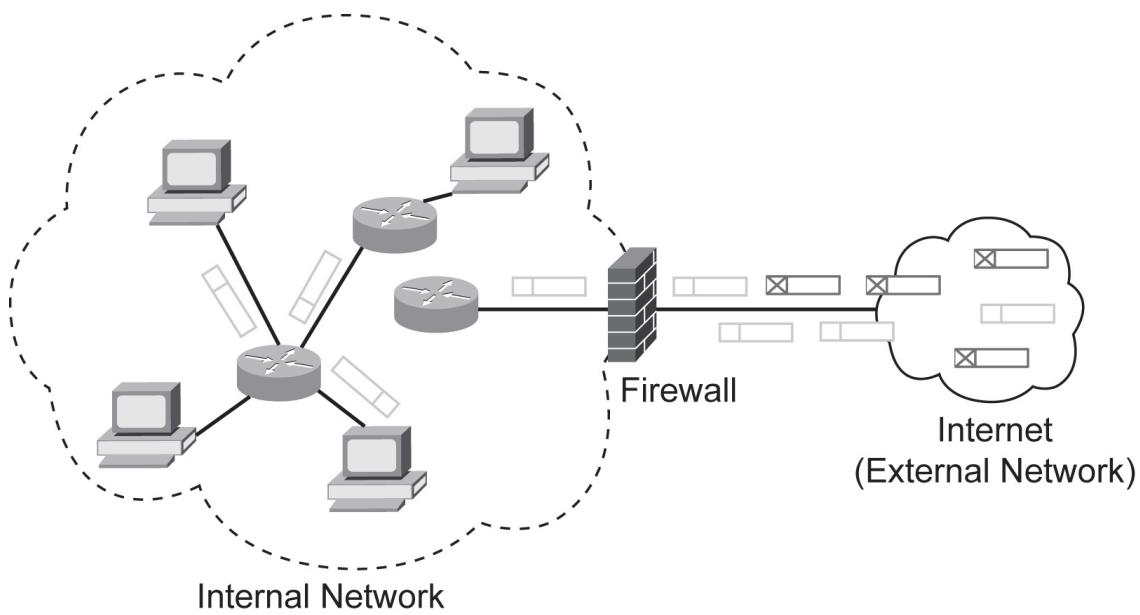


FIGURE 76.1 Example network consisting of an internal network (which is to be secured) and an external network (not trusted). The firewall controls access between these two networks, allowing and denying packets according to a security policy.

Firewalls may also inspect the contents (the data) of packets. This can be done to filter other packets (learn new connections), block packets that contain offensive information, and/or block intrusion attempts. Using the mail analogy again, in this case you open letters and determine what to accept based on what is inside. For example, you

An Agenda for Action for Network Firewalls

The following checklist lists the major tasks for network firewalls (check all tasks completed):

- 1. The use of network address translation (NAT) should be considered a form of routing, not a type of firewall.
- 2. Organizations should only permit outbound traffic that uses the source IP addresses in use by the organization.
- 3. Compliance checking is only useful in a firewall when it can block communication that can be harmful to protected systems.

- _____ 4. When choosing the type of firewall to deploy, it is important to decide whether the firewall needs to act as an application proxy.
 - _____ 5. Management of personal firewalls should be centralized to help efficiently create, distribute, and enforce policies for all users and groups.
 - _____ 6. In general, a firewall should fit into a current network's layout. However, an organization might change its network architecture at the same time as it deploys a firewall as part of an overall security upgrade.
 - _____ 7. Different common network architectures lead to very different choices for where to place a firewall, so an organization should assess which architecture works best for its security goals.
 - _____ 8. If an edge firewall has a demilitarized zone (DMZ), consider which outward-facing services should be run from the DMZ and which should remain on the inside network.
 - _____ 9. Do not rely on NATs to provide the benefits of firewalls.
 - _____ 10. In some environments, putting one firewall behind another may lead to a desired security goal, but in general such multiple layers of firewalls can be troublesome.
 - _____ 11. An organization's firewall policy should be based on a comprehensive risk analysis.
 - _____ 12. Firewall policies should be based on blocking all inbound and outbound traffic, with exceptions made for desired traffic.
 - _____ 13. Policies should take into account the source and destination of the traffic in addition to the content.
 - _____ 14. Many types of IPv4 traffic, such as that with invalid or private addresses, should be blocked by default.
 - _____ 15. Organizations should have policies for handling incoming and outgoing IPv6 traffic.
 - _____ 16. An organization should determine which applications may send traffic into or out of its network and make firewall policies to block traffic for other applications.
-

unfortunately have to accept bills, but you can deny credit card solicitations.

3. Firewall Security Policies

When a packet arrives at a firewall, a security policy is applied to determine the appropriate action. Actions include accepting the packet, which means the packet is allowed to travel to the intended destination. A packet can be denied, which means the packet is not permitted to travel to the intended destination (it is dropped or possibly is bounced back). The firewall may also log information about the packet, which is important to maintain certain services.

It is easy to consider a firewall policy as an ordered list of rules, as shown in **Table 76.1**. Each firewall rule consists of a set of tuples and an action. Each tuple corresponds to a field in the packet header, and there are five such fields for an Internet packet: Protocol, source address, source port, destination address, and destination port.

The firewall rule tuples can be fully specified or contain wildcards (*) in standard prefix format. However, each tuple represents a finite set of values; therefore, the set of all possible packets is also finite. (A more concise mathematical model will be introduced later in the chapter.) It is possible to consider the packet header consisting of tuples, but each tuple must be fully specified.

As packets pass through a firewall, their header information is sequentially compared to the fields of a rule. If a packet's header information is a subset of a rule, it is said to be a match, and the associated action, to accept or reject, is performed. Otherwise, the packet is compared to the next sequential rule. This is considered a *first-match policy* since the action associated with the first rule that is matched is performed. Other matching strategies are discussed at the end of this section.

Table 76.1

A Security Policy Consisting of Six Rules, Each of Which Has Five Parts (Tuples)

No.	Protocol	Source		Destination		Action
		IP	Port	IP	Port	
1	UDP	190.1.1.*	*	*	80	Deny
2	TCP	180.*	*	180.*	90	Accept
3	UDP	210.1.*	*	*	90	Accept
4	TCP	210.*	*	220.*	80	Accept
5	UDP	190.*	*	*	80	Accept
6	*	*	*	*	*	Deny

For example, assume that a packet has the following values in the header: The protocol is Transmission Control Protocol (TCP), source IP is 210.1.1.1, source port is 3080, destination IP is 220.2.33.8, and destination port is 80. When the packet arrives it is compared to the first rule, which results in no match since the rule is for User Datagram Protocol (UDP) packets. The firewall then compares the packet second rule, which results in no match since the source IP is different. The packet does not match the third rule, but it does match the fourth rule. The rule action is performed and so the packet is allowed to pass the firewall.

A default rule, or catch-all, is often placed at the end of a policy with action reject. The addition of a default rule makes a policy comprehensive, indicating that every packet will match at least one rule. In the event that a packet matches multiple rules, the action of the first matching rule is taken. Therefore the order of rules is very important.

If a default rule (a rule that matches all possible packets) is placed at the beginning of a first-match policy, no other rule will match. This situation is an anomaly referred to as *shadowing*. We will talk more about policy anomalies later in this chapter. Policies that employ this form of short-circuit evaluation are called *first-match policies* and account for the majority of firewall implementations.

Rule-Match Policies

Multiple rules of a single firewall policy may match a packet—for example, a packet could match rules 1, 5, and 6 of the policy in [Table 76.1](#). Given multiple possible matches, the rule-match policy describes the rule the firewall will apply to the packet. The previous section described the most popular match policy, first match, which will apply the first rule that is a match.

Other match policies are possible, including best match and last match. For best-match policies, the packet is compared against every rule to determine which rule most closely matches every tuple of the packet. Note that the relative order of the rules in the policy does not impact determining the best-match result; therefore shadowing is not an issue. It is interesting to note that best match is the default criterion for IP routing, which is not surprising since firewalls and routers do perform similar tasks. If a packet matches multiple rules with a last-match criterion, the action of the last rule matched is performed. Note that rule order is important for a last-match policy.

4. A Simple Mathematical Model for Policies, Rules, and Packets

At this point it is perhaps useful to describe firewall policies, firewall rules, and network packets using set theory.¹ The previous section defined the parts and fields of rules and packets as *tuples*. A tuple can be modeled as a set. For example, assume the tuple for IP source addresses is 198.188.150.*. Then this tuple represents the set of 256 addresses that range from 198.188.150.0 to 198.188.150.255. Each tuple of a packet con-

sists of a single value, which is expected, since a packet only has one source and one destination.

The tuples (which are sets) that form a rule collectively define a set of packets that match. For example, consider the following rule:

1. • Proto = TCP, SIP = 190.150.140.38, SP = 188,
2. • DIP = 190.180.39.* DP = 80, action = accept

This rule defines a set of 256 unique TCP packet headers with source address 190.150.140.38 and source port 188 destined for any of the 256 computers with destination port 80 and destination IP address 190.180.39.0 through 190.180.39.255, perhaps a Web server farm. Therefore the rule describes a set of 256 packets that will be accepted. If the source port was defined as *, the rule would describe a set of 16,777,216 different packet headers.

$$2^{16} \times 2^8 = 65,536 \times 256 = 16,777,216$$

Using set theory also provides a simple definition of a match. A match occurs when every tuple of a packet is a proper subset of the corresponding rule. In this chapter a proper set can be thought of as one set completely contained within another. For example, every tuple in the following packet is a proper subset of the preceding rule; therefore it is considered a match:

1. • Proto = TCP, SIP = 190.150.140.38, SP = 188,
2. • DIP = 190.180.39.188, DP = 80

A set model can also be used to describe a firewall policy. The list of rules in a firewall policy collectively describes a set of packets. There are three distinct (nonoverlapping) sets of possible packets. The first set, $A(R)$, describes packets that will be accepted by the policy R . The second set, $D(R)$, defines the set of packets that will be dropped by the policy. The last set, $U(R)$, is the set of packets that do not match any rule in the policy. Since the sets do not overlap, the intersection of $A(R)$, $D(R)$, and $U(R)$ should be the empty set.

Using set theory we can also define the set P that describes all possible packet headers, of which there are approximately 7.7×10^{25} possible packet headers. A packet is a single element in this large set.

Using accept, drop, nonmatch, and possible packet sets, we can describe useful attributes of a firewall policy. A firewall policy R is considered comprehensive if any packet from P will match at least one rule. In other words, the union of $A(R)$ and $D(R)$ equals P [therefore $A(R)D(R) = P$], or $U(R)$ is the empty set [therefore $U(R) = \emptyset$]. Of course, it is better if a policy is comprehensive, and generally the last rule (catch-all) makes this true.

Finally, these mathematical models also allow the comparison of policies, the most important reason for introducing a somewhat painful section. Assume two firewall policies R and S exist. We can say the two policies are equivalent if the accept, drop, and nonmatch sets are the same. This does not imply that the two policies have the same rules, just that given a packet, both policies will have the same action. This is an important property that will be mentioned again and again in this chapter.

5. First-Match Firewall Policy Anomalies

As described in the previous sections, for most firewalls the first rule that matches a packet is typically applied. Given this match policy, more specific rules (those that match few packets) typically appear near the beginning of the policy, whereas more general rules are located at the end. Using the set theory model, the number of elements in the rules sets increases as you move toward the last rule.

Unfortunately, it is easy to introduce anomalies when developing and managing a firewall policy. This is especially true as the policy grows in size (number of rules) and complexity. An anomaly is an unintended consequence of adding rules in a certain order.

A simple and very common anomaly is rule shadowing. Shadowing occurs when an earlier rule ri matches every packet that another lower rule rj matches, where i and j are rule numbers. Assume rules are numbered

sequentially starting at the first rule and $i < j$. Using the mathematical model, shadowing occurs when every tuple in rj is a proper subset of ri .

For example, shadowing occurs between the following two rules:

1. • Proto = TCP, SIP = 190.150.140.38, SP = 188,
2. • DIP = 190.180.39.* DP = 80, action = accept
3. • Proto = TCP, SIP = 190.150.140.38, SP = 188, DIP = 190.180.39.180 DP = 80, action = drop

What is the problem? Nothing, if the two rules have the same action (there is a performance issue described in the next section). However, if the rules have different actions, there is a potential issue. In the preceding example the second rule is never matched; therefore the packet (Proto = TCP, SIP = 190.150.140.38, SP = 188, DIP = 190.180.39.180 DP = 80) will always be accepted. Was this the intent? If so, the second rule should be removed.

Another policy anomaly is half shadowing, where only a portion of the packets of a later rule matches an earlier rule (although not necessarily half of the packets in the set). For example, consider the following two rules:

1. • Proto = TCP, SIP = 190.150.140.38, SP = 188,
2. • DIP = 190.180.39.* DP = 80, action = accept
3. • Proto = TCP, SIP = 190.150.140.38, SP = *,
4. • DIP = 190.180.39.180 DP = 80, action = drop

In this example, the second rule is partially shadowed by the first rule. By itself, the second rule will drop any TCP packet arriving from the address 190.150.140.38 and destined for the Web server (because of destination port 80) 190.180.39.180. When the first rule is added, a packet from the address 190.150.140.38 and port 188 will be accepted. Was this the intent? Only the firewall administrator would know. Regardless, it is difficult to detect.

Other firewall policy anomalies are possible. Unfortunately, detecting these problems is not easy, since the anomaly may be introduced on pur-

pose (then technically it is not an anomaly). This has created a new area of research, and some software packages are available to help find problems. However, only the administrator can ultimately determine whether the rule ordering is correct. Note that best-match policies do not have these issues, and this reason is often used to promote their use. However, best-match policies are typically considered difficult for the administrator to manage.

6. Policy Optimization

Given that a network firewall will inspect all packets transmitted between multiple networks, these devices need to determine the appropriate match with minimal delay. Often the number of firewall rules in a policy will impact the firewall performance. Given that every rule requires some processing time, more rules will require more time, on average. There are a few ways to improve firewall performance with regard to the security policy. Note that this section is more applicable to software-based than hardware-based firewalls.

Policy Reordering

Given a security policy, it may be possible to reorder the rules such that more popular rules appear earlier.² *More popular* refers to how often the rule is a match. For example, over time it is possible to determine how many times a rule is matched. Dividing this number by the total number of packets matched for the entire policy yields the probability that this rule is considered the first match.

If the match policy is first matched, then placing more popular rules earlier in the policy will reduce the average number of rule comparisons. The average number of rule comparisons performed, $E[n]$, is given by the following equation:

$$E[n] = \sum_{i=1}^n i \times p_i$$

where n is the number of rules in the policy and p_i is the probability that rule i is the first match. Although reordering is advantageous, it must be done so that the policy's integrity is maintained.

Policy integrity refers to the policy intent, so the policy will accept and deny the same packets before and after the reorganization of rules. For example, rule 6 in **Table 76.1** may be the most popular rule (the default deny), but placing it at the beginning of the policy does not maintain integrity. However, if rule two is more popular than rule one, it could be placed at the beginning of the policy and integrity will be maintained. Therefore the order between certain rules must be maintained.

This can be described mathematically using the models introduced in the earlier section. Assume a firewall policy R exists. After reordering the rules, let us call the firewall policy S . If $A(R) = A(S)$ and $D(R) = D(S)$, then the policies R and S are equivalent and integrity is maintained. As a result S can be used in place of R in the firewall, which should improve performance.

Although a simple concept, reordering rules to maintain integrity is provably difficult for large policies.^{3.4} Fortunately, commercial software packages are now available to optimize rules to improve performance.

Combining Rules

Another method for improving firewall performance is removing unnecessary rules. This can be accomplished by first removing redundant rules (rules that are shadowed with the same action). For example, the second rule here is unnecessary:

1. • Proto = TCP, SIP = 190.150.140.38, SP = 188,
2. • DIP = 190.180.39.* DP = 80, action = drop
3. • Proto = TCP, SIP = 190.150.140.38, SP = 188,
4. • DIP = 190.180.39.180 DP = 80, action = drop

This is because the first rule matches any packet the second rule does, and the first rule has the same action (different actions would be an anomaly, as described in the earlier sections).

Another example occurs when two nonshadowing rules can be combined into a single rule. Consider the following two rules:

1. • Proto = TCP, SIP = 190.150.140.38, SP = 188,
2. • DIP = 190.180.39.* DP = 80, action = accept
3. • Proto = UDP, SIP = 190.150.140.38, SP = 188,
4. • DIP = 190.180.39.* DP = 80, action = accept

These two rules can be combined into the following rule, which substitutes the wildcard for the protocol field:

1. • Proto = *, SIP = 190.150.140.38, SP = 188,
2. • DIP = 190.180.39.* DP = 80, action = accept

Combining rules to form a smaller policy is better in terms of performance as well as management in most cases, since fewer rules should be easier for the administrator to understand. Finding such combinations takes practice; fortunately, there are some software packages available to help.

Default Accept or Deny?

It may be worth a few lines to discuss whether a default accept policy provides better performance than a default deny. This debate occurs from time to time; generally speaking, the question is better answered with regard to management of the policy and security. Is it easier to define the appropriate policy in terms of what is denied or what should be accepted?

Assuming that the administrator defines one (accepted or denied), the default behavior becomes the other. A “define what is accepted and default deny” is the most common. It can be considered pessimistic, since it assumes that if you are not certain about a packet, then drop it.

7. Firewall Types

Firewalls can be categorized into three general classes: packet filters, stateful firewalls, and application layer firewalls.⁵ Each type provides a certain type of security and is best described within the context of a net-

work layer model—for example, the Open Systems Interconnect (OSI) or TCP/IP model, as shown in [Fig. 76.2](#).

Recall that the TCP/IP model consists of four basic layers: data link, networking (IP), transport (TCP and UDP), and application. Each layer is responsible for providing a certain service to the layer above it. The first layer (data link) is responsible for transmitting information across the local area network (LAN); examples include Ethernet and 802.11 networks. The network layer (routing, implemented IP) concerns routing information across interconnected LANs. The third layer (transport, implemented as TCP and UDP) concerns the end-to-end connection between communicating devices. The highest layer (application) is the application using the network.

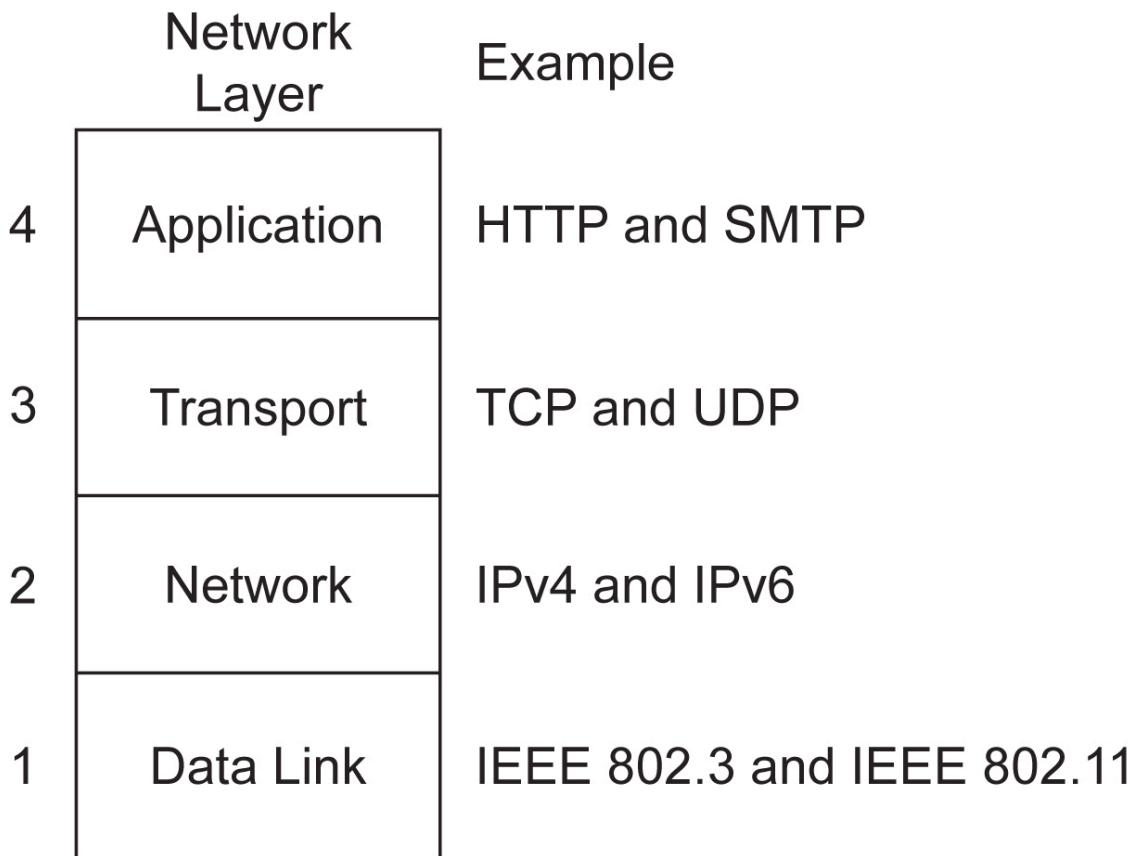


FIGURE 76.2 Layered model for computer networks and example implementations for each layer.

Packet Filter

A packet filter is the most basic type of a firewall since it only filters at the network and transport layers (layers two and three). Therefore a packet

filter's operations are similar to a network router's. The packet filter receives a packet, determines the appropriate action based on the policy, then performs the action on the packet. This will be based on the information from the network and transport layers. Therefore, a packet filter only considers the IP addresses (layer two information), the port numbers (layer one information), and the transport protocol type (layer three information). Furthermore, since all this information resides in the packet header, there is no need to inspect the packet data (payload). It is possible to filter based on the data link layer, but this chapter only considers the network layer and above. Another important note is that the packet filter has no memory (or state) regarding the packets that have arrived and departed.

Stateful Packet Firewalls

Stateful firewalls perform the same operations as packet filters but also maintain state about the packets that have arrived. Given this additional functionality, it is now possible to create firewall rules that allow network sessions (sender and receiver are allowed to communicate), which is critical given the client/server nature of most communications (that is, if you send packets, you probably expect something back). Also note the change in terminology from packet filter to firewall. Many people say that when state is added to a packet filter, it becomes a firewall. This is really a matter of opinion.

For example, assume a user located in the internal (protected) network wants to contact a Web server located in the Internet. The request would be sent from the user to the Web server, and the Web server would respond with the requested information. A packet filter would require two rules, one allowing departing packets (user to Web server) and another allowing arriving packets (Web server to user). There are several problems with this approach, since it is difficult to determine in advance what Web servers a user will connect to. Consider having to add a new rule for every Web server that is or would ever be contacted.

A stateful firewall allows connection tracking, which can allow the arriving packets associated with an accepted departing connection. Recall that

a connection or session can be considered all the packets belonging to the conversation between computers, both sender to receiver, and vice versa. Using the Web server example, a single stateful rule can be created that accepts any Web requests from the secure network and the associated return packets. A simple way to add this capability is to have the firewall add to the policy a new rule allowing return packets. Of course, this new rule would be eliminated once the connection is finished. Knowing when a connection is finished is not an easy task, and ultimately timers are involved. Regardless, stateful rules were a significant advancement for network firewalls.

Application Layer Firewalls

Application layer firewalls can filter traffic at the network, transport, and application layer. Filtering at the application layer also introduces new services, such as proxies. Application proxies are simply intermediaries for network connections. Assume that a user in the internal network wants to connect to a server in the external network. The connection of the user would terminate at the firewall; the firewall would then create a connection to the Web server. It is important to note that this occurs seamlessly to the user and server.

As a result of the proxy the firewall can potentially inspect the contents of the packets, which is similar to an intrusion detection system (IDS). This is increasingly important since a growing number of applications, as well as illegitimate users, are using nonstandard port numbers to transmit data. Application layer firewalls are also necessary if an existing connection may require the establishment of another connection—for example, the Common Object Resource Broker Architecture (CORBA).

Increasingly, firewalls and other security devices are being merged into a single device that can simplify management. For example, an intrusion prevention system (IPS) is a combination firewall and IDS. An IPS can filter packets based on the header, but it can also scan the packet contents (payload) for viruses, spam, and certain types of attacks.

8. Host and Network Firewalls

Firewalls can also be categorized based on where they are implemented or what they are intended to protect—host or network.⁶ Host firewalls typically protect only one computer. Host firewalls reside on the computer they are intended to protect and are implemented in software (this is described in the next section).

In contrast, network firewalls are typically standalone devices. Located at the gateway(s) of a network (for example, the point at which a network is connected to the Internet), a network firewall is designed to protect all the computers in the internal network. As a result, a network firewall must be able to handle high bandwidth, as fast as the incoming connection, and process packets quickly. A network firewall gives administrators a single point at which to implement and manage security, but it is also a single point of failure.

There are many different network configurations that involve firewalls. Each provides different levels of security and management complexity. These configurations are described in detail in a later section.

9. Software and Hardware Firewall Implementations

As described in the previous sections, a firewall applies a policy to an arriving packet to determine the appropriate match. The policy is an ordered list of rules, and typically the first rule that matches the packet is performed. This operation can be performed primarily in either software or hardware. Performance is the principal reason to choose one implementation.

Software firewalls are application software that can execute on commercial hardware. Most operating systems provide a firewall to protect the host computer (often called a *host firewall*). For example, iptables is the firewall application provided as a part of the Linux operating system. Several major firewall companies offer a software version of their network firewall. It is possible to buy off-the-shelf hardware (for example, a server) and run the firewall software. The advantage of software firewalls is their ability to upgrade without replacing the hardware. In addi-

tion, it is easier to add new features—for example, iptables can easily perform stateful filtering, NATing, and quality-of-service (QoS) operations. It is as simple as updating and configuring the firewall software.

Hardware firewalls rely on hardware to perform packet filtering. The policy and matching operation is performed in dedicated hardware—for example, using a field-programmable gate array. The major advantages of a hardware firewall are increased bandwidth and reduced latency. Note that bandwidth is the number of packets a firewall can process per unit of time, and latency is the amount of time required to process a packet. They are not the same thing, and IETF RFC 3511 provides a detailed description of the process of testing firewall performance.⁷

Hardware firewalls can operate at faster bandwidths, which translates to more packets per second (10 Gbps is easily achieved). In addition, hardware firewalls can operate faster since processing is performed in dedicated hardware. The firewall operates almost at wireline speeds; therefore, very little delay is added to accepted packets. This is important since more applications, such as multimedia, need QoS for their operation. The disadvantage is that upgrading the firewall may require replacement of hardware, which can be more expensive.

10. Choosing the Correct Firewall

The previous sections have described several categories of firewalls. Firewalls can be packet filters or stateful firewalls and/or provide application layer processing; implemented at the host or network or implemented in software or hardware. Given the possible combinations, it can be difficult to choose the appropriate technology.

When determining the appropriate technology, it is important to first understand the current and future security needs of the computer system being protected. Given a large number of hosts, a network firewall is probably the easiest to manage. Requiring and relying on every computer in an internal network to operate a host firewall may not be realistic.

Furthermore, updating the policy in a multiple host-based firewall system would be difficult. However, a single network firewall may imply that a single policy is suitable for all computers in the internal network. This generally is not the case when there are servers and computers in the internal network. More expensive network firewalls will allow the implementation of multiple policies or objects (described in more detail in the next section). Of course, if speed is an issue, a hardware firewall may justify the generally higher cost.

If scanning for viruses and spam and/or discovering network attacks are also requirements, a more advanced firewall is needed. Sometimes called an IPS, these advanced devices filter based on packet headers and inspect the data transmitted for certain signatures. In addition, these devices can monitor traffic (usage and connection patterns) for attacks. For example, a computer that attempts to connect to a range of ports on another computer is probably *port scanning*. This can be done to determine what network-oriented programs are running and in some cases even the operating system can be determined. It is a good idea to block this type of network reconnaissance, which an advanced firewall can do.

Although already introduced in this chapter, it is worth mentioning IETF RFC 3511 again. This document describes how firewalls should be tested to measure performance. This information helps the buyer understand the performance numbers cited by manufacturers. It is also important to ask whether the device was tested under RFC 3511 conditions.

11. Firewall Placement and Network Topology

A simple firewall typically separates two networks: one trusted (internal—for example, the corporate network) and one untrusted (external—for example, the Internet). In this simple arrangement, one security policy is applied to secure all the devices connected to the internal network. This may be sufficient if all the computers perform the same duties, such as desktop computers; however, if the internal network consists of different types of computers (in terms of the services provided), a single policy or level of protection is not sufficient or is difficult to create and maintain.

For example, the security policy for a Web server will be different from the security policy for a desktop computer. This is primarily due to the type of external network access each type of computer needs. Web servers would probably accept almost any unsolicited HTTP (port 80) requests arriving from the Internet. However, desktop computers probably do not serve Web pages and should not be subject to such requests.

Therefore it is reasonable to expect that different classes of computers will need different security policies. Assume an internal network consists of one Web server and several desktop computers. It is possible to locate the Web server on the outside, on the firewall (on the side of the external network), but that would leave the Web server without any firewall protection. Furthermore, given that the Web server is on the outside, should the administrator trust it?

Of course, the Web server could be located in the internal network (see [Fig. 76.3](#)) and a rule can be added to the policy to allow Web traffic to the Web server (often called *poking a hole*). However, if the Web server is compromised, the remaining computers in the internal network are vulnerable. Most attacks are multistage, which means the first target of attack is rarely the objective. Most attackers use one computer to compromise another until the objective is achieved. Therefore it is a good practice to separate machines and services, even in the internal network.

Demilitarized Zones

Another strategy often employed to provide different types of protection is a DMZ, as shown in [Fig. 76.3](#). Assume the firewall has three connections (sometimes called a *multihomed* device)—one for the external network (Internet), one for the Web server, and another for the internal network. For each connection to the firewall (referred to as an *interface*), a different firewall policy can be enforced, providing different forms of protection. The connection for the Web server is called the DMZ, and it prevents users from the external network getting direct access to the other computers in the internal network. Furthermore, if the Web server is compromised, the internal network still has some protection, since the intruder would have to cross the firewall again to access the internal network.

If the firewall only supports two interfaces (or just one policy), multiple firewalls can be used to achieve the same DMZ effect. The first firewall would be placed between the external network and the Web server. The second firewall would connect the Web server to the internal network. Given this design, the first firewall policy would be less restrictive than the second. Again, different levels of security are now possible.

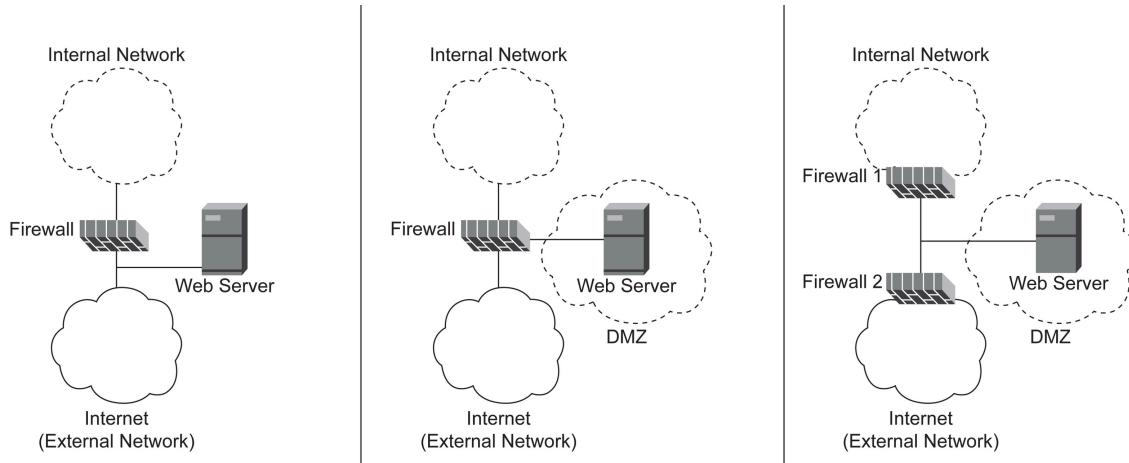


FIGURE 76.3 Example firewall configurations. Left configuration has a Web server outside the internal network. The middle configuration has the Web server in a demilitarized zone. The right configuration is another example of a demilitarized zone.

Grouping machines together based on similar firewall security needs is increasingly common and is seen as a good practice. Large networks may have server farms or a group of servers that perform similar services. As such, each farm is connected to a firewall and given a unique security policy. For example, users from the internal network may have access to administrative servers, but Web servers may have no access to the administrative servers. Such groupings are also referred to as *enclaves*.

Perimeter Networks

A perimeter network is a subnetwork of computers located outside the internal network.⁸ Given this definition, a DMZ can be considered a type of perimeter network. The primary difference between a DMZ and a perimeter network is the way packets arriving and departing the subnet-work are managed.

In a perimeter network, the device that connects the external network to the perimeter network is a *router*, whereas a DMZ uses a firewall to connect to the Internet. For a DMZ, a firewall policy will be applied to all packets arriving from the external network (Internet). The firewall can also perform advanced services such as NATing and packet payload inspection. Therefore it is easy to see that a DMZ offers a higher level of protection to the computers that are part of the perimeter and internal networks.

Two-Router Configuration

Another interconnection of subnetworks is the two-router configuration.⁹ This system consists of an external router, a bastion host, an internal router, and an internal network. The *bastion host* is a computer that serves as a filter and/or proxy for computers located in the internal network.

Before describing the specifics of the two-router configuration, let us define the duties of a bastion host. A bastion host is the first device any external computer will contact before accessing a computer in the internal network. Therefore the bastion host is fully exposed to the Internet and should be made as secure as possible. There are several types of bastion hosts, including victim machines that provide insecure but necessary services. For our discussion the bastion host will provide proxy services, shielding (to a limited degree) internal computers from external threats.

For the two-router configuration, the external network connects to the external router, which connects to the bastion host. The bastion host then connects to the internal router, which also connects to the internal network. The routers can provide limited filtering, whereas the bastion host provides a variety of proxy services—for example, HTTP, SSH, IRC, and File Transfer Protocol (FTP). This provides some level of security, since attackers are unaware of some internal network details. The bastion host can be viewed as a part of a very small perimeter network.

Compared to the DMZ, a two-router system provides less security. If the bastion host is compromised, the computers in the internal network are

not immediately vulnerable, but it would only be a matter of time before they were. Therefore the two-router design should be limited to separating internal subnetworks. If the internal router is a firewall, the design is considerably more secure.

Dual-Homed Host

A *dual-homed host* system consists of a single computer separating the external network from internal computers.¹⁰ Therefore the dual-homed computer needs at least two network interface cards (NICs). One NIC connects to the external network; the other connects to the internal network —hence the term *dual-homed*. The internal connection is generally a switch that connects the other internal computers.

The dual-homed computer is the location where all traffic arriving and departing the internal network can be processed. The dual-homed computer can perform various tasks such as packet filtering, payload inspection, NAT, and proxy services. Given the simple design and low cost, this setup is popular for home networks. Unfortunately, the dual-homed approach introduces a single point of failure. If the computer fails, then the internal network is isolated from the external network. Therefore this approach is not appropriate for businesses that rely on the Internet.

Network Configuration Summary

This section described various network configurations that can be used to provide varying levels of security. There are certainly variations, but this part of the chapter attempted to describe the most prevalent:

1. • *Demilitarized zones (DMZs)*. When correctly configured, DMZs provide a reasonable level of security. Servers that need to be available to the external network are placed outside the internal network but have a firewall between them and the external network.
2. • *Perimeter networks*. A perimeter network consists of a subnetwork of systems (again, those that need to be available to the external network) located outside the internal network. The perimeter subnet-

work is separated from the external network by a router that can provide some basic packet filtering.

3. • *Two-router configuration.* The two-router configuration places a bastion host between the internal and external networks. One router is placed between the internal network and bastion host, and the other router is placed between the bastion host and the external network. The bastion host provides proxy services, which affords some security (but not much).
4. • *Dual-homed configuration.* A dual-homed configuration has one computer that has at least two network connections—one connected to the external network and another to the internal network. All traffic must transmit through the dual-homed system; thus it can act as a firewall, NAT, and/or IDS. Unfortunately, this system has a single point of failure.

12. Firewall Installation and Configuration

Before a firewall is actually deployed, it is important to determine the required services and realize the vulnerabilities that may exist in the computer system that is to be secured. Determining the services requires a detailed understanding of how the computers in the network are interconnected, both physically and from a service-oriented perspective. This is commonly referred to as *object discovery*.

For example, given a database server, which services should the server provide? Which computers should be allowed to connect? Restated, which ports should be open and to whom? Often object discovery is difficult since it is common that a server will be asked to do various tasks over time. Generally a multiservice server is cheaper (one server providing Web, email, and database), but it is rarely more secure. For example, if a multiservice server is compromised via one service, the other services are vulnerable to attack. In other words, the rules in the firewall policy are usually established by the list of available services and secure computers.

Scanning for vulnerabilities is also helpful when you are installing a firewall. Several open-source tools are available to detect system vulnerabilities.

ties, including netstat, which shows open services. Why not simply patch the vulnerability? Perhaps the patch is not available yet, or perhaps the application is deemed necessary but it is simply insecure (FTP is an example). Network mappers such as Nessus are also valuable in showing what information about the internal network is available from the outside. Knowing the internal network layout is invaluable in attacking a system, since most modern attacks are multistaged. This means that one type of system vulnerability is typically leveraged to gain access elsewhere within the network.

A simple and unfortunately common security risk is a Web server that is connected to another internal server for data. Assume that Network File System (NFS) is used to gain access to remote data. If the Web server is compromised, which will probably occur at some time, then all the data inside the data server may be at risk (depending on how permissions have been set) and access to the data could be the true objective of the attacker. Therefore, understanding the interconnection of internal machines can help identify possible multistage attacks.

Of course the process of determining services, access rights, and vulnerabilities is not a one-time occurrence. This process should repeat over time as new computers, operating systems, users, and so on are introduced. Furthermore, firewall changes can cause disruption to legitimate users; these cases require tracing routes, defining objects, and reading policies. Managing a firewall and its policy requires constant vigilance.

13. Supporting Outgoing Services Through Firewall Configuration

As described in the first section, a firewall and the policy govern access to and from an internal network (the network being administered). A firewall applies a policy to arriving packets, then determines the type of access. The policy can be represented as an ordered set of rules; again, assume that the first-match criterion is used. When a packet arrives, it is compared to the first rule to determine whether it is a match. If it is, then the associated action is performed; otherwise the next rule is tested. Actions include accepting, denying, and logging the packet.

For a simple packet filter, each rule in the policy will describe a certain range of packet headers that it will match. This range of packets is then defined by describing certain parts of the packet header in the rule. For the Internet (TCP/IP networks) there are five such parts that can be described: source IP, source port, destination IP, destination port, and protocol.

Recall that the source IP is the address of the computer that originated the packet. The source port is the number associated with the application that originated the packet. Given the IP address and port number, it is possible to determine the machine and application, within reason. The destination IP and port number describe the computer and the program that will receive the packet. Therefore, given these four pieces of information, it is possible to control the access to and from a certain computer and program. The fifth piece of information is the communication protocol, UDP or TCP.

At this point it is important to also consider the direction of traffic. When referring to a packet, did it come from the external network and is it destined for an internal computer, or vice versa? If the packet is considered inbound, the source and destination addresses are in one order; outbound would reverse the order. Unfortunately, many firewalls will consider any arriving packet as inbound, regardless of where it originated (external or internal network), so the administrator must consider the direction when designing the policy. For example, iptables considers packets as locally or nonlocally generated. Locally generated packets are created at the computer running the firewall; all others are nonlocal, regardless of the source network.

Many firewalls can go beyond the five tuples (TCP/IP packet header parts) described. It is not uncommon to have a rule check the Medium Access Control (MAC) address or hardware address. This can be applied to filter-spoofed addresses. Filtering on the Type of Service field is also possible to treat packets differently—for better service, for example.

As previously described, maintaining the state of a connection is important for filtering traffic. For example, maintaining state allows the returning traffic to be accepted if the request was initiated from the internal

network. Note that in these simple cases we are only considering two computers communicating—for example, an internal workstation connecting to an external Web server.

Forms of State

The state of a connection can be divided into three main categories: new, established, and related. The new state indicates that this is the first packet in a connection. The established state has observed traffic from both directions, so the minimum requirement is that the source computer sends a packet and receives a packet in reply. The new state will change to *established* once the reply packet is processed by the firewall.

The third type of state is *related*, which is somewhat complicated. A connection is considered related if it is associated with an established connection. Therefore an established connection may create a new connection, separate from the original, which is considered related. The common example of this process is the FTP, which is used to transmit data from a source computer to a destination computer. The process begins with one connection from source to destination on port 21, the command connection. If there is data to be transferred, a second connection is created on port 20 for the data. Hence the data connection is related to the initial control connection. To simplify the firewall policy, it is possible to add a single rule to permit related connections.

In the previous example, the two computers communicating remained the same, but new connections were created, which can be managed in a table. However, understanding related connections is problematic for many new services. One example is the CORBA, which allows software components to be executed on different computers. This communication model may initiate new connections from different computers, similar to peer-to-peer networking. Therefore it is difficult to associate related connections.

Payload Inspection

Although firewalls originally only inspected the packet header, content filtering is increasingly commonplace. In this case the packet payload

(also called *contents* or *data*) is examined for certain patterns (analogous to searching for certain words on a page). These patterns, or signatures, could be for inappropriate or illegal content, spam email messages, or intrusion attempts. For example, it is possible to search for certain URLs in the packet payload.

The patterned searched for is often called a *signature*. If the pattern is found, the packet can be simply dropped, or the administrator may want to log the connection. In terms of intrusion signatures, this includes known patterns that may cause a buffer overflow in a network service.

Content filtering can be used to provide differentiated services as well. For example, if the firewall can detect that a connection is used for multimedia, it may be possible to provide more bandwidth or disconnect it, depending on the policy. Of course, content filtering assumes that the content is available (readable), which is not the case when encryption is used. For example, many worms encrypt their communications to prevent content filtering at the firewall.

Examining the packet payload normally requires significantly more processing time than normal header inspection. A signature may actually contain several patterns to match, specifying where they should occur relative to the packet beginning and the distance between patterns in the signature. This is only a short list of potential signature characteristics.

A signature can also span multiple packets—for example, a 20-byte signature could occur over two 10-byte IP fragments. Recall that IP may fragment packets based on the maximum transfer unit (MTU) of a link. Therefore the system may have to reassemble fragments before the scanning can begin. This necessary reassembly will further delay the transmission of data, which is problematic for certain types of applications (for example, multimedia). However, at this point, the discussion is more about IDSs than firewalls.

Over the years several techniques have been developed to decrease the amount of time required for payload inspection. Faster searching algorithms, dedicated hardware, and parallel searching techniques have all

shown promise in this regard. However, payload inspection at high bandwidths with low latency often requires expensive equipment.

14. Secure External Services Provisioning

Often we need a server that will provide services that are widely available to the external network. A Web server is a simple example of providing a service (Web pages) to a potentially large set of users (both honest and dishonest). As a result the server will be subjected to malicious intrusion attempts during its deployment.

Therefore systems that provide external services are often deployed on the edge or perimeter of the internal network. Given the location, it is important to maintain secure communications between it and other servers. For example, assume that the Web server needs to access a database server for content (PHP and MySQL); the connection between these machines must be secure to ensure proper operation.

A common solution to secure communications is the use of a virtual private network (VPN) that uses encryption to tunnel through an insecure network and provide secrecy. Advanced firewalls can create VPNs to different destinations, including mobile users. The first and most popular protocol for VPN is Internet Security Protocol (IPsec), which consists of standards from IPv6 ported to IPv4.

15. Network Firewalls for Voice and Video Applications

The next generation of network applications is expected to better leverage different forms of media. This is evident with the increased use of VoIP instead of traditional line-line telephones. Teleconferencing is another application that is seeing a steady increase in use because it provides an easy method for collaborating with others.

Teleoperations is another example that is seeing recent growth. These applications allow operators to control equipment that is at another location over the network (for example, telemedicine). Of course these examples

assume that the network can provide QoS guarantees, but that is a separate discussion.

Generally speaking, these applications require special handling by network firewalls. In addition, they normally use more than one connection. For example, the audio, video, and control information of a multimedia application often uses multiple network connections.

Multimedia applications also use multiple transport protocols. Control messages can be sent using TCP, which provides a reliable service between the sender and receiver. The media (voice and/or video) is typically sent using UDP. Often Real-Time Transport Protocol (RTP) is used, but this protocol is built on UDP. UDP is unreliable but faster than TCP, which is more important for multimedia applications.

As a result, these connections must be carefully managed by the firewall to ensure the proper operation of the application. This includes maintaining state across multiple connections and ensuring that packets are filtered with minimal delay.

Packet Filtering H.323

There are a few multimedia standards for transmitting voice and video over the Internet. Session Initiation Protocol (SIP) and H.323 are two examples commonly found in the Internet. The section briefly describes H.323 to illustrate the support required by network firewalls.

H.323 is the International Telecommunications Union (ITU) standard for videoconferencing. It is a high-level standard that uses other lower-level standards for the actual setup, transmission, control, and tear-down of a videoconference. For example, G.711 is used for encoding and decoding speech, and H.245 is used to negotiate the connections.

During H.323's operation, one port will be used for call setup using the static port 1720 (easy for firewalls). Each datastream will require one dynamically allocated TCP port for control and one dynamically allocated UDP port for data. As previously described, audio and video are transmitted separately.

Therefore an H.323 session will generate at least eight dynamic connections, which makes packet processing at the firewall very difficult. How does a firewall know which ports to open for an H.323 session? This is referred as a *lack of symmetry* between the computer located in the internal network and the computer located in the external network.

A stateful firewall can inspect the packet payloads and determine the dynamic connection port numbers. This information (negotiated port numbers) is placed in higher-level protocols, which are difficult to quickly parse and can be vendor specific.

In 2005 the ITU ratified the H.460.17/.18/.19 standards, which describe how to allow H.323 to traverse a firewall (or a NAT router/firewall, which essentially has the same problem). H.460.17 and H.460.18 deal with signaling, whereas H.460.19 concerns media. The H.460 standards require the deployment of stateful firewalls and updated H.323 equipment. This is a solution, but it remains a complex problem.

16. Firewalls and Important Administrative Service Protocols

There are a large number of administrative network protocols that are used to manage computer systems. These protocols are typically not complex to control at the firewall since dynamic connections are not used and little state information is necessary. The administrator should be aware of these services when designing the security policy, since many can be leveraged for attacks. This section reviews some of these important protocols.

Routing Protocols

Routing protocols are used to distribute routing information between routing devices. This information will change over time based on network conditions; therefore this information is critical to ensure that packets will get to their destinations. Of course, attackers can also use routing protocols for attacks. For example, maliciously setting routes such that a certain network is not reachable can be considered a denial-of-service

(DoS) attack. Securing routers and routing protocols is a continuing area of research. The firewall can help prevent these attacks (typically by not forwarding such information).

In considering routing protocols, it is important to first determine which devices in the internal network will need to receive and submit routing information. More than likely only devices that are directly connected to the external network will need to receive and respond to external routing changes—for example, the gateway router(s) for the internal network. This is primarily due to the hierarchical nature of routing tables, which does not require an external host to know the routing specifics of a distant subnetwork. As a result, there is typically no need to forward routing information from the external network into the internal network, and vice versa.

Routing Information Protocol (RIP) is the oldest routing protocol for the Internet. The two versions of RIP differ primarily by the inclusion of security measures. RIPv1 is the original protocol, and RIPv2 is the same but supports classless addresses and includes some security. Devices that use RIP will periodically (approximately every 30 s) broadcast routing information to neighboring hosts. The information sent by a host describes the devices they are directly connected to and the cost. RIP is not very scalable so is primarily used for small networks. RIP uses UDP to broadcast messages; port 520 is used by servers, whereas clients use a port above 1023.

Another routing protocol is Open Short Path First (OSPF), which was developed after RIP. As such OSPF is considered an improvement because it converges faster and it incorporates authentication. Interestingly, OSPF is not built on the transport layer but instead talks directly to IP. It is considered protocol 89 by the IP layer. OSPF messages are broadcast using two special multicast IP addresses: 224.0.0.5 (all SPF/link state routers) and 224.0.0.6 (all designated routers). The use of multicast addresses and setting the packet Time to Live (TTL) to one (which is done by OSPF) typically means a firewall will not pass this routing information.

Internet Control Message Protocol

Internet Control Message Protocol (ICMP) is used to send control messages to network devices and hosts. Routers and other network devices monitor the operation of the network. When an error occurs, these devices can send a message using ICMP. Messages that can be sent include destination unreachable, time exceeded, and echo request.

Although ICMP was intended to help manage the network, unfortunately attackers can use it as well. Several attacks are based on ICMP messages since they were originally allowed through the firewall. For example, simply forging a “destination unreachable” ICMP message can cause problems.

The program ping is one program that uses ICMP to determine whether a system is connected to the Internet (it uses the ICMP messages Echo Request and Echo Reply). However, this program can also be used for a smurf attack, which causes a large number of unsolicited ping replies to be sent toward one computer. As a result most firewall administrators do not allow ping requests or replies across the firewall.

Another program that uses ICMP is traceroute, which determines the path (list of routers) between a source and destination. Finding the path is done by sending multiple packets, each with an increasing TTL number (starting at one). When a router encounters a packet that it cannot forward due to the TTL, an ICMP message is sent back to the source. This reveals the router on the path (assuming that the path remains the same during the process). Most administrators do not want to provide this information, since it can show addresses assigned to hosts, which is useful to attackers. As a result firewalls are often configured to only allow traceroute requests originating from the internal network or limiting replies to traceroute originating from known external computers.

ICMP is built on the IP layer, like TCP and UDP. A firewall can filter these messages based on the message code field, which is a number that corresponds to each type of error message. Although this section described the problems with allowing ICMP messages through the firewall, an administrator may not want to block all ICMP packets. For example, MTU messages are important for the transmission of packets and probably should be allowed.

Network Time Protocol

Network Time Protocol (NTP) is a protocol that allows the synchronization of system clocks (from desktops to servers). Having synchronized clocks is not only convenient but required for many distributed applications. Therefore the firewall policy must allow the NTP service if the time comes from an external server.

NTP is a built-on UDP, where port 123 is used for NTP server communication and NTP clients use port 1023 (for example, a desktop). Unfortunately, like many legacy protocols, NTP suffers from security issues. It is possible to spoof NTP packets, causing clocks to set to various times (an issue for certain services that run periodically). There are several cases of NTP misuse and abuse where servers are the victim of DoS attacks.

As a result, if clock synchronization is needed, it may be better to provide an internal NTP server (master clock) that synchronizes the remaining clocks in the internal network. If synchronization is needed by an NTP server in the Internet, consider using a bastion host.

Central Log File Management

Almost every operating system maintains a system log where important information about a system's state is reported. This log is a valuable resource for managing system resources and investigating security issues.

Given that almost every system (especially a server) generates log messages, having this information at a central location is beneficial. The protocol syslog provides this functionality, whereby messages can be forwarded to a syslog server, where they are stored. An attacker will commonly attempt to flood the syslog server with fake messages in an effort to cover their steps or to cause the server disk to fill, causing syslog to stop.

Syslog runs on UDP, where syslog servers listen to UDP port 514 and clients (sending log messages) use a port above 1023. Note that a syslog

server will not send a message back to the client, but the syslog log server can communicate, normally using port 514.

Generally allowing syslog communication between the external and internal network is not needed or advised. Syslog communications should be limited to internal computers and servers; otherwise a VPN should be used to prevent abuse from others and to keep the information in the messages private.

Dynamic Host Configuration Protocol

The Dynamic Host Configuration Protocol (DHCP) provides computers essential information when connecting to an IP network. This is necessary because a computer (for example, a mobile laptop) does not have an IP address to use.

The computer needing an IP address will first send a broadcast request for an IP address. A DHCP server will reply with the IP address, netmask, and gateway router information the computer should use. The address provided comes from a pool of available IP addresses, which is managed by the DHCP server. Therefore the DHCP provides a method of sharing IP addresses among a group of hosts that will change over time.

The actual exchange of information is more elaborate than described here, but this is enough information for our discussion. In general the DHCP server providing addresses will be located in the internal network. As a result, this information should not be transmitted across the firewall that separates the internal and external networks. Why would you want to provide IP addresses to computers in the external network?

17. Internal IP Services Protection

Domain Name Server (DNS) provides the translation between the host-name and the IP address, which is necessary to send packets in the Internet. Given the number of hostnames in the Internet, DNS is built on a hierarchical structure. The local DNS server cannot store all the possible hostnames and IP addresses, so this server will need to occasionally re-

quest a translation from another DNS server located in the external network. As a result it is important to configure the firewall to permit this type of lookup.

In many cases the service provider provides the address of a DNS server that can be used to translate external hostnames. There is no need to manage a local DNS server in this case. However, it is possible to manage a local DNS, which allows the internal network to use local hostnames (these can be published to the external network). Some advanced firewalls can provide DNS, which can help hide internal computer hostnames and IP addresses. As a result, external computers have a limited view of the internal network.

Another important service that can be provided by the firewall is NAT. NAT is a popular method for sharing a smaller set of IP addresses across a larger number of computers. Recall that every packet has a source IP, source port, destination IP, and destination port. Assume that a small network only has one external IP address but has multiple computers that need to access the Internet. Note the external address is a routable address, whereas the internal computers would use a private address (addresses have no meaning outside the internal network). NAT will allow the internal machines to share the single external IP address.¹¹

When a packet arrives from a computer in the internal network, its source address is replaced with the external address, as shown in **Fig. 76.4**. The packet is sent to the destination computer, which returns a packet. The return packet has the external address (which is routable), so it is forwarded to the firewall. The firewall can then replace the external destination address with the correct internal destination address. What if multiple internal machines send a packet to a server in the external network? The firewall will replace the source address with the external address, but how will the firewall differentiate the return packets? NAT will also change the source port number, so each connection can be separated.

The NAT process described in the preceding paragraph is *source NAT*,¹² which works for packets initiated in the internal network. There is also *destination NAT*, which works in a similar fashion for packets initiated in

the external network. In this case the firewall needs to know which machine to forward packets to in the internal network.

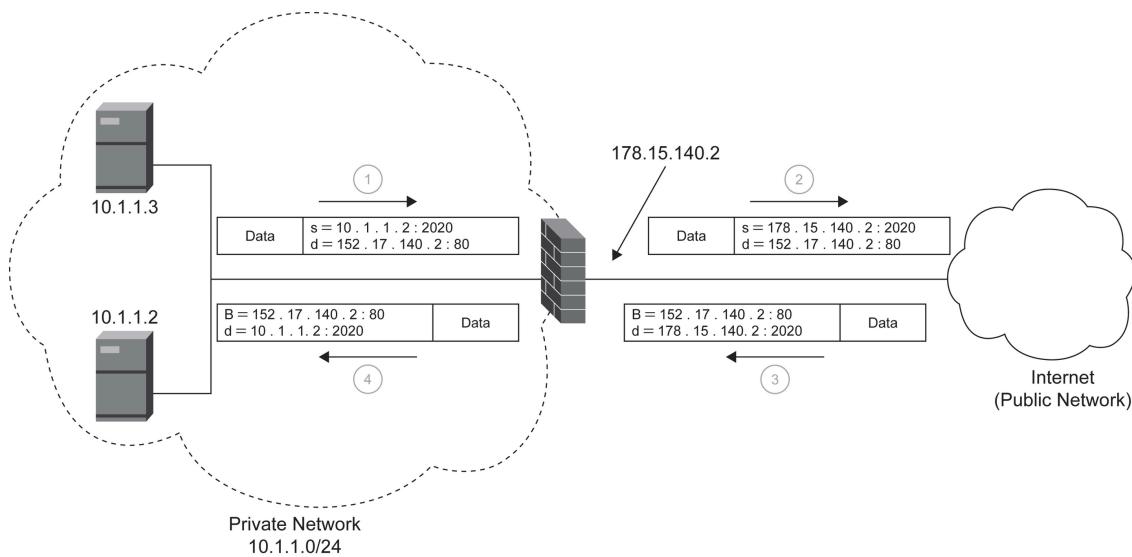


FIGURE 76.4 Example source network address translation (NAT). The connection originates from a computer in the internal network and is sent to a computer in the external network. Note the address and port number exchange performed at the firewall.

18. Firewall Remote Access Configuration

As described in the first section, firewalls are deployed to help maintain the privacy of data and authenticate the source. Privacy can be provided using encryption, for which there are several possible algorithms to use. These algorithms can be categorized as either secret key or public key. Secret key techniques use the same key to encrypt and decrypt information. Examples include IDEA, RC4, Twofish, and AES. Though secret key algorithms are fast, they require the key to be distributed between the two parties in advance, which is not trivial.

Public key encryption uses two keys—one to encrypt (the public key) and another to decrypt (the private key). The public key can be freely available for others to use to encrypt messages for the owner of the private key, since only the private key can decrypt a message. Key management sounds easy, but secure key distribution is difficult. How do you know the public key obtained is the correct one? Perhaps it is a man-in-middle attack. The Public Key Infrastructure, one method of distributing public keys, depends on a system of trusted key servers.

Authentication is another important component of security; it attempts to confirm a person who he or she claims to be. This can be done based on what the user has (ID card or security token) or by something a person knows (for example, a password). A very familiar method of authentication is requesting a username and password, which is common for VPNs.

Secrecy and authentication are also important when an entity manages multiple separate networks. In this case the administrator would like to interconnect the networks but must do so using an insecure network (for example, the Internet).

Tunneling from one firewall to another firewall can create a secure interconnection. This can be done using application proxies or VPN.

Application firewalls implement a proxy for each application supported. A user first contacts the firewall and authenticates before connecting to the server. The firewall then connects to the destination firewall, which then connects to the destination server. Three connections are thus involved.

An alternative is to construct a VPN from one firewall to another. Now a secure connection exists between the two networks. However, note that the VPN could also be used as an easy connection for an attacker who has successfully broken into one of the networks.

It is also important to note that tunneling can be used to transport packets over a network with a different transport protocol—for example, carrying TCP/IP traffic over Frame Relay.

19. Load Balancing and Firewall Arrays

As network speeds continue to increase, firewalls must continue to process packets with minimal delay (latency). Unfortunately, firewalls that can operate at these extreme data rates are also typically very expensive and cannot easily be upgraded to meet future demands. Load-balancing firewalls can provide an answer to this important problem.¹³

Load balancing (or *parallelization*) provides a scalable firewall solution for high-bandwidth networks and/or low-latency applications. This approach consists of an array of firewalls that process arriving packets in parallel. A simple system would consist of two load balancers connected to an array of firewalls, where each firewall is identically configured (same firewall policy), as depicted in [Fig. 76.5](#). One balancer connects to the Internet, then to the array (arriving traffic); the other balancer connects to the internal network, then to the array (departing traffic). Of course, one load balancer can be used instead of two separate load balancers.

When a packet arrives, it is sent to a firewall that currently has the lightest load (fewest number of packets awaiting processing), hence the term *load balancing*. As a result, the amount of traffic each firewall must process is roughly $1/n$ of the total traffic, where n is the number of firewalls in the array.

As with any new firewall implementation, the integrity of the policy must be maintained. This means that given a policy, a traditional single firewall and a load-balancing firewall will accept the same packets and deny the same packets. For static rules, integrity is provided, since the policy is duplicated at every firewall; therefore the set of accepted and denied packets at each firewall is also the same. As will be discussed in the next sections, maintaining integrity for stateful rules is not easy.

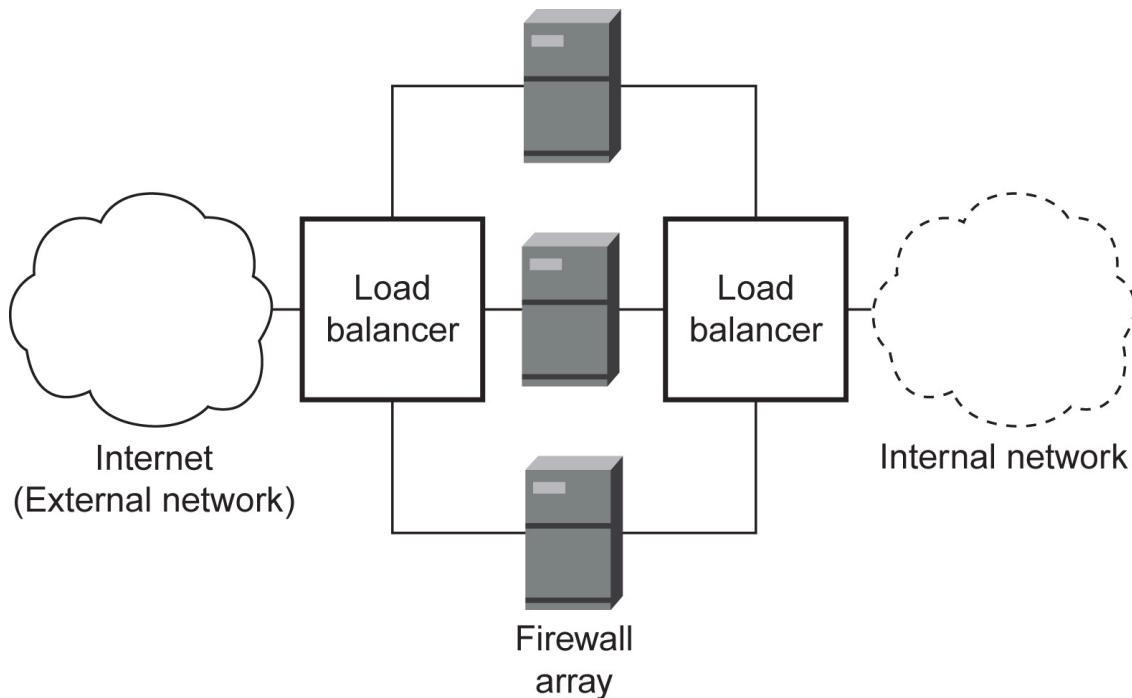


FIGURE 76.5 Load-balancing firewall array consisting of a three-firewall array and a load balancer.

Load Balancing in Real Life

A simple supermarket analogy can help describe the system and the potential performance increase. Consider a market consisting of an array of n cashiers. As with the firewall system, each cashier in the market is identical and performs the same duties. When a customer wants to pay for her items, she is directed to the cashier with the shortest line. The load balancer is the entity that would direct the customer, but in reality such a person rarely exists. A customer must guess which line is actually the best to join, which as we all know is not simple to determine.

Obviously, as more cashiers are added, the market can check out more customers. This is akin to increasing the bandwidth of the firewall system. Another important advantage of a load-balancing system is robustness. Even if a cashier takes a break (or a firewall in the array fails), the system will still function properly, albeit more slowly.

How to Balance the Load

An important problem with load-balancing firewalls is how to quickly balance the lines (queues) of packets. We are all aware that customers re-

quire different amounts of time to check out of a market. This is dependent on the number of items (which is observable) and their ability to pay (not easily observable). Similarly, it is difficult to determine how much time a packet will require at a firewall (for software-based systems). It will depend on the number of rules, organization of the rules, and which rule the packet will match.

A more important problem with load balancing is how to maintain state. As described in the preceding sections, some firewall rules will maintain the state of a connection. For example, a firewall rule may allow traffic arriving from the Internet only if an internal computer requested it. If this is the case, a new temporary rule will be generated to handle traffic arriving from the Internet. In a parallel system, where should this rule reside? Which firewall? The objective is to ensure that the integrity of the policy is maintained in the load-balancing system.

To use the market analogy again (and this will be a stretch), assume that a mother leaves her credit card with one cashier, then sends her children into the market to buy certain items. When the children are ready to pay for their items they must go to the cashier that has the credit card. If the children do not know which cashier has the credit card the load balancer must not only balance lines but also check a list to make certain the children join the correct line.

Maintaining state increases the amount of time the load balancer will spend per packet, which increases the latency of the system. An alternative solution is it to replicate the stateful rules across every firewall (or the credit card across every cashier). This requires an interconnection and state-aware program per firewall. Maintaining network connections is also difficult for applications that dynamically create new connections. Examples include FTP (one connection for control, the other for data), multimedia, and CORBA. Again, a new rule must be added to handle the new traffic.

Advantages and Disadvantages of Load Balancing

Given these issues, firewall load balancing is still done. There are several advantages and disadvantages to this approach. Disadvantages of load balancing include the following:

1. • *Load balancing is not trivial.* The load balancer seeks to ensure that the lines of packets across the array of firewalls remain equal. However, this assumes that the balancer can predict how much time a packet will require.
2. • *Maintaining state is difficult.* All packets that belong to a session will need to traverse the same firewall, or state information must be shared across the firewalls. Either solution is difficult to manage.

There are several advantages to load balancing:

1. • *Scalable solution for higher throughput.* If higher throughput is needed, adding more firewalls is simple and cost-effective.
2. • *Robustness.* If a firewall fails in the array, the integrity of the system remains. The only loss is throughput.
3. • *Easy policy management.* If the rules change, simply update the policy at each firewall.

The load balancer can be implemented in software or hardware.

Hardware load balancers provide better performance, but they will also have a much higher price.

20. Highly Available Firewalls

As previously discussed, a network firewall is generally considered easier to manage than multiple host-based firewalls. The administrator only manages one firewall and one policy, but this design also introduces a single point of failure. If the network firewall fails in the system, the entire internal network is isolated from the Internet. As a result, there is a real need to provide a highly available, or robust, firewall system.

The load-balancing system described in the previous section provides a greater degree of robustness. If a firewall in the array fails, the system is still functional; however, the capacity of the system is reduced, which is

considered acceptable under the circumstances. Unfortunately, the design still has a single point of failure—the load distributor. If the load distributor fails, the entire system fails.

A simple solution to this problem simply replicates the load distributor. The incoming connection is duplicated to both load distributors, and the distributors are then connected to the firewalls in the array. The distributors are interconnected via a lifeline to detect failure and possibly share information.

Load Balancer Operation

The two load balancers described in the previous section can operate in one of two modes: active-backup or active-active. In active-backup mode, one balancer operates as normal distributing packets to minimize delays and maintain state information. The second distributor is in backup mode and monitors the status of the active load balancer and duplicates any necessary state information. Upon load-balance failure, the backup device can quickly take over operation.

In contrast, active-active operation operates both load balancers in parallel. When a packet arrives at the system, it is processed by one of the load balancers, which forwards the packet to a firewall in the array. Of course, this seems as though there will be a load balancer for the load balancers, but this is not necessary. Under active-active mode, the load balancers use the lifeline to synchronize and determine which packets to process. Although the active-active mode can increase performance by using both load balancers simultaneously, it is more difficult to implement and complex to manage.

Interconnection of Load Balancers and Firewalls

In our simple example, the additional load balancer requires double the number of ports per firewall (one per load balancer). This design provides greater robustness but may also be cost prohibitive.

An alternative solution uses active-active mode and divides the array into two equal groups. One group is then connected to one load-balancer and

the other group is connected to the other. For example, consider an array of six firewalls. Three firewalls would be connected to one load balancer and the other six firewalls connected to the second load balancer.

Although this design only requires one port per firewall (on one side), if a load balancer fails, half the firewalls are nonoperational.

21. Firewall Management

Once a firewall has been deployed and policy created, it is important to determine whether it is providing the desired security. Auditing is the process of verifying the firewall and policy and consists of two steps. First, the administrator should determine whether the firewall is secure. If an attacker can exploit the firewall, the attacker has a significant advantage. Consider the information that can be gained just from knowing the firewall policy.

The firewall should be in a secure location and have the latest security patches (recall that many firewalls are implemented in software). Also ensure that the firewall only provides the necessary services, such as SSH, if remote access to the firewall is needed. Exploiting a firewall operating system or provided services is the most common method for breaking into a firewall. Therefore the services and access should be tightly controlled. User authentication with good passwords and secure connections should always be used.

Once the firewall has been secured, the administrator should review the policy and verify that it provides the security desired. Does it block the illegitimate traffic and permit legitimate traffic? This is not a trivial task, given the first-match criterion and the number of rules in the policy. It is easy to create firewall policies with anomalies, such as shadowing (a subsequent rule that is never matched because of an earlier rule). Some software packages are available to assist this process, but in general it is a difficult problem.

An administrator should periodically audit the firewall rules and test the policy to verify that the system performs as expected. In addition, the system should undergo penetration testing to verify correct implementation.

This includes seeded and blind penetration testing. *Seeded testing* includes detailed information about the network and configuration, so target systems and services can be tested. *Blind testing* is done without any knowledge of the system, so it is more complete but also more time-consuming.

Keeping backups of configurations and policies should be done in case of hardware failure or an intrusion. Logging at the firewall should also be performed, which can help measure performance. In addition, logs can show connections over time, which is useful for forensics and verifying whether the security policy is sufficient.

22. Summary

Network firewalls are a key component of providing a secure environment. These systems are responsible for controlling access between two networks, which is done by applying a security policy to arriving packets. The policy describes which packets should be accepted and which should be dropped. The firewall inspects the packet header and/or the payload (data portion).

There are several different types of firewalls, each briefly described in this chapter. Firewalls can be categorized based on what they inspect (packet filter, stateful, or application), their implementation (hardware or software), or their location (host or network). Combinations of the categories are possible, and each type has specific advantages and disadvantages.

Placement of the firewall with respect to servers and internal computers is key to the way these systems will be protected. Often servers that are externally available, such as Web servers, will be located away from other internal computers. This is often accomplished by placing these servers in a DMZ. A different security policy is applied to these computers so the access between computers in the DMZ and the internal network is limited.

Improving the performance of the firewall can be achieved by minimizing the rules in the policy (primarily for software firewalls). Moving more popular rules near the beginning of the policy can also reduce the number of rules comparisons that are required. However, the order of certain rules must be maintained (any rules that can match the same packet).

Parallel firewalls can provide greater performance improvements. These systems consist of a load balancer and an array of firewalls, where all the firewalls in the array are identical. When a packet arrives at the system, it is sent to one of the firewalls in the array. The load balancer maintains short packet queues, which can provide greater system bandwidth and possibly a lower latency.

Regardless of the firewall implementation, placement, or design, deployment requires constant vigilance. Developing the appropriate policy (set of rules) requires a detailed understanding of the network topology and the necessary services. If either of these items change (and they certainly will), that will require updating the policy. Finally, it is important to remember that a firewall is not a complete security solution but is a key part of a security solution.

Finally, let us move on to the real interactive part of this chapter: review questions/exercises, hands-on projects, case projects, and optional team case project. The answers and/or solutions by chapter can be found in [Appendix G](#).

Chapter Review Questions/Exercises

True/False

1. True or False? Network firewalls are a vital component for maintaining a secure environment and are often the first line of defense against attack.
2. True or False? When a packet arrives at a firewall, a hacker policy is applied to determine the appropriate action.
3. True or False? Multiple rules of a single firewall policy may match a packet—for example, a packet could match rules 2, 6, and 7 of the

policy.

4. True or False? For most firewalls, the first rule that matches a packet is not typically applied.
5. True or False? Given that a network firewall will not inspect all packets transmitted between multiple networks, these devices need to determine the appropriate match with minimal delay.

Multiple Choice

1. What refers to how often the rule is a match?
 - A. Worm
 - B. Virus
 - C. Backdoor
 - D. More popular
 - E. User-level Rootkit
2. What is the most basic type of a firewall since it only filters at the network and transport layers (layers two and three)?
 - A. Backdoor
 - B. Virus
 - C. Worm
 - D. Packet filter
 - E. User-level Rootkit
3. What perform the same operations as packet filters, but also maintain state about the packets that have arrived?
 - A. Stateful firewalls
 - B. Virus
 - C. Worm
 - D. Backdoor
 - E. User-level Rootkit
4. What can filter traffic at the network, transport, and application layer?
 - A. Application layer firewalls
 - B. Virus
 - C. Worm
 - D. Backdoor
 - E. User-level Rootkit
5. What can also be categorized based on where they are implemented or what they are intended to protect—host or network?

- A. Trojan Horse
- B. Firewalls
- C. Worm
- D. Backdoor
- E. User-level Rootkit

Exercise

Problem

Create a firewall policy that specifies how firewalls should handle inbound and outbound network traffic.

Hands-On Projects

Project

Identify all requirements that should be considered when determining which firewall to implement.

Case Projects

Problem

Create rulesets that implement the organization's firewall policy while supporting firewall performance.

Optional Team Case Project

Problem

Manage firewall architectures, policies, software, and other components throughout the life of the firewall solutions.

¹ Errin W. Fulp, Optimization of network firewall policies using directed acyclical graphs, In: Proceedings of the IEEE Internet Management

(Conference, 2005].

2 Errin W. Fulp, Optimization of network firewall policies using directed acyclical graphs, In: Proceedings of the IEEE Internet Management (Conference, 2005).

3 Errin W. Fulp, Optimization of network firewall policies using directed acyclical graphs, In: Proceedings of the IEEE Internet Management (Conference, 2005).

4 M. Yoon, Z. S. Zhang, Reducing the size of rule set in a firewall,” In Proceedings of the IEEE International Conference on (Communications, 2007).

5 J.R. Vacca, S. R. Ellis, Firewalls Jumpstart for Network and Systems (Administrators, Elsevier, 2005).

6 J.R. Vacca, S. R. Ellis, Firewalls Jumpstart for Network and Systems (Administrators, Elsevier, 2005).

7 B. Hickman, D. Newman, S. Tadjudin, T. Martin, Benchmarking Methodology for Firewall Performance, IETF RFC 3511, 2003.

8 J.R. Vacca, S. R. Ellis, Firewalls Jumpstart for Network and Systems (Administrators, Elsevier, 2005).

9 J.R. Vacca, S. R. Ellis, Firewalls Jumpstart for Network and Systems (Administrators, Elsevier, 2005).

10 J.R. Vacca, S. R. Ellis, Firewalls Jumpstart for Network and Systems (Administrators, Elsevier, 2005).

11 B. Hickman, D. Newman, S. Tadjudin, T. Martin, Benchmarking Methodology for Firewall Performance, IETF RFC 3511, 2003.

12 K. Egevang and P. Francis, The IP Network Address Translator (NAT), IETF RFC 1631, 1994.

¹³ Errin W. Fulp, Ryan J. Farley, A function-parallel architecture for high-speed firewalls, In Proceedings of the IEEE International Conference on (Communications, 2006).

Previous chapter

< [Chapter 75. Transmission Control
Protocol/Internet Protocol Packet Analysis](#)

Next chapter

[Chapter 77.
Penetration
Testing](#) >