CHAPTER

Firewalls

28

What You Will Learn

In this chapter, you will learn how firewalls add security to TCP/IP networks. We'll be working with both kinds of router-based firewalls: packet filters and stateful inspection.

You will learn about the types of dedicated firewalls that run on purpose-built hardware. We'll also examine firewall architectures and the use of DMZs. And because filtering works exactly the same with IPv6 as with IPv4, we will not have a special section on IPv6 firewalls.

If all data traveled the Internet encrypted inside VPNs, and all hosts only sent or received such data, the Internet would be a safer place. But the reality is messy—very messy—and denial of service attacks, hacker raids, spyware, spam, viruses, and worms make life interesting for everyone on-line.

As we write these words, teams are assembled in Las Vegas, Nevada, for the annual Defcon "contest." The name derives from Cold War "defense condition" levels and implies that hackers could have broken into military computers and started WW III, a plot device in several movies and books. Teams pay a small entry fee and compete in local and regional contests, all culminating in the finale in Las Vegas. The idea is to capture the secure "flags" or tokens on target systems set up for Defcon. All competitors' tokens are fair game, but, of course, you have to protect your own. (Taking over a competing team's network or Web server is considered a great coup.) Points are awarded for various successful exploits, and the winner is admired by all.

A certain percentage of people learning about networks and TCP/IP seem to indulge in some form of hacking at one time or another. It seems to be a rite of passage, like clubbing and drug experimentation. But most slackers eventually settle down and get real jobs, whereas a few others continue their dissolute ways. Some even make a career of their activities, as "white" or "black" hackers, and show up at places like Defcon. Hackers should never be judged solely on their appearance or demeanor, but only on their actions, which usually have consequences for everyone—intended or not.

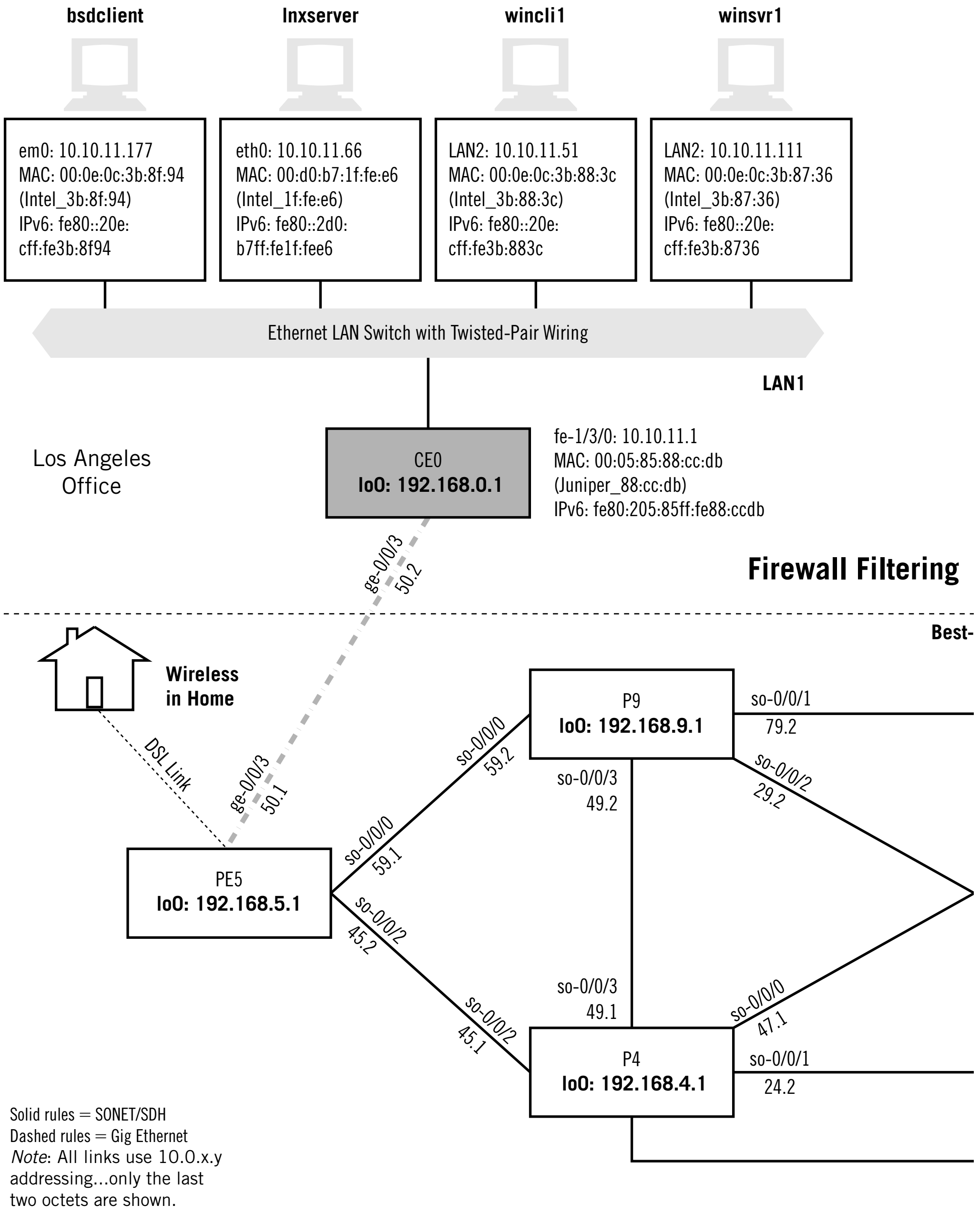
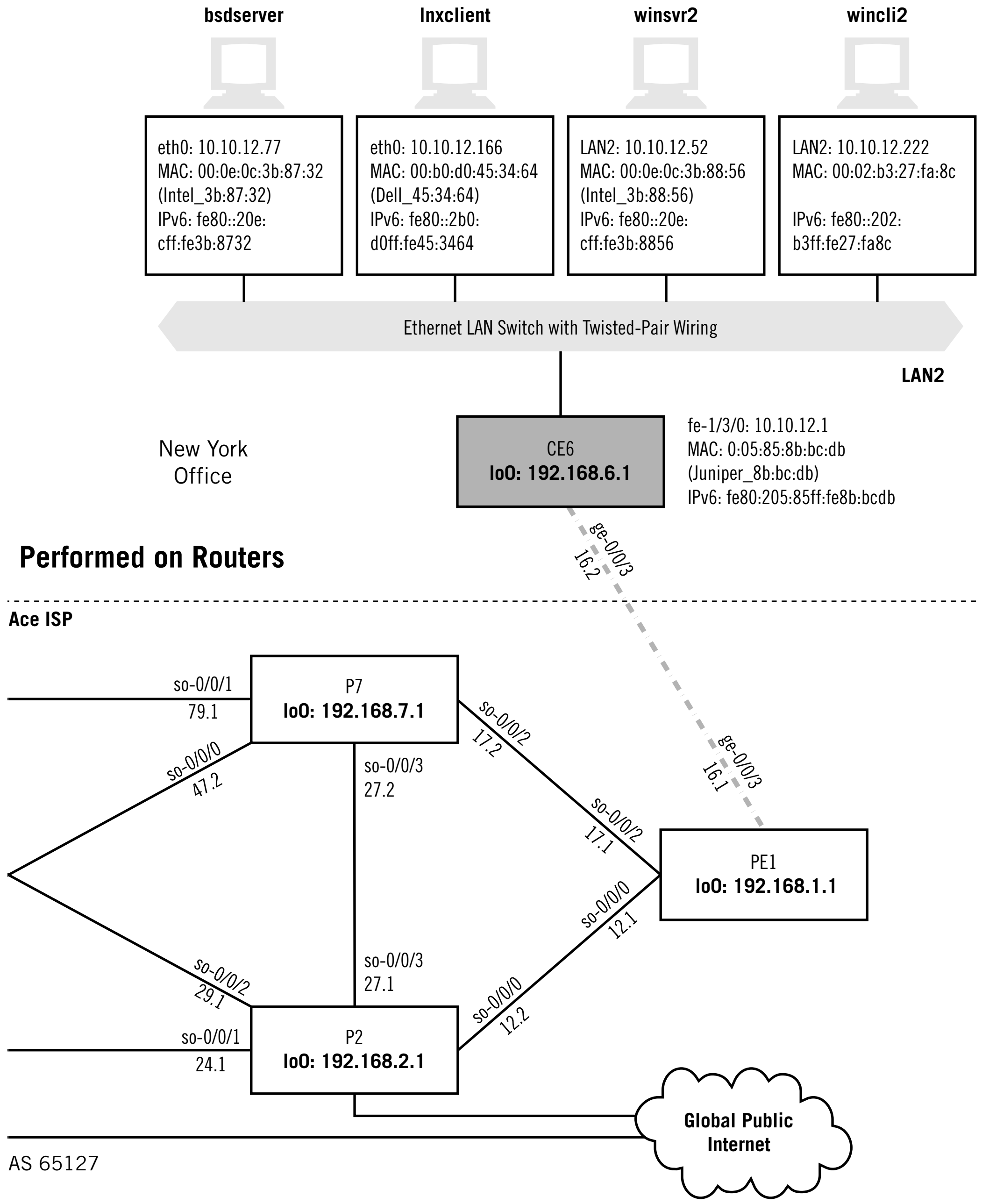


FIGURE 28.1

Firewalls on the Illustrated Network, showing how the firewall filtering is performed on the site routers.


bsdserver
lnxclient
winsvr2
wincli2
eth0: 10.10.12.77
MAC: 00:0e:0c:3b:87:32
(Intel_3b:87:32)
IPv6: fe80::20e:
cff:fe3b:8732
eth0: 10.10.12.166
MAC: 00:b0:d0:45:34:64
(Dell_45:34:64)
IPv6: fe80::2b0:
d0ff:fe45:3464
LAN2: 10.10.12.52
MAC: 00:0e:0c:3b:88:56
(Intel_3b:88:56)
IPv6: fe80::20e:
cff:fe3b:8856
LAN2: 10.10.12.222
MAC: 00:02:b3:27:fa:8c
IPv6: fe80::202:
b3ff:fe27:fa8c
Ethernet LAN Switch with Twisted-Pair Wiring
LAN2
New York Office
CE6
lo0: 192.168.6.1
fe-1/3/0: 10.10.12.1
MAC: 0:05:85:8b:bc:db
(Juniper_8b:bc:db)
IPv6: fe80:205:85ff:fe8b:bcdb
ge-0/0/3
16.2
Performed on Routers
Ace ISP
so-0/0/1
79.1
P7
lo0: 192.168.7.1
so-0/0/2
17.2
ge-0/0/3
16.1
so-0/0/0
47.2
so-0/0/3
27.2
so-0/0/2
17.1
PE1
lo0: 192.168.1.1
so-0/0/0
12.1
so-0/0/2
29.1
so-0/0/3
27.1
so-0/0/0
12.2
so-0/0/1
24.1
P2
lo0: 192.168.2.1
Global Public Internet
AS 65127

This chapter takes a look at firewalls, one technique for adding security to TCP/IP and the Internet. Firewalls can be hardware or software designed to protect individual hosts, clients, and servers or entire LANs from the one or more of the threats previously cited. We'll implement a couple of types of firewalls on our site routers, as shown in Figure 28.1.

WHAT FIREWALLS DO

Although the Illustrated Network has no dedicated firewall device (often called a firewall *appliance*), there are fairly sophisticated firewall capabilities built into our routers. So, we will configure firewall protection with two types of router-based firewall rules: *packet filters* and *stateful inspection*.

A Router Packet Filter

Let's do something fairly simple yet effective with a firewall packet filter on the Juniper Networks router on LAN2, CE6. Assume that malicious users on LAN1 are trying to harm `bsdserver` (`10.10.12.77`) on LAN2. We'll have to "protect" it from some of the hosts on LAN1.

We'll allow remote access with Telnet (this is just an example) or SSH from the `bsdclient` (`10.10.11.177`), and allow similar access attempts from `wincli1` (`10.10.11.51`), but log them. (What do those Windows guys want on the FreeBSD server?) We'll deny and log access from `lnxserver` (`10.10.11.66`) and `winsrv1` (`10.10.11.111`) because security policy for the organization has decided that users attempting remote access from servers are not allowed to do so.

The following is the firewall filter configured on CE6 and applied to the LAN2 interface. This filters IPv4 addresses, but we could easily make another to do the same thing for these hosts' IPv6 addresses. It is a good idea to keep in mind that `from` is more in the sense of "out of all packets," especially when the filter is applied on the output side of an interface. We also have to apply the filter to the `fe-1/3/0` interface, but this configuration snippet is not shown. There is a space between the three major terms of the `remote-access-control` filter: `allow-bsdclient`, `log-wincli`, and `deny-servers`. These names are strictly up to the person configuring the firewall filter.

```
set firewall family inet filter remote-access-control term
    allow-bsdclient from address 10.10.11.177/32; # bsdclient
set firewall family inet filter remote-access-control term
    allow-bsdclient from protocol tcp; # telnet and ssh use tcp
set firewall family inet filter remote-access-control term
    allow-bsdclient from port [ ssh telnet ]; # we could use numbers too
set firewall family inet filter remote-access-control term
    allow-bsdclient then accept; # allow bsdclient access

set firewall family inet filter remote-access-control term
    log-wincli1 from address 10.10.11.51/32; # wincli1
```

```
set firewall family inet filter remote-access-control term
    log-wincli1 from protocol tcp; # telnet and ssh use tcp
set firewall family inet filter remote-access-control term
    log-wincli1 from port [ ssh telnet ]; # we could use numbers too
set firewall family inet filter remote-access-control term
    log-wincli1 then log; # log wincli1 access attempts...
set firewall family inet filter remote-access-control term
    log-wincli then accept; # ...and allow wincli1 access

set firewall family inet filter remote-access-control term
    deny-servers from address 10.10.11.66/32; # lnxserver
set firewall family inet filter remote-access-control term
    deny-servers from address 10.10.11.111/32; # winsrv1
set firewall family inet filter remote-access-control term
    deny-servers from protocol tcp; # telnet and ssh use tcp
set firewall family inet filter remote-access-control term
    deny-servers from port [ ssh telnet ]; # we could use numbers too
set firewall family inet filter remote-access-control term
    deny-servers then log; # log server access attempts...
set firewall family inet filter remote-access-control term
    deny-servers then discard; # ...and silently discard those packets
```

When we try to remotely log in from `bsdclient` or `wincli1`, we succeed (and `wincli1` is logged). But when we attempt access from the servers, the following is what happens.

```
lnxserver# ssh 10.10.12.77
```

Nothing! We set the action to `discard`, which silently throws the packet away. A `reject` action at least sends an ICMP destination unreachable message back to the host. When we examine the firewall log on CE6, this is what we see. Action "`A`" is accept, and "`D`" is discard. We didn't log `bsdclient`, but caught the others. (The filter name is blank because not all filter names that are configured are available for the log.)

```
admin@CE6> show firewall log
Time     Filter     A Interface     Pro Source address   Destination Address
08:36:09 -          A fe-1/3/0.0    TCP 10.10.11.51      10.10.12.77
08:37:24 -          D fe-1/3/0.0    TCP 10.10.11.66      10.10.12.77
```

Stateful Inspection on a Router

Simple packet filters do not maintain a history of the streams of packets, nor do they know anything about the relationship between sequential packets. They cannot detect flows or more sophisticated attacks that rely on a sequence of packets with specific bits set. This degree of intelligence requires a different type of firewall, one that performs stateful inspection. (There are three types of firewall, as we'll see later.)

In contrast to a stateless firewall filter that inspects packets singly and in isolation, stateful filters consider state information from past communications and applications to make dynamic decisions about new communications attempts. To do this, stateful firewall filters look at *flows* or *conversations* established (normally) by five properties of TCP/IP headers: source and destination address, source and destination port, and protocol. TCP and UDP conversations consist of two flows: *initiation* and *responder*. However, some conversations (such as with FTP) might consist of two control flows and many data flows.

On a Juniper Networks router, stateful inspection is provided by a special hardware component: the Adaptive Services Physical Interface Card (AS PIC). We've already used the AS PIC to implement NAT in the previous chapter. This just adds some configuration statements to the services (such as NAT) provided by the special internal `sp-` (services PIC) interface.

Stateful firewalls do not just check a few TCP/IP header fields as packets fly by on the router. Stateful firewalls are intelligent enough that they can recognize a series of events as anomalies in five major categories.

1. IP packet anomalies
   - Incorrect IP version
   - Too-small or too-large IP header length field
   - Bad header checksum
   - Short IP total packet-length field
   - Incorrect IP options
   - Incorrect ICMP packet length
   - Zero TTL field
2. IP addressing anomalies
   - Broadcast or multicast packet source address
   - Source IP address identical to destination address (land attack)
3. IP fragmentation anomalies
   - Overlapping fragments
   - Missing fragments
   - Length errors
   - Length smaller or larger than allowed
4. TCP anomalies
   - Port 0
   - Sequence number 0 and flags field set to 0
   - Sequence number 0 with FIN/PSH/RST flags set
   - Disallowed flag combinations [FIN with RST, SYN/(URG/FIN/RST)]
   - Bad TCP checksum

5. UDP anomalies
   - Port 0
   - Bad header length
   - Bad UDP checksum

In addition, stateful firewall filters detect the following events, which are only detectable by following a flow of packets.

- SYN followed by SYN-ACK packets without an ACK from initiator
- SYN followed by RST packets
- SYN without SYN-ACK
- Non-SYN first packet in a flow
- ICMP unreachable errors for SYN packets
- ICMP unreachable errors for UDP packets

Stateful firewall filters, like other firewall filters, are also applied to an interface in the outbound or inbound direction (or both). However, the traffic on the interface must be sent to the AS PIC in order to apply the stateful firewall filter rules.

The AS PIC's `sp-` interface must be given an IP address, just as any other interface on the router. Traffic then makes its way to the AS PIC by using the AS PIC's IP address as a next hop for traffic on the interface. The next hop for traffic leaving the AS PIC (assuming the packet has not been filtered) is the "normal" routing table for transit traffic, `inet0`.

Stateful firewall filters follow the same `from` and `then` structure of other firewall filters. Keep in mind that `from` is more in the sense of "out of all packets," especially when the filter is applied on the output side of an interface. When applied to the LAN1 interface on the CE0 interface, in addition to detecting all of the anomalies previously listed, this stateful firewall filter will allow only FTP traffic onto the LAN unless it is from LAN2 and silently discards (rejects) and logs all packets that do not conform to any of these rules.

```
set stateful-firewall rule LAN1-rule match direction input-output;
set stateful-firewall rule LAN1-rule term allow-LAN2
  from address 10.10.12.0/24; # find the LAN2 IP address space
set stateful-firewall rule LAN1-rule term allow-LAN2
   then accept; # ...and allow it

set stateful-firewall rule LAN1-rule term allow-FTP-HTTP
   from application ftp; # find ftp flows
set stateful-firewall rule LAN1-rule term allow-FTP-HTTP
   then accept; #  ...and allow them

set stateful-firewall rule LAN1-rule term deny-other
   then syslog; # no 'from' matches all packets
set stateful-firewall rule LAN1-rule term deny-other
   then discard; # ...and syslogs and discards them
```

In the term `deny-other`, the lack of a `from` means that the term matches all packets that have not been accepted by previous terms. The `syslog` statement is the way that the stateful firewalls log events. We've also configured the interface `sp-1/2/0` and applied our stateful rule as `stateful-svc-set` (but the details are not shown).

Now when we try to run FTP to (for example) `lnxserver` from `bsdclient` or `wincli1`, we succeed. But watch what happens when we attempt to run FTP from one of the routers (the routers all support both FTP client and server software).

```
admin@CE6> ftp 10.10.11.66
```

Nothing! As before, this packet is silently discarded. But the stateful firewall filter gathers statistics on much more than simply "captured" packets.

```
admin@CE0> show services stateful-firewall statistics extensive
Interface: sp-1/2/0
  Service set: stateful-svc-set
    New flows:
      Accept: 7, Discard: 1, Reject: 0
    Existing flows:
      Accept: 35, Discard: 0, Reject: 0
    Drops:
      IP option: 0, TCP SYN defense: 0
      NAT ports exhausted: 0
    Errors:
      IP: 0, TCP: 0
      UDP: 0, ICMP: 0
      Non-IP packets: 0, ALG: 0
    IP errors:
      IP packet length inconsistencies: 0
      Minimum IP header length check failures: 0
      Reassembled packet exceeds maximum IP length: 0
      Illegal source address: 0
      Illegal destination address: 0
      TTL zero errors: 0, IP protocol number 0 or 255: 0
      Land attack: 0, Smurf attack: 0
      Non IP packets: 0, IP option: 0
      Non-IPv4 packets: 0, Bad checksum: 0
      Illegal IP fragment length: 0
      IP fragment overlap: 0
      IP fragment reassembly timeout: 0
TCP errors:
      TCP header length inconsistencies: 0
      Source or destination port number is zero: 0
      Illegal sequence number, flags combination: 0
      SYN attack (multiple SYNs seen for the same flow): 0
      First packet not SYN: 0
      TCP port scan (Handshake, RST seen from server for SYN): 0
      Bad SYN cookie response: 0
    UDP errors:
      IP data length less than minimum UDP header length (8 bytes): 0
      Source or destination port is zero: 0
      UDP port scan (ICMP error seen for UDP flow): 0
    ICMP errors:
      IP data length less than minimum ICMP header length (8 bytes): 0
      ICMP error length inconsistencies: 0
      Ping duplicate sequence number: 0
      Ping mismatched sequence number: 0

ALG drops:
      BOOTP: 0, DCE-RPC: 0, DCE-RPC portmap: 0
      DNS: 0, Exec: 0, FTP: 1
      H323: 0, ICMP: 0, IIOP: 0
      Login: 0, Netbios: 0, Netshow: 0
      Realaudio: 0, RPC: 0, RPC portmap: 0
      RTSP: 0, Shell: 0
      SNMP: 0, Sqlnet: 0, TFTP: 0
      Traceroute: 0
```

In the last section, `ALG drops` stands for application-level gateway drops, and we find the dropped FTP flow we attempted from the CE6 router. This shows the power and scope of stateful firewall filters.

TYPES OF FIREWALLS

Whether implemented as application software or as a special combination of hardware and software, firewalls are categorized as one of three major types, all of which have variations. Software firewalls can be loaded onto each host, but this only protects the individual host. Other software-based firewalls can be loaded onto a generic platform (Windows or Unix based) and used in conjunction with routers to protect the entire site. Alternatively, routers can be configured with policies (similar to routing policies), but designed to protect the networks attached to the router.

Most effective are very sophisticated packages of specialized hardware and state-of-the-art software, such as Juniper Networks Security Products. These dedicated devices are often called *appliances*, and operate much faster and scale much better than their general-purpose relatives. Software is updated frequently, as often as every 2 weeks, to ensure that customers have the latest capabilities for the effort to secure a site.

The three major types of firewall are the packet filter, application proxy, and stateful inspection. We've seen examples of packet filters and stateful firewalls, but each type has distinctive properties that should be described in some detail.

Packet Filters

Packet filters are the oldest and most basic form of firewall. Packet filters establish site security access rules (or *policies*) that examine the TCP/IP header of each packet and decide if it should be allowed to pass through the firewall. Policies can differ for inbound and outbound packets, and usually do. Many of the fields of the IP, TCP, or UDP header can be examined, but there is no concept of a session or flow of packets in this type of firewall.

Even basic DSL routers do a good job of implementing packet filters. For home networks, this might be adequate. But packet filters do not know much about the application that the packet represents or look at the value of the TCP flags. Packet filters cannot *dynamically* create access rules that allow responses which are associated with specific requests, for example.

Application Proxy

An application proxy is one of the most secure firewall types that can be deployed. The proxy sits between the protected network and the rest of the world. Every packet sent outbound is intercepted by the proxy, which initiates its own request and processes the response. If benign, the response is relayed back to the user. Thus, clients and servers never interact directly and the entire content of the packet can be inspected byte by byte if necessary. Even tricky applications such as Java code can be checked in a *Java sandbox* to assess effects before passing the applet on to a host.

Yet many organizations do anticipate employing application proxies today, and many that once did have abandoned them. Why? Well, proxies do not scale well and must handle twice the number of connections ("inside" and "outside") as all simultaneous users on the protected network. The obvious solution to all network load-related issues—multiple proxies—do not work well because there is no way to guarantee that a response is handled by the same proxy that handled the request.

The proxy also has trouble with proprietary or customized TCP/IP applications, where threats are not obvious or even well defined. But for limited use, such as protecting a Web site, an application proxy is a very attractive solution.

Stateful Inspection

A stateful inspection firewall is the choice for network protection today. Stateful inspection is really a very sophisticated version of a packet filter. All packets can be filtered, and almost every field and flag of the header at the IP and TCP layers can be inspected in a policy.

Moreover, this form of firewall understands the concept of the *state* of the session. So, when a client accesses a Web server, the firewall recognizes the response and can associate all of the packets sent in reply. This is a *dynamic* or *reflexive* firewall operation, and all reputable firewall products use this approach.

Of course, there are TCP/IP protocols, such as UDP or ICMP (and connectionless protocols in general), that have no defined "state" associated with them. Firewall vendors are free to be creative with how they handle these protocols, but the results have been remarkably consistent.

Many stateful inspection firewalls employ a form of application proxy for certain applications. For example, if the firewall is set to do URL filtering, an application proxy function can be coupled with this. This approach is often used with email today because many attachments are malicious either by accident or on purpose. However, as with any application proxy, this solution is difficult to scale or generalize (email attachment scanning is typically done apart from the firewall).

Today, some firewalls can also perform *deep inspection* of packet flows. These rules dig deep into the content of the packet, beyond the IP and TCP/UDP headers, and perform application-level scanning. If a firewall allows access to port 80 because there is a Web server on site, hackers will quickly find out that these packets pass right through the firewall. These firewalls not only protect Web sites, but can find email worms quickly and create regular expression (regex) rules to keep them from spreading. The general architecture of a stateful inspection firewall implemented as specialized hardware and software (an appliance) is shown in Figure 28.2.

An example of this architecture is the firewall product from Juniper Networks Security Products. It had been developed from the start with performance in mind, and runs an integrated security application to provide VPN, firewall, denial-of-service countermeasures, and traffic management.

The operating system is a specialized real-time OS that can preallocate memory to speed up task execution and help maintain a given rate of service. And in contrast

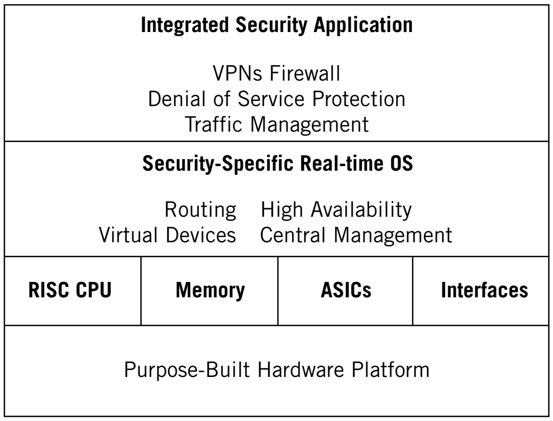

Integrated Security Application
VPNs Firewall
Denial of Service Protection
Traffic Management
Security-Specific Real-time OS
Routing
High Availability
Virtual Devices
Central Management
RISC CPU
Memory
ASICs
Interfaces
Purpose-Built Hardware Platform

FIGURE 28.2

Firewall appliance general architecture, showing how special hardware and software is used.

to packages built on an open-source Unix-based OS no one can review the source code looking for vulnerabilities. The OS is not distributed as widely as popular proprietary packages, and can support routing and virtual device multiplication—along with central management and high availability. (Larger firewalls pretty much have to support virtual devices, so this is really making a virtue out of a necessity.) The hardware is RISC based, with very fast memory (SDRAM) and ASICs—all designed to keep up with the interfaces' traffic flows.

DMZ

The biggest question facing firewall deployment is how to place the device to best protect publicly accessible servers. Cost and number of firewalls are related to decisions made in this area.

The answer to this location question usually involves the construction of a network DMZ ("demilitarized zone," another term like many others in the security field borrowed from the military). The DMZ is most useful when site protection is not absolute—that is, when it is not possible to deny *all* probes into the site from outside on the Internet (such as when a Web server or FTP server is available for general use). Without this requirement, the position of the firewall is almost always simply behind the router (as shown in Figure 28.3).

Even without a DMZ, it is possible to protect servers that require general Internet access. However, this protection is usually placed on the server itself, which then becomes a *bastion host*, which is still an untrusted host from the viewpoint of the internal network. A bastion host and firewall are shown in Figure 28.4.

It might sound odd that the bastion host, which might be the public Web server for the organization, needs a firewall to protect the internal network from the bastion host itself. But this is absolutely essential, and the bastion host should never be considered part of the internal network. Otherwise, if this host were compromised, the entire internal network would be at risk. For this reason, the bastion host in this configuration is not a good candidate for an e-commerce Web site or the endpoint of a VPN.

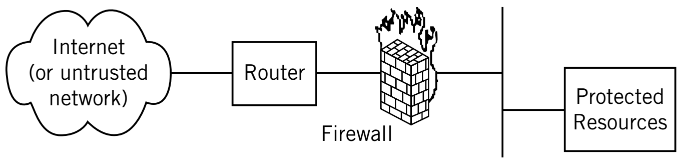


FIGURE 28.3

A single firewall positioned between router and LAN.

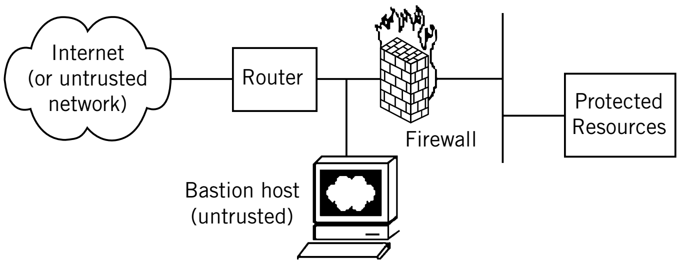


FIGURE 28.4

A firewall with bastion host between router and firewall (and therefore untrusted).

The DMZ concept has the ability to offer multiple types of protection—all in a flexible, scalable, and robust package. (DMZs can be designed with failover capabilities as well.) DMZs can be constructed with one or two firewalls, and two are better for security purposes.

With one firewall, the bastion host is reached only through the firewall itself, usually on a separate interface. The firewall can screen outside traffic (a "screened subnet"), perhaps allowing only access to port 80 for a Web server. Nothing is allowed in, of course, except in reply to an internal query (and even that is typically allowed only from specific hosts or on certain ports). This arrangement is shown in Figure 28.5.

The dual-firewall DMZ is the most sophisticated arrangement. There are both inner and outer firewalls, and the LAN between them is a true DMZ. Multiple servers, such as an anonymous FTP download server and a public Web server, can be protected in many ways. These devices can still be bastion hosts, but the protection on the DMZ servers

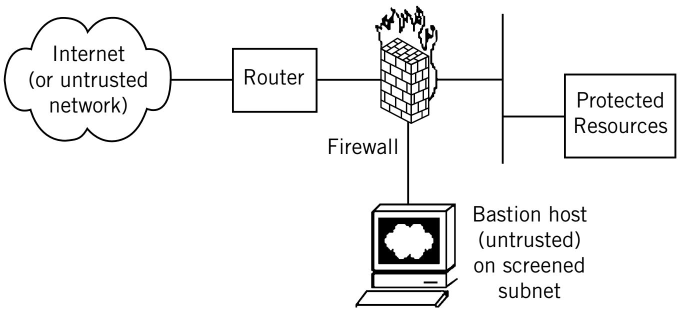


FIGURE 28.5

Firewall with bastion host and DMZ. Note the bastion host relation to the firewall.

themselves can be minimal because they all have the full protection of a firewall in whatever direction the traffic comes from or goes to. The dual-firewall DMZ is shown in Figure 28.6. The characteristics of these four basic firewall positions are compared in Table 28.1.

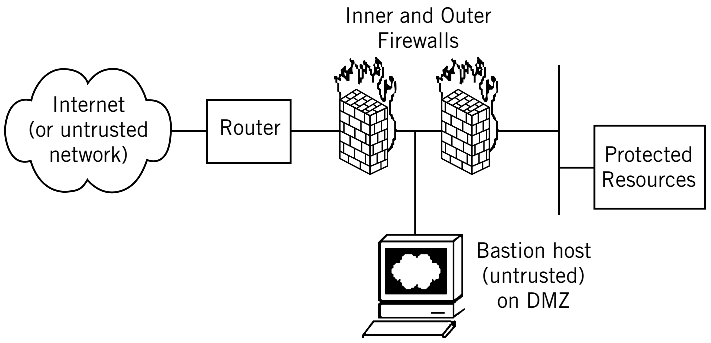

Inner and Outer Firewalls
Internet (or untrusted network)
Router
Protected Resources
Bastion host (untrusted) on DMZ

FIGURE 28.6

Dual firewalls with DMZ, showing how the bastion host is positioned on the DMZ.

Table 28.1 Advantages and Disadvantages of the Basic Firewall Designs

Type	Advantages	Disadvantages	Good for…
Single firewall	Inexpensive, easy to configure and maintain	Low security level, difficult to scale	Home or small office, no servers
Single firewall and bastion host	Lower cost than most alternatives	Bastion host vulner-able, difficult to scale	Small business with static content
Single firewall with screened subnet	Protects both local network and bastion host to some extent	Single point of failure, uses public addresses in some cases	Networks that need protected access to bastion host
Dual firewall and DMZ	Best control and very robust, scales nicely	More hardware and software, more work	Larger organizations

QUESTIONS FOR READERS

The filter listing that follows shows some of the concepts discussed in this chapter and can be used to answer the following questions.

```
set firewall family inet filter TEST term A from address 10.10.11.0/24;
set firewall family inet filter TEST term A from address 10.10.12.0/24;
set firewall family inet filter TEST term A from protocol [ udp tcp ];
set firewall family inet filter TEST term A from port [ 20 21 22 ];
set firewall family inet filter TEST term A then log;
set firewall family inet filter TEST term A then reject;
```

1. In the listing, which IP address will be selected out of all packets seen by the filter?
2. Which transport layer protocols will be selected by the filter?
3. Which applications are selected based on the port numbers given?
4. Will a log be kept of the selected packets?
5. Will the sender receive any notice that the packets have been blocked by a firewall filter?