CHAPTER 9

FIREWALLS

Ever since Cheswick and Bellovin wrote their epic book about building firewalls and tracking a wily hacker named Berferd, the thought of putting a web server (or any computer for that matter) on the Internet without installing a firewall in front of or on it has been considered suicidal. Equally as suicidal has been the frequent decision to throw firewall duties onto the network or, even worse, the system administrator's lap. Although these folks may understand the technical implications of a firewall, they don't live and breathe security and understand the mentality and techniques of the hacker (at least until they read this book a couple times). As a result, firewalls can be riddled with misconfigurations, allowing attackers to break into your network and cause you severe migraines. Given the proliferation of web-based attacks (as discussed in earlier chapters), firewalls have become nothing more than a speed bump on the information superhighway.

FIREWALL LANDSCAPE

Two types of firewalls dominate the market today: application proxies and packet-filtering gateways (and some hybrid combination of both). Although application proxies are widely considered more secure than packet-filtering gateways, their restrictive nature and performance limitations have constrained their adoption to primarily internal company traffic going out rather than traffic inbound to a company's web server or DMZ. On the other hand, packet-filtering gateways, or the more sophisticated *stateful* packet-filtering gateways, can be found in many larger organizations with high-performance inbound and outbound traffic requirements.

Firewalls have protected countless networks from prying eyes and malicious vandals—but they are far from a security panacea. Security vulnerabilities are discovered every year with just about every firewall on the market. What's worse, most firewalls are often misconfigured, unmaintained, and unmonitored, turning them into electronic doorstops (holding the gates wide open).

Make no mistake, a well-designed, -configured, and -maintained firewall is nearly impenetrable. Most skilled attackers know this. They will simply work around the firewall by exploiting trust relationships and weakest-link security vulnerabilities, or they will avoid it entirely by attacking through a VPN or dial-up account. Bottom line: Most attackers make every effort to work around a strong firewall. The goal here is to make your firewall strong.

As firewall administrators, we know the importance of understanding your enemy. Knowing the first few steps an attacker will perform to bypass your firewalls will take you a long way in detecting and reacting to an attack. In this chapter, we'll walk you through the typical techniques used today to discover and enumerate your firewalls, and we'll discuss a few ways attackers attempt to bypass them. With each technique, we'll discuss how you can detect and prevent attacks.

FIREWALL IDENTIFICATION

Almost every firewall will give off a unique electronic "scent." That is, with a little port scanning, firewalking, and banner grabbing, attackers can effectively determine the type, version, and rules of almost every firewall on the network. Why is this identification important? Because once an attacker has mapped out your firewalls, he can begin to understand their weaknesses and attempt to exploit them.

Direct Scanning: The Noisy Technique

Popularity:	10
Simplicity:	8
Impact:	2
Risk Rating:	***7***

The easiest way to look for your firewalls is by port-scanning specific default ports (as you learned in Chapter 2). Some firewalls on the market will uniquely identify themselves using simple port scans—you just need to know what to look for. For example, Check Point's FireWall-1 listens on TCP ports 256, 257, 258, and 259 (with Check Point NG listening on TCP ports 18210, 18211, 18186, 18190, 18191, and 18192 as well), and Microsoft's Proxy Server usually listens on TCP ports 1080 and 1745. With this knowledge, you'll find searching for these types of firewalls trivial with a port scanner such as ScanLine from Foundstone:

```
sl -pvh -t 23,80 68.4.190.1-254
```

NOTE Using the `--p` switch in ScanLine disables ICMP pinging before scanning. This is important because most firewalls do not respond to ICMP echo requests.

Both the dimwitted and the bold attacker will perform broad scans of your network in this manner, searching for these firewalls and looking for any chink in your perimeter armor. But the more dangerous attackers will comb your perimeter as stealthily as possible. Attackers can employ numerous techniques to fall under your radar, including randomizing pings, target ports (`-z`), target addresses (`-z`), and source ports (`-g`), as well as performing distributed source scans (meaning an attacker can use multiple computers on the Internet, each taking a small portion of the scanning targets).

If you think your intrusion detection system (IDS) will detect these more dangerous attackers, think again. Most IDSs come configured by default to hear only the noisiest or most clumsy port scans. Unless you highly sensitize your IDS and fine-tune your detection signatures, most of these sophisticated attacks will go completely unnoticed. You can produce such randomized scans by using the Perl scripts supplied on this book's companion website (http://www.osborne.com/he5).

Direct Scanning Countermeasures

Firewall scanning countermeasures in many ways mirror those discussed in Chapter 2, the scanning chapter. You'll need to either block these types of scans at your border routers or use some sort of intrusion-detection tool—either freeware or commercial. Even then, however, single port scans will not be picked up by default in most IDSs, so you'll need to tweak the system's sensitivity before detection can be relied on.

Detection To accurately detect the port scans using randomization, you'll need to fine-tune each of your port-scanning detection signatures. Refer to your IDS vendor's documentation for the details.

If you are using FireWall-1 for UNIX, you can use Lance Spitzner's utility for FireWall-1 port scan detection (http://www.enteract.com/~lspitz/intrusion.html). As covered in Chapter 2, his alert.sh script will configure Check Point to detect and monitor port scans and run a user-defined alert when triggered.

If you are using Linux firewalling, better known as netfilter/iptables, you have a plethora of detection tools available that will help identify those noisy attackers. One such tool includes IPPL, which is a daemon that runs in the background and will alert you to set specific parameters for logging suspicions packets. IPPL can be found at http://pltplp.net/ippl.

Prevention To prevent firewall port scans from the Internet, you'll need to block these ports on routers in front of the firewalls. If these devices are managed by your ISP, you'll need to contact them to perform the blocking. If you manage these devices yourself, you can use the following Cisco ACLs to explicitly block the scans discussed earlier:

```
access-list 101 deny tcp any any eq 256 log  ! Block Firewall-1 scans
access-list 101 deny tcp any any eq 257 log  ! Block Firewall-1 scans
access-list 101 deny tcp any any eq 258 log  ! Block Firewall-1 scans
access-list 101 deny tcp any any eq 259 log  ! Block Firewall-1 scans
access-list 101 deny tcp any any eq 1080 log  ! Block Socks scans
access-list 101 deny tcp any any eq 1745 log  ! Block Winsock scans
```

NOTE If you block Check Point's ports (256–259) at your border routers, you will be unable to manage the firewall from the Internet.

NOTE Your Cisco administrator should be able to apply the foregoing rules to the firewall without trouble. Simply enter "enable" mode and type the preceding lines one at a time. Then exit enable mode and type **write** to write them to the configuration file.

Also, all your routers should have a cleanup rule (if they don't deny packets by default), which will have the same effect as specifying the preceding explicit deny operations. A typical "deny all" rule looks something like this:

```
access-list 101 deny ip any any log  ! Deny and log any packet that got
through our ACLs above
```

As with any countermeasure, be sure to refer to your specific documentation and installation requirements before applying any recommendations.

Route Tracing

Popularity:	10
Simplicity:	8
Impact:	2
Risk Rating:	*7*

A more quiet and subtle way of finding firewalls on a network is to use traceroute. You can use UNIX's traceroute or Windows' tracert.exe tools to find each hop along the path to the target and to do some deduction. Linux's traceroute has the `–I` option, which performs traceroutes by sending ICMP packets, as opposed to its default UDP packet technique:

```
[sm]$ traceroute -I 192.168.51.100
traceroute to 192.168.51.101 (192.168.51.100), 30 hops max, 40 byte packets
 1  attack-gw (192.168.50.21)  5.801 ms  5.105 ms  5.445 ms
 2  gw1.smallisp.net (192.168.51.1)
 3  gw2.smallisp.net (192.168.52.2)
....

13  hssi.bigisp.net (10.55.201.2)
14  serial1.bigisp.net (10.55.202.1)
15  192.168.51.101 (192.168.51.100)
```

In the preceding output, chances are good that the system (10.55.202.1) just before the target (192.168.51.100) is the firewall, but we don't know for sure yet. We'll need to do a little more digging.

The preceding example is great if the routers between you and your target servers respond to ICMP time to live (TTL) expired packets. But some routers and firewalls are set up not to return these TTL expired packets (from both ICMP and UDP packets). In this case, the deduction is less scientific. All you can do is run traceroute, see which hop responds last, and deduce that this is either a full-blown firewall or at least the first router in the path that begins to block TTL expired packets. For example, here ICMP is being blocked to its destination, and there's no response from routers beyond `client-gw.smallisp.net`:

```
1 stoneface (192.168.10.33) 12.640 ms 8.367 ms
2 gw1.localisp.net (172.31.10.1) 214.582 ms 197.992 ms
3 gw2.localisp.net (172.31.10.2) 206.627 ms 38.931 ms
4 ds1.localisp.net (172.31.12.254) 47.167 ms 52.640 ms
...
```

```
14 ATM6.LAX2.BIGISP.NET (10.50.2.1) 250.030 ms 391.716 ms
15 ATM7.SDG.BIGISP.NET (10.50.2.5) 234.668 ms 384.525 ms
16 client-gw.smallisp.net (10.50.3.250)  244.065 ms !X * *
17 * * *
18 * * *
```

Route Tracing Countermeasure

The fix for traceroute information leakage is to restrict as many firewalls and routers from responding to ICMP TTL expired packets as possible. This is not always under your control because many of your routers are probably controlled by your ISP, but attempts should be made to motivate your ISP into action.

Detection To detect standard traceroutes on your border, you'll need to monitor for ICMP and UDP packets with a TTL value of 1.

Prevention To prevent traceroutes from being run over your border, you can configure your routers not to respond with TTL EXPIRED messages when they receive a packet with the TTL value of 0 or 1. The following ACL will work with Cisco routers:

```
access-list 101 deny ip any any 11 0 ! ttl-exceeded
```

Ideally, you'll want to block all unnecessary UDP traffic at your border routers altogether.

Banner Grabbing

Popularity:	10
Simplicity:	9
Impact:	3
Risk Rating:	***7***

Scanning for firewall ports is helpful in locating firewalls, but most firewalls do not listen on default ports like Check Point and Microsoft, so detection has to be deduced. You learned in Chapter 3 how to discover running application names and versions by connecting to the services found open and reading their banners. Firewall detection can be made in much the same way. Many popular firewalls will announce their presence by your simply connecting to them (of course, you will first have to find an open port to connect to by port scanning; refer to Chapter 2). For example, many proxy firewalls will announce their function as a firewall, and some will advertise their type and version. For instance, when we connect to a machine believed to be a firewall with netcat on port 21 (FTP), we see some interesting information:

```
C:\>nc -v -n 192.168.51.129 21
(UNKNOWN) [192.168.51.129] 21 (?) open
220 Secure Gateway FTP server ready.
```

The "Secure Gateway FTP server ready" banner is the telltale sign of an old Eagle Raptor box. Connecting further to port 23 (telnet) confirms the firewall brand name "Eagle," as shown here:

```
C:\>nc -v -n 192.168.51.129 23
(UNKNOWN) [192.168.51.129] 23 (?) open
Eagle Secure Gateway.
Hostname:
```

Finally, if you're still not convinced that our target host is a firewall, you can `netcat` to port 25 (SMTP), and it will tell you it is

```
C:\>nc -v -n 192.168.51.129 25
(UNKNOWN) [192.168.51.129] 25 (?) open
421 fw3.example.com Sorry, the firewall does not provide mail service to you.
```

As you can see in the preceding examples, banner information can provide attackers with valuable information in identifying your firewalls. Using this information, they can exploit well-known vulnerabilities or common misconfigurations.

Banner-Grabbing Countermeasure

The fix for this information leakage vulnerability is either to eliminate the open port on your firewall (this should not be allowed generally) or to limit the banner information given out. If you must leave the ports open on the external interface of your firewall, you can usually change the banner to read a legal warning reminding the offender that all attempts to connect will be logged. The specifics of changing default banners will depend largely on your specific firewall, so you'll need to check with your firewall vendor.

Prevention To prevent an attacker from gaining too much information about your firewalls from the banners they advertise, you can often alter the banner configuration files. Specific recommendations will depend on your firewall vendor.

Advanced Firewall Discovery

If port scanning for firewalls directly, tracing the path, and banner grabbing haven't proved successful, attackers will take firewall enumeration to the next level. Firewalls and their ACL rules can be deduced by probing targets and noticing the paths taken (or not taken) to get there.

Simple Deduction with nmap

Popularity:	4
Simplicity:	6
Impact:	7
Risk Rating:	**6**

nmap is a great tool for discovering firewall information, and we use it constantly. When nmap scans a host, it doesn't just tell you which ports are open or closed, it tells you which ports are being blocked. The amount (or lack) of information received from a port scan can tell a lot about the configuration of the firewall.

A filtered port in nmap signifies one of three things:

- No SYN/ACK packet was received.
- No RST/ACK packet was received.
- An ICMP type 3 message (Destination Unreachable) with code 13 (Communication Administratively Prohibited – [RFC 1812]) was received.

nmap will pull all three of these conditions together and report it as a "filtered" port. For example, when scanning www.example.com, we receive two ICMP packets telling us that its firewall blocks ports 23 and 111 from our particular system:

```
Starting nmap V. 2.08 by Fyodor (fyodor@dhp.com, www.insecure.org/nmap/)
Initiating TCP connect() scan against  (192.168.51.100)
Adding TCP port 53 (state Open).
Adding TCP port 111 (state Firewalled).
Adding TCP port 80 (state Open).
Adding TCP port 23 (state Firewalled).
Interesting ports on  (192.168.51.100):
Port    State       Protocol  Service
23      filtered    tcp       telnet
53      open         tcp        domain
80      open         tcp        http
111     filtered    tcp       sunrpc
```

The "Firewalled" state in the verbose preceding output results from receiving an ICMP type 3, code 13 (Admin Prohibited Filter) packet, as seen in the tcpdump output:

```
23:14:01.229743 10.55.2.1 > 172.29.11.207: icmp: host 172.32.12.4
Unreachable - admin prohibited filter
23:14:01.979743 10.55.2.1 > 172.29.11.207: icmp: host 172.32.12.4
Unreachable - admin prohibited filter
```

How does nmap associate these packets with the original ones, especially when they are only a few in a sea of packets whizzing by on the network? The ICMP packet sent back to the scanning machine houses all the data necessary to understand what's happening. The port being blocked is the 1-byte portion in the ICMP header at byte 0x41 (1 byte), and the filtering firewall sending the message is in the IP portion of the packet at byte 0x1b (4 bytes).

Finally, an nmap "unfiltered" port appears only when you scan a number of ports and receive an RST/ACK packet back. In the "unfiltered" state, either our scan is getting through the firewall and the target system is telling us that it's not listening on that port, or the firewall is responding for the target and spoofing its IP address with the RST/ACK flag set. For example, our scan of a local system gives us two unfiltered ports when it receives two RST/ACK packets from the same host. This event can also occur with some firewalls, such as Check Point (with the REJECT rule) when it responds for the target by sending back an RST/ACK packet and spoofing the target's source IP address:

```
[root]# nmap -sS -p1-300 172.18.20.55

Starting nmap V. 2.08 by Fyodor (fyodor@dhp.com, www.insecure.org/nmap/)
Interesting ports on  (172.18.20.55):
(Not showing ports in state: filtered)

Port    State       Protocol  Service
7       unfiltered  tcp       echo
53      unfiltered  tcp       domain
256     open         tcp        rap
257     open         tcp        set
258     open         tcp        yak-chat

Nmap run completed -- 1 IP address (1 host up) scanned in 15 seconds
```

The associated tcpdump packet trace shows the RST/ACK packets received:

```
21:26:22.742482 172.18.20.55.258 > 172.29.11.207.39667: S
415920470:1415920470(0) ack 3963453111 win 9112 <mss 536> (DF)
(ttl 254, id 50438)
21:26:23.282482 172.18.20.55.53 > 172.29.11.207.39667:
R 0:0(0) ack 3963453111 win 0 (DF) (ttl 44, id 50439)
21:26:24.362482 172.18.20.55.257 > 172.29.11.207.39667:
S 1416174328:1416174328(0) ack 3963453111 win 9112 <mss 536>
(DF) (ttl 254, id 50440)
21:26:26.282482 172.18.20.55.7 > 172.29.11.207.39667:
R 0:0(0) ack 3963453111 win 0 (DF) (ttl 44, id 50441)
```

Simple Deduction with nmap Countermeasures

There are two types of simple deduction nmap countermeasures, both of which are discussed below.

Detection The detection mechanisms for nmap scans are the same as those detailed in Chapter 2. We recommend customizing those detection mechanisms to extract just the scans that enumerate your firewalls.

Prevention To prevent attackers from enumerating router and firewall ACLs through the "ICMP Admin Prohibited Filter" technique, you can disable your router's ability to respond with the ICMP type 13 packet. On Cisco you can do this by blocking the device from responding to IP unreachable messages, like so:

```
no ip unreachables
```

Port Identification

Popularity:	5
Simplicity:	6
Impact:	7
Risk Rating:	***6***

Some firewalls have a unique footprint that is displayed as a series of numbers that are distinguishable from other firewalls. For example, Check Point will display a series of numbers when you connect to its SNMP management port, TCP 257. Although the mere presence of ports 256–259 on a system is usually a sufficient indicator for the presence of Check Point's FireWall-1, the following test will confirm it:

```
[root]# nc -v -n 192.168.51.1 257 (UNKNOWN) [192.168.51.1] 257 (?) open
        30000003

[root]# nc -v -n 172.29.11.191 257
(UNKNOWN) [172.29.11.191] 257 (?) open
        31000000
```

Port Identification Countermeasures

You can prevent connections to TCP port 257 (or any other Check Point port) by blocking them at your upstream routers. A simple Cisco ACL like the following can explicitly deny an attacker's attempt:

```
access-list 101 deny tcp any any eq 257 log  ! Block Firewall-1 scans
```

SCANNING THROUGH FIREWALLS

Don't worry, this section is not going to give the script kiddies some magical technique to render your firewalls ineffective. Instead, we will cover a number of techniques for dancing around firewalls and gather some critical information about the various paths through and around them.

Raw Packet Transmissions

Popularity:	3
Simplicity:	4
Impact:	8
Risk Rating:	**5**

hping, by Salvatore Sanfilippo (http://www.hping.org), works by sending ICMP, TCP (default mode), or UDP packets to a destination system/port and reporting the packets it gets back. hping returns a variety of responses depending on numerous conditions. Each packet in part or in whole can provide a fairly clear picture of the firewall's access controls. For example, by using hping, we can discover open, blocked, dropped, and rejected packets.

In the following example, hping reports that port 80 is open and ready to receive a connection. We know this because it received a packet with the SA flag set (a SYN/ACK packet).

```
[root]# hping2 192.168.0.2 -S -p 80 -n
HPING www.example.com (eth0 172.16.1.20): S set, 40 data bytes
60 bytes from 192.168.0.2: flags=SA seq=0 ttl=242 id=65121 win=64240
time=144.4 ms
```

Now we know an open port exists on our target, but we don't know where the firewall is yet. In our next example, hping reports receiving an ICMP unreachable type 13 packet from 192.168.70.2. In Chapter 2, you learned that ICMP type 13 is an ICMP Admin Prohibited Filter packet, which is usually sent from a packet-filtering router such as Cisco's IOS.

```
[root]# hping2 192.168.0.2 -S -p 23 -n
HPING 192.168.0.2 (eth0 172.16.1.20): S set, 40 data bytes
ICMP Unreachable type 13 from 192.168.0.1
```

Now it is confirmed: 192.168.70.2 is most likely our firewall, and we know it is explicitly blocking port 23 to our target. In other words, if the system is a Cisco router, it probably has a line like the following in its config file:

```
access-list 101 deny tcp any any 23 ! telnet
```

In the next example, we receive an RST/ACK packet back, signifying one of two things: either that the packet got through the firewall and the host is not listening to that port, or that the firewall rejected the packet (such is the case with Check Point's reject rule).

```
[root]# hping2 192.168.0.2 -S -p 22 -n
HPING 192.168.0.2 (eth0 172.16.1.20): S set, 40 data bytes
60 bytes from 192.168.0.2: flags=RA seq=0 ttl=59 id=0 win=0 time=0.3 ms
```

Because we received the ICMP type 13 packet earlier, we can deduce that the firewall (192.168.0.1) is allowing our packet through, but the host is just not listening on that port.

If the firewall you're scanning through is Check Point, hping will report the source IP address of the target, but the packet is really being sent from the external NIC of the Check Point firewall. The tricky thing about Check Point is that it will respond for its internal systems, sending a response and spoofing the target's address. When attackers hit one of these conditions over the Internet, however, they'll never know the difference, because the MAC address will never reach their machine (to tip them off).

Finally, when a firewall is blocking packets altogether to a port, you'll often receive nothing back:

```
[root]# hping 192.168.50.3 -S -p 22 -n
HPING 192.168.50.3 (eth0 192.168.50.3): S set, 40 data bytes
```

In this scenario, the hping result can have two meanings: the packet couldn't reach the destination and was lost on the wire, or the target host was not turned off (it may not exist) or, more likely, a device (probably our firewall, 192.168.70.2) dropped the packet on the floor as part of its ACL rules.

Raw Packet Transmissions Countermeasure

Preventing an hping attack is difficult. Your best bet is to simply block ICMP type 13 messages (as discussed in the preceding nmap scanning prevention section).

Firewalk

Popularity:	*3*
Simplicity:	*3*
Impact:	*8*
Risk Rating:	***4***

Firewalk (http://www.packetfactory.net/projects/firewalk) is a nifty little tool that, like a port scanner, will discover ports open behind a firewall. Written by Mike Schiffman (a.k.a. Route) and Dave Goldsmith, the utility will scan a host downstream from a firewall and report back the rules allowed to that host, without actually touching the target system.

Firewalk works by constructing packets with an IP TTL calculated to expire one hop past the firewall. The theory is that if the packet is allowed by the firewall, it will be allowed to pass and will expire as expected, eliciting an "ICMP TTL expired in transit" message. On the other hand, if the packet is blocked by the firewall's ACL, it will be dropped, and either no response will be sent or an ICMP type 13 Admin Prohibited Filter packet will be sent. The following scenario assumes that ports 135 through 138 and 140 are open behind the firewall.

```
[root]# firewalk -pTCP -S135-140 10.22.3.1
192.168.1.1
Ramping up hopcounts to binding host...
probe:  1  TTL:  1  port 33434:  expired from [exposed.example.com]
probe:  2  TTL:  2  port 33434:  expired from [rtr.isp.net]
probe:  3  TTL:  3  port 33434:  Bound scan at 3 hops [rtr.isp.net]
port 135: open
port 136: open
port 137: open
port 138: open
port 139:  *
port 140: open
```

The only problem we've seen when using Firewalk is that it can be highly unpredictable, because some firewalls will detect that the packet expires before checking their ACLs and send back an "ICMP TTL expired" packet anyway. As a result, Firewalk often assumes that all ports are open.

Firewalk Countermeasure

You can block "ICMP TTL expired" packets at the external interface level, but this may negatively affect its performance, because legitimate clients connecting will never know what happened to their connection.

Source Port Scanning

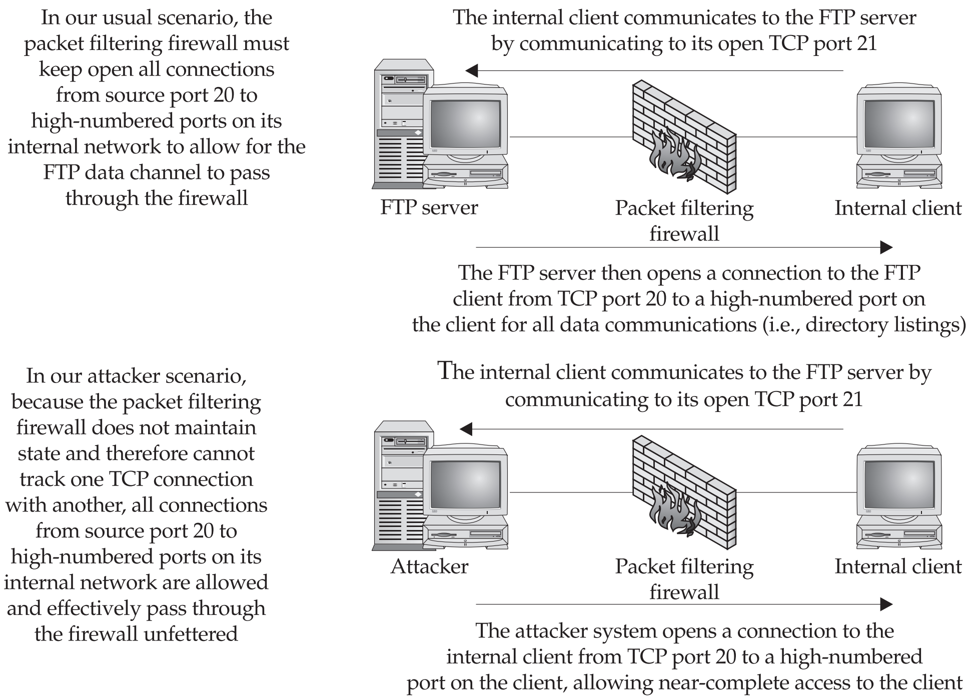
Traditional packet-filtering firewalls such as Cisco's IOS have one major drawback: They don't keep state! For many of you that seems obvious, right? But think about it for a moment. If the firewall cannot maintain state, it cannot tell whether the connection began outside or inside the firewall. In other words, it cannot completely control some transmissions. As a result, we can set our source port to typically allowed ports such as TCP 53 (zone transfers) and TCP 20 (FTP data) and then scan (or attack) to our heart's content.

To discover whether a firewall allows scans through a source port of 20 (FTP-data channel, for example), you can use nmap's `-g` feature:

```
nmap -sS -P0 -g 20 -p 139 10.1.1.1
```

NOTE You'll need to use the SYN or half-scan technique when using the static source port feature of nmap.

If ports come back as open, you will likely have a vulnerable firewall in your midst. To understand the scenario better, here's a diagram that details how the attack works:


In our usual scenario, the packet filtering firewall must keep open all connections from source port 20 to high-numbered ports on its internal network to allow for the FTP data channel to pass through the firewall
The internal client communicates to the FTP server by communicating to its open TCP port 21
FTP server
Packet filtering firewall
Internal client
The FTP server then opens a connection to the FTP client from TCP port 20 to a high-numbered port on the client for all data communications (i.e., directory listings)
In our attacker scenario, because the packet filtering firewall does not maintain state and therefore cannot track one TCP connection with another, all connections from source port 20 to high-numbered ports on its internal network are allowed and effectively pass through the firewall unfettered
The internal client communicates to the FTP server by communicating to its open TCP port 21
Attacker
Packet filtering firewall
Internal client
The attacker system opens a connection to the internal client from TCP port 20 to a high-numbered port on the client, allowing near-complete access to the client

You can now take advantage of the discovery that a firewall is not maintaining the state of its firewalled connections by launching attacks against vulnerable systems behind the firewall. Using a modified port redirector such as Fpipe from Foundstone, you can set the source port to 20 and then run exploit after exploit through the firewall. In addition, you can use the ever-popular netcat (http://netcat.sourceforge.net) to set your source port to 20 and then connect to open ports behind the firewall. Use the `-s` option to set your source port. As we have discussed in earlier chapters, netcat is your friend!

Source Port Scanning Countermeasure

The solutions to this vulnerability are simple but not all that glamorous. You'll need to either disable any communications that require more than one port combination (such as traditional FTP), switch to a stateful or application-based proxy firewall that keeps better control of incoming and outgoing connections, or employ firewall-friendly applications such as Passive FTP that do not violate the firewall rules.

PACKET FILTERING

Packet-filtering firewalls (including stateful firewalls) such as Check Point's FireWall-1, Cisco's PIX, and Cisco's IOS (yes, Cisco IOS can be considered a firewall) depend on access control lists (ACLs), or rules to determine whether traffic is authorized to pass into or out of the internal network. For the most part, these ACLs are well devised and difficult to get around. But every so often, you'll come across a firewall with liberal ACLs that allow some packets to pass.

Liberal ACLs

Popularity:	*8*
Simplicity:	*2*
Impact:	*2*
Risk Rating:	***4***

Liberal access control lists (ACLs) frequent more firewalls than we care to mention. Consider the case where an organization may want to allow its ISP to perform zone transfers. A liberal ACL such as "Allow all activity from the TCP source port of 53" might be employed rather than "Allow activity from the ISP's DNS server with a TCP source port of 53 and a destination port of 53." The risk that such misconfigurations pose can be truly devastating, allowing a hacker to scan the entire network from the outside. Most of these attacks begin by an attacker scanning a host behind the firewall and spoofing its source as TCP port 53 (DNS).

Liberal ACLs Countermeasure

Make sure your firewall rules limit who can connect where. For example, if your ISP requires zone transfer capability, then be explicit about your rules. Require a source IP address and hard-code the destination IP address (your internal DNS server) in the rule you devise.

If you are using a Check Point firewall, you can use the following rule to restrict a source port of 53 (DNS) to only your ISP's DNS (for this example, your ISP's DNS is 192.168.66.2 and your internal DNS is 172.30.140.1):

Source	Destination	Service	Action	Track
192.168.66.2	172.30.140.1	domain-tcp	Accept	Short

Check Point Trickery

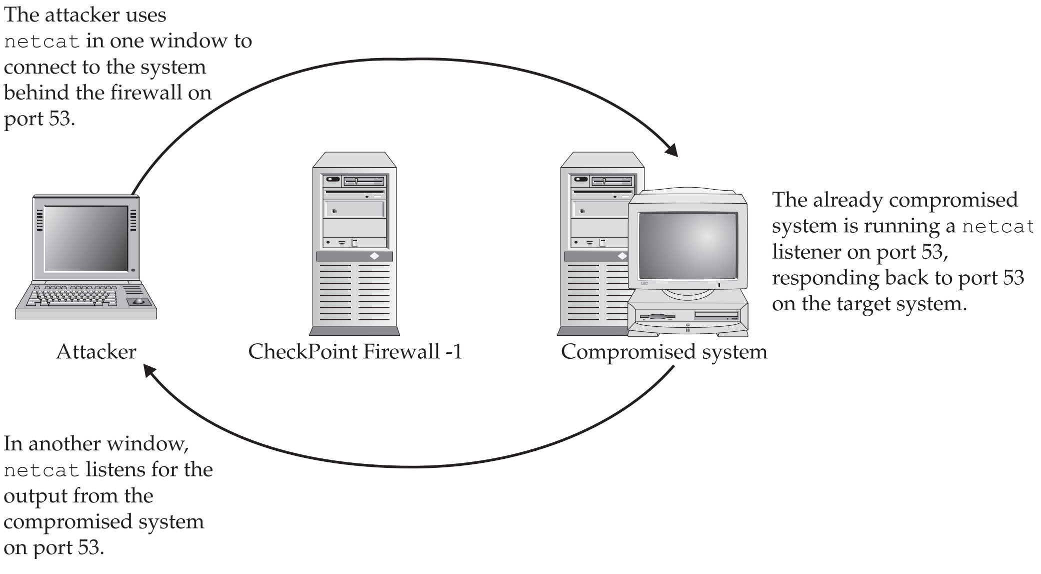
Popularity:	8
Simplicity:	2
Impact:	2
Risk Rating:	**4**

Check Point 3.0 and 4.0 provide ports open by default. DNS lookups (UDP 53), DNS zone transfers (TCP 53), and RIP (UDP 520) are allowed from *any* host to *any* host and are not logged. This sets up an interesting scenario once an internal system has been compromised.

You've already seen how easy it can be to identify a Check Point firewall. By using this new knowledge, attackers can effectively bypass the firewall rules. But there is a significant prerequisite to this attack. The attack only works once attackers have compromised a system behind the firewall or have tricked a user on a back-end system into executing a Trojan horse.

In either event, the end result is most likely a netcat listener on a compromised system inside your network. The netcat listener can either send back a shell or type commands that run locally on the remote system. These "back doors" will be discussed in detail in subsequent chapters, but a little description here may help you understand the problem.

As the following illustration shows, Check Point allows TCP port 53 through the firewall unlogged. When attackers set up a netcat listener on port 53 and shell back /bin/sh to their own machine, also listening on port 53, they will have a hole through your firewall to any system they've compromised.


The attacker uses netcat in one window to connect to the system behind the firewall on port 53.
The already compromised system is running a netcat listener on port 53, responding back to port 53 on the target system.
Attacker
CheckPoint Firewall -1
Compromised system
In another window, netcat listens for the output from the compromised system on port 53.

Check Point Trickery Countermeasure

Depending on your configuration needs, you can disable much of the traffic that is allowed by default. Be careful with this prevention fix, though, because it may disallow authorized traffic to flow through your firewall. Perform the following steps to restrict this access:

1. Within the Security Policy GUI, select Policy | Properties.
2. Clear the Accept check box of all functions that are unnecessary. For example, many sites do not need their users to perform DNS downloads. In this case, you can clear the Accept Domain Name Downloads option. The same technique can be used to disable RIP and DNS lookup traffic.
3. Create your own rule that allows DNS traffic from a specific authorized DNS server (as shown in the preceding "Liberal ACLs Countermeasure" section).

ICMP and UDP Tunneling

Popularity:	2
Simplicity:	1
Impact:	9
Risk Rating:	**4**

ICMP tunneling is the capability of wrapping real data in an ICMP header. Many routers and firewalls that allow ICMP ECHO, ICMP ECHO REPLY, and UDP packets through will be vulnerable to this attack. Much like the Check Point DNS vulnerability, the ICMP and UDP tunneling attack relies on an already compromised system behind the firewall.

Jeremy Rauch and Mike Schiffman put the tunneling concept to work and created the tools to exploit it: loki and lokid (the client and server). See http://www.phrack.org/phrack/51/P51-06 for the complete paper. Running the lokid server tool on a system behind a firewall allowing ICMP ECHO and ECHO REPLY enables attackers to run the client tool (loki), which wraps every command sent in ICMP ECHO packets to the server (lokid). The lokid tool will unwrap the commands, run the commands locally, and wrap the output of the commands in ICMP ECHO REPLY packets back to the attacker. Using this technique, attackers can completely bypass your firewall.

ICMP and UDP Tunneling Countermeasure

You can prevent this type of attack by disabling ICMP access through your firewall altogether or by providing granular access control on ICMP traffic. For example, the following Cisco ACL will disallow all ICMP traffic outside of the 172.29.10.0 subnet (the DMZ) for administrative purposes:

```
access-list 101 permit icmp any 172.29.10.0 0.255.255.255 8  ! echo
access-list 101 permit icmp any 172.29.10.0 0.255.255.255 0  ! echo-reply
access-list 102 deny   ip   any any log  ! deny and log all else
```

If your ISP tracks your system's uptime behind your firewall with ICMP pings (which we never recommend), then these ACLs will break their heartbeat function. Check with your ISP to find out whether they use ICMP pings to check your systems.

APPLICATION PROXY VULNERABILITIES

In general, application proxy vulnerabilities are few. Once you have secured the firewall itself and implemented solid proxy rules, you'll be hard pressed to bypass a proxy firewall. But never fear, misconfigurations are common.

Hostname: localhost

Popularity:	4
Simplicity:	2
Impact:	9
Risk Rating:	**5**

With some older UNIX proxies, it was easy to miss restricting local access. Despite authentication requirements for your users when accessing the Internet, it was possible for an internal user to gain local access on the firewall itself. Of course, this attack requires knowledge of a valid username and password on the firewall, but you'd be surprised how easy these are to guess sometimes. To check your proxy firewalls for this vulnerability, you can do the following when you receive this login screen:

```
C:\> nc -v -n 192.168.51.129 23
(UNKNOWN) [192.168.51.129] 23 (?) open
Eagle Secure Gateway.
Hostname:
```

1. Type in **localhost**.
2. Enter a known username and password (or guess a few).
3. If authentication works, you have local access on the firewall.
4. Run a local buffer overflow (such as rdist) or a similar exploit to gain root.

Hostname: Localhost Countermeasure

The fix for this misconfiguration depends largely on the specific firewall product. In general, you can provide a host-restriction rule that limits the access from a particular site. The ideal countermeasure is to not allow localhost logins.

Unauthenticated External Proxy Access

Popularity:	*8*
Simplicity:	*8*
Impact:	*4*
Risk Rating:	***7***

This scenario is more common with firewalls that employ transparent proxies, but we do see it from time to time. A firewall administrator will go to great lengths to secure the firewall and create strong access rules but then forget to block outside access. This risk is twofold: (1) an attacker can use your proxy server to hop all around the Internet, anonymously attacking web servers with web-based attacks such as CGI vulnerabilities and web fraud, and (2) an attacker can gain web access to your intranet. We've come across a firewall configured this way, and it allowed us to access the company's entire intranet.

You can check whether your firewall is vulnerable by changing your browser's proxy settings to point to the suspected proxy firewall. To do this in Netscape, perform the following steps:

1. Select Edit | Preferences.
2. Select the Advanced and Proxies subtrees.
3. Check the Manual Proxy Configuration button.
4. Select the View button.
5. Add the firewall in question in the HTTP address and select the port it is listening on. (This is usually 80, 81, 8000, or 8080 but will vary greatly; use nmap or a similar tool to scan for the correct port.)
6. Point your browser to your favorite website and note the status bar's activity.

If the browser's status bar displays the proxy server being accessed and the web page comes up, you probably have an unauthenticated proxy server.

Next, if you have the IP address of an internal website (whether its address is routable or not), you can try to access it in the same manner. You can sometimes get this internal IP address by viewing the HTTP source code. Web designers will often hard-code hostnames and IP addresses in the HREFs of web pages.

Unauthenticated External Proxy Access Countermeasure

The prevention for this vulnerability is to disallow proxy access from the external interface of the firewall. Because the technique for doing this is highly vendor dependent, you'll need to contact your firewall vendor for further information.

The network solution is to restrict incoming proxy traffic at your border routers. This can be easily accomplished with some tight ACLs on your routers.

WinGate Vulnerabilities

The popular Windows 95/NT/2000 proxy firewall WinGate (http://www.wingate.com) has been known to have a couple of vulnerabilities. Most of these stem from lax default parameters, including unauthenticated telnet, SOCKS, and Web. Although access to these services can be restricted by user (and interface), many simply install the product "as is" to get it up and running—forgetting about security.

Unauthenticated Browsing

Popularity:	*9*
Simplicity:	*9*
Impact:	*2*
Risk Rating:	***7***

Like many misconfigured proxies, certain WinGate versions (specifically 2.1d for Windows) allow outsiders to browse the Internet completely anonymously. This is important for attackers who target web server applications in particular, because they can hack to their heart's content with little risk of getting caught. As a general rule, you have little defense against web attacks, because all traffic is tunneled in TCP port 80 or encrypted in TCP port 443 (SSL).

To check whether your WinGate servers are vulnerable, follow these steps:

1. Attach to the Internet with an unfiltered connection (preferably dial-up).
2. Change your browser's configuration to point to a proxy server.
3. Specify the server and port in question.

Also vulnerable in a default configuration is the unauthenticated SOCKS proxy (TCP 1080). As with the open Web proxy (TCP 80), an attacker can browse the Internet, bouncing through these servers and remaining almost completely anonymous (especially if logging is turned off).

Unauthenticated Browsing Countermeasure

To prevent this vulnerability with WinGate, you can simply restrict the bindings of specific services. Perform the following steps on a multihomed system to limit where proxy services are offered:

1. Select the SOCKS or WWW Proxy Server properties.
2. Select the Bindings tab.
3. Check the Connections Will Be Accepted on the Following Interface Only button and then specify the internal interface of your WinGate server.

The Real Treat for the Attacker: Unauthenticated Telnet

Popularity:	9
Simplicity:	9
Impact:	6
Risk Rating:	8

Worse than anonymous web browsing is unauthenticated telnet access (one of the core utilities in the hacker's toolbox). By connecting to telnet on a misconfigured WinGate server, attackers can use your machines to hide their tracks and attack freely.

To search for vulnerable servers, perform the following steps:

1. Using telnet, attempt to connect to a server.
2. If you receive the following text, enter a site to connect to:

```
[root]#  telnet 172.29.11.191
Trying 172.29.11.191...
Connected to 172.29.11.191.
Escape character is '^]'.
Wingate> 10.50.21.5
```

3. If you see the new system's login prompt, you have a vulnerable server. Here's an example:

```
Connecting to host 10.50.21.5...Connected
SunOS 5.6
login:
```

Unauthenticated Telnet Countermeasure

The prevention technique for this vulnerability is similar to the "unauthenticated browsing" vulnerability mentioned earlier. Simply restrict the bindings of specific services in WinGate to resolve the problem. You can do this on a multihomed system by performing the following steps:

1. Select the Telnet Server properties.
2. Select the Bindings tab.
3. Check the Connections Will Be Accepted on the Following Interface Only button and then specify the internal interface of your WinGate server.

File Browsing

Popularity:	*9*
Simplicity:	*9*
Impact:	*9*
Risk Rating:	***9***

Default WinGate 3.0 installations allow anyone to view files on the system through their management port (8010). To check whether your system is vulnerable, run all the following:

```
http://192.168.51.101:8010/c:/
http://192.168.51.101:8010//
http://192.168.51.101:8010/..../
```

If your system is vulnerable, you'll be able to browse each file in the directory and navigate in and out of directories at will. This is dangerous because some applications store usernames and passwords in the clear. For example, if you use Computer Associates' Remotely Possible or ControlIT to remotely control your servers, the usernames and passwords for authentication either are stored in the clear or are obfuscated by a simple substitution cipher (see Chapter 13).

File Browsing Countermeasure

Upgrade to the current version of WinGate at http://www.wingate.com.

SUMMARY

In reality, a well-configured firewall can be incredibly difficult to bypass. But using information-gathering tools such as traceroute, hping, and nmap, attackers can discover (or at least deduce) access paths through your router and firewall as well as the type of firewall you are using. Many of the current vulnerabilities are due to misconfigurations in the firewall or a lack of administrative monitoring, but either way the effect can lead to a catastrophic attack if exploited.

Some specific weaknesses exist in both proxies and packet-filtering firewalls, including unauthenticated Web, telnet, and localhost logins. For the most part, specific countermeasures can be put in place to prevent the exploitation of this vulnerability. In some cases, only detection is possible.

Many believe that the inevitable future of firewalls will be a hybrid of both application proxy and stateful packet-filtering technology and will provide some techniques for limiting misconfigurations. Currently, many of the high-end firewalls include deep packet inspection capabilities, which allow the firewall to act in a stateful manner for speed, but provide proxy-like security by being able to peer into the actual packets looking for malicious traffic at the application level.

Finally, we always get what firewalls we use. We have tried the full gamut of freeware and commercial firewalls. Many are excellent. One firewall that has stood out for our needs is Astaro. Astaro is a Linux-based firewall with a plethora of features, including antispam, intrusion detection via Snort, antivirus, and several built in proxies (HTTP, DNS, and so on). It can be installed easily and provides excellent projection. You could spend hours trying to configure all the open-source software yourself, or you could get Astaro for free (for home users) at http://www.astaro.com/firewall_network_security/buy. Whatever firewall you decide to use, always make sure you configure and test it before deployment.