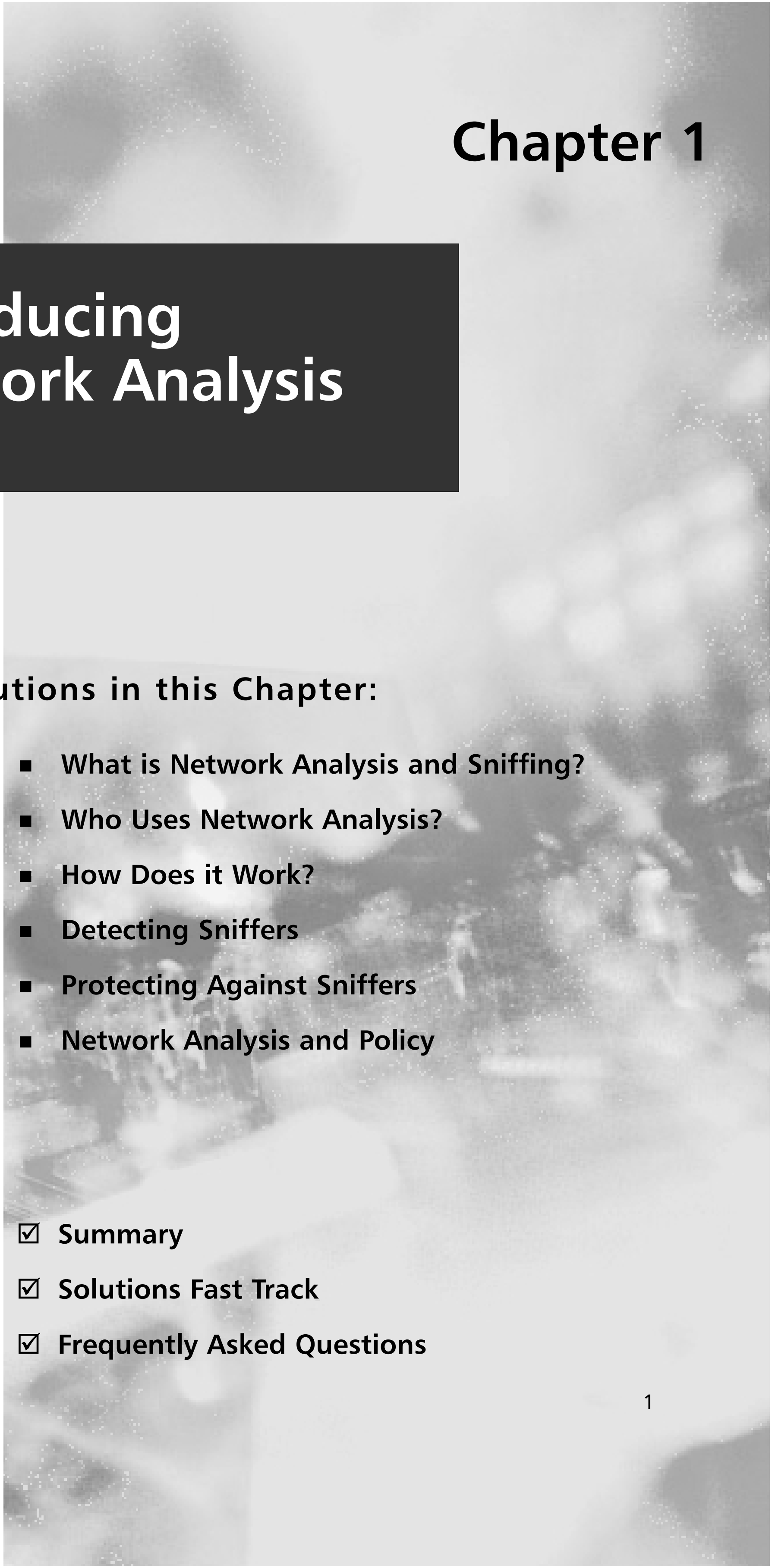


Chapter 1

Introducing Network Analysis

Solutions in this Chapter:

- What is Network Analysis and Sniffing?
- Who Uses Network Analysis?
- How Does it Work?
- Detecting Sniffers
- Protecting Against Sniffers
- Network Analysis and Policy

☑ Summary

☑ Solutions Fast Track

☑ Frequently Asked Questions

Introduction

"Why is the network slow?" "Why can't I access my e-mail?" "Why can't I get to the shared drive?" "Why is my computer acting strange?" If you are a systems administrator, network engineer, or security engineer you have probably heard these questions countless times. Thus begins the tedious and sometimes painful journey of troubleshooting. You start by trying to replicate the problem from your computer. Sure enough, you can't get to anything on the local network or the Internet either. Now what? Go to each of the servers and make sure they are up and functioning? Check that your router is functioning? Check each computer for a malfunctioning network card?

What about this scenario: you go to your main access switch, or border router, and configure one of the unused ports for port mirroring. You plug in your laptop, fire up your network analyzer, and see thousands of User Datagram Protocol (UDP) packets destined for port 1434 with various, apparently random, Internet Protocol (IP) addresses. You immediately apply access filters to block these packets from entering or exiting your network until you do more investigating. A quick search on the Internet holds the answer. The date is January 25, 2003, and you have just been hit with the SQL Slammer worm. You were able to contain the problem relatively quickly thanks to your knowledge and use of your network analyzer.

What is Network Analysis and Sniffing?

Network analysis is the process of capturing network traffic and inspecting it closely to determine what is happening on the network. A network analyzer decodes, or dissects, the data packets of common protocols and displays the network traffic in human-readable format. Network analysis is also known by several other names: traffic analysis, protocol analysis, sniffing, packet analysis, and eavesdropping to name a few. Sniffing tends to be one of the most popular terms in use today. However, as you will see later in this chapter, due to malicious users it has had a negative connotation in the past.

A network analyzer can be a standalone hardware device with specialized software, or it can simply be software that you install on your desktop or laptop computer. Network analyzers are available both free and commercially. Differences between network analyzers tend to depend on features such as the number of supported protocol decodes, the user interface, and graphing and statistical capabilities. Other differences include inference capabilities, such as expert analysis features, and the quality of packet decodes. Although several network analyzers all decode the same protocols, some may decode better than others.

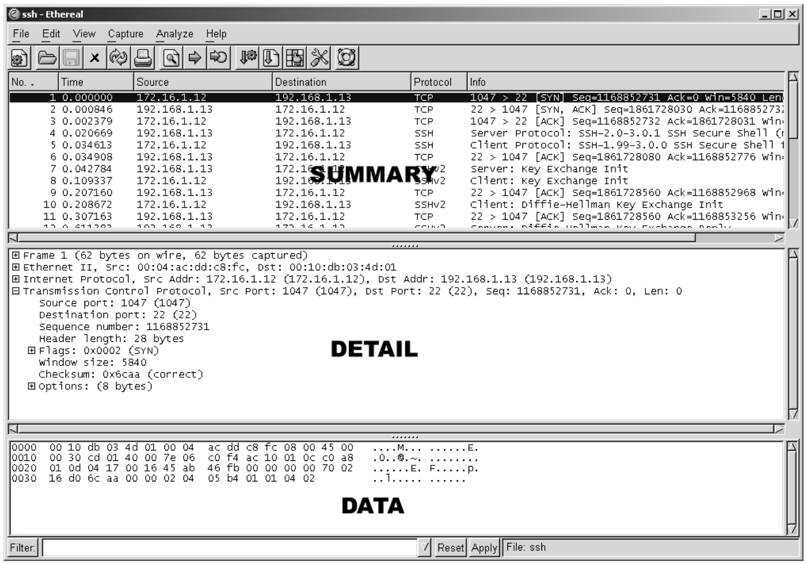
NOTE

Sniffer (with a capital "S") is a trademark owned by Network Associates referring to its Sniffer product line. However, it has become common industry usage that a "sniffer" (with a lower case "s") is a program that captures and analyzes network traffic.

Figure 1.1 shows the Ethereal Network Analyzer display windows. A typical network analyzer displays the captured traffic in three panes:

- **Summary** This pane displays a one line summary of the capture. Fields usually include date, time, source address, destination address, and the name and information about the highest-layer protocol.
- **Detail** This pane provides all of the details for each of the layers contained inside the captured packet in a tree-like structure.
- **Data** This pane displays the raw captured data both in hexadecimal and ASCII format.

Figure 1.1 Example Network Analyzer Display



```
No. .  Time       Source          Destination     Protocol  Info
     1 0.000000   172.16.1.12     192.168.1.13    TCP       1047 > 22 [SYN] Seq=1168852731 Ack=0 Win=5840 Len
     2 0.000846   192.168.1.13    172.16.1.12     TCP       22 > 1047 [SYN, ACK] Seq=1861728030 Ack=116885273
     3 0.002379   172.16.1.12     192.168.1.13    TCP       1047 > 22 [ACK] Seq=1168852732 Ack=1861728031 Win
     4 0.020669   192.168.1.13    172.16.1.12     SSH       Server Protocol: SSH-2.0-3.0.1 SSH Secure Shell (
     5 0.034613   172.16.1.12     192.168.1.13    SSH       Client Protocol: SSH-1.99-3.0.0 SSH Secure Shell 
     6 0.034908   192.168.1.13    172.16.1.12     TCP       22 > 1047 [ACK] Seq=1861728080 Ack=1168852776 Win
     7 0.042784   192.168.1.13    172.16.1.12     SSHv2     Server: Key Exchange Init
     8 0.109337   172.16.1.12     192.168.1.13    SSHv2     Client: Key Exchange Init
     9 0.207160   192.168.1.13    172.16.1.12     TCP       22 > 1047 [ACK] Seq=1861728560 Ack=1168852968 Win
    10 0.208672   172.16.1.12     192.168.1.13    SSHv2     Client: Diffie-Hellman Key Exchange Init
    11 0.307163   192.168.1.13    172.16.1.12     TCP       22 > 1047 [ACK] Seq=1861728560 Ack=1168853256 Win
```

```
⊞ Frame 1 (62 bytes on wire, 62 bytes captured)
⊞ Ethernet II, Src: 00:04:ac:dd:c8:fc, Dst: 00:10:db:03:4d:01
⊞ Internet Protocol, Src Addr: 172.16.1.12 (172.16.1.12), Dst Addr: 192.168.1.13 (192.168.1.13)
⊟ Transmission Control Protocol, Src Port: 1047 (1047), Dst Port: 22 (22), Seq: 1168852731, Ack: 0, Len: 0
    Source port: 1047 (1047)
    Destination port: 22 (22)
    Sequence number: 1168852731
    Header length: 28 bytes
  ⊞ Flags: 0x0002 (SYN)
    Window size: 5840
    Checksum: 0x6caa (correct)
  ⊞ Options: (8 bytes)
```

```
0000  00 10 db 03 4d 01 00 04  ac dd c8 fc 08 00 45 00   ....M... ......E.
0010  00 30 cd 01 40 00 7e 06  c0 f4 ac 10 01 0c c0 a8   .0..@.~. ........
0020  01 0d 04 17 00 16 45 ab  46 fb 00 00 00 00 70 02   ......E. F.....p.
0030  16 d0 6c aa 00 00 02 04  05 b4 01 01 04 02         ..l..... ......
```

A network analyzer is a combination of hardware and software. Although there are differences in each product, a network analyzer is composed of five basic parts:

- **Hardware** Most network analyzers are software-based and work with standard operating systems (OSs) and network interface cards (NICs). However, there are some special hardware network analyzers that offer additional benefits such as analyzing hardware faults including: Cyclic Redundancy Check (CRC) errors, voltage problems, cable problems, jitter, jabber, negotiation errors, etc. Some network analyzers only support Ethernet or wireless adapters, while others support multiple adapters and allow users to customize their configuration. Sometimes you will also need a hub or a cable tap to connect to the existing cable.
- **Capture driver** This is the part of a network analyzer that is responsible for actually capturing the raw network traffic from the cable. It will also filter out the traffic that you want and store the data in a buffer. This is the core of a network analyzer and you cannot capture data without it.
- **Buffer** This component stores the captured data. Data can be stored in a buffer until it is full, or in a rotation method such as "round robin" where the newest data replaces the oldest data. Buffers can be disk-based or memory-based.
- **Real-time analysis** This feature analyzes the data as it comes off the cable. Some network analyzers use this to find network performance issues, and network intrusion detection systems do this to look for signs of intruder activity.
- **Decode** This component displays the contents of the network traffic with descriptions so that it is human-readable. Decodes are specific to each protocol, so network analyzers tend to vary in the number of decodes they currently support. However, new decodes are constantly being added to network analyzers.

NOTE

Jitter is a term used to describe the random variation in the timing of a signal. Electromagnetic interference and crosstalk with other signals can cause jitter. *Jabber* is when a device is improperly handling electrical signals, thus affecting the rest of the network. Faulty network interface cards can cause jabber.

Who Uses Network Analysis?

System administrators, network engineers, security engineers, system operators, even programmers, all use network analyzers. Network analyzers are invaluable tools for diagnosing and troubleshooting network problems. Network analyzers used to be dedicated hardware devices that were very expensive. New advances in technology have allowed for the development of software network analyzers. This makes it more convenient and affordable for administrators to effectively troubleshoot a network. It also brings the capability of network analysis to anyone who wishes to perform it.

The art of network analysis is a double-edged sword. While network, system, and security professionals use it for troubleshooting and monitoring of the network, intruders can also use network analysis for harmful purposes. A network analyzer is a tool, and like all tools they can be used for both good and bad intentions.

The following list describes a few reasons why administrators use network analyzers:

- Converting the binary data in packets to human-readable format
- Troubleshooting problems on the network
- Analyzing the performance of a network to discover bottlenecks
- Network intrusion detection
- Logging network traffic for forensics and evidence
- Analyzing the operations of applications
- Discovering a faulty network card
- Discovering the origin of a Denial of Service (DoS) attack

- Detecting spyware
- Network programming to debug in the development stage
- Detecting a compromised computer
- Validating compliance with company policy
- As an educational resource when learning about protocols
- For reverse-engineering protocols in order to write clients and supporting programs

How are Intruders Using Sniffers?

When used by malicious individuals, sniffers can represent a significant threat to the security of your network. Network intruders often use network sniffing to capture valuable, confidential information. The terms sniffing and eavesdropping have often been associated with this practice. However, sniffing is now becoming a non-negative term and most people use the terms sniffing and network analysis interchangeably.

Using a sniffer in an illegitimate way is considered a passive attack. It does not directly interface or connect to any other systems on the network. However, the computer that the sniffer is installed on could have been compromised using an active attack. The passive nature of sniffers is what makes detecting them so difficult. We will discuss the methods used to detect sniffers later in this chapter.

The following list describes a few reasons why intruders are using sniffers on the network:

- Capturing clear-text usernames and passwords
- Compromising proprietary information
- Capturing and replaying Voice over IP telephone conversations
- Mapping a network
- Passive OS fingerprinting

Obviously, these are illegal uses of a sniffer, unless you are a penetration tester whose job it is to find these types of weaknesses and report them to an organization.

For sniffing to occur, an intruder must first gain access to the communication cable of the systems that are of interest. This means being on the same shared network segment, or tapping into the cable somewhere between the path of communications. If the intruder is not physically present at the target system or communications access point, there are still ways to sniff network traffic. These include:

- Breaking into a target computer and installing remotely controlled sniffing software.
- Breaking into a communications access point, such as an Internet Service Provider (ISP) and installing sniffing software.
- Locating/finding a system at the ISP that already has sniffing software installed.
- Using social engineering to gain physical access at an ISP to install a packet sniffer.
- Having an insider accomplice at the target computer organization or the ISP install the sniffer.
- Redirecting communications to take a path that includes the intruder's computer.

Sniffing programs are included with most *rootkits* that are typically installed on compromised systems. Rootkits are used to cover the tracks of the intruder by replacing commands and utilities and clearing log entries. They also install other programs such as sniffers, key loggers, and backdoor access software. Windows sniffing can be accomplished as part of some RAT (Remote Admin Trojan) such as SubSeven or Back Orifice. Often intruders will use sniffing programs that are configured to detect specific things, such as passwords, and then electronically send them to the intruder (or store them for later retrieval by the intruder). Vulnerable protocols for this type of activity include telnet, FTP, POP3, IMAP, SMTP, HTTP, rlogin, and SNMP.

One example of a rootkit is T0rnKit, which works on Solaris and Linux. The sniffer that is included with this rootkit is called t0rns and is installed in the hidden directory /usr/srec/.puta. Another example of a rootkit is Lrk5 (Linux Rootkit 5), which installs with the linsniff sniffer.

Intruders commonly use sniffer programs to control back doors. One method is to install a sniffer on a target system that listens for specific information. Then, backdoor control information can be sent to a neighboring system. The sniffer picks this up, and acts appropriately on the target computer. This type of backdoor control is often hard for investigators to detect, since it looks like the innocent neighbor system is the compromised target.

cd00r is an example of a backdoor sniffer that operates in non-promiscuous mode, making it even harder to detect. Using a product like Nmap to send a series of Transmission Control Protocol (TCP) SYN packets to several predefined ports will trigger the backdoor to open up on a pre-configured port. More information about Cdoor can be found at www.phenoelit.de/stuff/cd00r.c.

NOTE

A *rootkit* is a collection of trojan programs that are used to replace the real programs on a compromised system in order to avoid detection. Some common commands that get replaced are *ps, ifconfig,* and *ls*. Rootkits also install additional software such as sniffers.

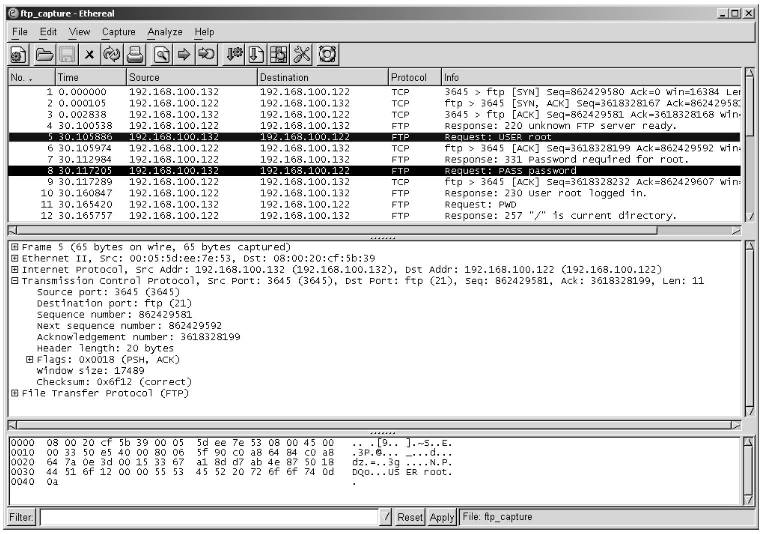
NOTE

Nmap is a network scanning tool used for network discovery and auditing. It can send raw IP packets to destination ports on target systems.

What does Sniffed Data Look Like?

We have done a lot of talking about sniffers and what they are used for, but the easiest way to grasp the concepts previously discussed is watching a sniffer in action. Figure 1.2 shows a capture of a simple FTP session from a laptop to a Sun Solaris system. The two highlighted packets show you just how easy it is to sniff the username and password. In this case, the username is "root" and the password is "password". Of course, allowing root to FTP into a system is a very poor security practice; this is just for illustration purposes!

Figure 1.2 Example of Sniffing a Connection



```
No. .  Time       Source           Destination      Protocol  Info
     1 0.000000   192.168.100.132  192.168.100.122  TCP       3645 > ftp [SYN] Seq=862429580 Ack=0 Win=16384 Ler
     2 0.000105   192.168.100.122  192.168.100.132  TCP       ftp > 3645 [SYN, ACK] Seq=3618328167 Ack=86242958
     3 0.002838   192.168.100.132  192.168.100.122  TCP       3645 > ftp [ACK] Seq=862429581 Ack=3618328168 Win=
     4 30.100538  192.168.100.122  192.168.100.132  FTP       Response: 220 unknown FTP server ready.
     5 30.105886  192.168.100.132  192.168.100.122  FTP       Request: USER root
     6 30.105974  192.168.100.122  192.168.100.132  TCP       ftp > 3645 [ACK] Seq=3618328199 Ack=862429592 Win=
     7 30.112984  192.168.100.122  192.168.100.132  FTP       Response: 331 Password required for root.
     8 30.117205  192.168.100.132  192.168.100.122  FTP       Request: PASS password
     9 30.117289  192.168.100.122  192.168.100.132  TCP       ftp > 3645 [ACK] Seq=3618328232 Ack=862429607 Win=
    10 30.160847  192.168.100.122  192.168.100.132  FTP       Response: 230 User root logged in.
    11 30.165420  192.168.100.132  192.168.100.122  FTP       Request: PWD
    12 30.165757  192.168.100.122  192.168.100.132  FTP       Response: 257 "/" is current directory.
```

```
⊞ Frame 5 (65 bytes on wire, 65 bytes captured)
⊞ Ethernet II, Src: 00:05:5d:ee:7e:53, Dst: 08:00:20:cf:5b:39
⊞ Internet Protocol, Src Addr: 192.168.100.132 (192.168.100.132), Dst Addr: 192.168.100.122 (192.168.100.122)
⊟ Transmission Control Protocol, Src Port: 3645 (3645), Dst Port: ftp (21), Seq: 862429581, Ack: 3618328199, Len: 11
    Source port: 3645 (3645)
    Destination port: ftp (21)
    Sequence number: 862429581
    Next sequence number: 862429592
    Acknowledgement number: 3618328199
    Header length: 20 bytes
  ⊞ Flags: 0x0018 (PSH, ACK)
    Window size: 17489
    Checksum: 0x6f12 (correct)
⊞ File Transfer Protocol (FTP)
```

```
0000  08 00 20 cf 5b 39 00 05  5d ee 7e 53 08 00 45 00   .. .[9.. ].~S..E.
0010  00 33 50 e5 40 00 80 06  5f 90 c0 a8 64 84 c0 a8   .3P.@... _...d...
0020  64 7a 0e 3d 00 15 33 67  a1 8d d7 ab 4e 87 50 18   dz.=..3g ....N.P.
0030  44 51 6f 12 00 00 55 53  45 52 20 72 6f 6f 74 0d   DQo...US ER root.
0040  0a                                                 .
```

Common Network Analyzers

A simple search on SecurityFocus (www.securityfocus.org/tools/category/4) shows the diversity and number of sniffers available. Some of the most prominent ones are:

- **Ethereal** Of course, this one is the topic of this book! Ethereal is obviously one of the best sniffers available. It is being developed as a free commercial quality sniffer. It has numerous features, a nice graphical user interface (GUI), decodes for over 400 protocols, and it is actively being developed and maintained. It runs on both UNIX-based systems and Windows. This is a great sniffer to use, even in a production environment. It is available at www.ethereal.com.
- **WinDump** This is the Windows version of tcpdump available at http://windump.polito.it. It uses the WinPcap library and runs on Windows 95/98/ME/NT/2000/XP.
- **Network Associates Sniffer** This is one of the most popular commercial products available. Now marketed under McAfee Network Protection Solutions, Network Associates has an entire Sniffer product line for you to peruse at www.nai.com.

- **Windows 2000/NT Server Network Monitor** Both Windows 2000 Server and NT Server have a built-in program to perform network analysis. It is located in the Administrative tools folder, but is not installed by default, so you may have to add it from the installation CD.
- **EtherPeek** This is a commercial network analyzer by WildPackets. There are versions for both Windows and Mac, as well as other network analysis products that can be found at www.wildpackets.com.
- **Tcpdump** This is the oldest and most common network sniffer. The Network Research Group (NRG) of the Information and Computing Sciences Division (ICSD) at Lawrence Berkeley National Laboratory (LBNL) developed tcpdump. It is command line-based and runs on UNIX-based systems. It is being actively developed and maintained at www.tcpdump.org.
- **Snoop** This command line network sniffer is included with the Sun Solaris operating system. It is especially competent at decoding Sun-specific protocols.
- **Sniffit** This network sniffer runs on Linux, SunOS, Solaris, FreeBSD and IRIX. It is available at http://reptile.rug.ac.be/~coder/sniffit/sniffit.html.
- **Snort** This is a network intrusion detection system that uses network sniffing. It is actively developed and maintained at www.snort.org. For more information, refer to *Snort 2.0:Intrusion Detection* (Syngress Publishing, ISBN: 1-931836-74-4)
- **Dsniff** This is very popular network sniffing package. It is a collection of programs to sniff specifically for interesting data such as passwords, and to facilitate the sniffing process such as evading switches. It is actively maintained at www.monkey.org/~dugsong/dsniff.
- **Ettercap** This sniffer is designed specifically to sniff in a switched network. It has built-in features such as password collecting, OS fingerprinting, and character injection. It runs on several platforms including Linux, Windows, and Solaris. It is actively maintained at http://ettercap.sourceforge.net.
- **Analyzer** This is a free sniffer for the Windows OS that is being actively developed by the makers of WinPcap and WinDump at

Politecnico di Torino. It can be downloaded from http://analyzer.polito.it.

- **Packetyzer** This is a free sniffer for the Windows OS that uses Ethereal's core logic. It tends to run a version or two behind the current release of Ethereal. It is actively maintained by Network Chemistry at www.networkchemistry.com/products/packetyzer/index.html.

Notes from the Underground…

Carnivore or Vegetarian?

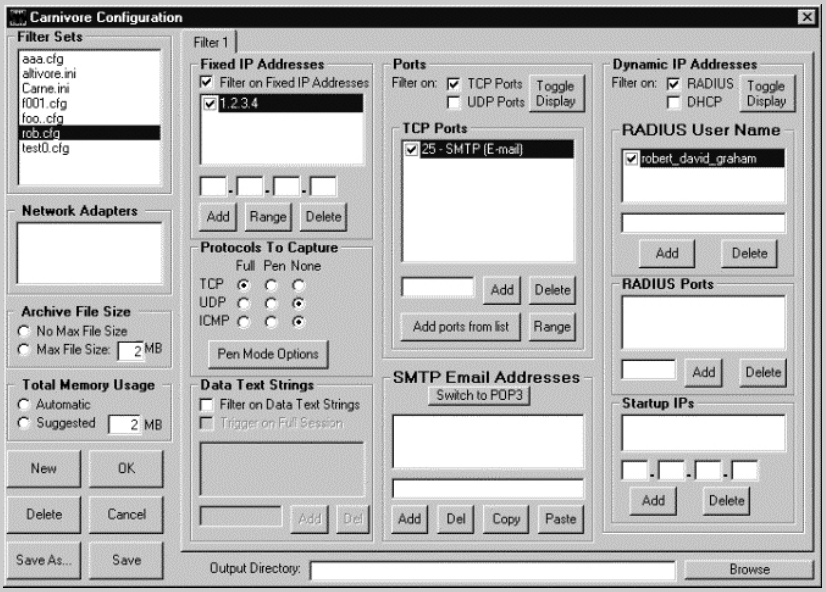
No talk about network analyzers would be complete without the mention of Carnivore. While certainly not a commonly used network analyzer, it has created a lot of talk in the security world as well as the media. Carnivore is the code name for the FBI's network analyzer. It is used to monitor relevant communications among selected individuals as part of a criminal investigation. Its name has been changed to DCS100 in an attempt to obscure its image and to calm the public's fear of its misuse. When necessary, federal agents will arrive at a suspect's ISP with a "black box", which is merely a dedicated server running Windows 2000 or NT and the FBI's Carnivore software preloaded. The server is placed on the ISP's trunk to read header information for any traffic going to or coming from the suspect. This was rather common at numerous ISPs after September 11, 2001.

Many people have been concerned about the use of Carnivore and its ability to intercept all traffic, mostly due to privacy issues. They are concerned about how Carnivore works, how it could be misused by law enforcement, and the privacy debate over cable taps in general.

Carnivore is an Internet wiretap designed by the U.S. Federal Bureau of Investigation (FBI). It is designed with the special needs of law enforcement in mind. For example, some court orders might allow a pen-register monitoring of just the From/To e-mail addresses, whereas other court orders might allow a full capture of the e-mail. A summary of Carnivore's features can be seen within the configuration program shown in Figure 1.3.

Continued

Figure 1.3 Carnivore Configuration Program



Carnivore Configuration

Filter Sets

aaa.cfg
altivore.ini
Carne.ini
f001.cfg
foo..cfg
rob.cfg
test0.cfg

Network Adapters

Archive File Size
○ No Max File Size
○ Max File Size: 2 MB

Total Memory Usage
○ Automatic
○ Suggested 2 MB

Filter 1

Fixed IP Addresses
☑ Filter on Fixed IP Addresses
☑ 1.2.3.4

Protocols To Capture

	Full	Pen	None
TCP	●	○	○
UDP	○	○	●
ICMP	○	○	●

Data Text Strings
☐ Filter on Data Text Strings
☐ Trigger on Full Session

Ports
Filter on: ☑ TCP Ports ☐ UDP Ports

TCP Ports
☑ 25 - SMTP (E-mail)

SMTP Email Addresses

Dynamic IP Addresses
Filter on: ☑ RADIUS ☐ DHCP

RADIUS User Name
☑ robert_david_graham

RADIUS Ports

Startup IPs

Output Directory:

The features are:

- **Filter sets** The settings are saved in configuration files; the user can quickly change the monitoring by selecting a different filter set.
- **Network adapters** A system may have multiple network adapters; only one can be selected for sniffing at a time.
- **Archive file size** A limit can be set on how much data is captured; by default, it fills up the disk.
- **Total memory usage** Network traffic may come in bursts faster than it can be written to disk; memory is set aside to buffer the incoming data.
- **Fixed IP address** All traffic to/from a range of IP addresses can be filtered. For example, the suspect may have a fixed IP address of 1.2.3.4 assigned to their cable modem. The FBI might get a court order allowing them to sniff all of the suspect's traffic.

Continued

- **Protocols to capture** Typically, a court order will allow only specific traffic to be monitored, such as SMTP over TCP. In Pen mode, only the headers are captured.
- **Data text strings** This is the Echelon feature that looks for keywords in traffic. A court order must specify exactly what is to be monitored, such as an IP address or e-mail account. Such wide-open keyword searches are illegal in the United States. The FBI initially denied that Carnivore had this feature.
- **Ports** A list of TCP and UDP ports can be specified. For example, if the FBI has a court order allowing e-mail capture, they might specify the e-mail ports of 25 (SMTP), 110 (POP3), and 143 (IMAP).
- **SMTP e-mail addresses** A typical scenario is where Carnivore monitors an ISPs e-mail server, discarding all e-mails except those of the suspects. An e-mail session is tracked until the suspect's e-mail address is seen, then all the packets that make up the e-mail are captured.
- **Dynamic IP addresses** When users dial up the Internet, they are logged in via the RADIUS protocol, which then assigns them an IP address. Normally, the FBI will ask the ISP to reconfigure their RADIUS servers to always assign the same IP address to the suspect, and will then monitor all traffic to/from that IP address. Note: if you are a dial-up user and suspect the FBI is after you, check to see if your IP address is the same every time you dial up. Sometimes this isn't possible. Carnivore can be configured to monitor the RADIUS protocol and dynamically discover the new IP address assigned to the suspect. Monitoring begins when the IP address is assigned, and stops when it is unassigned.

The FBI developed Carnivore because other existing utilities do not meet the needs of law enforcement. When an e-mail is sent across the wire, it is broken down into multiple packets. A utility like mailsnarf will reassemble the e-mail back into its original form. This is bad because the suspect's defense attorneys will challenge its accuracy: Did a packet get dropped somewhere in the middle that changes the meaning of the e-mail? Did a packet from a different e-mail somehow get inserted into the message? By capturing the raw packets rather than reassembling

Continued

them, Carnivore maintains the original sequence numbers, ports, and timestamps. Any missing or extra packets are clearly visible, allowing the FBI to defend the accuracy of the system.

Another problem that the FBI faces is minimization of the sniffed data. When the FBI wiretaps your line, they must assign an agent to listen in. If somebody else uses your phone (like your spouse or kids), they are required to turn off the tape recorders. In much the same way, Carnivore is designed to avoid capturing anything that does not belong to the suspect. A typical example would be using Carnivore to monitor the activities of a dial-up user. Carnivore contains a module to monitor the RADIUS traffic that is used by most ISPs to authenticate the user and assign a dynamic IP address. This allows Carnivore to monitor only that user without intercepting any other traffic.*

The following websites have more information on Carnivore:

- www.fbi.gov
- www.robertgraham.com/pubs/carnivore-faq.html
- www.stopcarnivore.org

*Excerpt from Robert Graham's chapter in *Hack Proofing Your Network, Second Edition*. Syngress Publishing 1-928994-70-9.

How Does It Work?

This section provides an overview of how all of this sniffing takes place. It gives you a little background on how networks and protocols work; however, there are many excellent resources out there that fill entire books themselves! The most popular and undoubtedly one of the best resources is Richard Stevens' "TCP/IP Illustrated, Vol. 1 – 3".

Explaining Ethernet

Ethernet is the most popular protocol standard used to enable computers to communicate. A protocol is like speaking a particular language. Ethernet was built around a principle of a shared medium where all computers on the local network segment share the same cable. It is known as a *broadcast* protocol because when a computer has information to send, it sends that data out to all other computers on the same network segment. This information is divided up into manageable chunks called packets. Each packet has a header, which is like an envelope containing the addresses of both the destination and source computers. Even though this information is sent out to all computers on a segment, only the computer with the matching destination address will respond. All of the other computers on the network still see the packet, but if they are not the intended receiver they will disregard and discard it, unless a computer is running a sniffer. When you are running a sniffer, the packet capture driver that we mentioned earlier will put the computer's NIC into what is known as promiscuous mode. This means that the sniffing computer will be able to see all of the traffic on the segment regardless of who it is being sent to. Normally computers run in non-promiscuous mode, listening for information only designated for themselves. However, when a NIC is in promiscuous mode it can see conversations to and from all of its neighbors.

Ethernet addresses are known as Media Access Control (MAC) addresses, hardware addresses, or sometimes just Ethernet addresses. Since many computers may share a single Ethernet segment, each must have an individual identifier. These identifiers are hard-coded on to the NIC. A MAC address is a 48-bit number, also stated as a 12-digit hexadecimal number. This number is broken down into two halves, the first 24-bits identify the vendor of the Ethernet card, and the second 24-bits is a serial number assigned by the vendor.

The following steps will allow you to view your NIC's MAC address:

- **Windows 9*x*** Access **Start | Run**, and type **winipcfg.exe**. The MAC address will be listed as "Adapter Address".
- **Windows NT/2000/XP** Access the command line and type **ipconfig /all**. The MAC address will be listed as "Physical Address".
- **Linux and Solaris** Type **ifconfig –a** at the command line. The MAC address will be listed as "HWaddr" on Linux and "ether" on Solaris.

You can also view the MAC addresses of other computers that you have communicated with recently, by using the command **arp –a**. More will be discussed about this in the "Defeating Switches" section.

MAC addresses are unique, and no two computers should have the same one. However, this is not always the case. Occasionally there could be a manufacturing error that would cause more than one network interface card to have the same MAC address, but mostly, people will change their MAC addresses on purpose. This can be done with a program, such as ifconfig, that will allow you to fake your MAC address. Faking your MAC address is also called *spoofing*. Also, some adapters allow you to use a program to reconfigure the runtime MAC address. And lastly with the right tools and skill you can physically re-burn the address into the network interface card.

NOTE

Spoofing is the altering of network packet information such as the IP source address, MAC address, or even an e-mail address. This is often done to masquerade as another device in order to exploit a trust relationship, or to make tracing the source of attacks difficult. Address spoofing is also used in denial of service (DoS) attacks, such as Smurf, where the return address of network requests are spoofed to be the IP address of the victim.

Understanding the OSI model

The International Standards Organization (ISO) developed the Open Systems Interconnection (OSI) model in the early 1980's to describe how network protocols and components work together. It divides network functions into seven layers, and each layer represents a group of related specifications, functions, and activities.

The layers of the OSI model are:

- **Application layer** This topmost layer of the OSI model is responsible for managing communications between network applications. This layer is not the application program itself, although some applications may have the ability and the underlying protocols to perform application layer functions. For example, a Web browser is an application, but it is the underlying Hypertext Transfer Protocol (HTTP) protocol that provides the application layer functionality. Examples of application layer protocols include File Transfer Protocol (FTP), Simple Network Management Protocol (SNMP), Simple Mail Transfer Protocol (SMTP), and Telnet.
- **Presentation layer** This layer is responsible for data presentation, encryption, and compression.

- **Session layer** The session layer is responsible for creating and managing sessions between end systems. The session layer protocol is often unused in many protocols. Examples of protocols at the session layer include NetBIOS and Remote Procedure Call (RPC).
- **Transport layer** This layer is responsible for communication between programs or processes. Port or socket numbers are used to identify these unique processes. Examples of transport layer protocols include: TCP, UDP, and Sequenced Packet Exchange (SPX).
- **Network layer** This layer is responsible for addressing and delivering packets from the source computer to the destination computer. The network layer takes data from the transport layer and wraps it inside a packet or datagram. Logical network addresses are generally assigned to computers at this layer. Examples of network layer protocols include IP and Internetwork Packet Exchange (IPX). Devices that work at this layer are routers and Layer 3 switches.
- **Data link layer** This layer is responsible for delivering frames between NICs on the same physical segment. Communication at the data link layer is generally based on MAC addresses. The data link layer wraps data from the network layer inside a frame. Examples of data link layer protocols include Ethernet, Token Ring, and Point-to-Point Protocol (PPP). Devices that operate at this layer include bridges and switches.
- **Physical layer** This layer defines connectors, wiring, and the specifications on how voltage and bits pass over the cabled or wireless media. Devices at this layer include repeaters, concentrators, hubs, and cable taps. Devices that operate at the physical layer do not have an understanding of network paths.

NOTE

The terms *frame* and *packet* tend to be used interchangeably when talking about network traffic. However, the difference lies in the various layers of the OSI model. A frame is a unit of transmission at the data link layer. A packet is a unit of transmission at the network layer, however many people use the term packet to refer to data at any layer.

The OSI model is very generic and can be used to explain virtually any network protocol. Various protocol suites are often mapped against the OSI model for this purpose. A solid understanding of the OSI model aids tremendously in network analysis, comparison, and troubleshooting. However, it is also important to remember that not all protocols map nicely to the OSI model. For example, TCP/IP was designed to map to the U.S. Department of Defense (DoD) model. In the 1970s, the DoD developed its four-layer model. The core Internet protocols adhere to this model.

The DoD model is merely a condensed version of the OSI model. Its four layers are:

- **Process layer** This layer defines protocols that implement user-level applications such as mail delivery, remote login, and file transfer.
- **Host-to-host layer** This layer handles the connection, data flow management, and retransmission of lost data.
- **Internet layer** This layer is responsible for delivering data from source host to destination host across a set of different physical networks that connect the two machines.
- **Network access layer** This layer handles the delivery of data over a particular hardware media.

Notes from the Underground…

The TCP/IP Protocols

You will be seeing a lot of references in this book to TCP/IP and its associated protocols, specifically IP, TCP, and UDP. TCP/IP, developed by the Defense Advanced Research Projects Agency (DARPA), is the most widely used routed protocol today. IP is a Layer 3 protocol that contains addressing and control information that allows packets to be routed. IP is a connectionless protocol; therefore, it provides unreliable best-effort packet delivery service. Since IP only provides best-effort delivery, a packet may be discarded during transmission. All IP packets consist of a header and a payload (data from upper layers).

At the transport layer of the TCP/IP stack, the two commonly used protocols are TCP and UDP. The headers for both of these protocols

Continued

include a source and destination port number, which are used to determine the application or process that the TCP segment or UDP datagram originate from and destined to. TCP is a connection-oriented protocol, and UDP is a connectionless protocol. The TCP header includes sequence and acknowledgment numbers for reliable delivery. When IP needs reliable, guaranteed transfers it depends on TCP to provide this functionality.

Since TCP is a connection-oriented protocol it creates a dialog between the two communicating hosts to establish a connection. This is known as the three-way handshake. It starts by Host A sending a *SYN* packet to Host B letting it know that it wants to talk. Host B then responds with a *SYN/ACK*, saying that it is available to talk. Host A then finalizes the connection with an ACK.

TCP can also use the sliding window principle. The sliding window algorithm allows a buffer to be placed between the application program and the network data flow. Data received from the network is placed into this buffer until the application is ready to read it. The window is the amount of data that can be fetched into the buffer before an acknowledgment must be sent. Examples of applications that use TCP include FTP, Telnet, Network File System (NFS), SMTP, HTTP, Domain Name System (DNS), and Network News Transfer Protocol (NNTP). Examples of applications that use UDP include DNS, Routing Information Protocol (RIP), NFS, SNMP, and Dynamic Host Configuration Protocol/Boot Protocol (DHCP/BOOTP). As you can see, some applications (such as DNS and NFS) can use both protocols.

Notes from the Underground…

Writing Your Own Sniffer

There is an excellent paper titled "Basic Packet-Sniffer Construction from the Ground Up" by Chad Renfro located at www.unixgeeks.org/security/newbie/security/sniffer/sniffer_construction.txt. In this paper he presented a very basic 28-line packet sniffer written in C, called sniff.c. Even if you aren't a programmer, Chad explains the program line by line in an

Continued

easy to understand manner. The program demonstrates the use of the RAW_SOCKET device to read TCP packets from the network and print basic header information to std_out. For simplicity, the program operates in non-promiscuous mode, so you would first need to put your interface in promiscuous mode by using the **ifconfig eth0 promisc** command.

There is also a header file that has to be copied into the same directory as sniff.c. It provides standard structures to access the IP and TCP fields. The structures identify each field in the IP and TCP header. It contains more information than what the sniff.c actually uses, but it least it is there to build upon.

To run the program, copy the sniff.c and headers.h into the same directory, and enter the command **gcc -o sniff sniff.c**. This will compile the program and create and executable file called sniff, which can be run by typing **./sniff**. The following text shows the output of the sniff program when I attempted a TELNET and FTP connection:

```
Bytes received :::    48
Source address ::: 192.168.1.1
IP header length ::: 5
Protocol ::: 6
Source port ::: 1372
Dest port  ::: 23
Bytes received :::    48
Source address ::: 192.168.1.1
IP header length ::: 5
Protocol ::: 6
Source port ::: 1374
Dest port  ::: 21
```

Once you are done capturing data, you can end the program by typing **CTRL-C**. You may also want to remove your interface from promiscuous mode by typing the command **ifconfig eth0 –promisc**.

CSMA/CD

Ethernet uses the Carrier Sense Multiple Access/Collision Detection (CSMA/CD) protocol for devices on the network to exchange data. The term *multiple access* refers to the fact that many network devices attached to the same segment have the opportunity to transmit. Each device is given an equal opportunity; no device has priority over any other. *Carrier sense* describes how an Ethernet interface on a network device listens to the cable before transmitting. The network interfacer ensures that there are no other signals on the cable before it transmits. An Ethernet interface also listens while transmitting to ensure that no other network device transmits data at the same time. When two network devices transmit at the same time, a *collision* occurs. Since Ethernet interfaces listen to the media while they are transmitting, they are able to identify the presence of others through their *collision detection* method. If a collision occurs, the transmitting device will wait a random amount of time before retransmitting. This function is known as *random backoff*.

Traditionally, Ethernet operation has been *half duplex*. This means that an interface may either transmit or receive data, but it cannot do both at the same time. If more than one network interface on a segment tries to transmit at the same time, a collision occurs, as per CSMA/CD. When a crossover cable is used to connect two devices or a single device is attached to a switch port, only two interfaces on the segment need to transmit or receive and no collisions occur. This is because the transmit (TX) of device A is connected to the receive (RX) of device B, and the TX of B is connected to the RX of A. The collision detection method is therefore no longer necessary, so interfaces can be placed in *full-duplex* mode of operation. This mode allows network devices to transmit and receive at the same time, thereby increasing performance.

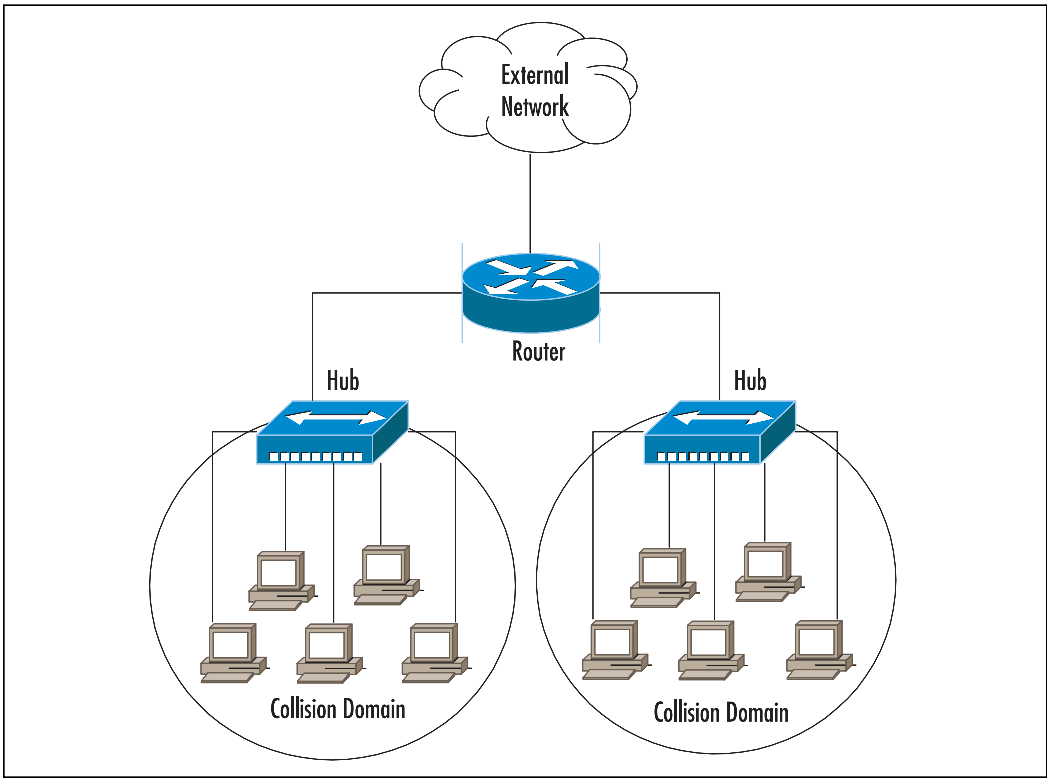
Hardware: Taps, Hubs, and Switches, Oh My!

Cable taps are hardware devices that assist in connecting to the network cable. Tap stands for Test Access Point, and you can use this device to access any cable between computers, hubs, switches, routers, and other devices. Taps are available in full or half-duplex for 10, 100, and 1000 Mbps Ethernet links. They are also available in various multi-port sizes. Following is a list of some popular cable tap products:

- Net Optics carries several types of network taps for copper and fiber cables. They can be viewed at www.netoptics.com.
- The Century Tap family is available by Shomiti at www.shomiti.net/shomiti/century-tap.html. They offer a variety of taps for copper and fiber cables.

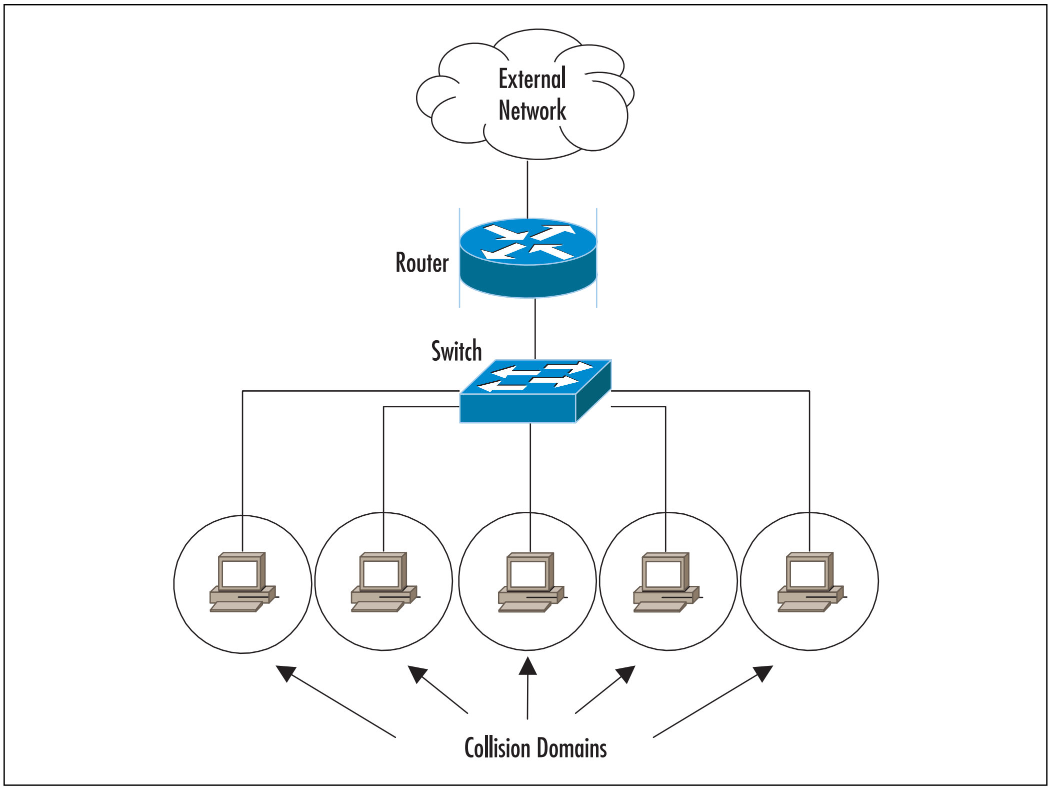
A hub is a device that allows you to connect multiple hosts together on a shared medium, such as Ethernet. When a computer sends information, it travels into the hub and the hub blindly forwards the information to all other computers connected to it. As we explained before with Ethernet, the computer that the information was intended for will recognize its own MAC address in the packet header and then accept the data. The area that the hub forwards all information to is known as a *collision domain*, or *broadcast domain*. A hub has only one collision domain for all of the traffic to share. Figure 1.4 shows a network architecture with collision domains related to hubs. Large collision domains not only makes sniffing easier, but also create performance issues like bandwidth hogging or excessive traffic on the hub.

Figure 1.4 Hub Collision Domains



A switch operates very differently from a hub. It is also used to connect computers together on a shared medium; however, when a switch receives information from a computer it doesn't just blindly send it to all other computers. A switch will actually look at the packet header to locate the destination MAC address. A switch maintains a list of all MAC addresses and corresponding ports on the switch that the computers are connected to. It will then forward the packets to the specified port. This narrows the collision domain, or broadcast domain to a single port, as shown in Figure 1.5. This type of collision domain will also provide a definite amount of bandwidth for each connection rather than a shared amount on a hub. Since the price of switches has fallen dramatically in the last few years, there is no reason to not replace hubs with switches, or to choose switches when purchasing new equipment. Also, some of the more costly switches often include better technology to make them more resistant to sniffing attacks.

Figure 1.5 Switch Collision Domains



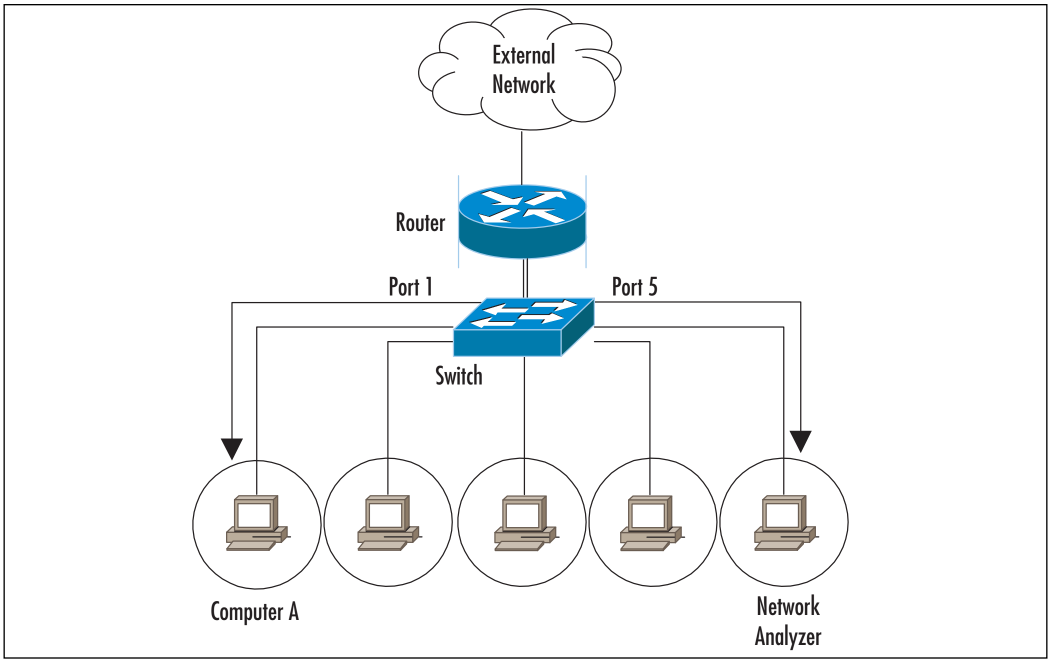
As you can see from the diagrams, hubs make sniffing easier, and switches make it more difficult. However, switches can be tricked, as discussed in the "Defeating Switches" section.

Port Mirroring

What if you are working in a network that uses switches and you want to perform network analysis legitimately? You are in luck, as most switches and routers come with a feature known as *port mirroring,* or *port spanning.* To mirror ports, you need to configure the switch to duplicate the traffic from a port you want to monitor to a port you are connected to with your network analyzer. This feature was designed just for this purpose, to analyze network traffic for troubleshooting.

Using port spanning does not interfere with the normal operation of switches, but you always want to check the documentation of the exact switch you are configuring and periodically check the device's logs. You won't affect the switch, but you will increase the amount of traffic on a specific destination port, so make sure your properly configured network analyzer is the destination port. Please consult the documentation for your specific switch to learn the exact command to enable port mirroring. Figure 1.6 shows the process of port mirroring. The switch is configured to mirror all port 1 traffic to port 5. The network analyzer will see any traffic to and from Computer A. Sometimes administrators will mirror the uplink port on a switch; that way they will see all traffic to and from the switch and all of its ports.

Figure 1.6 Port Mirroring



NOTE

Span means Switched Port ANalyzer. Cisco uses the word *span* to describe the concept of port mirroring. To span a port in Cisco terms is the same as mirroring a port.

Defeating Switches

We mentioned earlier that the use of switches in your network makes sniffing more difficult. In theory, on a switch you should only see traffic destined for you own computer. Notice we didn't say that switching eliminates sniffing. There are ways to trick a switch, or to get around its technology. The following list describes several ways in which a switch can be defeated:

- **Switch Flooding** Some switches can be made to act like a hub, where all packets are broadcast to all computers. This can be accomplished by overflowing the switch address table with all kinds of fake MAC addresses. This is known as a device *failing open*, thus removing all security provisions. Devices that *fail close* will incorporate some sort of security measure, such as shutting down all communications. The Dsniff package comes with a program called *macof* that is designed to perform switch MAC address flooding. It can be downloaded from http://monkey.org/~dugsong/dsniff.
- **ARP Redirects** When a computer needs to know the MAC address of another computer, it will send an ARP request. Each computer maintains an ARP table to store the MAC addresses of other computers that it has talked to. ARPs are broadcast on a switch, so all computers on that switch will see the request and the response. There are several methods that use ARP to trick a switch into sending traffic somewhere it shouldn't. First, an intruder can subvert a switch by sending out an ARP claiming to be someone else as the MAC address. An intruder can also send an ARP claiming to be the router, in which case computers will try to send their packets through the intruder's computer. Or, an intruder can send an ARP request just to one victim, claiming to be the router, at which point the victim will start forwarding packets to the

intruder. All of these tricks will allow an intruder to see information that he/she is not supposed to see.

- **ICMP Redirect** Sometimes computers are on the same physical segment, the same switch, but different logical segments. This means they are in different IP subnets. When Computer A wants to talk to Computer B it will send its request through a router. The router knows that they are on the same physical segment, so it will send an ICMP Redirect to Computer A letting it know that it can send its packets directly to Computer B. An intruder, Computer X, could send a fake ICMP redirect to Computer A, claiming that it should send Computer B's packets to Computer X.
- **ICMP Router Advertisements** These advertisements inform computers of who the router is. An intruder could send these types of advertisements out claiming to be the router, and computers will start to forward all packets through the intruder.
- **MAC Address Spoofing** An intruder can pretend to be using a different computer by spoofing its MAC address. Sending out packets with the source address of the victim will trick the switch. The switch will enter the spoofed information into its table and begin sending packets to the intruder. But what about the victim, who is still on the switch and sending updates causing the switch to change the table back? This can be solved by taking the victim offline with some sort of DoS attack, then redirecting the switch and continuing with communications. The intruder could also broadcast out the traffic that he receives to ensure that the victim computer still receives the packets. Some switches have a countermeasure that will allow you to statically assign a MAC address to a port. This may be difficult to manage if you have a large network, but it will eliminate MAC spoofing.

  To spoof your MAC on Linux or Solaris when you are connected locally, you can simply use ifconfig as follows:

  ```
  ifconfig eth0 down
  ifconfig eth0 hw ether 00:02:b3:00:00:AA
  ifconfig eth0 up
  ```

  Register the MAC on all hosts by broadcast ping (and use Control C to close the ping): **ping -c 1 -b 192.168.1.255**

Now you can sniff all traffic to the computer that owns this MAC address.

- **Reconfigure port spanning on the switch** As we mentioned earlier, switch ports can be configured to see traffic destined for other ports. An intruder could perform this by connecting to the switch via Telnet or some other default backdoor. The intruder could also use SNMP if it is not secured.
- **Cable taps** As mentioned earlier, cable taps can be used to physically tap into the cable. Tapping into the uplink cable on a switch will show you all of the traffic entering and exiting that switch.

There are many methods of defeating switches, but this is contingent upon how a switch operates. Not all of the methods discussed will work, especially with newer, more technologically savvy switches. The Dsniff FAQ contains some good information for sniffing in a switched environment. It can be located at http://monkey.org/~dugsong/dsniff/faq.html.

Detecting Sniffers

Remember earlier that we said sniffers are a form of passive attack. They don't interact with any devices or transmit any information, thus making them very difficult to detect. Although tricky, detecting sniffers is possible. The easiest method is to check your network interfaces to see if they are in promiscuous mode. On UNIX-based systems the command **ifconfig –a** will list the network adapters on the system. Look for the PROMISC flag in the output, such as in the following example:

```
[root@localhost root]# ifconfig -a
eth0      Link encap:Ethernet  HWaddr 00:02:B3:06:5F:5A
          inet addr:192.168.1.2  Bcast:192.168.1.255  Mask:255.255.255.0
          UP BROADCAST RUNNING PROMISC MULTICAST  MTU:1500  Metric:1
          RX packets:204 errors:0 dropped:0 overruns:0 frame:0
          TX packets:92 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:100
          RX bytes:46113 (45.0 Kb)  TX bytes:5836 (5.6 Kb)
          Interrupt:11 Base address:0x1800 Memory:e8120000-e8120038
```

If ifconfig is not detecting a sniffer that you know is currently installed and in promiscuous mode, you can try using the **ip link** command, a handy TCP/IP interface configuration and routing utility. The following example shows the output from the ip command:

```
[root@localhost root]# ip link
1: lo: <LOOPBACK,UP> mtu 16436 qdisc noqueue
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
2: eth0: <BROADCAST,MULTICAST,PROMISC,UP> mtu 1500 qdisc pfifo_fast qlen 100
    link/ether 00:02:b3:06:5f:5a brd ff:ff:ff:ff:ff:ff
```

Detecting promiscuous mode on Windows systems is more difficult because there are no standard commands that will list that type of information. However, there is a free tool called PromiscDetect, developed by Arne Vidstrom, that will detect promiscuous mode network adapters for Windows NT, 2000, and XP. It can be downloaded from http://ntsecurity.nu/toolbox/promiscdetect. The following example shows the output of PromiscDetect, the D-link adapter is in normal operation mode, but the Intel adapter has Ethereal running on it:

```
C:\>promiscdetect
PromiscDetect 1.0 - (c) 2002, Arne Vidstrom (arne.vidstrom@ntsecurity.nu)
                   - http://ntsecurity.nu/toolbox/promiscdetect/
Adapter name:
 - D-Link DWL-650 11Mbps WLAN Card
Active filter for the adapter:
 - Directed (capture packets directed to this computer)
 - Multicast (capture multicast packets for groups the computer is a member
of)
 - Broadcast (capture broadcast packets)
Adapter name:
 - Intel(R) PRO/100 SP Mobile Combo Adapter
Active filter for the adapter:
 - Directed (capture packets directed to this computer)
 - Multicast (capture multicast packets for groups the computer is a member
of)
 - Broadcast (capture broadcast packets)
 - Promiscuous (capture all packets on the network)
WARNING: Since this adapter is in promiscuous mode there could be a sniffer
         running on this computer!
```

Unfortunately some sniffers can cover their tracks by hiding the promiscuous flags. Also, if the sniffer was installed on a compromised system by using a rootkit, the intruder has most likely replaced commands like ifconfig. The following list describes several other methods that could be used to detect sniffers on the network:

- Monitor DNS reverse lookups. Some sniffers will perform DNS queries to resolve IP addresses to host names. Performing a network ping scan or pinging your entire network address space could trigger this activity.
- Send TCP/IP packets to all IP addresses on the same Ethernet segment, but with fake MAC addresses. Normally the network interface card will drop packets with the wrong MAC address. However, some systems, when in promiscuous mode, will answer with a reset packet (RST). This may also work in a switched environment since switches forward broadcast packets that they don't have MAC addresses listed for. Many newer sniffers have build in defenses for this technique by altering the way they handle MAC addresses.
- Carefully monitor hub ports. Ideally you would have a network diagram and your cables would be labeled. Then, if something unusual appeared, such as a new device or a newly active hub port, you would recognize it. However, in reality, wiring closets and cabling can be a nightmare. If your hubs are being monitored with a protocol such as SNMP via a network management system, you may be able to use this information to detect any unusual connects and disconnects.
- Remember how ARP is used to link IP addresses to MAC addresses. Normally an ARP is sent out as a broadcast to everyone. However, you could send out an ARP to a non-broadcast address, followed by a broadcast ping. No one should have your information in his or her ARP table except the sniffer because it was listening to all traffic, even the non-broadcast traffic. Therefore the computer with the sniffer would respond.
- Use a honeypot. A honeypot is a server that is set up to monitor the activity of intruders. It contains fake data and services. In this case you could create fake administrator or user accounts on the honeypot and then create connections across the network to it using clear text protocols such as Telnet or FTP. If there are sniffers monitoring for user

names and passwords they will see the honeypot and the intruder will eventually try to log into it. Honeypots run intrusion detection software to monitor activity, and special signatures can be added to trigger alerts when the fake accounts are used.

- Carefully monitor your hosts. This includes disk space, CPU utilization, and response times. Sniffers gradually consume disk space each day as they log traffic, and they can sometimes put a noticeable load on the CPU. When the infected computer's resources become consumed it will respond more slowly than normal.

There are several tools that can be used to detect sniffers on your network. Many of them are outdated and no longer actively maintained, and sometimes just hard to find. Also, newer sniffers have been rewritten to evade their detection. However, we want to take a moment to mention some of them.

- **PromiScan Ver 0.27** This is a free program by Security Friday that is up-to-date and actively maintained. It runs on Windows 2000 and XP and requires the WinPcap driver. It can scan the local network looking for remote promiscuous mode adapters, using ARP packets. It can be downloaded from www.securityfriday.com/ToolDownload/PromiScan/promiscan_doc.html.
- **AntiSniff** This program was originally written by L0pht, but is no longer supported or maintained. Archived Windows and UNIX versions can be downloaded from http://packetstormsecurity.nl/sniffers/antisniff.
- **Sentinel** This free program performs remote promiscuous detection, and runs on various versions of BSD and Linux. It requires the libpcap and libnet libraries to operate. It can be downloaded from www.packetfactory.net/projects/sentinel.
- **Neped** Network Promiscuous Ethernet Detector is a free UNIX-based program originally written by the Apostols Group to remotely detect promiscuous mode network interface cards on Linux computers. It only detects on a subset of Linux systems with unpatched kernels before version 2.0.36. The Apostols website no longer exists and neped can be difficult to find. Currently there is a version located at www.dsinet.org/tools/network-sniffers/neped.c.

- **Check Promiscuous Mode (CPM)** This is a free UNIX-based program developed by CERT/CC in response to increased network sniffing. More information, including the program, can be obtained from www.cert.org/advisories/CA-1994-01.html.
- **Ifstatus** This is a free UNIX-based program to detect promiscuous mode interfaces on Solaris and AIX systems. It can be downloaded from ftp://ftp.cerias.purdue.edu/pub/tools/unix/sysutils/ifstatus.
- **Promisc.c** This is a free UNIX-based program to detect promiscuous mode interfaces on Linux and some SunOS systems. It can be downloaded from www.dsinet.org/tools/network-sniffers/promisc.c.

Protecting Against Sniffers

So far you have learned what sniffing is and how it works.You have also learned some of the tricks that can be used by intruders to sniff where they aren't supposed to, and some not-so-foolproof methods of detecting sniffers. None of this sheds much of a positive light on your plight to protect your network and data. Fortunately there are some methods that you can use on your network that offer protection against the passive attack known as sniffing.

We talked earlier about using switches on your network instead of hubs. However, we also learned the methods used to defeat switches. Using switches is a network best practice that will allow increased performance and security that should be used regardless of existing methods to evade them. While switches will present a barrier to casual sniffing, the best method of protecting your data is encryption. Encryption is the best form of protection against traffic interception, on public networks as well as your own internal networks. Intruders will still be able to sniff the traffic, but the data will appear unreadable. Only the intended recipient should be able to decrypt and read the data. Some methods of encryption still leave the headers in cleartext, so the intruder will be able to see the source and destination addresses and possibly map the network, but the data will be obscured. Other forms of encryption will also mask the header portion of the packet.

A virtual private network (VPN) uses encryption and authentication to provide secure communications over an otherwise insecure network. VPNs protect the transmission of data over the Internet, and even your internal network. However, if an intruder compromises either of the end nodes of a VPN, the protection is rendered useless.The following list describes some of the VPN methods in use today that will protect your data against sniffing:

- **Secure Shell (SSH)** SSH is an application-level VPN that runs over TCP to secure client-to-server transactions. This is often used for general logins and to administer servers remotely. It is typically used to replace Telnet, FTP, and Berkley Services "r" commands. However, since any arbitrary TCP protocol can be tunneled through an SSH connection, it can be used for numerous other applications. SSH provides authentication by RSA or DSA asymmetric key pairs. The headers in an SSH session are not encrypted, so an intruder will still be able to view the source and destination addresses.
- **Secure Sockets Layer (SSL)/Transport Layer Security (TLS)** SSL was originally developed by Netscape Communications to provide security and privacy to Internet sessions. It has been replaced by TLS as stated in RFC 2246. TLS provides security at the transport layer and overcomes some security issues of SSL. It is used to encapsulate the network traffic of higher-level applications such as LDAP, HTTP, FTP, NNTP, POP3, and IMAP. It provides authentication and integrity via digital certificates and digital signatures.
- **IP Security (IPSec)** IPSec is a network-level protocol that incorporates security into the IPv4 and IPv6 protocols directly at the packet level by extending the IP packet header. This allows the ability to encrypt any higher layer protocol. It is currently being incorporated into routing devices, firewalls, and clients for securing trusted networks to one another. IPSEC provides several means for authentication and encryption, supporting quite a few public key authentication ciphers and symmetric key encryption ciphers. It can operate in tunnel mode to provide a new IP header that will mask the original source and destination addresses.

One-time passwords (OTP) is another method to protect against sniffing. S/key, One-time Passwords In Everything (OPIE), and other one-time password techniques will protect against the collection and reuse of passwords. They operate by using a challenge-response method, and a different password is transmitted each time authentication is needed. The passwords that a sniffer collects will be useless since they are only used once. Smart cards are a popular method of implementing one-time passwords.

E-mail protection is a hot topic for both companies and individuals. Two methods of protecting e-mail, by encrypting it in transit and in storage, are Pretty

Good Privacy (PGP) and Secure Multipurpose Internet Mail Extensions (S/MIME). Each of these methods also provides authentication and integrity by the use of digital certificates and digital signatures.

Network Analysis and Policy

There is one very important topic that we would like to take time to address. Before cracking open your newly installed network analyzer at work, please read your company policy! A properly written and comprehensive "Appropriate Use" network policy will more than likely prohibit you from running network analyzers. Usually the only exception to this is if network analysis is in your job description. Also, just because you may provide security consulting services for company clients, does not mean that you can use your sniffer on the company network. However, if you are an administrator and are allowed to legitimately run a sniffer, you can use it to enforce your company's security policy. If your security policy prohibits the use of file sharing applications such as KaZaA, Morpheus, or messaging services such as Internet Relay Chat (IRC) or Instant Messenger, you could use your sniffer to detect this type of activity.

Also, if you provide security services for clients, such as an ethical hacker who performs penetration testing, be sure that the use of a sniffer is included in your Rules of Engagement. Be very specific about how, where, and when it will be used. Also provide clauses, such as Non-Disclosure Agreements, that will exempt you from the liability of learning confidential information.

Another word of caution: many ISPs prohibit the use of sniffers in their "Appropriate Use" policy. If they discover that you are using one while attached to their network, they may disconnect your service. The best place to experiment with a sniffer is on your own home network that is not connected to the Internet. All you really need is two computers with a crossover cable between them. You can use one as a client, and install server services on the other, such as Telnet, FTP, Web, and mail. Install the sniffer on one or both computers and have fun!

NOTE

You can also download packet traces from numerous websites and read them with your network analyzer to get used to analyzing and interpreting packets. The HoneyNet Project at http://project.honeynet.org has monthly challenges and other data for analysis.

Summary

Network analysis is the key to maintaining an optimized network and detecting security issues. Proactive management can help find issues before they turn into serious problems and cause network downtime or compromise confidential data. In addition to identifying attacks and suspicious activity, you can use your network analyzer data to identify security vulnerabilities and weaknesses and enforce your company's security policy. Sniffer logs can be correlated with IDS, firewall, and router logs to provide evidence for forensics and incident handling. A network analyzer allows you to capture data from the network, packet by packet, decode the information, and view it in an easy to understand format. Network analyzers are easy to find, often free, and easy to use; they are a key part of any administrator's toolbox.

We covered the basics of networking, Ethernet, the OSI model, and hardware that is used in a network architecture. Believe me, we only scratched the surface here. A good networking and protocols reference should be on every administrator's bookshelf. This will come in very handy when you discover some unknown or unusual traffic on your network.

As an administrator, you should also know how to detect the use of sniffers by intruders. You should keep up to date on the methods that intruders use to get around security measures that are meant to protect against sniffing. As always, you will also need to make sure that your computer systems are up to date with patches and security fixes to protect against rootkits and other backdoors.

We also covered a variety of methods used to protect your data from eavesdropping by sniffers. You should always remain up to date on the latest security technologies, encryption algorithms, and authentication processes. Intruders are constantly finding ways to defeat current security practices, thus more powerful methods are developed. A good example is the cracking of the DES encryption scheme and its subsequent replacement with Triple Data Encryption Standard (3DES).

Finally, remember the rule of network analysis—only do it if you have permission. A happy, curious, up-and-coming administrator could easily be mistaken as an intruder. Make sure you have permission or use your own private network to experiment.

Solutions Fast Track

What is Network Analysis and Sniffing?

☑ Network analysis is capturing and decoding network data.

☑ Network analyzers can be hardware or software, and are available both free and commercially.

☑ Network analyzer interfaces usually have three panes: summary, detail, and data.

☑ The five parts of a network analyzer are: hardware, capture driver, buffer, real-time analysis, and decode.

Who Uses Network Analysis?

☑ Administrators use network analysis for troubleshooting network problems, analyzing the performance of a network, and intrusion detection.

☑ When intruders use sniffers, it considered is a passive attack.

☑ Intruders use sniffers mostly to capture user names and passwords, collect confidential data, and map the network.

☑ Sniffers are a common component of a rootkit.

☑ Intruders are using sniffers to control backdoor programs.

How Does it Work?

☑ Ethernet is a shared medium that uses MAC, or hardware, addresses.

☑ The OSI model has seven layers and represents a standard for network communication.

☑ Hubs send out information to all hosts on the segment, creating a shared collision domain.

☑ Switches have one collision domain per port and keep an address table of the MAC addresses that are associated with each port.

☑ Port mirroring is a feature that allows you to sniff on switches.

☑ Switches make sniffing more difficult, however the security measures in switch architectures can be overcome by a number of methods, thus allowing the sniffing of traffic designated for other computers.

Detecting Sniffers

☑ Sometimes sniffers can be detected on local systems by looking for the promiscuous mode flag.

☑ There are several tools available that attempt to detect promiscuous mode by using various methods.

☑ Carefully monitoring your hosts, hub and switch ports, and DNS reverse lookups can assist in detecting sniffers.

☑ Honeypots are a good method to detect intruders on your network who are attempting to use compromised passwords.

☑ Newer sniffers are smart enough to hide themselves from traditional detection techniques.

Protecting Against Sniffers

☑ Switches offer some, but little protection against sniffers.

☑ Encryption is the best method of protecting your data from sniffers.

☑ SSH, SSL/TLS, and IPSEC are all forms of VPNs that operate at various layers of the OSI model.

☑ IPSec tunnel mode can protect the source and destination addresses in the IP header by appending a new header.

Network Analysis and Policy

☑ Make sure you have permission to use a sniffer on a network that is not your own.

☑ Read the appropriate use policies of your ISPs before using a sniffer.

☑ If you are hired to assess a computer network, and plan to use a sniffer, make sure you have some sort of non-disclosure agreements in place, because you may have access to confidential data.

☑ One-time passwords render compromised passwords useless.

☑ E-mail should be protected while in transit and storage with some type of data encryption method.

Frequently Asked Questions

The following Frequently Asked Questions, answered by the authors of this book, are designed to both measure your understanding of the concepts presented in this chapter and to assist you with real-life implementation of these concepts. To have your questions about this chapter answered by the author, browse to **www.syngress.com/solutions** and click on the **"Ask the Author"** form. You will also gain access to thousands of other FAQs at ITFAQnet.com.

Q: I ran a switch flooding program against my switch and it didn't do anything, why not?

A: Some newer switches are resilient to some of the older flooding tools.

Q: I have hubs daisy-chained through the floors of my company's building, is that all one collision domain?

A: Yes! Hubs do not have any intelligence built into them to know where to send data, so they will blindly forward it on to everyone. So every hub that is connected together is seeing traffic for all ports.

Q: When I run Ethereal on my Linux system, I don't see the PROMISC flag in the ifconfig –a output.

A: Ethereal uses the libpcap program to perform packet capturing and filtering. Some newer versions if libpcap use a different method of putting an interface into promiscuous mode that ifconfig cannot detect.

Q: Will adding encryption to my network decrease performance?

A: Yes, encrypting and decrypting data can be resource–intensive, depending on several factors including the type of encryption algorithm and length of the key. However, depending on your network architecture, end users may not notice the difference in performance.

Q: What if an attacker compromises a host that I am using a VPN client on?

A: Your VPN would basically offer a safe and secure environment for the attacker to run wild! For example, you connect your work laptop at home to the Internet over dial-up or high-speed Internet, and your system is compromised via a trojan. Your connections back to the office are secured via a VPN connection which gets enabled once you connect to your mail server or other protected work resources. The attacker then has the ability to access these resources that are otherwise protected by your VPN.

Q: I still don't understand how one-time passwords work.

A: Let me give you an example. You are provided with an RSA Secure ID hardware token. This is a small device that has a screen on it with some numbers that change every sixty seconds. These numbers are your responses to the RSA server challenges, i.e. your password. The token and the server are synchronized, so when you log in, the server presents you with a challenge, i.e. asks you your password, and you type in whatever number is showing on your Secure ID token screen at the time. You will be authenticated for this session, but next time you login it will be a different number, hence a one-time password.