



Global Information Assurance Certification Paper

Copyright SANS Institute
Author Retains Full Rights

This paper is taken from the GIAC directory of certified professionals. Reposting is not permitted without express written permission.

Interested in learning more?

Check out the list of upcoming events offering
"Advanced Incident Response, Threat Hunting, and Digital Forensics (Forensics
at <http://www.giac.org/registration/gcfa>

An Endeavor Down the Forensic Highway

**By
Kristy Westphal**

**GCFA Practical
Submission Attempt #2
Version 1.3 (March 20, 2003)**

**Submission Date:
September 29, 2003**

<u>Abstract</u>	3
<u>Part 1- Analyze an Unknown Binary</u>	3
<u>Binary Details:</u>	3
<u>Key words:</u>	5
<u>Program Description:</u>	5
<u>Forensic Details:</u>	8
<u>Active Binary Analysis:</u>	10
<u>Program Identification:</u>	11
<u>Legal Implications:</u>	14
<u>Interview Questions:</u>	14
<u>Additional Information:</u>	15
<u>Part 2- Option 1 Perform Forensic Analysis on a system (a.k.a. The Case of the Music Manic)</u>	16
<u>Synopsis of Case Facts:</u>	16
<u>System Description:</u>	16
<u>Hardware:</u>	17
<u>Image Media:</u>	17
<u>Media Analysis of System:</u>	19
<u>Review of running processes:</u>	28
<u>Review of Internet history and favorites files:</u>	29
<u>Review of cookies:</u>	31
<u>Review of registry files:</u>	34
<u>Review of most recent files opened:</u>	40
<u>Notable event log entry:</u>	43
<u>Review of the memory image:</u>	45
<u>Timeline Analysis:</u>	46
<u>Recover Deleted Files:</u>	49
<u>String Search:</u>	53
<u>Review of Sniffer Logs:</u>	59
<u>Conclusions:</u>	60
<u>Part 3- Legal Issues of Incident Handling</u>	61
<u>Question A:</u>	61
<u>Question B:</u>	63
<u>Question C:</u>	64
<u>Question D:</u>	64
<u>Question E:</u>	65
<u>Appendix A: Regmon output</u>	66
<u>Appendix B: Filemon output</u>	68
<u>Resources utilized:</u>	70

Abstract

The following document provides a detailed analysis and explanation of three core concepts in the field of computer forensics:

- Analysis of an unknown binary
- A forensic analysis of a computer system
- Analysis and translation of legal issues surrounding incident handling

In this document I apply these concepts to actual systems, provide a detailed description of the how the analysis was performed, and provide conclusions in terms that can be understood by those who do not necessarily have a technical background.

This paper has been updated since its original submission to include more detail in regards to the investigation process itself, the tools and methods that were used to perform the analysis and develop the associated conclusions.

Part 1- Analyze an Unknown Binary

Binary Details:

In this section, a binary file of unknown origin and purpose was analyzed by means of a forensic methodology.

Name of the file: target2.exe

Mactime information:

[Note: Mactime refers to the date and time that a file was Modified, Accessed and Created. The Windows NTFS file system tracks this information on all files.]

To extract this information, a Perl based script called "mac.pl" by Hcarvey was utilized because it is a reliable tool used to gain this type of information. This script prints out the following information (in this order) when run against a directory: File (name of the file), Size, File Owner, Last Access, Last Modification, and Creation date. When run against the unknown binary (target2.exe), the following information was generated:

\\target2.exe,26793,BUILTIN\Administrators,Mon Aug 4 19:05:25 2003,Thu Feb 20 12:45:48 2003,Thu Feb 20 12:45:48 2003

The August access date represents the day I extracted this to disk. This occurred when the file was written out to the file system of my laptop, changing the access date. The zipped file contained the binary in its original state, with an original file date of February 20, 2003, 12:45 p.m. It was difficult to avoid changing the access date, due to the fact that I needed to unzip the file in order to examine it.

File Owner: Unfortunately, it picked up the OS file system permissions, so it looks to be owned by the administrator. It is difficult to avoid this when writing a file to a new file system. The best time to obtain this information is when the file is originally discovered on a compromised system because any movement of the file changes the attributes.

File Size (bytes): 26,793 (this was obtained by reviewing the file attributes in a DOS prompt window. The "dir" command was typed and reveals file attribute information about all files in a directory.) This output matches the results of the mac.pl script seen previously.

MD5: MD5 is an algorithm that "takes as input a message of arbitrary length (like a file) and produces as output a 128-bit 'fingerprint' or 'message digest' of the input". (<http://www.oberon.ethz.ch/ethoberon/defs/MD5.Def.html>) This output should never change unless the file itself has changed. This is a helpful tool to use during an investigation when changes to system files and binaries are suspected because they can be compared to the original file. (There was no MD5 checksum provided with the binary; therefore, it was difficult to guarantee what I worked with was actually not modified from the original file.)

The MD5 signature in this case was: 848903a92843895f3ba7fb77f02f9bf1

Screen Shot of MD5:

```

09/21/2000 04:56p      19,216 TOKENM.SVS
09/21/2000 04:54p      86,016 TOKENMON.EXE
09/20/2000 04:25p      15,713 TOKENMON.HLP
06/15/2001 01:24p      39,424 touch.exe
09/29/2001 04:24p         9,728 tput.exe
06/16/2001 11:27a      34,304 tr.exe
09/12/2001 10:03p         7,168 umount.exe
03/30/2001 11:29a      19,456 uname.exe
06/16/2001 11:27a      25,600 uniq.exe
01/14/2001 02:24a      92,672 unzip.exe
03/24/2002 01:48p      45,672 uptime.exe
04/26/1998 02:21p      49,152 user2sid.exe
03/06/1997 05:08a          4 VIEWDISK.CFG
10/16/2001 02:06p      24,576 volume_dump.exe
03/06/2001 11:16a      40,960 walksam.exe
06/16/2001 11:27a      31,232 wc.exe
01/21/2001 11:15p     130,048 wget.exe
05/08/2001 07:41a         4,608 which.exe
03/30/2001 11:29a      19,456 whoami.exe
03/26/2001 03:25p      81,920 WinError.dll
11/19/2002 12:10p         7,068 WinHex.cfg
06/03/2002 07:04a         1,117 winhex.cnt
06/29/2002 07:04a      574,464 WinHex.exe
04/10/2002 01:46p      233,472 wipe.exe
01/17/2000 03:17p      12,800 xargs.exe
03/30/2001 11:28a      50,176 zcat.exe
01/13/2001 11:40p      58,880 zip.exe
01/21/2002 01:28p     126,976 zlib.dll
04/10/2002 03:18p      51,712 zlibU.dll
04/10/2002 03:18p     191,488 zlibU.pdb

260 File(s)      33,496,212 bytes
2 Dir(s)  21,438,328,832 bytes free

C:\Documents and Settings\kristywestphal\My Documents\windows forensics\win2k_xp
>md5sum target2.exe
848903a92843895f3ba7fb77f02f9bf1 *C:\Documents and Settings\kristywestphal\My
Documents\windows forensics\win2k_xp\target2.exe

C:\Documents and Settings\kristywestphal\My Documents\windows forensics\win2k_xp
>
  
```

As seen above, the tool used to obtain this information was “md5sum” written by George Garner. It’s a Windows based version of the popular md5 algorithm used on Unix platforms. This screen shot was provided to corroborate the MD5 output provided above.

Key words:

The following text strings were extracted from the binary file using the method described in the next section. They appear to be significant because they refer to the author of the program, the protocol and executables that it runs, as well as phrases that might assist in a search for the original code. Key words include:

Spoof, ICMP, Loki, cmd.exe, smsses.exe, reg.exe, “Hello from MFC!”, Backdoor.

Program Description:

In order to analyze the binary, I performed the following steps:

1. Determine the file type
2. Read the contents of the binary file
3. Identify any unknown files
4. Use the Internet to research text strings that are found in the binary file
5. Use a disassembler to even more closely review the code in the binary file
6. Run the program in an isolated environment and identify any changes or affect it may or may not have on a system.

To begin, I ran a command called “file” against the binary (e.g. “file target2.exe”). File is a Unix command that determines general characteristics of the file that you run it against. The results from this were:

target2.exe: MS-DOS executable (EXE), OS/2 or MS Windows

This was the first indication that this was an executable file for the Windows platform.

Next, I wanted to assess what the file did as accurately as possible without executing it. To do this, I used some tools that could display the compiled code in a readable format. If you open the binary with a text editor like Notepad, you would see a hodge podge of letters and numbers that make no sense.

The first tool that I used was bintext from Foundstone. Bintext was used in this situation because it runs on Windows, and bintext is a commonly used program that allows you to read a binary file as if it were written in plain text. The very first line of the code that can be seen when using bintext indicated that it was not able to be run in DOS mode, which again leads me to think that it was a Windows program of some sort. The second hint was the reference to kernel32.dll, as well as some other dlls, which are library types employed on the Windows platform. Then, I came upon the following text:

“impossible to create raw ICMP socket
RAW ICMP SendTo:
Icmp BackDoor V0.1
Code by Spoof. Enjoy yourself!
Your PassWord:
cmd.exe
Exit OK!”

This further convinced me that it was in fact a Windows binary (because of the reference to cmd.exe, or the Command (also referred to as a DOS prompt), and provided me some text strings (words) to look at more closely. These strings can later be used to do searches on the Internet to locate the original code, possibly even the author of the code, and also to better understand what the code does.

Scrolling further down in the bintext view, there were several references to smsses.exe and reg.exe. There was a line quoting “Hello from MFC!”, as well as an Internet address and reference to the C\$ share on 199.107.97.191. The C\$ share is a hidden share on a Windows system that allows you to connect to the root or C drive of that system.

I discovered other strings including. “Local Partners Access”, and “Local Print Manager Service”. It also makes mention of “Open service failed” and “Query Service Failed”. Later on, it refers to the service being disabled and trying to change the way the services are started by a system. This type of reference (based on a google search) seemed to lead towards some type of Print service being manipulated by the executable.

A whois (whois was a way to look up Internet addresses) search on ARIN (www.arin.net or the American Registry of Internet Numbers) revealed that the IP address found in the code belongs to:

OrgName: CERFnet customer - Azusa Pacific University
OrgID: [CCAPU-1](#)
Address: 901 E. Alostia Ave.
City: Azusa
StateProv: CA
PostalCode: 91702
Country: US

NetRange: [199.107.96.0 - 199.107.99.255](#)
CIDR: 199.107.96.0/22
NetName: [CERF-AZUSA](#)
NetHandle: [NET-199-107-96-0-1](#)
Parent: [NET-199-105-0-0-1](#)
NetType: Reassigned

Comment:

RegDate: 1996-08-09

Updated: 1997-10-11

TechHandle: [CERF-HM-ARIN](#)

TechName: AT&T Enhanced Network Services

TechPhone: +1-858-812-5000

TechEmail: notify@attens.com

Having noted the apparent origination of the file, I turned to Internet search pages to see if any of the text strings identified could be located elsewhere. I concentrated initially on the executables: smsses.exe and reg.exe. Using Google (www.google.com), I searched for each individual file. Nothing was found for smsses.exe, but it appears that reg.exe was a part of the Windows Resource kit, which was a collection of administrative tools that Microsoft provides to help system administrators manage Windows servers. Reg.exe helps to automate making changes to the Registry files either locally or remotely. (<http://www.tburke.net/info/suptools/topics/reg.htm>)

Furthermore, searches for the “Hello for MFC!” string revealed that MFC could possibly be Microsoft Foundation Classes, which could mean that this was created by someone learning about programming. Microsoft Foundation Classes is an application framework that may have been used to create this executable.

The string “Code by SpooF” did not yield any substantial hits, even after trying several different search engines. Neither did the string “impossible to create raw ICMP socket” by itself. However, coupled with the phrase “ICMP Backdoor”, it led me to believe that ICMP was the protocol employed to make this binary work. ICMP is typically used as a network troubleshooting protocol, and is not normally seen as a method to backdoor a system (backdoor meaning enabling an uncommon way for someone to access a system, usually for unauthorized purposes).

The strings “Local Partners Access” and “Local Print Manager Service” did not yield any hits specific to the program that I searched for. Plenty of extraneous references, but nothing that got me closer to the program!

It was not until I transferred the file over to a Linux 7.3 system to perform a strings view (issuing the command “strings target2.exe | more”) that I was able to spot the keyword “loki” (loki appeared after the mention of password in the text seen above). This did not show up in the review done using bintext, possibly because it was not interpreted by bintext as an ASCII text string. However, I was not able to spot the IP address in the strings search, or any reference to a reg.exe binary. This proved to me the importance of always using more than one tool to do a similar function to ensure that all the pertinent details are captured.

By definition, “the strings utility looks for ASCII strings in a binary file. A string is any sequence of 4 or more printing characters ending with a newline or a null character.” (<http://docs.sun.com/db/doc/816-0210/6m6nb7mm2?a=view>). The bintext tool specifically looks for ASCII text, Unicode (double byte ANSI) and Resource strings. This means that it looks for regular text, but mis-identified the “loki” string as something else, possibly Unicode text, and therefore didn’t display it to the screen. Unicode text is a form of text that provides a unique number for every character, regardless of the computer platform that it is used on. (<http://www.unicode.org/standard/WhatIsUnicode.html>)

Forensic Details:

The next step in the process was to try and break down the actual compiled code as much as possible to see if any further clues could be revealed. To do this, a disassembler tool called Win32Program Disassembler (full results included with submission) was used. This specific program was used because it is free, and publicly available on the Internet, and provides the functionality required to look at the binary code. The program works as follows:

```
disassem target2.exe > disassembler_info.txt
```

Where disassem is the initial program and target2.exe is the compiled program that we wish to examine. The output of the program is redirected via the “>” character to a file called disassembler_info.txt. I can then look at the output file to review the results.

The disassembler identified the program as having a timestamp of Wednesday, November 27, 2002 23:53:13, which appears to be the last date that it was compiled. A review of the results shows the string “Hello from MFC!” was isolated right away in the analysis, as were the number of imported modules:

Number of Imported Modules = 6 (decimal)

- Import Module 001: KERNEL32.dll
- Import Module 002: ADVAPI32.dll
- Import Module 003: WS2_32.dll
- Import Module 004: MFC42.DLL
- Import Module 005: MSVCRT.dll
- Import Module 006: MSVCP60.dll

These modules are all dynamic link libraries that are utilized in the Windows operating system. Not being completely familiar with what each did, research on the Internet yielded the following:

Kernel32.dll handles memory management, input/output operations, and system interrupts. Essentially, it is the “middle man” between applications and hardware. (http://www.webopedia.com/TERM/K/kernel32_dll.html)

Advapi32.dll (Advanced Windows 32 Base API DLL) Advanced Application Programming Interface (API) services library supporting numerous APIs including many security and registry calls. It is a standard windows system dynamic link library. (<http://www.liutilities.com/products/wintaskspro/dlllibrary/advapi32/>)

Ws2_32.dll (WinSock 2.0 32bit) contains the Windows Sockets API used by most Internet and network applications to handle network connections. It is also a standard system dynamic link library. (http://www.liutilities.com/products/wintaskspro/dlllibrary/ws2_32/)

Msvcrt.dll (Microsoft Visual C++ Run Time library) This library provides some very basic features for programmers, such as tools to compare strings, and do math (sin/cos/etc).
<http://www.fortunecity.com/skyscraper/fortune/570/msvcrt.html>

Mfc42.dll (Microsoft Foundation Classes Version 4.2) This DLL provides an Application Framework, known as MFC (Microsoft Foundation Classes). This framework provides an object oriented, easier programming interface for Windows.
<http://www.fortunecity.com/skyscraper/fortune/570/mfc42.html>

Msvcp60.dll (Microsoft C Runtime Library) contains Standard C programming Library Functions such as printf, memcpy and cos. This is not a standard system dll.
<http://www.liutilities.com/products/wintaskspro/dlllibrary/msvcp60/>

It is important know what these modules do because it helps to paint a picture of how the unknown binary will interact with the operating system. Now that I know it uses common Windows libraries, interfaces with the network somehow, and accesses the registry, I can try to monitor this activity when I execute the program.

Next, the disassembler got into decoding the Assembly code (where it examines the various areas that the program executes) of the program. It was tough to decipher the information; however, I was able to follow the various strings that I identified in the strings searches of the code, so at least I was able to see where each step of the program was utilized.

I was able to pick up a few remnants that I was not able to see before, which made this effort worthwhile. It appeared that there were a few more inputs other than just the password "loki". At one point, it appeared to take user input (at the %s) and issue another option "-i" to run. This can be seen here:

```
:004020F0 8B442404      mov eax, dword[esp+04]
:004020F4 83EC10        sub esp, 010
:004020F7 53            push ebx
:004020F8 33DB          xor ebx, ebx
:004020FA 83F801        cmp eax, 001
:004020FD 55            push ebp
:004020FE 891D34444000  mov dword[00404434], ebx
```

```

:00402104 891D30444000      mov dword[00404430], ebx
:0040210A 0F8EE5000000      jle 004021F5
:00402110 83F803            cmp eax, 003
:00402113 0F85FF000000      jne 00402218
:00402119 56                push esi
:0040211A 57                push edi
:0040211B 8B7C2428          mov edi, dword[esp+28]
:0040211F 8B4708            mov eax, dword[edi+08]
:00402122 50                push eax
:00402123 6890404000        push 00404090
                        (StringData)"%s"
:00402128 6890454000        push 00404590
:0040212D FF15C0304000      call dword[004030C0 ->00003516 sprintf]
                        ;;call MSVCRT.sprintf
:00402133 8B4704            mov eax, dword[edi+04]
:00402136 83C40C            add esp, 00C
:00402139 BEF0414000      mov esi, 004041F0
                        (StringData) "-i"

```

It is interesting to note that no reference was made to either the reg.exe binary or to the text string "password", both seen in the strings search. I assume this is because they are not elements that can be decoded by the disassembler program.

Active Binary Analysis:

In order to study its live activity, I copied the unknown binary to a Windows 2000 workstation that was on an isolated network. I ran Winalysis on the isolated system to get a snapshot of the environment (Winalysis is essentially a change management tool for Windows, gathering information such as the current state of files, registry, services, shares, users and other key system details). Then, I had two programs running in the background: Regmon and Filemon, to see how/if it interacted with the systems registry and files. Both are tools that monitor activity to the registry and to any file on the system, respectively.

Furthermore, a debugger was enlisted to help understand if any activity occurred when the process ran as a service. A debugger will help to step through code to understand what it does while it was executing. In addition to Winalysis, a "netstat -a" was captured with a known good binary, meaning that I used a copy of the netstat program from a cdrom that was created by me with known good version of the tool. Netstat displays open network connections on a system. Sometimes, the attacker can change programs on compromised systems in order to hide their activity. Also on the first run, I turned off my anti-virus software so that the program could run freely and not be stopped if it was identified as a virus.

Next, the program was executed from a DOS prompt. I was disappointed when it appeared that absolutely nothing happened. There was quite a bit of interaction with

both the registry and files on the system, but no ports were opened, nothing appeared to be written to the disk, and no service continued to run. Therefore, the debugger was not employed.

The complete output of regmon for target2.exe activity can be seen in Appendix A. It was significant because it shows exactly what the unknown binary accessed in the registry during the time of its execution. Upon execution, the program tries to access several registry keys in a couple of different hives of the registry that are not there. When the program finds keys that are there, it will open them, query them and then close them. No modification was attempted during execution. Otherwise, no significant findings in the registry were identified.

The complete output of filemon for target2.exe can be seen in Appendix B. It was significant for analysis because it shows all files that were touched by the binary while it was running. The list was fairly short, and it can be seen where the various dll files identified from the disassembler program were accessed. First, it tries several times and successfully opens the WS3_32.dll, in what I suspect was an attempt to access the network. Then it successfully calls the WS2Help.dll again. I assume that it was not executing properly, and then called the help dll for assistance. Then, I noticed that it calls the MFC42.dll and the MSVCP60.dll, but then I noticed another file that was not referenced in the disassembler output: MFC42LOC.dll. Not knowing exactly how this dll operates, I suspect that the MFC42.dll may have called the MFC42LOC.dll to assist, but I was not able to confirm this.

Lastly, it is interesting to note that neither the KERNEL32.dll nor the ADVAPI32.dll files were accessed. I have to wonder if this contributed to the fact that the binary did not appear to function as intended.

When nothing happened, I fired up my sniffer, plugged the NIC into an isolated hub, activated the NIC, and then started my anti-virus software. No interaction with the network or anti-virus software was observed.

The Winalysis snapshot provided no changes worth noting.

As a final step, the binary was renamed from target2.exe to smsses.exe and the process was repeated. No variance from the behavior previously observed was noted. Since I did not have another system with a different version of Windows available to test the binary on, I was not able to test it elsewhere.

Program Identification:

Upon first attempts, I searched the Internet extensively, with no success. I followed every link that I could click from within the pages where I searched for the code. I even searched newsgroups, to no avail. I came very close to finding programs that resembled what this one should have done, as described below:

Found a ping backdoor that was really close:

<http://mcking.8u8.com/hkwz18.htm>

Another really close one:

<http://ouah.kernsh.org/progs/007shell.tgz>

This was also really close:

<http://packetstormsecurity.nl/UNIX/penetration/rootkits/allinone.c>

However, somehow, my original searches neglected a very key piece of information that finally leads me to find an exact match on one of the snippets of text. When I originally searched, I made the mistake of just searching on “ICMP Backdoor” and not the complete line of “ICMP Backdoor V0.1”. When I went this route:

<http://www.google.com/search?hl=en&ie=UTF-8&oe=UTF-8&q=Icmp+BackDoor+V0.1>

I came across a website that was chock full of code, all written in a combination of English and Italian. Sure enough, this one:

http://www.s0ftpj.org/tools/icmp_tunnel.h

Yielded the text strings: impossible to create raw ICMP socket

RAW ICMP SendTo:

Icmp BackDoor V0.1

This was the module that attempted to open the icmp socket in order to tunnel through to another system. But I didn't have the remainder of the program (the parts that were created to run on Windows).

As a last resort, I went to the Asuza University website, saw that there was a honeynet project there, and searched through what was posted on the www.Honeynet.org site in hopes that a match was available.

My initial theory was that this was part of a rootkit. This was based on the fact that I was unable to find the code by itself anywhere. However, after looking at several Windows rootkits, I was unable to locate it in this manner (the rootkits included: Irootkit, tcpip_lib2, alpha_031 and a program called inject).

I also checked virus databases, hoping to find a clue there. I found a reference to a virus that used the reg.exe file: [W32.Alcarys.G@mm](#). Unfortunately, no other reference to ICMP was made.

I did find some references to Loki, the most detail coming from the Network Associates webpage. Though the virus signature that it had was first found in 1992 and didn't seem remotely close to what I was looking for, it did mention that all versions were resident in memory, which reminded me that I had not dumped the memory after I ran

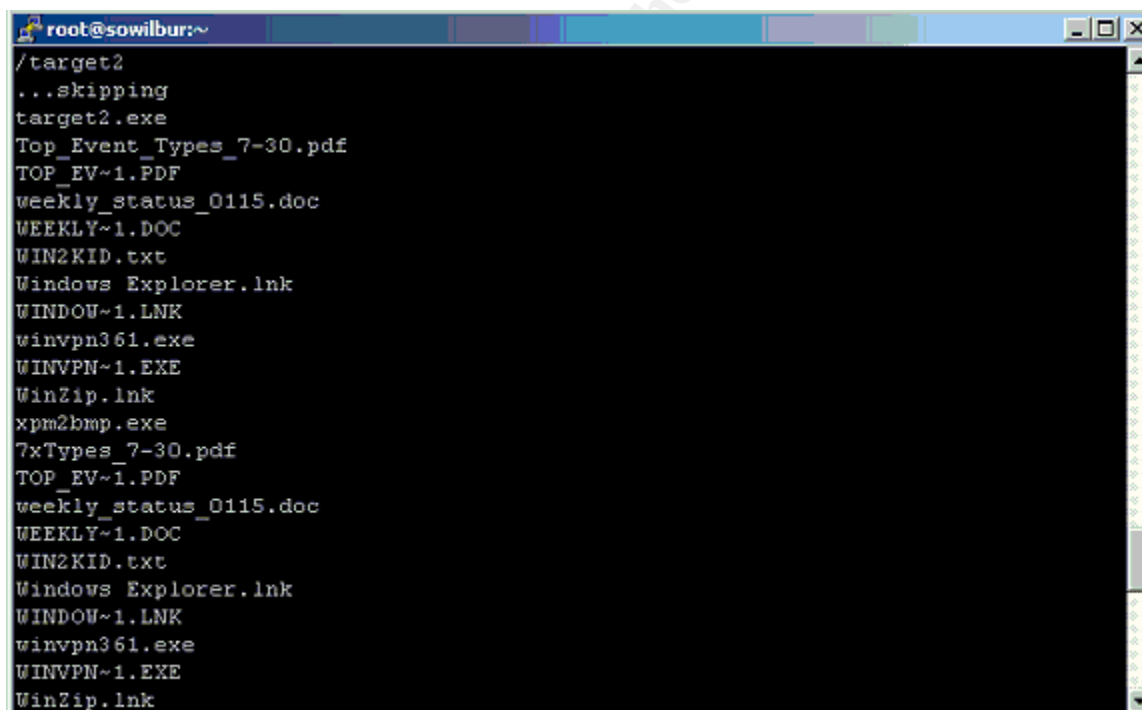
the binary. So I fired up my test Windows 2000 box again to see what I could find. I did a dump of the memory before and after to see if any traces could be identified. This was done by using the dd command (dd is a command that converts and copies a file to the output that you specify. dd originated as a Unix command, and was ported to Windows by George Garner) The command executed was:

```
dd if=\\.\PhysicalMemory of=c:\mem.img
```

Where Physical Memory is what is being read in by the dd command, and sent to output file c:\mem.img.

When I examined the output file, I was interested to discover that the occurrence of target2.exe came up seven times. Then I transferred the image to a Linux box so that I could do an easier analysis with better tools (for some reason I am able to find better tools that happen to run on Linux based operating systems), and looked at the fragments right around where target2 was listed. Uninterestingly enough, it looks like it wasn't actually running, just listed there in amongst other files that were around it on my desktop.

Example of target2.exe in memory image:



```
root@sowilbur:~  
/target2  
...skipping  
target2.exe  
Top_Event_Types_7-30.pdf  
TOP_EV~1.PDF  
weekly_status_0115.doc  
WEEKLY~1.DOC  
WIN2KID.txt  
Windows Explorer.lnk  
WINDOW~1.LNK  
winvpn361.exe  
WINVPN~1.EXE  
WinZip.lnk  
xpm2bmp.exe  
7xTypes_7-30.pdf  
TOP_EV~1.PDF  
weekly_status_0115.doc  
WEEKLY~1.DOC  
WIN2KID.txt  
Windows Explorer.lnk  
WINDOW~1.LNK  
winvpn361.exe  
WINVPN~1.EXE  
WinZip.lnk
```

Based on this analysis, I concluded that target2.exe was a program that attempts to tunnel via the ICMP protocol to a Windows box, in order to be able to execute the cmd prompt. The box that I tried to run it on was probably patched so that the vulnerability or hole that it tries to exploit was not available (I noted that the two dlls that were not

touched when the file was executed to have been patched numerous times on the system that I executed it on).

Legal Implications:

If I could have determined that this program has been executed on the system, there is possible restitution under Federal law. The first determination to make, in this type of situation, was whether this was intentional conduct. If so, under 18 U.S.C. 1030, the maximum penalty for this type of crime for first-time offenders would be a fine and 10 years imprisonment. The maximum penalty for any further incidences would potentially be a fine and 20 years in prison. I was not able to find any penalties for this type of crime under Arizona state law, the state where I reside.

In April 2003, amendments were made to 18 U.S.C. 1030, as well as to the definition of loss in “the case of a computer hack” was expanded to include: the cost of responding to an offense as well as cost in conducting a damage assessment, restoring data, a program or system or information to its condition prior to the offense. Specifically, this helps to address the overall costs of handling an incident rather than just the after-effects.

I should be able to determine if this was executed on a system or not because I have determined what it does and what its fingerprints are. With this information and an examination of the system that it was found on, I will be able to gather enough evidence to say whether it worked or not.

If I determine that no laws were broken, and subsequently I find that this was placed on the computer by someone other than an authorized user on this system, then depending upon where this person was (either external or internal to the company), I might be able to pursue them through an information security policy violation. This person did not have proper authorization (as warned in our system banners), so this would be the first policy violation. Secondly, this was not an approved internal software program, so it also violates our software usage policy. In my company, this behavior can lead to disciplinary action up to termination of employment.

Resources:

<http://www.cybercrime.gov/s&smanual2002.htm>
<http://www4.law.cornell.edu/uscode/18/1030.html>
<http://www.ussc.gov/2003guid/2003amendments.pdf>

Interview Questions:

I think in a real situation, where I was attempting to get cooperation from a suspected attacker, I might try to appeal to their technical intelligence. If I relay to them what I have been able to find so far, they might feel compelled to help me better understand what the code really did; maybe even brag about their accomplishment. This approach

would have to be carried out carefully, though, because it could backfire and leave the suspect thinking that you are merely stupid.

Having laid out what I have been able to prove so far, I would ask the suspect the following:

- 1) What were you trying to do with the code?
- 2) How exactly does it work?
- 3) What is the password?
- 4) How did you get it on the system in the first place?
- 5) Did you put it in startup to run it each time the system boots, or does it only need to run once?
- 6) Why did you pick this system?

Additional Information:

<http://www.oberon.ethz.ch/ethoberon/defs/MD5.Def.html> MD5 definition
<http://web.vip.hr/inga.vip/index.htm> Debuggy (the debugger that I was attempting to use. The debugger program worked, my potentially malicious code did not)
<http://www.geocities.com/~sangcho/disasm.html> Win32Program Disassembler
[http://users.erols.com/gmgarner/forensics/forensic%20acquisition%20utilities-bin-1.0.0.1032%20\(beta1\).zip](http://users.erols.com/gmgarner/forensics/forensic%20acquisition%20utilities-bin-1.0.0.1032%20(beta1).zip) The forensic acquisition utilities written by George Garner.
<http://www.foundstone.com/resources/termsofuse.htm?file=bintext.zip> Link to the bintext utility.

<http://www.tburke.net/info/suptools/topics/reg.htm> (reg.exe description)
<http://docs.sun.com/db/doc/816-0210/6m6nb7mm2?a=view> (Definition of Strings)
<http://www.unicode.org/standard/WhatIsUnicode.html> (Definition of Unicode)
<http://www.winalysis.com/wanaly300.exe> Winalysis evaluation copy

<http://mcking.8u8.com/hkwz18.htm>
<http://ouah.kernsh.org/progs/007shell.tgz>
<http://packetstormsecurity.nl/UNIX/penetration/rootkits/allinone.c>

<http://www.google.com/search?hl=en&ie=UTF-8&oe=UTF-8&q=icmp+BackDoor+V0.1>

http://www.s0ftpj.org/tools/icmp_tunnel.h

Legal Resources:

<http://www.cybercrime.gov/s&smanual2002.htm>
<http://www4.law.cornell.edu/uscode/18/1030.html>
<http://www.ussc.gov/2003guid/2003amendments.pdf>

Part 2- Option 1 Perform Forensic Analysis on a system (a.k.a. The Case of the Music Manic)

Please note that the names and scenario in this section have been sanitized to protect the innocent. The investigation itself is factual.

Synopsis of Case Facts:

It was an average, ordinary day in my work life when I was notified of possible malicious activity in conjunction with a company PC and the Internet. Apparently, someone had been accessing a server external to the company, and transferring proprietary company information to this server. To further complicate the case, no proof had been obtained up until this point, and the entire investigation had to be done without the knowledge of the suspect. I offered to look into it closer to see what I could uncover.

The available facts in this case were minimal. The suspect was going to a server named Exodus, bragging to another co-worker about how he was able to telnet into a main company computer system and then link the two sessions together. This co-worker was the person that reported the activity.

The approach for the investigation includes:

- 1) Monitor Internet activity of the suspect with cooperation from the Network department, for the period of April 7, 2003 to April 16, 2003
- 2) Make a forensic image of the suspect's hard drive and memory in order to take an inventory of activity
- 3) Analyze evidence gathered and make conclusions based on the evidence.

Information of his interactions in and outside the company was reviewed by gathering various artifacts from the system image and the network sniffer activity. Files, registry settings, running tasks, and string searches on both the hard drive and memory images were collected. However, after much analysis and consideration, it appears that the suspect merely enjoyed downloading music from the Internet, working on his own websites hosted outside of the company, and made the occasional effort to get some work completed. Details of this investigation are described in the following sections.

System Description:

The system analyzed in this case was a Windows 2000 Professional workstation (version 5.0.2195 Service Pack 3 Build 2195) that was used as the suspect's daily work computer. The suspect's job entailed managing data for large databases (more of a data processor role than a DBA). In order to get a better look at the suspect's PC, it was taken from the suspect's work area, with the explanation that it needed some maintenance done to it. I had free reign over the computer for 72 hours, as it was seized on a Friday and did not need to be returned until the following Monday. It should be noted that the PC was turned off during the investigation; thus, destroying any

possibility of gathering live evidence in memory or current running processes; however, this was necessary to avoid alerting the suspect of the investigation.

User profiles that had logged in to this system included: the administrator account (directory last written 3/26/03), a fellow co-worker (written 4/11/03), the suspect (written 4/17/03), the generic account for pc technicians (written 3/12/03), and two SMS accounts (written 3/12/03).

Hardware:

Evidence ID	TAG001
Make	Compaq Deskpro EN
Serial Number	6S18-XXXX-XXXX (masked intentionally)
Drives	Fujitsu PB16HE 20GB, 18.6 GB available, Used 3.14 GB, Free 15.4 GB
	A: 3.5" Floppy Drive
	CD-ROM 48x Drive
Memory	128 MG
CPU	133 MHz
Modem?	No
NIC Card	Intel Pro/100 VM
Serial ports	2 9-pin
Printer port	One port
Microphone	One port
Headset	One port

Note: the discrepancy between the total size of the disk (20 GB) and space available (18.6 GB) was due to the installation of the NTFS file system. When an NTFS file system was built, a certain percentage (approximately 12%) is reserved for the operating system. It was in this area where the critical physical disk information was physically located on a disk. This data was stored in files known as metafiles, or files that contain data about data. We can also find the \$MFT file. Known as the Master File Table, this file acts like a table of contents for the info on the disk, holding the reference list of where all data physically resides on the disk.

The original computer was delivered to me by one of our technicians who was aware of the situation details. After the image was made, the copy of the disk remained in my possession, locked in my office when I was not directly using it for analysis. No one else had access to the disk throughout the investigation.

Image Media:

To obtain an image of the system, the seized computer was delivered to me. Being new to the forensic imaging process, I felt the best way that I had available to me at the time was to hook up power, keyboard, mouse, and monitor (realizing that this will change the image somewhat with new drivers applied), and power the system on. The

preferred method to image the disk would have been to hook it up as a secondary disk via SCSI cable and then image it. I also chose this method because at the time because I did not have the proper cables available to follow the preferred method.

Once the system was booted up, I logged into the computer with an administrator account and assigned an IP address to the NIC to that of a private subnet (192.168.0.15). A crossover cable was connected to the suspect's CPU and the other end connected to my forensic laptop. At this point, a CD-ROM with the forensic tools was placed in the suspect's read-only drive.

My forensic laptop was a Windows 2000 Dell computer with a USB 2.0 external hard drive. The hard drive had been sterilized via the use of two data wiping tools: first, the drive was connected to a Linux image in the Dell laptop, and the command "dd" was used to destroy any possible data that was on the drive (command issued was "dd if=/dev/zero of=/dev/sda1". Note: the disk was one large partition, so this command was sufficient to capture the entire physical disk). (Note #2: dd was a command that converts and copies a file to the output that you specify. dd originated as a Unix command, but was ported to Windows by George Garner).

dd was utilized as the image tool (both here and to make a disk copy) because it is a trusted and well-known imaging tool in the forensic community. Then, to make sure that the drive was completely clean, the drive was connected to the Windows 2000 image, where the tool "wipe" (from George Garner) was used to overwrite whatever may have been left. The format of the "wipe" command:

wipe \\.\D:

Where \\.\D: refers to the logical drive that my USB disk came up as when attached to my forensic laptop. Again, being new to the forensic process, I did not feel comfortable using \\.\PhysicalDrive# to wipe the disk, as I was not completely clear how this command might react if accidentally pointed to the wrong drive. Since this disk was one large partition, I felt it most timely and the best option available at the time to use the logical reference.

Feeling confident that I had a clean disk to work with, while still on the Windows 2000 image on my forensic laptop, the forensic netcat tool was employed to open a listening port ("nc -v -n -l -p 33333 -k md5 -verify -sparse -O suspectmem.img"). The options on the netcat command are as follows:

- v for verbosity.
- n numeric only IP addresses, no DNS resolution.
- l listen mode (for inbound connections).
- p 33333 was the TCP port that netcat was listening on.
- k md5 indicates the hash algorithm that should be used on the image to compute the checksum.
- verify verifies every file copied in the image and creates an md5 checksum of each

-sparse for sparse file support (files that contain large amounts of zero data)
-O suspectmem.img was the name of the output file

I captured what was loaded into memory first regardless of the fact that I had powered on the box myself. In order to be thorough, I wanted to make sure that the suspect hadn't loaded anything on the system that might have been running in memory.

On the suspect machine, I opened a cmd prompt, cd'd to the CD-ROM (drive D:) then from the forensic cd ran the netcat command with the following parameters:

```
nc -v -n -k md5 -l \\.\PhysicalMemory 192.168.0.10 33333
```

Where 192.168.0.10 was the IP address of my Dell forensic laptop. This command actually redirects its output to my forensic laptop through port 33333, which it was listening on due to the previous command issued. The "-l" in this case stands for the source of what I wanted to transfer, this being the actual physical memory.

This process was repeated for the hard disk, where PhysicalMemory was replaced by C:, and the output file was called suspectdisk.img. Being a 20GB drive, this process took approximately nine hours to complete. I chose to do a logical image mostly out of ignorance (as a physical image was often preferred method to make the image with). When this investigation was conducted I was still very new to the procedure and did what first came to mind. Unfortunately, this mistake did not occur to me until long after the disk was returned to the suspect.

Once the image process was completed, the system was returned to its original configuration, including the following steps:

- 1) change the network card settings back to obtain address automatically through DHCP
- 2) recent document listing was cleared because after the image was made, I took a look around for other items of interest that I needed to gather for my investigation
- 3) my forensic tool cd was removed from the drive

After the imaging process was over, I realized an error: I failed to use the cmd.exe from the cd. This was duly noted as I moved along in the investigation.

During the investigation, I realized yet another mistake: I failed to collect much of the volatile information, such as the open ports via the command "netstat -a". I believe that I was fairly focused on the making of the image and failed to collect this information, which would have been very useful later on.

Media Analysis of System:

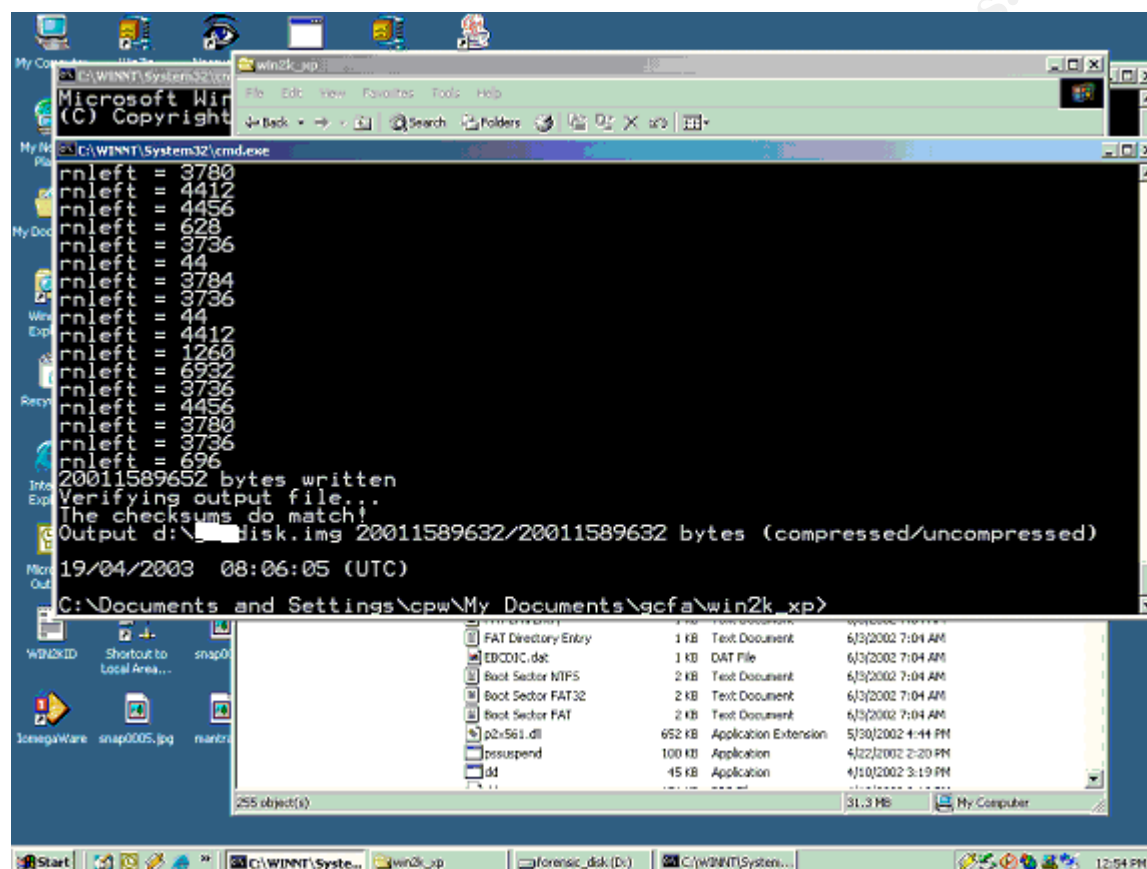
The integrity of the physical memory and the disk were confirmed by using the "-k md5" parameters from the netcat command during the disk copy. These files were retained as a record keeping measure. If required, I could use these signatures to verify that I

had not changed the images in any way during the course of my investigation. This was best used when you keep the seized original disk to make comparisons to the image copies that have been made. The MD5 signatures were:

Disk MD5 a5ee74ea1fe01926b46db2cbbecf4bff

Memory MD5 889abee36403fdf2abf9e1e94cc0e278

Screen shot of the disk image checksum output to corroborate signatures:



The media analysis of this disk was conducted using Autopsy, a GUI front-end for the forensic tool TASK (The Sleuth Kit) developed by Brian Carrier of @stake, and through use of the command line tools provided in TASK sans GUI. These tools were selected because 1) they are freely available and more importantly 2) they are well-known tools that have proven themselves in the forensic community. Autopsy makes data analysis much more efficient and easier to do because the information is displayed in an easy-to-manuever and read format, and would have significantly shortened my analysis time had I been able to get it to work properly.

I discovered a problem with Autopsy and my disk image in the middle of this project. When I made my image of the original system, I used the external USB disk drive to store the copy. When I mounted the USB disk to my Linux system, I was unable to properly view the image within Autopsy to see the complete listing of directories that

existed on the disk image. At first, I thought that I had made an incomplete image and merely had not copied the entire disk somehow (possibly the result of making the logical image). This was not the case, though, as when I created another file listing of the entire image using a different tool (see the timeline analysis section), directories that I could not view in Autopsy were indeed available.

This truly perplexed me for a while, until I realized that I could simply use the command line tools to examine the image. I diagnosed the problem as being related to the fact that I was mounting the USB disk drive, and then trying to further mount the image on the disk drive, which didn't seem to work. So, when I originally mounted the external disk, I used the command "mount /dev/sda1 /mnt/usbhd". Now on the mounted partition /mnt/usbhd, a file existed there called suspectdisk.img. This was my disk image. Using Autopsy, I pointed the image to examine to /mnt/usbhd/suspectdisk.img. This worked partially, but I was only able to view the Master File Table and a couple of regular directories on the disk.

Then, I tried to mount just the disk image by using the command: "mount /mnt/usbhd/suspectdisk.img /mnt/image", but the operating system would not let me do this. After trying to install the latest Linux NTFS driver and several other techniques, none of which worked, I thought that it might be a limitation of my Linux 7.3 kernel. I believe that I would have had to update to Linux 8.0 and I did not have the time to do this. So, I worked with the command line tools instead.

To understand what I was experiencing, here are the screen shots of the image from the root directory.

Image, Part 1

FILE ANALYSIS KEYWORD SEARCH FILE TYPE IMAGE DETAILS META DATA DATA UNIT HELP CLOSE									
/mnt/usbhd/ ADD NOTE GENERATE MD5 LIST OF FILES									
ALL DELETED FILES									
EXPAND DIRECTORIES									
DEL	Type dir / in	NAME	WRITTEN	ACCESSED	CHANGED	Size	UID	GID	META
	r /r	\$AllDef	2003.03.12 18:56:56 (GMT)	2003.03.12 18:56:56 (GMT)	2003.03.12 18:56:56 (GMT)	2560	48	0	4-128-4
	r /r	\$BadClus	2003.03.12 18:56:56 (GMT)	2003.03.12 18:56:56 (GMT)	2003.03.12 18:56:56 (GMT)	0	0	0	8-128-2
	r /r	\$BadClus:\$Bad	2003.03.12 18:56:56 (GMT)	2003.03.12 18:56:56 (GMT)	2003.03.12 18:56:56 (GMT)	2831720448	0	0	8-128-1
	r /r	\$Bitmap	2003.03.12 18:56:56 (GMT)	2003.03.12 18:56:56 (GMT)	2003.03.12 18:56:56 (GMT)	610712	0	0	6-128-1
	r /r	\$Error	2003.03.12 18:56:56 (GMT)	2003.03.12 18:56:56 (GMT)	2003.03.12 18:56:56 (GMT)	8192	48	0	7-128-1
	d /d	\$Extend/	2003.03.12 18:56:56 (GMT)	2003.03.12 18:56:56 (GMT)	2003.03.12 18:56:56 (GMT)	344	0	0	11-144-4
	r /r	\$LogFile	2003.03.12 18:56:56 (GMT)	2003.03.12 18:56:56 (GMT)	2003.03.12 18:56:56 (GMT)	67108864	0	0	2-128-1
	r /r	\$MFT	2003.03.12 18:56:56 (GMT)	2003.03.12 18:56:56 (GMT)	2003.03.12 18:56:56 (GMT)	26670080	0	0	0-128-1
	r /r	\$MFTMirr	2003.03.12 18:56:56 (GMT)	2003.03.12 18:56:56 (GMT)	2003.03.12 18:56:56 (GMT)	4096	0	0	1-128-1
	r /r	\$Secure:\$SDH	2003.03.12 18:56:56 (GMT)	2003.03.12 18:56:56 (GMT)	2003.03.12 18:56:56 (GMT)	336	0	0	9-144-1
	r /r	\$Secure:\$SDS	2003.03.12 18:56:56 (GMT)	2003.03.12 18:56:56 (GMT)	2003.03.12 18:56:56 (GMT)	336936	0	0	9-128-0

This screen shot shows the first page of what I could see using the Autopsy tool. The \$filename format files are all metadata files that make up the initial sectors of a physical disk. It was noted that this listing showed a very large \$BadClus:\$Bad file, which usually lists bad sectors on the disk. However, if this disk truly had that many bad clusters, then nothing else would have existed on this disk. I suspected that this was where the remainder of the information on the disk that I was not able to access was located.

The \$MFT and \$Secure metadata files can also be seen here. The \$MFT was the Master File Table, or main table of contents for every physical file location on the disk, and the \$Secure files hold security information for the disk. Metadata files are files that you would not see on the disk normally, but only through some sort of forensic analysis such as this. They are underlying disk files that hold "data about data".

Image, Part 2

FILE ANALYSIS KEYWORD SEARCH FILE TYPE IMAGE DETAILS META DATA DATA UNIT HELP CLOSE									
ALL DELETED FILES									
	r/r	iSecur:88D3	2003.03.12	2003.03.12	2003.03.12	336936	0	0	9-128-0
			18:56:56 (GMT)	18:56:56 (GMT)	18:56:56 (GMT)				
EXPAND DIRECTORIES									
	r/r	iSecur:8811	2003.03.12	2003.03.12	2003.03.12	56	0	0	9-144-16
			18:56:56 (GMT)	18:56:56 (GMT)	18:56:56 (GMT)				
	r/r	iUpCase	2003.03.12	2003.03.12	2003.03.12	131072	0	0	10-128-1
			18:56:56 (GMT)	18:56:56 (GMT)	18:56:56 (GMT)				
	r/r	iVolume	2003.03.12	2003.03.12	2003.03.12	0	48	0	3-128-3
			18:56:56 (GMT)	18:56:56 (GMT)	18:56:56 (GMT)				
	d/d	_f	2003.04.18	2003.04.19	2003.04.19	168	48	0	5-144-6
			05:03:06 (GMT)	01:47:28 (GMT)	01:47:23 (GMT)				
	r/r	accids.exe	2002.07.23	2003.04.10	2003.04.09	150528	0	0	2693-128-3
			02:05:04 (GMT)	02:00:36 (GMT)	03:40:52 (GMT)				
	r/r	accsetup.exe	2002.07.23	2003.04.10	2003.04.09	163840	0	0	2694-128-3
			02:05:04 (GMT)	02:00:36 (GMT)	03:40:52 (GMT)				
	r/r	AUTOEXEC.BAT	2003.03.13	2003.03.13	2003.03.26	0	0	0	4490-128-1
			02:16:04 (GMT)	02:16:04 (GMT)	01:14:54 (GMT)				
	r/r	boot.ini	2003.03.26	2003.04.15	2003.03.26	192	0	0	2724-128-10
			01:14:46 (GMT)	01:00:24 (GMT)	01:14:54 (GMT)				
	d/d	Compaq/	2003.03.13	2003.04.18	2003.03.13	456	0	0	6818-144-1
			03:43:20 (GMT)	05:02:17 (GMT)	03:43:20 (GMT)				
	r/r	CONFIG.SYS	2003.03.13	2003.03.13	2003.03.26	0	0	0	4489-128-1
			02:16:04 (GMT)	02:16:04 (GMT)	01:14:54 (GMT)				
	d/d	Documents and Settings/	2003.04.11	2003.04.19	2003.04.11	56	0	0	2731-144-7
			02:55:54 (GMT)	01:47:28 (GMT)	02:55:54 (GMT)				
	✓ r/r	ffastun.ffe	2003.04.18	2003.04.18	2003.04.18	4843	0	0	4106-128-3
			05:03:06 (GMT)	05:03:10 (GMT)	05:03:06 (GMT)				

This was the continuation of what I was able to view through Autopsy. Here, you can see the last of the metadata files and the beginning of the root directories. You can see the first few executables, the autoexec.bat file, and the Compaq directory. The last directory that was seen here, Documents and Settings, was the last directory that can be seen in this listing of the root directory. Normally, you would be able to see all of the other directories in a Windows root system, such as WINNT, Program Files, and in the case of this specific image, Inoculan (my company's anti-virus software). Instead, as can be seen above, the listing of deleted files begins, as denoted in red.

© SANS Institute

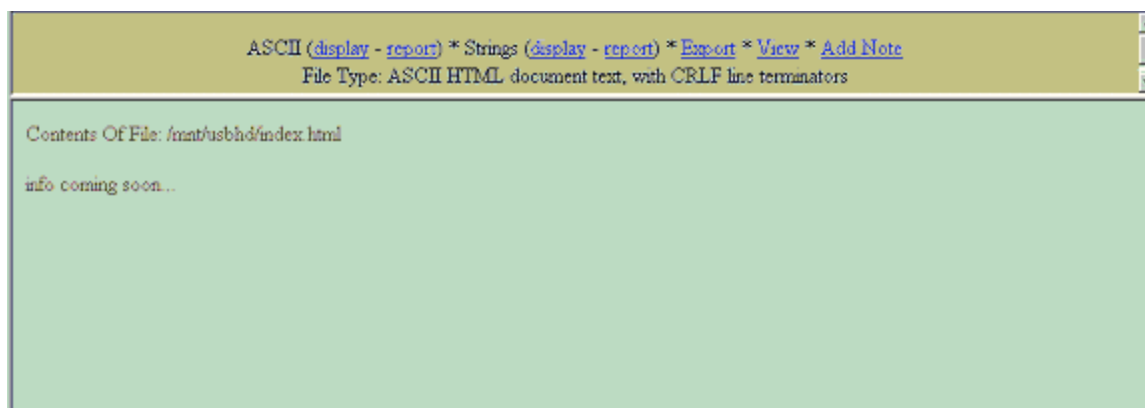
Image, Part 3

FILE ANALYSIS KEYWORD SEARCH FILE TYPE IMAGE DETAILS META DATA DATA UNIT HELP CLOSE									
ALL DELETED FILES									
EXPAND DIRECTORIES	d/d	Documents and Settings/	2003.04.11 02:55:54 (GMT)	2003.04.19 01:47:28 (GMT)	2003.04.11 02:55:54 (GMT)	56	0	0	2731-144-7
	✓ r/r	ffastun.ffe	2003.04.18 05:03:06 (GMT)	2003.04.18 05:03:10 (GMT)	2003.04.18 05:03:06 (GMT)	4843	0	0	4106-128-3 (realloc)
	✓ r/r	ffastun.ffi	2003.04.18 05:03:06 (GMT)	2003.04.18 05:03:10 (GMT)	2003.04.18 05:03:06 (GMT)	106496	0	0	23466-128-3 (realloc)
	✓ r/r	ffastun.ifo	2003.04.18 06:07:07 (GMT)	2003.04.18 06:07:10 (GMT)	2003.04.18 06:07:07 (GMT)	102444	0	0	6688-128-3 (realloc)
	✓ r/r	ffastun.ifo	2003.04.18 05:03:06 (GMT)	2003.04.18 05:03:10 (GMT)	2003.04.18 05:03:06 (GMT)	118784	0	0	25406-128-3 (realloc)
	✓ r/r	ffastun0.fff	2003.04.17 05:01:32 (GMT)	2003.04.18 05:36:16 (GMT)	2003.04.17 05:01:32 (GMT)	402	0	0	6689-128-1 (realloc)
	✓ r/r	ffastun0.fff	2003.04.18 05:03:06 (GMT)	2003.04.18 05:03:10 (GMT)	2003.04.18 05:03:06 (GMT)	446464	0	0	25408-128-3 (realloc)
	✓ r/r	index.html	2003.04.08 04:07:01 (GMT)	2003.04.09 03:04:24 (GMT)	2003.04.08 06:06:49 (GMT)	313	0	0	21939-128-1 (realloc)
	✓ r/r	INCSSETUP.LOG	2003.03.21 04:19:01 (GMT)	2003.03.29 01:05:02 (GMT)	2003.03.21 04:19:01 (GMT)	70	0	0	10310-128-1 (realloc)
	✓ r/r	INCSSETUP.LOG	2003.03.21 04:19:01 (GMT)	2003.03.29 01:05:02 (GMT)	2003.03.21 04:19:01 (GMT)	70	0	0	10310-128-1

This is the third of four screens and is a continuation of the previous screen. This picks up where the previous screen left off, showing more of the deleted files in the root directory of the file system. One item of note that was seen in this picture was the index.html file. This was interesting first off because it has been deleted. Secondly, it was interesting because it was found here, where normally, the index.html file, representing the content for a website, is located in the directory where other website files can be found. None of these files are seen here.

When I take a closer look at the contents of this file (and luckily, it has not been overwritten by another file), I see:


© SANS Institute



This turned out to be a clue as to what activity he was really up to, as was discovered later in my analysis. This file turns out to be the main page for a website that my suspect was maintaining.

© SANS Institute 2003, All rights reserved.

Image, Part 4

FILE ANALYSIS KEYWORD SEARCH FILE TYPE IMAGE DETAILS META DATA DATA UNIT HELP CLOSE										
										
ALL DELETED FILES	✓	r/r	index.html	2003.04.08 04:07:01 (GMT)	2003.04.09 03:04:24 (GMT)	2003.04.08 06:06:49 (GMT)	313	0	0	21939- 128-1 (realloc)
EXPAND DIRECTORIES	✓	r/r	INOSSETUP.LOG	2003.03.21 04:19:01 (GMT)	2003.03.29 01:05:02 (GMT)	2003.03.21 04:19:01 (GMT)	70	0	0	10310- 128-1 (realloc)
	✓	r/r	INOSSETUP.LOG	2003.03.21 04:19:01 (GMT)	2003.03.29 01:05:02 (GMT)	2003.03.21 04:19:01 (GMT)	70	0	0	10310- 128-1 (realloc)
	✓	r/r	NTDETECT.COM	2003.03.13 02:40:40 (GMT)	2003.03.13 02:40:40 (GMT)	2003.03.13 02:40:40 (GMT)	34724	0	0	2699- 128-4 (realloc)
	✓	r/r	NTDETECT.COM	2003.03.13 02:40:40 (GMT)	2003.03.13 02:40:40 (GMT)	2003.03.13 02:40:40 (GMT)	34724	0	0	2699- 128-4 (realloc)
	✓	r/r	ntldr	2003.03.13 02:40:40 (GMT)	2003.03.13 02:40:40 (GMT)	2003.03.13 02:40:40 (GMT)	214432	0	0	2695- 128-4 (realloc)
	✓	r/r	ntldr	2003.03.13 02:40:40 (GMT)	2003.03.13 02:40:40 (GMT)	2003.03.13 02:40:40 (GMT)	214432	0	0	2695- 128-4 (realloc)
	✓	r/r	pagefile.sys	2003.04.19 01:47:18 (GMT)	2003.04.19 01:47:18 (GMT)	2003.04.19 01:47:18 (GMT)	201326592	0	0	24-128- 1 (realloc)
	✓	r/r	pagefile.sys	2003.04.19 01:47:18 (GMT)	2003.04.19 01:47:18 (GMT)	2003.04.19 01:47:18 (GMT)	201326592	0	0	24-128- 1 (realloc)

This was the fourth and final screen shot of the root directory. It shows the last entry seen in root. As displayed, the final files in this directory have been deleted and the cluster reallocated for use of other data (denoted by the (realloc) in the last column on the right). When data was deleted on a disk, the entry in the Master File Table was changed to reflect this, but unless that data in the specific cluster on the disk was overwritten with new data, it was not fully deleted. In this case, the clusters on the disk were marked with a “realloc” label for reuse, but since the files that were written to them still have size (indicated by the number in the 7th column), they still existed and could be read if necessary.

After I recognized and moved on from the Autopsy issue, a general review of the files was completed. The only serious modifications to the operating system appeared to be update files from Microsoft Update. There were no suspicious entries found that indicated that the operating system had been modified, and no real indication that any rogue programs had been installed. I base this conclusion on my general review of the disk, and that I saw no evidence that any malicious files had been downloaded and installed, other than the ones that I was able to identify. Details of this analysis are in the following paragraphs.

Shares available on computer: ADMIN\$, C\$, CHEYUPD\$ (Anti-Virus software), and IPC\$

A review of installed programs took place by initially looking at the Start Menu program listing. Unauthorized software (other than our normal corporate image) was discovered immediately. The unauthorized programs installed included the following:

The Mozilla Browser- Phoenix version

Winamp (an mp3 player)

Putty (a freeware ssh client. Ssh stands for secure shell, and serves as an encrypted version of the telnet program)

D-Shock (another mp3 player)

In his startup menu, the following programs were noted:

A common fax program used internally to our company

Anti-Virus Software

Live Menu

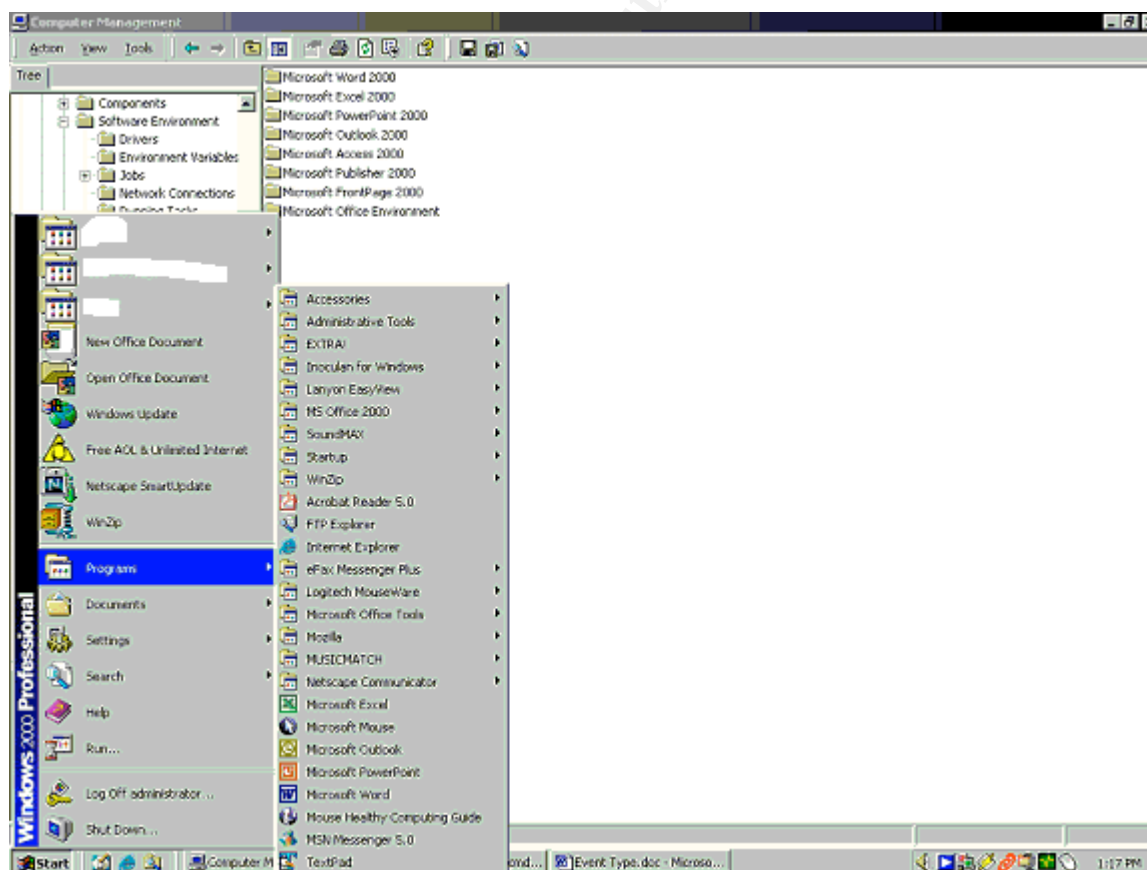
Microsoft Fast Find

Microsoft Office Task Bar

Office Startup

Winzip Quick Pick

Complete program listing:

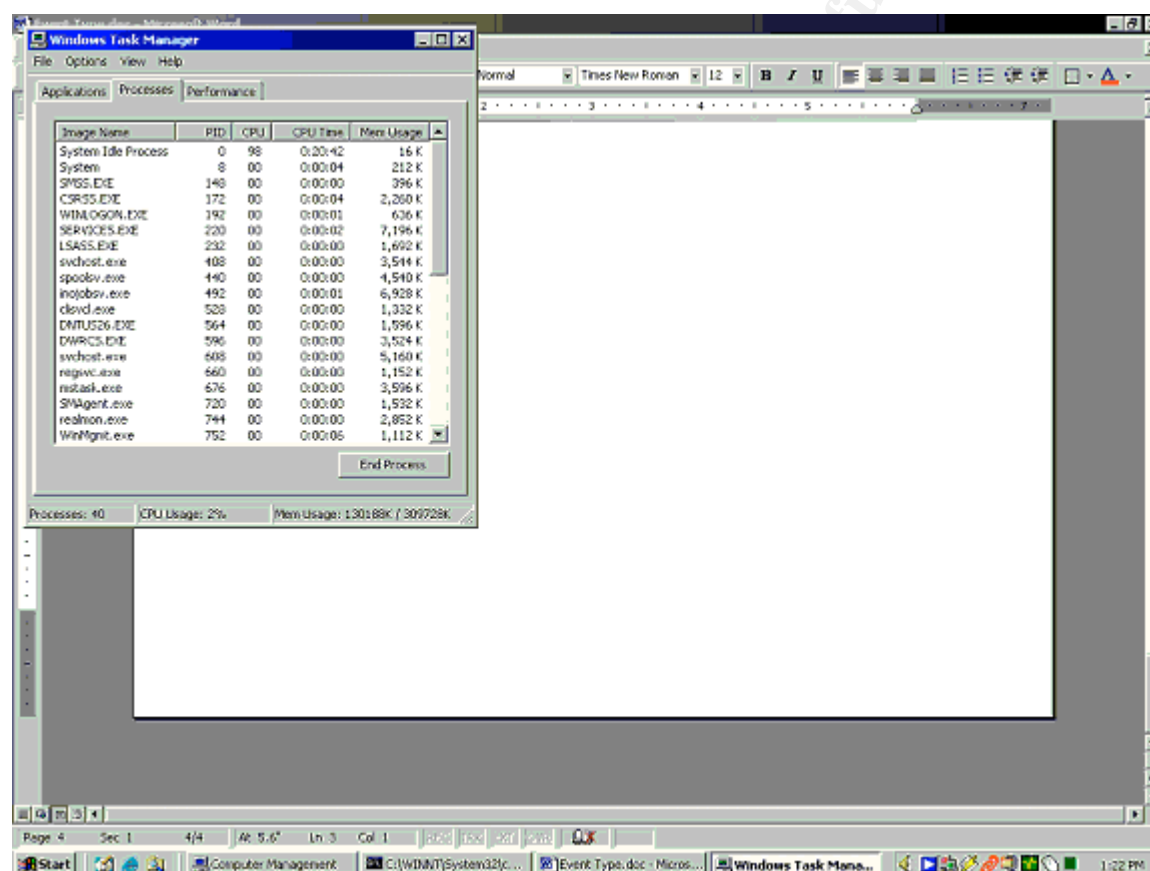


A more thorough review of all files and programs was completed during the timeline analysis, detailed in a later section.

Review of running processes:

The method used to review running process was to examine the files listed in the Windows Task Manager. Since I had originally failed to capture a listing of open network ports, this analysis of the Task Manager processes may not be complete, but it was the best information available at this time.

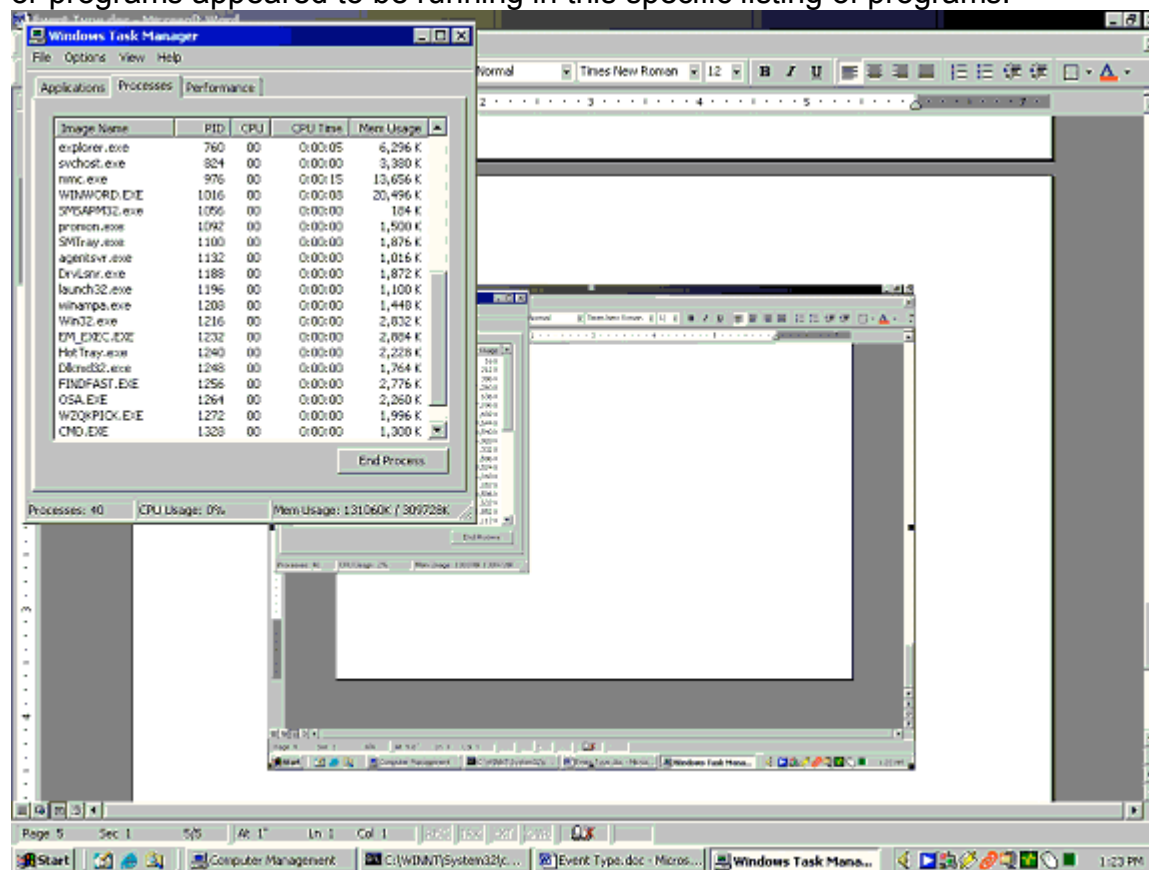
Screen shots from Task Manager:



This screen shot captures the first set of processes that could be seen in the Task Manager prior to scrolling down in the Window. This screen shot was taken after the original system had been imaged, and was taken on the suspect's original system, not the image. These were taken afterwards so that I wouldn't alter the image any more than I already had by booting it.

As seen here, the System Idle process tops the list, using approximately 98% of the CPU. This was normal system usage for a desktop that was not being heavily utilized

nor has any type of process running out of control. A comparison of the processes on the suspect's PC with a known, clean, similar system showed that no rogue processes or programs appeared to be running in this specific listing of programs.



Task Manager, Part 2 (please note that there were 2 other process running that were not captured in the screen shots: SMSMon32.exe and taskmgr.exe). This was the continuation of processes in the Task Manager that can be seen once the scroll bar on the window was pulled downwards. Other than the two processes listed previously, all other running processes were captured in this screen shot.

Again, a comparison of processes with a known, clean, similar system was conducted, and no unusual processes or programs were found to be running.

It should be noted that without the open network port information, the conclusion that no rogue or unusual processes were running on the system couldn't be 100% verified. It could be that a process could be running, hidden from view in the native task manager. However, no signs of rogue processes were found elsewhere, so it was unlikely that Windows processes would run in this manner.

Review of Internet history and favorites files:

Unlike the suspect's cookie information in the next section, snapshots of the suspect's Internet History were not taken when the computer was first seized. So, I had to take the long route and recover them from the disk image using the icat program.

The icat program is a tool that comes with TASK. Its purpose is to read data in the inodes of a disk. An inode (as defined by Hyperdictionary.com, <http://www.hyperdictionary.com/computing/inode>) is "a data structure holding information about files in a Unix file system. There is an inode for each file and a file is uniquely identified by the file system on which it resides and its inode number on that system". In order to get the inode number required by the icat command, I had to use another program called fls (also from TASK) to list the files on the disk image, with their inode numbers. The command issued to obtain this file listing is as follows:

```
fls -f ntfs -m / -r /mnt/usbhd/suspectdisk.img > fls_all_files
```

Translation of this command:

-f ntfs refers to the type of file system that fls was to make its listing from. This program can be used on other types of file systems like Unix or FAT.

-m refers to the mount point that I would like fls to organize the files that it outputs.

Using the "/" will have the files listed starting with a "/".

-r indicates that we want a recursive file listing of the image that fls will be examining.

This means that I would like to examine subdirectories as well.

/mnt/usbhd/suspectdisk.img was the full path to where the image was located.

> fls_all_files was the output file where I would like the file listing to be written to.

Once this file was created, I can read the file and look for the specific files that I want to view with the icat command. When I looked in the fls file that I had created of all files on the disk, I found there to be eight index.dat files (inode numbers: 6588, 17416, 21475, 23353, 22074, 24163, 25511, 25693). Index.dat files hold the Internet usage history information for a user, and were located on the disk in a directory under the user's Documents and Settings directory. When a user is created on, or logs into, a Windows system, by default, a directory is created under the root Documents and Settings directory by their user name. In this case, I found these files in the \Documents and Setting\suspect\Local Settings\History\History.IE5.

I noted their inode numbers, and with the help of icat recovered all eight files.

The syntax of the icat command is as follows:

```
icat -f ntfs /mnt/usbhd/suspectdisk.img '6588' > index1.dat
```

Translation of this command:

-f ntfs identifies the type of file system that icat will examine

/mnt/usbhd/suspectdisk.img was the full path to the filesystem image being examined

'6588' was the inode number that we are trying to read. This number was replaced with the respective inode numbers and the command is executed for each one separately. > index1.dat was the file where I redirected the output of this command to. This was changed to reflect a new file output name each time it was executed.

There were quite a few entries in each to weed through, but I was able to do so with the help of the Internet Explorer History Viewer (by Scott Ponder, Phillips Ponder Company, <http://www.phillipsponder.com/orderForm.htm>). This tool imports each index.dat file and reads it like plain text. This tool was used because it was exclusively written to read Internet Explorer History files, is also well known and proven within the forensic community. The same information that this tool provides can be verified by using the strings command; however, the data in the index.dat files was much more cleanly presented (and easier to read) using this tool as opposed to a strings review. Highlights included:

- Suspect visited his email account at hotmail, freeshell, earthlink and yahoo quite frequently.
- Numerous and varied news portals, message boards and music sites.
- Google used frequently
- Visited a site at bittersweetirony.com (this site was for a band, which our suspect happens to be a member of and maintains himself. Throughout this review, more and more activity came back to this site, and the previous screen shot "info coming soon" was also found on the site).
- Visited a lot of sites actually related to his job
- Suspect visited his bank website
- Visited the Exodus website where his weblog is stored
- A lot of food related sites
- A lot of entertainment (movies) sites
- Even a visit to my intranet security site! (I maintain an internal company site related to security).

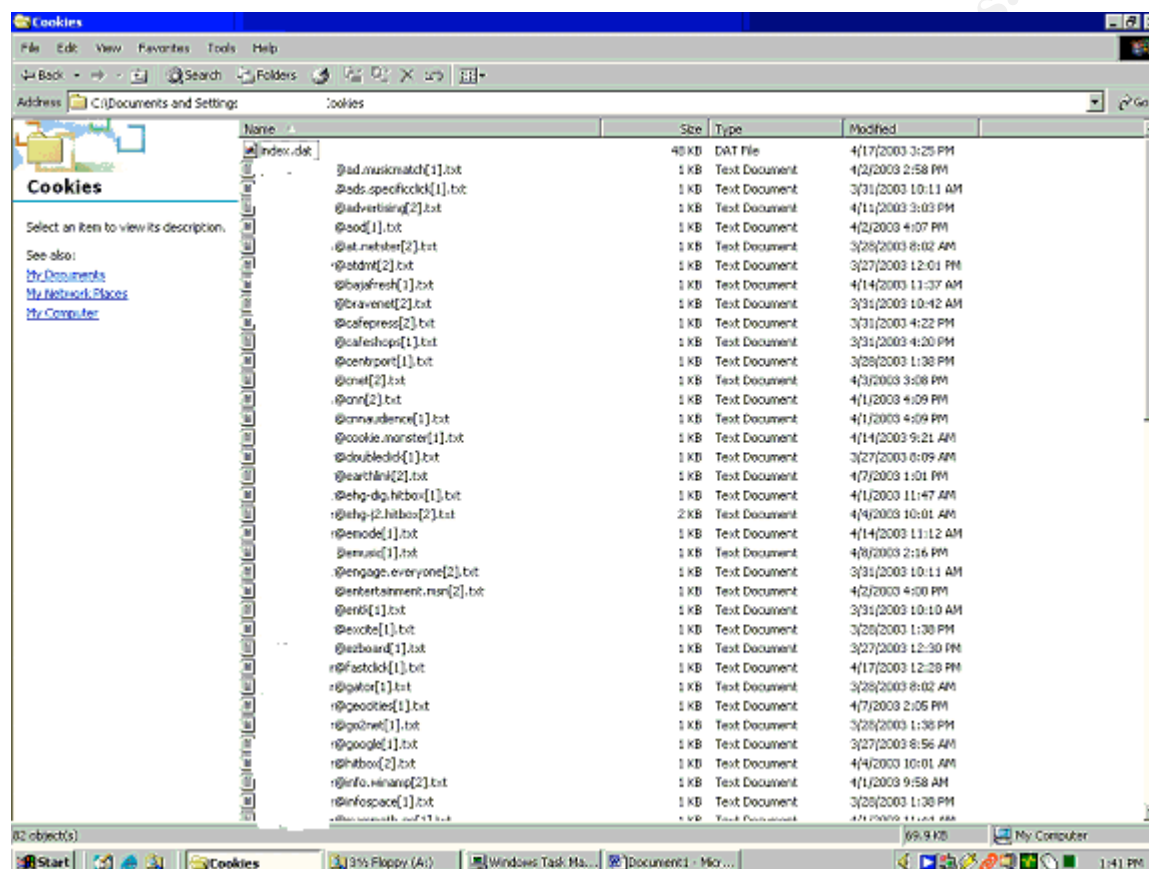
Other than this wide array of categories, nothing terribly alarming came from the history file review. I did follow up on his activity in his weblog, where he complained about a wide variety of things in both his personal and work life, as well as direct complaints about the company.

Review of cookies:

A review of the suspect's cookie files did not reveal anything inconsistent with the review of his Internet history. A cookie, per the definition from Webopedia.com (<http://www.webopedia.com/TERM/c/cookie.html>) was "a message given to a Web browser by a Web server. The browser stores the message in a text file. The message is then sent back to the server each time the browser requests a page from the server. The main purpose of cookies is to identify users and possibly prepare customized Web pages for them." My suspect's cookies were taken directly from the directory that they were stored in on his harddrive: c:\documents and settings\suspect's_user_name\cookies.

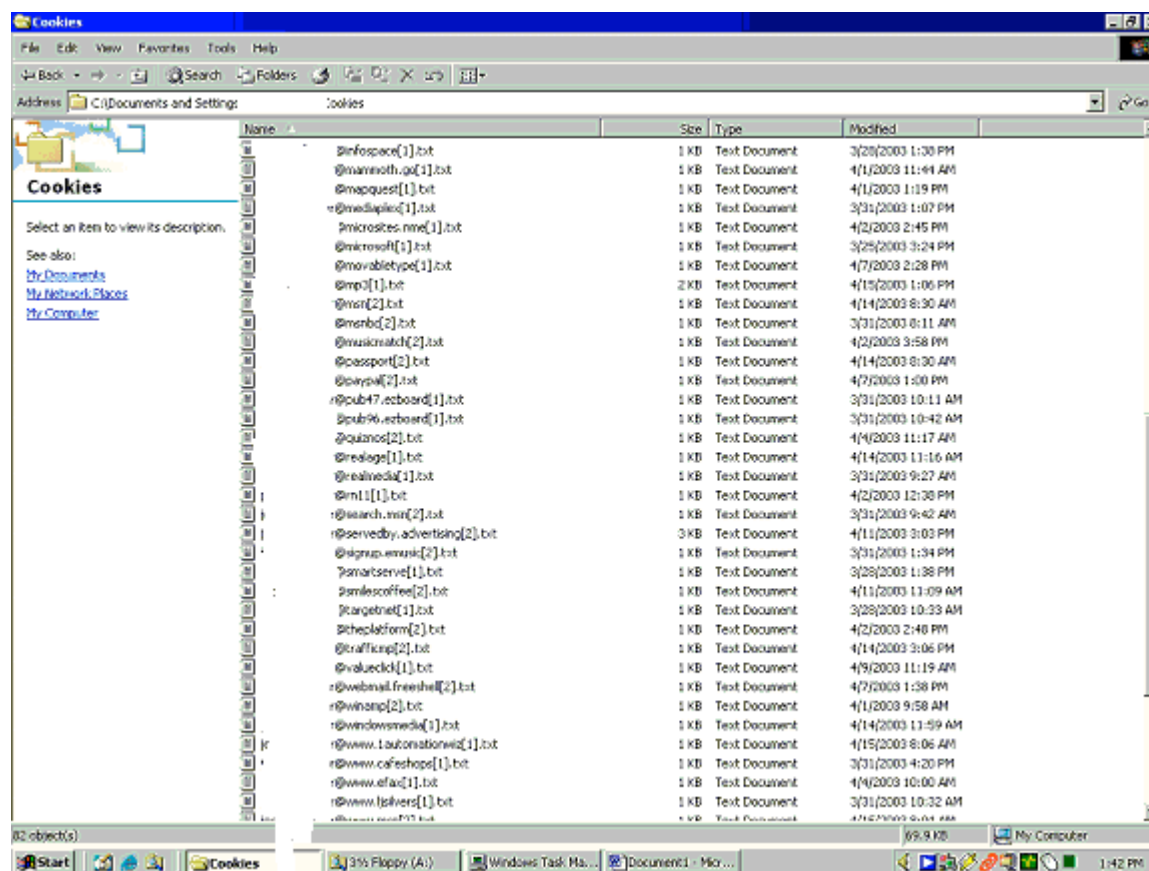
Cookies were dated from 3/27/03 to 4/17/03. This would indicate that either the suspect deleted his cookies on or before March 27th, or that he might have received this PC around that time (employees in his department tend to rotate through several stations on a regular basis). The timeline analysis will be able to give me some insight as to whether this could be the case (seen in a later section). He had a total of 82 cookies.

Listing of cookies, part 1:



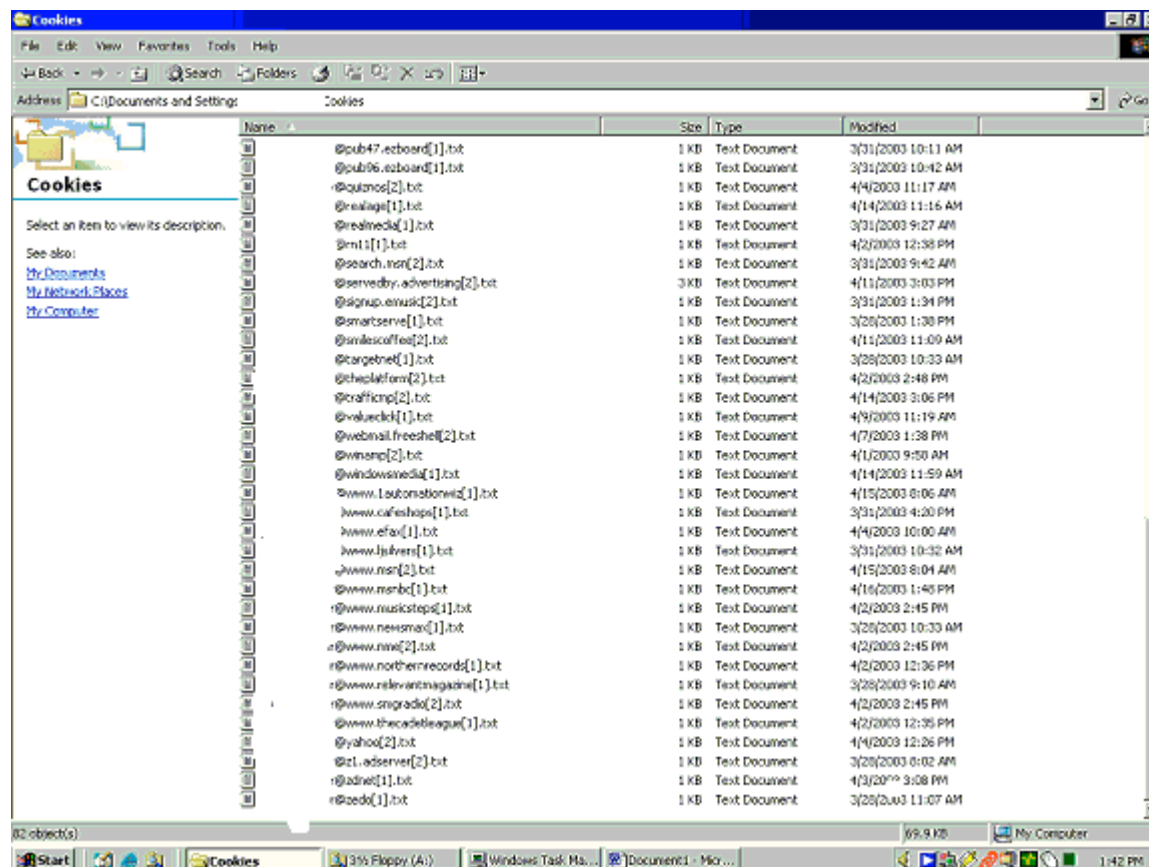
This screen shot was the first listing of cookies set on the suspect system. Most were self explanatory, ranging from ads, to CNN, to entertainment to google. Nothing out of the ordinary was presented in this listing.

List of cookies, part 2:



This screen shot was a continuation from the previous page as I scrolled down the window that listed the cookies. Here, I could see references ranging from Mapquest to Microsoft to "passport", which is used to access to his hotmail account, his mail account at freeshell, and some restaurant websites. No unusual cookies were noted in this listing.

Listing of cookies, part 3:



This was the third and final screen shot displaying the listing of cookies for the suspect. This final list shows references to cookies for music related sites, MSNBC, and Yahoo. No unusual listings were noted in this last listing of cookies.

Review of registry files:

Since the registry review was not done while the PC was in my possession, I had to once again do this the hard way. I pulled the individual files that make up the registry from the image itself using icat program. Similar to the method used to recover the Internet History files, I found the file's inode names from the same fls listing and noted their corresponding inode number. The files that were recovered include:

```
C:\winnt\system32\config\security
C:\winnt\system32\config\software
C:\winnt\system32\config\sam
C:\winnt\system32\config\system
C:\winnt\system32\config\default
```

These files make up the system's registry. Once these files were recovered, I used strings command (e.g. strings security | more. Strings is a Unix command that allows

binary information (or compiled program code) to be viewed as though it were a regular text file.) When I redirect the output of the strings search through the more command, this allows me to stop the ongoing, automatic scrolling of the screen and just view one “page” or screen of information at a time). This was not the preferred method; however, without the machine in hand, I did not have a choice.

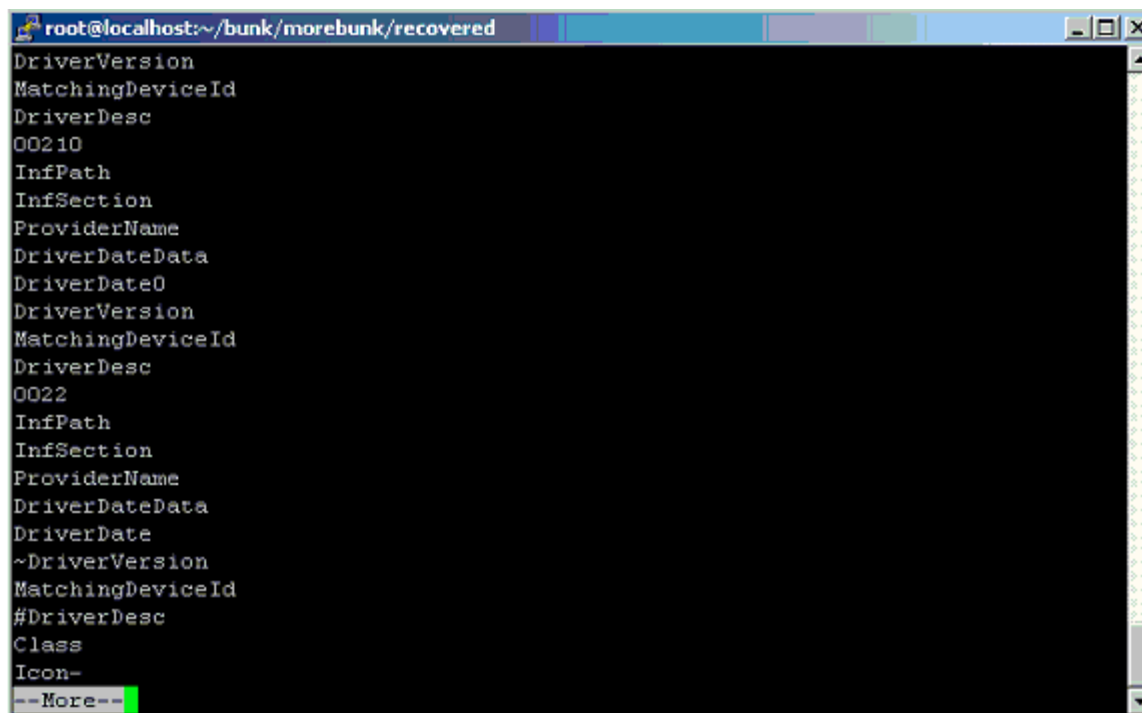
First, I reviewed the Security registry file, which was approximately 33K in size. The most interesting tidbit in this review was the reference to the EFS Encryption File Certificate and a File Encryption Key. I did not see anything on the disk that appeared to be encrypted, but the existence of the EFS Encryption prompted me to look for signs of encrypted information.

The second file reviewed was the Software key, coming in at a whopping 13MG. Right at the top of this file was another reference to “OpenWithEncryptionMenu”. I was not sure if this might have been in reference to putty, the ssh client that he used, or if he was really using EFS or if someone else using the system might be using encryption. (See the strings search on the word “encrypt” for further details.) Further review of the Software file showed several Secure Certificates that were used by the fax program that was installed for normal work purposes.

A file this size had a lot to review, but after paging through the entire file, nothing else was found to be out of the ordinary.

The third registry key reviewed was the SAM key, which was very small (24K). Nothing unusual stuck out in the review of this file. It appeared to have user Security Identifier (SID) information, which you would expect to find there. (Note: SID is the unique alphanumeric identifier assigned to a user when their account is created).

The fourth registry key was the System key, slightly larger at 2MG. There were many entries referring to drivers; however, this is not necessarily unexpected:

A terminal window titled 'root@localhost:~/bunk/morebunk/recovered' displays the output of a 'strings' command. The output lists various system-related strings, including 'DriverVersion', 'MatchingDeviceId', 'DriverDesc', '00210', 'InfPath', 'InfSection', 'ProviderName', 'DriverDateData', 'DriverDate0', 'DriverVersion', 'MatchingDeviceId', 'DriverDesc', '0022', 'InfPath', 'InfSection', 'ProviderName', 'DriverDateData', 'DriverDate', '~DriverVersion', 'MatchingDeviceId', '#DriverDesc', 'Class', 'Icon-', and '--More--'. The terminal has a black background with white text. The window's title bar shows standard Linux window controls (minimize, maximize, close) and the terminal's title. A vertical scrollbar is visible on the right side of the terminal window.

```
root@localhost:~/bunk/morebunk/recovered
DriverVersion
MatchingDeviceId
DriverDesc
00210
InfPath
InfSection
ProviderName
DriverDateData
DriverDate0
DriverVersion
MatchingDeviceId
DriverDesc
0022
InfPath
InfSection
ProviderName
DriverDateData
DriverDate
~DriverVersion
MatchingDeviceId
#DriverDesc
Class
Icon-
--More--
```

This particular screen shot was taken during the strings examination of the System file. While paging through the text displayed as output from the strings command, I saw many references to drivers, and this was an example how of the data was presented.

Also PNP (I assume plug and play), display, and layout were mentioned frequently, along with other devices on the system. Then I happened upon:

System screen shot, first in a series:

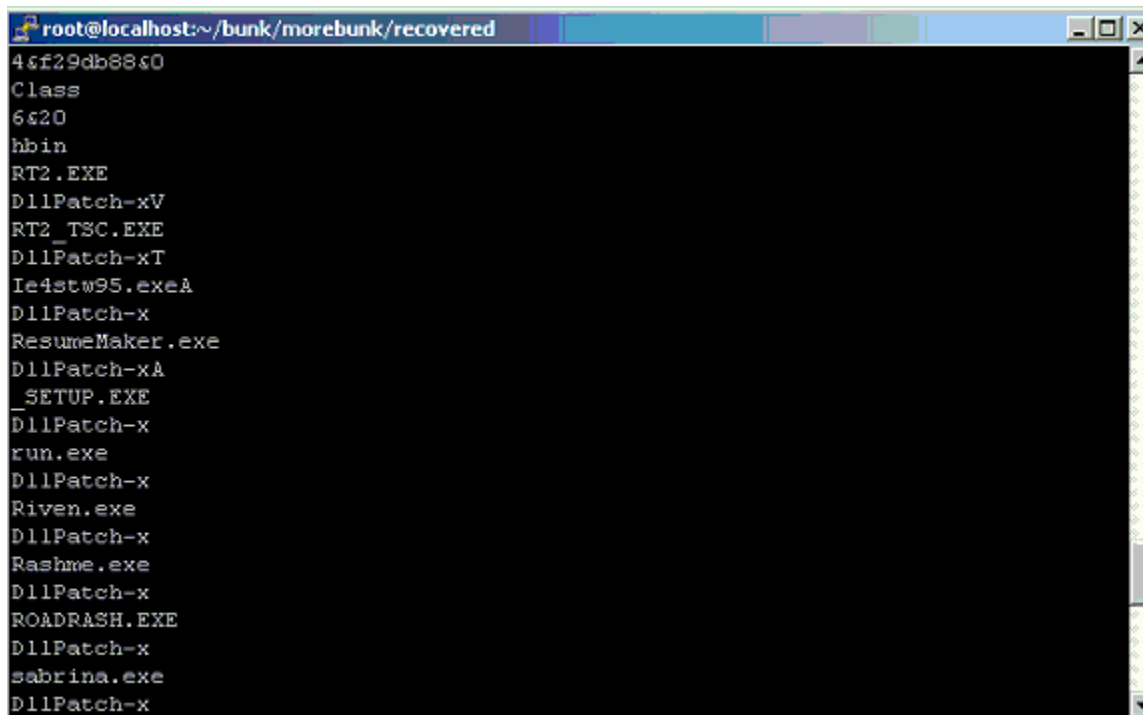
© SANS Institute 2003

```
root@localhost:~/bunk/morebunk/recovered
Winm
MIDI9
VID_
Schemes
Defaulte
aChannels
MediaResources0
DirectSound
(45F
Application Compatibility
Appl
ANDRETTI.EXE34492A63000BEA000
DSAPPHACKID_MODIFYCSBFILUREr
ANDR
DD2.EXE3288834400092BF0
DSAPPHACKID_DISABLEDEVICE
DD2.
DD2H.EXE328882C500092DF8
DSAPPHACKID_DISABLEDEVICE
FIFA
FIFARTWC.EXE345FB52D00183C00S
DSAPPHACKID_PADCURSORSn
FIFAWC.EXE35320039001AE400L
--More--
```

As seen in this screen shot of another part of the System file, the word “Hack” was referred to several times. This in itself was alarming, but also of note here were the many executable files that are listed, beginning with Andretti.exe. A google search on this executable (<http://www.google.com/search?hl=en&ie=UTF-8&oe=UTF-8&q=andretti.exe>) pointed to some sort of racing game based on Mario Andretti.

© SANS Institute 2003

System series screen shot 2:



```
root@localhost:~/bunk/morebunk/recovered
4sf29db88s0
Class
6s20
hbin
RT2.EXE
DllPatch-xV
RT2_T3C.EXE
DllPatch-xT
Ie4stw95.exeA
DllPatch-x
ResumeMaker.exe
DllPatch-xA
_SETUP.EXE
DllPatch-x
run.exe
DllPatch-x
Riven.exe
DllPatch-x
Rashme.exe
DllPatch-x
ROADRASH.EXE
DllPatch-x
sabrina.exe
DllPatch-x
```

This was another part of the System file. This area reveals the executable “ResumeMaker.exe”, “Rashme.exe”, “Roadrash.exe” and “Sabrina.exe”. I turned to Google again, and each yielded the following:

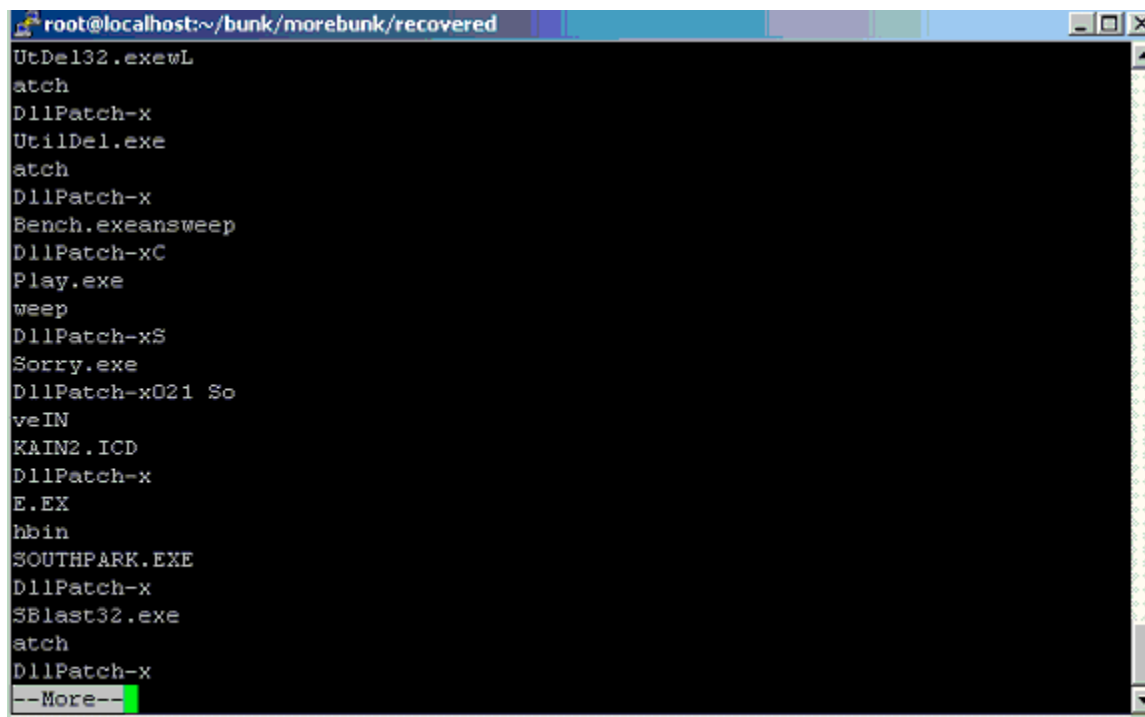
The search for ResumeMaker: <http://www.google.com/search?hl=en&lr=&ie=UTF-8&oe=UTF-8&q=ResumeMaker.exe> yielded nothing. However, <http://www.google.com/search?hl=en&lr=&ie=UTF-8&oe=UTF-8&q=ResumeMaker> yielded a few websites that referred to a software program for writing resumes.

The search for Rashme.exe: <http://www.google.com/search?hl=en&lr=&ie=UTF-8&oe=UTF-8&q=Rashme.exe> yielded a site that listed games by name and executable program name, and the name of the game was RoadRash. That site was: <http://www.kahncentral.net/files/games.txt>

The search for Roadrash.exe: <http://www.google.com/search?hl=en&lr=&ie=UTF-8&oe=UTF-8&q=Roadrash.exe> yielded quite a few references to a virus that used this executable. Specifically: <http://www.symantec.com/avcenter/venc/data/w32.nogrov@mm.html>. It was quite possible that this was a virus and was not detected by the anti-virus software because the version that we use often malfunctions, not keeping up to date on signatures. This particular virus was discovered in July 2003, after this disk image was created.

The search for Sabrina.exe: <http://www.google.com/search?hl=en&lr=&ie=UTF-8&oe=UTF-8&q=Sabrina.exe> yielded some references to mass email messages containing this file, which self installs itself. See: <http://easyweb.easynet.co.uk/~gcaselon/spam/babette.html>. This could have been another virus, but no references came up on virus websites.

System screen shot 3:



This was the last screen shot taken of the System file. Noted here were “Bench.exe”, “Play.exe”, “Sorry.exe”, and “SOUTHPARK.EXE” files. Searches on these files provided the following information:

Bench exe: <http://www.google.com/search?hl=en&lr=&ie=UTF-8&oe=UTF-8&q=bench.exe> turned up too many hits to be conclusive. It did not look malicious from the results that were yielded from the google search.

Play.exe: <http://www.google.com/search?q=play.exe&hl=en&lr=&ie=UTF-8&oe=UTF-8> yielded numerous references to the Real Jukebox player. See: <http://www.ifwlite.com/messages/Real%20Jukebox%20-20Uninstall%20problem%20with%20play.exe.txt>

Sorry.exe: <http://www.google.com/search?q=sorry.exe&hl=en&lr=&ie=UTF-8&oe=UTF-8> also yielded quite a few hits, but was another possible virus: http://www.pspl.com/virus_info/worms/orora.htm

Southpark.exe: <http://www.google.com/search?hl=en&lr=&ie=UTF-8&oe=UTF-8&q=southpark.exe> yielded numerous references to a virus:
<http://www.symantec.com/avcenter/venc/data/w32.southpark.worm.html>

It became evident from the review of the System file that a lot of unauthorized software had been installed, and that this system may have numerous viruses that did not appear to be fully cleaned off of the system. A review of the timeline file revealed that none of these files remained on the partition, so perhaps they were in fact removed, but not cleared from the registry.

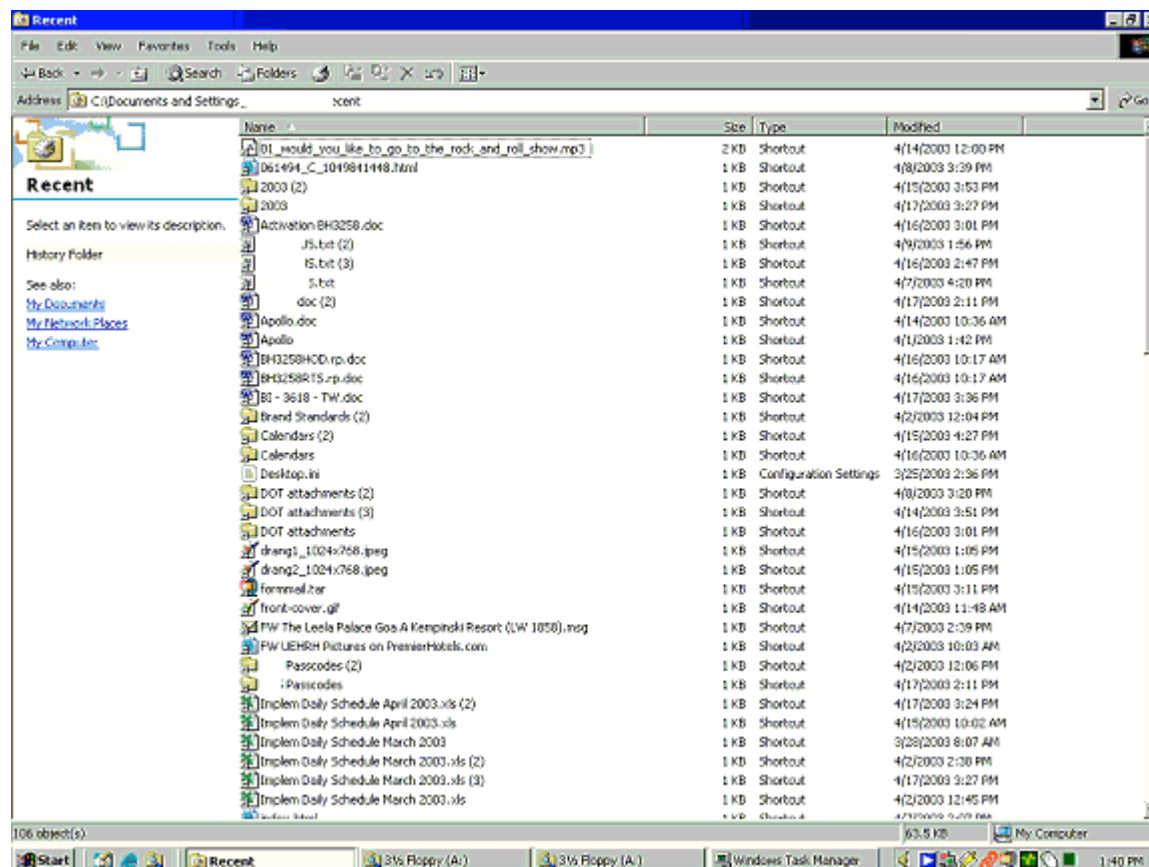
The Default registry file was reviewed last, sizing up to be 147K. Nothing unusual was noted and it seemed to contain fairly standard Windows entries (such as file types and other standard user specific entries like games (hearts) and Internet History).

Review of most recent files opened:

A review of the suspect's most recently viewed files was conducted, hoping this would provide further insight into his activity, especially since there was a concern that information that he was using was leaving the company. No anomalies were discovered. All that he opened appeared to be in line with the rest of his activity, including Internet surfing to work on his personal website, and several work files being opened. Some of the programs identified on the hard drive were also noted as having been executed recently.

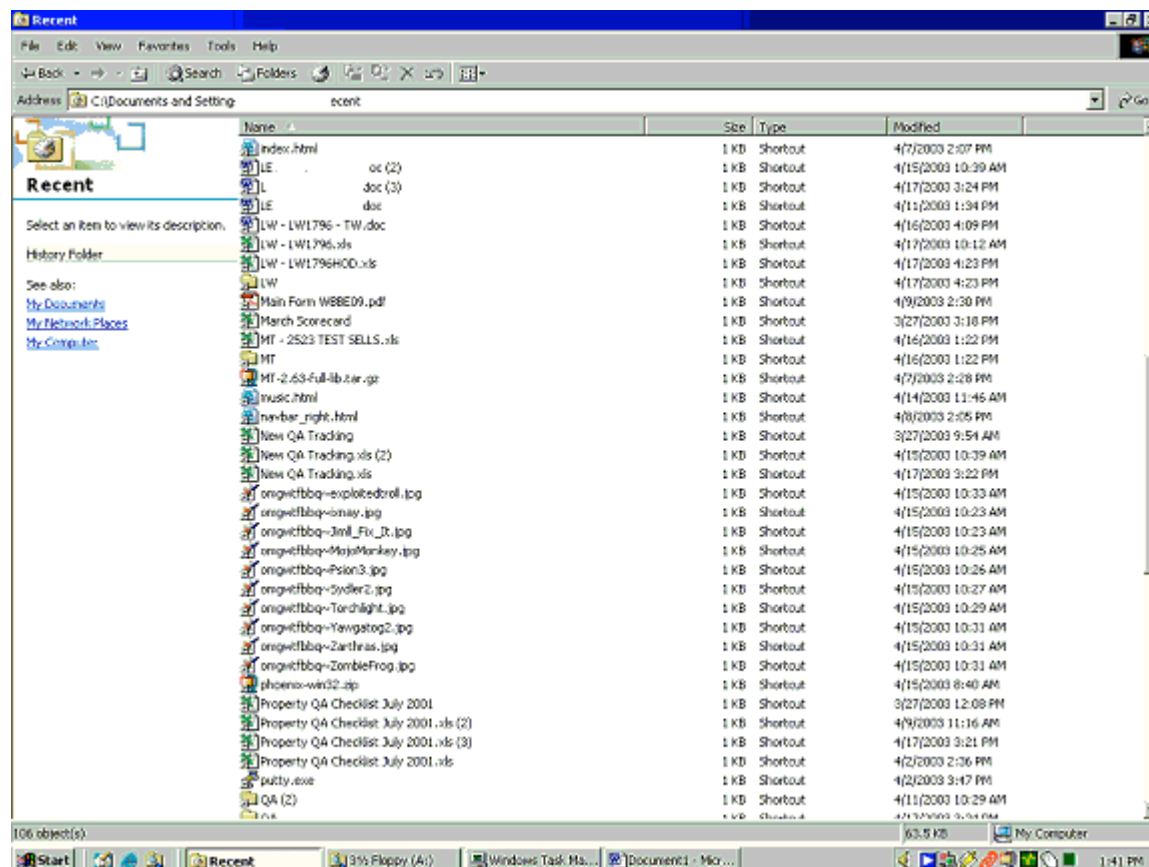
© SANS Institute 2003, All rights reserved. Author retains full rights.

Most recent file part 1:



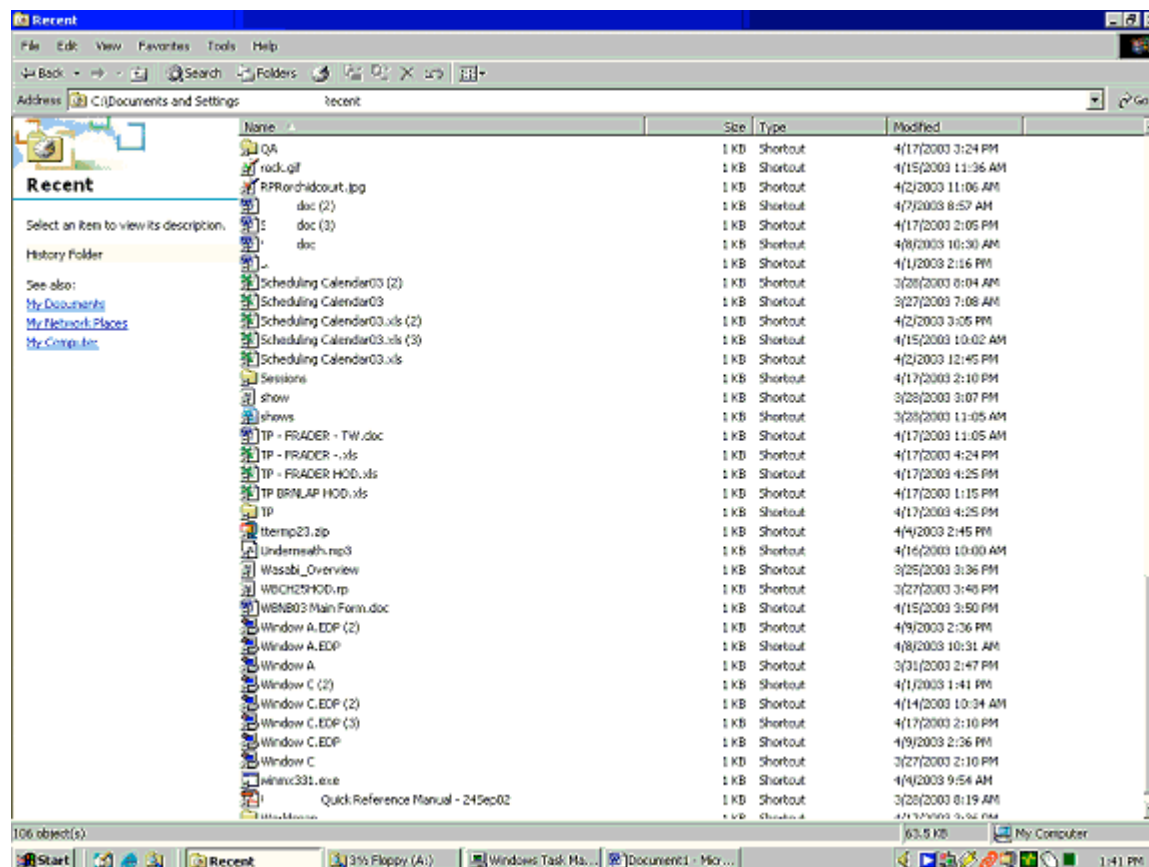
As seen in this screen shot of the Most Recently Used files, which was taken from another directory under the suspect's Documents and Settings directory, one of the first files listed was an mp3. Also noted about $\frac{3}{4}$ of the way down the screen was the formmail.tar file. I discovered in my analysis that the suspect was maintaining his own website. Formmail is a free tool to allow sending email from a webpage that was apparently being set up on the suspect's website. The remainder of these files appears to be work related.

Most recent files part 2:



In this continuation of the suspect's most recent files, I saw more work related files. Then about half way through, I saw several jpeg files that seem related to games; however, when I searched for them in Google, there were too many hits to be conclusive. Towards the bottom of the screen, the phoenix-win32.zip file was listed. This was the executable for the latest version (at the time) of the Mozilla browser. Lastly, the putty.exe file accessed on 4/2/03 is a freeware ssh client.

Most recent files part 3:

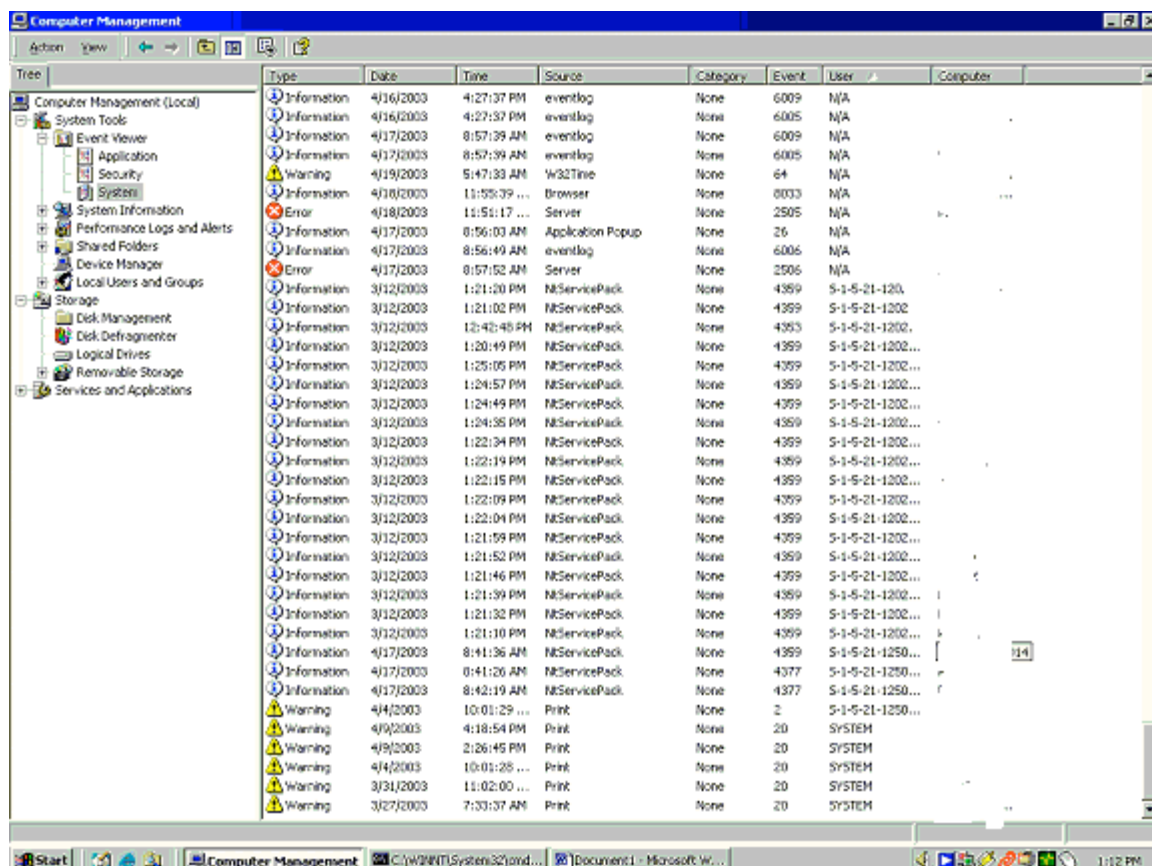


This last screen shot of the most recently used files shows more work related files being accessed including: a tterm23.zip file is TeraTerm pro, a telnet client that was used by my company, and an mp3 file and the Mozilla executable (winmx331.exe). I was also assuming that the reference to the “shows” page might be related to entertainment.

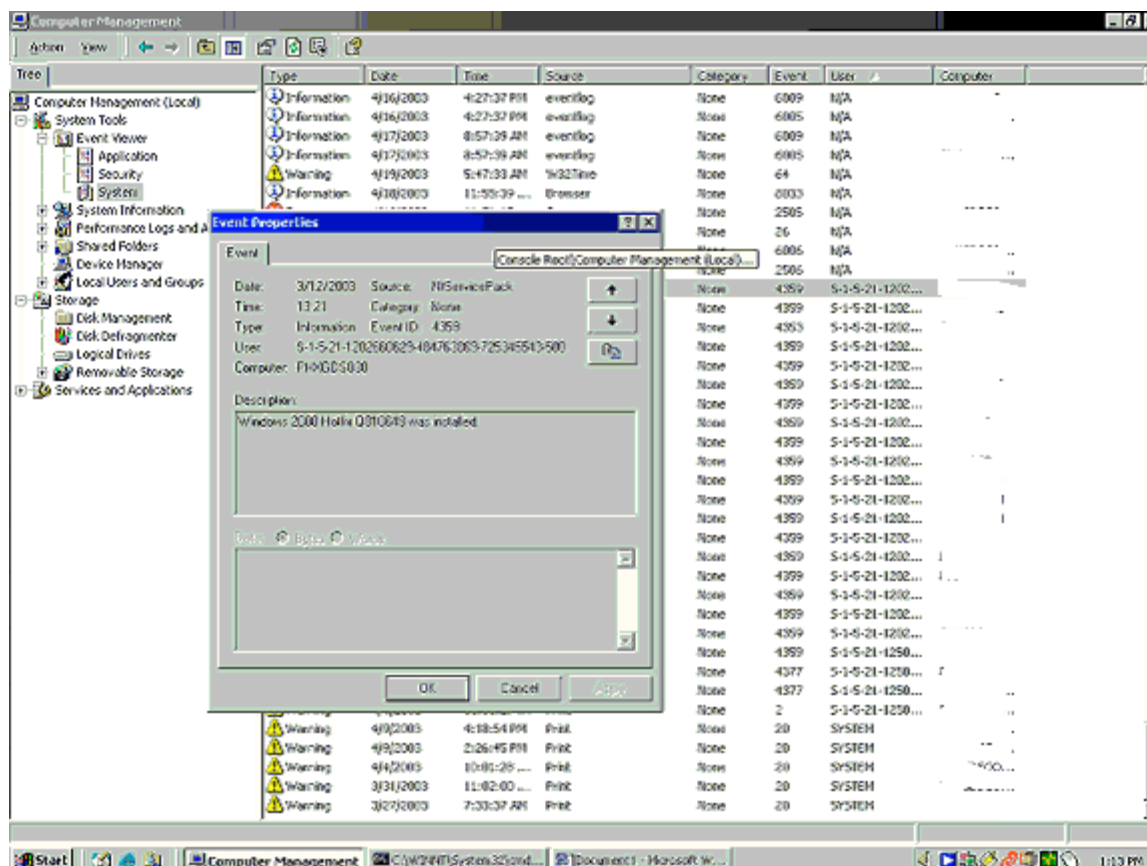
No malicious activity was detected from this review.

Notable event log entry:

© SANS Institute



This initial screen shot of the system log, taken from the suspect's system after the disk image was taken, shows the recent software/operating system patching activity on the system. Combined with the media analysis, this supports the conclusion that no significant activity occurred on the disk other than some patching.



To make this entry easier to read, the information in the screen above is re-displayed here:

Event Type: Information
 Event Source: NtServicePack
 Event Category: None
 Event ID: 4359
 Date: 4/17/2003
 Time: 8:41:36 AM
 Computer: PHXxxxxxx
 Description:
 Windows 2000 Hotfix Q331953 was installed.

This was a close-up look at the service pack entry in the system log. This entry confirmed that the system had recently had a service pack applied.

Review of the memory image:

In addition to the disk media image, an image of the memory was obtained from the suspect's PC in case he had any malicious program(s) running on the box. The approximately 132MG file held various remnants of access to dlls, SMS initialization, and normal program interaction. To examine it, I used the strings command and sent

the strings output to the “more” command like so: “strings | more”. This was a tedious process; however, it was the best way to review the file. It also showed footprints of the fax software that was used internally, as noted by the type of secure certificate that it uses.

I did find the suspect’s POP mail address (yet another mail account at Earthlink) lodged in here, as well as his password. There were other references to files that I was not familiar with so I made a short list of ones that caught my eye and researched them. They included:

..\..\halmpls\i386\mpprocst.c – not found anywhere

dialer.exe- a Microsoft program that allows phone dialing over the Internet.

Dsssig.exe- supposedly the Signature for DSS Provider (dssbase.dll). Also mentioned in some other threads as either a virus or a suspicious file.

C:\winnt\system32\Sndrec32.exe- the Windows Sound Recorder.

Unregmp2.exe- a file that has to do with Windows Media Player registration.

D:\nt\private\windows\appcompat_wu\shims\layer\win2lpropogatelayer.cpp- no matches on this. Possibly a low level windows file.

Some program called MUSICMATCH Jukebox

Sources:

http://www.itec.suny.edu/scsys/vms/system%20tools/windows/RCM/docs/winnt/nt_compare.htm

<http://www.liutilities.com/products/wintaskspro/processlibrary/sndrec32/>

<http://support.microsoft.com/default.aspx?kbid=329771>

Timeline Analysis:

Complete timeline file included in the submitted zip file.

I created the timeline file using the same fls output that I used to obtain the inode information for the Internet History files and the Registry files. In order to generate a true timeline file from the fls output, it needed to be put in date order, from oldest to newest, using the mactime program (another TASK utility). Mactime sorts all of the files by date (noting the modified, created and/or accessed time). The command issued was:

```
mactime -b fls_all_files > timeline_from_fls
```

Translation of this command:

-b fls_all_files refers to the fls output file that mactime needs to examine
> timeline_from_fls refers to the file I wanted to redirect the output of the mactime command to.

At the very beginning of the timeline file was the `WINNT/system32/spool/drivers/w32x86/Efxsndkm.dll` file created on June 2, 1980. Many of the files that followed were similarly created by various Windows programs and a previous version of the anti-virus program that we had been using (Innoculan). Files were dated from February 29, 1988 until April 18, 2003. Some of the older programs here include files from Netscape and Microsoft Office.

I tend to like to look at timeline files from the newest files for some reason instead of weeding through the entire file initially. So, jumping right to the end, I see some SMS files (newly created, indicated by the mac fingerprint), some dlls (accessed), and some other system files accessed. This activity could have resulted when the machine was shut down and when it was started back up again.

There were some executables that were listed in the timeline that I was not familiar with, so I researched them. They include:

xpicleanup.exe- Apparently a part of the Mozilla package.
pptico.exe- Apparently something related to Powerpoint.

Sources:

<http://fresh.t-systems-sfr.com/pc/src/www/.warix/mozilla-win32-1.3.1-talkback.zip.html>
http://www.und.ac.za/und/arch/arch/www/gallery/Web_Govind/Application%20Data/Microsoft/Installer/%7B00020409-78E1-11D2-B60F-006097C998E7%7D/

Of course, when I scroll to the previous day, the last thing our suspect seemed to access was his winamp program, playing some mp3s. Also, a patch of some sort to Office was installed on the 17th, which was in line with what was captured from the event log, as there was an entry made to NTUninstall. In fact, it looks like either a lot of patches were installed on this date, or some new software was installed, as there were several accesses to files in system32, and the winnt/inf directory that day. It appears to have been Windows Service Pack 4 since there were several references to it:

`/Documents and Settings/suspect/Local Settings/Temporary Internet
Files/Content.IE5/BJDNBPSO/Q811493_W2K_SP4_X86_EN_f191e3802536d4870dfe141e754af04[1].exe`

Early in the day, there were references to Windows Update, so this was probably what happened:

`7937 .ac -/rwxrwxrwx 0 0 7190-128-3 /Program Files/WindowsUpdate/V4/iuident.cab`

On April 16, I noticed the continuing trend of lots of file accesses to Temporary Internet files, as well as files accessed within the Mozilla browser that were installed. There was some use of ftpx on this day, but I traced it down to what appeared to be internal reports

that get transferred to and from a particular internal server. I couldn't find any evidence that these files were leaving the company.

This file was found to be accessed on April 14:

/Documents and Settings/suspect/Recent/MT-2.63-full-lib.tar.gz.lnk

This appears to be a Movable Type library, probably for the suspect's own website.

(Source: http://tecfa.unige.ch/perso/fabien/projects/blogs/install_mt.html)

Files were ftp'd on 3/27, 3/31, 4/2, 4/3, 4/7, 4/8, 4/11. Further research was done to determine what types of files were being ftp'd, to confirm whether this was in fact proprietary data. It was an internal report that specified room rate information, but no proprietary information. Although I can not verify that this information was not ftp'd to a server that the suspect used for personal gain, the type of file that was being ftp'd fit into the department's flow of work, as if he was delivering a manual report to an internal ftp server.

The remainder of the activity looked largely the same from a timeline perspective. To assist in my analysis, I searched for .exe files to see if any further unfamiliar executables could be found. Executables that were accessed were:

setup_wm.exe- Windows Media Setup

/WINNT/system32/logagent.exe- appears to be associated with a video driver.

putty.exe- this was a free ssh client.

hinv32.exe- an SMS-related file.

/Program Files/Windows NT/Accessories/ImageVue/kodakprv.exe – non-approved Kodak imaging software.

textpad- I believe this was an internal, proprietary application that works with our other internal applications.

/Program Files/Windows Journal Viewer/jntview.exe- not sure, but appears to be a valid Microsoft program.

/WINNT/system32/wisptis.exe- an application that helps to build Tablet PC applications.

/WINNT/system32/netsh.exe- a tool an administrator can use to configure and monitor Windows 2000-based computers at a command prompt. Very interesting!

/WINNT/system32/mshta.exe- the hypertext application interpreter (also Microsoft)

/Program Files/FTP Explorer/setbrows.exe is a part of the ftpx client. The setbrowse program just tells the help file where to find the user's browser.

/WINNT/Installer/{ABEB838C-A1A7-4C5D-B7E1-8B4314B00543}/Msbllco.Exe- not sure.

/Program Files/MUSICMATCH/MUSICMATCH Jukebox/mmjb.exe- The Music Match Juke box.

/Program Files/MUSICMATCH/MUSICMATCH Jukebox/MMFWLaunch.exe- Also part of Music Match.

/Program Files/MUSICMATCH/MUSICMATCH Jukebox/MMJBBurn.exe- Yet another part of Music Match.

/Program Files/Winamp3/mc.exe- The MAKI Compiler MAKI stands for Make A Killer Interface. It is the scripting language for Winamp3.

/Program Files/Windows NT/dw15.exe- appears to be an error reporting program for SQL 2000. I thought this strange, but I have it on my system as well.
/Program Files/WinMX/WinMX.exe- Some sort of sound card driver.
/WINNT/Installer/{00010409-78E1-11D2-B60F-006097C998E7}/misc.exe (changed)- this was all over the map, anything from a valid file to something related to porn pop-ups. Not sure on this one, but feel that it was valid since no signs of malicious behavior had been found elsewhere.
/WINNT/hh.exe - looks to be part of an earlier service pack.

Sources:

<http://www.video-drivers.com/drivers/15/15358.htm>
<http://www.microsoft.com/mspress/books/sampchap/5958d.asp>
<http://support.microsoft.com/default.aspx?scid=http://support.microsoft.com:80/support/kb/articles/Q242/4/68.ASP&NoWebContent=1>
<http://forums.techguy.org/t150909/s5f2289bd752dc0a24b993271993a2309.html>
http://www.winamp.com/nsdn/winamp3x/skinning/tutorials/4.1-drawer_script.shtml
<http://www.microsoft.com/technet/treeview/default.asp?url=/technet/prodtechnol/sql/deploy/confeat/CERSTEP.asp>
<http://www.soundcard-drivers.com/drivers/85/85018.htm>

Not a lot of real system activity on this system occurs until the mid-March timeframe. So, it appears that this user had not been using this PC for a great deal of time.

Recover Deleted Files:

Two methods were used to view all deleted files. A first pass was done using Autopsy (mostly because I didn't realize the disk anomaly until after I started some analysis). The second pass was to use the fls tool to look at what files were deleted. Both tools have a different way of displaying deleted files, which I will explain a bit before proceeding.

As mentioned in the Media Analysis section, when data is deleted on a disk, the entry in the Master File Table is changed, but unless that data in the specific cluster on the disk is overwritten with new data, it is not fully deleted. The Autopsy browser actually uses the fls tool behind the scenes to display deleted files. However, because of the issue with the way data was presented in Autopsy and I was not able to see it all, I had to use fls. Fls, the file listing program that comes with TASK, was used with the following parameters:

```
./fls -rpd -f ntfs /mnt/usbhd/suspectdisk.img > output/fls_deleted
```

This will provide a list of all files that were marked for deletion in the Master File Table. The translation of this command is as follows:

-rpd refers to displaying the information recursively, or to display all directory and sub-directory information, p indicates that the full path for each file should be displayed, and d stands for deleted files

-f ntfs was the file system type that fls will be reading in this case

/mnt/usbhd/suspectdisk.img was the name of the file system image that fls will be reading

> output/fls_deleted was the filename where I would like the output redirected

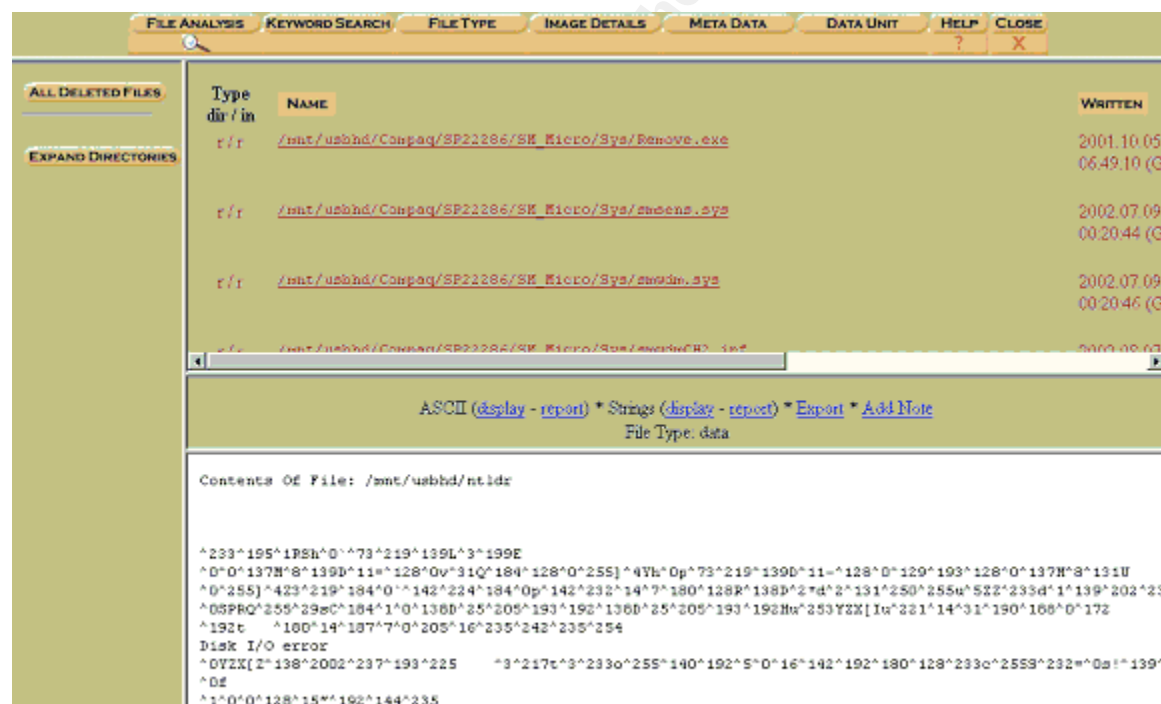
This second command was applied to gather the undeleted file information:

```
./fls -ru -f ntfs /mnt/usbhd/suspectdisk.img > output/fls_undeleted
```

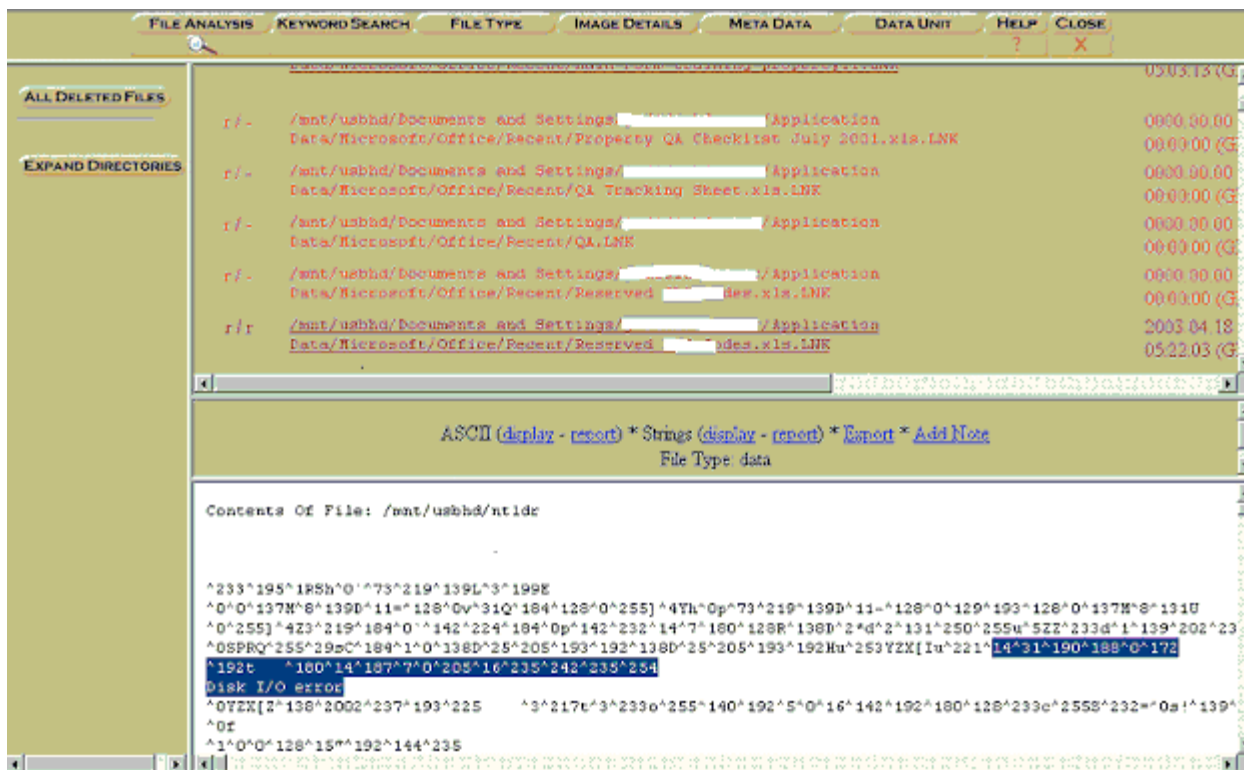
Where all of the above parameters are the same, but in this case:

-ru stands for recursive listing of undeleted files.

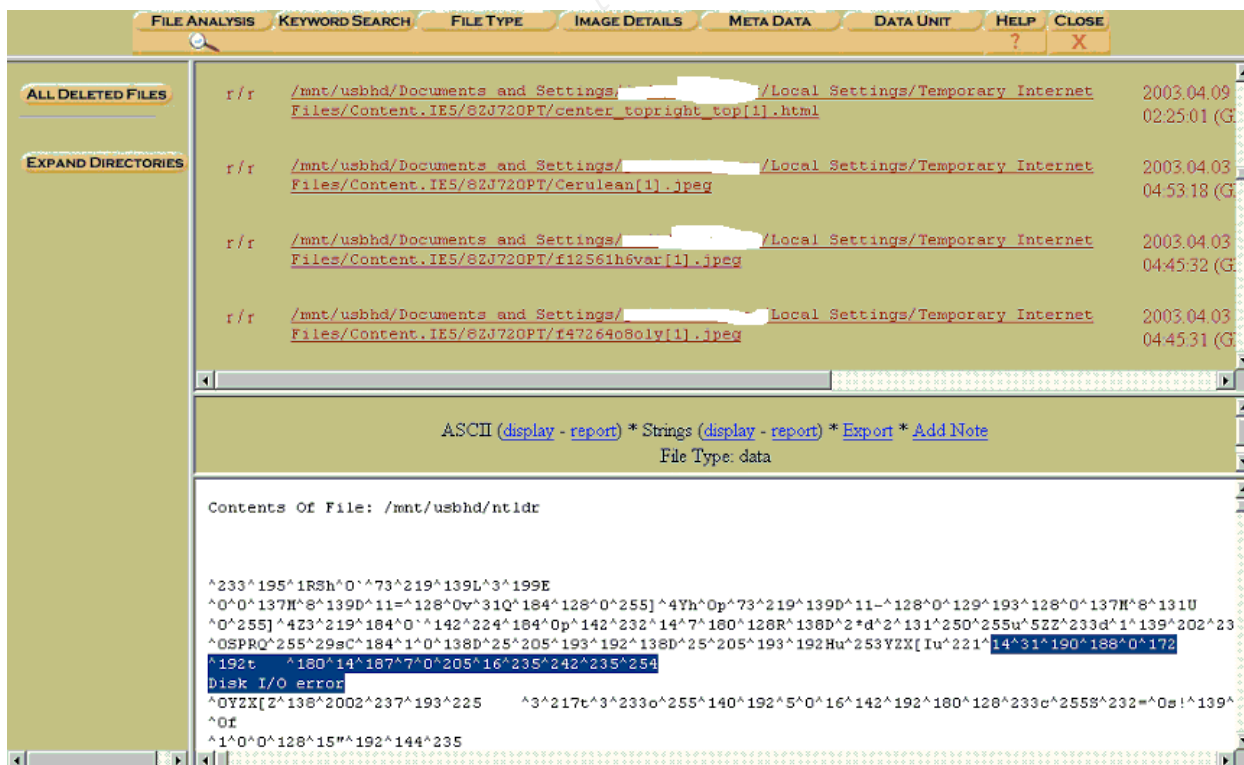
Using Autopsy, I displayed all deleted files that could be displayed, which included the suspect's Documents and Settings directory and the root of the disk. From the root, there were many files deleted by SMS and what appears to be a driver update for Compaq:



Many of the deleted files from the suspect's directory were normal work files: spreadsheets, documents, etc:



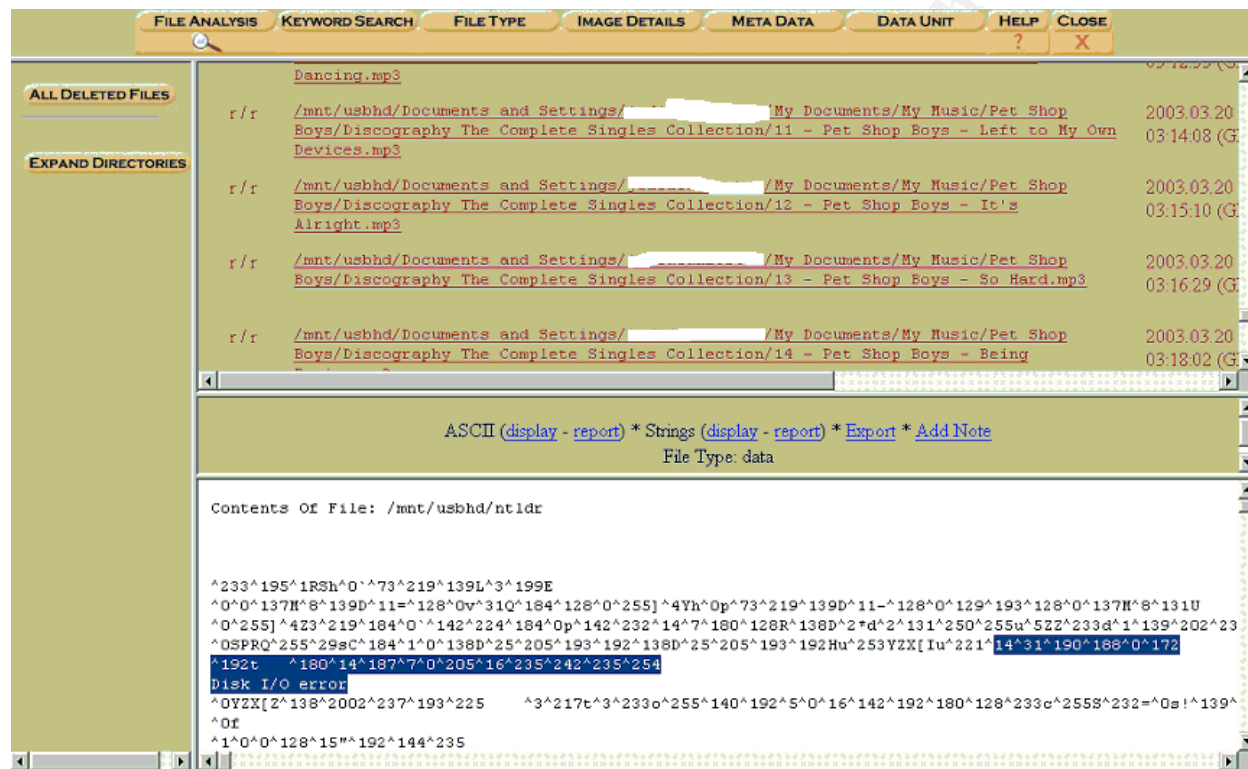
There were also plenty of deleted files from the suspect's Mozilla cache directory:



The suspect also appears to have cleared his cookies around the first of April as there were numerous references to deleted cookies.

The first reference I found to an mp3 was on April 16. The file had been deleted. Around 3/26/03, a version of Dshock was deleted. Then I came across quite a few files deleted in the suspect's IE Content directory, no doubt temporary files from his constant surfing of the Internet (there were even some work related files in there!).

Then I stumbled across the suspect's deletion of a great number of mp3 files.



There were no other strange files that were noted being deleted from the root...mainly systems files and the old pagefile.sys files.

The second stage of my analysis was to run an fls against the image. This produced quite a bit more information on the deleted files, but still nothing of concern.

On the deleted search, I was able to view deleted files from the anti-virus software directory, Program Files (which appeared to be all normal software upgrade activity), a program in Program Files called MUSICMATCH that had been deleted, as well as winamp, and information on patch installations in the WINNT directory. It appeared that the suspect might have upgraded his Mozilla client as there was quite a bit of activity in the program directories for this application. The same behavior was noted with the phoenix application (another version of the Mozilla browser). The suspect's Temporary Internet files directory also had quite a few deletions (perhaps it was cleaned out).

For the undeleted files: There was a long list in this category. A cursory review looked like the majority were pictures (my guesstimate was from websites), url/html files, cookies, and system files (the results of running updates) like dlls and sys files. There were 701 instances of .png files, 263 bitmaps, 1271 jpegs and 3640 gifs. None appeared to be inappropriate (although one did have the word bikini in it). There was no reason to suspect that any type of steganography was being used because no software was found that fit that would enable the suspect to do this. Steganography is “the art of writing in cipher, or in characters, which are not intelligible except to persons who have the key; cryptography”,

<http://dictionary.reference.com/search?q=steganography> . Oftentimes, Steganography is used to hide messages within pictures. Otherwise, no unusual activity was noticed.

String Search:

Several string searches of the entire disk image were conducted. To accomplish this, the Unix strings command was used. Strings is a tool that allows you to read printable characters in a binary or compiled program file. The exact command employed for each search was:

```
strings /mnt/usbhd/suspectdisk.img | grep "text string" > output_file
```

Translation of this command:

strings /mnt/usbhd/suspectdisk.img was how I wanted to view this particular file, by running it as a parameter of the strings command

| is the pipe symbol, or a way to redirect command output to be used in another command

grep is another Unix program that was used to do pattern searching and matching in files

“text string” was replaced by whichever key word that I desired to look for at the time
> output_file was where I wanted the results from the grep command to be sent

Each search took a great deal of time because the disk image was so large (12+ hours per search). I decided that because it took so long to conduct these searches, they would be limited to a short, prioritized list. Several notes were made while doing a preliminary review of the workstation, which helped to narrow the searches to a few key words. The suspect seemed to visit the hostname Exodus numerous times during our surveillance. In addition, a thorough analysis of the suspect’s files revealed that the word hacker showed up several times. Lastly, to ensure that no illegal use of mp3 files was underway, mp3 was searched for, as well as ftp, ssh, takasi (one of the ftp sites that he had visited), Yahoo, and his site: bittersweetirony.com.

Additionally, the media was searched for IP addresses in case there was activity with locations other than the Exodus server.

The searches resulted in the following:

1. **Exodus** gave 1081 hits. These hits were reviewed closely, and it was determined that work was being done on a couple of the websites that the suspect maintains. One dealt with promoting a band that the suspect belongs to. The other was a web diary of sorts where the suspect droned on about his life and how much he “loved” his job.
2. **Hacker** came up with 525 hits, but after a thorough review, it was determined that these were all in articles that the suspect read while he was supposed to be working.
3. **mp3** resulted in 10943 instances. There were many files and references to files stored on this hard drive.
4. The **ip search** resulted in largely nothing. Oddly enough, the entire list of 2679 hits came from a cluster marked \$BadClus and nothing significant was gained as they were references to Windows Update process or pieces of the processes related to surfing the web.
5. References to the **takasi** ftp site totaled 62. This was one of the possible sites that the suspect could have transferred files to and from.
6. A string search for **yahoo** resulted in 5566 hits. Much of this seems to be references to the suspect’s mail account at Yahoo.
7. A string search for **ftp** resulted in 21110 hits! Much of this turned out to be log files, setup files, registry settings, but not necessarily the information that I was looking for.
8. References to the suspect’s website, **bittersweetirony**, resulted in 2002 hits.
9. A final strings search for **ssh** resulted in 50228 hits. This particular search warranted a lot of scrutiny since I knew that the suspect was using ssh to communicate to other locations. Much of what appeared here was insignificant as the search picked up references to ssh in normal system activity, such as the SMS process “GetSMSShellProcessHandle”.

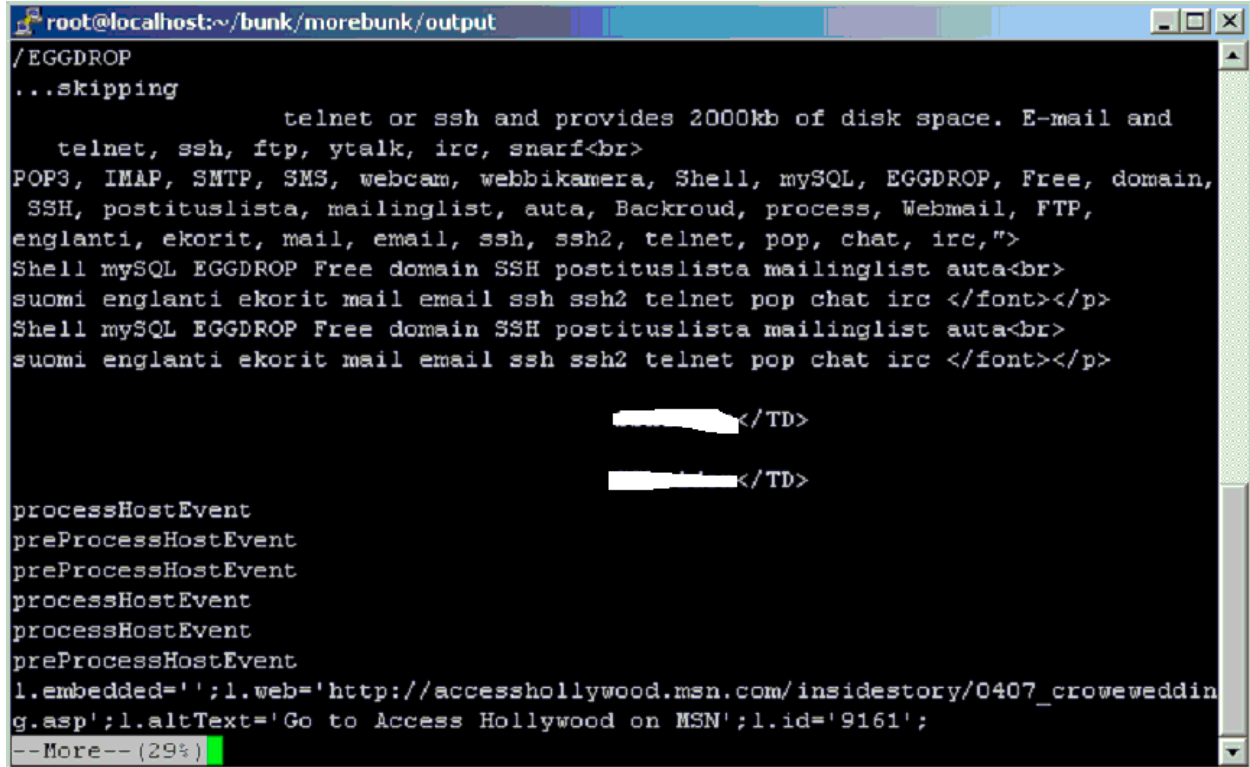
```
root@localhost:~/bunk/morebunk/output
<a class=fl href=/search?hl=en&lr=&ie=UTF-8&oe=UTF-8&q=related:www.der-keiler.de/Newsgr
e/Newsgr oups/comp.security.ssh/2003-02/0064.html>Similar pages</a></font><!--n--
> <blockquote class=g><p class=g><!--m--><a href=http://www.der-keiler.de/Newsgr
oups/comp.security.ssh/2003-02/0060.html>comp.security.ssh: <b>Freeshell</b>. <b>
org</b></a><br><font size=-1> <b>...</b> of my name: &quot;Re: <b>Freeshell</b>.
<b>org</b>&quot;; <b>...</b> From: werted &lt;abuse@werted.tk> Date: Tue, 4 F
eb<br>
<br><font color=#008000>www.der-keiler.de/Newsgr oups/comp.security.ssh/ 2003-02/
0060.html - 6k - </font><a class=fl href=http://[redacted]/search?q=cache:U
Aaso22kztoC:www.der-keiler.de/Newsgr oups/comp.security.ssh/2003-02/0060.html+fre
eshell.org+offline&hl=en&ie=UTF-8>Cached</a> - <a class=fl href=/search?hl=en&lr
=&ie=UTF-8&oe=UTF-8&q=related:www.der-keiler.de/Newsgr oups/comp.security.ssh/200
3-02/0060.html>Similar pages</a><br>[ <a class=fl href=/search?hl=en&lr=&ie=UTF-
8&oe=UTF-8&q+=site:www.der-keiler.de+freeshell.org+offline>More results from www
.der-keiler.de</a> ]</font><!--n--> </blockquote><p class=g><!--m--><a href=http
://lists.freshrpms.net/pipermail/rpm-list/2002-April/000813.html><b>Offline</b>
IMAP experiences?</a><br><font size=-1> <b>...</b> Next message: <b>Offline</b>
IMAP experiences <b>...</b> anymore).=20 --=20 Michel Alexandre Salim GPG<br>
[redacted]/TD>
[redacted]/TD>
ACCOUNT MGR : GetSMSShellProcessHandle returned a NULL handle. GetLastError retu
--More-- (52%)
```

This screen shot shows part of the results from the search on the ssh text string. Towards the bottom of the screen, the typical reference to ssh is seen: GetSMSShellProcessHandle.

I was a little concerned when I came across a reference to "EGGDROP" which is an IRC Bot (Internet Chat Relay program). However, the other references immediately after this entry indicated that this was merely a website that supported eggdrop as a means of communication.

© SANS Institute 2003

Eggdrop reference 1:

A screenshot of a terminal window with a blue title bar. The title bar text is 'root@localhost:~/bunk/morebunk/output'. The terminal content shows the output of the Eggdrop bot. It starts with '/EGGDROP' and '...skipping'. Then it lists various services: 'telnet or ssh and provides 2000kb of disk space. E-mail and telnet, ssh, ftp, ytalk, irc, snarf
POP3, IMAP, SMTP, SMS, webcam, webbkamera, Shell, mySQL, EGGDROP, Free, domain, SSH, postituslista, mailinglist, auta, Backroud, process, Webmail, FTP, englantti, ekorit, mail, email, ssh, ssh2, telnet, pop, chat, irc,>'. This is followed by two identical lines: 'Shell mySQL EGGDROP Free domain SSH postituslista mailinglist auta
suomi englantti ekorit mail email ssh ssh2 telnet pop chat irc </p>Shell mySQL EGGDROP Free domain SSH postituslista mailinglist auta
suomi englantti ekorit mail email ssh ssh2 telnet pop chat irc </p>'. There are two lines of redacted text, each followed by '</TD>'. Then it lists several 'processHostEvent' and 'preProcessHostEvent' messages. Finally, it shows an embedded HTML link: 'l.embedded='';l.web='http://accesshollywood.msn.com/insidestory/0407_crowewedding.asp';l.altText='Go to Access Hollywood on MSN';l.id='9161';'. The bottom of the window shows '--More-- (29%)' with a green cursor.

The reference can be seen in this screen shot near the top right corner.

© SANS Institute 2003

Eggdrop reference 2:

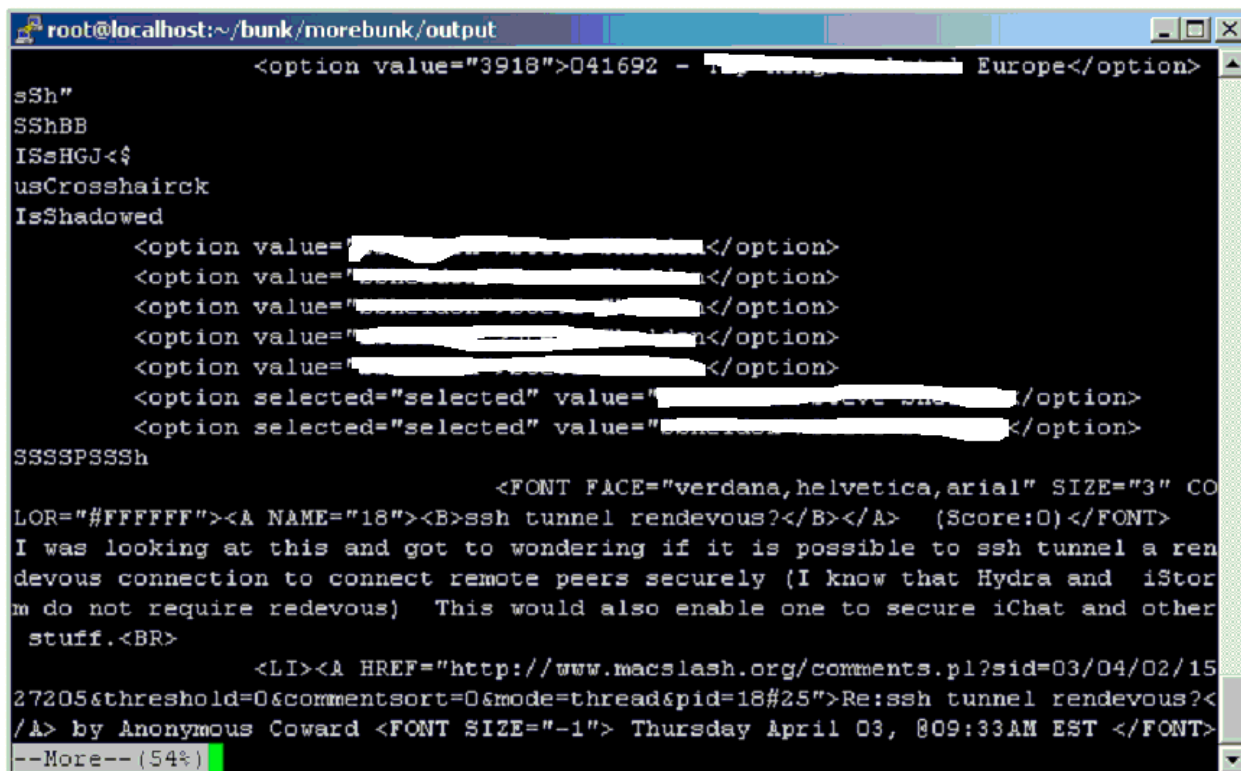
```
root@localhost:~/bunk/morebunk/output
processes allowed:</b> Reasonable background processes (personal crontab allowed
). No bots/proxy/bnc.<br><b>Other info:</b> Requires a one-time donation of USD
$5 through Paypal.<br></td><td width="190" align="right"><script src="http://rcm
.amazon.com/e/cm?t=bylurnetfreess-20&l=st1&search=unix&mode=books&p=9&o=1&bg1=CCC
CCC&lcl=00659C&lt1=_blank" type="text/JavaScript"></script><noscript><MAP NAME="
boxmap"><AREA SHAPE="RECT" COORDS="1, 103, 92, 112" HREF="http://rcm.amazon.com/
e/cm/privacy-policy.html?o=1" target=main><AREA COORDS="0,0,10000,10000" HREF="h
http://www.amazon.com/exec/obidos/redirect-home/bylurnetfreess-20" target=main></M
AP></noscript>
</td></tr></table><br></p><table border="0" width="95%" cellpadding="5" cellspacing="5">
<tr class="heading"><td><a name="66"><a href="http://www.cgladue.com/"><
img src="../images/gohome.png" alt="" border="0">Cgladue.com</a>&nbsp;<a href="m
ailto:cgladue@optn.org">
</a></td><td class="upd"><b>Last updated:</b> 2003-02-19</td></tr><tr class="te
xt"><td><b>Hardware:</b> Pentium PRO 200MHz DUAL CPU with 128 MB of RAM and 20GB
SCSI-2 hard drives.
<br><b>Operating system:</b> Linux SMP kernel 2.4.18<br><b>Quota:</b> Unlimited.
<br><b>Internet connection:</b> 1.5Mbps ADSL.<br><b>Services/programs:</b> Email
(SMTP/POP3), IRC (BitchX), 1 bg process (eggdrop, screen, etc), FTP access, WWW
homepage hosting, WWW domain hosting ($20.00 a year) (http://www.yourname.com),
WWW virtual hosting http://yourname.cgladue.com) (Free!!), PHP 4, MySQL, cgi-bi
n, webmail.<br><b>IRC access:</b> Yes.<br><b>Background processes allowed:</b> Y
--More-- (30%)
```

The reference in this screen to eggdrop can be seen four lines from the bottom, and in the center. The other text around this reference refers to text on a website that was browsed on this system.

Furthermore, I found a reference to a message board where a posting referred to tunneling POP (mail traffic) over ssh:

© SANS Institute

Ssh reference:



```
root@localhost:~/bunk/morebunk/output
<option value="3918">041692 - [redacted] Europe</option>
sSh"
SShBB
ISsHGJ<$
usCrosshairck
IsShadowed
  <option value="[redacted]">[redacted]</option>
  <option value="[redacted]">[redacted]</option>
  <option value="[redacted]">[redacted]</option>
  <option value="[redacted]">[redacted]</option>
  <option value="[redacted]">[redacted]</option>
  <option selected="selected" value="[redacted]">[redacted]</option>
  <option selected="selected" value="[redacted]">[redacted]</option>
SSSSPSSSh
  <FONT FACE="verdana,helvetica,arial" SIZE="3" CO
  LOR="#FFFFFF"><A NAME="18"><B>ssh tunnel rendezvous?</B></A> (Score:0)</FONT>
  I was looking at this and got to wondering if it is possible to ssh tunnel a ren
  devous connection to connect remote peers securely (I know that Hydra and iStor
  m do not require redevous) This would also enable one to secure iChat and other
  stuff.<BR>
  <LI><A HREF="http://www.macslash.org/comments.pl?sid=03/04/02/15
  27205&threshold=0&commentsort=0&mode=thread&pid=18#25">Re:ssh tunnel rendezvous?<
  /A> by Anonymous Coward <FONT SIZE="-1"> Thursday April 03, 809:33AM EST </FONT>
--More-- (54%)
```

The discussion regarding ssh was displayed at the bottom half of this screen shot.

This made me wonder if the suspect was merely looking or attempting to tunnel a protocol himself. However, I was not able to find anything that supported this conclusion.

10. After looking at some other areas on this disk, it became apparent that I needed to look for the word encryption in case there was a transfer of files occurring via an encrypted session. I searched for the term **encrypt**, and 3490 instances were found on the disk. It appeared that the references that were pulled had a lot to do with either the encryption function within Netscape (SSL based), encrypted cookies as the suspect made his way around the Internet, or looking at help files on the term encryption for use within Microsoft products (it seems that he was trying to encrypt his mail, since there were also references to SMTP as he looked through the help files). There was also a reference to encrypted mp3s from the MUSICMATCH program that he had installed.

While reviewing the output of the encryption search, these other references were discovered. I was unfamiliar with these particular files, so I researched them further.

[688 1408][18 Sep 10:55:56] fwuserc_loadtopo: Added gw 193.128.xxx.xxx to userc_encrypt_DNS

[596 920][24 Sep 12:56:49] fwuserc_loadtopo: Added gw 193.128.xxx.xxx to userc_encrypt_DNS

This activity seemed unusual; however, this address resolved to a business unit within our company. At first look, it is not normal for some sort of external DNS activity to be occurring on a desktop (DNS is the domain name service used to translate Internet addresses to server names).

('http://www.koingosw.com/products/password_retriever.shtml') This also sounded suspicious, but turned out to be a program to store passwords, not sniff them off of the network.

cb32d479.exe- This turned out to be the Netscape installer.

Source: http://ftp.cvut.cz/communicator/4.79/windows/windows95_or_nt/base_install/

Review of Sniffer Logs:

When the case was first brought to my attention, I sought the assistance of our Network team to use their sniffer to monitor the Internet usage of the suspect, as we had no other tool to aid us in monitoring his Internet activity at the time. This aided me in my investigation in two ways: 1) I would be able to see if, and what type of, files were in fact being transferred outside the company, and 2) I could look for aberrant behavior from the system which might help me narrow down the review of his hard disk for malicious code.

It was noted that there was a time skew between my system's current time and the time on the server that held the sniffer. Therefore, while the date was often helpful, the exact time of the access to Internet locations was ignored. The suspect only worked during day hours and to my knowledge, and there was no remote access enabled to the system during off hours. The department that this individual worked in does not require it to perform their jobs.

To ensure that the traffic belonged to the suspect, it was verified that he was at work on this specific PC on the days that the sniffer traffic was recorded. One notable exception was a co-worker who was logged in on April 10th. The data on this day was not used in this review.

Regarding the media analysis, there was a lot of information to sift through in the sniffer logs. I decided up front that only non-http traffic would be examined, as the real focus of the case was whether information was being transferred outside of the company. The http traffic could be examined at a later date if necessary.

The first sniffer log was created on April 7, 2003. The first traffic reviewed was a secure shell session to a server at a consultancy in Austin, TX. Very little information was found about the host of this server, and it was difficult to determine what activity occurred because the entire session was encrypted.

The second session of concern was an ftp session to the suspect's Yahoo account. Without legal assistance, I would not be able to get a closer look at this data. But its existence was noted for future reference if required.

Yet another ftp server was located in a later capture, but it was used to download rfc822 from an anonymous server. RFC822 deals with the standard format of ARPA Internet Text Messages, not to transfer files outside of the company.

A third ftp session was discovered on April 15th. This was a connection to the same server that the suspect had been noted as having an ssh session to in the earlier captures. This ftp session was to takasi.freeshell.org, which is a "networked community of free software authors". It was noted that a lot of time was spent at this site.

Conclusions:

As far as my suspect in this case goes, it was evident that I had a music lover who maintaining his own website. His bragging about being able to telnet outside the company was no more dangerous than just bragging. I found no evidence whatsoever of corruption on his disk, no signs of hacking tools, no signs of disk wiping tools, no sniffers, no backdoor programs or any efforts to steal company data. If anything, he possibly violated the approved software policy (and possibly the music copyright laws) and wasted a lot of company time on email, maintaining his site, and downloading music.

However, there was one notable exception to this conclusion. It was difficult to be 100% conclusive that all of the files (save for perhaps the files found in the suspect's My Documents directory) all belonged to him. This computer was available essentially by anyone in his department, and was actually logged into by one of those co-workers. A review of the co-worker's files made it appear that he did not use this particular computer often; however, that does not mean that he was above suspicion. Along this same line of thought, it was difficult to ascertain who wrote out the many third part tools because they all took administrative privileges when written to the disk.

It would have been very useful in this investigation if I had sought to correlate what activity had taken place on the internal server where the files were certainly transferred to; however, it was believed that generic accounts were used by this department to access the server, so even this method could not have been 100% accurate in determining who made the transfers.

There were numerous lessons learned from this project, both from a forensic perspective and from a security policy perspective. I will address the forensic lessons learned first.

Although it was very convenient to have the external USB hard drive available to dump images to, I learned the hard way that I should have been more prepared to work with it.

My version of Linux was not current enough to be able to properly mount the drive to make my analysis easier. This didn't turn out to be a huge problem, because I was still able to use the command line programs to get what I needed, but I spent too much time analyzing and trying to fix this issue, and if this had been a real urgent case that I needed to turn around quickly, I would have been in deep trouble. As it turns out, I was able to do enough analysis outside the media analysis to support my conclusions.

Secondly from a forensic perspective, I made a huge mistake when I mounted the copy of the disk image to do my analysis. I didn't mark it read only, and changed the access times on the file. This would not be a proper thing in an official investigation either. Furthermore, I neglected to use netstat -a when I had my hands on the original system, so I was not able to make any conclusions as to which ports may have been open on the system. I believe that I made this oversight because I was not as concerned with finding malicious code as I was finding evidence that the suspect was transferring files outside the company; however, this makes for an incomplete analysis.

Regarding lessons learned about the company's security policy, I realized from this incident that our Incident Handling policy needed to be updated to include better standards regarding correlation of information. I had to run around and gather a lot of evidence from different sources that weren't necessarily prepared to handle my requests (mostly from a workload perspective). So, in order to be prepared to do more cross-departmental correlation efforts in the future as far as Internet monitoring and forensic analysis, I have initiated training classes with the technical groups in my company to help get a more common understanding of what support was needed as well as how the process should work.

Part 3- Legal Issues of Incident Handling

Question A:

What, if any, information can you provide to the law enforcement officer over the phone during the initial contact?

The minimum information that I must provide at the time of initial contact is whether or not I have the logs (and if I do, then I would say that I do). No further detail is required of me at this time, and I would not provide any until I look into the issue further.

At the time of the initial contact, with no warrant or other evidence provided at that time, I as the ISP can verify that I will look into it, look at the relevant log files, and assure the inquiring authority that I will be cooperative in the case. It does me no good to not cooperate, because 1) you never know when you will need law enforcement's assistance in the future, and 2) you should verify that the account in question was in fact used in the attack or not. Lastly, I would want to examine more closely my own environment to ensure that no peripheral damage is being done at the time, to stop the bleeding if other damage was still going on, and to eradicate the issue.

Legally, I would not need to confirm or deny whether this account did belong to one of our systems or users, but I will certainly review the issue in accordance with my Incident Response policy.

This issue can be very tricky as there are many principles of the law that apply. For one, if I act as an instrument of the law and conduct the search, then that falls under the Fourth Amendment. If I do it privately though, based on some other notion that I should look into it, then it does not.

With the reasonable expectation of privacy notion not necessarily applying to me as a third-party service provider, I do not legally have responsibility to do exactly what was requested without jeopardizing the privacy of my clients, but as a business owner who needs to ensure that my clients retain their privacy as long as they keep their activity legal, then I need to minimize what I provide to law enforcement. A fine line, but both parties can be satisfied. The Fourth Amendment does not appear to apply to service providers in this case.

Either way, my company would have to provide consent to the officer if they wanted to review the logs without a warrant.

According to the Arizona Revised Statutes (I work in Arizona), I would need permission from the owner or operator of the system to release any confidential security information; however, as the ISP, I am the owner, so I would have to make the call. It's interesting to note that the Arizona law does not refer to stored records, but rather concentrates on security information, which implies to me that it is more focused on configuration and account information than log files.

The full statute:

13-2316.02. Unauthorized release of proprietary or confidential computer security information; exceptions; classification

A. A person commits unauthorized release of proprietary or confidential computer security information by communicating, releasing or publishing proprietary or confidential computer security information, security-related measures, algorithms or encryption devices relating to a particular computer, computer system or network without the authorization of its owner or operator.

B. The following are exempt from this section:

1. The release by publishers, vendors, users and researchers of warnings or information about security measures or defects in software, hardware or encryption products if the release of the warnings or information was not specific to a particular owner's or operator's computer, computer system or network.
2. The release of security information among the authorized users of a computer, computer system or network or the notification to the owner or operator of a computer, computer system or network of a perceived security threat.

3. The release of security information in connection with the research, development and testing of security-related measures, products or devices if the release of the security information was not specific to a particular owner's or operator's computer, computer system or network.

C. At the conclusion of any grand jury, hearing or trial, the court shall preserve pursuant to section 44-405 any proprietary computer security information that was admitted in evidence or any portion of a transcript that contains information relating to proprietary computer security information.

D. Unauthorized release of proprietary or confidential computer security information is a class 6 felony, unless the security information relates to a critical infrastructure resource, in which case it is a class 4 felony.

Source: <http://www.azleg.state.az.us/ars/13/02316-02.htm>

Again, to summarize, I would acknowledge to law enforcement that I did have the logs files, without giving them specific information. Then, I would immediately launch my own investigation to assess and contain any damage done to my systems.

Question B:

What must the law enforcement officer do to ensure you preserve this evidence if there was a delay in obtaining any required legal authority?

According to Title 18 U.S.C. 2703, if the proper legal authority is not available, a letter issued by the government is good for 90 days (with a subsequent 90 day renewable period available). After reviewing Arizona state laws, it would appear that this same principle applies for state law enforcement. So, if I received such a request stating that log files to be used as evidence in this case needed to be preserved until they could be legally handed over (with the warrant, court order or subpoena that was due), then I would be obligated to do so, and will comply.

An interesting twist to Arizona state law was that evidence does not necessarily need to be suppressed if it was seized pursuant to a search warrant and if done via a good faith mistake or technical violation. These terms are defined as:

1. "Good faith mistake" means a reasonable judgmental error concerning the existence of facts that if true would be sufficient to constitute probable cause.

2. "Technical violation" means a reasonable good faith reliance on:

(a) A statute that was subsequently ruled unconstitutional.

(b) A warrant that was later invalidated due to a good faith mistake.

(c) A controlling court precedent that was later overruled, unless the court overruling the precedent orders the new precedent to be applied retroactively.

So, if evidence was obtained prior to the warrant being presented and it falls into one of the categories described above, it still may be admissible in court (<http://www.azleg.state.az.us/ars/13/03925.htm>)

Question C:

What legal authority, if any, does the law enforcement officer need to provide to you in order for you to send him your logs?

To answer this question, I classify the information that the law enforcement agent is to be after as stored records or other information, not stored contents of electronic communications. In this case, I would simply need a Court Order under the 18 U.S.C. § 2703(d).

Under the Electronic Communications Privacy Act (ECPA), I would have to be very careful what information I handed over to law enforcement, because I might be violating the act by handing over account information from other customers not involved in the crime.

I could also voluntarily hand over information that fell into one of the following categories (for non-customer content) without violating the ECPA:

- 1) the disclosure "may be necessarily incident to the rendition of the service or to the protection of the rights or property of the provider of that service," § 2702(c)(3);
- 2) the provider "reasonably believes that an emergency involving immediate danger of death or serious physical injury to any person" justifies disclosure, § 2702(c)(4); or
- 3) the disclosure was made with the consent of the intended recipient, or pursuant to a court order or legal process § 2702(c)(1)-(2).

Source: <http://www.cybercrime.gov/s&smanual2002.htm>

Question D:

What other investigative activity are you permitted to do at this time?

Now that I have been made aware of a compromise, I am permitted to examine our systems to make sure that the integrity of my systems has not been compromised, and also to contain the damage being done and verify that no peripheral damage has taken place.

It's in my best interest to fully investigate the case to ensure that no wrongdoing from our perspective has occurred, and also to ensure that the integrity of our systems was not compromised. This falls under my Incident Handling policy: identify the problem, analyze the system, collect data, clean up the system, and then return it to an operational state. I would go even further to make sure that peripheral systems were not involved in the attack, or that any of our other customers were affected.

I am permitted to take every precaution to fully understand what my company's involvement was to 1) make sure that a similar incident cannot occur again, and 2) that I was fully cooperating with the law in the case. If something else was uncovered in the case that is not covered by the warrant or court order, then I do not have to volunteer that information.

Question E:

How would your actions change if your logs disclosed a hacker gained unauthorized access to your system at some point, created an account for him/her to use, and used THAT account to hack into the government system?

My actions would only change slightly in the sense that I would proactively notify the proper external authority, as well as the proper internal authority. If this was in fact a government system, and I found information on the hack in my system logs, I would voluntarily hand them over right away. The USA Patriot Act clarified that I can do this without the proper legal paperwork from the law enforcement agency, especially if I knew this situation involved an emergency, where someone's life was in danger, terrorism is involved or if child pornography was involved. Furthermore, if I act in good faith and provide the logs, I can avoid civil and/or criminal charges.

<http://www.usdoj.gov/criminal/cybercrime/usc2702.htm>

© SANS Institute 2003, Author retains full rights.

Appendix A: Regmon output

5125	22.45843	target2.exe:1512	OpenKey	HKLM\Software\Microsoft\Windows NT\CurrentVersion\Image File Execution Options\target2.exe	NOTFOUND	
5126	22.4585	target2.exe:1512	OpenKey	HKLM\Software\Microsoft\Windows NT\CurrentVersion\Image File Execution Options\target2.exe	NOTFOUND	
5127	22.45925	target2.exe:1512	OpenKey	HKLM\Software\Microsoft\Windows NT\CurrentVersion\Image File Execution Options\target2.exe	NOTFOUND	
5292	23.20839	target2.exe:1512	OpenKey	HKLM\Software\Microsoft\Windows NT\CurrentVersion\Image File Execution Options\target2.exe	NOTFOUND	
5293	23.21052	target2.exe:1512	OpenKey	HKLM\System\CurrentControlSet\Control\Session Manager	SUCCESS	Key: 0xE2E72600
5294	23.21059	target2.exe:1512	QueryValue	HKLM\System\CurrentControlSet\Control\Session Manager\SafeDllSearchMode	NOTFOUND	
5295	23.21067	target2.exe:1512	CloseKey	HKLM\System\CurrentControlSet\Control\Session Manager	SUCCESS	Key: 0xE2E72600
5296	23.21106	target2.exe:1512	OpenKey	HKLM\SOFTWARE\Microsoft\Windows NT\CurrentVersion\Winlogon	SUCCESS	Key: 0xE2E72600
5297	23.21112	target2.exe:1512	QueryValue	HKLM\SOFTWARE\Microsoft\Windows NT\CurrentVersion\Winlogon\LeakTrack	NOTFOUND	
5299	23.21201	target2.exe:1512	CloseKey	HKLM\SOFTWARE\Microsoft\Windows NT\CurrentVersion\Winlogon	SUCCESS	Key: 0xE2E72600
5300	23.21229	target2.exe:1512	OpenKey	HKLM	SUCCESS	Key: 0xE2E72600
5301	23.21237	target2.exe:1512	OpenKey	HKLM\Software\Microsoft\Windows NT\CurrentVersion\Diagnostics	NOTFOUND	
5302	23.21368	target2.exe:1512	OpenKey	HKLM\System\CurrentControlSet\Control\Error Message Instrument\	NOTFOUND	
5303	23.21428	target2.exe:1512	OpenKey	HKLM\Software\Microsoft\Windows NT\CurrentVersion\Compatibility32	SUCCESS	Key: 0xE2DEADC0
5304	23.21435	target2.exe:1512	QueryValue	HKLM\Software\Microsoft\Windows NT\CurrentVersion\Compatibility32\target2	NOTFOUND	
5305	23.21441	target2.exe:1512	CloseKey	HKLM\Software\Microsoft\Windows NT\CurrentVersion\Compatibility32	SUCCESS	Key: 0xE2DEADC0
5306	23.21454	target2.exe:1512	OpenKey	HKLM\Software\Microsoft\Windows NT\CurrentVersion\Compatibility2	SUCCESS	Key: 0xE2DEADC0
5307	23.21464	target2.exe:1512	QueryValue	HKLM\Software\Microsoft\Windows NT\CurrentVersion\Compatibility2\target20.0	NOTFOUND	
5308	23.2147	target2.exe:1512	CloseKey	HKLM\Software\Microsoft\Windows NT\CurrentVersion\Compatibility2	SUCCESS	Key: 0xE2DEADC0
5309	23.21481	target2.exe:1512	OpenKey	HKLM\Software\Microsoft\Windows NT\CurrentVersion\IME	SUCCESS	Key: 0xE2DEADC0
5310	23.21487	target2.exe:1512	QueryValue	HKLM\Software\Microsoft\Windows NT\CurrentVersion\IME	NOTFOUND	
5311	23.21493	target2.exe:1512	CloseKey	Compatibility\target2	SUCCESS	Key: 0xE2DEADC0
5312	23.21547	target2.exe:1512	OpenKey	HKLM\System\CurrentControlSet\Control\Session Manager\AppCompatibility\target2.exe	NOTFOUND	
5313	23.21578	target2.exe:1512	OpenKey	HKLM\Software\Microsoft\Windows NT\CurrentVersion\Windows	SUCCESS	Key: 0xE2DEADC0
5317	23.22162	target2.exe:1512	QueryValue	HKLM\Software\Microsoft\Windows NT\CurrentVersion\Windows\ApplInit_DLLs	SUCCESS	
5318	23.22174	target2.exe:1512	CloseKey	HKLM\Software\Microsoft\Windows	SUCCESS	Key: 0xE2DEADC0

				NT\CurrentVersion\Windows		
5385	23.24981	target2.exe:1512	OpenKey	HKCU	SUCCESS	Key: 0xE2E392E0
5386	23.24997	target2.exe:1512	OpenKey	HKLM\System\CurrentControlSet\Control\Nls\MUILanguages	SUCCESS	Key: 0xE2D92360
5387	23.25003	target2.exe:1512	CloseKey	HKLM\System\CurrentControlSet\Control\Nls\MUILanguages	SUCCESS	Key: 0xE2D92360
5388	23.25013	target2.exe:1512	OpenKey	HKCU\Software\Policies\Microsoft\Control Panel\Desktop	NOTFOUND	
5389	23.25041	target2.exe:1512	OpenKey	HKCU\Control Panel\Desktop	SUCCESS	Key: 0xE2D92360
5390	23.25048	target2.exe:1512	QueryValue	HKCU\Control Panel\Desktop\MultiUILangageId	NOTFOUND	
5391	23.25054	target2.exe:1512	CloseKey	HKCU\Control Panel\Desktop	SUCCESS	Key: 0xE2D92360
5392	23.25059	target2.exe:1512	CloseKey	HKCU	SUCCESS	Key: 0xE2E392E0
5403	23.29987	target2.exe:1512	OpenKey	HKLM\System\CurrentControlSet\Control\ServiceCurrent	SUCCESS	Key: 0xE2DEADC0
5404	23.29995	target2.exe:1512	QueryValue	HKLM\System\CurrentControlSet\Control\ServiceCurrent\Default	SUCCESS	0xE
5405	23.30004	target2.exe:1512	CloseKey	HKLM\System\CurrentControlSet\Control\ServiceCurrent	SUCCESS	Key: 0xE2DEADC0
5658	38.36003	target2.exe:1512	CloseKey	HKLM	SUCCESS	Key: 0xE2E72600

Appendix B: Filemon output

9956	3:20:18 PM target2.exe:1512	IRP_MJ_QUERY_INFORMA TION	C:\Documents and Settings\cpw\Desktop\target2.exe	SUCCESS
9957	3:20:18 PM target2.exe:1512	IRP_MJ_CREATE	C:\Documents and Settings\cpw\Desktop	SUCCESS
9958	3:20:18 PM target2.exe:1512	IRP_MJ_READ*	C:\Documents and Settings\cpw\Desktop\target2.exe	SUCCESS
9959	3:20:18 PM target2.exe:1512	FSCTL_IS_VOLUME_MOUN TED	C:\Documents and Settings\cpw\Desktop	SUCCESS
9960	3:20:18 PM target2.exe:1512	FASTIO_QUERY_OPEN	C:\WINNT\System32\WS2_32.dll	SUCCESS
9961	3:20:18 PM target2.exe:1512	FSCTL_IS_VOLUME_MOUN TED	C:\Documents and Settings\cpw\Desktop	SUCCESS
9962	3:20:18 PM target2.exe:1512	FASTIO_QUERY_OPEN	C:\WINNT\System32\WS2_32.dll	SUCCESS
9963	3:20:18 PM target2.exe:1512	FSCTL_IS_VOLUME_MOUN TED	C:\Documents and Settings\cpw\Desktop	SUCCESS
9964	3:20:18 PM target2.exe:1512	FASTIO_QUERY_OPEN	C:\WINNT\System32\WS2_32.dll	SUCCESS
9965	3:20:18 PM target2.exe:1512	FSCTL_IS_VOLUME_MOUN TED	C:\Documents and Settings\cpw\Desktop	SUCCESS
9966	3:20:18 PM target2.exe:1512	FSCTL_IS_VOLUME_MOUN TED	C:\Documents and Settings\cpw\Desktop	SUCCESS
9967	3:20:18 PM target2.exe:1512	IRP_MJ_CREATE	C:\WINNT\System32\WS2_32.dll	SUCCESS
9968	3:20:18 PM target2.exe:1512	IRP_MJ_CLEANUP	C:\WINNT\System32\WS2_32.dll	SUCCESS
9969	3:20:18 PM target2.exe:1512	IRP_MJ_CLOSE	C:\WINNT\System32\WS2_32.dll	SUCCESS
9970	3:20:18 PM target2.exe:1512	FSCTL_IS_VOLUME_MOUN TED	C:\Documents and Settings\cpw\Desktop	SUCCESS
9971	3:20:18 PM target2.exe:1512	FASTIO_QUERY_OPEN	C:\WINNT\System32\WS2_32.dll	SUCCESS
9972	3:20:18 PM target2.exe:1512	FSCTL_IS_VOLUME_MOUN TED	C:\Documents and Settings\cpw\Desktop	SUCCESS
9973	3:20:18 PM target2.exe:1512	FASTIO_QUERY_OPEN	C:\WINNT\System32\WS2_32.dll	SUCCESS
9974	3:20:18 PM target2.exe:1512	FSCTL_IS_VOLUME_MOUN TED	C:\Documents and Settings\cpw\Desktop	SUCCESS
9975	3:20:18 PM target2.exe:1512	FASTIO_QUERY_OPEN	C:\WINNT\System32\WS2_32.dll	SUCCESS
9976	3:20:18 PM target2.exe:1512	FSCTL_IS_VOLUME_MOUN TED	C:\Documents and Settings\cpw\Desktop	SUCCESS
9977	3:20:18 PM target2.exe:1512	FSCTL_IS_VOLUME_MOUN TED	C:\Documents and Settings\cpw\Desktop	SUCCESS
9978	3:20:18 PM target2.exe:1512	IRP_MJ_CREATE	C:\WINNT\System32\WS2HELP.D LL	SUCCESS
9979	3:20:18 PM target2.exe:1512	IRP_MJ_CLEANUP	C:\WINNT\System32\WS2HELP.D LL	SUCCESS
9980	3:20:18 PM target2.exe:1512	IRP_MJ_CLOSE	C:\WINNT\System32\WS2HELP.D LL	SUCCESS
9981	3:20:18 PM target2.exe:1512	FSCTL_IS_VOLUME_MOUN TED	C:\Documents and Settings\cpw\Desktop	SUCCESS
9982	3:20:18 PM target2.exe:1512	FASTIO_QUERY_OPEN	C:\WINNT\System32\WS2_32.dll	SUCCESS
9983	3:20:18 PM target2.exe:1512	FSCTL_IS_VOLUME_MOUN TED	C:\Documents and Settings\cpw\Desktop	SUCCESS
9984	3:20:18 PM target2.exe:1512	FASTIO_QUERY_OPEN	C:\WINNT\System32\WS2_32.dll	SUCCESS
9985	3:20:18 PM target2.exe:1512	FSCTL_IS_VOLUME_MOUN TED	C:\Documents and Settings\cpw\Desktop	SUCCESS
9986	3:20:18 PM target2.exe:1512	FASTIO_QUERY_OPEN	C:\WINNT\System32\WS2_32.dll	SUCCESS
9987	3:20:18 PM target2.exe:1512	FSCTL_IS_VOLUME_MOUN TED	C:\Documents and Settings\cpw\Desktop	SUCCESS
9988	3:20:18 PM target2.exe:1512	FSCTL_IS_VOLUME_MOUN TED	C:\Documents and Settings\cpw\Desktop	SUCCESS
10057	3:20:19 PM target2.exe:1512	IRP_MJ_CREATE	C:\WINNT\System32\MFC42.DLL	SUCCESS
10058	3:20:19 PM target2.exe:1512	IRP_MJ_CLEANUP	C:\WINNT\System32\MFC42.DLL	SUCCESS
10059	3:20:19 PM target2.exe:1512	IRP_MJ_CLOSE	C:\WINNT\System32\MFC42.DLL	SUCCESS
10060	3:20:19 PM target2.exe:1512	IRP_MJ_READ*	C:\WINNT\system32\mfc42.dll	SUCCESS
10061	3:20:19 PM target2.exe:1512	FSCTL_IS_VOLUME_MOUN TED	C:\Documents and Settings\cpw\Desktop	SUCCESS
10062	3:20:19 PM target2.exe:1512	FASTIO_QUERY_OPEN	C:\WINNT\System32\WS2_32.dll	SUCCESS

10063	3:20:19 PM target2.exe:1512	FSCTL_IS_VOLUME_MOUNTED	C:\Documents and Settings\cpw\Desktop	SUCCESS
10064	3:20:19 PM target2.exe:1512	FASTIO_QUERY_OPEN	C:\WINNT\System32\WS2_32.dll	SUCCESS
10065	3:20:19 PM target2.exe:1512	FSCTL_IS_VOLUME_MOUNTED	C:\Documents and Settings\cpw\Desktop	SUCCESS
10066	3:20:19 PM target2.exe:1512	FASTIO_QUERY_OPEN	C:\WINNT\System32\WS2_32.dll	SUCCESS
10067	3:20:19 PM target2.exe:1512	FSCTL_IS_VOLUME_MOUNTED	C:\Documents and Settings\cpw\Desktop	SUCCESS
10068	3:20:19 PM target2.exe:1512	FSCTL_IS_VOLUME_MOUNTED	C:\Documents and Settings\cpw\Desktop	SUCCESS
10142	3:20:19 PM target2.exe:1512	IRP_MJ_CREATE	C:\WINNT\System32\MSVCP60.dll	SUCCESS
10143	3:20:19 PM target2.exe:1512	IRP_MJ_CLEANUP	C:\WINNT\System32\MSVCP60.dll	SUCCESS
10144	3:20:19 PM target2.exe:1512	IRP_MJ_CLOSE	C:\WINNT\System32\MSVCP60.dll	SUCCESS
10145	3:20:19 PM target2.exe:1512	IRP_MJ_READ*	C:\WINNT\system32\MSVCP60.DLL	SUCCESS
10146	3:20:19 PM target2.exe:1512	FSCTL_IS_VOLUME_MOUNTED	C:\Documents and Settings\cpw\Desktop	SUCCESS
10147	3:20:19 PM target2.exe:1512	FASTIO_QUERY_OPEN	C:\WINNT\System32\WS2_32.dll	SUCCESS
10156	3:20:19 PM target2.exe:1512	IRP_MJ_READ*	C:\WINNT\system32\mf42.dll	SUCCESS
10194	3:20:19 PM target2.exe:1512	IRP_MJ_READ*	C:\WINNT\system32\mf42.dll	SUCCESS
10195	3:20:19 PM target2.exe:1512	FSCTL_IS_VOLUME_MOUNTED	C:\Documents and Settings\cpw\Desktop	SUCCESS
10196	3:20:19 PM target2.exe:1512	FASTIO_QUERY_OPEN	C:\WINNT\System32\MFC42LOC.DLL	SUCCESS
10197	3:20:19 PM target2.exe:1512	FSCTL_IS_VOLUME_MOUNTED	C:\Documents and Settings\cpw\Desktop	SUCCESS
10198	3:20:19 PM target2.exe:1512	FASTIO_QUERY_OPEN	C:\WINNT\System32\MFC42LOC.DLL	SUCCESS
10199	3:20:19 PM target2.exe:1512	IRP_MJ_READ*	C:\WINNT\system32\MSVCP60.DLL	SUCCESS
10200	3:20:19 PM target2.exe:1512	IRP_MJ_READ*	C:\Documents and Settings\cpw\Desktop\target2.exe	SUCCESS
10201	3:20:19 PM target2.exe:1512	IRP_MJ_READ*	C:\Documents and Settings\cpw\Desktop\target2.exe	SUCCESS
11654	3:20:34 PM target2.exe:1512	IRP_MJ_READ*	C:\WINNT\system32\mf42.dll	SUCCESS
11655	3:20:34 PM target2.exe:1512	IRP_MJ_READ*	C:\WINNT\system32\MSVCP60.DLL	SUCCESS
11656	3:20:34 PM target2.exe:1512	IRP_MJ_CLEANUP	C:\Documents and Settings\cpw\Desktop	SUCCESS
11657	3:20:34 PM target2.exe:1512	IRP_MJ_CLOSE	C:\Documents and Settings\cpw\Desktop	SUCCESS

Resources utilized:

From Part 1

<http://www.oberon.ethz.ch/ethoberon/defs/MD5.Def.html> MD5 definition
<http://web.vip.hr/inga.vip/index.htm> Debuggy (the debugger that I was attempting to use. The debugger program worked, my potentially malicious code did not)
<http://www.geocities.com/~sangcho/disasm.html> Win32Progam Disassembler
[http://users.erols.com/gmgarner/forensics/forensic%20acquisition%20utilities-bin-1.0.0.1032%20\(beta1\).zip](http://users.erols.com/gmgarner/forensics/forensic%20acquisition%20utilities-bin-1.0.0.1032%20(beta1).zip) The forensic acquisition utilities written by George Garner.
<http://www.foundstone.com/resources/termsofuse.htm?file=bintext.zip> Link to the bintext utility.
<http://www.tburke.net/info/suptools/topics/reg.htm> (reg.exe description)
[Http://docs.sun.com/db/doc/816-0210/6m6nb7mm2?a=view](http://docs.sun.com/db/doc/816-0210/6m6nb7mm2?a=view) (Definition of Strings)
<http://www.unicode.org/standard/WhatIsUnicode.html> (Definition of Unicode)
<http://www.winalysis.com/wnaly300.exe> Winalysis evaluation copy

Dll info:

http://www.webopedia.com/TERM/K/kernel32_dll.html
<http://www.liutilities.com/products/wintaskspro/dlllibrary/advapi32/>
<http://www.fortunecity.com/skyscraper/fortune/570/msvcrt.html>
<http://www.fortunecity.com/skyscraper/fortune/570/mfc42.html>
<http://www.liutilities.com/products/wintaskspro/dlllibrary/msvc60/>

Code links:

<http://mcking.8u8.com/hkwz18.htm>
<http://ouah.kernsh.org/progs/007shell.tgz>
<http://packetstormsecurity.nl/UNIX/penetration/rootkits/allinone.c>
<http://www.google.com/search?hl=en&ie=UTF-8&oe=UTF-8&q=icmp+BackDoor+V0.1>
http://www.s0ftpj.org/tools/icmp_tunnel.h

Legal Resources:

<http://www.cybercrime.gov/s&smanual2002.htm>
<http://www4.law.cornell.edu/uscode/18/1030.html>
<http://www.ussc.gov/2003guid/2003amendments.pdf>

Others:

<http://neworder.box.sk/newsread.php?newsid=4182> (Information on Kernel rootkits)
<http://www.winalysis.com/default.htm> (winalysis)
<http://users.erols.com/gmgarner/forensics> (wipe utility)
<http://www.sysinternals.com/ntw2k/source/filemon.shtml> (filemon)
<http://www.sysinternals.com/ntw2k/source/regmon.shtml> (regmon)
<http://prdownloads.sourceforge.net/sleuthkit/sleuthkit-1.65.tar.gz?download> (TASK)
<http://prdownloads.sourceforge.net/autopsy/autopsy-1.74.tar.gz?download> (Autopsy)
<http://www.geocities.com/~sangcho/disasm.html> Win32Progam Disassembler

From Part 2

Sources:

<http://www.hyperdictionary.com/computing/inode> Definition of inode
<http://www.phillipsponder.com/orderForm.htm> How to order the Internet History Viewer
<http://www.webopedia.com/TERM/c/cookie.html> Definition of a cookie
<http://www.google.com/search?hl=en&ie=UTF-8&oe=UTF-8&q=andretti.exe> Google search for Andretti
<http://www.google.com/search?hl=en&lr=&ie=UTF-8&oe=UTF-8&q=ResumeMaker.exe> First google search for Resume Maker
<http://www.google.com/search?hl=en&lr=&ie=UTF-8&oe=UTF-8&q=ResumeMaker> 2nd Google search on Resume Maker
The search for Rashme.exe: <http://www.google.com/search?hl=en&lr=&ie=UTF-8&oe=UTF-8&q=Rashme.exe>
Game site: <http://www.kahncentral.net/files/games.txt>
The search for Roadrash.exe: <http://www.google.com/search?hl=en&lr=&ie=UTF-8&oe=UTF-8&q=Roadrash.exe>
<http://www.symantec.com/avcenter/venc/data/w32.nogrov@mm.html>. Reference to roadrash as part of a virus
The search for Sabrina.exe: <http://www.google.com/search?hl=en&lr=&ie=UTF-8&oe=UTF-8&q=Sabrina.exe>
Thread referencing Sabrina: <http://easyweb.easynet.co.uk/~gcaselton/spam/babette.html>
Google search for bench.exe: <http://www.google.com/search?hl=en&lr=&ie=UTF-8&oe=UTF-8&q=bench.exe>
Google search for play.exe: <http://www.google.com/search?q=play.exe&hl=en&lr=&ie=UTF-8&oe=UTF-8>
Second reference for play.exe: <http://www.jfwlite.com/messages/Real%20Jukebox%20-20Uninstall%20problem%20with%20play.exe.txt>
Google search for sorry.exe: <http://www.google.com/search?q=sorry.exe&hl=en&lr=&ie=UTF-8&oe=UTF-8>
Second reference to sorry.exe as a virus: http://www.pspl.com/virus_info/worms/orora.htm
Google search for Southpark.exe: <http://www.google.com/search?hl=en&lr=&ie=UTF-8&oe=UTF-8&q=southpark.exe>
Reference for southpark.exe as a virus: <http://www.symantec.com/avcenter/venc/data/w32.southpark.worm.html>
References to the various programs found in disk and memory review:
http://www.itec.suny.edu/scsys/vms/system%20tools/windows/RCM/docs/winnt/nt_compare.htm
<http://www.liutilities.com/products/wintaskspro/processlibrary/sndrec32/>
<http://support.microsoft.com/default.aspx?kbid=329771>
<http://fresh.t-systems-sfr.com/pc/src/www/.warix/mozilla-win32-1.3.1-talkback.zip.html>
http://www.und.ac.za/und/arch/arch/www/gallery/Web_Govind/Application%20Data/Microsoft/Installer/%7B00020409-78E1-11D2-B60F-006097C998E7%7D/

http://tecfa.unige.ch/perso/fabien/projects/blogs/install_mt.html
<http://www.video-drivers.com/drivers/15/15358.htm>
<http://www.microsoft.com/mspress/books/sampchap/5958d.asp>
<http://support.microsoft.com/default.aspx?scid=http://support.microsoft.com:80/support/kb/articles/Q242/4/68.ASP&NoWebContent=1>
<http://forums.techguy.org/t150909/s5f2289bd752dc0a24b993271993a2309.html>
http://www.winamp.com/nsdn/winamp3x/skinning/tutorials/4.1-drawer_script.ihtml
<http://www.microsoft.com/technet/treeview/default.asp?url=/technet/prodtechnol/sql/deploy/confeat/CERSTEP.asp>
<http://www.soundcard-drivers.com/drivers/85/85018.htm>
http://www.koingosw.com/products/password_retriever.shtml
http://ftp.cvut.cz/communicator/4.79/windows/windows95_or_nt/base_install/

From Part 3

SANS Forensics Course Materials

<http://www.azleg.state.az.us/ars/13/02316-02.htm>
<http://www.azleg.state.az.us/ars/13/03925.htm>
<http://www.cybercrime.gov/s&smanual2002.htm>
<http://www.usdoj.gov/criminal/cybercrime/usc2702.htm>

© SANS Institute 2003, Author retains full rights.