



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>

GIAC Certified Forensic Analyst (GCFA) Practical Assignment Version 1.3

by
Dennis daCruz

© SANS Institute 2003. Author retains full rights.

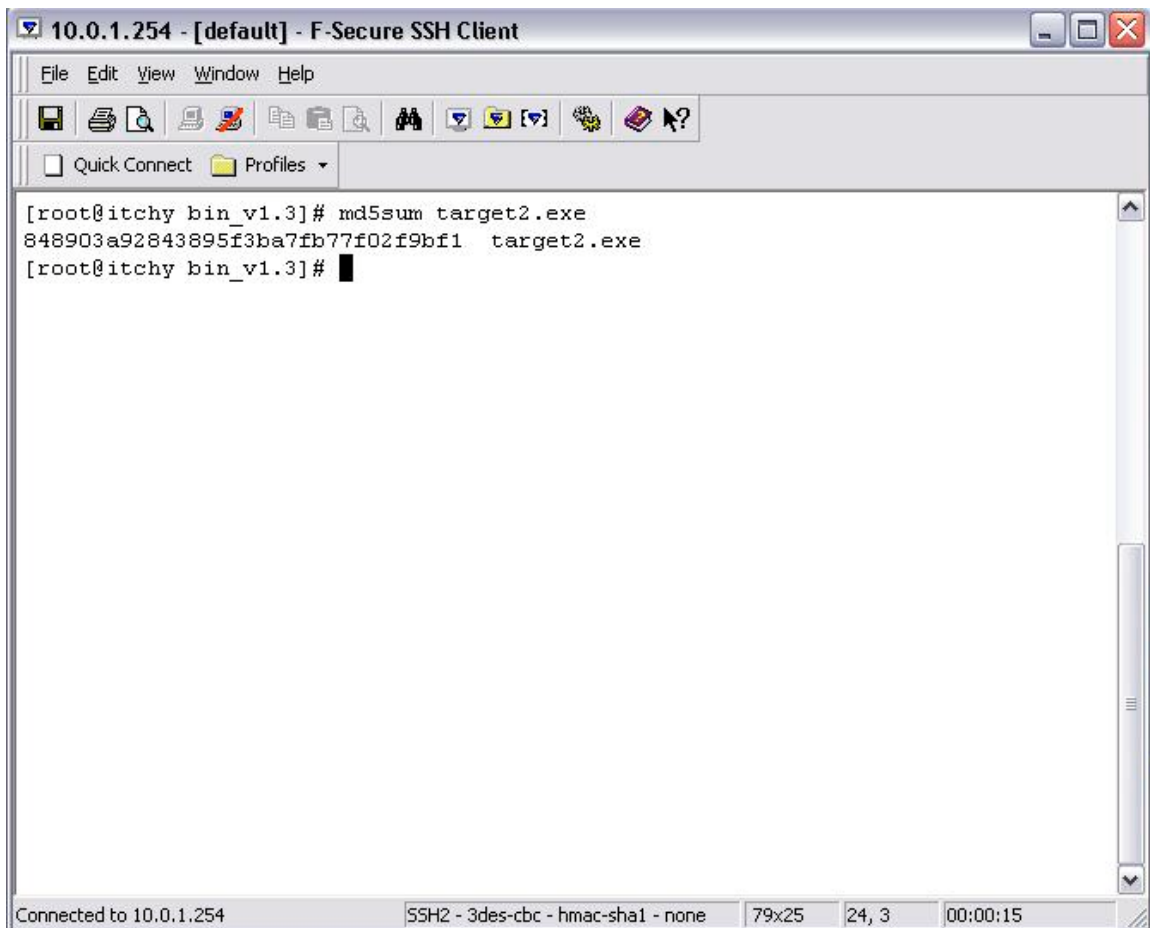
Section I: Binary File Analysis

The first assignment is to investigate an unknown binary. The following procedures will be used to investigate the binary file:

1. Obtain the binary file from the designated location.
(http://www.giac.org/gcfa/binary_v1.3.zip)
2. Gather stat information about the file, information such as the md5sum, the MAC times, type of file, file size, and file owner of the file.
3. Gather string information from the binary. String information could include ASCII and Unicode information from the binary file.
4. Investigate information gathered from the string search on the binary.
5. Use IDA disassembler to gather some Assembly instruction from the binary file.
6. Configure a test environment with proper monitoring tools and isolate, the system, and run the binary.
7. Run the binary in the isolated environment gathering data on what files the binary uses.
8. Document the conclusion found while investigating the binary, including potential countermeasures to the binary.
9. Document potential legal issues with the use of the binary file.
10. Provide a series of questions to ask anyone involved in the creation or discovery of this unknown binary.

Binary Details:

The binary file to be investigated was downloaded from the sans website (http://www.giac.org/gcfa/binary_v1.3.zip). The initial step was to extract the binary and verify the MD5 checksum of the binary. The name of the extracted file was target2.exe. The next step was to calculate the hash value of the binary.

The image is a screenshot of an F-Secure SSH Client window. The title bar reads "10.0.1.254 - [default] - F-Secure SSH Client". The menu bar includes "File", "Edit", "View", "Window", and "Help". Below the menu is a toolbar with various icons. A status bar at the top shows "Quick Connect" and "Profiles". The main terminal area displays the following text:

```
[root@itchy bin_v1.3]# md5sum target2.exe
848903a92843895f3ba7fb77f02f9bf1 target2.exe
[root@itchy bin_v1.3]#
```

The bottom status bar of the window shows "Connected to 10.0.1.254", "SSH2 - 3des-cbc - hmac-sha1 - none", "79x25", "24, 3", and "00:00:15".

screenshot of md5sum of the target2.exe binary

There was no MD5 provided with the file, so it will have to be assumed that this MD5 is the correct value for the executable. The next step is to use the Unix utility file to get some information about the type of file that target2.exe is.

#file target2.exe

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

The file command has shown that the file is a Windows executable. The next step to take in analyzing the binary is to gather the Modify, Access, and Create time information on the binary. The Modify, Access, and Create (MAC) times are stored by the filesystem and provide information about when a file was changed, last accessed, or created on the filesystem. Other statistics such as file owner and file size should also need to be gathered. This information could be gathered using mac_daddy or the Unix utility stat. In this case the stat command will be used because it provides all the required data within one command.

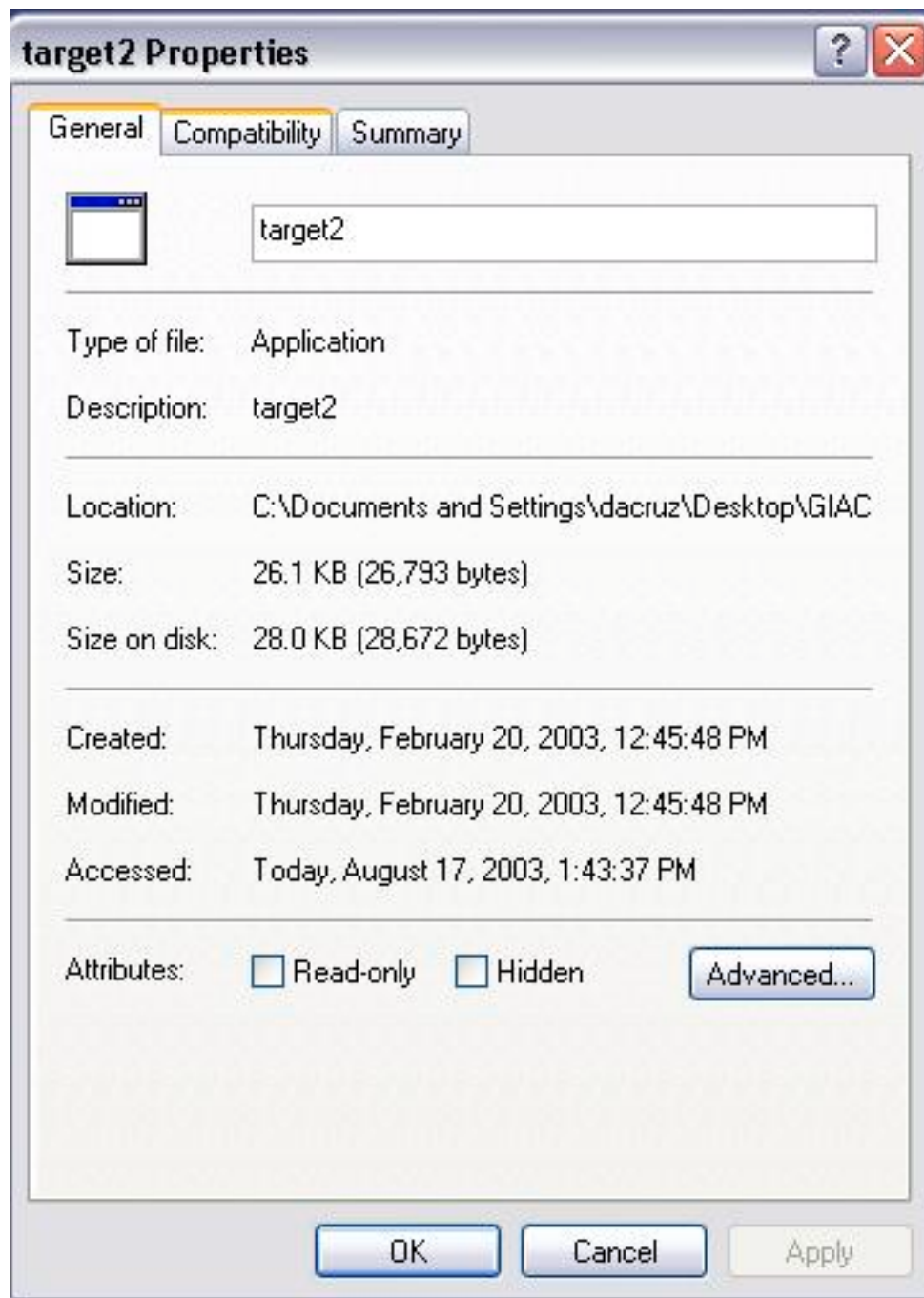
stat target2.exe

File: "target2.exe"

Size: 26793	Blocks: 56	IO Block: 4096	Regular File
Device: 303h/771d	Inode: 246267	Links: 1	
Access: (0644/-rw-r--r--)	Uid: (500/ dacruz)	Gid: (500/ dacruz)	
Access: Thu Aug 7 18:43:04 2003			
Modify: Thu Aug 7 18:43:04 2003			
Change: Thu Aug 7 18:43:04 2003			

The file size is 26793 bytes and the current owner is dacruz, dacruz is the local user account on the forensics system. The MAC times on the file are all August 7, 2003 (the date of the analysis). Because this program appears to be a Windows binary, the analysis of this file will continue using a Windows system. The date information on the stat command did not appear accurate; the information will be view using the property sheet on a Windows system.

© SANS Institute 2003, Author retains full rights.



property information on the target2.exe binary.

The property information showed the modified and creation date as February 20, 2003 at 12:45:48 PM.

Program Description

The next step in the analysis of the binary is to gather any string information from the binary file. The Bintext utility from Foundstone will be used to gather the data.

BinText – can be used to extract Unicode, ASCII text, and Resource strings from a binary file.

<http://www.foundstone.com/index.htm?subnav=resources/navigation.htm&subcontent=/resources/proddesc/bintext.htm>

The following output was captured from the binary using the bintext utility.

File pos	Mem pos	ID	Text
=====	=====	==	=====
0000004D	0040004D	0	!This program cannot be run in DOS mode.
000001D0	004001D0	0	.text
000001F8	004001F8	0	.rdata
0000021F	0040021F	0	@.data
00000248	00400248	0	.rsrc
000011D0	004011D0	0	D\$,QPR
000011FC	004011FC	0	D\$ j'P
0000121E	0040121E	0	T\$,j'RP
000012FE	004012FE	0	T\$,VRS
00001327	00401327	0	D\$ j'P
00001349	00401349	0	T\$,j'RP
00001408	00401408	0	L\$ j'Q
0000142B	0040142B	0	D\$,j'PQ
00001540	00401540	0	D\$0QPR
0000156E	0040156E	0	D\$\$j'P
00001590	00401590	0	T\$0j'RP
00001678	00401678	0	T\$0URV
000016A1	004016A1	0	D\$\$j'P
000016C3	004016C3	0	T\$0j'RP
00001803	00401803	0	D\$ j'PQ
000019AF	004019AF	0	T\$\$QRj
000019CE	004019CE	0	D\$\$PW
00001BD6	00401BD6	0	h0A@
00001CEA	00401CEA	0	SPhxD@
00001D10	00401D10	0	SQhpD@
00001D65	00401D65	0	D\$@SPS
00001E16	00401E16	0	T\$ RP
00001E77	00401E77	0	USSSP3

00002050	00402050	0	x!xu\
00002056	00402056	0	x"iuy
0000205C	0040205C	0	D\$PQ
0000207A	0040207A	0	x#tuP
00002270	00402270	0	IQh@A@
000022B4	004022B4	0	t1h@D@
0000243E	0040243E	0	Ht Ht
00002460	00402460	0	Ph<B@
0000249D	0040249D	0	T\$(QR
00002528	00402528	0	L\$0PQ
000032EA	004032EA	0	Ph0C@
000032F2	004032F2	0	Sleep
000032FE	004032FE	0	HeapAlloc
00003310	00403310	0	GetProcessHeap
00003324	00403324	0	TerminateProcess
00003330	00403330	0	ReadFile
00003340	00403340	0	PeekNamedPipe
0000334E	0040334E	0	CloseHandle
00003360	00403360	0	CreateProcessA
0000336E	0040336E	0	CreatePipe
0000337A	0040337A	0	WriteFile
0000338A	0040338A	0	GetLastError
00003396	00403396	0	LocalAlloc
000033A6	004033A6	0	KERNEL32.dll
000033C4	004033C4	0	StartServiceCtrlDispatcherA
000033D8	004033D8	0	SetServiceStatus
000033F6	004033F6	0	RegisterServiceCtrlHandlerA
0000340C	0040340C	0	CloseServiceHandle
0000341E	0040341E	0	ControlService
00003434	00403434	0	QueryServiceStatus
00003444	00403444	0	OpenServiceA
00003456	00403456	0	CreateServiceA
00003468	00403468	0	OpenSCManagerA
00003478	00403478	0	DeleteService
00003488	00403488	0	StartServiceA
000034A0	004034A0	0	ChangeServiceConfigA
000034B4	004034B4	0	QueryServiceConfigA
000034C4	004034C4	0	ADVAPI32.dll
000034D0	004034D0	0	WSAIoctl
000034DC	004034DC	0	WSASocketA
000034E8	004034E8	0	WS2_32.dll
000034F4	004034F4	0	MFC42.DLL
00003506	00403506	0	memmove
00003518	00403518	0	fprintf
00003522	00403522	0	sprintf
		0	perror


```

0000353E 0040353E 0 printf
00003546 00403546 0 MSVCRT.dll
00003552 00403552 0 __str
00003554 00403554 0 __dllonexit
00003562 00403562 0 _onexit
0000356C 0040356C 0 _exit
00003574 00403574 0 _XcptFilter
00003582 00403582 0 __p__initenv
00003592 00403592 0 __getmainargs
000035A2 004035A2 0 _initterm
000035AE 004035AE 0 __setusermatherr
000035C2 004035C2 0 _adjust_fdiv
000035D2 004035D2 0 __p__commode
000035E2 004035E2 0 __p__fmode
000035F0 004035F0 0 __set_app_type
00003602 00403602 0 _except_handler3
00003616 00403616 0 _controlfp
00003624 00403624 0 ??0Init@ios_base@std@@@QAE@XZ
00003644 00403644 0 ??1Init@ios_base@std@@@QAE@XZ
00003664 00403664 0 ??0_Winit@std@@@QAE@XZ
0000367C 0040367C 0 ??1_Winit@std@@@QAE@XZ
00003692 00403692 0 MSVCP60.dll
00004049 00404049 0 ERROR 3
00004055 00404055 0 ERROR 2
00004061 00404061 0 ERROR 1
0000406C 0040406C 0 impossible create raw ICMP socket
00004098 00404098 0 RAW ICMP SendTo:
000040AE 004040AE 0 ===== Icmp BackDoor
V0.1 =====
000040F4 004040F4 0 ===== Code by Spoof. Enjoy Yourself!
0000411E 0040411E 0 Your PassWord:
00004138 00404138 0 cmd.exe
00004142 00404142 0 Exit OK!
00004150 00404150 0 Local Partners Access
0000416A 0040416A 0 Error UnInstalling Service
0000418A 0040418A 0 Service UnInstalled Sucessfully
000041B2 004041B2 0 Error Installing Service
000041CE 004041CE 0 Service Installed Sucessfully
000041F5 004041F5 0 Create Service %s ok!
0000420D 0040420D 0 CreateService failed:%d
00004229 00404229 0 Service Stopped
0000423D 0040423D 0 Force Service Stopped Failed%d
00004260 00404260 0 The service is running or starting!
00004288 00404288 0 Query service status failed!
000042A8 004042A8 0 Open service failed!
000042C1 004042C1 0 Service %s Already exists

```

```

000042FC 004042FC 0 smsses.exe
00004309 00404309 0 Open Service Control Manager failed:%d
0000430C 0040430C 0 Local Printer Manager Service
00004338 00404338 0 Start service successfully!
00004358 00404358 0 Starting the service failed!
00004378 00404378 0 starting the service <%s>...
00004398 00404398 0 Successfully!
000043A8 004043A8 0 Failed!
000043B4 004043B4 0 Try to change the service's start type...
000043E0 004043E0 0 The service is disabled!
000043FC 004043FC 0 Query service config failed!
000062DB 004062DB 0 ??????
00005064 00405064 0 Hello from MFC!
000060F3 004060F3 0 \\winnt\system32\smsses.exe
00006181 00406181 0 \\winnt\system32\smsses.exe
000062B3 004062B3 0 \\199.107.97.191\C$
0000632F 0040632F 0 \\winnt\system32
000063A7 004063A7 0 \\winnt\system32\reg.exe
0000642F 0040642F 0 \\winnt\system32\reg.exe
000064B7 004064B7 0 \\winnt\system32\reg.exe
0000653F 0040653F 0 \\winnt\system32\reg.exe
000065BD 004065BD 0 \\winnt\system32\reg.exe
00006645 00406645 0 \\winnt\system32\reg.exe
000066CD 004066CD 0 \\winnt\system32\reg.exe
00006755 00406755 0 \\winnt\system32\reg.exe
000067DD 004067DD 0 \\winnt\system32\reg.exe
00005062 00405062 1 Hello from MFC!

```

There are several pieces of valuable information contained within the bintext output. First several dynamic link libraries are mentioned:

- Kernel32.dll – Responsible for memory management, Input/Output functions, and interrupts.
- Advapi32.dll – Advanced API services library support Application Programming Interfaces such as security and registry calls.
- Ws2_32.dll – Windows Sockets library, used by network applications to handle network connections.
- Mfc42.dll – Microsoft Foundation functions used by C++ applications.
- Msvcrt.dll – Microsoft C library.
- Msvcp60 – Microsoft C library.

Because the application is using C and C++ libraries it is probable that the application was written in C or C++. The application is probably a network client or server because the application is calling the Windows Socket dll. Several of

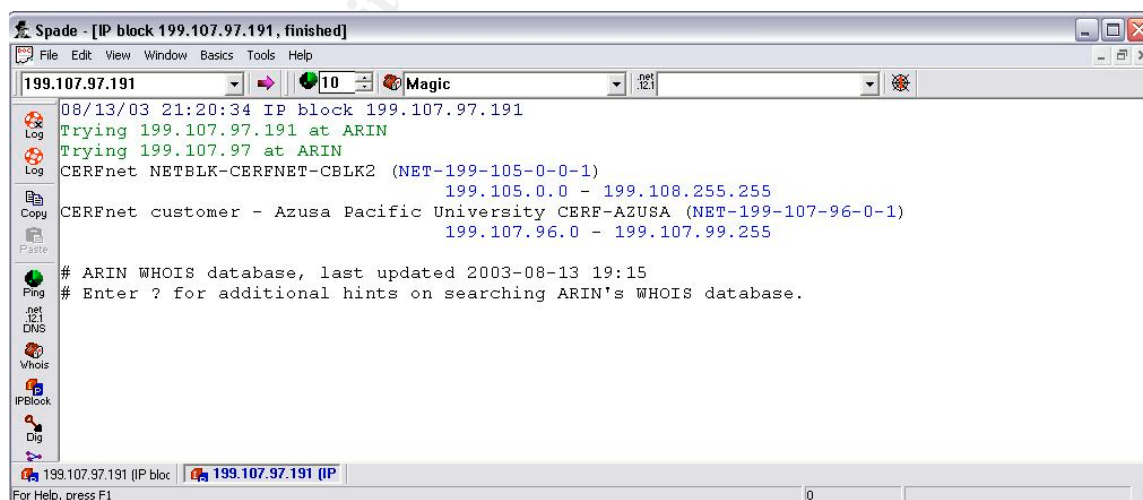
the entries above each Dynamic Link Library entry are possibly functions that the application is using from each Dynamic Link Library file.

The reg.exe and smsses.exe were also also mentioned in the bintext output. The reg.exe application is part of the Windows NT 4.0 Server Resource Kit, and is an application used to access the registry in batch scripts. The executable smsses.exe is not a standard Windows application (that I could find), there is an smss.exe application. The smss.exe application is the session manager subsystem, and is used to initialize system environment variables. The referenced smsses.exe file is possibly an application that is related to the target2.exe file.

Other lines provide clues about the possible use of the application. Several references to ICMP are seen, including references to creating an ICMP socket, and ICMP BackDoor V0.1, which is possibly the name of the executable. There is also an entry for cmd.exe, which is the Windows command line application. The cmd.exe is used to generate a shell for the user who accesses this application. Several references to installing, starting, and creating services are mentioned it is possible that the applications loads as a Windows service. Loading as a service would allow the application to restart at boot time, and potentially run with system level privileges.

Two lines identify possible sources of the code, the line "Code by SpooF. Enjoy Yourself!", shows that the author's name could perhaps be SpooF. The line [\\199.107.97.191\C\\$](http://199.107.97.191/C$), provides an address that some information is being directed to, through a windows share. This site could be referenced to update a list of compromised system or to perhaps be used to store more tools.

The application Sam Spade was used to perform a whois on the address:



Screenshot of Sam Spade application

The address is registered to Azusa Pacific University. The Sam Spade utility (<http://www.samspade.org/ssw>) is an application used to gather information about organizations by using their IP address or Name registration information. A whois query is a specific query to gather information about a company from a network address. These tools provided a way to associate the IP address in the strings output to an organization or in this case a University.

After reviewing the above output from the target2.exe, the file was analyzed using the Unix utility strings to verify the output. The strings utility is used to gather string information from a binary file. During the analysis an additional line was seen in the output:

```
impossibile creare raw ICMP socket
RAW ICMP SendTo:
===== Icmp BackDoor V0.1
=====
===== Code by Spoof. Enjoy Yourself!
Your PassWord:
loki
cmd.exe
Exit OK!
```

It is possible that *loki* is the password required to access the command shell or is possibly a reference to the LOKI application. Information on LOKI was found at <http://www.phrack.org/phrack/51/P51-06>, the author of LOKI states:

“LOKI2 is an information-tunneling program. It is proof of concept work intending to draw attention to the insecurity that is present in many network protocols. In this implementation, we tunnel simple shell commands inside of ICMP_ECHO/ICMP_ECHOREPLY and DNS name lookup query/reply traffic”

IDA Disassembler was used to extract the assembly language from the target2.exe file.

IDA Disassembler – an interactive program disassembler.

<http://www.datarescue.com/idabase/idadown.htm>

The output is very large and I’m not proficient at reading Assembly code.

The initial header information from the Assembly dump provided some information on the type of binary:

```
;
; File Name   :C:\Documents and Settings\dacruz\Desktop\target2.exe
; Format      : Portable executable for IBM PC (PE)
; Section 1. (virtual address 00001000)
```

```

; Virtual size           : 000018FC ( 6396.)
; Section size in file  : 00002000 ( 8192.)
; Offset to raw         data for section: 00001000
; Flags                 60000020: Text Executable Readable
; Alignment             : 16 bytes ?
; OS type               : MSWindows
; Application type: Executable32bit

```

Some information was gathered on the system calls that were made to the DLL files from the application.

```

; Section 2. (virtual address 00003000)
; Virtual size: 0000069E ( 1694.)
; Section size in file: 00001000 ( 4096.)
; Offset to rawdata for section: 00003000
; Flags40000040: Data Readable
; Alignment: 16 bytes ?
;
; Imports from ADVAPI32.dll
;
; Segment type:Pure data
_rdatasegmentpara public 'DATA' use32
assume cs:_rdata
;org 403000h
RegisterServiceCtrlHandlerA dd ?; DATA XREF: .text:00402263 r
StartServiceCtrlDispatcherA dd ?; DATA XREF: _main+122 r
SetServiceStatus dd ?; DATA XREF: .text:0040228E r
; .text:00402301 r
QueryServiceConfigA dd ?; DATA XREF: sub_0_402580+25 r
ChangeServiceConfigA dd?; DATA XREF: sub_0_402580+8A r
StartServiceAdd ?; DATA XREF: sub_0_402580+CE r
DeleteServiceAdd ?; DATA XREF: sub_0_4024D0+6E r
OpenSCManagerAdd ?; DATA XREF: sub_0_402320+E r
; sub_0_4024D0+D r
; Establish a connection to theservice
; control manager on the specified computer
; and opens thespecified database
CreateServiceAdd ?; DATA XREF: sub_0_402320+5F r
CloseServiceHandle dd ?; DATA XREF: sub_0_402320+190 r
; sub_0_4024D0+82 r ...
OpenServiceAdd ?; DATA XREF: sub_0_402320+AA r
; sub_0_4024D0+2C r
QueryServiceStatus dd ?; DATA XREF: sub_0_402320+D1 r

```

```

ControlServicedd ?; DATA XREF: sub_0_402320+112 r
; sub_0_402450+48 r
; sub_0_402580+B r
; Send a control code to a Win32 service
dd 0
;
; Imports from KERNEL32.dll
;
HeapAllocdd ?; DATA XREF: sub_0_4018C0+D6 r
LocalAllocdd ?; DATA XREF: sub_0_402580+B r
GetLastErrordd ?; DATA XREF: sub_0_402320+1D r
; sub_0_402320+72 r ...
WriteFiledd ?; DATA XREF: sub_0_401EE0+4A r
CreatePipeddd ?; DATA XREF: sub_0_401CD0+D r
CreateProcessAdd ?; DATA XREF: sub_0_401CD0+D0 r
CloseHandledd ?; DATA XREF: sub_0_401CD0+F1 r
PeekNamedPipeddd ?; DATA XREF: sub_0_401CD0+103 r
; sub_0_401EE0+86 r
ReadFiledd ?; DATA XREF: sub_0_401CD0+14B r
; sub_0_401EE0+AE r
TerminateProcess dd ?; DATA XREF: sub_0_401A00+DE r
; sub_0_401A00+129 r ...
GetProcessHeapdd ?; DATA XREF: sub_0_4018C0+CF r
Sleepdd ?; DATA XREF: sub_0_4010F0+279 r
; sub_0_401460+285 r ...
dd 0
;
; Imports from MFC42.DLL
;
; public: virtual __thiscall CWinApp::~CWinApp(void)
??1CWinApp@@@UAE@XZ dd ?; DATA XREF: .text:00402736 r
; public: __thiscall CWinApp::CWinApp(char const *)
??0CWinApp@@@QAE@PBD@Z dd ?; DATA XREF: CWinApp::CWinApp(char
const *) r
dd 0
;
; Imports from MSVCP60.dll
;
; public: __thiscall std::_Winit::_Winit(void)
??0_Winit@std@@@QAE@XZ dd ?; DATA XREF: sub_0_402700+5 r
; public: __thiscall std::ios_base::_Init::_Init(void)
??1Init@ios_base@std@@@QAE@XZ dd ?; DATA XREF: .text:004026E5 r
; public: __thiscall std::ios_base::_Init::_Init(void)
??0Init@ios_base@std@@@QAE@XZ dd ?; DATA XREF: sub_0_4026C0+5 r
; public: __thiscall std::_Winit::~~_Winit(void)
??1_Winit@std@@@QAE@XZ dd ?; DATA XREF: .text:00402725 r

```

```

;
; Imports from MSVCRT.dll
dd 0
;
strstrdd ?; DATA XREF: sub_0_401A00+1DD r
__dllonexitdd ?; DATA XREF: j__dllonexit r
timedd ?; DATA XREF: sub_0_401A00+84 r
; sub_0_401A00+155 r ...
printfdd ?; DATA XREF: _main+77 r
; sub_0_402320+29 r ...
__controlfpdd ?; DATA XREF: __controlfp r
__except_handler3 dd ?; DATA XREF: .text:004028F0 r
__set_app_typedd ?; DATA XREF: start+2C r
__p__fmodedd ?; DATA XREF: start+41 r
__p__commodedd ?; DATA XREF: start+4F r
memmovedd ?; DATA XREF: sub_0_401080+F r
exitdd ?; DATA XREF: sub_0_4010A0+38 r
; sub_0_401880+1C r ...
fprintfdd ?; DATA XREF: sub_0_4010A0+2D r
__jobdd ?; DATA XREF: sub_0_4010A0+1F r
sprintfdd ?; DATA XREF: sub_0_4010F0+2C6 r
; sub_0_401720+6E r ...
perrordd ?; DATA XREF: sub_0_4010F0+144 r
; sub_0_401460+148 r ...
__exitdd ?
__adjust_fdivdd ?; DATA XREF: start+5D r
__XcptFilterdd ?; DATA XREF: __XcptFilter r
__onexitdd ?; DATA XREF: __onexit+D r
__inittermdd ?; DATA XREF: __initterm r
__getmainargsdd ?; DATA XREF: start+B5 r
__p__initenvdd ?; DATA XREF: start+CA r
__setusermatherr dd ?; DATA XREF: start+7C r
dd 0
;
; Imports from WS2_32.dll
;
socketdd ?; DATA XREF: sub_0_4010A0+F r
WSACleanupdd ?; DATA XREF: sub_0_401880+27 r
htonsdd ?; DATA XREF: sub_0_4010F0+5D r
; sub_0_4010F0+71 r ...
closesocketdd ?; DATA XREF: sub_0_401A00+EB r
; sub_0_401A00+136 r ...
sendtoddd ?; DATA XREF: sub_0_4010F0+135 r
; sub_0_4010F0+260 r ...
WSASocketAdd ?; DATA XREF: sub_0_4018C0+1B r
gethostnamedd ?; DATA XREF: sub_0_4018C0+3C r

```

```

WSAIoctl; DATA XREF: sub_0_4018C0+C2 r
recvfrom; DATA XREF: sub_0_4018C0+DC r
gethostbyname; DATA XREF: sub_0_4018C0+47 r
WSAGetLastError; DATA XREF: sub_0_4018C0+E2 r
WSAStartup; DATA XREF: sub_0_401880+10 r
bind; DATA XREF: sub_0_4018C0+9F r
inet_addr; DATA XREF: sub_0_4018C0+76 r
dd 0
unk_0_403128db 0FFh;; DATA XREF: start+5 o

```

The output from the IDA Disassembler showed the various functions that were accessed in each Dynamic Link Library file. The most critical functions called belong to the WS2_32.dll, this is the Windows Socket Library.

One last tool to use to gather information from the binary is objdump. The objdump utility is a Unix utility to dump the object files from a binary file, the (-p) option will be used to extract the headers from the file.

```

target2.exe:  file format efi-app-ia32
Characteristics 0x10f

                        relocations stripped
                        executable
                        line numbers stripped
                        symbols stripped
                        32 bit words

Time/Date              Wed Nov 27 23:53:13 2002
ImageBase              0000000000400000
SectionAlignment       0000000000001000
FileAlignment          0000000000001000
MajorOSVersion         4
MinorOSVersion         0
MajorImageVersion      0
MinorImageVersion      0
MajorSubsystemVersion  4
MinorSubsystemVersion  0
Win32Version           00000000
SizeOfImage            00006000
SizeOfHeaders          00001000
Checksum              00000000
Subsystem              00000003
DllCharacteristics      00000000

```

(Windows CUI)

SizeOfStackCommit	0000000000001000	
SizeOfHeapReserve	0000000000100000	
SizeOfStackReserve	0000000000100000	
SizeOfHeapCommit	0000000000001000	
LoaderFlags		00000000
NumberOfRvaAndSizes	00000010	

The Data Directory

Entry 0 0000000000000000 00000000 Export Directory [.edata (or where ever we found it)]

Entry 1 0000000000003134 0000008c Import Directory [parts of .idata]

Entry 2 0000000000005000 000000a0 Resource Directory [.rsrc]

Entry 3 0000000000000000 00000000 Exception Directory [.pdata]

Entry 4 0000000000000000 00000000 Security Directory

Entry 5 0000000000000000 00000000 Base Relocation Directory [.reloc]

Entry 6 0000000000000000 00000000 Debug Directory

Entry 7 0000000000000000 00000000 Description Directory

Entry 8 0000000000000000 00000000 Special Directory

Entry 9 0000000000000000 00000000 Thread Storage Directory [.tls]

Entry a 0000000000000000 00000000 Load Configuration Directory

Entry b 0000000000000000 00000000 Bound Import Directory

Entry c 0000000000003000 00000128 Import Address Table Directory

Entry d 0000000000000000 00000000 Delay Import Directory

Entry e 0000000000000000 00000000 Reserved

Entry f 0000000000000000 00000000 Reserved

There is an import table in .rdata at 0x403134

The Import Tables (interpreted .rdata section contents)

vma:	Hint	Time	Forward	DLL	First
	Table	Stamp	Chain	Name	Thunk
00003134	000031f8	00000000	00000000	00003396	00003038

DLL Name: KERNEL32.dll
vma: Hint/Ord Member-Name Bound-To

32f0	409 HeapAlloc
3388	456 LocalAlloc
3378	282 GetLastError
336c	735 WriteFile
335e	67 CreatePipe
334c	68 CreateProcessA
333e	27 CloseHandle
332e	505 PeekNamedPipe

330e	670 TerminateProcess
32fc	320 GetProcessHeap
3322	536 ReadFile
32e8	662 Sleep

00003148 000031c0 00000000 00000000 000034b4 00003000

DLL Name: ADVAPI32.dll
vma: Hint/Ord Member-Name Bound-To

33d6	398 RegisterServiceCtrlHandlerA
33a4	435 StartServiceCtrlDispatcherA
33c2	430 SetServiceStatus
	349e 336 QueryServiceConfigA

3486	45 ChangeServiceConfigA
3476	434 StartServiceA
3466	120 DeleteService
3454	325 OpenSCManagerA
3442	76 CreateServiceA
33f4	52 CloseServiceHandle
3432	327 OpenServiceA
341c	341 QueryServiceStatus
340a	53 ControlService

0000315c 000032ac 00000000 00000000 000034dc 000030ec

DLL Name: WS2_32.dll
vma: Hint/Ord Member-Name Bound-To

80000017	23 <none>
80000074	116 <none>
80000009	9 <none>
80000003	3 <none>
80000014	20 <none>
34ce	61 WSASocketA
80000039	57 <none>
80000034	52 <none>
34c2	37 WSALocatl
80000011	17 <none>
8000006f	111 <none>
80000073	115 <none>
80000002	2 <none>

00003170 0000322c 00000000 00000000 000034e8 0000306c
8000000b 11 <none>

DLL Name: MFC42.DLL
vma: Hint/Ord Member-Name Bound-To

8000032f 815 <none>
80000231 561 <none>

00003184 0000324c 00000000 00000000 00003546 0000308c

DLL Name: MSVCRT.dll
vma: Hint/Ord Member-Name Bound-To

352a 709 strstr
3552 85 __dllonexit
3534 720 time
353c 670 printf
3614 183 _controlfp
3600 202 _except_handler3
35ee 129 __set_app_type
35e0 111 __p__fmode
35d0 106 __p__commode
34f2 664 memmove
34fc 585 exit
3504 600 fprintf
350e 275 _iob
3516 690 sprintf
3520 668 perror
356a 211 _exit
35c0 157 _adjust_fdiv
3572 72 _XcptFilter
3560 390 _onexit
35a0 271 _initterm
3590 88 __getmainargs
3580 100 __p__initenv
35ac 131 __setusermatherr

00003198 00003238 00000000 00000000 00003692 00003078

DLL Name: MSVCP60.dll
vma: Hint/Ord Member-Name Bound-To

```

3642          265  ??1Init@ios_base@std@@@QAE@XZ
3622          158  ??20Init@ios_base@std@@@QAE@XZ
3862          165  ??10_Winit@std@@@QAE@XZ
367a          269  ??1_Winit@std@@@QAE@XZ

000031ac    00000000 00000000 00000000 00000000 00000000

```

This output provided similar information as the IDA disassembler, however an extra date was gathered at the top of the output: Wed Nov 27 23:53:13 2002. This is possibly the compile time of the target2.exe application.

The next possible step could be to use a debugger program to walk through execution of the binary file. Possible tools could include the SoftIce Debugger or the Microsoft Windows Debugger. These tasks are beyond my abilities and should be performed by someone with a programming background.

From the data gathered in the last two sections, we can deduce the following conclusions:

- The application is likely an Windows ICMP backdoor application
- The applications was last used around Feb 20,2003, going by last modified and created dates.
- The application appears to be attempting to install itself as a service, possibly called Local Print Manager Service. The Local Print Manager Service name cannot be confirmed but I found no mention of that specific service in any Windows documentation.
- The application possibly provides a cmd.exe shell to the attacker.
- The application uses several Dynamic Link Libraries located on the system.
- The application or part of the application was written by someone named Spoof.
- There is an IP address referenced in the strings output 199.107.97.191.
- Smsses.exe and reg.exe are executables referenced in the code. Smsses.exe is not a common Windows application.

Forensic Details

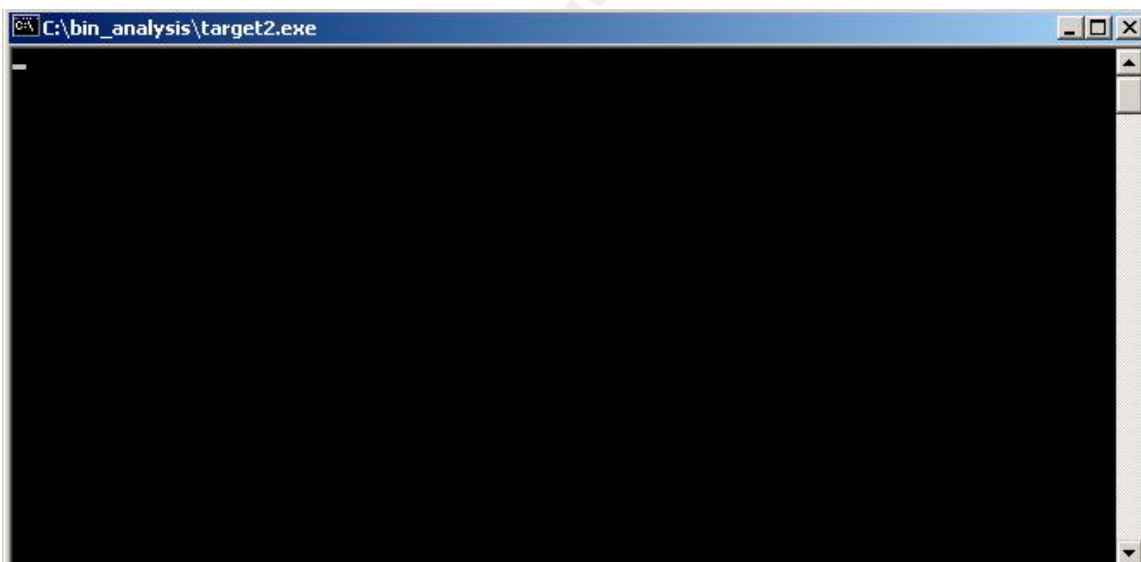
The program appears to be a Windows binary file. The binary will be analyzed using a Windows 2000 SP4 Server Computer and a Windows NT 4.0SP6 Computer. Several tools will be used to monitor the systems during the analysis.

- **Regmon** – a utility to monitor registry activity, regmon can be used to display what applications are accessing the registry and what keys are being accessed. <http://www.sysinternals.com/ntw2k/source/regmon.shtml>

- **Filemon** – a utility to display real time filesystem activity.
<http://www.sysinternals.com/ntw2k/source/filemon.shtml>
- **Netmon** - a utility to monitor network connections. http://www.futures-inc.com/Tools_and_origins/n.html
- **listdlls** – a command line utility to display what Dynamic Link Libraries are used by an application.
<http://www.sysinternals.com/ntw2k/freeware/listdlls.shtml>

The next step is to using the two Lab systems to attempt to run the application. Prior to running the application, regmon, filemon, listdlls, and netmon were installed in order to provide some level of monitoring for the application. The applications were each filtered to monitor only output from the target2.exe application. The binary file will be run from the Windows NT system first. As the file was launched an error was received. The error stated that the MSVCP60.DLL was not found.

I searched through the system and the Dynamic Link Library files were not installed on this system. This was a default install of Windows NT with Service Pack 6. It appears that this binary will not run on a default Windows NT system. Next the binary will be run from the Windows 2000 System. The program ran for about ten seconds, it displayed a DOS window and then exited.



Screenshot of the DOS Windows that the target2.exe application displayed.

Regmon, Filemon, Netmon, and listdlls were all running and capturing data. Here is the output from the listdlls application, this shows all the Dynamic Link Library files that the application uses while loading:

```
target2.exe pid: 1328
```

```
Command line: target2.exe
```

Base	Size	Version	Path
0x00400000	0x6000		C:\bin_analysis\target2.exe
0x77f80000	0x7b000	5.00.2195.6685	C:\WINNT\system32\ntdll.dll
0x7c4e0000	0xb9000	5.00.2195.6688	C:\WINNT\system32\KERNEL32.dll
0x7c2d0000	0x62000	5.00.2195.6710	C:\WINNT\system32\ADVAPI32.dll
0x77d30000	0x6e000	5.00.2195.6753	C:\WINNT\system32\RPCRT4.DLL
0x75030000	0x14000	5.00.2195.6601	C:\WINNT\system32\WS2_32.dll
0x78000000	0x45000	6.01.9844.0000	C:\WINNT\system32\MSVCRT.DLL
0x75020000	0x8000	5.00.2134.0001	C:\WINNT\system32\WS2HELP.DLL
0x6c370000	0xfb000	6.00.9586.0000	C:\WINNT\system32\MFC42.DLL
0x77f40000	0x3c000	5.00.2195.6660	C:\WINNT\system32\GDI32.dll
0x77e10000	0x65000	5.00.2195.6688	C:\WINNT\system32\USER32.DLL
0x780c0000	0x61000	6.00.8168.0000	C:\WINNT\system32\MSVCP60.dll

This output confirmed many of the dlls that were seen in the bintext output of the file. Some other dll files were referenced:

- Ntdll.dll – Dynamic Link Library that control Windows NT System functions
- GDI32.dll – Dynamic Link Library that contains Windows Graphic Device Interface to display 2D graphics in Windows.
- USER32.dll – Dynamic Link Library that provides functions for the Windows user interface.

A screenshot of the Task Manager was also taken while the application was running, the second from the last entry displays the target2.exe application.

© SANS Institute 2003. Author retains full rights.

Image Name	PID	CPU	CPU Time	Mem Usage	Peak Mem Usage	PF Delta	VM Size
System Idle Process	0	99	1:15:59	16 K	16 K	0	0 K
System	8	00	0:00:07	212 K	1,260 K	0	24 K
WINLOGON.EXE	148	00	0:00:00	1,100 K	9,284 K	0	6,052 K
SMSS.EXE	184	00	0:00:00	356 K	2,012 K	0	1,068 K
CSRSS.EXE	208	00	0:00:00	1,788 K	3,244 K	0	1,200 K
SERVICES.EXE	260	00	0:00:00	5,796 K	5,888 K	0	2,896 K
LSASS.EXE	272	00	0:00:00	5,548 K	5,588 K	0	2,792 K
DLLHOST.EXE	320	00	0:00:01	8,224 K	8,248 K	0	3,944 K
termsrv.exe	380	00	0:00:00	3,348 K	3,364 K	0	1,728 K
svchost.exe	496	00	0:00:00	3,924 K	3,948 K	0	1,444 K
spoolsv.exe	524	00	0:00:00	4,660 K	4,768 K	0	2,572 K
msdtc.exe	552	00	0:00:00	5,600 K	5,616 K	0	1,892 K
svchost.exe	688	00	0:00:00	8,232 K	8,320 K	0	4,460 K
inetd32.exe	708	00	0:00:00	2,144 K	2,144 K	0	648 K
jconfigNT.exe	720	00	0:00:00	1,616 K	1,620 K	0	376 K
hjavaw.exe	740	00	0:00:00	1,108 K	1,648 K	0	276 K
LLSSRV.EXE	752	00	0:00:00	2,216 K	2,216 K	0	720 K
nvsvc32.exe	788	00	0:00:00	1,444 K	2,992 K	0	368 K
regsvc.exe	820	00	0:00:00	1,096 K	1,104 K	0	276 K
scardsvr.exe	844	00	0:00:00	1,328 K	1,376 K	0	308 K
javaw.exe	852	00	0:00:00	10,104 K	11,092 K	0	9,936 K
mstask.exe	920	00	0:00:00	3,212 K	3,212 K	0	1,080 K
snmp.exe	964	00	0:00:00	4,020 K	4,024 K	0	1,604 K
WinMgmt.exe	1028	00	0:00:03	264 K	5,612 K	0	784 K
svchost.exe	1044	00	0:00:00	6,396 K	6,432 K	0	4,600 K
inetinfo.exe	1052	00	0:00:03	9,396 K	9,396 K	0	5,832 K
nutsv4.exe	1076	00	0:00:00	1,948 K	1,948 K	0	556 K
dfssvc.exe	1144	00	0:00:00	1,552 K	1,560 K	0	452 K
DLLHOST.EXE	1348	00	0:00:00	5,252 K	5,284 K	0	1,548 K
explorer.exe	1384	00	0:00:02	2,356 K	8,668 K	1	5,500 K
svchost.exe	1476	00	0:00:00	3,360 K	3,412 K	0	1,512 K
TASKMGR.EXE	1580	00	0:00:00	1,612 K	2,480 K	0	700 K
Netscp.exe	1616	00	0:00:00	16,800 K	17,004 K	0	9,196 K
winampa.exe	1620	00	0:00:00	1,136 K	1,140 K	0	280 K
WZQKPICK.EXE	1632	00	0:00:00	1,288 K	1,288 K	0	340 K
Netmon.exe	1640	00	0:00:00	3,248 K	3,248 K	0	1,192 K
target2.exe	1652	00	0:00:00	1,212 K	1,212 K	0	324 K
wuaudlt.exe	1664	00	0:00:00	4,916 K	4,936 K	0	2,520 K

Screenshot of taskmanager while target2.exe was running.

The screenshot from the Windows Taskmanager shows that target2.exe ran as PID 1652 and consumed 1212K of memory and 324K of Virtual Memory. The following screenshot shows the regmon application capturing data while the application was running.

Registry Monitor - Sysinternals: www.sysinternals.com						
File Edit Options Help						
#	Time	Process	Request	Path	Result	Other
18	690.28355061	target...	QueryValue	HKLM\System\CurrentControlSet\Contr...	SUCCE...	0x0
19	690.28357258	target...	CloseKey	HKLM\System\CurrentControlSet\Contr...	SUCCE...	Key: 0xE14...
20	690.28365186	target...	OpenKey	HKLM\SOFTWARE\Microsoft\Windo...	SUCCE...	Key: 0xE14...
21	690.28367274	target...	QueryValue	HKLM\SOFTWARE\Microsoft\Windo...	NOTFO...	
22	690.28369441	target...	CloseKey	HKLM\SOFTWARE\Microsoft\Windo...	SUCCE...	Key: 0xE14...
23	690.28374146	target...	OpenKey	HKLM	SUCCE...	Key: 0xE14...
24	690.28377698	target...	OpenKey	HKLM\Software\Microsoft\Windows N...	NOTFO...	
25	690.28480852	target...	OpenKey	HKLM\System\CurrentControlSet\Contr...	NOTFO...	
26	690.28565124	target...	OpenKey	HKLM\Software\Microsoft\Windows N...	SUCCE...	Key: 0xE2B...
27	690.28568135	target...	QueryValue	HKLM\Software\Microsoft\Windows N...	NOTFO...	
28	690.28570575	target...	CloseKey	HKLM\Software\Microsoft\Windows N...	SUCCE...	Key: 0xE2B...
29	690.28575466	target...	OpenKey	HKLM\Software\Microsoft\Windows N...	SUCCE...	Key: 0xE29...
30	690.28581191	target...	QueryValue	HKLM\Software\Microsoft\Windows N...	NOTFO...	
31	690.28583430	target...	CloseKey	HKLM\Software\Microsoft\Windows N...	SUCCE...	Key: 0xE29...
32	690.28587847	target...	OpenKey	HKLM\Software\Microsoft\Windows N...	SUCCE...	Key: 0xE2D...
33	690.28590938	target...	QueryValue	HKLM\Software\Microsoft\Windows N...	NOTFO...	
34	690.28593203	target...	CloseKey	HKLM\Software\Microsoft\Windows N...	SUCCE...	Key: 0xE2D...
35	690.28615226	target...	OpenKey	HKLM\System\CurrentControlSet\Contr...	NOTFO...	
36	690.28619357	target...	OpenKey	HKLM\Software\Microsoft\Windows N...	SUCCE...	Key: 0xE2D...
37	690.28621419	target...	QueryValue	HKLM\Software\Microsoft\Windows N...	SUCCE...	''''
38	690.28624565	target...	CloseKey	HKLM\Software\Microsoft\Windows N...	SUCCE...	Key: 0xE2D...
39	690.28696462	target...	OpenKey	HKCU	SUCCE...	Key: 0xE2D...
40	690.28699384	target...	OpenKey	HKLM\System\CurrentControlSet\Contr...	NOTFO...	
41	690.28702979	target...	OpenKey	HKCU\Control Panel\Desktop	SUCCE...	Key: 0xE28...
42	690.28706974	target...	QueryValue	HKCU\Control Panel\Desktop\MultiUI...	NOTFO...	
43	690.28709059	target...	CloseKey	HKCU\Control Panel\Desktop	SUCCE...	Key: 0xE28...

Regmon utility screenshot, while target2.exe was running.

The following output shows the log file from regmon utility:

```

1 690.23772735 explorer.exe:1464 OpenKey
   HKLM\System\CurrentControlSet\Control\Session Manager\AppCompatibility\target2.exe
   NOTFOUND
2 690.23816090 explorer.exe:1464 OpenKey
   HKLM\Software\Microsoft\Windows\CurrentVersion\App Paths\target2.exe NOTFOUND
3 690.23820200 explorer.exe:1464 OpenKey
   HKLM\Software\Microsoft\Windows\CurrentVersion\App Paths\target2.exe NOTFOUND
4 690.23877938 explorer.exe:1464 OpenKey HKLM\Software\Microsoft\Windows
   NT\CurrentVersion\Image File Execution Options\target2.exe NOTFOUND
5 690.23943747 target2.exe:1520 OpenKey HKLM\Software\Microsoft\Windows
   NT\CurrentVersion\Image File Execution Options\target2.exe NOTFOUND
6 690.23947312 target2.exe:1520 OpenKey HKLM\Software\Microsoft\Windows
   NT\CurrentVersion\Image File Execution Options\target2.exe NOTFOUND
7 690.23982387 target2.exe:1520 OpenKey HKLM\Software\Microsoft\Windows
   NT\CurrentVersion\Image File Execution Options\target2.exe NOTFOUND
8 690.24261245 target2.exe:1520 OpenKey HKLM\Software\Microsoft\Windows
   NT\CurrentVersion\Image File Execution Options\target2.exe NOTFOUND
9 690.27533382 CSRSS.EXE:208 OpenKey
   HKCU\Console\C:_bin_analysis_target2.exe NOTFOUND
10 690.27535118 CSRSS.EXE:208 OpenKey
   HKCU\Console\C:_bin_analysis_target2.exe NOTFOUND
11 690.28319314 target2.exe:1520 OpenKey

```



```

        HKLM\System\CurrentControlSet\Control\Terminal Server    SUCCESSKey:
0xE14BCF20
12  690.28322276    target2.exe:1520 QueryValue
        HKLM\System\CurrentControlSet\Control\Terminal Server\TSAppCompatSUCCESS0x0
13  690.28325359    target2.exe:1520 CloseKey
        HKLM\System\CurrentControlSet\Control\Terminal Server    SUCCESSKey:
0xE14BCF20
14  690.28336748    target2.exe:1520 OpenKey
        HKLM\System\CurrentControlSet\Control\Session Manager    SUCCESSKey:
0xE14BCF20
15  690.28339750    target2.exe:1520 QueryValue
        HKLM\System\CurrentControlSet\Control\Session Manager\SafeDllSearchMode
        NOTFOUND
16  690.28342096    target2.exe:1520 CloseKey
        HKLM\System\CurrentControlSet\Control\Session Manager    SUCCESSKey:
0xE14BCF20
17  690.28353347    target2.exe:1520 OpenKey
        HKLM\System\CurrentControlSet\Control\Terminal Server    SUCCESSKey:
0xE14BCF20
18  690.28355061    target2.exe:1520 QueryValue
        HKLM\System\CurrentControlSet\Control\Terminal Server\TSAppCompatSUCCESS0x0
19  690.28357258    target2.exe:1520 CloseKey
        HKLM\System\CurrentControlSet\Control\Terminal Server    SUCCESSKey:
0xE14BCF20
20  690.28365186    target2.exe:1520 OpenKey HKLM\SOFTWARE\Microsoft\Windows
NT\CurrentVersion\Winlogon SUCCESS Key: 0xE14BCF20
21  690.28367274    target2.exe:1520 QueryValue HKLM\SOFTWARE\Microsoft\Windows
NT\CurrentVersion\Winlogon\LeakTrack NOTFOUND
22  690.28369441    target2.exe:1520 CloseKey HKLM\SOFTWARE\Microsoft\Windows
NT\CurrentVersion\Winlogon SUCCESS Key: 0xE14BCF20
23  690.28374146    target2.exe:1520 OpenKey HKLMSUCCESS Key: 0xE14BCF20
24  690.28377698    target2.exe:1520 OpenKey HKLM\Software\Microsoft\Windows
NT\CurrentVersion\Diagnostics NOTFOUND
25  690.28480852    target2.exe:1520 OpenKey
        HKLM\System\CurrentControlSet\Control\Error Message Instrument\ NOTFOUND
26  690.28565124    target2.exe:1520 OpenKey HKLM\Software\Microsoft\Windows
NT\CurrentVersion\Compatibility32 SUCCESS Key: 0xE2B0D380
27  690.28568135    target2.exe:1520 QueryValue HKLM\Software\Microsoft\Windows
NT\CurrentVersion\Compatibility32\target2 NOTFOUND
28  690.28570575    target2.exe:1520 CloseKey HKLM\Software\Microsoft\Windows
NT\CurrentVersion\Compatibility32 SUCCESS Key: 0xE2B0D380
29  690.28575466    target2.exe:1520 OpenKey HKLM\Software\Microsoft\Windows
NT\CurrentVersion\Compatibility2 SUCCESS Key: 0xE29E3C00
30  690.28581191    target2.exe:1520 QueryValue HKLM\Software\Microsoft\Windows
NT\CurrentVersion\Compatibility2\target20.0 NOTFOUND
31  690.28583430    target2.exe:1520 CloseKey HKLM\Software\Microsoft\Windows
NT\CurrentVersion\Compatibility2 SUCCESS Key: 0xE29E3C00
32  690.28587847    target2.exe:1520 OpenKey HKLM\Software\Microsoft\Windows
NT\CurrentVersion\IME Compatibility SUCCESS Key: 0xE2DA5240
33  690.28590938    target2.exe:1520 QueryValue HKLM\Software\Microsoft\Windows
NT\CurrentVersion\IME Compatibility\target2 NOTFOUND
34  690.28593203    target2.exe:1520 CloseKey HKLM\Software\Microsoft\Windows
NT\CurrentVersion\IME Compatibility SUCCESS Key: 0xE2DA5240
35  690.28615226    target2.exe:1520 OpenKey
        HKLM\System\CurrentControlSet\Control\Session Manager\AppCompatibility\target2.exe
        NOTFOUND

```

36	690.28619357	target2.exe:1520	OpenKey	HKLM\Software\Microsoft\Windows NT\CurrentVersion\Windows	SUCCESS	Key: 0xE2DA5240
37	690.28621419	target2.exe:1520	QueryValue	HKLM\Software\Microsoft\Windows NT\CurrentVersion\Windows\Applnit_DLLs	SUCCESS	""
38	690.28624565	target2.exe:1520	CloseKey	HKLM\Software\Microsoft\Windows NT\CurrentVersion\Windows	SUCCESS	Key: 0xE2DA5240
39	690.28696462	target2.exe:1520	OpenKey	HKCU	SUCCESS	Key: 0xE2DA5240
40	690.28699384	target2.exe:1520	OpenKey	HKLM\System\CurrentControlSet\Control\Nls\MUILanguages	NOTFOUND	
41	690.28702979	target2.exe:1520	OpenKey	HKCU\Control Panel\Desktop	SUCCESS	Key: 0xE2889CE0
42	690.28706974	target2.exe:1520	QueryValue	HKCU\Control Panel\Desktop\MultiUILanguageId	NOTFOUND	
43	690.28709059	target2.exe:1520	CloseKey	HKCU\Control Panel\Desktop	SUCCESS	Key: 0xE2889CE0
44	690.28711285	target2.exe:1520	CloseKey	HKCU	SUCCESS	Key: 0xE2DA5240
45	690.28798225	target2.exe:1520	OpenKey	HKLM\System\CurrentControlSet\Control\ServiceCurrent	SUCCESS	Key: 0xE2DA5240
46	690.28800736	target2.exe:1520	QueryValue	HKLM\System\CurrentControlSet\Control\ServiceCurrent\Default	SUCCESS	0x14
47	690.28803600	target2.exe:1520	CloseKey	HKLM\System\CurrentControlSet\Control\ServiceCurrent	SUCCESS	Key: 0xE2DA5240
48	705.27777501	target2.exe:1520	CloseKey	HKLM	SUCCESS	Key: 0xE14BCF20

Several key from the HKEY_LOCAL_MACHINE and HKEY_CURRENT_USER Hives were queried or opened. Many keys were not found.

- Lines 1 – 8, include the target2.exe attempting to open keys HKLM\System\CurrentControlSet\Control\Session Manager\AppCompatibility\target2.exe and HKLM\Software\Microsoft\Windows NT\CurrentVersion\Image File Execution Options\target2.exe. Neither of these keys is located on the system.
- Lines 9 and 10, involve the CSRSS.EXE application attempting to Open the HKCU\Console\C:_bin_analysis_target2.exe. The CSRSS.exe application is the Windows Client Server Runtime SubSystem and is used to perform windows and graphics functions for Windows subsystems, creating and deleting threads, and running the 16 bit virtual DOS environment. This key is again not found on the analysis system.
- Lines 11 – 19, involve target2.exe opening, successfully, several keys related to windows Terminal Server and Session Manager. Line 15 the HKLM\System\CurrentControlSet\Control\Session Manager\SafeDllSearchMode key was not found.
- Lines 20 – 22, involve the target2.exe application opening and closing keys in the HKLM\SOFTWARE\Microsoft\Windows NT\CurrentVersion\Winlogon.
- Lines 24 and 25, involve the target2.exe application looking for registry keys under HKLM, the keys are not found on the system.

- Lines 26 – 31 involve the target2.exe application opening and closing keys within the HKLM\Software\Microsoft\Windows NT\CurrentVersion\Compatibility32 and HKLM\Software\Microsoft\Windows NT\CurrentVersion\Compatibility2 registry directories.
- Lines 32 – 34, involve the application target2.exe opening, querying, and closing HKLM\Software\Microsoft\Windows NT\CurrentVersion\IME Compatibility. The value being queried is target2 and is not found at that registry location.
- Line 35 and 40, involve the application target2.exe looking for registry keys HKLM\System\CurrentControlSet\Control\Session Manager\AppCompatibility\target2.exe and HKLM\System\CurrentControlSet\Control\Nls\MUILanguages, neither key is found.
- Lines 36 – 38, involve the application target2.exe querying the Applnit_DLLS key under HKLM\Software\Microsoft\Windows NT\CurrentVersion\Windows.
- Lines 41 – 44, involve the application target2.exe querying the Multilanguaged value from the HKCU\Control Panel\Desktop registry location.
- Lines 45 – 48, involve the application target2.exe opening, querying, and closing keys within HKLM\System\CurrentControlSet\Control\ServiceCurrent.

The following screenshot shows the filemon application:

#	Time	Process	Request	Path	Result	Other
58	4:23:28 PM	target2.exe	QUERY INFORMATION	C:\bin_analysis\MSVCP60.dll	FILE NOT F...	Attributes: Error
59	4:23:28 PM	target2.exe	QUERY INFORMATION	C:\bin_analysis\MSVCP60.dll	FILE NOT F...	Attributes: Error
60	4:23:28 PM	target2.exe	QUERY INFORMATION	C:\WINNT\system32\MSVCP60.dll	SUCCESS	Attributes: A
61	4:23:28 PM	target2.exe	OPEN	C:\WINNT\system32\MSVCP60.dll	SUCCESS	Options: Open Access: Exe...
62	4:23:28 PM	target2.exe	CLOSE	C:\WINNT\system32\MSVCP60.dll	SUCCESS	
63	4:23:28 PM	target2.exe	QUERY INFORMATION	C:\bin_analysis\target2.exe.Local	FILE NOT F...	Attributes: Error
64	4:23:28 PM	CSRSS.E...	QUERY INFORMATION	C:\bin_analysis\target2.exe	SUCCESS	Attributes: RA
65	4:23:28 PM	CSRSS.E...	QUERY INFORMATION	C:\bin_analysis\target2.exe	SUCCESS	Attributes: RA
66	4:23:28 PM	CSRSS.E...	OPEN	C:\bin_analysis\target2.exe	SUCCESS	Options: Open Access: All
67	4:23:28 PM	CSRSS.E...	QUERY INFORMATION	C:\bin_analysis\target2.exe	SUCCESS	Attributes: RA
68	4:23:28 PM	CSRSS.E...	SET INFORMATION	C:\bin_analysis\target2.exe	SUCCESS	FileBasicInformation
69	4:23:28 PM	CSRSS.E...	READ	C:\bin_analysis\target2.exe	SUCCESS	Offset: 0 Length: 12
70	4:23:28 PM	CSRSS.E...	QUERY INFORMATION	C:\bin_analysis\target2.exe	SUCCESS	Length: 26793
71	4:23:28 PM	CSRSS.E...	QUERY INFORMATION	C:\bin_analysis\target2.exe	SUCCESS	Length: 26793
72	4:23:28 PM	CSRSS.E...	CLOSE	C:\bin_analysis\target2.exe	SUCCESS	
73	4:23:28 PM	target2.exe	QUERY INFORMATION	C:\WINNT\system32\MFC42LOC.DLL	FILE NOT F...	Attributes: Error
74	4:23:28 PM	target2.exe	QUERY INFORMATION	C:\WINNT\system32\MFC42LOC.DLL	FILE NOT F...	Attributes: Error
75	4:23:28 PM	Regmon.e...	DIRECTORY	C:\bin_analysis\	SUCCESS	FileBothDirectoryInformation:...
76	4:23:28 PM	Regmon.e...	QUERY INFORMATION	C:\bin_analysis\target2.exe	SUCCESS	Attributes: RA
77	4:23:28 PM	Regmon.e...	QUERY INFORMATION	C:\bin_analysis\target2.exe	SUCCESS	Attributes: RA
78	4:23:28 PM	Regmon.e...	OPEN	C:\bin_analysis\target2.exe	SUCCESS	Options: Open Access: All
79	4:23:28 PM	Regmon.e...	QUERY INFORMATION	C:\bin_analysis\target2.exe	SUCCESS	Attributes: RA
80	4:23:28 PM	Regmon.e...	SET INFORMATION	C:\bin_analysis\target2.exe	SUCCESS	FileBasicInformation
81	4:23:28 PM	Regmon.e...	READ	C:\bin_analysis\target2.exe	SUCCESS	Offset: 0 Length: 12
82	4:23:28 PM	Regmon.e...	QUERY INFORMATION	C:\bin_analysis\target2.exe	SUCCESS	Length: 26793
83	4:23:28 PM	Regmon.e...	QUERY INFORMATION	C:\bin_analysis\target2.exe	SUCCESS	Length: 26793
84	4:23:28 PM	Regmon.e...	CLOSE	C:\bin_analysis\target2.exe	SUCCESS	
85	4:23:28 PM	Regmon.e...	DIRECTORY	C:\bin_analysis\	SUCCESS	FileBothDirectoryInformation:...

Screenshot of filemon, while target2.exe is running.

The following output is from the log file from filemon:

```
14:23:28 PM explorer.exe:1464 QUERY INFORMATION C:\bin_analysis\target2.exe
SUCCESS Attributes: RA
2 4:23:28 PM explorer.exe:1464 OPEN C:\bin_analysis\target2.exe
SUCCESS Options: Open Access: All
3 4:23:28 PM explorer.exe:1464 READ C:\bin_analysis\target2.exe
SUCCESS Offset: 0 Length: 24
4 4:23:28 PM explorer.exe:1464 OPEN
C:\bin_analysis\target2.exe:_Raec25ph4sudbf0hAaq5ehw3Nf:$DATA FILE NOT
FOUND Options: Open Access: All
5 4:23:28 PM explorer.exe:1464 CLOSE C:\bin_analysis\target2.exe
SUCCESS
6 4:23:28 PM explorer.exe:1464 OPEN C:\bin_analysis\target2.exe
SUCCESS Options: Open Access: All
7 4:23:28 PM explorer.exe:1464 READ C:\bin_analysis\target2.exe
SUCCESS Offset: 0 Length: 24
8 4:23:28 PM explorer.exe:1464 OPEN
C:\bin_analysis\target2.exe:_Raec25ph4sudbf0hAaq5ehw3Nf:$DATA FILE NOT
FOUND Options: Open Access: All
9 4:23:28 PM explorer.exe:1464 OPEN
C:\bin_analysis\target2.exe\{4c8cc155-6c1e-11d1-8e41-00c04fb9386d}:$DATA
FILE NOT FOUND Options: Open Access: All
10 4:23:28 PM explorer.exe:1464 OPEN
C:\bin_analysis\target2.exe\_SummaryInformation:$DATA FILE NOT FOUND
Options: Open Access: All
11 4:23:28 PM explorer.exe:1464 OPEN
C:\bin_analysis\target2.exe\Docf__SummaryInformation:$DATA FILE NOT
FOUND Options: Open Access: All
12 4:23:28 PM explorer.exe:1464 OPEN
C:\bin_analysis\target2.exe\_SummaryInformation:$DATA FILE NOT FOUND
Options: Open Access: All
13 4:23:28 PM explorer.exe:1464 OPEN
C:\bin_analysis\target2.exe\Docf__SummaryInformation:$DATA FILE NOT
FOUND Options: Open Access: All
14 4:23:28 PM explorer.exe:1464 OPEN
C:\bin_analysis\target2.exe\_SummaryInformation:$DATA FILE NOT FOUND
Options: Open Access: All
15 4:23:28 PM explorer.exe:1464 OPEN
C:\bin_analysis\target2.exe\Docf__SummaryInformation:$DATA FILE NOT
FOUND Options: Open Access: All
16 4:23:28 PM explorer.exe:1464 OPEN
C:\bin_analysis\target2.exe\_DocumentSummaryInformation:$DATA FILE NOT
FOUND Options: Open Access: All
17 4:23:28 PM explorer.exe:1464 OPEN
C:\bin_analysis\target2.exe\Docf__DocumentSummaryInformation:$DATA FILE
```

NOT FOUND Options: Open Access: All

18 4:23:28 PM explorer.exe:1464 OPEN
C:\bin_analysis\target2.exe\:_SummaryInformation:\$DATA FILE NOT FOUND
Options: Open Access: All

19 4:23:28 PM explorer.exe:1464 OPEN
C:\bin_analysis\target2.exe\:_Docf__SummaryInformation:\$DATAFILE NOT
FOUND Options: Open Access: All

20 4:23:28 PM explorer.exe:1464 OPEN
C:\bin_analysis\target2.exe\:_SummaryInformation:\$DATA FILE NOT FOUND
Options: Open Access: All

21 4:23:28 PM explorer.exe:1464 OPEN
C:\bin_analysis\target2.exe\:_Docf__SummaryInformation:\$DATAFILE NOT
FOUND Options: Open Access: All

22 4:23:28 PM explorer.exe:1464 OPEN
C:\bin_analysis\target2.exe\:_SebiesnrMkudrfcolaamtykdDa:\$DATAFILE NOT
FOUND Options: Open Access: All

23 4:23:28 PM explorer.exe:1464 OPEN
C:\bin_analysis\target2.exe\:_Docf__SebiesnrMkudrfcolaamtykdDa:\$DATAFILE
NOT FOUND Options: Open Access: All

24 4:23:28 PM explorer.exe:1464 CLOSE C:\bin_analysis\target2.exe
SUCCESS

25 4:23:28 PM explorer.exe:1464 DIRECTORY C:\bin_analysis\
SUCCESS FileBothDirectoryInformation: target2.exe

26 4:23:28 PM explorer.exe:1464 QUERY INFORMATION
C:\bin_analysis\target2.exe SUCCESS Attributes: RA

27 4:23:28 PM explorer.exe:1464 DIRECTORY C:\bin_analysis\
SUCCESS FileBothDirectoryInformation: target2.exe

28 4:23:28 PM explorer.exe:1464 QUERY INFORMATION
C:\bin_analysis\target2.exe\desktop.ini PATH NOT FOUND Attributes: Error

29 4:23:28 PM explorer.exe:1464 OPEN C:\bin_analysis\target2.exe
SUCCESS Options: Open Access: All

30 4:23:28 PM explorer.exe:1464 QUERY INFORMATION
C:\bin_analysis\target2.exe SUCCESS Attributes: RA

31 4:23:28 PM explorer.exe:1464 SET INFORMATION
C:\bin_analysis\target2.exe SUCCESS FileBasicInformation

32 4:23:28 PM explorer.exe:1464 READ C:\bin_analysis\target2.exe
SUCCESS Offset: 0 Length: 64

33 4:23:28 PM explorer.exe:1464 READ C:\bin_analysis\target2.exe
SUCCESS Offset: 216 Length: 64

34 4:23:28 PM explorer.exe:1464 READ C:\bin_analysis\target2.exe
SUCCESS Offset: 288 Length: 4

35 4:23:28 PM explorer.exe:1464 READ C:\bin_analysis\target2.exe
SUCCESS Offset: 308 Length: 4

36 4:23:28 PM explorer.exe:1464 CLOSE C:\bin_analysis\target2.exe
SUCCESS

37 4:23:28 PM explorer.exe:1464 QUERY INFORMATION
C:\bin_analysis\target2.exe SUCCESS Attributes: RA

38 4:23:28 PM explorer.exe:1464 QUERY INFORMATION
C:\bin_analysis\target2.exe SUCCESS Attributes: RA

39 4:23:28 PM explorer.exe:1464 OPEN C:\bin_analysis\target2.exe
SUCCESS Options: Open Access: Execute

```

41 4:23:28 PM explorer.exe:1464 CLOSE C:\bin_analysis\target2.exe
40 4:23:28 PM explorer.exe:1464 QUERY INFORMATION
C:\bin_analysis\target2.exe SUCCESS Length: 26793
42 4:23:28 PM target2.exe:1520 OPEN C:\bin_analysis SUCCESS Options:
Open Directory Access: Traverse
43 4:23:28 PM target2.exe:1520 QUERY INFORMATION
C:\bin_analysis\WS2_32.dll FILE NOT FOUND Attributes: Error
44 4:23:28 PM target2.exe:1520 QUERY INFORMATION
C:\bin_analysis\WS2_32.dll FILE NOT FOUND Attributes: Error
45 4:23:28 PM target2.exe:1520 QUERY INFORMATION
C:\WINNT\system32\WS2_32.dll SUCCESS Attributes: A
46 4:23:28 PM target2.exe:1520 OPEN C:\WINNT\system32\WS2_32.dll
SUCCESS Options: Open Access: Execute
47 4:23:28 PM target2.exe:1520 CLOSE C:\WINNT\system32\WS2_32.dll
SUCCESS
48 4:23:28 PM target2.exe:1520 QUERY INFORMATION
C:\bin_analysis\WS2HELP.DLL FILE NOT FOUND Attributes: Error
49 4:23:28 PM target2.exe:1520 QUERY INFORMATION
C:\bin_analysis\WS2HELP.DLL FILE NOT FOUND Attributes: Error
50 4:23:28 PM target2.exe:1520 QUERY INFORMATION
C:\WINNT\system32\WS2HELP.DLL SUCCESS Attributes: A
51 4:23:28 PM target2.exe:1520 OPEN C:\WINNT\system32\WS2HELP.DLL
SUCCESS Options: Open Access: Execute
52 4:23:28 PM target2.exe:1520 CLOSE C:\WINNT\system32\WS2HELP.DLL
SUCCESS
53 4:23:28 PM target2.exe:1520 QUERY INFORMATION
C:\bin_analysis\MFC42.DLL FILE NOT FOUND Attributes: Error
54 4:23:28 PM target2.exe:1520 QUERY INFORMATION
C:\bin_analysis\MFC42.DLL FILE NOT FOUND Attributes: Error
55 4:23:28 PM target2.exe:1520 QUERY INFORMATION
C:\WINNT\system32\MFC42.DLL SUCCESS Attributes: A
56 4:23:28 PM target2.exe:1520 OPEN C:\WINNT\system32\MFC42.DLL
SUCCESS Options: Open Access: Execute
57 4:23:28 PM target2.exe:1520 CLOSE C:\WINNT\system32\MFC42.DLL
SUCCESS
58 4:23:28 PM target2.exe:1520 QUERY INFORMATION
C:\bin_analysis\MSVCP60.dll FILE NOT FOUND Attributes: Error
59 4:23:28 PM target2.exe:1520 QUERY INFORMATION
C:\bin_analysis\MSVCP60.dll FILE NOT FOUND Attributes: Error
60 4:23:28 PM target2.exe:1520 QUERY INFORMATION
C:\WINNT\system32\MSVCP60.dll SUCCESS Attributes: A
61 4:23:28 PM target2.exe:1520 OPEN C:\WINNT\system32\MSVCP60.dll
SUCCESS Options: Open Access: Execute
62 4:23:28 PM target2.exe:1520 CLOSE C:\WINNT\system32\MSVCP60.dll
SUCCESS
63 4:23:28 PM target2.exe:1520 QUERY INFORMATION
C:\bin_analysis\target2.exe.Local FILE NOT FOUND Attributes: Error
64 4:23:28 PM CSRSS.EXE:208 QUERY INFORMATION
C:\bin_analysis\target2.exe SUCCESS Attributes: RA
65 4:23:28 PM CSRSS.EXE:208 QUERY INFORMATION

```

```

66  C:\bin_analysis\target2.exe SUCCESS Attributes: RA
    4:23:28 PM CSRSS.EXE:208 OPEN C:\bin_analysis\target2.exe
    SUCCESS Options: Open Access: All
67  4:23:28 PM CSRSS.EXE:208 QUERY INFORMATION
    C:\bin_analysis\target2.exe SUCCESS Attributes: RA
68  4:23:28 PM CSRSS.EXE:208 SET INFORMATION
    C:\bin_analysis\target2.exe SUCCESS FileBasicInformation
69  4:23:28 PM CSRSS.EXE:208 READ C:\bin_analysis\target2.exe
    SUCCESS Offset: 0 Length: 12
70  4:23:28 PM CSRSS.EXE:208 QUERY INFORMATION
    C:\bin_analysis\target2.exe SUCCESS Length: 26793
71  4:23:28 PM CSRSS.EXE:208 QUERY INFORMATION
    C:\bin_analysis\target2.exe SUCCESS Length: 26793
72  4:23:28 PM CSRSS.EXE:208 CLOSE C:\bin_analysis\target2.exe
    SUCCESS
73  4:23:28 PM target2.exe:1520 QUERY INFORMATION
    C:\WINNT\system32\MFC42LOC.DLL FILE NOT FOUND Attributes: Error
74  4:23:28 PM target2.exe:1520 QUERY INFORMATION
    C:\WINNT\system32\MFC42LOC.DLL FILE NOT FOUND Attributes: Error
75  4:23:28 PM Regmon.exe:1488 DIRECTORY C:\bin_analysis\
    SUCCESS FileBothDirectoryInformation: target2.exe
76  4:23:28 PM Regmon.exe:1488 QUERY INFORMATION
    C:\bin_analysis\target2.exe SUCCESS Attributes: RA
77  4:23:28 PM Regmon.exe:1488 QUERY INFORMATION
    C:\bin_analysis\target2.exe SUCCESS Attributes: RA
78  4:23:28 PM Regmon.exe:1488 OPEN C:\bin_analysis\target2.exe
    SUCCESS Options: Open Access: All
79  4:23:28 PM Regmon.exe:1488 QUERY INFORMATION
    C:\bin_analysis\target2.exe SUCCESS Attributes: RA
80  4:23:28 PM Regmon.exe:1488 SET INFORMATION
    C:\bin_analysis\target2.exe SUCCESS FileBasicInformation
81  4:23:28 PM Regmon.exe:1488 READ C:\bin_analysis\target2.exe
    SUCCESS Offset: 0 Length: 12
82  4:23:28 PM Regmon.exe:1488 QUERY INFORMATION
    C:\bin_analysis\target2.exe SUCCESS Length: 26793
83  4:23:28 PM Regmon.exe:1488 QUERY INFORMATION
    C:\bin_analysis\target2.exe SUCCESS Length: 26793
84  4:23:28 PM Regmon.exe:1488 CLOSE C:\bin_analysis\target2.exe
    SUCCESS
85  4:23:28 PM Regmon.exe:1488 DIRECTORY C:\bin_analysis\
    SUCCESS FileBothDirectoryInformation: target2.exe
86  4:23:43 PM target2.exe:1520 CLOSE C:\bin_analysis SUCCESS

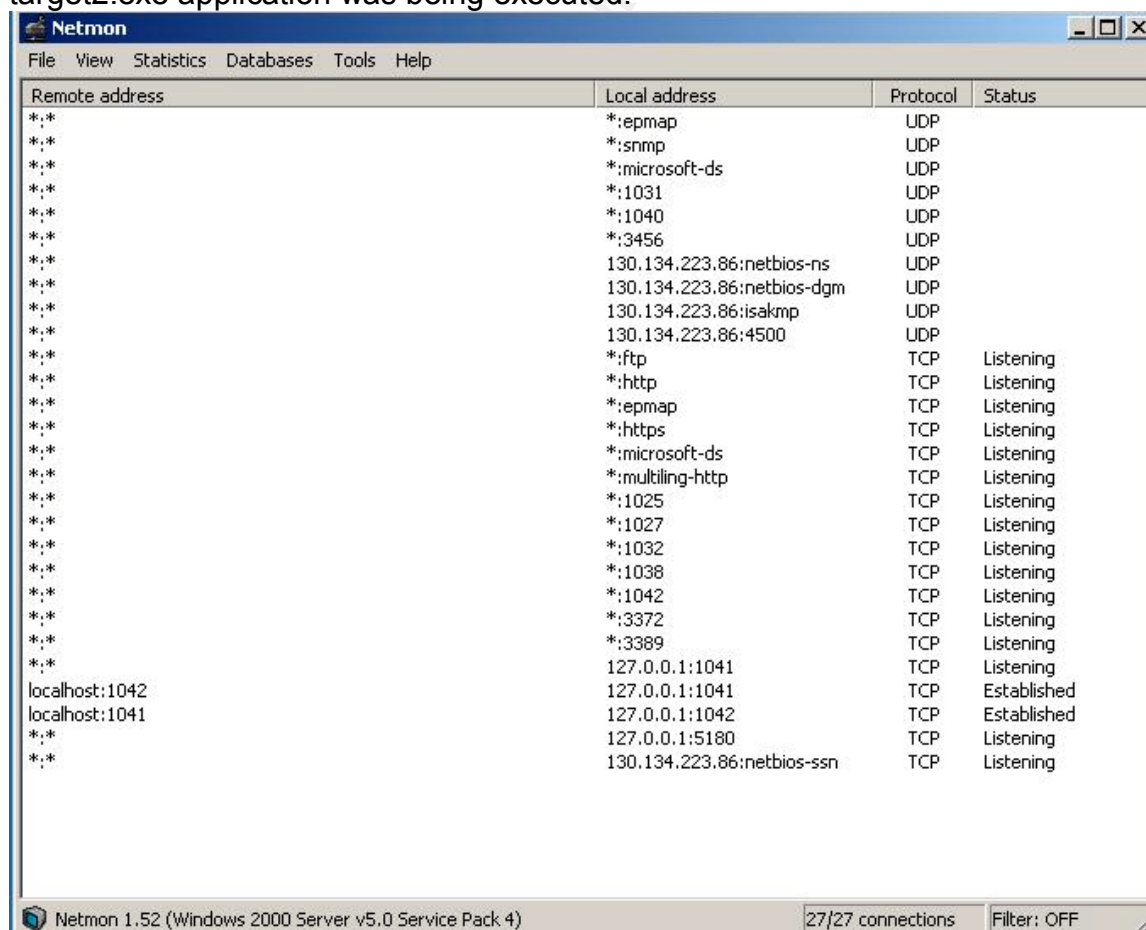
```

The output for the filemon application shows target2.exe accessing the Dynamic Link Libraries and also shows several attempts to find files that are not there. Lines 4 and 8-23 all show attempts by explorer.exe to open files associated with target2.exe that are not found on the filesystem. Line 28 shows the file looking for the desktop.ini file. The desktop.ini file is used to customize the Windows

user interface, an example might be for a user to customize their folder style or icon. The following the is the breakdown of the DLL file access information.

- Lines 42-45 show target2.exe attempting to locate the WS2_32.dll file starting at the directory where target2.exe was run. Line 45 shows the application find the dll file at C:\WINNT\system32\WS2_32.dll. Lines 46 and 47 show the applications opening (Execute) and closing access to the WS2_32.dll.
- Lines 48-50 show target2.exe attempting to locate WS2HELP.DLL starting at the location where the binary was run from. WS2HELP.DLL is found on Line 50 at C:\WINNT\SYSTEM32\WS2HELP.DLL. Lines 51 and 52 show target2.exe opening (executing) and closing the WS2HELP.DLL file.
- Lines 53-57 involve the MFC42.DLL file, the file is located on line 55 at C:\WINNT\system32\MFC32.DLL and is executed on line 56 and closed on line 57.
- Lines 58-62 involve the MSVCP60.DLL, which is located on line 60 at C:\WINNT\system32\MSVCP60.DLL, the file is executed and closed on lines 61 and 62.

The following is a screenshot of the Netmon utility that was running while the target2.exe application was being executed.



Screenshot of the Netmon application that was running while target2.exe was running.

The Netmon utility did not capture any useful information possibly because the binary failed to execute properly.

The target2.exe application while executing did appear to be an MS-DOS binary file, which could not run on a default load of Windows NT 4.0SP6 or Windows 2000 Server SP4. The application opened, queried, and closed several values in the local registry, mainly under the HKEY_LOCAL_MACHINE and HKEY_CURRENT_USER registry hives. The output from filemon confirms that the application uses local Dynamic Link Library files on the system and attempts to find a file called desktop.ini located in the directory where the application is run from.

Program Identification

The program is most likely an ICMP backdoor program for Windows. Searches were conducted using <http://www.google.com>, <http://www.altavista.com>, <http://www.yahoo.com>, and www.alltheweb.com. The following patterns were used during the searches: "Windows ICMP Backdoor", "Code by SpooF", "ICMP Backdoor" Windows, "ICMP Backdoor V0.1", and "199.107.97.191". No programs or information were found about an Windows ICMP Backdoor program. It is probable that this application is part of a larger Windows rootkit.

Defense

This backdoor program allows unauthorized access to systems using ICMP. This type of vulnerability should be prevented through the proper implementation of firewalls. ICMP should not be allowed to cross a perimeter firewall system. Even point-to-point ICMP rules allow for the potential of address spoofing. If ICMP is required for normal operation some type of authentication should be required to allow the ICMP to cross the firewall and the source and destination addresses should be limited. Providing some form of authentication would prevent applications from initiating ICMP connections through the firewall.

Norton Anti-Virus did not discover the application, so the application could potentially exist on a system and go undetected by an Anti-Virus program.

The use of the ICMP Backdoor application could be detected using Intrusion Detection Systems. The strings output of the program does not mention encryption so it is possible that the ICMP packets used in the communication would be cleartext. The ICMP packets would potentially be larger than normal ICMP packets.

Impact

This program appears to be an ICMP backdoor program. This commands appears to access the registry and provides a command shell to a user who enters a password, and the password is possibly loki. The program appears to attempt to install itself as a service. By installing as a service the ICMP backdoor program could provide system level access to anyone connecting to the system. Using ICMP to connect to the system this program allows attackers to potentially circumvent security measures such as network or host based firewalls. Local system controls are circumvented because the application provides a system level shell. Anyone who accesses the compromised system using the covert channel will have complete control of the system.

Legal Implications

The ICMP Backdoor program provides the potential for unauthorized access to a compromised system. Depending on who installed the binary and how the binary was installed on the machine would determine the legal implications of discovering the binary.

It is possible that a local administrator installed the binary to provide a backdoor to systems under his control. This type of installation would most likely violate local security policies and should result in some punishment and education for the administrator. The local firewall policies regarding the use of ICMP should also be investigated. The risk of allowing ICMP into a network must be weighed versus the potential use by an organization. In most cases ICMP should not be allowed to or from all internal systems.

If the binary was installed through some other means such as part of an email virus or Trojan horse program then it could be potentially difficult to track down the originator. However systems that attempted to access the computer using the backdoor program could be tracked. User who accessed this system could be guilty of violating the Computer Fraud Waste and Abuse Act.

Interview Questions

Several questions about the binary file need to be answered in order to determine the intent of the individual who installed the binary.

1. Who is the administrator of the system where the binary was found?
2. What users have access to the system? (List the user accounts on the system in question).
3. Because this binary affects Windows systems is this computer part of domain?
4. Has this system been returned to the network?
5. If the system has been returned to the network what steps were done to clean the system?
6. Has any software from the Internet recently been installed on the compromised system?
7. What users receive email on the compromised system?

8. What is the computer, where the binary was found, used for? Is the system mission critical?
9. What analysis, if any, analysis was done on the compromised system?
10. Where any other systems checked for the presence of this binary?
11. Is the compromised system or any other system accessible via ICMP? (Check Firewall and Intrusion Detection Logs)
12. Is the password to access the backdoor loki, or was that a reference in the code to the original Loki concept mentioned in phrack?
13. Why does the application attempt to load as a service? Why not just start from the registry?
14. Is this binary part of a larger rootkit?
15. Is this binary distributed as part of a worm?

Other Sources of Information

- <http://www.phrack.org/phrack/51/P51-06>. This article provided background information about LOKI and about covert channels in general.
- <http://www.sysinternals.com>. This website provided many of the tools used in this investigation. In addition information about the tool usage was gathered from the website.
- <http://www.securityfocus.com/infocus/1637>. This article from Securityfocus.com provided some information about how to reverse engineer hostile code.
- <http://www.foundstone.com>. This website provide bintext which was useful in gathering the initial string information from the binary.
- <http://www.liutilities.com/products/wintaskpro/dlllibrary/>. This website provided valuable information on the Dynamic Link Libraries.

Option 1 Section II

System Background

I was notified by the local IT Security Manager to begin analysis on a system that had been submitted to him by our Desktop support group. The desktop group apparently noticed an unusual entry in the process listing on the system. There were two entries titled "keylogger", the technician working on the computer immediately notified his supervisor. The supervisor then notified the local IT Security manager. Before analyzing the system the IT Security manager and myself conducted an interview with the computer technician and his supervisor regarding the computer.

The first question that was asked was when the technicians came in possession of the system. The technician responded by saying they had been in possession of the computer for several months. The user who had previously used the system was given a new system because this system had become unstable.

The next question asked was why they had the computer so long? The technician stated that they were going to keep the computer to use for their student aids when they came in over the summer. They were going to upgrade the system when they noticed the "keylogger" entry in the process listing.

The final question asked was if he or anyone else in the shop had installed any software or done any upgrades? He stated they had never upgraded the computer. They had done some work on the computer when it was still in the user's possession but after several attempts to fix the system the box was replaced. The technician claimed that no work had been done on the system since it was replaced.

The IT Security manager then informed the supervisor that we would be confiscating the computer at least long enough to image the drive and would probably keep the original drive for evidence. The supervisor said they would like the original computer back but that we could keep the drive.

The IT Security manager and myself took the system at approximately 2:15pm on May 15, 2003. The system was brought to the IT Security laboratory. This is a locked office with controlled access. The IT Security manager decided to notify the supervisor of the employee that was previously using the system. The supervisor was informed that an investigation was occurring on a system that belonged to one of his employees. The supervisor was asked a single question, was the employee in question the only user of the system. The supervisor responded that each employee in that area has an administrative system. He defined administrative system as a system for reading email and surfing the Internet. He also stated that this employee works in a secure location and

access to this area would be limited to a handful of employees (approximately six).

The current chain of custody or timeline of ownership for this system is:

System Possession	Dates of possession
Original owner	Unknown – January 2003
Desktop Support	January 6, 2003 – May 15, 2003
IT Security	May 15, 2003 – 2:15pm

Documentation was started showing the chain of custody and the desktop technicians signed for the period of time the computer was in their possession and then the IT Security manager signed for the date he picked the system up from the technicians and I signed for the system when the system was received from the IT Security manager.

The procedures that will be followed for investigating this system will be:

1. Current chain of custody will be established and system will be relocated to secure location for imaging and analysis.
2. A logical image of the system will be completed.
3. The original harddrive will be kept in a safe under the control of the IT Security Manager.
4. The logical image will be analyzed and any finding will be provided to management.
5. If necessary additional firewall, webproxy, mail, or Intrusion Detection System logs will be analyzed.
6. If necessary our agency law enforcement group would capture a physical image for the target drive. My findings would also be turned over to law enforcement for further analysis.

System Hardware

System is a Dell OptiPlex GX1P computer, with internal zip drive, CDRom, and floppy drive.
Tag # 2004824
Serial Number 5E84K
Model Number MMP

From the interview with the technician his documentation stated that the system was Windows 98, 128MB RAM, 9GB harddrive, and the processor was a Pentium III 500Mhz. He did not know the filesystem type. All this information would be verified on the system.

The harddrive was a Maxtor UltiMax.
Tag# 2004824A
Serial Number: W40XCANA

Model Number: 9102404

In order to perform the imaging on the system a Mandrake Linux 9.1 system was used. This system is often used to capture disk images in the field. Before the system is used a procedure to wipe the drive and reinstall Linux is performed to make sure of a clean filesystem prior to imaging. The process is to boot the laptop from Fire 0.4 CDROM (more details on the Fire CDROM in the imaging section) and complete the following set of commands:

#dd if=/dev/random of=/dev/hda

#dd if=/dev/zero of=/dev/hda

#dd if=/dev/zero of=/dev/hda

This procedure writes random data to the harddrive and then overwrites the data with zeros twice, this eliminates any data from previous installations and investigations from tampering with the current investigation.

The forensics system is a Dell inspiron 3700, 500Mhz Pentium III, 256MB RAM, 18GB internal harddrive.

Serial Number: 159-245-94

Model Number: PPX

Each potential investigator has a system assigned to them for performing system analysis. The forensic systems are locked in a safe controlled by the IT Security manager. Each time the forensics system is removed from the safe a log is kept and the investigator must sign the evidence out of the safe. These procedures are used to provide chain of custody for the evidence being analyzed.

The following tools were also installed on the forensics laptop:

Tool Name	URL
The SleuthKit 1.6.2	http://sleuthkit.sourceforge.net/index.php
Netcat 3.20.96	http://www.atstake.com/research/tools/network_utilities
Md5sum 2.0.21	Standard Linux utility
Regutils	ftp.cs.mun.ca/pub/regutils
Mac_daddy	http://www.incident-response.org/mac_daddy.html
Fire ISO	http://biatchux.dmzs.com/

A md5 checksum of common the tools used during the investigations is maintained, the md5sum were either gathered from websites or if the value is not available the value is calculated locally. Any time these tools are reinstalled the md5sum is compared to the previous value.

Media imaging

In order to capture a logical image, the forensics laptop and the system under investigation were both connected to a network hub. This hub has no other

connections other than the connections for the laptop and the system under investigation.

In order to do the image a FIRE 0.4 CDROM (the same one used for the forensics system cleaning) is used to boot the target system and copy the target harddrive across the network to the forensics laptop. FIRE is a product that can be used to conduct system investigations using a CDROM. FIRE has several different methods of booting and provides a variety of ways to gather data.

The FIRE CDROM has an MD5SUM of:
ae810533dc3ae95e4036b2d665bd5f1a fire-0.4a.iso

Prior to booting the FIRE CDROM the system BIOS (Basic Input Output System) was checked to verify that the system would boot from CDROM and also to verify the date on the system. The floppy drive was listed as the first boot device, followed by the harddrive. This was modified to the CDROM first, Floppy second, and harddrive third. The system clocked showed:

System clock Data/time: 10:11 5/17/03 Actual Date/Time: 10:16 5/17/03
--

In addition the memory in the system was verified to be 128Mbytes. This confirmed what the technician had stated.

After gathering this data and modifying the system BIOS it is time to boot from the FIRE CDROM and perform the image capture. FIRE presents the user with several choices on how to access the system:

- Using the FIRE tools, through a menu driven system (with virtual consoles)
- Booting in X Windows and using the tools manually
- Booting into console mode and using the tools manually

I prefer to use the tools manually and also like to have several windows open so I choose to boot into X Windows.

In order to begin the imaging process I first wanted to verify the partition layout. To do this I ran:

#fdisk /dev/hda (option p)

This command printed the partition table for the primary IDE drive in the target system. The partition was all contained within the first partition (/dev/hda1). The second step was to configure the network between the two systems. On the target system I entered:

#ifconfig eth0 192.168.0.22 netmask 255.255.255.0

This establishes the IP address for the target system to 192.168.0.22 with a 24 bit netmask. For the forensics system the following command was entered:

```
#ifconfig eth0 192.168.0.34 netmask 255.255.255.0
```

This establishes the IP address for the forensics system as 192.168.0.34 with a 24 bit netmask. In order to verify connectivity between the two a system a ping command was run from the forensics system to the target system:

```
#ping 192.168.0.22
```

```
PING 192.168.0.22 (192.168.0.22) from 192.168.0.34 : 56(84) bytes of data.  
64 bytes from 192.168.0.22: icmp_seq=1 ttl=128 time=0.417 ms  
64 bytes from 192.168.0.22: icmp_seq=2 ttl=128 time=0.402 ms
```

This confirms that the two systems are able to communicate and file transfer should now be possible. In order to do the imaging the dd command will be used to perform the bit for bit copying and netcat will be used to copy the data to the forensics system. Dd is a Unix utility used to copy all the data from one source to a target, the data includes free space on a drive.

The first step is to establish the netcat listener on the forensics system. The command to perform this action is:

```
# nc -l -p 31337 > hda1.img
```

This command establishes a netcat session listening on port 31337 and write whatever comes into that port into a file called hda1.img. Now on the target system we will begin the disk imaging.

```
#dd if=/dev/hda1 | nc 192.168.0.34 31337
```

This command will perform the bit for bit copy and copy the output of the dd command is sent to the netcat (nc) command with establishes a connection to the forensic system on port 31337. Once this command is complete we will perform two checksums one on the original drive and a second on the image file on the forensics system. To complete this process we will start by establishing the netcat listener on the forensics system:

```
#nc -l -p 31337 > hda1.org.md5sum
```

Again this netcat command is listening on port 31337 but this time is writing to a file called hda1.org.md5sum. Then on the target system run:

```
# md5sum /dev/hda1 | nc 192.168.0.34 31337
```


This command will create an md5 checksum of the original drive and copy the output to the netcat command. Finally we need a checksum of the imaged file from the forensics system.

#md5sum hda1.img > hda1.img.md5sum

Once these two checksums are created verify the output is the same. The checksums are used to validate the integrity of the image, the image file must be exactly the same as the original media.

#cat *md5sum

```
149e3ad6212ab0f04bcdf7378b0dfe88 hda1.img
149e3ad6212ab0f04bcdf7378b0dfe88 /dev/hda1
```

Because the output is identical this validates that the image file is the same as the original drive. At this point the CDROM is removed from the system and the system is powered down. The FIRE CDROM never mounted the original harddrive, and no analysis was attempted from the original image. This provides proof that the original harddrive was never modified.

A physical image of the target system was completed by our law enforcement agency. The physical image was created immediately following our logical image of the drive. The local policy at our location states that in any investigation that has potential criminal ramifications our law enforcement agency will complete the physical image of the target system. Our law enforcement agency has specific equipment to perform the duplication. All checksums between our logical image, the original image, and the duplicate image are verified.

The harddrive from the target system is removed from the system and is labeled and placed in a static bag and locked in a safe controlled by the IT Security manager. The computer itself is returned to the desktop support office. The IT Security Manager received and signed for the drive on May 19, 2003 at 9:00am. The duplicated physical drive is stored onsite under the supervision of the law enforcement agency. At the end of the investigation a briefing on our findings will be provided to the law enforcement agency.

Media Analysis

At this point the keylogger binary needs to be found and when and how did the binary get on the system. The first place to look for information on the binary is in the system registry. In addition the web cache and possibly mail files will be investigated to verify that the user did not receive the keylogger through a virus or web download.

The first step to the investigation is to mount the image and get some initial information about the filesystem. To mount the image I used to command:

```
#mkdir /mnt/forensics
#mount -t auto -o ro,nodev,noexec,noatime,loop /root/forensics/hda1.img
/mnt/forensics
```

This first command created a directory to mount the image. The second command mounted the image. The options to the mount command “-t auto” told Linux to attempt to figure out the file system, since I did not know for sure what the file system was. The “-o” specifies the options to the mount command in this case, read-only, no devices, do not update access time, and this is a loopback filesystem. The last two options specify the file to mount and where to mount the file. Next I ran the following command to get some information about the file system:

```
#mount
```

This command returns all the filesystems mounted on the system. The entry I’m concerned with is:

```
/root/forensics/hda1.img on /mnt/forensics type vfat
(ro,noexec,nodev,noatime,loop=/dev/loop0)
```

This entry tells me the filesystem is a vfat filesystem. In order to get more details I ran:

```
#fsstat -f fat /root/forensics/hda1.img > fsstat.out
#cat fsstat.out
```

FILE SYSTEM INFORMATION

```
-----
File System Type: FAT
OEM: MSWIN4.1
Volume ID: 131008783
Volume Label: NO NAME
File System Type: FAT32
```

META-DATA INFORMATION

```
-----
Range: 2 - 319443970
Root Directory: 2
```

CONTENT-DATA INFORMATION

```
-----
Sector Size: 512
Cluster Size: 8192
Sector of First Cluster: 19544
```

Total Sector Range: 0 - 19984795 FAT 0 Range: 32 - 9787 FAT 1 Range: 9788 - 19543 Data Area Sector Range: 19544 - 19984795

The fsstat command is part of the TASK toolset and provides some low-level information about the filesystem. This tells me that the filesystem type is FAT32.

The FAT32 filesystem is a type of File Allocation Table filesystem and is included with Windows 95 OSR2, Windows 98, and Windows ME. FAT32 uses 32 bit cluster identifiers and provides cluster sizes as large as 32KB. Unlike other FAT filesystems, the FAT32 root directory is not stored at a predefined location on the volume and the root directory has no upper limit on the size. In addition FAT32 provides a second copy of the boot sector for reliability. On the image the root directory is contained at cluster number two and the cluster size is 8KB.

The next step that will be performed on the image is to mount the image and perform an ASCII dump of the registry files. The two registry files on the system (either Windows 95 or more likely Windows 98) are the system.dat and user.dat files. The file system is still mounted read-only and both files are located at /mnt/forensics/windows directory.

In order to perform this operation the regutils programs were downloaded from: ftp://ftp.cs.mun.ca. The regutils commands allow Linux users to investigate Windows 9x registry files. The regedit command from the regutils toolset is the program that will be used to get information from the registry files.

The commands that were used to perform an ASCII dump of the two registry files were:

```
#!/root/regutils-0.10/regedit -f /mnt/forensics/windows/system.dat  
>/root/forensics/system.dat.txt  
#!/root/regutils-0.10/regedit -f /mnt/forensics/windows/user.dat  
>/root/forensics/user.dat.txt
```

The two commands read the registry files and output the binary files to ASCII files for manipulation. Before looking for the keylogger binary the operating system will be identified by looking through the registry. The registry key [HKEY_LOCAL_MACHINE\Software\Microsoft\Windows\CurrentVersion] provided some insight into the operating system.

[HKEY_LOCAL_MACHINE\Software\Microsoft\Windows\CurrentVersion] provided "FirstInstallDateTime"=hex:41,84,8e,25 "ProductName"="Microsoft Windows 98" "SystemRoot"="c:\\windows"
--

```
"Version"="Windows 98"  
"VersionNumber"="4.10.1998"
```

This information provided the version of Windows that the system was running and where the system root was located. The next registry key that requires investigating is:

```
\HKEY_LOCAL_MACHINE\Software\Microsoft\Windows\CurrentVersion\Run
```

This key will show the programs that are being started from the registry. This is a common area for Trojans or viruses to live because they are semi hidden and can be started at boot time. If the file is not located in the registry it could be located in a Windows initialization (ini) file or in the autoexec.bat file or config.sys file. Here is the output from that key:

```
[HKEY_LOCAL_MACHINE\Software\Microsoft\Windows\CurrentVersion\Run]  
"Adaptec DirectCD"="C:\Program Files\DirectCD\DIRECTCD.EXE"  
"AtiCwd32"="Aticwd32.exe"  
"AtiKey"="Atitask.exe"  
"Disknag"="C:\DELL\DISKNAG.EXE"  
"LoadPowerProfile"="Rundll32.exe powrprof.dll,LoadCurrentPwrScheme"  
"NodeMngr"="C:\DM\bin\NodeMngr.exe"  
"POINTER"="C:\PROGRA~1\MSHARD~1\point32.exe"  
"ScanRegistry"="C:\Windows\Scanregw.exe /autorun"  
"System Policies"="\\\\xxxx\sys\public\9xpol.exe"  
"System Tray"="SysTray.Exe"  
"SystemTray"="SysTray.Exe"  
"TaskMonitor"="c:\windows\taskmon.exe"  
"TCASUTIEXE"="TCAUDIAG.EXE -off"  
"Win32SysTray"="C:\WINDOWS\DESKTOP\WIN323.exe"
```

Several programs are started from this registry key every time this system starts. Most program are related to the CDROM burner (must have been an external burner, because the CDROM in the system is not a burner) or the video driver. The last entry is one of interest the file is called WIN323.exe and is located in the C:\windows\Desktop directory. This file could possibly be the keylogger binary.

After finding the above information in the system.dat file, I started looking through the user.dat file. While looking through this file I did not find any information regarding the keylogger or win323.exe but I did find reference to an interesting program called Camouflage.

```
[HKEY_USERS\DEFAULT\Software\Camouflage]  
  
[HKEY_USERS\DEFAULT\Software\Camouflage\CamouflageFile]  
"0"="C:\Program Files\Qualcomm\Eudora Mail\Attach\Breathe - Faith  
Hill.mp3"
```

```

[HKEY_USERS\DEFAULT\Software\Camouflage\frmMain]
"Height"=dword:00000d02
"Left"=dword:00001716
"Top"=dword:0000177f
"Width"=dword:00001ce3
"WindowState"=dword:00000000

[HKEY_USERS\DEFAULT\Software\Camouflage\frmMain\CamouflageFileList]
"Accessed"=dword:00000000
"Attributes"=dword:000003e8
"Created"=dword:00000000
"File"=dword:00001130
"Modified"=dword:00000000
"Size"=dword:000004b0

[HKEY_USERS\DEFAULT\Software\Camouflage\frmMain\UncamouflageFileList
]
"Accessed"=dword:00000000
"Attributes"=dword:000003e8
"Created"=dword:00000000
"File"=dword:00001130
"Modified"=dword:00000000
"Size"=dword:000004b0

[HKEY_USERS\DEFAULT\Software\Camouflage\OutputFile]
"0"="C:\\WINDOWS\\Desktop\\alliwant.mp3"

[HKEY_USERS\DEFAULT\Software\Camouflage\OutputFolder]
"0"="C:\\WINDOWS\\Desktop\\Jlo"
"1"="C:\\WINDOWS\\Desktop\\My Folder\\Joe"
"2"="C:\\WINDOWS\\Desktop\\Luniz"
"3"="C:\\WINDOWS\\Desktop\\Nas\\Nas"

```

It is possible that the user is using this camouflage program to possibly hide certain files or perhaps keylogger data files. I decided to search the www.google.com for information regarding the Camouflage program. Some information about the camouflage program was found at:
<http://www.jjtc.com/stegoarchive/stego/software.html>:

Camouflage (Freeware)

Camouflage is a Windows-based program that allows you to hide files by scrambling them and then attaching them to the end of the file of your choice. Password protection included. The hidden file can be detected by examining the raw file data and seeing that the hidden file has been added

*after the normal carrier data, cut this will only appear as gibberish since the data is encrypted.*⁸

More details about the Camouflage program could be required if there is a keylogger application on the system or if data from the keylogger application cannot be found.

Prior to looking at the binary some more information about the target system will be acquired such as when the operating system was installed and what applications are installed on the system.

Information about the operating system will be gathered first. The stat utility will be run against kernel32.dll file in the /mnt/forensics/windows/system32 directory and also against the explorer.exe file in the /mnt/forensics/windows directory. This two files should provide information about when this system was installed.

#stat kernel32.dll

```
File: "kernel32.dll"
  Size: 471040      Blocks: 928      IO Block: 8192  Regular File
Device: 700h/1792d  Inode: 1312080  Links: 1
Access: (0755/-rwxr-xr-x) Uid: (  0/   root) Gid: (  0/   root)
Access: Mon May 11 19:01:00 1998
Modify: Mon May 11 19:01:00 1998
Change: Mon Dec 31 23:00:00 1979
```

#stat ../explorer.exe

```
File: "../explorer.exe"
  Size: 180224      Blocks: 352      IO Block: 8192  Regular File
Device: 700h/1792d  Inode: 1310993  Links: 1
Access: (0755/-rwxr-xr-x) Uid: (  0/   root) Gid: (  0/   root)
Access: Mon May 11 19:01:00 1998
Modify: Mon May 11 19:01:00 1998
Change: Mon May 11 19:01:00 1998
```

It appears the system was probably installed on May 11, 1998 at 19:01. Next a list of installed applications will be gathered. The following applications were found on the system:

- Eudora (email client)
- Netscape 4.7 (web browser)
- Easy CD Creator
- Microsoft Office 97
- Filemaker Pro 4
- Filemaker Pro 5

- Visio
- Canvas 6
- AutoCAD

In order to determine the last time the user received or sent email some files in the Eudora directory will be investigated.

#stat Eudora.exe

```
File: "Eudora.exe"
Size: 2510905      Blocks: 4912      IO Block: 8192  Regular File
Device: 700h/1792d Inode: 1314647   Links: 1
Access: (0755/-rwxr-xr-x) Uid: ( 0/  root) Gid: ( 0/  root)
Access: Wed Apr 11 15:57:04 2001
Modify: Wed Apr 11 15:57:04 2001
Change: Tue Sep 18 10:39:54 2001
```

#stat in.mbx

```
File: "In.mbx"
Size: 114901      Blocks: 240      IO Block: 8192  Regular File
Device: 700h/1792d Inode: 1314621   Links: 1
Access: (0755/-rwxr-xr-x) Uid: ( 0/  root) Gid: ( 0/  root)
Access: Tue Feb 19 10:46:38 2002
Modify: Tue Feb 19 10:46:38 2002
Change: Thu Feb 14 10:39:54 2002
```

These two commands show that the binary was modified and accessed April of 2001 but the inbox was modified on February 19, 2002. It is possible that the February 19 date was the last day the user accessed email from this system.

The Easy CD Creator program is also interesting because this system has no internal CDROM burner so why would this application be installed. It is possible the CD burner could have been used to move data off the system during an upgrade.

#stat Creatr32.exe

```
File: "Creatr32.exe"
Size: 2027008      Blocks: 3968      IO Block: 8192  Regular File
Device: 700h/1792d Inode: 1314971   Links: 1
Access: (0755/-rwxr-xr-x) Uid: ( 0/  root) Gid: ( 0/  root)
Access: Mon Nov 30 02:50:00 1998
Modify: Mon Nov 30 02:50:00 1998
Change: Mon Nov 30 02:50:00 1998
```

These dates are November 30, 1998, it appears that the CD Creator application has not been used for several years.

Binary file analysis

Now that some operating system and application information have been gathered on the target system the suspicious binary file will be investigated to determine if it is in fact the keylogger application. The file and stat commands will be used to get some initial information about the file. The Unix utility strings will be used to look at the win323.exe binary. Before running strings on the file the stat and file commands will be used to get some information about the file. Here is the output from the file and stat commands:

#file win323.exe

win323.exe: MS-DOS executable (EXE), OS/2 or MS Windows

stat win323.exe

```
File: "win323.exe"
Size: 28672    Blocks: 64      IO Block: 8192   Regular File
Device: 700h/1792d Inode: 625202    Links: 1
Access: (0755/-rwxr-xr-x)  Uid: (  0/   root)   Gid: (  0/   root)
Access: Fri Dec 21 18:31:16 2001
Modify: Fri Dec 21 18:31:16 2001
Change: Fri Dec 21 18:31:16 2001
```

The output of the file command confirms that it is a binary file. The output of the stat command gives me details above the size of the file 28672 bytes and the MAC times on the file (all Dec 21 18:31:16:2001). The command to gather the strings in the binary file:

```
#strings -a /mnt/forensics/windows/Desktop/win323.exe
>/root/forensics/output/win323.txt
```

Here is the output of that command:

```
!This program cannot be run in DOS mode.
RichA
.text
.data
.rsrc
MSVBVM60.DLL
[Qsn
```


RsaTQs
TQskcDs
Pss
UQst
RssADs
QsmYOs
'Os0XQsaUQs?
Rsn[Ps
OsFUDs4
ADstEDs
UQsPOQs
EDs<
FDsF
Qs"DDsj
MDsS
Project1
KeyLogger
Sensible KeyLogger
wwwwwwwwwwwwx
wwwwwwwwwwwwx
tDDDDDDDD@
tDDDDDDDDGppw
tDDDDDDDDGppw
tDDDDDDDDDDDDH
wwwwwwwwwwwwx
Form1
txtFileName
c:\keylog.txt
TimerSave
Timer1
Text1
Label2
Output File Name:
menAbout
&About
menAknow
&Acknowledgements
menBy
VB5!
Win32
KeyLogger
Project1
4x\$@
Project1
KeyLogger
KeyTrap

C:\Program Files\Microsoft Visual Studio\VB98\VB6.OLB

TimerSave

Form

Timer1

menBy

VB5!

Win32

KeyLogger

Project1

4x\$@

Project1

KeyLogger

KeyTrap

Form

C:\Program Files\Microsoft Visual Studio\VB98\VB6.OLB

TimerSave

Timer1

menBy

menAknow

txtFileName

menAbout

Label2

Text1

user32

GetAsyncKeyState

hD\$@

Form_Load

GetKeyState

h4%@

advapi32.dll

RegCloseKey

hd'@

RegCreateKeyExA

RegOpenKeyExA

RegQueryValueExA

h@(@

RegSetValueExA

__vbaFileClose

__vbaVarAdd

RegDeleteKeyA

RegDeleteValueA

h4)@

VBA6.DLL

__vbaWriteFile

__vbaFileOpen

__vbaOnError

__vbaVarCat
 __vbaStrVarCopy
 __vbaFreeVarOverflow
 __vbaFreeVar
 __vbaFpl4
 __vbaFreeVarList
 __vbaFreeObjList
 __vbaVarSub
 __vbaI4Var
 __vbaStrVarVal
 __vbaVarTstGt
 __vbaObjSet
 __vbaLenBstr
 __vbaVarMove
 __vbaVarCopy
 __vbaSetSystemError
 __vbaFreeStrList
 __vbaStrCat
 __vbaStrMove
 __vbaFreeObj
 __vbaFreeStr
 __vbaHresultCheckObj
 __vbaNew2
 __vbaUI1I2
 __vbaGenerateBoundsError
 __vbaStrCopy
 __vbaExitProc
 __vbaResume
 __vbaVargVarMove
 __vbaStrVarMove
 __vbaError
 __vbaVarVargNofree
 __vbaStrToUnicode
 __vbaStrToAnsi

(SVW

F<u&

F<u!

F<uL

F<u/

F<uL

F<uL

F<uL

F<uL

F<uL

F<u!

F<u!

F<u!
F<u!
F<u!
F<u!
F<u!
F<u!
F<u!
F<u!
F<u!
F<u!
F<uL
F<uH
F<uL
F<uL
haA@
h<'@
h %@
0SVW
h_G@
h_G@
(SVW
h4H@
MSVBVM60.DLL
__vbaVarTstGt
__vbaVarSub
__Clcos
__adj_fptan
__vbaVarMove
__vbaVarVargNofree
__vbaFreeVar
__vbaStrVarMove
__vbaLenBstr
__vbaFreeVarList
__adj_fdiv_m64
__vbaFreeObjList
__adj_fprem1
__vbaResume
__vbaStrCat
__vbaError
__vbaWriteFile
__vbaSetSystemError
__vbaHresultCheckObj
__adj_fdiv_m32
__vbaExitProc

```

__vbaOnError
__adj_fdiv_m16i
__vbaObjSet
__adj_fdivr_m16i
__Clsin
__vbaVargVarMove
__vbaChkstk
__vbaFileClose
EVENT_SINK_AddRef
__vbaGenerateBoundsError
DllFunctionCall
__adj_fpatan
EVENT_SINK_Release
__vbaUI112
__Clsqrt
EVENT_SINK_QueryInterface
__vbaExceptHandler
__vbaStrToUnicode
__adj_fprem
__adj_fdivr_m64
__vbaFPException
__vbaStrVarVal
__vbaVarCat
__Cllog
__vbaErrorOverflow
__vbaFileOpen
__vbaNew2
__adj_fdiv_m32i
__adj_fdivr_m32i
__vbaStrCopy
__vbaFreeStrList
__adj_fdivr_m32
__adj_fdiv_r
__vbaI4Var
__vbaVarAdd
__vbaStrToAnsi
__vbaVarCopy
__vbaFpl4
__Clatan
__vbaStrMove
__vbaStrVarCopy
__allmul
__Cltan
__Clexp
__vbaFreeStr
__vbaFreeObj

```

```
wwwwwwwwwwwwwwwx
tDDDDDDDD@
wwwwwwwwwwwwwwwx
tDDDDDDDDGppw
tDDDDDDDDGppw
tDDDDDDDDDDDDH
wwwwwwwwwwwwwwwx
```

There is some valuable information contained within this output. There are several references to the word Keylogger and one that references “sensible keylogger”. There is some reference to a file called C:\keylog.txt and references to several dynamic Load Library (DLL) files. The program could be written in visual basic because there are several references to “vba”. It is probable that this is the keylogger application. There was another file located in the same location as the discovered binary, the file was named dat.cab. This is an unusual location for an apparent Microsoft CAB file. This file should be examined to determine what it contains.

#file dat.cab

dat.cab: ASCII text, with CRLF, CR line terminators

This looks unusual for the file command to return ASCII text for what appears to be a Windows cabinet file (at least has the cabinet file extension). I decided to try to find another cab file on the image and run the file command against that file to compare the results.

#file /mnt/forensics/windows/options/cabs/net9.cab

/mnt/forensics/windows/options/cabs/net9.cab: Microsoft cabinet file data, v1.3

The file command verified the output for an actual Microsoft Cabinet file. So the dat.cab is possibly not an actual Microsoft cabinet file. Before looking at the file the stat command will be used to get information about the dat.cab file:

#stat dat.cab

```
File: "dat.cab"
Size: 135647   Blocks: 272   IO Block: 8192   Regular File
Device: 700h/1792d Inode: 625201   Links: 1
Access: (0755/-rwxr-xr-x) Uid: (  0/   root) Gid: (  0/   root)
Access: Tue May 13 11:17:30 2003
Modify: Tue May 13 11:17:30 2003
Change: Fri Mar 22 10:52:46 2002
```

The file size for the file is 135647 bytes and the modify and access times are both May 13, 2003, with a change time of March 22, 2002. Now that information

about the file has been gathered the file will be examined with a text editor to see what is in the file. When the file was opened it contains 33864 Lines and 135647 characters (only a portion of the file is displayed). Mainly the file consists of "" but there are some commands in the file such as:

```
"(Cant Undo...Bksp...t32
"sysedit
"logger
"
"key"
""
"keylogger
""
""
"i
"
"syncconfig.exe"
"debu
"/"
"/"
"/ordpa...Bksp...d...Bksp...
"
"k(Cant Undo)sysconfigex...Bksp...e"
">cb"
"a"
"cab
//"
```

There are also several other entries that are possibly passwords or other personal information (not displayed). It appears that some of the files for collecting data have been found. Very quickly the keylogger application and some keystroke information have been discovered on the system.

How the keylogger software got on the system whether it was deliberate or possibly from some virus sent by email still needs to be determined. One way to gather information is to run the strings command against the raw image file looking for any reference to the word "keylogger".

#strings -a /root/forensics/hda1.img | grep keylogger

The output of this command produced 1211 Lines of references to keylogger, mainly websites and HTML code (some of the keylogger pages are probably in the users cache). The various websites included:

```
http://keylogger.com/images/contactoff.gif
AllTheWeb.com: Web pages results for `dat.cab keylogger'
```

```

http://www.alltheweb.com/search?cat=web&lang=any&query=dat.cab+keylogger
http://ln.doubleclick.net/ad/atw.ly.ln/r_atw;kw=ghost+keylogger+faq;h=misc;prov=
fast_atw;pos=2;sz=320x20;tile=2;category=adult_atw;ratio=1_3;ord=5978990?
http://home.swipnet.se/~w-94075/keylogger/
AllTheWeb.com: Web pages results for `download ghost 1.0 keylogger'
http://www.alltheweb.com/search?cat=web&lang=any&query=download+ghost+1
.0+keylogger
http://pa.yahoo.com/pa?q=keylogger&s=9035784
http://www.keylogger.hp
http://www.keyloggerhp
http://www.keylogger.net
Results for 'ghost keylogger'
http://search.cert.org/query.html?rq=0&col=allcert&ht=0&qp=&qs=&qc=&pw=100
%25&ws=1&la=&qm=0&st=1&nh=25&lk=1&rf=2&oq=&rq=0&si=1&qt=ghost+keyl
ogger
http://keylogger.com/images/top4.gif
http://www.keylogger.net/transparent.
http://www.keylogger.com/ik97v12s.exe

```

This is just a portion of the file. The last entry above could indicate the actual binary download. It also appears that someone is doing a query looking for “download ghost 1.0 keylogger” and also “dat.cab + keylogger”. This further proves that the above dat.cab file is involved with the keylogger application. This evidence confirms that someone on this computer was actively gathering information about keyloggers.

The current company policy dictated that Netscape was the only authorized browser, and this policy was enforced with User-Agent-Strings verification at our corporate webproxy system. The location of the Netscape cache files is: /mnt/forensics/Program Files/Netscape/Users/user_name. There were 27 html files in the directory and over 1000 images in the directory. I decided to get a listing of the HTML files to show when they were created:

#ls -l *htm

```

-rwxr-xr-x 1 root root 5684 Feb 12 2002 bookmark.htm
-rwxr-xr-x 1 root root 14984 Mar 11 2002 m01f3n08.htm
-rwxr-xr-x 1 root root 23826 Mar 11 2002 m030bm1m.htm
-rwxr-xr-x 1 root root 18727 Mar 11 2002 m062dtt2.htm
-rwxr-xr-x 1 root root 26374 Mar 11 2002 m07e8h0l.htm
-rwxr-xr-x 1 root root 4602 Mar 20 2002 m0ick9mm.htm
-rwxr-xr-x 1 root root 24191 Mar 20 2002 m0m148bs.htm
-rwxr-xr-x 1 root root 17938 Mar 11 2002 m0s8hn7s.htm
-rwxr-xr-x 1 root root 21325 Mar 11 2002 m1io3vg4.htm
-rwxr-xr-x 1 root root 8376 Mar 20 2002 m1n9l0pp.htm
-rwxr-xr-x 1 root root 4531 Mar 20 2002 m1vq2u34.htm

```


-rwxr-xr-x	1	root	root	6234	Mar 11	2002	mu23jdib.htm
-rwxr-xr-x	1	root	root	127588	Mar 11	2002	mua370dl.htm
-rwxr-xr-x	1	root	root	10434	Mar 20	2002	muilt08g.htm
-rwxr-xr-x	1	root	root	4236	Mar 20	2002	mup43qmk.htm
-rwxr-xr-x	1	root	root	3820	Mar 11	2002	mv3i9l62.htm
-rwxr-xr-x	1	root	root	3570	Mar 20	2002	mv7p4kg6.htm
-rwxr-xr-x	1	root	root	9848	Mar 19	2002	mv9gr921.htm
-rwxr-xr-x	1	root	root	1506	Mar 20	2002	mvfidq0o.htm
-rwxr-xr-x	1	root	root	1019	Mar 11	2002	mvim45vp.htm
-rwxr-xr-x	1	root	root	9840	Mar 11	2002	mvnbot7u.htm
-rwxr-xr-x	1	root	root	11041	Mar 20	2002	mvprcvh8.htm
-rwxr-xr-x	1	root	root	9302	Mar 11	2002	mvqrgauf.htm
-rwxr-xr-x	1	root	root	11991	Mar 20	2002	mvqt7725.htm
-rwxr-xr-x	1	root	root	11116	Mar 20	2002	mvspjis9.htm
-rwxr-xr-x	1	root	root	683294	Mar 20	2002	mvta6shq.htm
-rwxr-xr-x	1	root	root	7974	Mar 19	2002	mvvvrdo.htm

The stat command on the win323.exe binary file showed the MAC time as Fri Dec 21 18:31:16 2001, all of these HTM files were created after that date. To see if these files had any data on the keylogger application I ran:

#grep keylogger *.htm

The output of the command verified that they keyword keylogger was contained within some of the htm files. I was confused about the dates, the original application was installed before these sites were visited (according to their file times). Had the user returned to gather more information? Possibly the user had discovered the keylogger and was trying to get information about it?

I also decided to look in other common data areas to see if any keylogger information was there. The directories I choose were:

/mnt/forensics/windows/Temporary Internet Files/

/mnt/forensics/windows/temp

/mnt/forensics/windows/Application Data/Mozilla/Profiles

Searching these directories for keylogger information produced no further evidence. Next I looked for the index.dat file for Internet Explorer and the netscape.hst file for Netscape. These two files maintain a list of all sites a user has browsed using that web browser. The index.dat file (Internet Explorer) is located at /mnt/forensics/windows/Temporary Internet Files/

#stat /mnt/forensics/windows/Temporary Internet Files/index.dat

File: "index.dat"

Size: 49152 Blocks: 96 IO Block: 8192 Regular File
Device: 700h/1792d Inode: 1449537 Links: 1

```
Access: (0755/-rwxr-xr-x) Uid: ( 0/ root) Gid: ( 0/ root)
Access: Tue May 13 11:17:30 2003
Modify: Tue May 13 11:17:30 2003
Change: Wed Sep 15 07:15:46 1999
```

The index.dat file did not contain any information regarding the keylogger and looking at the access and modify date on the file it is possible that the desktop technicians used Internet Explorer to surf the web.

I did not expect to find any information in the Internet Explorer file. Internet Explorer was not allowed through our web proxy system when the original user had the computer. Any web surfing the user did regarding the keylogger application would have been done with Netscape. The two files of interest with Netscape are /mnt/forensics /Program Files/Netscape/users/user_name/netscape.hst and /mnt/forensics /Program Files/Netscape/users/user_name/fat.db these two files record all web activity with Netscape. The stat and file commands will be used to gather information about the two files.

#stat /mnt/forensics /Program Files/Netscape/users/user_name/fat.db

```
File: "fat.db"
Size: 1212416      Blocks: 2368      IO Block: 8192  Regular File
Device: 700h/1792d Inode: 1446664   Links: 1
Access: (0755/-rwxr-xr-x) Uid: ( 0/ root) Gid: ( 0/ root)
Access: Fri Mar 22 10:59:38 2002
Modify: Fri Mar 22 10:59:38 2002
Change: Thu Feb 7 13:09:16 2002
```

#stat /mnt/forensics /Program Files/Netscape/users/user_name/netscape.hst

```
File: "netscape.hst"
Size: 323584      Blocks: 640      IO Block: 8192  Regular File
Device: 700h/1792d Inode: 1444576   Links: 1
Access: (0755/-rwxr-xr-x) Uid: ( 0/ root) Gid: ( 0/ root)
Access: Fri Mar 22 10:59:38 2002
Modify: Fri Mar 22 10:59:38 2002
Change: Mon Oct 30 09:29:30 2000
```

#file netscape.hst

netscape.hst: Berkeley DB 1.85 (Hash, version 2, native byte-order)

#file fat.db

fat.db: Berkeley DB 1.85 (Hash, version 2, native byte-order)

These are binary files so any data gathered will have to be done with the strings command. The following command will be used to see how many if any lines in the file contain the keyword keylogger.

```
#strings fat.db | grep keylogger | wc -l
```

```
137
```

```
#strings netscape.hst | grep keylogger | wc -l
```

```
146
```

The strings and grep commands are being piped to the unix utility wc, which provides a word counts function. The fat.db file contained 137 references to keylogger and netscape.hst contained 146 references. The strings and grep commands will be run again this time with their output captured to files that will be maintained as evidence. Here are some of the keylogger references contained in the netscape.hst file:

```
http://keylogger.com/images/contactoff.gif
http://keylogger.com/images/contactoff.gif
http://keylogger.com/images/contactoff.gif
http://keylogger.com/images/contactoff.gif
AllTheWeb.com: Web pages results for `dat.cab keylogger'
http://www.alltheweb.com/search?cat=web&lang=any&query=dat.cab+keylogger
http://ln.doubleclick.net/ad/atw.ly.ln/r_atw;kw=ghost+keylogger+faq;h=misc;prov=
fast_atw;pos=2;sz=320x20;tile=2;!category=adult_atw;ratio=1_3;ord=5978990
?
http://ln.doubleclick.net/ad/atw.ly.ln/r_atw;kw=win32+keylogger;h=misc;prov=fast
_atw;pos=2;sz=320x20;tile=2;!category=adult_atw;ratio=1_3;ord=9990153?
http://www.keylogger.com/images/side-products.gif
http://www.keylogger.com/images/downloadsoff.gif
http://www.alltheweb.com/go/1/H/web/http/www.keylogger.net/index.html
Ghost Keylogger, an invisible keylogger with email fun
Ghost Keylogger, an invisible keylogger with email fun
Ghost Keylogger, an invisible keylogger with email functionality.
http://www.keylogger.net/index.ht
Ghost Keylogger, an invisible keylogger with email f
Ghost Keylogger, an invisible keylogger with email functionality.
http://www.keylogger.net/index.html
Results for 'ghost keylogger'
Results for 'ghost keylogger'
```

The entries in these Netscape files are key in determining that someone was actively attempting to install a keylogger on this system. The keylogger was not installed inadvertently in fact it appears some was searching for keylogger information.

MACTime Analysis

MACtimes are time information stored by the filesystem to identify created, accessed and modified times on files. This information can be used to show file system modifications such as when a system was installed, when applications were installed, and when files were created.

The mac_daddy script will be run over the mounted image to gather the MAC times. The mac_daddy script will gather the time information on the mounted file system, it will not gather data on deleted files. The command to do this was:

```
#mac_daddy /mnt/forensics > /root/forensics/output/mac_daddy.out
```

The size of the created file is 4704486 bytes and I also performed a MD5sum on the file:

```
#md5sum mac_daddy.out
```

```
932a73dd85892422b37f949c2090f47a mac_daddy.out
```

The output of the mac_daddy file was then analyzed looking for dat.cab entries.

```
# grep dat.cab mac_daddy.out
```

Jan 14 2002 07:14:20	135744	..c -rwxr-xr-x	root	root	/mnt/forensics/dat.cab
Feb 22 2002 09:09:02	135744	ma. -rwxr-xr-x	root	root	/mnt/forensics/dat.cab
Mar 22 2002 10:52:46	135647	..c -rwxr-xr-x	root	root	/mnt/forensics/windows/Desktop/dat.cab
Mar 27 2002 13:02:22	341	..c -rwxr-xr-x	root	root	/mnt/forensics/windows/Recent/dat.cab.lnk
Mar 27 2002 13:02:24	341	ma. -rwxr-xr-x	root	root	/mnt/forensics/windows/Recent/dat.cab.lnk
	135647	ma. -rwxr-xr-x	root	root	/mnt/forensics/windows/Desktop/dat.cab

The fourth and fifth entries are symbolic links for a dat.cab file that showed up in one of the recent files on the windows system. This could mean some viewed the file May 27 2002 at 13:02. There are two dat.cab files:

- The first dat.cab file is directly on the C:\ drive and was created on Jan 14, 2002 and was modified and accessed on Feb 22, 2002 9:09:02.
- The second dat.cab file is located in the windows/Desktop directory (this is the file I originally discovered). This file was created March 22, 2002 10:52 and was modified and accessed on May 27, 2002. March 22nd was also the date that the netscape.hst and fat.db files were last modified.

The next thing to investigate is the dat.cab file located in the root directory of the system.

#stat /mnt/forensics/dat.cab

```
File: "dat.cab"
Size: 135744      Blocks: 272      IO Block: 8192   Regular File
Device: 700h/1792d Inode: 900952   Links: 1
Access: (0755/-rwxr-xr-x) Uid: (  0/   root) Gid: (  0/   root)
Access: Fri Feb 22 09:09:02 2002
Modify: Fri Feb 22 09:09:02 2002
Change: Mon Jan 14 07:14:20 2002
```

The size of this file is 135744 bytes (this file is a little larger than the first dat.cab file). The Modify and Access time for the file is Feb 22, 2002 and the change time is Jan 14, 2002 these dates are closer to the binary installation date of Dec 21, 2001. This data file was created prior to the first examined file. Now the contents of the file need to be examined.

#cat /mnt/forensics/dat.cab

Again this file was filled with "" but there was also clear evidence of emails, email addresses, and passwords being captured. The data captured in the second file appears much more sensitive than the data captured in the file C:\windows\Desktop\dat.cab. This is a sample of the data that was captured much of the data in this file was potentially sensitive and is not included.

```
"da...Bksp...C"^M
"mollette"^M
"daollet...Bksp.....Bksp.....Bksp.....Bksp.....Bksp.....Bksp..."^M
"(Cant Undo)lGuy...Bksp.....Bksp...s"^M
"Hey ...Bksp...Af...Bksp.....Bksp...ds...Bksp.....Bksp...a...Bksp.....Bksp...s this you
Axam? l...Bksp...ts me xxx."^M
```

This data is definitely different than the first keystroke capture file. This is the second data file that has been recovered.

#grep win323.exe mac_daddy.out

```
Dec 21 2001 18:31:16 28672 mac -rwxr-xr-x root root
/mnt/forensics/windows/Desktop/win323.exe
```

This command confirmed what the stat of the file had shown the keylogger executable had been installed on Dec 21, 2001 18:31.

```
# grep ik97v12s.exe mac_daddy.out
```

This command returned nothing. This could mean that the original download executable was possibly deleted and has been overwritten by the Operating System.

Several pieces of information have been gathered but are there more dat.cab files that were deleted and there could also be more evidence in cache html files that were deleted. In order to get at that data I would have to use the Sleuthkit (TASK) to generate a timeline complete with deleted file information. The commands that were used to create the second timeline using tools from TASK 1.62 were:

```
#fls -f fat32 -m / -r /root/forensics/hda1.img > ./output/fls.out
#ils -f fat32 -m -r /root/forensics/hda1.img > ./output/ils.out
#cat ils.out fls.out > mac.out
#mactime -b mac.out > timeline.out
```

The fls command is used to list file and directory names from a forensics image (fls manpage). The parameters for the fls command are -f the filesystem type (fat32) -m lists the mount point for the image, and -r means to recursively list directories. The ils command is used to list inode information and by default lists only inodes of deleted files. The parameters to the ils command are -f the filesystem type (fat32) -m display output in format for the mactime program to read the data, -r means to recursively list directories. The cat command is used to combine the two files into one file called mac.out. Finally the mactime program from TASK is used to generate the timeline. The -b option for mactime is used to specify the file. Next the Unix utility grep will be used to look for dat.cab files in the timeline.out file:

```
# grep dat.cab timeline.out
```

```
Mon Jan 14 2002 07:14:20 135744 ..c -/-rwxrwxrwx 0 0 85 /dat.cab (DAT.CAB)
135744 m.. -/-rwxrwxrwx 0 0 85 /dat.cab (DAT.CAB)
Fri Feb 22 2002 12:11:34 32979 ..c -/-rwxrwxrwx 0 0 14178842 /WINDOWS/DESKTOP/dat.cab (_AT.CAB)
(deleted)
Fri Mar 08 2002 08:41:56 22887 m.. -/-rwxrwxrwx 0 0 14178851 /WINDOWS/DESKTOP/Copy of dat.cab
(_OPYOF~1.CAB) (deleted)
22887 ..c -/-rwxrwxrwx 0 0 14178851 /WINDOWS/DESKTOP/Copy of dat.cab
(_OPYOF~1.CAB) (deleted)
22887 .a. -/-rwxrwxrwx 0 0 14178851 /WINDOWS/DESKTOP/Copy of dat.cab
(_OPYOF~1.CAB) (deleted)
135744 .a. -/-rwxrwxrwx 0 0 85 /dat.cab (DAT.CAB)
54 .a. -/-r-xr-xr-x 0 0 14178822 /WINDOWS/DESKTOP/~$dat.cab (_$DAT.CAB) (deleted)
54 .a. -/-r-xr-xr-x 0 0 14178824 /WINDOWS/DESKTOP/~$dat.cab (_$DAT.CAB) (deleted)
32979 .a. -/-rwxrwxrwx 0 0 14178842 /WINDOWS/DESKTOP/dat.cab (_AT.CAB) (deleted)
54 .a. -/-r-xr-xr-x 0 0 83 /~$dat.cab (_$DAT.CAB) (deleted)
Fri Mar 22 2002 10:16:28 32979 m.. -/-rwxrwxrwx 0 0 14178842 /WINDOWS/DESKTOP/dat.cab (_AT.CAB)
(deleted)
135647 ..c -/-rwxrwxrwx 0 0 14178827 /WINDOWS/DESKTOP/dat.cab (DAT.CAB)
Fri Mar 22 2002 10:57:06 54 ..c -/-r-xr-xr-x 0 0 14178824 /WINDOWS/DESKTOP/~$dat.cab (_$DAT.CAB)
(deleted)
```

Fri Mar 22 2002 10:57:08	54 m.. -/r-xr-xr-x 0	0	14178824 /WINDOWS/DESKTOP/~\$dat.cab (_\$DAT.CAB) (deleted)
	54 ..c -/r-xr-xr-x 0	0	83 /~\$dat.cab (_\$DAT.CAB) (deleted)
Fri Mar 22 2002 11:04:40	54 m.. -/r-xr-xr-x 0	0	83 /~\$dat.cab (_\$DAT.CAB) (deleted)
Fri Mar 22 2002 11:05:02	54 ..c -/r-xr-xr-x 0	0	14178822 /WINDOWS/DESKTOP/~\$dat.cab (_\$DAT.CAB) (deleted)
	54 m.. -/r-xr-xr-x 0	0	14178822 /WINDOWS/DESKTOP/~\$dat.cab (_\$DAT.CAB) (deleted)
Wed Mar 27 2002 13:02:22	341 ..c -/rwxrwxrwx 0	0	14605329 /WINDOWS/RECENT/dat.cab.lnk (DATCAB~1.LNK)
Wed Mar 27 2002 13:02:24	341 m.. -/rwxrwxrwx 0	0	14605329 /WINDOWS/RECENT/dat.cab.lnk (DATCAB~1.LNK)

This command found the two dat.cab files that have already been investigated and the symbolic links that were discovered in the mac_daddy output (those already discovered files are bolded). The other discovered files need to be investigated for more information. The Files that are called "Copy of dat.cab" should also be compared to the original files to make sure they are the same.

The grep command was used to look for win323.exe and also to discover if any other files were modified or created at or about the same time as the win323.exe file:

grep win323.exe timeline.out

Thu Dec 20 2001 15:03:32	522 m.. -rwxrwxrwx 0	0	28473672 <hda1.img-_2202001.LOG-dead-28473672>
Fri Dec 21 2001 18:31:16	28672 m.c -/rwxrwxrwx 0	0	14178844 /WINDOWS/DESKTOP/win323.exe (WIN323.EXE)
Fri Dec 21 2001 19:25:56	889 m.c -/rwxrwxrwx 0	0	20544582 /RECYCLED/_C68.TXT (deleted)
	889 m.c -rwxrwxrwx 0	0	20544582 <hda1.img-_C68.TXT-dead-20544582>

The win323.exe file and the C68.TXT file were the only files impacted all day on Dec 21. Here is another output from the grep command

Tue May 13 2003 00:00:00	28672 .a. -/rwxrwxrwx 0	0	14178844 /WINDOWS/DESKTOP/win323.exe (WIN323.EXE)
--------------------------	-------------------------	---	---

This information shows that the win323 file was accessed on May 13, 2003. There were several other files accessed at this time mostly in the windows directory this is possibly when this system was rebooted by the desktop technicians prior to noticing the keylogger application.

String Searches

Several string searches and file searches were conducted during the investigation. Searches were conducted over binary files including the win323.exe file, the netscape.hst file, fat.db, index.dat, the MACtime output, and the logical image of the target system. The following patterns were searched for:

- Keylogger
- Camouflage
- win323.exe
- ik97v12s.exe
- dat.cab

File Recovery

At this point in the investigation several files that have been deleted from the original image need to be recovered. In order to recover the files the `icat` utility from the Sleuthkit will be used. In order to recover the file the inode of the file must be provided. The inode information can be gathered from the MACTIME analysis or by doing a `ls -li` on the filesystem.

An inode is a data structure holding information about files in a Unix file system. There is an inode for each file and each file is uniquely identified by the file system by the inode. Deleted files can be recovered from the filesystem because often when a file is deleted it really is not destroyed. The pointer to the file is removed but the data that was contained at that disk location is potentially not overwritten. However this hidden data can be overwritten by the operating system. Each operating system writes data to the filesystem differently some operating systems will quickly reuse these deleted pointers and will overwrite the hidden data.

All of the deleted `dat.cab` files and the `C68.txt` file should be recovered from the image. The first file to recover is the `_C68.txt` file that was deleted on the same day that the `win323.exe` was installed on the system.

`#icat -f fat32 hda1.img 20544582 > C68.txt`

This is a beta release v0.9 of a stealth keylogger.

If for any reason you are unable to get this program to start when windows starts simply create a shortcut into the windows startup folder or any other way you can think of.

When first run on the remote computer it should automatically create the required reg entries to run at startup BUT i have not been able to test this fully. It should be automatic from then on.

Before running the program it must be in the folder its going to stay in. E.G the main windows directory.(C:\windows) and the file must be named whatever you want it to be. This is to ensure the reg entries are correct.

The log file created by this program is called `dat.cab` and is located in the same directory as the program executable.

thats it

enjoy:)

recovered and saved as 85.inode and the md5sum of the file was created. The md5sum was a04574fcd64703fc1f4eb94fffde4452 which was identical to the dat.cab located at C:\.

The inode located at 14178842 was recovered and investigated using the following commands:

```
#icat -f fat32 /root/forensics/hda1.img 14178842 > 14178842.inode
#strings -a 14178842.inode
```

```
1.0.1                                15   30722   116As
pect WinSet

                2.2
15   30723   1861CustomView Director

                2.1
                15   30724   119Asymetrix DVP Capture

                4.0
                15   30725   119Digital Video Producer

4.0                                15   30726   124IRMA for
```

This file does not appear to be a keylogger data file. However it was kept as evidence, the md5sum for the file is: 90b9129ab8b03addc5f8c21d44d8fa33.

The last of this series of inodes to recover is inode 14178851. The commands used to recover and investigate this file are:

```
#icat -f fat32 /root/forensics/hda1.img 14178851 > 14178851.inode
#strings -a 14178851.inode
```

```
"p"
"ci
"
"g"
"debu
"
""
"/"
"/"
"/ordpa...Bksp...d...Bksp...
"
""
```

```
"k(Cant Undo)sysconfigex...Bksp...e"
"x"
"."
"
"
""
""
""
""
""
"keylogger"
```

This file appeared very much like the first keylogger dat.cab file located at C:\Windows\Desktop, but it was much smaller (1989 Lines and 8192 Characters). It is possibly a subset of that first file. The md5sum for this recovered file and probable third different dat.cab file is:
faff53f70e4287cad19239a9b0adfb60.

Now that deleted dat.cab files have been recovered what other files have been deleted from the system that can be recovered. To find out what had been deleted the grep command will be used to create a list of deleted files.

#grep deleted timeline.out > deletedfiles.txt

This produced a text document with 2626 entries. I noticed that several entries from C:\Program Files\Netscape\Users\user_name were deleted on Fri Feb 15 2002 06:43:16. This could possibly be the user cleaning his cache directory. There are also several tmp files being removed from the recycle directory. There was other cab files listed in the deleted file list:

```
Fri Feb 22 2002 09:09:02 135744 m.. -/rwxrwxrwx 0 0 14178853
/WINDOWS/DESKTOP/datorig.cab (_ATORIG.CAB) (deleted)
```

This file needs to be recovered and investigated to see if the data is the same or different from previous keystroke captures.

#icat -f fat32 /root/forensics/hda1.img 14178853 > 14178853.inode
#strings -a cabfile.txt

```
"///"
//;/
"/"
"。(Cant Undo)"
"implem...Bksp...3a...Bksp.....Bksp.....Bksp...s....."
"/"
"/"
```

```

"/"
"/"
"67"
"12949b16b716"
"/"
gustavo...Bksp....Bksp....Bksp....Bksp....Bksp....Bksp....Bksp....Bksp...
..Bksp...name"
name
"naa2001"
"hotmail.com"
"essigr14jessjean24"
...Bksp....Bksp....Bksp....Bksp....Bksp....Bksp....Bksp....Bksp....Bksp
....Bksp....Bksp....Bksp....Bksp....Bksp....jesjan88
"se"
"autocad24b716b716"
"Annivrsary T Om...Bksp....Bksp....Bksp....Bksp...In 2 ...Bksp....Bksp....Bk
sp...@ Days"
"sex"
"user name"
"..."
"AAm"
"da...Bksp...C"
"mollette"
"daollet...Bksp....Bksp....Bksp....Bksp....Bksp....Bksp..."
"(Cant Undo)lGuy...Bksp....Bksp...s"
"Hey ...Bksp...Af...Bksp....Bksp...ds...Bksp....Bksp...a...Bksp....Bksp...s t
his you name? l...Bksp...ts me name."
"/"
"/"
"/"
"/"
"/"
"/"

```

This data file was not the same as any previous dat.cab file. The md5sum of this the fourth different dat.cab file was: 9a5575bff76380f8ffdbecb5bcaf3ded. This appears to be the last data file to be recovered. However some other information was found in the deleted file listing:

```

1152 .a. -/rwxrwxrwx 0 0 11406931 /MYDOCU~1/camouflage.reg
(_AMOF~1.REG) (deleted)

```

This data was registry information for the camouflage program that was previously displayed. Perhaps the user was investigating the footprint the camouflage program was leaving on his system.

One other thing looking at the timeline.out file for the system is that there is a huge gap between File system modifications from April 2, 2002 until May 9, 2003.

Tue Apr 02 2002 07:37:18	3072 m..	0	0	17771813	/WINDOWS/TEMP/~DF5B3E.TMP
	3072 m..	0	0	17771814	/WINDOWS/TEMP/~DF5BAC.TMP
Fri May 09 2003 00:00:00	67072 .a.	0	0	1806504	/WINDOWS/SYSTEM/cabinet.dll
	1728 .a.	0	0	1805076	/WINDOWS/SYSTEM/NDSWAN16.DLL
	3072 .a.	0	0	17771822	/WINDOWS/TEMP/~DF68FC.TMP
	20480 .a.	0	0	1805410	/WINDOWS/SYSTEM/WOW32.DLL

This output makes it appear that the user stopped using the system in April and not in January. Why was the system not used for such a long period of time?

Conclusions

The forensics on this system proved that there was a keylogger application on this system that was installed on Dec 21, 2001. The application was installed in the registry and was started every time the system was rebooted. The keylogger application was not very well hidden (perhaps a trial or beta version) it was installed on the desktop and left a fingerprint in the process manager listed as "keylogger". The keylogger application captured system commands, email messages, potential passwords, and other information was captured by the keylogger and was archived on the system.

In addition web cache files were found on the system that showed someone using the system had been actively searching for information about keyloggers. Cache files on the system showed activity that occurred after the application had already been installed. Other information showed up in strings information on the drive and the cache files for this evidence was not recovered. In addition a keylogger readme file was recovered that was deleted on the day that the keylogger application was installed. The installation of the keylogger application does not appear to have occurred through a Trojan horse or any kind of virus.

The keylogger was installed while the computer was in the user's possession and data from the application was copied and deleted also while in the user's possession. Either the user installed the keylogger to monitor the activity on his computer (perhaps suspicious of co-workers) or a co-worker installed the keylogger on the system while the user was gone. Either scenario violates local security policies as well as Federal wiretaps laws.

The timeline shows that there was no file activity from April 2, 2002 until May 9, 2003. This concurs with the technician's story of the computer sitting for a long period of time with no activity.

The user also had several MP3 files located on the system that appeared to be in his mailbox. There may be questions about whether he created these files or

was receiving them through email. The local policy restricts access to common MP3 sites through our webproxy system. Because of this the user webproxy logs were pulled for a large period of time to determine his Internet activity.

In addition email logs were checked looking for evidence of two things:

1. Was the keylogger application sending data to an offsite email address. There was not any apparent configuration information in the registry for the keylogger, so it was difficult to obtain any potential addresses. So in order to be certain email records were checked to see if any potential data was sent offsite. It did not appear that this had occurred but the logs were given to the investigative authorities.
2. The second reason was to look for a potential source for the MP3 files on the user's system. These records were also given to the investigative authorities.

Other systems that were located is this physical location where checked for evidence of keyloggers or of any other tampering. Any other system that the user had access to was also inspected for any evidence. This user did not have administrative control over any other system. No other evidence was found.

Because of the potential for wiretapping this case was turned over to our investigative authorities. The following files were submitted to the investigators:

File name	Description	Md5sum
14178822.inode	Inode 14178822 recovered dat.cab.file	3d3702173fb194b12fc19ad46852991f
14178824.inode	Inode 14178824 recovered dat.cab file	3d3702173fb194b12fc19ad46852991f
83.inode	Inode 83 recovered dat.cab file	3d3702173fb194b12fc19ad46852991f
14178842.inode	Inode 14178842 recovered dat.cab file	90b9129ab8b03addc5f8c21d44d8fa33
14178851.inode	Inode 14178851 recovered dat.cab file	faff53f70e4287cad19239a9b0adfb60
dat.cab.1	The earliest created dat.cab file.	a04574fcd64703fc1f4eb94ffde4452

/dev/hda1	This is not a file this is the original harddrive checksum	149e3ad6212ab0f04bcaf7378b0dfe88
hda1.img	The created image file	149e3ad6212ab0f04bcaf7378b0dfe88
C68.txt	The recovered keylogger readme file	bba5a483579b6f2f067268f0cc9ed480
dat.cab2	The second created dat.cab file	7e9fba68448e0112f21e5b670efafc9f
deleted.out	A list of all the deleted files from the timeline.out file	4e15e91fbd2ee36f8a79f8171f8876ce
fls.out	Output from the fls command	0df7c39db6ed5d393601b27a2cb2b937
fsstat.out	Output of the fsstat command	e927903bd0bba412130a21329ee42793
ils.out	Output of the ils command	c54b5fe9481e4dbb00212a0dfd878d76
Keyloggerstrings.out	Output of the strings command on the hda1.img file looking for the keyword "keylogger"	45576f1b6661e1d2ffe12a6fdac5b870
mac_daddy.out	Output of the mac_daddy perl script	932a73dd85892422b37f949c2090f47a
Smallertime.out	Mactime output for period of 12/01/01 – 05/17/03	4948827681725ecab7c7441e948497fe
Strings_win323.exe	Strings on the win323.exe binary file	68e8f5a772e2d0b321f791da9308e32f
System.dat.txt	Text representation of the system.dat file provided by regutils	d41d8cd98f00b204e9800998ecf8427e

Timeline.out	MAC time output provided by TASK	f3b0b66e0929e1c3918a0b4c44a5089e
User.dat.txt	Text representation of the user.dat file provided by regutils	62042de394dfc30ac19e1dc299f5b973

© SANS Institute 2003, Author retains full rights.

Section 3

You are the system administrator for an Internet Service Provider that provides Internet access to paying customers. You receive a telephone call from a law enforcement officer who informs you that an account on your system was used to hack into a government computer. He asks you to verify the activity by reviewing your logs and determine if your logs reflect whether or not the activity was initiated there or from another upstream provider. You review your logs and can only determine a valid user account logged in via a dialup account during the period of the suspicious activity.

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

Because the Internet Service Provider (ISP) provides access to the public (in this case paying customers) the Electronic Communications Privacy Act governs the disclosure of information. The ISP may disclose subscriber information only under certain conditions:

- When the subscriber consents (banner or user agreement).
- To protect the provider's rights and property
- To the government if provider believes that an emergency causing danger or possible bodily harm could occur without disclosure.
- To any person other than a government entity.

Unless one of those exceptions are in place the information cannot be disclosed to the officer without a court order or subpoena. This would protect the ISP from potential civil liability.

According to doj website <http://www.cybercrime.gov/s&manual2002.htm>

*"Defendants will occasionally raise a Fourth Amendment challenge to the acquisition of account records and subscriber information held by Internet service providers using less process than a full search warrant. As discussed in a later chapter, the Electronic Communications Privacy Act permits the government to obtain transactional records with an "articulable facts" court order, and basic subscriber information with a subpoena. See 18 U.S.C. Â§Â§ 2701-2712."*⁵

The courts have ruled in several cases that account records maintained by an Internet Service Provider do not have a reasonable expectation of privacy. According to <http://www.cybercrime.gov/s&manual2002.htm>:

"These statutory procedures comply with the Fourth Amendment because customers of Internet service providers do not have a reasonable expectation of privacy in customer account records maintained by and for

the provider's business. See United States v. Hambrick, 55 F. Supp. 2d 504, 508 (W.D. Va. 1999), aff'd, 225 F.3d 656 (4th Cir. 2000) (unpublished opinion) (finding no Fourth Amendment protection for network account holder's basic subscriber information obtained from Internet service provider); United States v. Kennedy, 81 F. Supp. 2d 1103, 1110 (D. Kan. 2000) (same). This rule accords with prior cases considering the scope of Fourth Amendment protection in customer account records. See, e.g., United States v. Fregoso, 60 F.3d 1314, 1321 (8th Cir. 1995) (holding that a telephone company customer has no reasonable expectation of privacy in account information disclosed to the telephone company); In re Grand Jury Proceedings, 827 F.2d 301, 302-03 (8th Cir. 1987) (holding that customer account records maintained and held by Western Union are not entitled to Fourth Amendment protection).”⁵

The Electronics Communications Privacy Act dictates that law enforcement must answer three questions to determine the procedures required to obtain information from an Internet Service Provider.

1. Is the network provider a public or private provider? Can a normal person subscribe to this provider's service or does the provider only provide access for employees. In this case we have paying customers so we are a public provider.
2. What type of information is being sought, is it subscriber information, transaction information, or content of communications? In this case law enforcement is requesting subscriber information about a user account.
3. Finally is the ISP voluntarily providing the information or is law enforcement requesting the information? In this case we know law enforcement is actively requesting this information.

18 U.S.C. § 2703 offers five mechanisms that law enforcement can utilize to gather information from a source:

1. Subpoena
2. Subpoena with prior notice to subscriber or customer
3. § 2703(d) court order
4. § 2703(d) court order with prior notice to subscriber or customer
5. Search Warrant

In this case the law enforcement agent is requesting basic subscriber information, he would need at a minimum a subpoena to request this information. Any of the other mechanisms could also be used to request this information.

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

The officer can communicate to the Internet Service Provider to preserve data prior to receiving the court order or subpoena. The reason for allowing this is there is no consistency in how ISPs maintain logs. Each ISP may log different

information and may maintain the logs for different periods of time. Allowing the officer to request that the logs be preserved protects law enforcement from missing evidence between the periods of initial contact versus getting the court order for the records. To minimize the risk of losing evidence 18 U.S.C. § 2703(f) allows the government to request the ISP maintain logs.

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

*"In general, investigators should communicate with network service providers before issuing subpoenas or obtaining court orders that compel the providers to disclose information. Law enforcement officials who procure records under ECPA quickly learn the importance of communicating with network service providers. This is true because every network provider works differently. Some providers retain very complete records for a long period of time; others retain few records, or even none. Some providers can comply easily with law enforcement requests for information; others struggle to comply with even simple requests. These differences result from varied philosophies, resources, hardware and software among network service providers. Because of these differences, agents often will want to communicate with network providers to learn how the provider operates before obtaining a legal order that compels the provider to act."*⁵

There are no laws that require Internet Service Providers to maintain records for any period of time, however the officer under the ECPA can request the provider to maintain the current their current records.

"In general, no law regulates how long network service providers must retain account records in the United States. Some providers retain records for months, others for hours, and others not at all. As a practical matter, this means that evidence may be destroyed or lost before law enforcement can obtain the appropriate legal order compelling disclosure. For example, agents may learn of a child pornography case on Day 1, begin work on a search warrant on Day 2, obtain the warrant on Day 5, and then learn that the network service provider deleted the records in the ordinary course of business on Day 3. To minimize this risk, ECPA permits the government to direct providers to "freeze" stored records and communications pursuant to 18 U.S.C. Â§ 2703(f)." ⁵

The law enforcement officer can make the request either using the phone or preferably using a fax or email request in order to provide the ISP a paper record and ensures that the request was communicated clearly. There are two other facts to § 2703(f) letters first current logs can be maintained but this letter will not cover future records. The second fact is that not all providers can fully comply with these requests.

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

Because the logs are stored information and not real-time the Electronic Communications Privacy Act again governs the disclosure of information. Depending on the type of information in the logs the provider has exceptions to determine whether to disclose the information or to require a court order. If the logs contain content of communications the following exceptions can allow the ISP to disclose the logs to law enforcement:

- The disclosure was made with the consent of the originator or the recipient of the communication, § 2702(b)(3)
- Protection for provider assets, is the attacker a threat to the provider
- If the contents were inadvertently obtained by the service provider and it appears to pertain to a crime.
- An emergency or immediate danger or serious bodily injury requires immediate disclosure.
- Child protection and Sexual Predator Punishment Act of 1998 42 U.S.C § 13032 provides immediate disclosure, 18 U.S.C. § 2702(b)(6)(B)
- The disclosure is made to the intended recipient.

If the logs the officer is requesting is for content of communication then with consent from the government institution that was attacked under the first exception the logs could be provided to law enforcement. This would only be applicable to transaction records for this account to the government agency.

However it appears that the logs in question are access records and not communication content. These logs contain a different set of exceptions:

- When the subscriber consents (banner or user agreement).
- To protect the provider's rights and property
- To the government if provider believes that an emergency causing danger or possible bodily harm could occur without disclosure.
- To any person other than a government entity.

It does not appear that any of these exceptions apply in this case. The rules for receiving the logs are similar to the rules for question 1.

Recent amendments to 18 U.S.C. § 2703 (c) provided by the USA Patriot Act of 2001 allow law enforcement to use a subpoena to obtain the following additional information:

- Records of session times and durations
- Any assigned network addresses
- Remote IP Addresses that a customer connects to the provider

- Means and source of payment information; this information may be important in tracking an individual to a crime.
4. What other “investigative” activity are you permitted to conduct at this time?

Because of the potential risk the ISP should insure that all logging is activated and the log files should be secured and backed up to another media to prevent possible compromise of the log data.

The “provider exception” §2511(2)(a)(i) allows the provider to conduct reasonable monitoring to protect the providers “rights or property” However this right cannot be used to gather evidence of a crime not related to their own rights or property. This essentially means that the provider could monitor their systems more closely to monitor for any attacks on their own systems. However this exception does not permit providers to conduct unlimited monitoring.

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

“Providers investigating unauthorized use of their systems have broad authority to monitor and then disclose evidence of unauthorized use under §2511(2)(a)(i), but should attempt to tailor their monitoring and disclosure so as to minimize the interception and disclosure of private communications unrelated to the investigation.”⁵

However the provider exception does not authorize the ISP to monitor communications between the authorized user and the government location that was under attack. Law enforcement cannot request the ISP to perform additional monitoring.

5. 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?

If the user is an intruder and not an authorized customer of the ISP additional monitoring could be conducted possibly by the ISP or by law enforcement. Management and the legal counsel of the ISP should be consulted to determine what the company policy is regarding unauthorized access. One of the exceptions to the wiretap act is the “computer trespasser” exception 2511(2)(i) this states that a wiretap can be conducted if the suspect is not an authorized user of the system. Law enforcement could be used at this point to intercept communications to and from the suspect.

A “computer trespasser” cannot be a person known by the provider to have an existing contractual relationship for use of the system.

A second exception to the Wiretap Act could also be used. The §2511(2)(a)(i) “provider” exception allows the ISP to conduct reasonable monitoring to protect their systems from further damage. This is not a criminal investigator’s privilege. This means that law enforcement cannot use the “provider exception” to monitor a users activities. It is important to note that this still does not permit unlimited monitoring by the ISP or by law enforcement.

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

“Providers investigating unauthorized use of their systems have broad authority to monitor and then disclose evidence of unauthorized use under §2511(2)(a)(i), but should attempt to tailor their monitoring and disclosure so as to minimize the interception and disclosure of private communications unrelated to the investigation.”⁵

In addition since a server belonging to the ISP has been compromised the system should be analyzed to determine how the intruder gained unauthorized access to the system. This could involve creating a forensic image of the server to analyze.

© SANS Institute 2003, Author retains full rights.

References

Section I

- 1) Stevens, W. Richard. TCP/IP Illustrated, Volume 1. Reading: Addison Wesley Longman, Inc, 1994.
- 2) Cole, Eric. Hackers Beware. Indianapolis: New Riders, 2002. 601-602.
- 3) McClure, Stuart and Scambray, Joel and Kurtz, George. Hacking Exposed 4th edition. Berkeley: McGraw-Hill/Osborne, 2003. 565-566.
- 4) Krutz, Ronald L. and Vines, Russell Dean. The CISSP Prep Guide. New York: John Wiley and Sons, Inc, 2001.
- 5) Daemon9 route@infonexus.com, LOKI2 (the implementation). 01 September 1997. URL: <http://www.phrack.org/phrack/51/P51-06>.
- 6) Dittrich, David, "The stacheldraht distributed denial of service attack tool". 29 December 1999.
[URL:http://project.honeynet.org/papers/enemy/ddos.txt](http://project.honeynet.org/papers/enemy/ddos.txt).
- 7) Lee, Rob and Zeltzer, Lenny. System Forensics, Investigation and Response: Advanced Forensics. 2003. The SANS Institute.
- 8) Foundstone Inc. "Foundstone enterprise software and services." 2003. URL: <http://www.foundstone.com>.
- 9) Russinovich, Mark and Cogswell, Bryce. "Freeware Sysinternals". 30 July 2003. URL: <http://www.sysinternals.com>.
- 10) DataRescue. "IDA Pro Dissembler 4.51". 07 July 2003. URL: <http://www.datarescue.com>.

Section II

- 1) Kruse, Warren G and Heiser, Jay G. Computer Forensics Incident Response Essentials. Indianapolis: Addison-Wesley, 2002.
- 2) Mandia, Kevin and Proise, Chris. Incident Response Investigating Computer Crime. Berkeley: Osborne/MacGraw-Hill, 2001.
- 3) Caloyannides, Michael A. Computer Forensics and Privacy. Norwood: Artech House, Inc, 2001. 145-152.

- 4) Carrier, Brian. "The Sleuth Kit". 10 June 2003.
[URL:http://www.sleuthkit.org/sleuthkit/index.php](http://www.sleuthkit.org/sleuthkit/index.php).
- 5) @stake Inc. "Research tools: Network Utility Tools:" 2003.
[URL:http://www.atstake.com/research/tools/network_utilities/](http://www.atstake.com/research/tools/network_utilities/)
- 6) Sourceforge. "Project: DMZS-Biatchux Bootable CD Distro: Summary": 14 May 2003. [URL:http://www.sourceforge.net/projects/biatchux](http://www.sourceforge.net/projects/biatchux).
- 7) Rendell, Michael. "regutils – win9x registry and ini file manipulation for unix". No date. [URL:http://web.cs.mun.ca/~michael/regutils/](http://web.cs.mun.ca/~michael/regutils/).
- 8) StegoArchive.com. "Steganography Software". 2003.
[URL:http://www.jitc.com/stegoarchive/stego/software.html](http://www.jitc.com/stegoarchive/stego/software.html).
- 9) Lee, Rob and Zeltzer, Lenny. System Forensics, Investigation and Response: Advanced Forensics. 2003. The SANS Institute.
- 10) Green, John and Carrier, Brian. System Forensics, Investigation and Response: The Coroner's Toolkit, TASK, and Autopsy. 2003. The SANS Institute.
- 11) Lee, Rob. System Forensics, Investigation and Response: Windows 2000/XP Forensics. 2003. The SANS Institute.
- 12) Lee, Rob. System Forensics, Investigation and Response: Forensic and Investigative Essentials. 2003. The SANS Institute.
- 13) Green, John. System Forensics, Investigation and Response: Basic Forensic Principles Illustrated with Linux. 2003. The SANS Institute.
- 14) Soloman, David A., Russinovich, Mark E. Inside Microsoft Windows 2000 3rd edition. Redmond: Microsoft Press, 2000.

Section III

- 1) Mandia, Kevin and Proise, Chris. Incident Response Investigating Computer Crime. Berkeley: Osborne/MacGraw-Hill, 2001.
- 2) Krutz, Ronald L. and Vines, Russell Dean. The CISSP Prep Guide. New York: John Wiley and Sons, Inc, 2001.
- 3) US Department of Justice, "United States code annotated title 18. Crimes and criminal procedure". 19 May 2003.
[URL:http://www.cybercrime.gov/ECPA2701_2712.htm](http://www.cybercrime.gov/ECPA2701_2712.htm).

- 4) US Department of Justice, "Field Guidance on New Authorities that relate to computer crime and electronic evidence enacted in the USA Patriot Act of 2001". 05 November 2001.
[URL: http://www.cybercrime.gov/PatriotAct.htm](http://www.cybercrime.gov/PatriotAct.htm).
- 5) US Department of Justice, "Searching and Seizing Computers and Obtaining Electronic Evidence in Criminal Investigations". July 2002.
URL: <http://www.cybercrime.gov/s&smanual2002.htm>
- 6) Salgado, Richard P. System Forensics, Investigation and Response: Frameworks and Best Practices: Managerial and Legal Issues. 2003. The SANS Institute.

© SANS Institute 2003, Author retains full rights.