GIAC Certified Forensic Analyst (GCFA)

FAILED ATTEMPT
Practical Assignment
Version 1.3
FAILED ATTEMPT

Forensic Analysis with F.I.R.E.

David M. Zendzian

dmz at dmzs.com

May 30, 2003

NOTE: This practical did not receive a passing grade for GCFA certification. It is, according to GIAC testers:

"clear that you have grasped many of the key concepts from your SANS training and experience. Unfortunately, there are several areas that lacked enough content to pass"

Table of Contents

Abstract

This practical will examine an unknown malware binary as well as a known compromised system image utilizing Open Source tools The Sleuth Kit, Autopsy, strings, hexedit and many other system analysis tools; all of which are packaged together in a bootable CD that creates self-contained, secure & trusted Forensics & Incident Response Environment, F.I.R.E.

Through the course of various investigation techniques the validity of FIRE as an incident response environment will be proven by:

- Examination and identification of purpose of an unknown binary recovered from a compromised system
- Identify what can and can not be accomplished with the data provided with the data provided
- Examine possible legal ramifications for the people involved in the deployment of the unknown binary.
- Identify what other information could have been verified if other image and system information had been provided.

This practical will also go into detailed description and document the processes followed to setup and then utilize FIRE as a Forensic & Incident Response Environment. This is further demonstrated through the analysis of a known compromised system image provided by one of the Digital Forensics Research Workshops Honey Pot challenges.

Analyze Unknown Binary

For this analysis, I make the assumption that the compromised binary arrived through a download of a compress floppy image from a remote system operator. There is no prior chain of custody so all evidence regarding the state of the system the binary was recovered from and information on how the binary was identified & discovered is not available.

Given these circumstances, it will only be possible to determine outlying characteristics of the binary image:

- Possibly where it came from
- What the binary's purpose is
- It may be possible to identify when the system was compromised & the binary installed.
- If we are fortunate we may also be able to discover which user id facilitated the compromise of the system in question.

The analysis of the target binary was performed with F.I.R.E., the Forensics & Incident Response Environment. FIRE is a self-contained bootable CD that provides a forensically sound Incident Response examination Environment on any system with VMWare or a bootable CDROM.

FIRE is an Open Source, Sourceforge hosted project, with active forums and archives of current & previous images. More information and downloadable ISO's can be found at the project website: http://fire.dmzs.com.

Binary Details

For purposes of this analysis:

binary_v1.3.zip was acquired from the GIAC web site url: http://www.giac.org/gcfa/binary_v1.3.zip on May 30, 2003.

There was no information provided regarding the system that target2.exe was acquired from or who had acquired, created or posted the binary onto the GIAC web site.

The received binary appears to be a winzip compressed zip file of a binary win32 executable labeled target2.exe. Nothing else is known about this file.

Winzip only stores the file name and the last timestamp on the file. Since winzip does not store any other file system specifics there is not much more we can determine from the external characteristics without getting an image of the file system it came from.

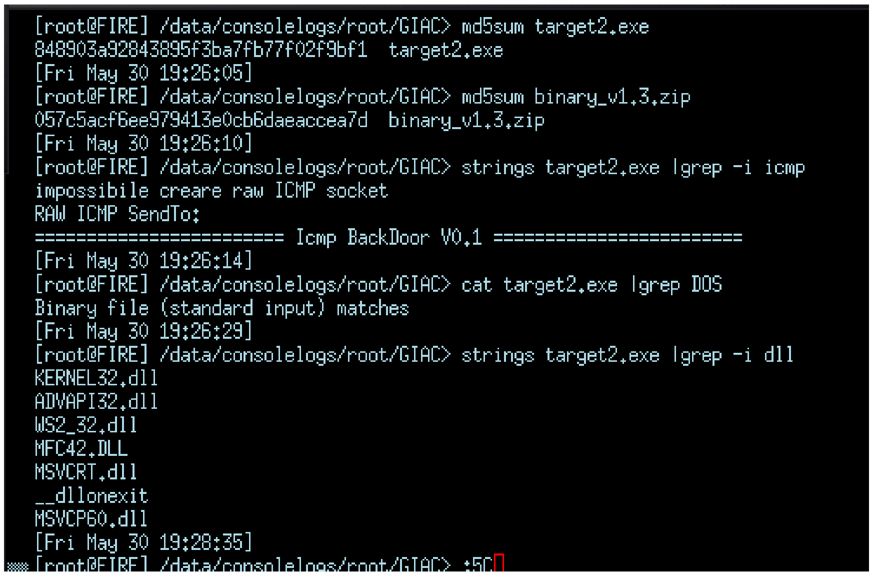
According to the information available from winzip, the file target2.exe was last modified on 2/20/2003 at 12:45 PM. The available CRC for the file within the ZIP is: D185FD18.

If the image was a combination of a tar of the file, a dd of the drive block table & sectors the file occupied along with the SAM file and system registry information, we could have gotten the user ID as well as more detailed time information such as when it was last Modified, Accessed or Changed (MAC) what registry functions it updated and much more information prior to having to examine the binary contents of the target file.

Switching over the GNU unzip, I extracted target2.exe with the –XX option. This option extracts the original owner information from the zip file. While this will extract the user information with the file it is extracted in a windows ID format. Since the archive did not contain the registry or SAM information from the compromised system I will be unable to determine the username of the user owning target2.exe.

By examining WinXP system permissions of the target2.exe extracted with GNU unzip –XX I was able to determine one user ID that was unknown to my local system. This userid is: S-1-5-21-1123561945-1708537768-1801674531-513.

To verify the validity of the archive I ran both md5sum on both the zip file & the extracted target2.exe. The winzip file binary_v1.3.zip provided is 5687 bytes and has a MD5 sum of 057c5acf6ee979413e0cb6daeaccea7d. The target2.exe within the zip is 5567 bytes within the archive and 26793 bytes once extracted. The MD5 sum of target2.exe is 848903a92843895f3ba7fb77f02f9bf1.



```
[root@FIRE] /data/consolelogs/root/GIAC> md5sum target2.exe
848903a92843895f3ba7fb77f02f9bf1  target2.exe
[Fri May 30 19:26:05]
[root@FIRE] /data/consolelogs/root/GIAC> md5sum binary_v1.3.zip
057c5acf6ee979413e0cb6daeaccea7d  binary_v1.3.zip
[Fri May 30 19:26:10]
[root@FIRE] /data/consolelogs/root/GIAC> strings target2.exe |grep -i icmp
impossibile creare raw ICMP socket
RAW ICMP SendTo:
====================== Icmp BackDoor V0.1 ======================
[Fri May 30 19:26:14]
[root@FIRE] /data/consolelogs/root/GIAC> cat target2.exe |grep DOS
Binary file (standard input) matches
[Fri May 30 19:26:29]
[root@FIRE] /data/consolelogs/root/GIAC> strings target2.exe |grep -i dll
KERNEL32.dll
ADVAPI32.dll
WS2_32.dll
MFC42.DLL
MSVCRT.dll
__dllonexit
MSVCP60.dll
[Fri May 30 19:28:35]
[root@FIRE] /data/consolelogs/root/GIAC> :5C
```

The archive and download site did not contain any MD5 sums of either of these file, so it was not possible to verify that these files had not been tampered with since their acquisition.

With no other external verifications that could be made on the target system, I proceeded to analyze the internals of target2.exe. Using the command 'strings', which parses an input file and outputs readable strings that are contained within the file much was discovered that assisted in the identification of the purpose of the binary.

While the order the strings come in may not be the order they are executed, many programmers sequentially develop their applications. So a first basic assumption can sometimes be made while analyzing strings output is that the order of the strings within the application is the order they were coded in. As such, possibly also the order the functions are executed in.

Browsing through the output, the first few usable strings seen deal with creating & starting services. This could be that the application creates a windows service if it doesn't exist, and then goes about starting it. It could have been part of an initial installation or just a function of the application to ensure it is setup to restart once the system reboots.

The next major item seen within the output is the creation of a RAW ICMP socket followed by:

```
=================== Icmp BackDoor V0.1 ========================
========= Code by Spoof. Enjoy Yourself!
 Your PassWord:
loki
cmd.exe
```

This is one of the most distinguishing items discovered in the analysis of target2.exe. If these strings are part of the application execution stream, it appears as if the program may be an ICMP back-door to a cmd.exe shell.

Forensic Details

To get a greater analysis of the application I utilized the local hex editor within FIRE, hexedit. While browsing target2.exe several interesting items were noted that were not identified in the strings output that would confirm the function of the application and possibly who was also involved in its creation and or distribution.

The initial look into target2.exe with hexedit reveal a name, 'Rich', in the beginning of the file. Having this name so close to the beginning of the file might reveal either part of the identify of the person who compiled the application, possibly who write the application to the file system or it might be the name of the person who created the zip file.

As I continued through the file the next large group of useful information showed me several functions, DLLs and registry entries that should be investigated on the hacked system to see if they were modified around the same time as when target2.exe was installed.

A few executable files are referenced later on near some SMB:
\winnt\system32\smesses.exe
\winnt\system32\reg.exe

I could not find any reference to smesses.exe through extensive internet searches with http://www.google.com - http://www.altavista.com - http://www.webcrawler.com - http://www.securityfocus.com - http://packetstormsecurity.com, so this may be part of the back-door or some other application installed to compromise other parts of the session. The reference to reg.exe leads me to conclude that the application is querying and possibly modifying registry entries on the compromised system.

Soon after the first SMB instance and right after the smesses.exe was the following IP address was identified. This IP Address could be either where the application is connecting to, or allowing connections from: 199.107.97.19

Several DLL files were also identified within the application. These files should be compared with the original DLL from another non-compromised windows machine and possibly investigated for additional back-doors or hidden system calls in case they are used for more than just basic system library calls:
KERNEL32.dll
ADVAPI32.dll
WS2_32.dll
MSVCRT.dll
MSVCP60.dll

Another interesting message discovered was something that wasn't picked up by strings since it had hex code 00 in between each character. While I knew that this character was the NULL character, I did not know why it was within the text until one evening while discussing various things with William Salusky, forger of FIRE. He mentioned that the string with NULL characters between each character was part of how Unicode strings were stored. He suggested that I use a Win32 version of strings that was compiled with Unicode support and it would have picked up this string in a strings output.

```
00005060   00 00 0F 00  48 00 65 00  6C 00 6C 00  6F 00 20 00  ....H.e.l.l.o. .
00005070   66 00 72 00  6F 00 6D 00  20 00 4D 00  46 00 43 00  f.r.o.m. .M.F.C.
```

Now knowing this new message within the application, what was it there for? This is most likely a message from the hacking team that built this version of loki, the people who hacked into the system & left this back-door. However since the application was compiled with windows strings (Unicode) support, it may have been compiled with other windows libraries such as the Microsoft Foundation Class (MFC) and this could be a message from that library class. However, since this application is most likely dynamically linked it is more likely that this message is from the group or individual that compiled this binary or wrote the backdoor.

Next I examined the binary with ‘objdump’ which allowed me to view library information about a binary executable. Using the –p option to print the object header information, including which library calls are made some of the application functions are revealed.

The first thing that was noticed was a date, maybe the date the binary was compiled: Wed Nov 27 23:53:13 2002. Following that, within the .rdata section of the object the various functions called from the linked DLLs are listed:

```
DLL Name: KERNEL32.dll
vma:  Hint/Ord Member-Name
32f0     409  HeapAlloc
3388     456  LocalAlloc
3378     282  GetLastError
336c     735  WriteFile
335e      67  CreatePipe
334c      68  CreateProcessA
333e      27  CloseHandle
332e     505  PeekNamedPipe
3322     536  ReadFile
330e     670  TerminateProcess
32fc     320  GetProcessHeap
32e8     662  Sleep

DLL Name: ADVAPI32.dll
vma:  Hint/Ord Member-Name
33d6     398  RegisterServiceCtrlHandlerA
33a4     435  StartServiceCtrlDispatcherA
33c2     430  SetServiceStatus
349e     336  QueryServiceConfigA
3486      45  ChangeServiceConfigA
3476     434  StartServiceA
3466     120  DeleteService
3454     325  OpenSCManagerA
3442      76  CreateServiceA
33f4      52  CloseServiceHandle
3432     327  OpenServiceA
341c     341  QueryServiceStatus
340a      53  ControlService

DLL Name: WS2_32.dll
```

```
vma:  Hint/Ord Member-Name
80000017          23
80000074         116
80000009           9
80000003           3
80000014          20
34ce       61  WSASocketA
80000039          57
80000034          52
34c2       37  WSAIoctl
80000011          17
8000006f         111
80000073         115
80000002           2
8000000b          11

DLL Name: MFC42.DLL
vma:  Hint/Ord Member-Name
8000032f         815
80000231         561
The Import Address Table is identical

DLL Name: MSVCRT.dll
vma:  Hint/Ord Member-Name
352a      709  strstr
3552       85  __dllonexit
3534      720  time
353c      670  printf
3614      183  _controlfp
3600      202  _except_handler3
35ee      129  __set_app_type
35e0      111  __p__fmode
35d0      106  __p__commode
34f2     664  memmove
34fc     585  exit
3504      600  fprintf
350e      275  _iob
3516      690  sprintf
3520      668  perror
356a      211  _exit
35c0      157  _adjust_fdiv
3572       72  _XcptFilter
3560      390  _onexit
35a0      271  _initterm
3590       88  __getmainargs
3591             3580      100  __p___initenv
```

```
3592          35ac    131  __setusermatherr

DLL Name: MSVCP60.dll
vma:  Hint/Ord Member-Name
3662     165  ??0_Winit@std@@QAE@XZ
3642     265  ??1Init@ios_base@std@@QAE@XZ
3622     158  ??0Init@ios_base@std@@QAE@XZ
367a     269  ??1_Winit@std@@QAE@XZ
```

These functions show that the application was doing several processes:

- The Kernel interface was dealing with pipes and handles so the application was talking to interfaces, processes or other applications.
- The ADVAPI showed that the application was doing something to the systems services.
- Next is seen Socket & IOCTL calls, so the application is definitely communicating with external applications through a socket.
- The next function group shows basic Terminal I/O communications through the standard MSVCRT library.

This clearly shows that the application is interfaces with the system services and communicates terminal services to multiple network connections.

One of the more interesting features of objdump is its ability to disassemble the application and show the application code in assembly as output. If there was the time, this is the best method of examining the application. However reading assembly is not a task to be done with a short deadline and I put off this final method of identifying every function of the application.

Having exhausted the methods of examination I could perform with objdump, I proceeded to utilize another application available under FIRE. One of the last tests I ran was to test the target2.exe file with a commercial grade virus scanner to determine what it would find. FIRE comes with the OpenSource version of F-Prot, http://www.f-prot.com. One simple menu item or "/usr/local/f-prot/update-defs.sh" from a command shell and FIRE will download the latest virus definition files.

Then I simply executed: "f-prot target2.exe".

However, F-Prot was not able to detect anything within this file, so although the test for malware is assisted by virus scanning, it did nothing to assist our binary identification on this test.

```
F-PROT 3.11b
SIGN.DEF created 30. May 2003
SIGN2.DEF created 30. May 2003
MACRO.DEF created 26. May 2003
```

```
Search: target2.exe
Action: Report only
Files: Attempt to identify files
Switches: <none>

Results of virus scanning:

Files: 1
MBRs: 0
Boot sectors: 0
Objects scanned: 1

Time: 0:00

No viruses or suspicious files/boot sectors were found.
```

Program Identification

All of this evidence clearly leads me to decide that this binary as an ICMP backdoor to cmd.exe. The default password may be “loki” or that may just be a reference back to the original loki ICMP backdoor implementation. This version was possibly originally coded by Spoof who may be part of a hacker group called MFC. However, it may not have been installed by spoof but by the local user Rich; or it may be downloadable through some hacker site..

Some basic investigation into loki & icmp through Google reveal that Loki is a UNIX based ICMP backdoor application that was originally presented in the August 1996 issue of Phrack by daemon9. Some more investigation through online archives such as http://packetstormsecurity.com reveal the code to the loki application for the *NIX environment.

The copy of loki2 for *NIX platforms can be found at:
http://packetstormsecurity.nl/crypt/misc/loki2.tar.gz

Looking though the code to loki2 there are several items missing that were found in the binary being investigated.

The first is that there is no banner “Icmp BackDoor V0.1”, mention of Spoof or MFC. So most likely the code for this windows version was based on loki2, but it clearly isn’t directly from this code.

Truly identifying non-refutable characteristics beyond the basics mentioned above would require additional information from the original host system. While the conclusions I have made thus far detail what is seen directly within the binary, there are so many interdependencies and related files & applications that were not included so a concise identification of this application is not completely possible without gathering some of this missing information.

Legal Implication

If this image was acquired on a system within the United States, and the activity occurred in such an environment where it was used without authorization then there would be a possibility of prosecuting the individual or individuals that installed and operated the application analyzed.

If such activity could be proved and it irrevocably proven that specific people were involved, then those people could be looking at from 1 to 10 years of prison and they could also be fined for the damages caused by their activity.

Without having a complete image of the system and no detailed documents regarding this case, only a binary image, it is impossible to determine several key items necessary to prosecute anyone in conjunction with the image provided.

There is no evidence regarding what system the binary was provided from. So it is unknown if this application was found on another web site, captured from an internet stream in download, attached to an email, sent through IRC or saved from an Instant Message. Not knowing any of these details leads to too many speculations as to the origin of the file making the file itself inadmissible in any legal proceedings.

There is no information regarding the case the image was acquired under, and no chain of custody with digital signatures of the images and logs as they were captured. This would add many questions regarding the procedures used in the discovery of the binary, the procedures used in the capture of the binary and the documentation detailing whom did the work and to whom it was transferred. With no digital verification of the binary image, there is no way to verify the integrity of the image and confirm that it hasn't been tampered with since it was captured.

With the information provided it would be impossible to prove anyone was guilty of a crime. Even if this information was regarding an attempt to dismiss an employee for violation of corporate policy regarding the use of non-authorized applications, there is still significant question regarding the origination of the tested image as well as the integrity of the image itself.

However, if a few questions could be answered then the analysis of this image could be documented in more detail providing irrefutable evidence that could lead to legal prosecution or termination of employment.

Interview Questions

With all that is unknown about the binary that was analyzed, there are several questions that need to be asked of the person who delivered the binary as well as from the person who identified the binary and the system administrator of these users and the compromised system.

1) Are you the administrator of the system the binary is from?
2) Do any other users have direct access to the system?
3) Are there any users or administrators that have recently (in the last year) left the organization that left under non-hospitable conditions?
4) What are the primary and "other" uses of the compromised system?
5) Is the system physically locked in a secure location?
6) Is the system physically locked closed? Keyboard/CDROM?
7) How was the binary discovered?
8) What time / date / location did the compromise occur?
9) Was anything done on the system prior to extracting binary?
10) Is there a complete system image of the compromised system?
11) Is there any image of the system memory?
12) What was the process utilized to acquire the binary?
13) Was the compromised system shut down or unplugged?
14) Is the compromised system still online?
15) Is there any network traffic captured that shows communication with the captured binary?
16) Is there a firewall in front of the compromised system?
17) If so, are there any logs identifying the attack that compromised the system or any traffic to/from the system once it was compromised
18) What are the firewall rules that allow the traffic to the server? Are there any new rules that opened this access? If so, who has access to change these rules?
19) Is there any IDS traffic or alert/alarm logs?
20) Do you have any additional information to provide?

Additional Information

There were many tools used in the identification of various pieces of the target2.exe binary.

First, F.I.R.E., the Forensics & Incident Response Environment was utilized as a base investigative environment. It provides a clean, logging environment that has tools and basic features built in to facilitate the incident investigator. Information on FIRE can be found at http://fire.dmzs.com.

The second most used resource in my investigation was the Internet. Specifically a few key search & security sites:

http://www.google.com – excellent searches of both web & newsgroups
http://www.webcrawler.com – another good place to find information
http://www.altavista.com – not used much more, but still good for news
http://packetstormsecurity.com – security news, docs & file archive
http://www.securityfocus.com – good security news research
http://www.phrack.org – source for loki write-up & loki2 code publish

also available at packetstorm

The final tool utilized in my in-depth analysis is the OpenSource version of F-Prot virus scanner. This tool allow for offline virus scanning of any medium by booting the machine from floppy or CD (FIRE). Information can be found at http://www.f-prot.com.

Perform Forensic Tool Validation

Scope

There are new products out every year to facilitate the acquisition and analysis of computer incidents. While there are many expensive commercial products available, I have chosen to use an Open Source tool called F.I.R.E., Forensics & Incident Response Environment, to do all of my data acquisition & analysis.

This entire paper has been an exercise in the use of FIRE for the examination of captured malware. This final section will examine how FIRE has integrated image analysis tools into it's configuration that facilitate the logging & analysis needed to provide strong evidence that can be utilized in a court of law.

I will be examining data from one of the Digital Forensics Research Workshop (http://www.dfrws.org/) images to briefly outline that features of FIRE, especially as it relates to the analysis of system images for data recovery & searching. The Image and information on the workshop can be found at http://www.honeynet.org/scans/scan24/.

While it is possible for FIRE to gather a significant amount of forensic evidence from images and most data sources, this analysis will only be reviewing the process and procedure of searching & recovering deleted files from the floppy image and leave the application of the other investigative processes to the reader and other documenters of FIRE.

The investigation into the details of the image will be performed through another Open Source application, Autopsy which is built upon the forensics tools from the Sleuth Kit, both of which are integral parts of the forensics capabilities included with FIRE.

Tool Description

F.I.R.E. is a bootable Linux CD that turns any machine into a forensics workstation. FIRE was created & is maintained by William Salusky (change at dmzs.com). The first release was available in early 2002. Soon following the initial release a Sourceforge project for FIRE was put together and released on February 12, 2002 under the project name biatchux. You can find the project Sourceforge page at http://www.sourceforge.net/projects/biatchux which contains information regarding project statistics that is not found at the main project page.

FIRE is an active project producing new versions every 1-2 months. The ISO has gone from 70M to 600M+ and has seen peek download months reach 50,000+. There have been over 120,000 downloads of FIRE to date (85,000 just this year)! More information about fire & the latest ISO used in this analysis can be found at the project homepage: http://fire.dmzs.com.

There are 2 quick ways of using FIRE:

- The ISO can be burnt it to a CD & boot from it on your workstation or laptop.
- The ISO can be booted from within VMWare, just mark the ISO as a CD-Image and boot away.

Since it boots an entire system without touching the local system, FIRE needs to run on a system with plenty of RAM to have enough room for the RAM Disk it creates and still leave some for the execution of X and the tools used for analysis. With such memory requirements, I would suggest using a system with 128M or more RAM. My personal forensic workstation has 512M, giving me some extra space to work with a few small images directly in my RAM Disk allowing for very fast analysis of these small images.

While I haven't utilized it, there is an option within FIRE where it will check the local floppy for user customized configuration information. This could be used to pre-configure the session for the data being investigated or to have common settings / menus / etc in place for the current operator.

One of the advantages of FIRE running within a RAM Disk is that it does not touch the system or images being analyzed allowing it to be a forensically clean & sound testing environment. It also logs as much information as is possible to the /data/ directory allowing for a single place to be archived when work is completed.

To facilitate the documentation of the startup process of my analysis on the GIAC & DFRW images, I booted FIRE into a VMWare virtual machine (http://www.vmware.com)[1]. This allowed me to capture screenshots of the FIRE boot-up & setup process.

Once I had the initial configuration documented I booted my standard forensics workstation with my FIRE CD, and then sent the console back to my windows VNC listener where I performed my analysis & captured the screenshots to include in this document.

[1] The current stable version of VMWare does not support the Video Frame Buffer FIRE uses for the local X display. If you are using fire & Want to work with the X-Console then you will need to send the display to a remote VNC listener. It is worth noting that the frame buffer within latest beta version of VMWare (4.0) supports the frame buffer F.I.R.E. utilizes, allowing for the local X display to work directly with VMWare. While I could have installed the Beta 4.0 VMWare and made my analysis easier, I continued utilizing a remote VNC console to demonstrate how a forensic analysis could be performed remotely allowing an Incident Response team to respond to global incidents more rapidly.

While I utilized my windows vnc listener to make the documentation of my processes easier, if I had chosen to use BSD or Linux as my forensic console workstation then I could have opened up the ability for more global remote analysis by utilizing the included utilities htc & hts. HTC & HTS create an HTTP tunnel for any protocol. Since HTTP is allowed through most firewall configurations, this would allow for a remote analyst to investigate a case from anywhere on the internet.

While the wrapping of VNC through htc/hts isn't very difficult, I leave it to the reader to experiment with the fun & excitement of tunneling your FIRE session out through a corporate proxy server to an awaiting forensic workstation across the internet.

The actual dirty-work performed in my forensic analysis was primarily performed with the Open Source tools Autopsy & The Sleuth Kit by Brian Carrier (brian at sleuthkit.org).

Autopsy, as defined at its website http://www.sleuthkit.org/autopsy/desc.php, "is a graphical interface to the command line digital forensic analysis tools in The Sleuth Kit. Together, The Sleuth Kit and Autopsy provide many of the same features as commercial digital forensics tools for the analysis of Windows and UNIX file systems (NTFS, FAT, FFS, EXT2FS, and EXT3FS). Autopsy contains the following features that allow for detailed information gathering of evidence on the analysis of your images:

- Case Management
- File Analysis
- File Content Analysis
- Hash Databases
- File Type
- Timeline of File Activity
- Keyword Search
- Meta Data Analysis"[i]
- Image Details
- Image Integrity
- Notes
- Reports
- Logging
- Open Design
- Client Server Model

The FIRE ISO version I used for my analysis is v0.4a. Most tools, especially Autopsy have been upgraded to their latest version as of the date of the 0.4a release. However, like all active projects today the version changes regularly. Currently autopsy is up to version 1.7.1, while it is 1.7.0 on FIRE 0.4a. With all the regular updates to FIRE, I expect to see a new release in the next month that will include that and other product updates.

Test Apparatus

My test involved 1 forensics workstation and 1 notebook for my workstation display. The workstation is an Intel Celeron 1.7 GHz with 512M RAM and 2 IDE

Drives (10 & 20 Gig). The workstation will be running the FIRE instance and doing all of the 'work' of this job.

My display notebook is a Compaq Evo N400c with 192M ram running Windows XP and my vnc listener. The machines are connected through a 100/10 switching hub and each machine has 100M interface cards allowing maximum bandwidth throughput.

Environmental Conditions

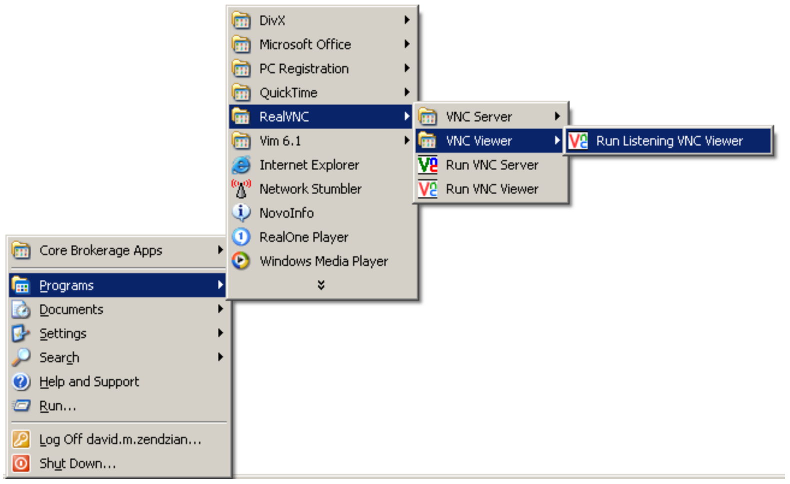
All of the testing was performed on the FIRE workstation. This contained all of the data and results preventing any loss or contamination from outside influences.

The FIRE console was setup to display through VNC onto my display workstation. This connection sent VNC over a 100M Ethernet connection. There was a slight amount of traffic on the test switch, but it wasn't enough to disrupt any of the VNC session and did not cause any issues on the results achieved or recorded since all results and tests were made within the FIRE environment.

The images to be tested were acquired via the net. The DFRW image included an MD5 to confirm the validity of the downloaded image. This was not available for the GIAC image so the validity of that image could not be confirmed.

Description of the Procedures

To prepare to receive the VNC FIRE session, I first started a VNC Viewer on my display workstation in "Listen Mode". Under windows, there should be an option for it under the RealVNC/VNC Viewer folder:



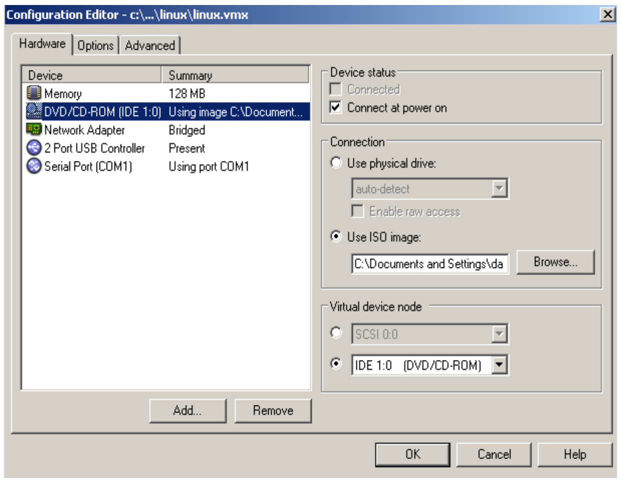
Once the Viewer was ready, I started VMWare Workstation with my latest image of FIRE set to be an IDE CDROM Image. Then, following the details you will see below, I booted FIRE and sent it's X console to the waiting VNC listener.

Once this initial configuration was documented, I turned off VMWare & booted my workstation with my FIRE CDROM sending the VNC console to my display workstation using the same process as described below and then began my analysis of the target images.

To function within VMWare, I configured a virtual machine with > 128M of RAM which is the minimum FIRE needs to be able to create the RAM Disks it needs and still have enough memory to perform all necessary operations.

Within the VMWare configuration I only configured 1 IDE disk that was set to be a CDROM with the "Use ISO image" option checked and my FIRE ISO Image selected.

My VMWare configuration screen is below:



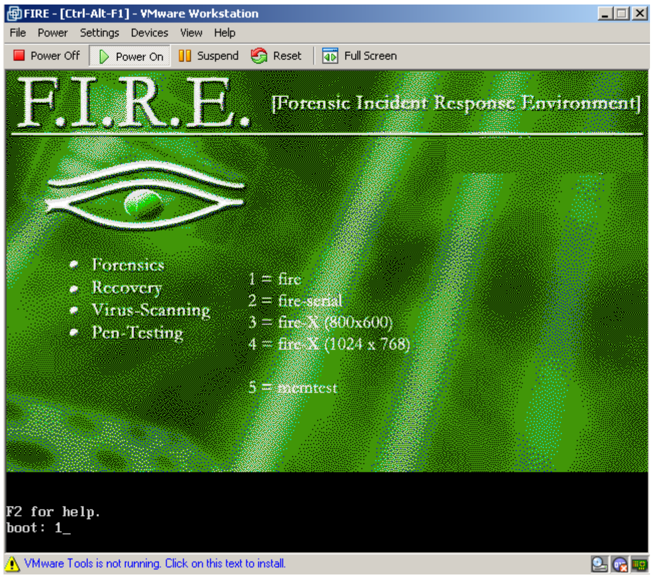
Device	Summary
Memory	128 MB
DVD/CD-ROM (IDE 1:0)	Using image C:\Document...
Network Adapter	Bridged
2 Port USB Controller	Present
Serial Port (COM1)	Using port COM1

Device status
[ ] Connected
[✓] Connect at power on

Connection
( ) Use physical drive:
auto-detect
[ ] Enable raw access
(●) Use ISO image:
C:\Documents and Settings\da

Virtual device node
( ) SCSI 0:0
(●) IDE 1:0 (DVD/CD-ROM)

Upon powering up, I chose to go with a standard FIRE session.



1 = fire
2 = fire-serial
3 = fire-X (800x600)
4 = fire-X (1024 x 768)

5 = memtest

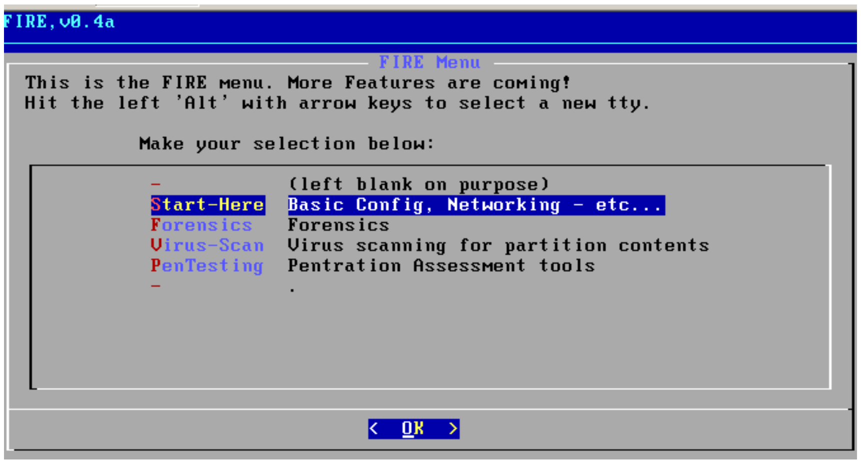
```
F2 for help.
boot: 1_
```

VMware Tools is not running. Click on this text to install.

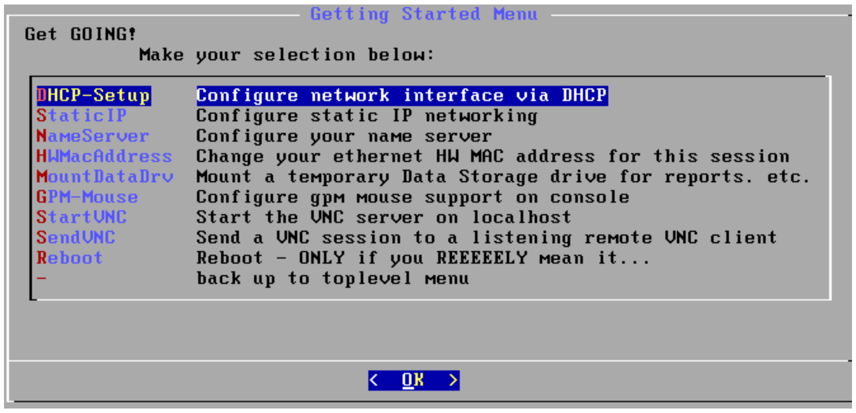
If this was a true incident response that required detailed logging, then I could have chosen to start with the console log to serial port & captured all output to another console that could be logged and used in the analysis report.

The serial console could also be used to facilitate remote analysis by connecting an auto-answer modem to the serial port and having the investigator dial into the awaiting serial console.

Once FIRE is running, the first thing to be done is to get it on the network. Since I am only testing a single binary I use the quickest method available, DHCP. Most of the initial workstation configuration options can be found under the "Start-Here" menu.



```
FIRE,v0.4a
                              FIRE Menu
 This is the FIRE menu. More Features are coming!
 Hit the left 'Alt' with arrow keys to select a new tty.

          Make your selection below:

             -          (left blank on purpose)
             Start-Here Basic Config, Networking - etc...
             Forensics  Forensics
             Virus-Scan Virus scanning for partition contents
             PenTesting Pentration Assessment tools
             -          .

                              < OK >
```



```
                       Getting Started Menu
 Get GOING!
           Make your selection below:

 DHCP-Setup    Configure network interface via DHCP
 StaticIP      Configure static IP networking
 NameServer    Configure your name server
 HWMacAddress  Change your ethernet HW MAC address for this session
 MountDataDrv  Mount a temporary Data Storage drive for reports. etc.
 GPM-Mouse     Configure gpm mouse support on console
 StartVNC      Start the VNC server on localhost
 SendVNC       Send a VNC session to a listening remote VNC client
 Reboot        Reboot - ONLY if you REEEEELY mean it...
 -             back up to toplevel menu

                              < OK >
```

As you can see DHCP isn't the only option. Going with a static IP or even spoofed MAC is possible if you are attempting to analyze a system or set of systems where you think that the attacker may be monitoring network traffic and you want to have your traffic appear as either something already there.

Normally if the FIRE VNC scripts were functioning properly, the next 2 steps would be to:

1) StartVNC
2) SendVNC

However one of these 2 is broken, so I utilize the following script to start the XFont Server/VNC Server & then send the VNC Server to the awaiting listener.

```
#!/bin/sh
# Start XFS, vnc server & then send to remote display ($1)

/usr/X11R6/bin/xfs -droppriv -daemon
/usr/bin/vncserver :0 -geometry 1024x768 -depth 16 -name Fire -rfbwait 50000 \
        -rfbauth /home/.vnc/passwd -desktop Fire -localhost
/usr/bin/vncconnect -display prometheus:0 $1
```

(note: pre v.0.4 the name was prometheus, 0.4 &> it is sol)

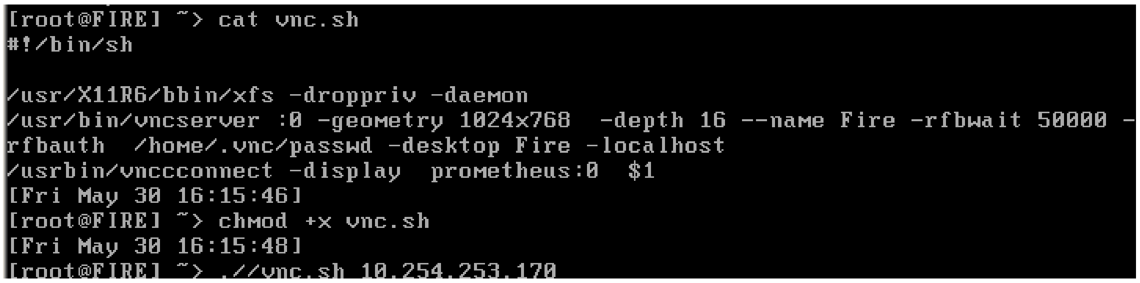
To get a shell prompt to input and execute this script either use ATL + left or right arrow keys or ALT+F[123456]. The root password is 'firefire'.

A plethora of unix tools that are not accessible through the Dialog or X menus can be found through the FIRE console shell. A partial listing of some tools is below. A more complete list can be found at http://fire.dmzs.com/?section=tools.

Name	**Description**	**License**
Autopsy v1.7.1	The Autopsy Forensic Browser is an HTML-based graphical interface to The Sleuth Kit and standard UNIX utilities. Autopsy automates many of the tasks required during a digital forensic analysis using the TASK collection of powerful command line tools as a foundation. Since this graphical interface is separate from the file system tools, an investigator can still use a command line interface if Autopsy cannot accomplish the desired outcome.	GNU General Public License (GPL)
bsed	binary stream editor	GNU General Public License (GPL)
chkrootkit v0.40	Chkrootkit is a tool to locally check for signs of a rootkit	chkrootkit license
CmosPwd v4.2	Cmos password recovery tools Works with the following BIOSes - ACER/IBM BIOS - AMI BIOS - AMI WinBIOS 2.5 - Award 4.5x/4.6x - Compaq (1992) - Compaq (New version) - IBM (PS/2, Activa, Thinkpad) - Packard Bell -	GNU General Public License (GPL)

	Phoenix 1.00.09.AC0 (1994), a486 1.03, 1.04, 1.10 A03, 4.05 rev 1.02.943, 4.06 rev 1.13.1107 - Phoenix 4 release 6 (User) - Gateway Solo - Phoenix 4.0 release 6 - Toshiba - Zenith AMI	
cryptcat	encryption enabled netcat	(GPL)
dcfldd - (or edd, enhanced dd)	the original dd tool enhanced with MD5 hashing built it. development work completed by DoD Computer Forensics lab.	GNU General Public License (GPL)

The final step in preparing my forensic workstation was to execute my VNC script & start working on the image to be analyzed.

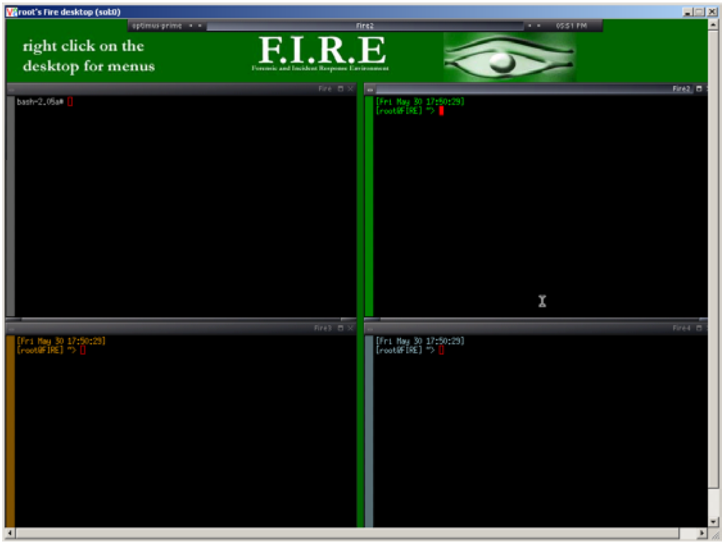


```
[root@FIRE] ~> cat vnc.sh
#!/bin/sh

/usr/X11R6/bbin/xfs -droppriv -daemon
/usr/bin/vncserver :0 -geometry 1024x768  -depth 16 --name Fire -rfbwait 50000 -
rfbauth  /home/.vnc/passwd -desktop Fire -localhost
/usrbin/vncconnect -display  prometheus:0  $1
[Fri May 30 16:15:46]
[root@FIRE] ~> chmod +x vnc.sh
[Fri May 30 16:15:48]
[root@FIRE] ~> .//vnc.sh 10.254.253.170
```

Once the script is executed it:

1) Starts an X Font server as a reduced priv daemon
2) Starts a 1024x768 16 color VNC Server named Fire & desktop Fire
3) Sends the display to the host specified on the command line

This sends the FIRE console VNC session to the listening viewer:


root's Fire desktop (sol:0)
optimus-prime
Fire2
05:51 PM
right click on the desktop for menus
F.I.R.E
Forensic and Incident Response Environment
Fire
Fire2
Fire3
Fire4

```
bash-2.05a#
```

```
[Fri May 30 17:50:29]
[root@FIRE] ~>
```

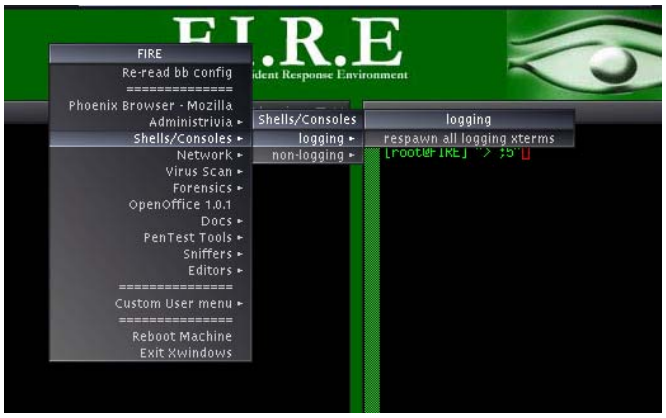
```
[Fri May 30 17:50:29]
[root@FIRE] ~>
```

```
[Fri May 30 17:50:29]
[root@FIRE] ~>
```

Now the binaries & images can be analyzed in a safe, secure environment.

A proper analysis needs to have the user's activity logged. This feature is provided through 'logging' FIRE console sessions with the UNIX 'script' commands.

To do the analysis with the proper auditing of the analysis, restart all the consoles with the sessions logged by right clicking on the desktop to get the menu then going to "Shells/Consoles->logging->respawn" all logging terms:



```
[root@FIRE] ~> ;5"
```

Be careful with this option. If you select it again you will lose your current terminal windows and whatever they were doing so that 4 new ones can be started up. The session data for those sessions is still in /data/consolelogs/, but whatever activity they were performing will be lost.

The terminals are all logged to /data/consolelogs/$user/$date-$tty.log. The timing output of script is also logged the same filename with a .timing extension on the end.

The logging of these terminals is one of the very few contributions I have made to the development of FIRE. The logging is accomplished through a simple wrapper script around the terminal application 'script.' This wrapper script is called 'consh[ii].'

This is a simple script that performs the following functions to record a user's session. The script performs the following functions:

- Trap all possible breaks a user may put on the script (CTRL-C, …)
- Load the default /etc/profile settings
- Set variable LOGDIR to /data/consolelogs/
- Test if the 1st argument is –c, if it is then he calling application is login or sshd (I use the same script to log interactive sessions on systems that users access in my networks).
- Depending on if $1 was –c, set a variable to the shell to execute (bash, …)
- If the USER variable isn't set, set with 'whoami' information

- Gather the current window's tty information & store in variable TTY. If the session has no TTY then use a default of nottylog as the directory to log to for the session
- Get timestamp & save in DATE variable
- Execute script logging to /data/consolelogs/USER/[TTYNOLOG]DATE-TTY script file and outputting 2> or stderr which is the script timing file to the same syntax file ended with .timing.

The application 'script' was not meant to be a security tool, and the user who's terminals are being logged can still modify the data that is archived in script's output, but it is a good starting point to document the process an investigator used in performing an analysis.

Also, some applications may have problems functioning 100% correctly if script & the application have problems interacting together due to various terminal characteristics. One such problem is an occasional screen glitch with vim on some formats of files.

FIRE also comes with the perl script 'replay' that goes with the script application. The replay script takes script & timing output files and replays them, keystroke by keystroke, back on the reporting console.

For example, to replay the script of my session on ttyp0 I would enter:

```
replay May30-182215-tty_ttyp0.log.timing May30-182215-tty_ttyp0.log
```

This would then replay, with all keystrokes, the session I ran with my ttyp0 terminal session.

If this had been an actual incident, then the incident responder could have used FIRE with dd to create the image[2] used in the analysis. This image could then have been stored on a local USB or Firewire drive or to a network SMB/Samba or NFS drive for archiving and future analysis purposes.

The DFRW image was downloaded from their website and they provided an MD5 sum of the image so that we were able to verify the integrity of the image from the time it was created.

```
[root@FIRE] ~/GIAC/honeynet> md5sum image.zip
b676147f63923e1f428131d59b1d6a72  image.zip
```

[2] Be sure to only do analysis on images of drives & data and when creating the image don't forget to turn on the MD5 flag so you can verify the integrity of your image at a future point.

If the image needs to be mounted to examine the contents, be sure to mount read only, no setuid no exec no dev and no atime so that the image integrity isn't compromised.

```
mount –o ro,loop,nosuid,noexec,nodev,noatime <dev> <mountpoint>
```

The rest of the analysis primarily uses TASK & Autopsy which result in browser output that will also be saved to the /data/ directory so it also can be archived and used as evidence and technical reminders for when presented in court.

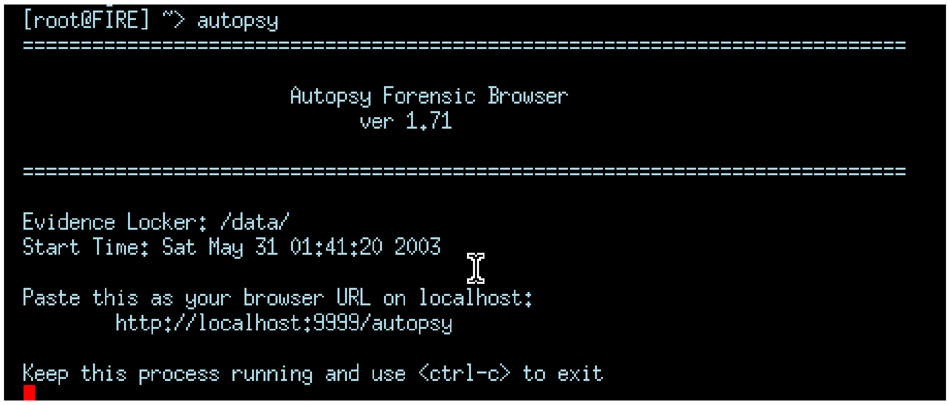
Once all of the command line analysis is completed, unmount the image and prepare to analyze with Autopsy. FIRE is primarily suited to the analysis of the local system it is running on top of. In fact if you select autopsy from the X Menu (Right click on desktop->Forensics->autopsy) the system will automatically scan the local machine and create the appropriate /data/fsmorgue file needed for autopsy to function.

This is not what I needed, so I manually created /data/fsmorgue and started autopsy so I could analyze the image.

My /data/fsmorgue image file contains only 1 line for the image I am reviewing:

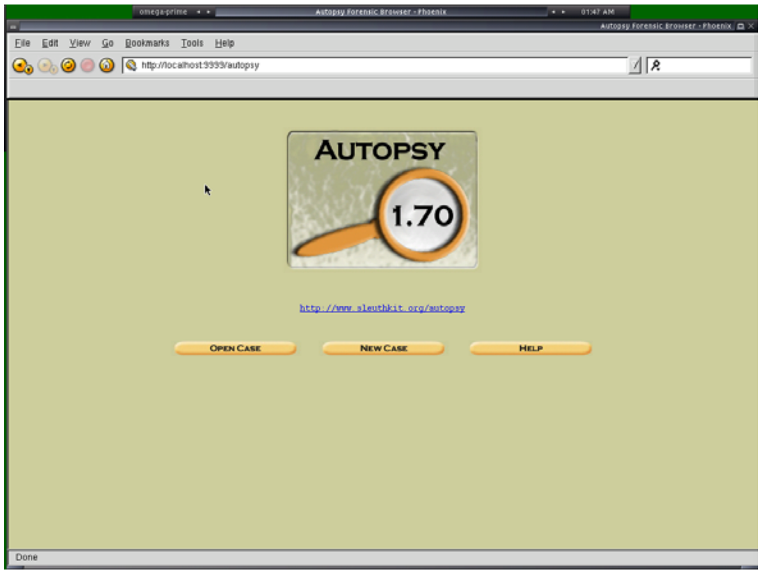
```
/home/root/GIAC/honeynet/image  vfat    /       PST
```

I then start autopsy from the command line



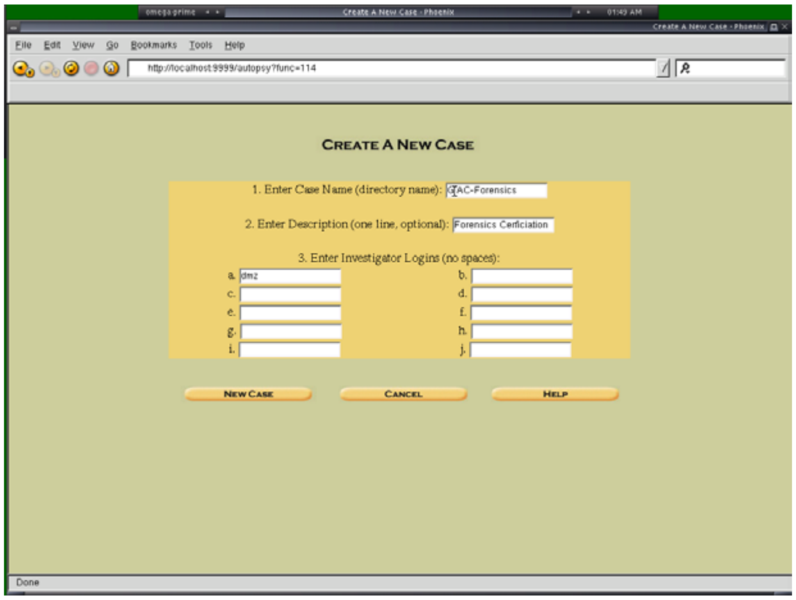
```
[root@FIRE] ~> autopsy
============================================================================

                       Autopsy Forensic Browser
                             ver 1.71

============================================================================

Evidence Locker: /data/
Start Time: Sat May 31 01:41:20 2003

Paste this as your browser URL on localhost:
        http://localhost:9999/autopsy

Keep this process running and use <ctrl-c> to exit
```

Once Autopsy is running, it provides a link to point a browser at to begin the image analysis. Now I start up a browser & go to the link that is displayed in my autopsy startup.


Autopsy Forensic Browser - Phoenix
http://localhost:9999/autopsy
AUTOPSY
1.70
http://www.sleuthkit.org/autopsy
OPEN CASE
NEW CASE
HELP
Done

Selecting “New Case” and fill in the new case information:

- Case Name: GIAC-Forensics
- Description: Forensics Certification
- Investigator: dmz (David M. Zendzian)


Create A New Case - Phoenix
http://localhost:9999/autopsy?func=114
NEW CASE
CANCEL
HELP
Done

CREATE A NEW CASE

1. Enter Case Name (directory name): GIAC-Forensics

2. Enter Description (one line, optional): Forensics Cerficiation

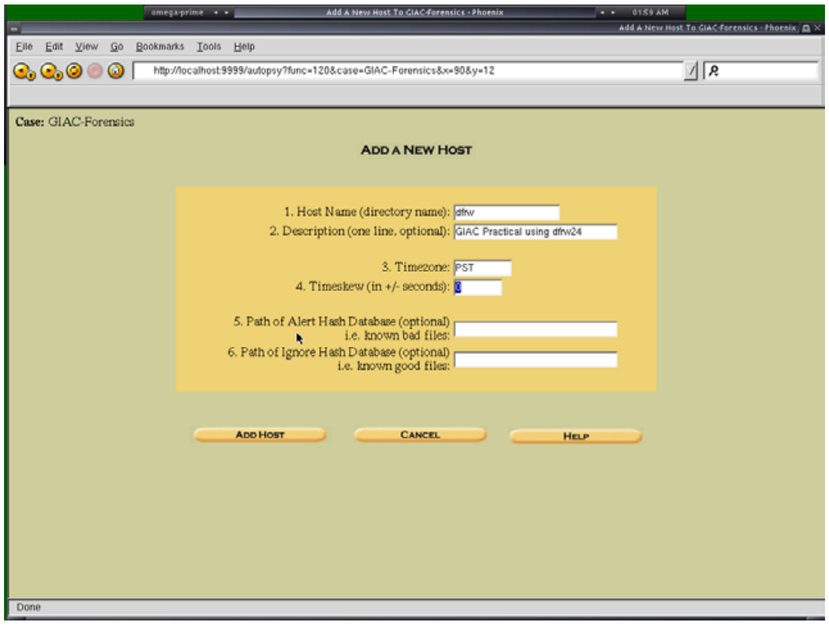
3. Enter Investigator Logins (no spaces):

a. dmz

Once New Case is selected, the data from this session is stored in the autopsy evidence locker which is /data/<CASE-NAME> (GIAC-Forensics in this CASE). The Case Gallery should now be opened with only 1 case in the gallery and that being the new case that was created.

Having selected that case I click on OK which brings me to the Host Gallery. Since I am starting from scratch there should be no hosts defined. I select ‘Add Host’ and fill in the template as shown below.

- Host Name: dfrw
- Description: GIAC Practical using dfrw24
- Timezone: PST


omega-prime
Add A New Host To GIAC-Forensics - Phoenix
01:59 AM
File Edit View Go Bookmarks Tools Help
Add Host
Cancel
Help
Done

http://localhost:9999/autopsy?func=12&case=GIAC-Forensics&x=90&y=12

Case: GIAC-Forensics

ADD A NEW HOST

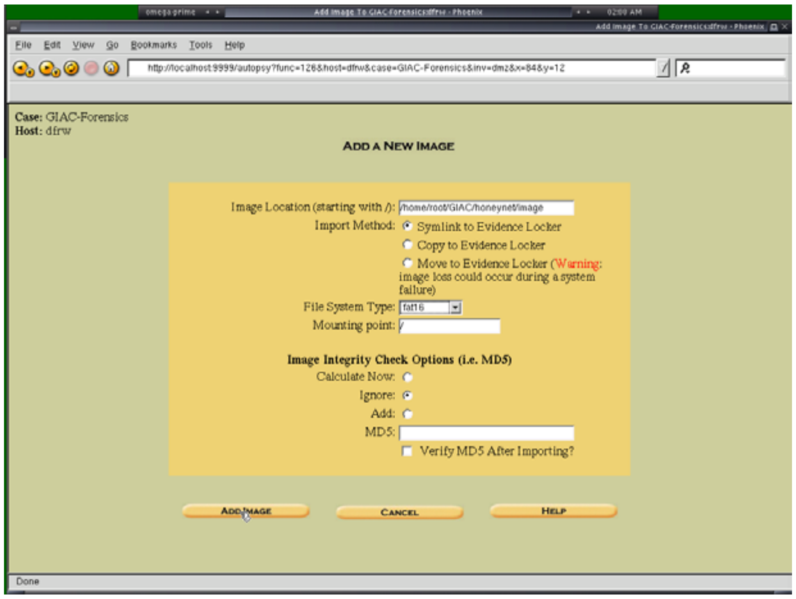
1. Host Name (directory name): dfrw
2. Description (one line, optional): GIAC Practical using dfrw24
3. Timezone: PST
4. Timeskew (in +/- seconds): 0
5. Path of Alert Hash Database (optional) i.e. known bad files:
6. Path of Ignore Hash Database (optional) i.e. known good files:

Now that the Case & Host are defined only the Images need to be added and then the analysis of the image can begin. Just select "Add Image" and fill in the information for each image to be reviewed.

After getting this far into the configuration, I discover that either the current version of FIRE or the version of autopsy I am using does not use the fsmorgue file I had created. However, to keep with standard practices I still made sure that the /data/fsmorgue was as I thought it should be thus continuing the established procedure I have for utilizing FIRE.

The Create Image screen for my simple test:
- Image Location: /home/root/GIAC/honeynet/image
- Input Method: I chose Symlink to conserve my RAM Disk space
- File System Type: fat16

I ran into an error in this release of Autopsy, which may be fixed in 1.7.1, but when I attempted to MD5 test the input image I kept getting errors. So for the purposes of this test I selected the Ignore option for MD5 & continued on as I verified the MD5 of the image manually in a previous step.


Add Image To GIAC-Forensics:dfrw - Phoenix
File Edit View Go Bookmarks Tools Help
Add Image
Cancel
Help
Done

http://localhost:9999/autopsy?func=12&host=dfrw&case=GIAC-Forensics&inv=dmz&x=84&y=12

Case: GIAC-Forensics
Host: dfrw

ADD A NEW IMAGE

Image Location (starting with /): /home/root/GIAC/honeynet/image
Import Method: [✓ Symlink to Evidence Locker] Copy to Evidence Locker Move to Evidence Locker (Warning: image loss could occur during a system failure)
File System Type: fat16
Mounting point: /

Image Integrity Check Options (i.e. MD5)
Calculate Now: [✓ Ignore:] Add:
MD5:
Verify MD5 After Importing?

Once you select Add Image Autopsy will return you back to the Image list where you can either add a new image or work with the already defined images. For my tests I selected the / image just created & click OK to bring up the main analysis window.

The analysis choices available are:

- File Analysis where you can browse the various files available on the image, including deleted files that you are able to recover.
- Keyword search where you can search the image for various keywords
- File Type that runs the application sorter which counts the various file types on the image and gives a total of each type
- Image Details which contains summary data about the image being analyzed
- Meta Data is a search feature that allows you to enter a meta data number & autopsy will gather and display the details for the selected item
- Data Unit is the final option, and it allows for the entry of a sector number that will be retrieved and displayed in the browser window.

Autopsy & TASK are 2 OpenSource tools that by themselves can stand alone and do a majority of the work in a forensics investigation. Put together seamlessly with the other tools available on FIRE and the investigator has a stable, secure, auditable environment to produce non-disputable results that can be repeatedly created by any qualified forensics analyst.

Once all of the analysis is completed, my final procedure is to tar the /data/ & /home/root/ directorys to store offline, create MD5 of the tar & and ls –lR of the same directories & then store all information offline and with the incident response folder.

Criteria for approval

For the analysis to prove a success, the workstation must be able to identify any deleted data from the test image, provide for a method to recover the deleted file, search the image for various keywords and view the timeline of the file system to determine any additional clues as to what happened on the system.

The method that has already been gone through to setup the analysis is part of the Autopsy procedure, and allows for any user to proceed with a forensic analysis of image media with little confusion since the routine is fixed and very intuitive with its step-by-step wizard that sets up each case and image to be reviewed.

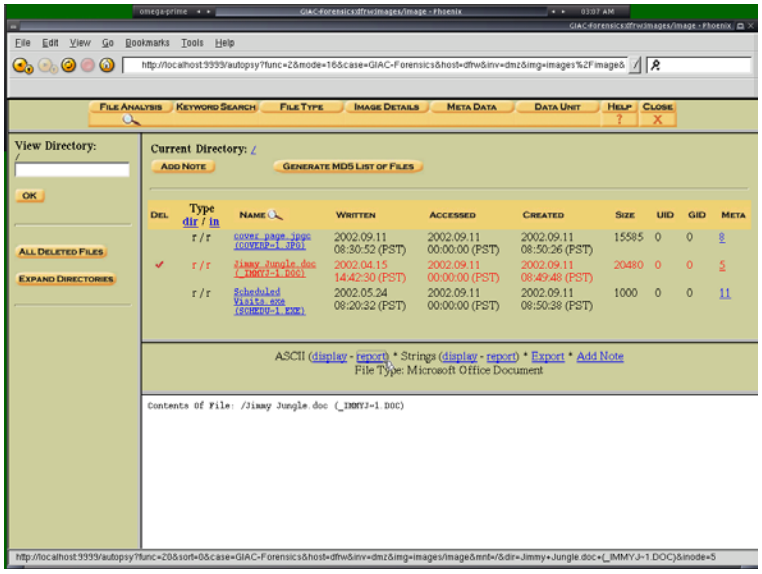
All of the data gathered from autopsy is stored in it's locker in /data/ allowing for all the information to be easily marked (MD5) & saved for verification and review of what was identified.

Since all of this procedure is being run on an image and on top of the FIRE environment, there is nothing on either the running FIRE workstation or within the process of analysis that can alter the data being analyzed. The easiest way to verify this is to logon and do a 'mount' command. The only mounted systems should be the local RAM Disk, which is erased when the system reboots, and the CDROM itself.

If you are analyzing any of the local drives to the FIRE workstation, then the local drives may be mounted, or autopsy may just analyze the image directly. However that is not recommended and if those drives need to be analyzed then take a image and go through that image offline.

Data and Results

At the end of the procedures above everything was setup to run the autopsy tests on the test image. The first test is the File Analysis. Once this report is run it is easy to flag the RED file as one that was deleted and worth examining in greater detail.



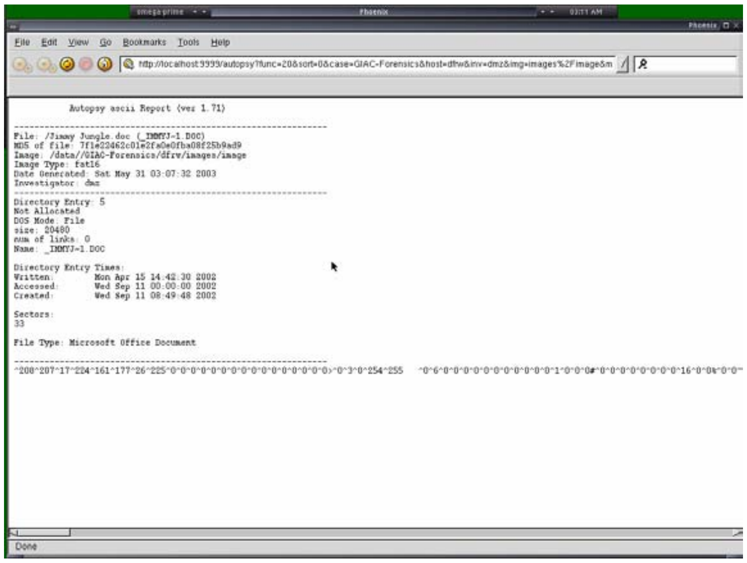
Current Directory: /

Del	Type dir / in	Name	Written	Accessed	Created	Size	UID	GID	Meta
	r / r	cover page.jpgc (COVERP~1.JPG)	2002.09.11 08:30:52 (PST)	2002.09.11 00:00:00 (PST)	2002.09.11 08:50:26 (PST)	15585	0	0	8
✓	r / r	Jimmy Jungle.doc (_IMMYJ~1.DOC)	2002.04.15 14:42:30 (PST)	2002.09.11 00:00:00 (PST)	2002.09.11 08:49:48 (PST)	20480	0	0	5
	r / r	Scheduled Visits.exe (SCHEDU~1.EXE)	2002.05.24 08:20:32 (PST)	2002.09.11 00:00:00 (PST)	2002.09.11 08:50:38 (PST)	1000	0	0	11

ASCII (display - report) * Strings (display - report) * Export * Add Note
File Type: Microsoft Office Document

```
Contents Of File: /Jimmy Jungle.doc (_IMMYJ~1.DOC)
```

If the report is run on the deleted file then a good amount of information can be gathered about the file, including MAC times, MD5, Size, Name, Type and sector information.



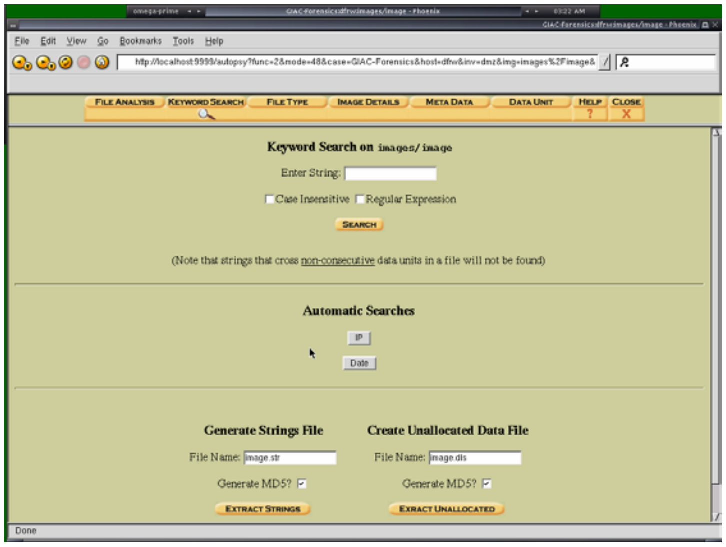
```
          Autopsy ascii Report (ver 1.71)

---------------------------------------------------------------
File: /Jimmy Jungle.doc (_IMMYJ~1.DOC)
MD5 of file: 7f1e22462c01e2fa0e0fba08f25b9ad9
Image: /data//GIAC-Forensics/dfrw/images/image
Image Type: fat16
Date Generated: Sat May 31 03:07:32 2003
Investigator: dmz
---------------------------------------------------------------
Directory Entry: 5
Not Allocated
DOS Mode: File
size: 20480
num of links: 0
Name: _IMMYJ~1.DOC

Directory Entry Times:
Written:        Mon Apr 15 14:42:30 2002
Accessed:       Wed Sep 11 00:00:00 2002
Created:        Wed Sep 11 08:49:48 2002

Sectors:
33

File Type: Microsoft Office Document

---------------------------------------------------------------
^208^207^17^224^161^177^26^225^0^0^0^0^0^0^0^0^0^0^0^0^0^0^0^0>^0^3^0^254^255    ^0^6^0^0^0^0^0^0^0^0^0^0^1^0^0^0#^0^0^0^0^0^0^0^0^16^0^0%^0^0"
```

Now that a deleted file is identified it needs to be recovered so that the contents can be examined as part of the case. To recover the file, select it from the File Analysis window and choose 'export' from the bottom window.

Save the exported file into the /data/ system. Then utilizing openoffice, wordview, strings, hexedit and other tools examine the contents of the exported file.

The keyword search provided by autopsy searches the entire image, including the slack space that is only going to be used by deleted files and hackers attempting to hide files.

Selecting the Keyword Search from the menu brings up a very simple menu. Just enter the keyword to search for and select Search. There are 2 quick automatic searches available to search by for Dates & IP Addresses as well as 2 quick options to generate a strings file or an unallocated dataspace file and save to the local file system.



http://localhost:9999/autopsy?func=2&mode=48&case=GIAC-Forensics&host=dfrw&inv=dmz&img=images%2Fimage&

Keyword Search on images/image

Enter String:

Case Insensitive Regular Expression

(Note that strings that cross non-consecutive data units in a file will not be found)

Automatic Searches

Generate Strings File

File Name: image.str

Generate MD5? ✓

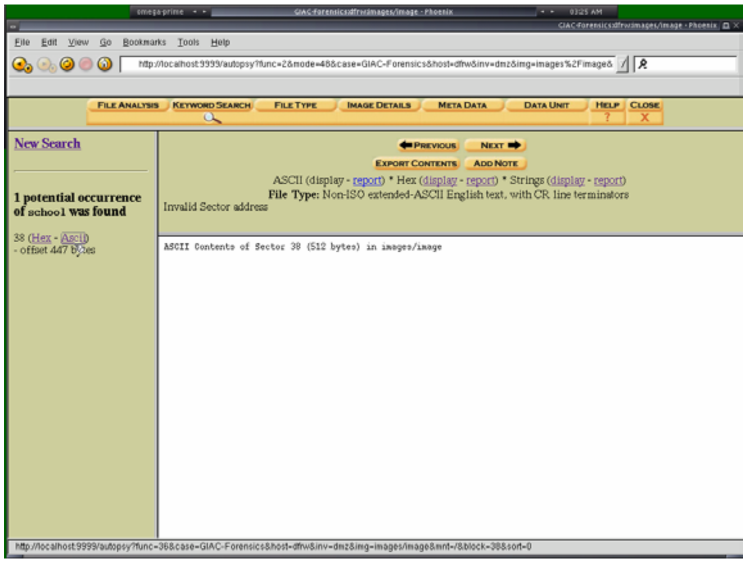
Create Unallocated Data File

File Name: image.dls

Generate MD5? ✓

The case that is being tested against was searching for collaborating evidence about a drug dealer selling on school grounds. A quick keyword search for “school” finds one potential occurance.

Selecting the Ascii or Hex contents of the found occurrence will display the details of that occurance on the right side. From there the information can be reported or exported out to the local file system.



http://localhost:9999/autopsy?func=2&mode=48&case=GIAC-Forensics&host=dfrw&inv=dmz&img=images%2Fimage&

1 potential occurrence of school was found

38 (Hex - Ascii)
- offset 447 bytes

ASCII (display - report) * Hex (display - report) * Strings (display - report)

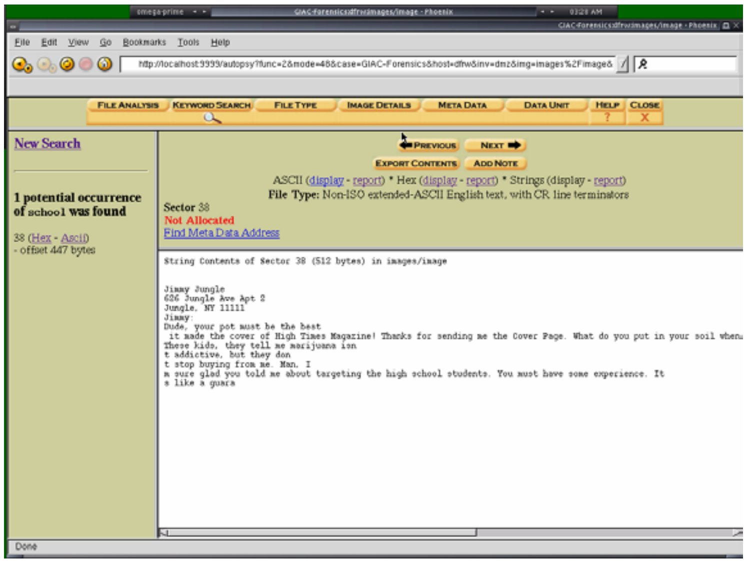
File Type: Non-ISO extended-ASCII English text, with CR line terminators

Invalid Sector address

```
ASCII Contents of Sector 38 (512 bytes) in images/image
```

http://localhost:9999/autopsy?func=36&case=GIAC-Forensics&host=dfrw&inv=dmz&img=images/image&mnt=/&block=38&sort=0

And then the data that was needed to verify that the offender did have data on the image the matched our keyword search as well as provided evidence that proves the story being investigated.



http://localhost:9999/autopsy?func=2&mode=4&&case=GIAC-Forensics&host=dfrw&inv=dmz&img=images%2Fimage&

1 potential occurrence of school was found

38 (Hex - Ascii) - offset 447 bytes

ASCII (display - report) * Hex (display - report) * Strings (display - report)
File Type: Non-ISO extended-ASCII English text, with CR line terminators

Sector 38
Not Allocated
Find Meta Data Address

```
String Contents of Sector 38 (512 bytes) in images/image


Jimmy Jungle
626 Jungle Ave Apt 2
Jungle, NY 11111
Jimmy:
Dude, your pot must be the best
 it made the cover of High Times Magazine! Thanks for sending me the Cover Page. What do you put in your soil when
These kids, they tell me marijuana isn
t addictive, but they don
t stop buying from me. Man, I
m sure glad you told me about targeting the high school students. You must have some experience. It
s like a guara
```

Done

The final step in the validation of FIRE & Autopsy is the analysis of the image showing the file activity timeline. This information can be used to verify the creation, deletion and modification of certain evidence files.

To get to the File Activity Time Lines, the current window needs to be closed, so click on the “Close” or “X” button in the upper right.

Once the screen is back to the image selection menu, click on the File Activity Time Lines button. This will bring up a new screen with the following options:

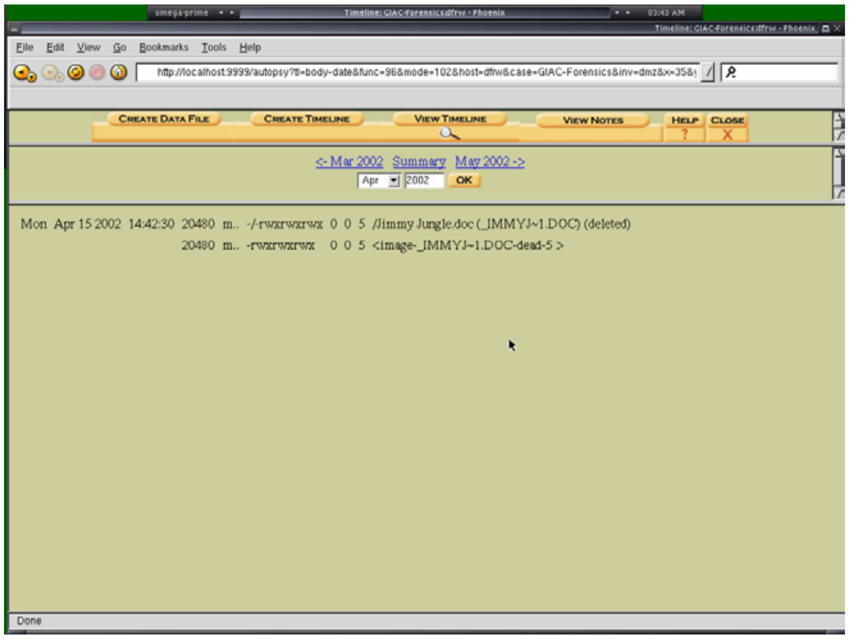
- Create Data File
- Create Timeline
- View Timeline
- View Notes

Each option does pretty much what it states. First the Data File needs to be created, so that option is selected and the image we are testing is selected. The output name is entered by default as body, this is OK so OK is selected. This brings up the Create Timeline menu.

The nice thing about how Autopsy has put these screens together is how each image datafile can be created with separate unique data timelines for each search and each search is saved with it’s MD5 for review & verification. at any future point.

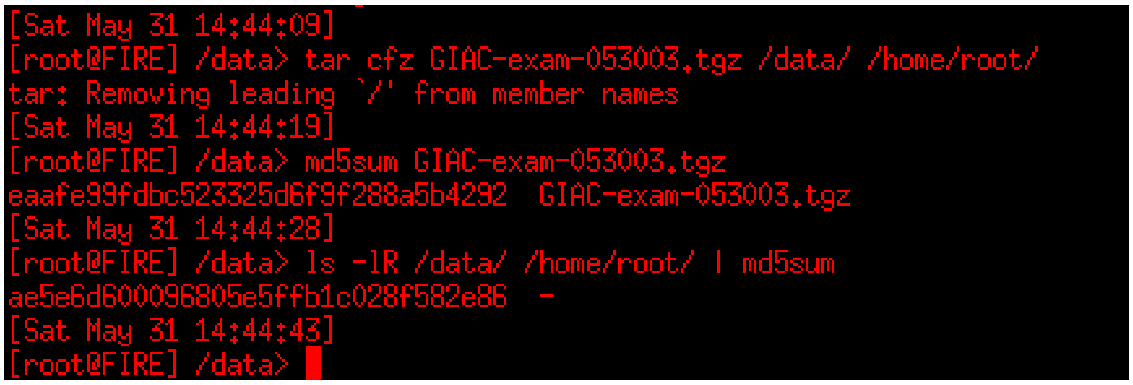
Once the date range is selected & OK is clicked on, autopsy will generate the file timeline and display in a frame within the browser window. This result, along with all other results can be found in the /data/ directory if viewing them through autopsy browser isn't an easily accessible or other search options are needed to fully review the output files.

Below are the results of the timeline for this simple search through the image for this test:


omega-prime
Timeline: GIAC-Forensics:dfrw - Phoenix
03:43 AM
File Edit View Go Bookmarks Tools Help
http://localhost:9999/autopsy?tl=body-date&func=96&mode=102&host=dfrw&case=GIAC-Forensics&inv=dmz&x=35&
CREATE DATA FILE
CREATE TIMELINE
VIEW TIMELINE
VIEW NOTES
HELP
CLOSE
<- Mar 2002 Summary May 2002 ->
Apr 2002 OK
Done

```
Mon  Apr 15 2002  14:42:30  20480  m..  -/-rwxrwxrwx  0  0  5  /Jimmy Jungle.doc (_IMMYJ~1.DOC) (deleted)
                            20480  m..  -rwxrwxrwx    0  0  5  <image-_IMMYJ~1.DOC-dead-5 >
```

Now that all of the information has been gathered, an archive of the autopsy report, along with all console logs and any other settings and files created for the analysis is archived, MD5 checked & stored with the case log and also in an external secure environment.



```
[Sat May 31 14:44:09]
[root@FIRE] /data> tar cfz GIAC-exam-053003.tgz /data/ /home/root/
tar: Removing leading `/' from member names
[Sat May 31 14:44:19]
[root@FIRE] /data> md5sum GIAC-exam-053003.tgz
eaafe99fdbc523325d6f9f288a5b4292  GIAC-exam-053003.tgz
[Sat May 31 14:44:28]
[root@FIRE] /data> ls -lR /data/ /home/root/ | md5sum
ae5e6d600096805e5ffb1c028f582e86  -
[Sat May 31 14:44:43]
[root@FIRE] /data> 
```

Analysis

The nice feature available through autopsy is the ability to save each investigation in its own case folder. This allows for either the investigator or some other party doing a review to both have access to the results of the investigation without having to step through the entire investigation process.

The reviewer has the option of reviewing the entire case through the text files created, or just start up autopsy again and use a browser to present this information to either law enforcement, management or any other entity needing to verify the results of the investigation.

If the results are viewed through the browser, then the reviewer can easily step through the various report and search menus available through autopsy to display the specifically recovered or identified files found in the investigation.

If the results are saved from the /data/CASE/ directory, where CASE is the name of the case being investigated (GIAC-Forensics in this example), then they can be imported into presentations or reports being submitted regarding this investigation.

Presentation

While FIRE & Autopsy present the results of an Image analysis in a clean browser window, this format may not be suitable for presentation in court or executive staff room.

In the course of a normal investigation involving many images, searches and reports, the data leading to the evidence needed in court may only be a fraction of all the data recovered.

The data saved from the various Autopsy reports are a combination of flat files & a few html files. While they are not exactly the most intuitive, if used in combination with the case session log & investigator notes a detailed report could be generated documenting the:

- Incident Timeline
- Activity Report (Surrounding incident)
- Recovered files specific to incident
- Details of files recovered
- Investigator notes regarding special items found in investigation
- MD5 check of all report and image data to verify that the items identified in this report match with the data created with the incident investigation.

If this report was being presented in a court of law or executive staff room, then most of the people receiving the report may not understand some of the details found in the timeline/activity report & details of the files & other recovered or identified items.

Anyone who would have to present this information to such groups should also be prepared to instruct the recipients on what some of the technical terms and techniques are. A few of the items that should be documented include:

- MAC Time
- Access Control (777/rwx/…)

- I-Node
- UID / GID
- Meta Data
- Sectors
- Allocation List
- MD5
- Slack Space or Unallocated Data

Conclusion

FIRE not only includes the current leading tools in Open Source Forensics, but it packages them together in such a way that provides for an instant environment to perform incident response as well as present that information to local or remote users.

Every test that needing to be done to search for, identify & verify data needed for a forensic investigation was readily available within the package FIRE provides.

The only missing items are features that the application developers already know about and are actively creating. Those features include cleaner presentation interfaces and more automation of the various standard routines that investigators commonly use while following an investigation.

Overall none of the missing or non-complete features causes any forensic investigator to not utilize these tools to perform their jobs, and with each new release more and more standard functions are automated allowing for rapid investigation utilizing a set of standardized techniques.

Additional Information

FIRE, Autopsy & TASK are actively developed Open Source projects. If there are features that they do not perform then anyone can contact the developers to see if such features are in the plan. One of the great features of Open Source is that if someone desires to really see some function in an application they can easily add it and contribute it to the rest of the community thereby becoming one of the contributors to the Open Source project community!

Most of the include applications within FIRE, including FIRE itself, have Sourceforge projects that contain active forums discussing various methods of applying the tools as well as troubleshooting of already addressed issues.

If you have problems, comments, complements or any other feedback, the developers of FIRE & Autopsy/The Sleuth Kit (TASK) always are open for assistance.

FIRE: William Salusky: change at dmzs.com
Autopsy/Sleuthkit: carrier at sleuthkit.org

Legal Issues of Incident Handling

Information security legislation has been around since the telephone was created and information was able to be transmitted over vast distances. Many of the bills and acts created by the US and State governments within the United States of America are intended to protect individuals, businesses, critical systems, information and infrastructure.

Depending on where an offense occurred as well as where the perpetrator was located different laws would apply to each scenario and individual involved. Some of the more applicable laws include the following:

- FCC 1996 Telecommunications act which was the first major revision to communications legislation since around 1934. http://www.fcc.gov/telecom.html
- The Computer Fraud and Abuse Act of 1986 which was updated in 1994, 1996 and 2001, http://www.usdoj.gov/criminal/cybercrime/1030_new.html, identifies non-authorized access and use of computer systems and what the basic ramifications such criminal activity would be.
- Electronic Communications Privacy Act of 1986, http://legal.web.aol.com/resources/legislation/ecpa.html, sets out the provisions for access, use, disclosure, interception and privacy protections of electronic communications.
- The Digital Millennium Copyright Act of 1998, http://www.loc.gov/copyright/legislation/dmca.pdf, was enacted to protect copyright owners from illegal activity surrounding the use and or distribution of their copyrighted material.
- Homeland Security Act of 2001, http://www.nist.gov/director/ocla/HR_5005_Enrolled.pdf, has many far reaching items that effect much electronic information. This act overlaps the wiretap and other previous acts created by the US Congress.
- The E-Sign act of 2000, http://www.amc.army.mil/amc/ci/matrix/documents/public_law/electronicsignatures.pdf, which allows for the use of electronic signatures as a valid method of accepting the non-repudiation of the signer as well as requiring businesses to reasonably demonstrate that they can accept submitted material electronically. This is especially useful to those who wish to prove their evidence is securely stored.
- The Computer Security Enhancement act of 1997, http://thomas.loc.gov/cgi-bin/query/z?c105:h.r.1903.eh:, which worked toward updating the standards of security used especially toward encryption.
- The Computer Security Act of 1987, http://csrc.nist.gov/secplcy/csa_87.txt
- The Children's Online Privacy Act of 1998 (COPPA), http://www.cdt.org/legislation/105th/privacy/coppa.html, protects children's privacy online

- Gramm-Leach-Bliley Act, http://www.senate.gov/~banking/conf/, which started the modern protection of email opt-out options along with the protection of personal non-public information.
- Health Insurance Portability & Accountability Act of 1996 (HIPPA), http://thomas.loc.gov/cgi-bin/bdquery/z?d104:HR03103:|TOM:/bss/d104query.html, which provided similar protections that GLB did for the financial sector to the health industry.

Technology laws are constantly changing, and with them the landscape has become filled with various entities attempting to analyze the legislation and interpret what it means to their subscriber base; be it a business or active citizen.

To keep track of some of the latest bills and other legal activity surrounding the Internet would be a full time endeavor. To assist with searching current and historic legislation regarding Technology the Center for Democracy & Technology created an informative website. Their website, http://www.cdt.org/, contains legal activity happening from the 105th Congress in 1997 to current activity occurring today.

Another excellent resource reviewing the ever changing landscape of legislative technological issues is the Michigan Telecommunications & Technical Law Review. The MTTLR "was one of the first law journals to use interactive media to promote informed discourse about the interrelated legal, social, business, and public policy issues raised by emerging technologies."[iii]

One final watchdog center is the Electronic Privacy Information Center, http://www.epic.org. EPIC was created in 1994 and is based out of Washington, DC. EPIC focuses public attention on emerging civil liberties issues and to protect privacy, the First Amendment and Constitutional values[iv].

Not only do people need to keep track of all of these changes, but then there are specific changes occurring within each state. Some states have excellent listings of the Information Security litigation occurring within their state, others will require contacting the state directly. A few of the states that offer interesting litigation and references include:

- California offers the California legislative page, http://www.leginfo.ca.gov/. One interesting item from California is from the Business & Professional code section 17538.4 and 17538.45 which protect against unauthorized faxes and SPAM. Many computers in California now include header information on their mail servers notifying potential spammers that their computer is located in California & as such is protected against unauthorized SPAM through the above mentioned legal codes.
- Virginia is another state that is actively fighting for Information & Security legislation to protect the residences and businesses located with VA.

Information on legislation in VA can be found at:
http://www.wutchathink.com/goverment/legislation.htm.

Given the scenario of a System Operator for an ISP being called by a law enforcement officer attempting to track down a suspect, depending on the relationship of the Company & System Administrator answering the call with the law enforcement individual and department calling there are several possible outcomes to each of the possible question they might request.

If the officer is requesting information during an initial contact, and the individual and company being contacted has an established relationship with the officer there is a possibility that the requested information may be provided to the officer.

If the information the officer is asking for is part of the ISPs direct business and not a co-located customer, then they have the ability to choose whether they wish to disclose. While the wiretap act may prevent many instances of monitoring, within a private business there is much flexibility over what they have the right to monitor.

However with all of the overlapping possibilities of clients, such as possible health care or banking institutions or even other 3rd party providers utilizing the network facilities at the ISP, the Admin may not be able to provide any answers to the law enforcement officer. They may need to bring in the 3rd party, involve their legal team and have warrants delivered prior to disclosing any information.

If the officer is required to get a warrant to access the information they are requesting, then the administrator may need to preserve the state of the effected system as well as log in a secure method all activity on that system.

If possible, the administrator should take a binary image of the effected system, including memory, drive and any specific application data and store in a secure location. The MD5sum of the digital media can be provided to the law enforcement official so they can have proof that the media was not tampered with from the time of the initial request to when they were able to acquire their copy.

If the system remains active after the image is created, then as much activity as is possible should be logged to a secure media. Either log the system log to a line printer or to a remote syslog server that archives it's logs to read-only media on a regular basis.

While all of these items are being done, the System Administrator should be keeping a notebook of all activity that was performed, all conversations and any other notes that may be taken. Each item should have a time-stamp of when it occurred; especially paying attention to what time-zone the activity occurred in.

Finally the Administrator should keep within the notes, on specific pages designated for them, a chain-of-custody to track what happened with each item being tracked as it passes from admin to admin to investigator and beyond.

If it is possible, all of these notes should be kept in a digital medium so that each activity can be MD5sum'd or likewise digitally signed to verify that the notes and information within them have not been altered.

If the officer insists on receiving any of the notes, logs or data from the system administrator then the variety of legal statutes could either allow them to get the information quickly or slowly through a long legal process.

With the enactment of the Homeland Security Act, certain departments within the US Government now have the ability to perform wire taps, impound computer equipment, monitor network activity and engage direct confrontations with possible suspects. In fact under this act the government does not even have to disclose that they are investigating anyone in particular!

So depending on what the activity was, who was involved and what the ramifications of that activity were the system administrator may be forced to give up any and all information; but in many situations outside of national security any law enforcement officer needs to prove just cause in disrupting the business and acquiring the equipment the are investigating.

The just cause that an officer needs to acquire is not much more complicated that they currently have to go through for a wire tap on a suspect. They just need to prove to a Judge that there is criminal activity or intent occurring on the system in question and they can usually receive the authorization they are looking for.

If the Administrator or Company they are working with is involved in criminal activity, there is little they can do to prevent law enforcement from acquiring any equipment, logs, reports or views of network traffic. Just like any other officer not needing a warrant to follow a criminal inside his house if he was following the criminal in an active pursuit.

In a typical business environment the System Administrator, if allowed by his company policy and has authorization from company management, can also proceed in monitoring the system and user in question. Since the user and system are utilizing the network and space of the company, they have the right to monitor their own equipment, users and network. As such the administrator can setup network and system monitors to track what the user is doing as well as what they are attempting to do.

Care should be taken to not alarm the suspect, or impede on their civil rights during the investigation, and extra care should be made to not effect the reliability of the system and services being monitored through the setup and use of the monitoring tools.

If in the course of this additional monitoring the company notices that the suspect is about to actually engage in criminal activities, then they would be required to report that activity to the authorities. They may, depending on the situation, have to only continue monitoring and gathering evidence.

However, if the suspect is actively engaging in activity that could cause harm to someone then they may have to act by removing access of the suspect and turning over the equipment and records to the law enforcement team that should be on-site by this time.

The laws today are extremely complex. They change from state to state, and from country to country. Attempting to fit and analyze what one scenario is not very simple and having to deal with law enforcement from multiple organizations and possibly multiple countries only adds to the confusion. No administrator should be left alone to deal with the issues that have been addressed. They have access to their company's legal team, their executive staff to facilitate interfacing into these various entities.

If the system administrator works for any sizeable organization, then they should work to establish a relationship with their local city and federal computer crimes task force officials, and be sure to establish a policy and procedure for dealing with a variety of possible situations that may occur. After all it is easier to follow a checklist than to have to react in the heat of the moment.

End Notes & References

[i] Brian Carrier. “Sleuthkit & Autopsy”, 2003, URL: http://www.sleuthkit.org/autopsy/desc.php

[ii] David M. Zendzian, FIRE, /bin/consh script that can be found in all releases of FIRE:

```
#!/bin/sh -e
trap "/bin/echo Goodby $USER;exit 0" 1 2 3 4 5 6 7 10 15
. /etc/profile
LOGDIR=/data/consolelogs/
export LOGDIR

# Set the shell
#   if consh run with argument, that is the shell to run, else default to /bin/bash
if [ "$1" = "-c" ]; then
  SHELL="/bin/bash $1 '$2 $3 $4 $5 $6 $7 $8 $9'"
  TOEXEC="/bin/bash $1 '$2 $3 $4 $5 $6 $7 $8 $9'"
else
  SHELL="/bin/bash"
  TOEXEC=/bin/bash
fi
export SHELL TOEXEC

# if $USER not set, default to unknown
if [ "$USER" = "" ]; then
  USER=`whoami`
  export USER
fi

# get tty, if unknown use smaller notty name
TTY=tty_`tty | xargs -i basename {}`
if [ "$TTY" = "tty_not a tty" ]; then
  TTY=tty_none
  NOTTYLOG=nottylog/
  export TTY NOTTYLOG
fi

# set date
DATE=`date +%h%d-%H%M%S`
export DATE
mkdir -p $LOGDIR/$USER/$NOTTYLOG

/usr/bin/script -a -q -f -t $LOGDIR$USER/$NOTTYLOG$DATE-$TTY.log $TOEXEC \
                                        2>$LOGDIR$USER/$NOTTYLOG$DATE-$TTY.log.timing
```

[iii] “The Michigan Telecommunications & Technology Law Review”, 1994-2003, URL: http://www.mttlr.org/

[iv] Electronic Privacy Information Center, “About Epic”, 2003, URL: http://www.epic.org/epic/about.html