



Global Information Assurance Certification Paper

Copyright SANS Institute
Author Retains Full Rights

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

Interested in learning more?

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

GIAC Certified Forensic Analyst Practical Assignment Analysis of a floppy disk and use of SSH as a forensic tool

Version 1.5

Part 1
Part 2 Option 2

Layne Bro
Sept. 15, 2004

Practical Summary

This paper is submitted as response to the practical requirements for the GIAC Certified Forensic Analyst certification.

Part 1 is a forensic analysis of a floppy disk that was confiscated from an employee from the fictitious company "Ballard Industries." The paper shows the step by step process used to analyze the disk and extract data that was hidden, deleted, or otherwise obfuscated from the casual observer. The evidence contained in this section of the paper could be used in a legal case against the employee caught with the disk.

Part II is a validation of the use of SSH as a tool for forensic work. The tool is introduced and described. Then the tool is used in the creation of disk images and shown to be both reliable and efficient. The tool is recommended for use by Forensic Analysts due to its speed, relative ease of use, and accuracy.

© SANS Institute 2000 - 2005, Author retains full rights.

Part 1: Forensic Analysis of a floppy disk

Case Synopsis

Ballard Industries, a designer of fuel cell batteries, suspects a leak of proprietary information and/or intellectual property from their company to a competitor.

Ballard Industries produces specialized batteries used by thousands of companies around the world. For several years, Ballard has successfully marketed a new fuel cell battery. Recently they noticed that several clients were no longer reordering this battery from them. The VP of sales called several companies and discovered that one of Ballards major competitors, Rift, Inc., has been taking orders for this new battery. This was a new fuel cell battery that was unique and only available from Ballard Industries.

Initial investigations into the possible leak or misuse of proprietary information have turned up little. The only unusual event discovered was a floppy disk being taken out of the R&D labs, which is against company policy. Robert Leszczynski was carrying out the disk on April 26, 2004 at approximately 4:45 pm Mountain Standard Time.

The suspicion is that a customer database, or portions of it, along with other proprietary data were removed from the company and somehow obtained by Rift Inc.

David Keen, the security administrator, has asked for the confiscated floppy to be analyzed. A report of findings will be compiled and delivered back to Mr. Keen.

The only item currently being analyzed is identified as follows:

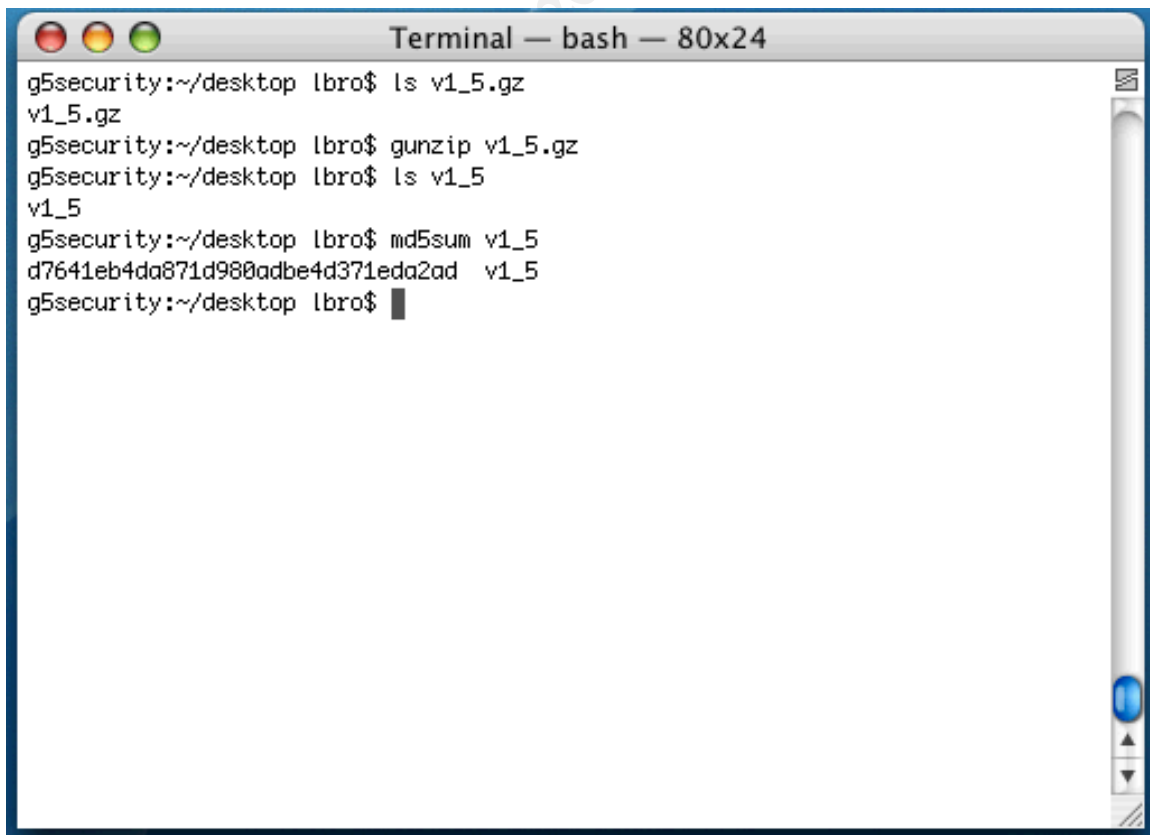
- Tag# fl-260404-RJL1
- 3.5 inch TDK floppy disk
- MD5: d7641eb4da871d980adbe4d371eda2ad fl-260404-RJL1.img
- fl-260404-RJL1.img.gz

Examination Details

An image of the disk, in compressed format was obtained from a centralized server at http://www.giac.org/GCFA_assignment.php. This image was downloaded to the primary security analysis workstation: g5security. This workstation is configured as follows:

- Apple Dual 2GHz PowerPC G5 Workstation
 - S/N G84194W2NVB
- 4 GB DDR SDRAM
- 1 x 160 G internal drive (operating system and executables)
- 1 x 500 G external drive (Drive images for analysis)
- Mac OS X 10.3.5
- The Sleuth Kit 2.03
- Autopsy 1.72

The first action performed was to verify the image downloaded was identical to the original image supplied by the security administrator. To accomplish this, the file needed to be uncompressed, and then create an md5sum to compare with the md5sum provided by Mr. Keen.



```
Terminal — bash — 80x24
g5security:~/desktop lbro$ ls v1_5.gz
v1_5.gz
g5security:~/desktop lbro$ gunzip v1_5.gz
g5security:~/desktop lbro$ ls v1_5
v1_5
g5security:~/desktop lbro$ md5sum v1_5
d7641eb4da871d980adb4d371eda2ad v1_5
g5security:~/desktop lbro$
```

Comparing the two md5sums shows that the original image and the downloaded image are identical.

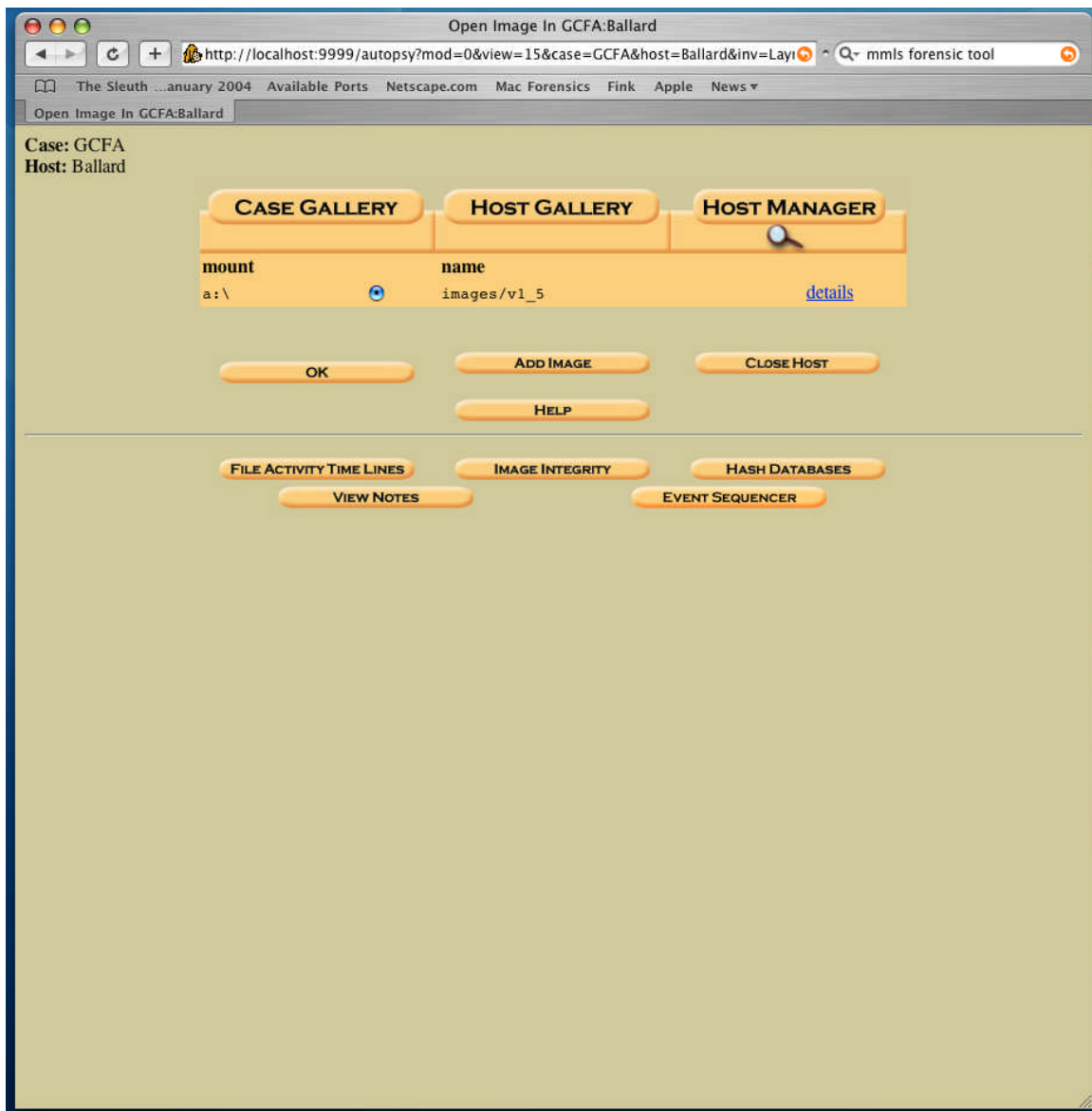
Now that we know the copy we have for analysis and the original image are identical, we can proceed with analysis.

This analysis workstation has Autopsy installed, which is a good choice for analysis of this disk image. It will perform much of the detail work automatically and allow the analyst to concentrate on the information being discovered.

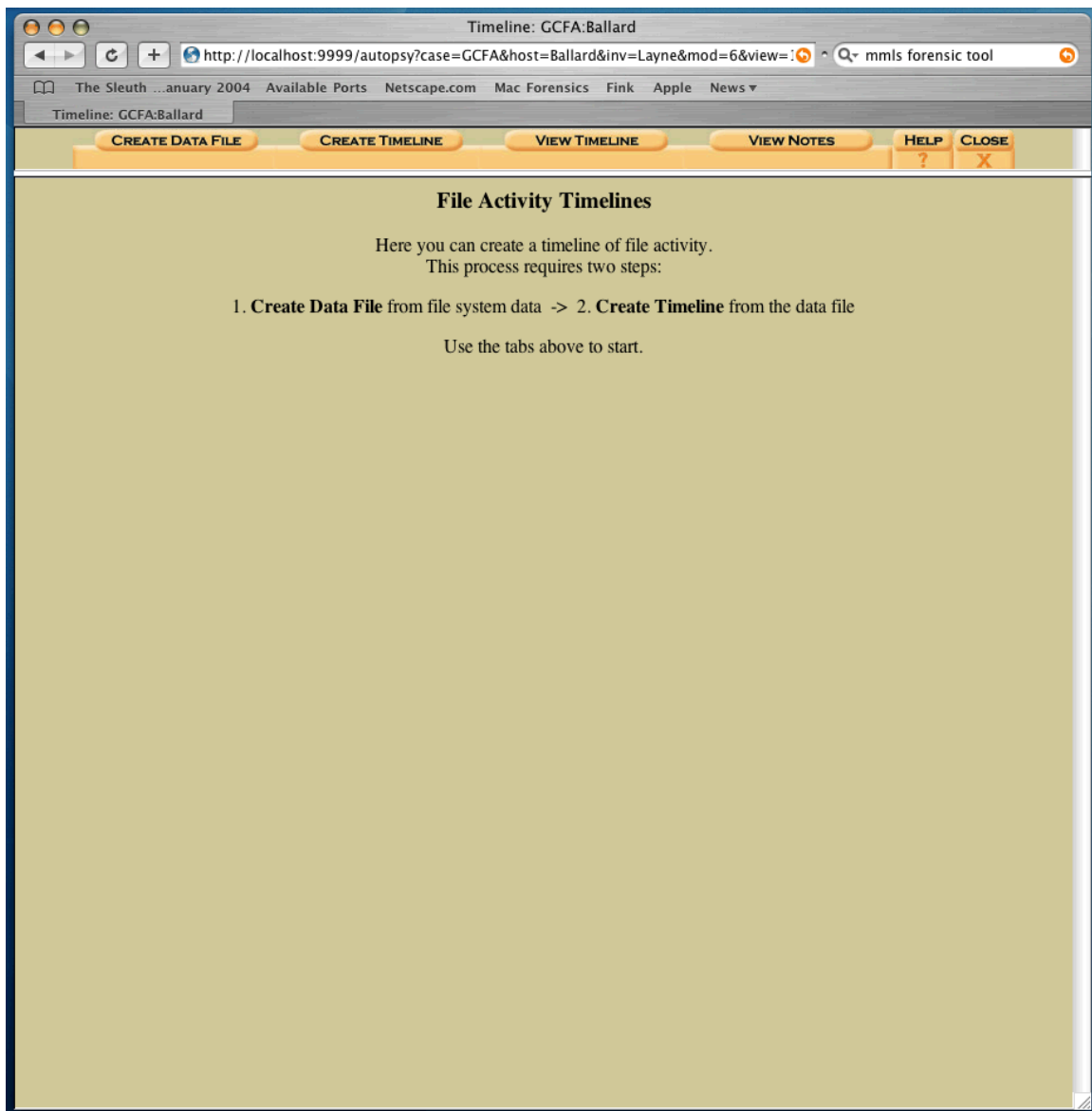
We go ahead and launch Autopsy and get the case setup, the image imported, and we are ready to go.

The first analysis step we want to perform is to create a timeline. In this window we can simply click on the “File Activity Timeline” button.

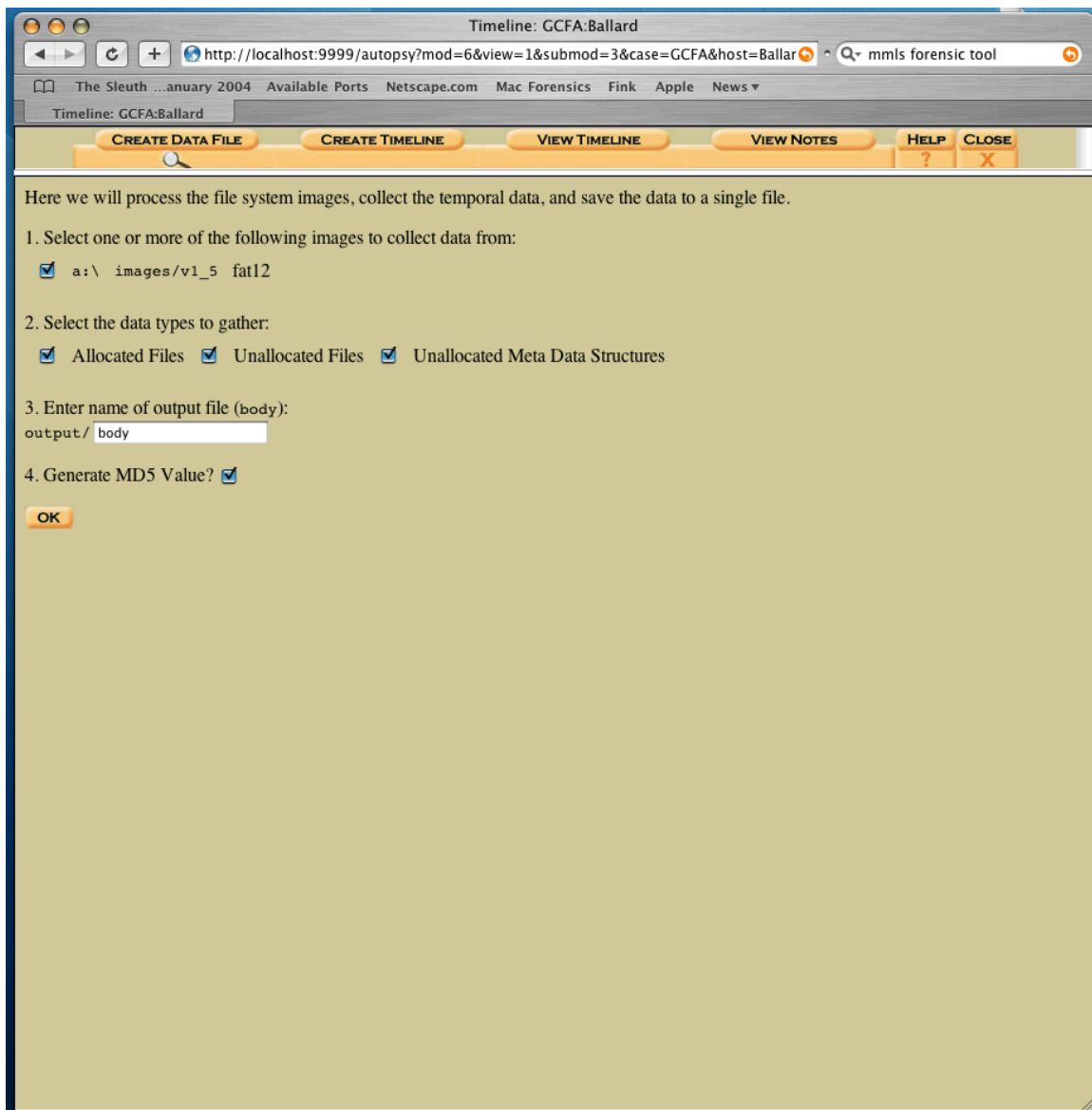
© SANS Institute 2000 - 2005, Author retains full rights.

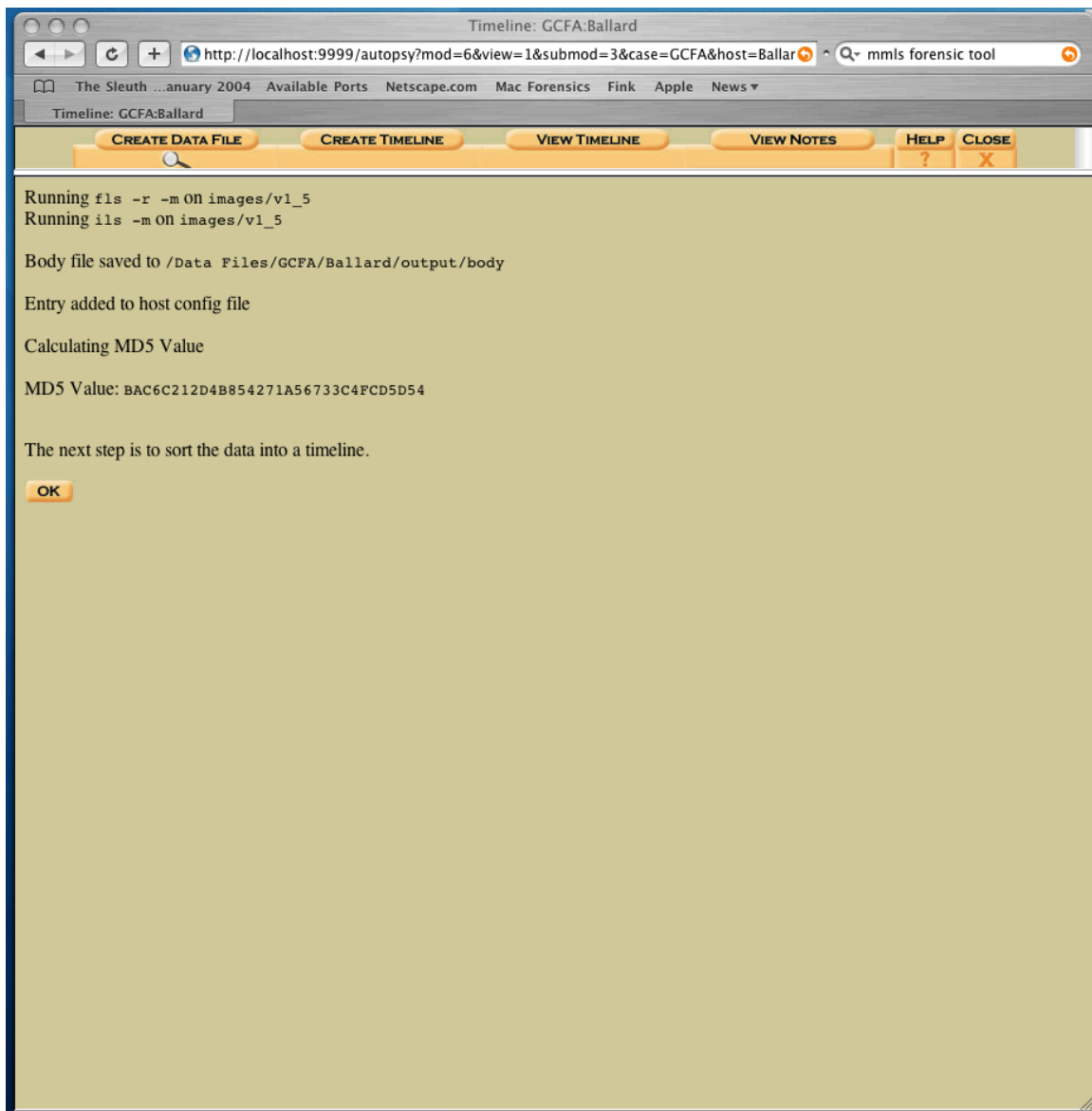


We are now presented with a new page and instructions on how to create a timeline in Autopsy. It is, as indicated, a two step process.



We'll go ahead and follow the instructions to create our timeline. First, we'll click on "Create Data File." Making sure to select the image we need to create a timeline for, we'll then click "OK"





And we'll continue our timeline creation by clicking on "OK." On the next screen, since we have no time limits to input, we'll go ahead and click on "OK" again.

Timeline: GCFA:Ballard

http://localhost:9999/autopsy?body=body&mod=6&view=1&submod=5&host=Ballard mmls forensic tool

The Sleuth ...anuary 2004 Available Ports Netscape.com Mac Forensics Fink Apple News

Timeline: GCFA:Ballard

CREATE DATA FILE CREATE TIMELINE VIEW TIMELINE VIEW NOTES HELP CLOSE

Now we will sort the data and save it to a timeline.

1. Select the data input file (body):
☒ body

2. Enter the starting date:
None: ☒
Specify: ☐ Sep 1 2004

3. Enter the ending date:
None: ☒
Specify: ☐ Sep 1 2004

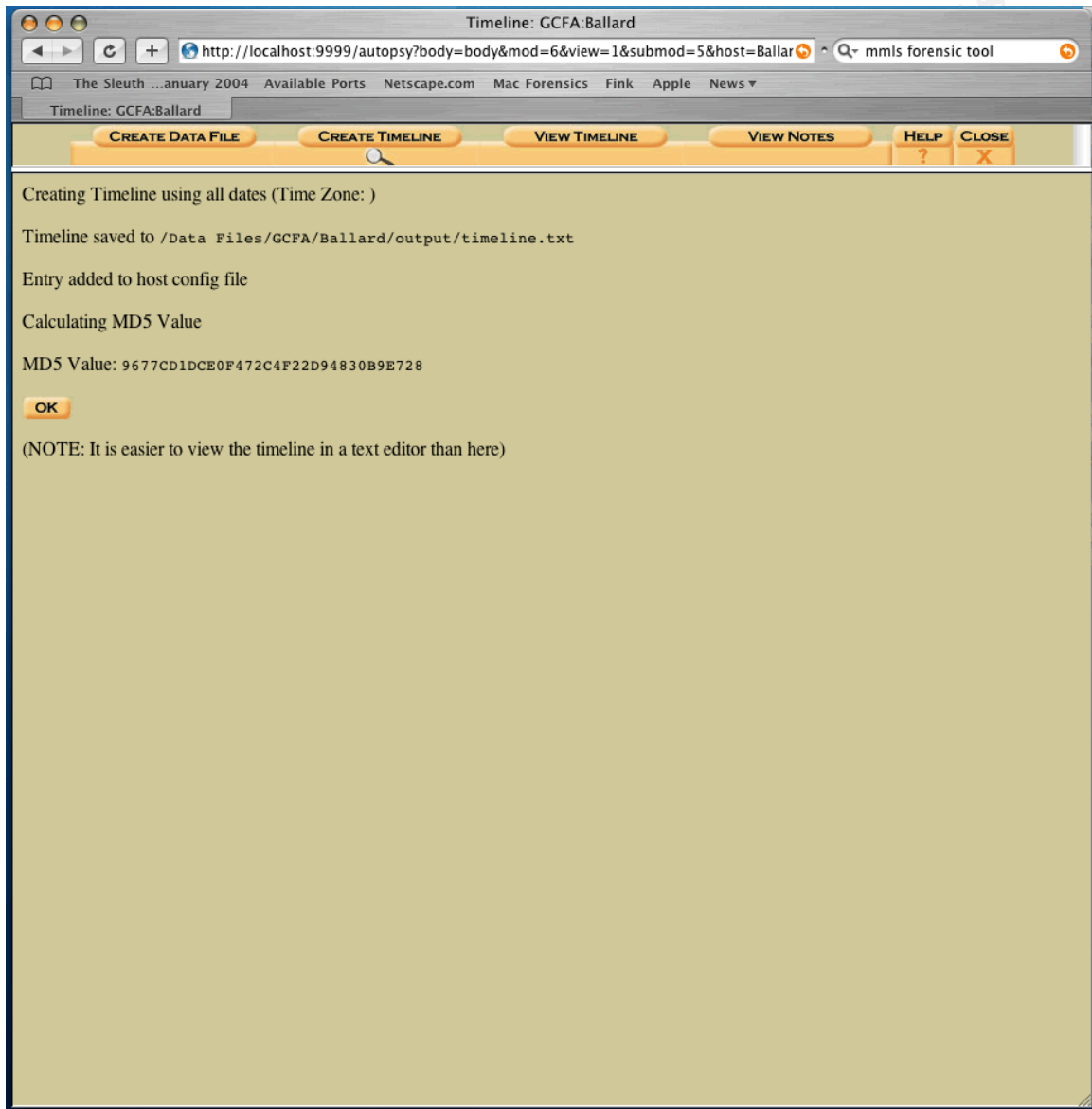
4. Enter the file name to save as:
output/ timeline.txt

5. Generate MD5 Value? ☒

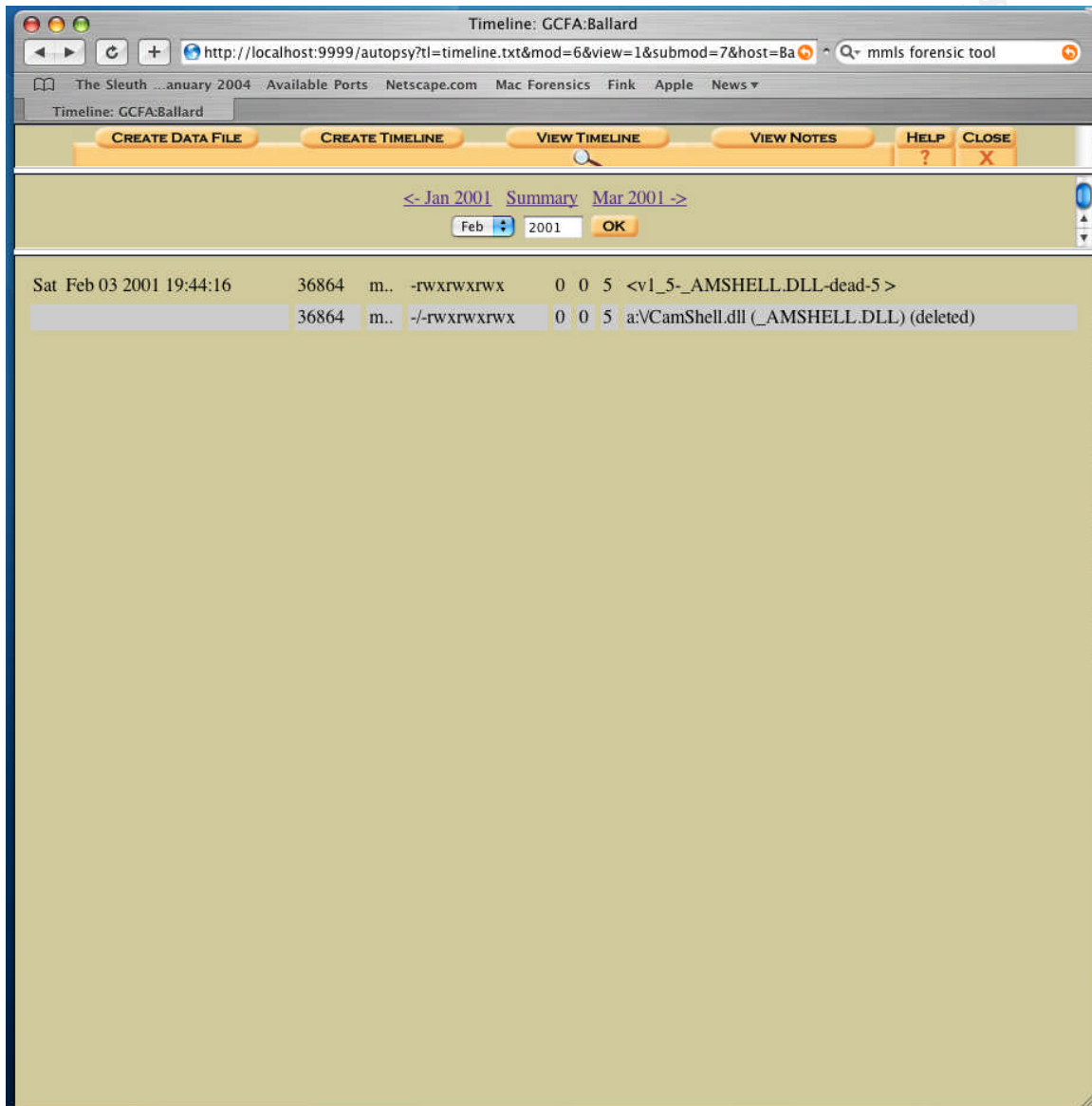
OK

© SANS INSTITUTE

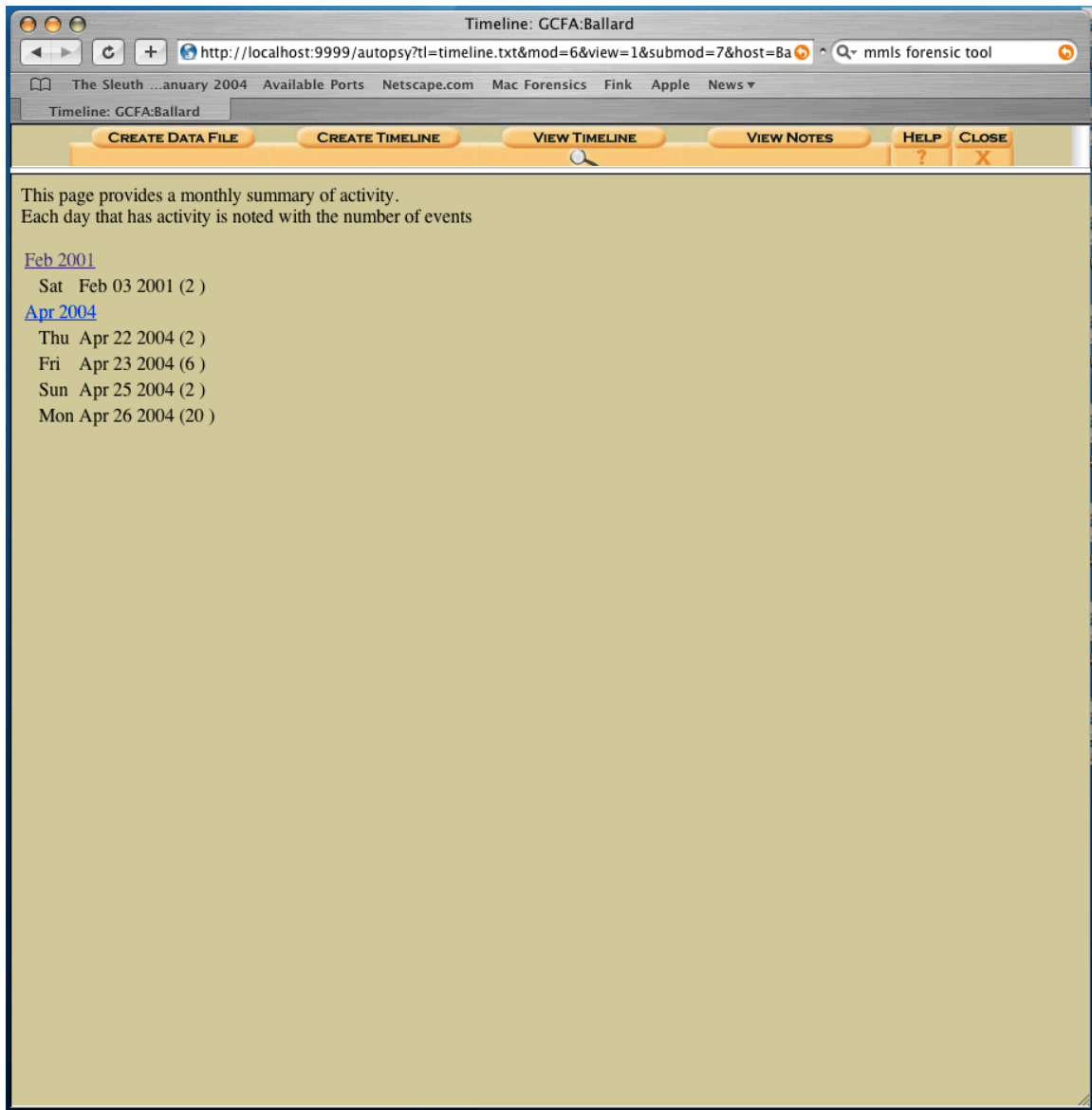
And we receive a window indicating that the timeline was created. However, as the screen indicates, it is often easier to view the timeline in a text editor than in the browser.



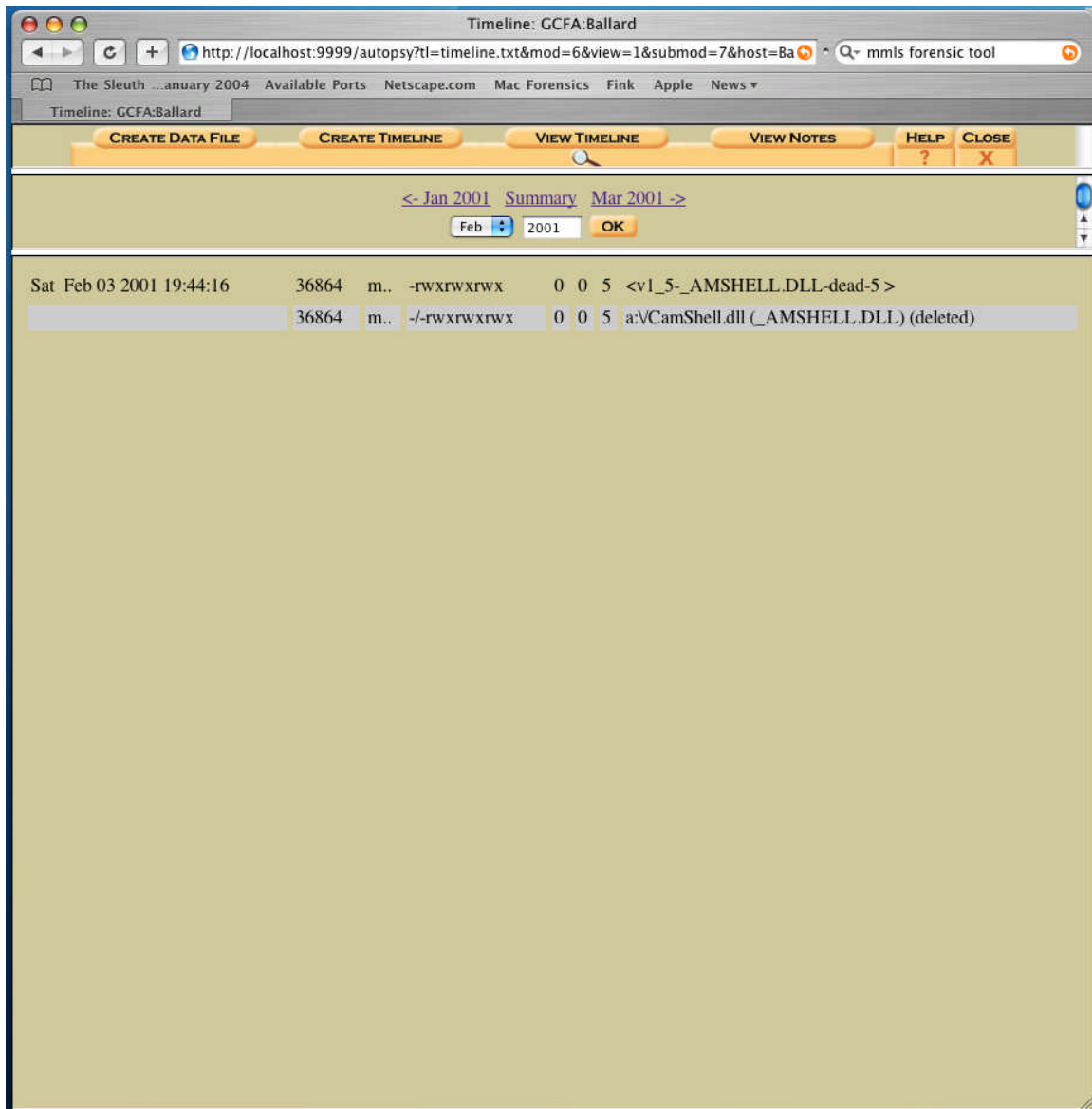
For this small of an image, it should be ok to view the image in the browser. After clicking “OK” in the above screen, we are given a month-by-month view of the timeline.



We can click on the Summary field and receive a listing of all dates in the timeline and the number of activities on that date.



Lets start by exploring the items in February. By clicking on the “Feb 2001” link in the window, we are taken straight to the listing for that month.



Here we see that the file camshell.dll was last modified on Feb 3, 2001 @ 19:44:16. That file has since been deleted as noted by the (deleted) at the end of the entry. If we click on summary and then on April 2004, we'll see lots more activity on this disk.

Timeline: GCFA:Ballard

http://localhost:9999/autopsy?tl=timeline.txt&mod=6&view=1&submod=7&host=Balla mmls forensic tool

The Sleuth ...anuary 2004 Available Ports Netscape.com Mac Forensics Fink Apple News

Timeline: GCFA:Ballard

CREATE DATA FILE CREATE TIMELINE VIEW TIMELINE VIEW NOTES HELP CLOSE

<- Mar 2004 Summary May 2004 ->

Apr 2004 OK

Thu Apr 22 2004 16:31:06	32256	m..	-/-rwxrwxrwx	0 0 13	a:\Internal_Lab_Security_Policy1.doc (INTERN~1.DOC)
	33423	m..	-/-rwxrwxrwx	0 0 17	a:\Internal_Lab_Security_Policy.doc (INTERN~2.DOC)
Fri Apr 23 2004 10:53:56	727	m..	-/-rwxrwxrwx	0 0 28	a:_ndex.htm (deleted)
	727	m..	-/-rwxrwxrwx	0 0 28	<v1_5-_ndex.htm-dead-28 >
Fri Apr 23 2004 11:54:32	215895	m..	-/-rwxrwxrwx	0 0 23	a:\Remote_Access_Policy.doc (REMOTE~1.DOC)
Fri Apr 23 2004 11:55:26	307935	m..	-/-rwxrwxrwx	0 0 20	a:\Password_Policy.doc (PASSWO~1.DOC)
Fri Apr 23 2004 14:10:50	22528	m..	-/-rwxrwxrwx	0 0 27	a:\Acceptable_Encryption_Policy.doc (ACCEPT~1.DOC)
Fri Apr 23 2004 14:11:10	42496	m..	-/-rwxrwxrwx	0 0 9	a:\Information_Sensitivity_Policy.doc (INFORM~1.DOC)
Sun Apr 25 2004 00:00:00	0	.a.	-/-rwxrwxrwx	0 0 3	a:\RJL (Volume Label Entry)
Sun Apr 25 2004 10:53:40	0	m.c	-/-rwxrwxrwx	0 0 3	a:\RJL (Volume Label Entry)
Mon Apr 26 2004 00:00:00	307935	.a.	-/-rwxrwxrwx	0 0 20	a:\Password_Policy.doc (PASSWO~1.DOC)
	215895	.a.	-/-rwxrwxrwx	0 0 23	a:\Remote_Access_Policy.doc (REMOTE~1.DOC)
	36864	.a.	-/-rwxrwxrwx	0 0 5	<v1_5-_AMSHLL.DLL-dead-5 >
	727	.a.	-/-rwxrwxrwx	0 0 28	a:_ndex.htm (deleted)
	36864	.a.	-/-rwxrwxrwx	0 0 5	a:\CamShell.dll (_AMSHLL.DLL) (deleted)
	727	.a.	-/-rwxrwxrwx	0 0 28	<v1_5-_ndex.htm-dead-28 >
	42496	.a.	-/-rwxrwxrwx	0 0 9	a:\Information_Sensitivity_Policy.doc (INFORM~1.DOC)
	22528	.a.	-/-rwxrwxrwx	0 0 27	a:\Acceptable_Encryption_Policy.doc (ACCEPT~1.DOC)
	33423	.a.	-/-rwxrwxrwx	0 0 17	a:\Internal_Lab_Security_Policy.doc (INTERN~2.DOC)
	32256	.a.	-/-rwxrwxrwx	0 0 13	a:\Internal_Lab_Security_Policy1.doc (INTERN~1.DOC)
Mon Apr 26 2004 09:46:18	36864	.c	-/-rwxrwxrwx	0 0 5	<v1_5-_AMSHLL.DLL-dead-5 >
	36864	.c	-/-rwxrwxrwx	0 0 5	a:\CamShell.dll (_AMSHLL.DLL) (deleted)
Mon Apr 26 2004 09:46:20	42496	.c	-/-rwxrwxrwx	0 0 9	a:\Information_Sensitivity_Policy.doc (INFORM~1.DOC)
Mon Apr 26 2004 09:46:22	32256	.c	-/-rwxrwxrwx	0 0 13	a:\Internal_Lab_Security_Policy1.doc (INTERN~1.DOC)
Mon Apr 26 2004 09:46:24	33423	.c	-/-rwxrwxrwx	0 0 17	a:\Internal_Lab_Security_Policy.doc (INTERN~2.DOC)
Mon Apr 26 2004 09:46:26	307935	.c	-/-rwxrwxrwx	0 0 20	a:\Password_Policy.doc (PASSWO~1.DOC)
Mon Apr 26 2004 09:46:36	215895	.c	-/-rwxrwxrwx	0 0 23	a:\Remote_Access_Policy.doc (REMOTE~1.DOC)

And three more entries that didn't fit on the screen.

Mon Apr 26 2004 09:46:44	22528	.c	-/-rwxrwxrwx	0 0 27	a:\Acceptable_Encryption_Policy.doc (ACCEPT~1.DOC)
Mon Apr 26 2004 09:47:36	727	.c	-/-rwxrwxrwx	0 0 28	a:_ndex.htm (deleted)
	727	.c	-/-rwxrwxrwx	0 0 28	<v1_5-_ndex.htm-dead-28 >

Here we find several more items of interest.

Index.htm which was last modified on Friday Apr 23 2004 10:53:56 but is now deleted.

Index.htm and camshell.dll were both last accessed Monday April 26, 2004 at 00:00:00

Then, camshell.dll was last changed on Monday April 26 2004 09:46:18 and index.htm was last changed on Monday April 26 2004 09:47:36.

Other items of note on this page are the unusual times of 00:00:00 on April 25th and April 26th. We can't be entirely sure of the time that these changes were made, but the fact that two nights in a row a change was made at precisely the same time and an unusual time causes concern. Also, on the 26th, every file on the disk was accessed at the same moment. This leads one to think that there was possibly some type of automation in accessing these files.

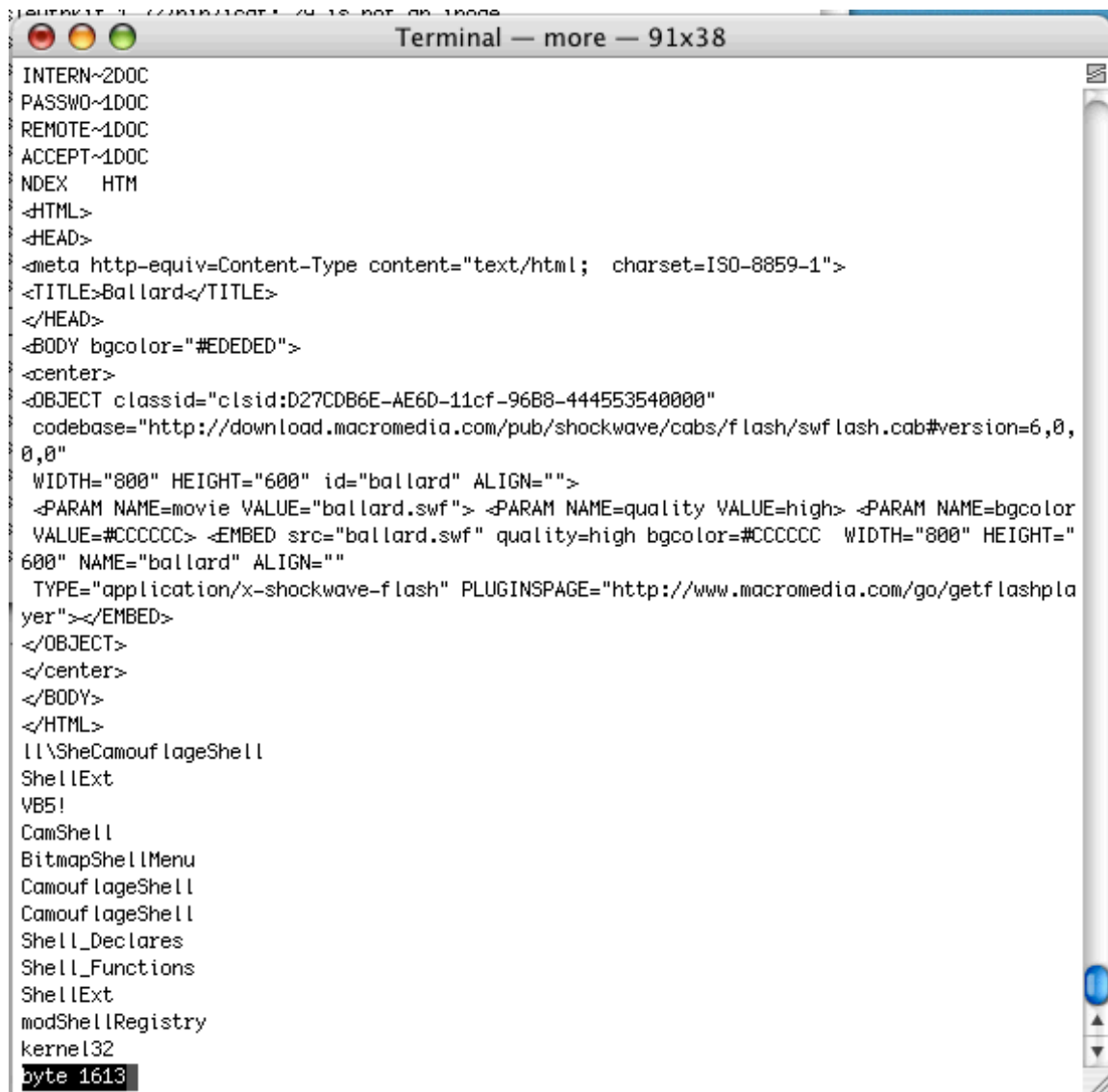
The changed value cannot be manually modified like the modified or accessed times can be (through use of utimes), but this value will reflect the actual time that utimes was used to modify the other values. For instance, its possible that the perpatrator used utimes to modify the last accessed value of all the files on the disk to be 00:00:00. This would be supported by the fact that every file on the disk was last changed within a few seconds of each other on the morning of Monday April 26th, from 09:46:18 through 09:47:36. He would have modified the files in the following order:

Camshell.dll
Information_Sensitivity_Policy.doc
Internal_Lab_Security_Policy1.doc
Internal_Lab_Security_Policy.doc
Password_Polciy.doc
Remote_Access_Policy.doc
Acceptable_Encryption_Policy.doc
Index.htm

So at this point, it looks like Robert manually modified the time stamps to be identical.

We also see that index.htm and Camshell.dll have been deleted. The other documents appear to be Microsoft Word documents, although it's interesting that the last changed and accessed times have been modified.

Lets do a keyword search on the unallocated portions of the disk and see if we find anything interesting. We do find several interesting items:

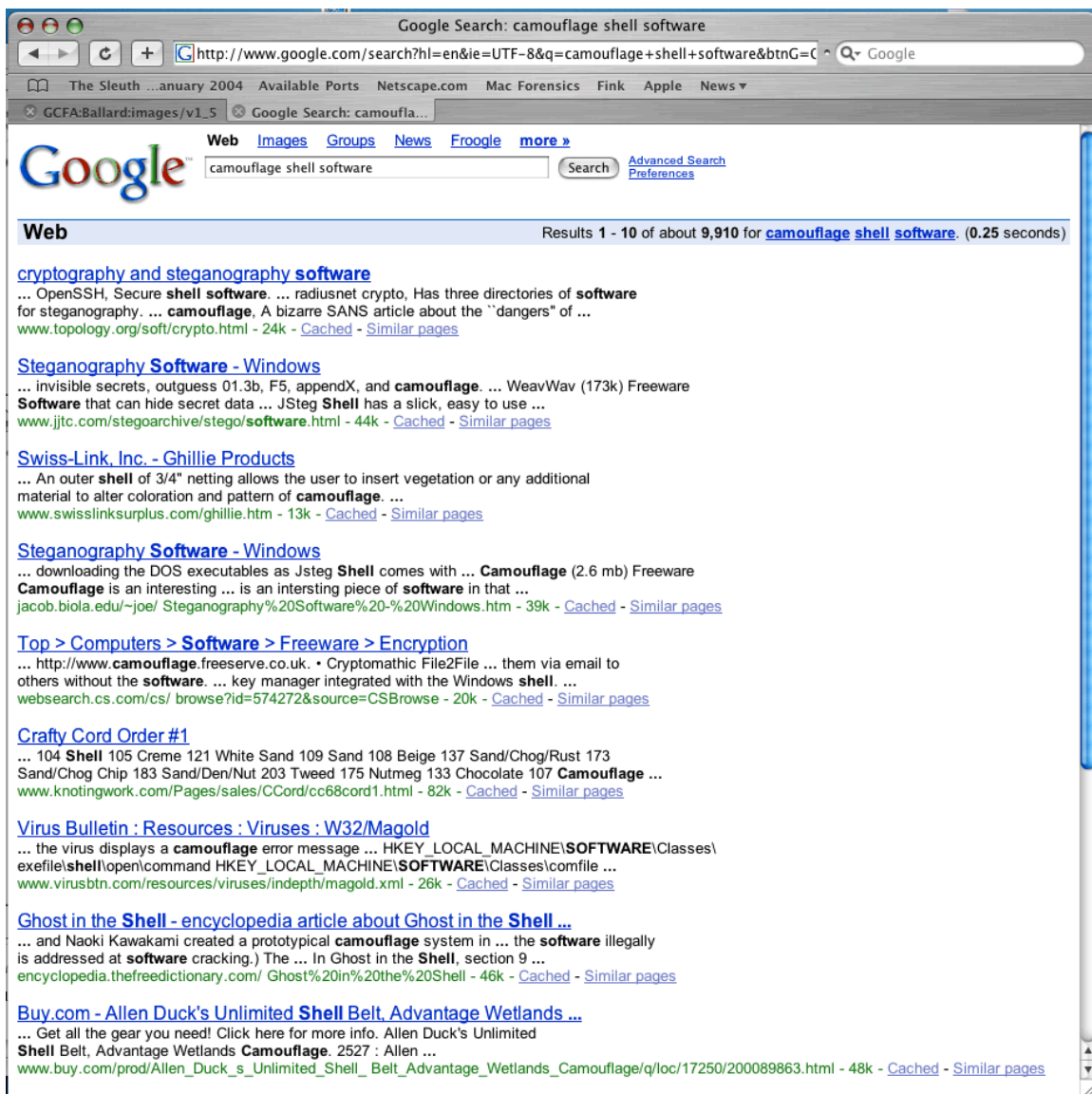


```
Terminal — more — 91x38
INTERN~2DOC
PASSWO~1DOC
REMOTE~1DOC
ACCEPT~1DOC
NDEX   HTM
<HTML>
<HEAD>
<meta http-equiv=Content-Type content="text/html; charset=ISO-8859-1">
<TITLE>Ballard</TITLE>
</HEAD>
<BODY bgcolor="#EDEDED">
<center>
<OBJECT classid="clsid:D27CDB6E-AE6D-11cf-96B8-444553540000"
codebase="http://download.macromedia.com/pub/shockwave/cabs/flash/swflash.cab#version=6,0,0,0"
WIDTH="800" HEIGHT="600" id="ballard" ALIGN="">
<PARAM NAME=movie VALUE="ballard.swf"> <PARAM NAME=quality VALUE=high> <PARAM NAME=bgcolor
VALUE=#CCCCCC> <EMBED src="ballard.swf" quality=high bgcolor=#CCCCCC WIDTH="800" HEIGHT="
600" NAME="ballard" ALIGN=""
TYPE="application/x-shockwave-flash" PLUGINSPAGE="http://www.macromedia.com/go/getflashpla
yer"></EMBED>
</OBJECT>
</center>
</BODY>
</HTML>
ll\SheCamouflageShell
ShellExt
VB5!
CamShell
BitmapShellMenu
CamouflageShell
CamouflageShell
Shell_Declares
Shell_Functions
ShellExt
modShellRegistry
kernel32
byte 1613
```

```
Terminal — more — 91x38
FIShellExtInit
C:\My Documents\WB Programs\Camouflage\Shell\IctxMenu.tlb
IContextMenu_TLB
IContextMenu_GetCommandString
IContextMenu_InvokeCommand
__vbaRedim
__vbaUbound
__vbaVar2Vec
__vbaRecDestruct
__vbaLsetFixstr
__vbaLsetFixstrFree
__vbaLenBstr
__vbaFreeVarList
__vbaFixstrConstruct
__vbaVarTstEq
__vbaVarMove
__vbaVarCopy
__vbaVarDup
7m_szFile
IContextMenu
IShellExtInit
pidlFolder
lpdobj
hKeyProgID
hMenu
indexMenu
idCmdFirst
idCmdLast
uFlags
idCmd
pwReserved
pszName
cchMax
lpcmi
pVfk
pIVR
Pj@j
byte 3184
```

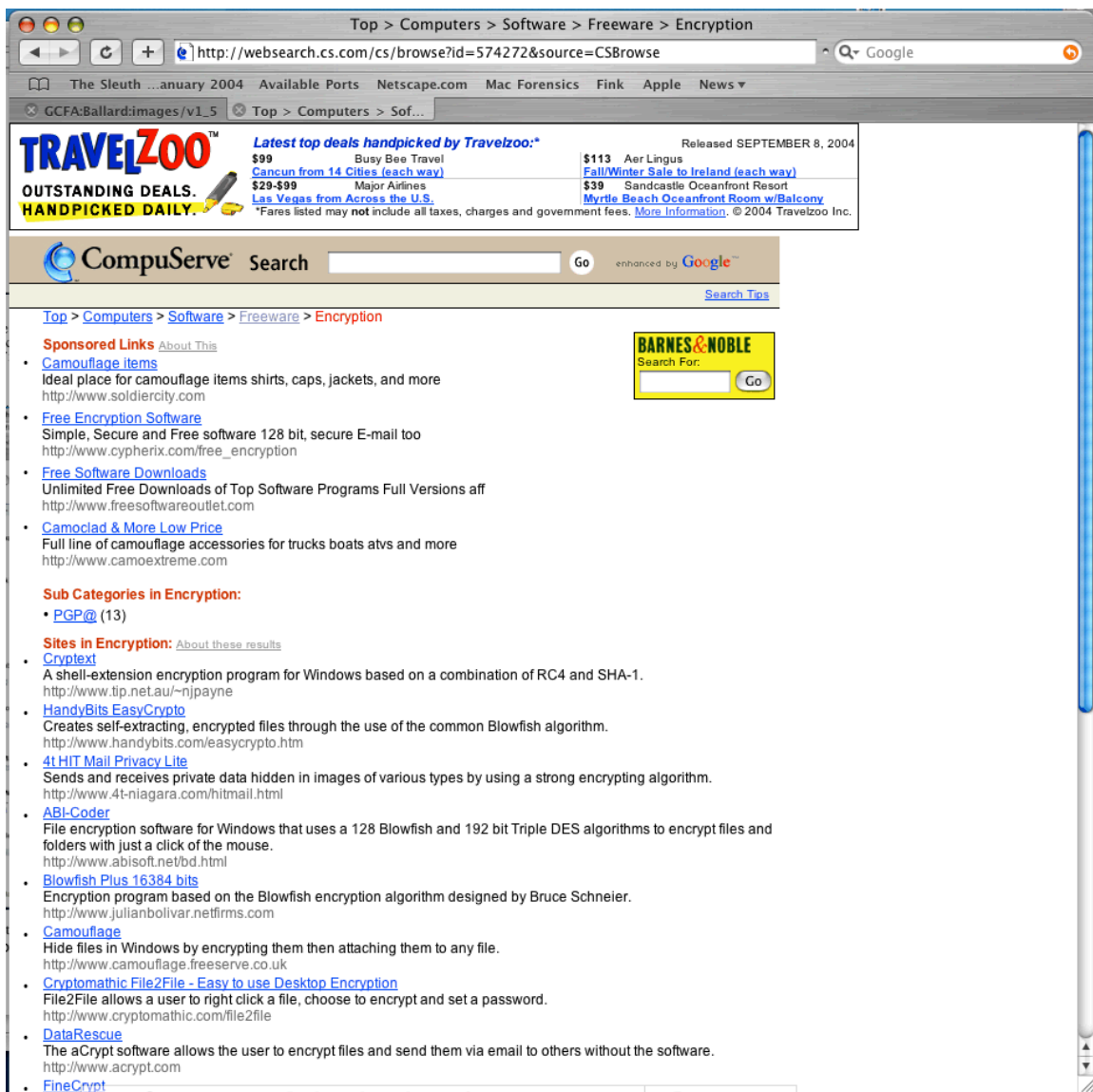
In the first screen shot, we find what appears to be an html page and several references to a camouflage shell – or camshell. In the second screen shot, we see several more references to camouflage and what appear to be Visual Basic commands.

Next we can go to the Internet and run some basic searches for some of our keywords.



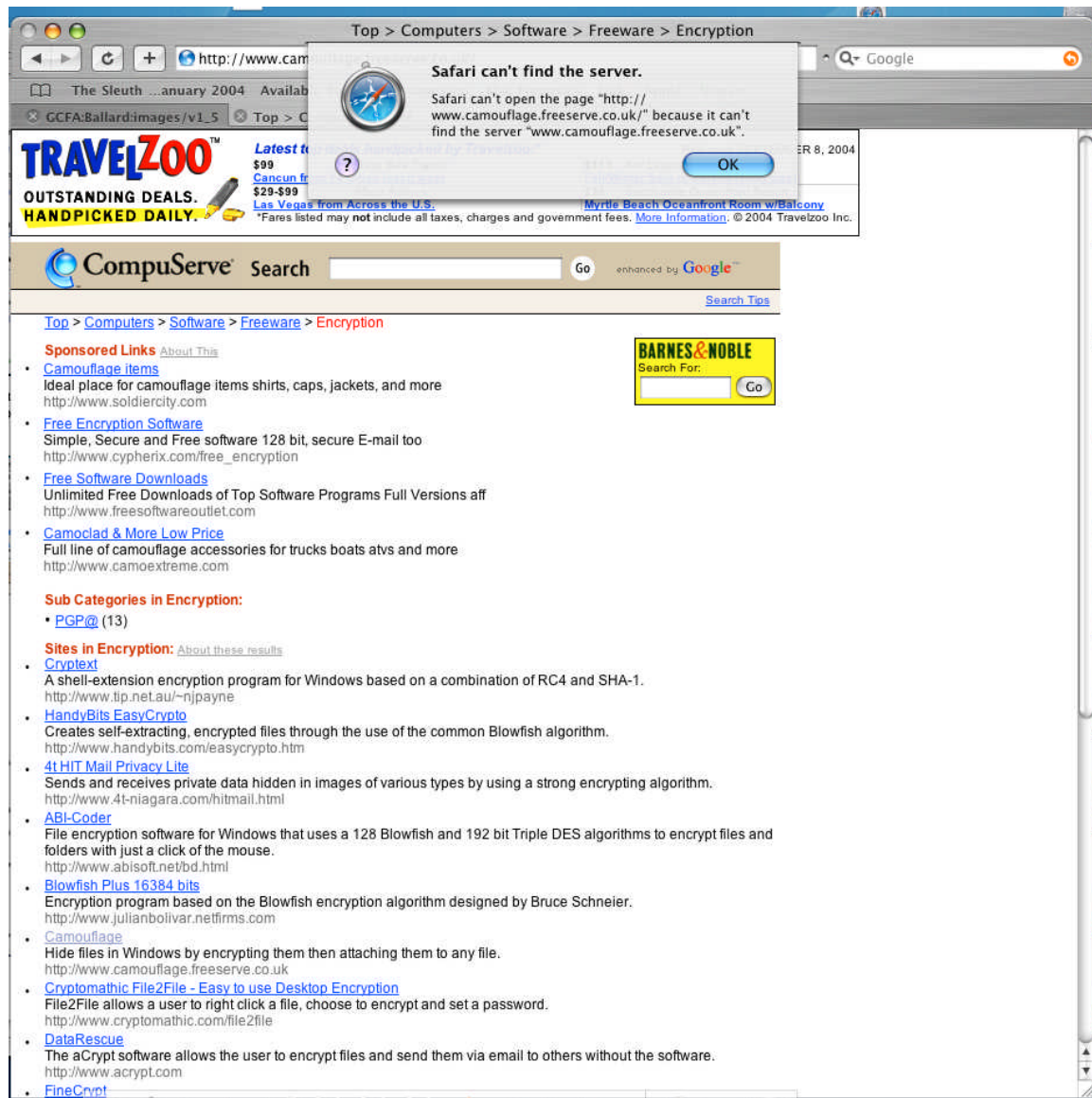
I like the fifth one down because it references another key word that was on our drive image: <http://www.camouflage.freemove.co.uk>

So selecting that we get:



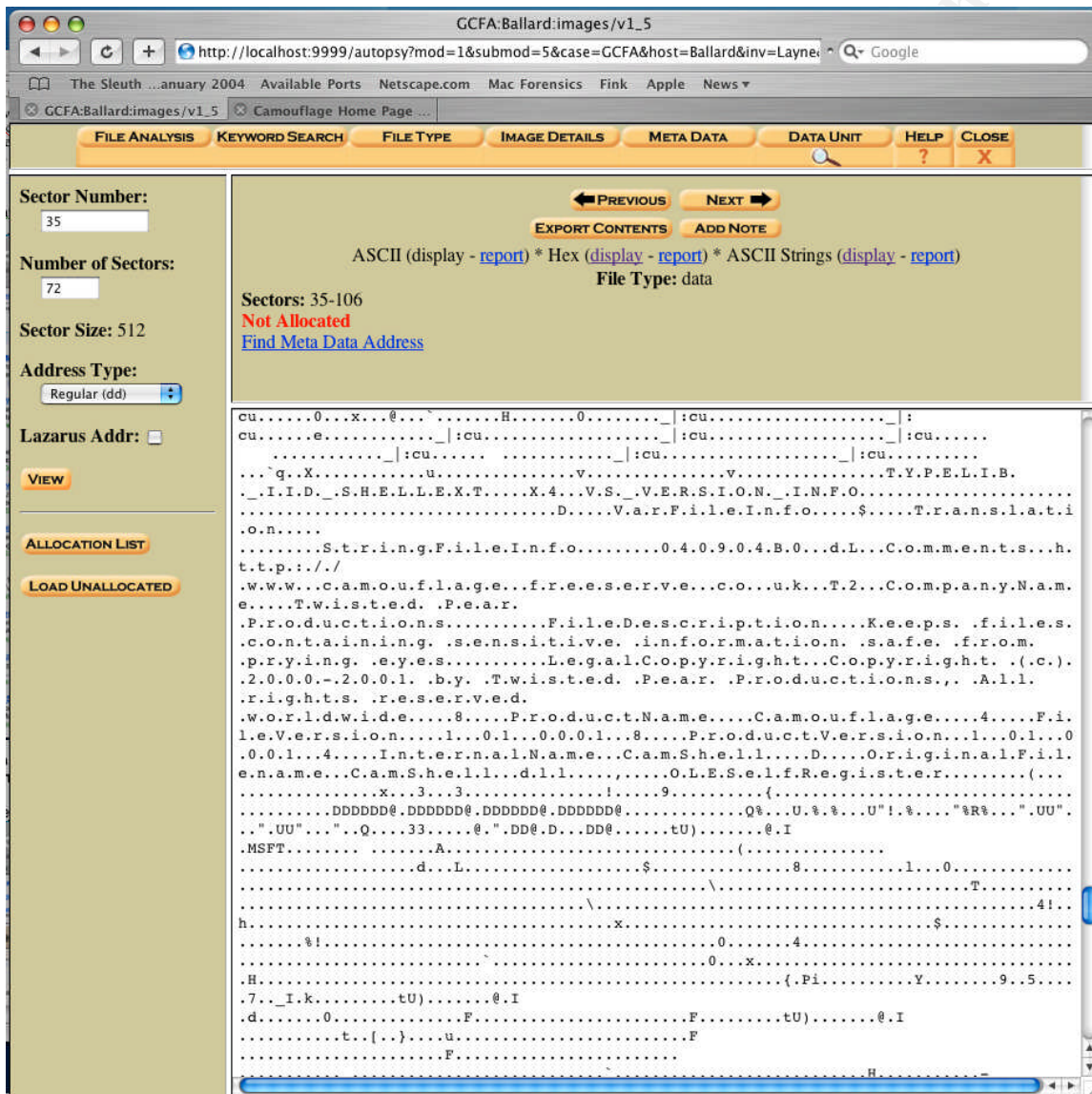
Down near the bottom of the screen, we see a link for “Camouflage” which allows the user to “Hide files in Windows by encrypting them then attaching them to any file.”

However, clicking on this link gives us:

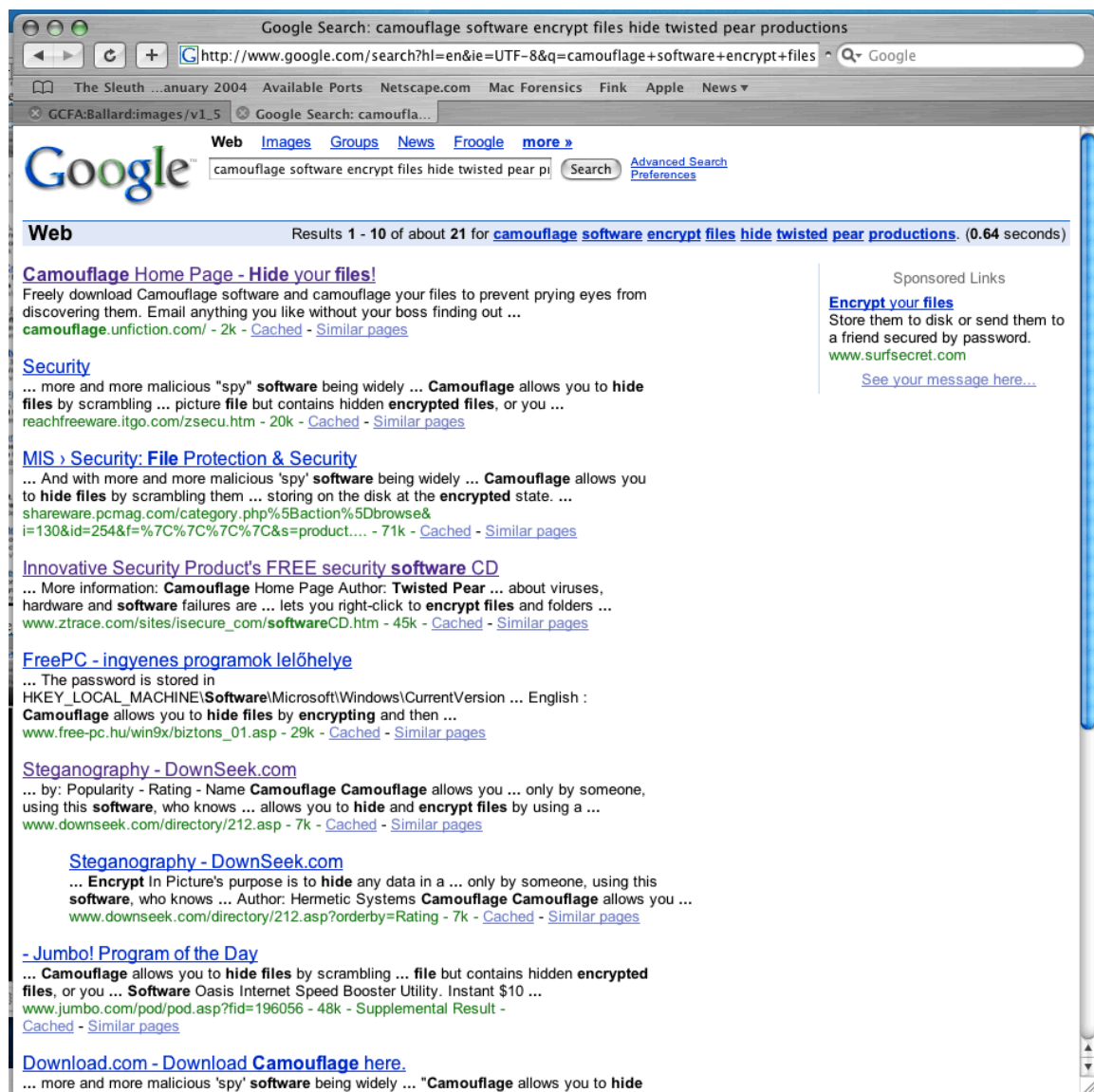


So, the original home page is gone. Lets do some more searching and see if we can find any other sites for this software.

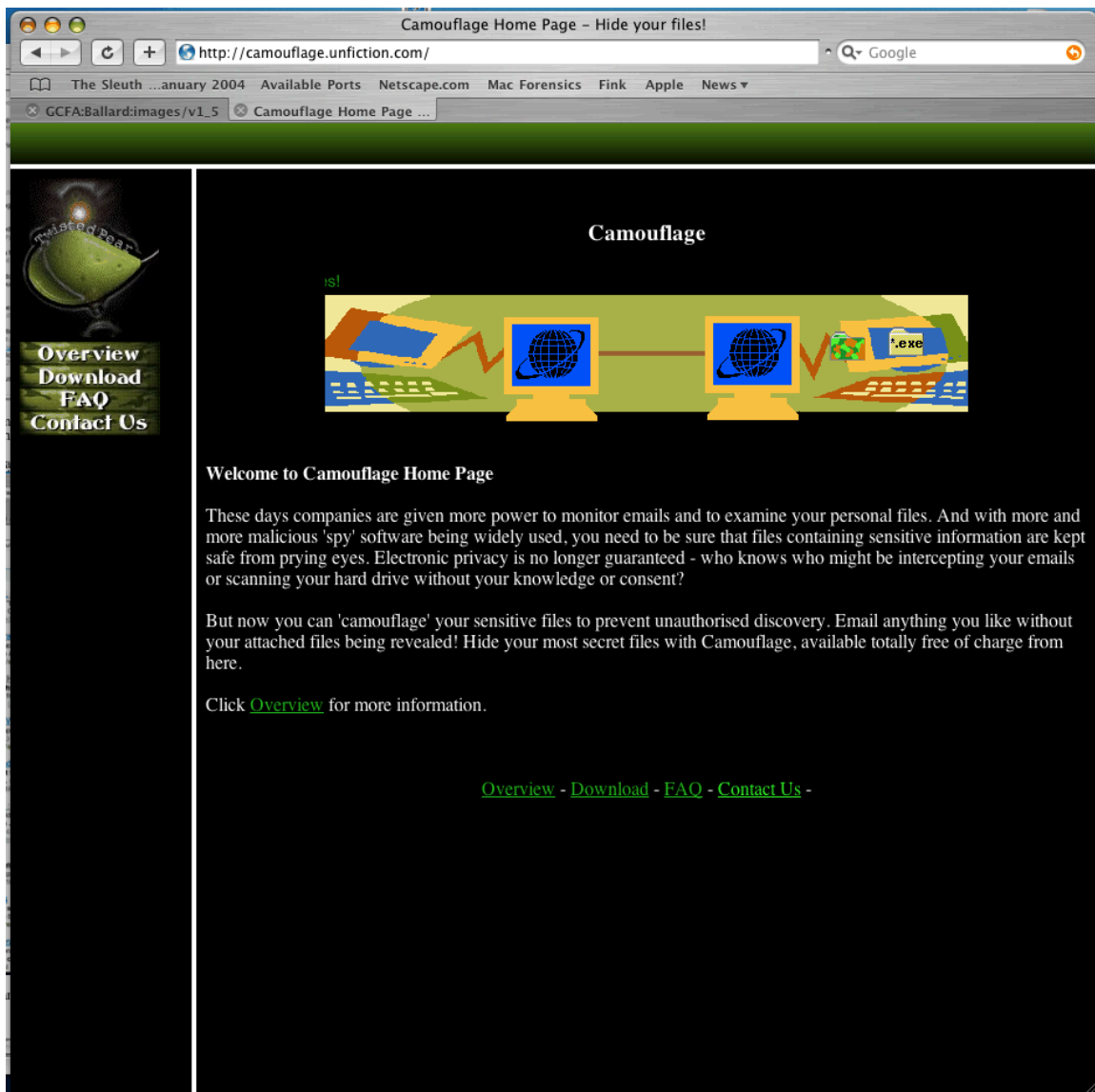
Lets try viewing the camshell.dll file in ascii, strings, and hex format and see what it shows. We get some additional words that might help: "Twisted Pear Productions".

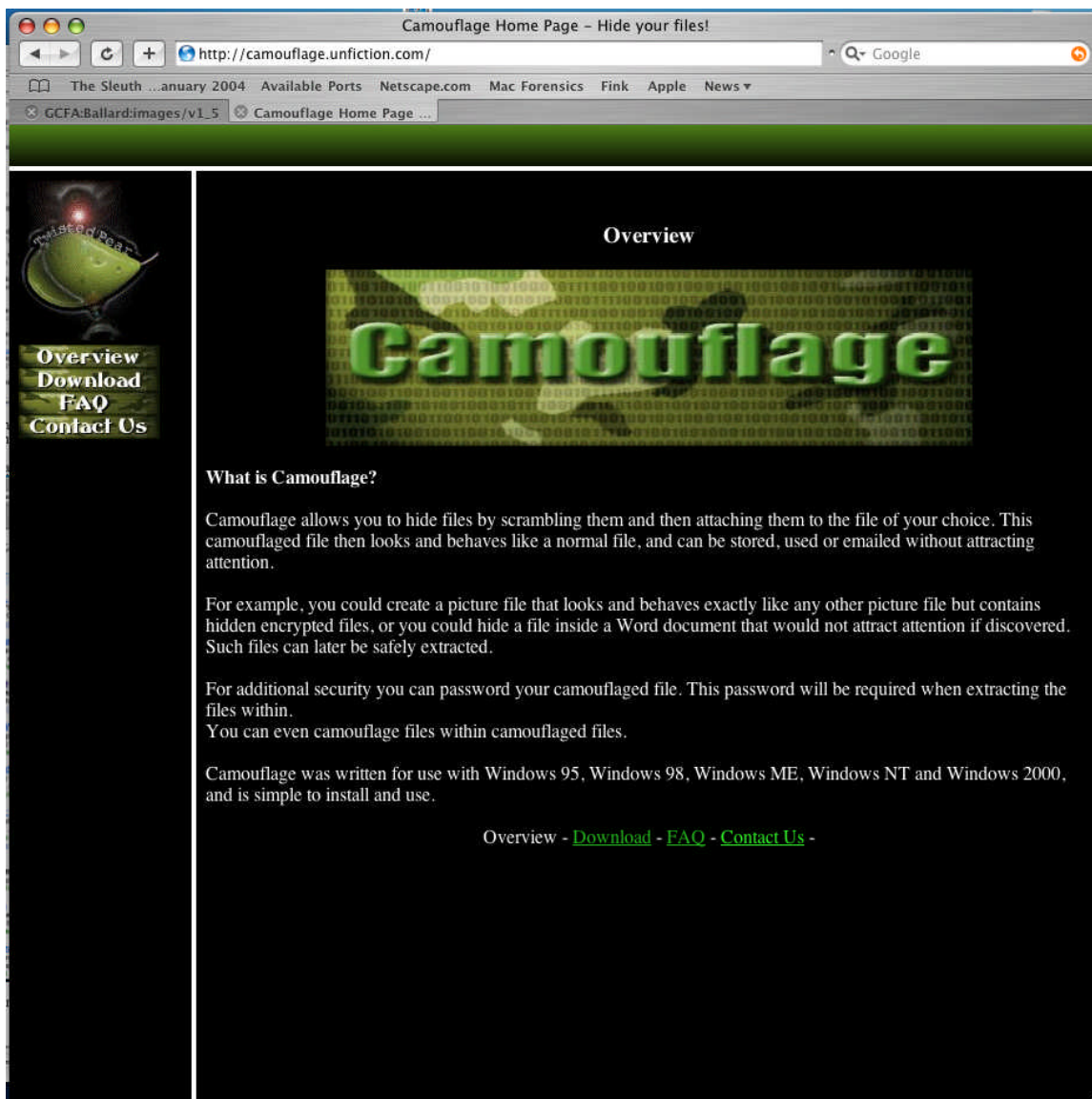


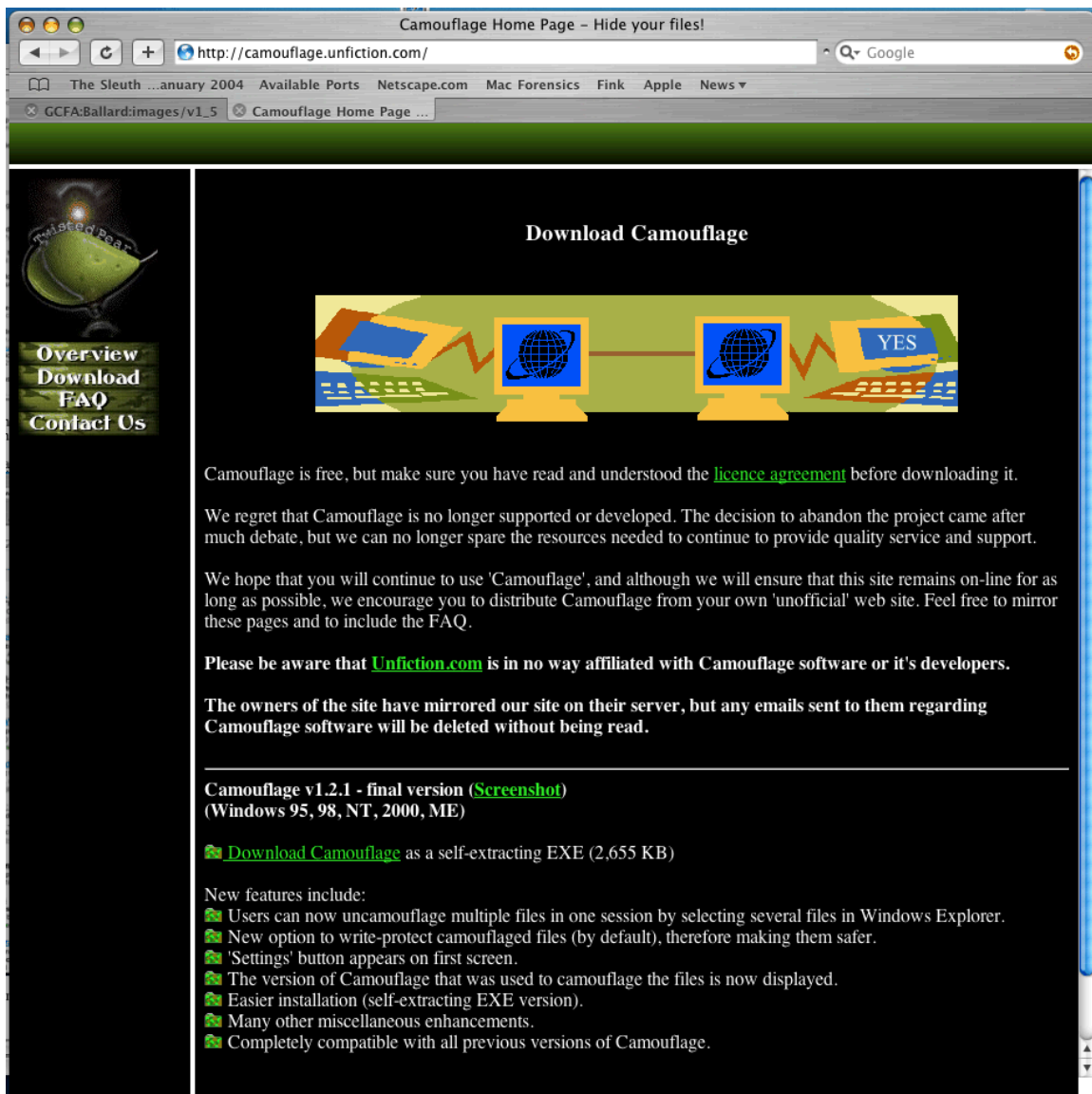
Running another search with more terms gives us better results:



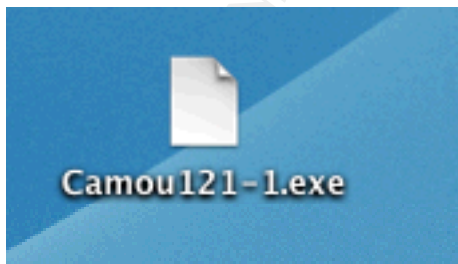
Clicking on the “Camouflage Home Page” gives us a great discovery.







So lets go ahead and download the tool and see what we get.

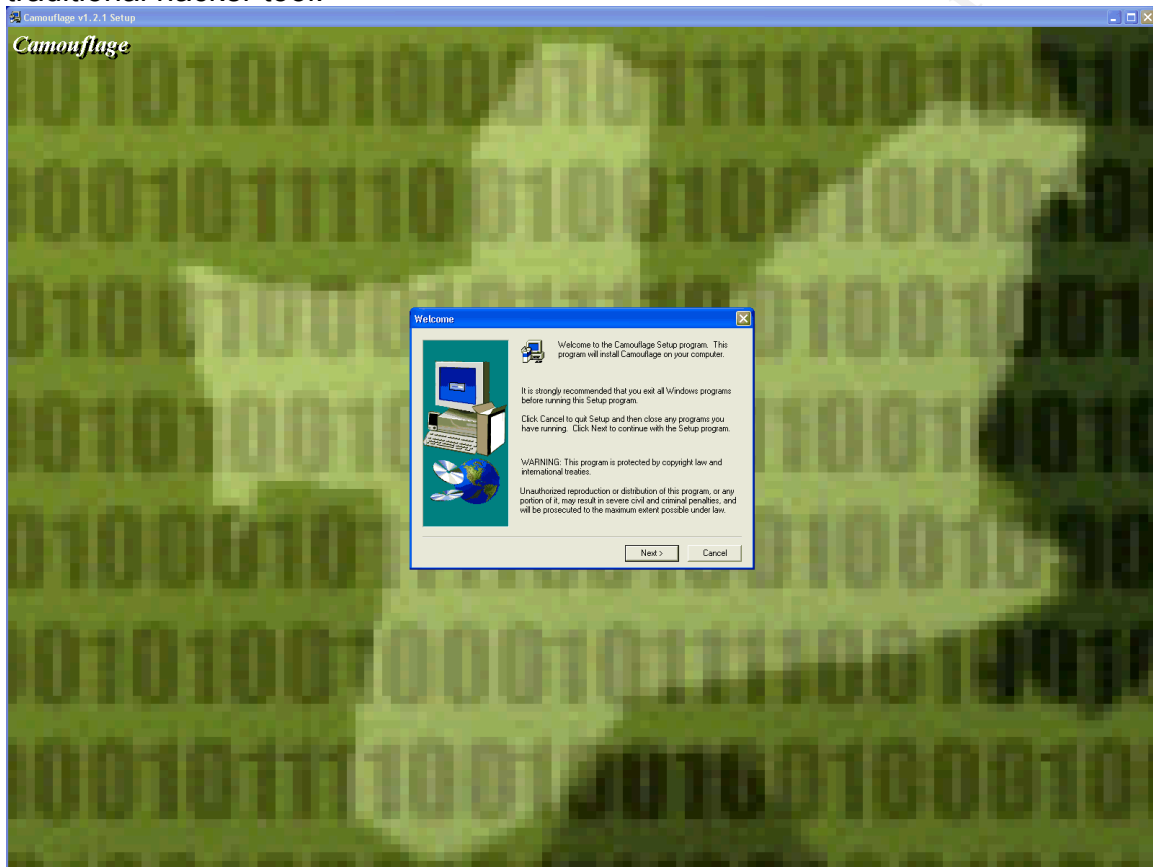


We have a windows executable file. We'll need to use a windows system to do some testing with this software.

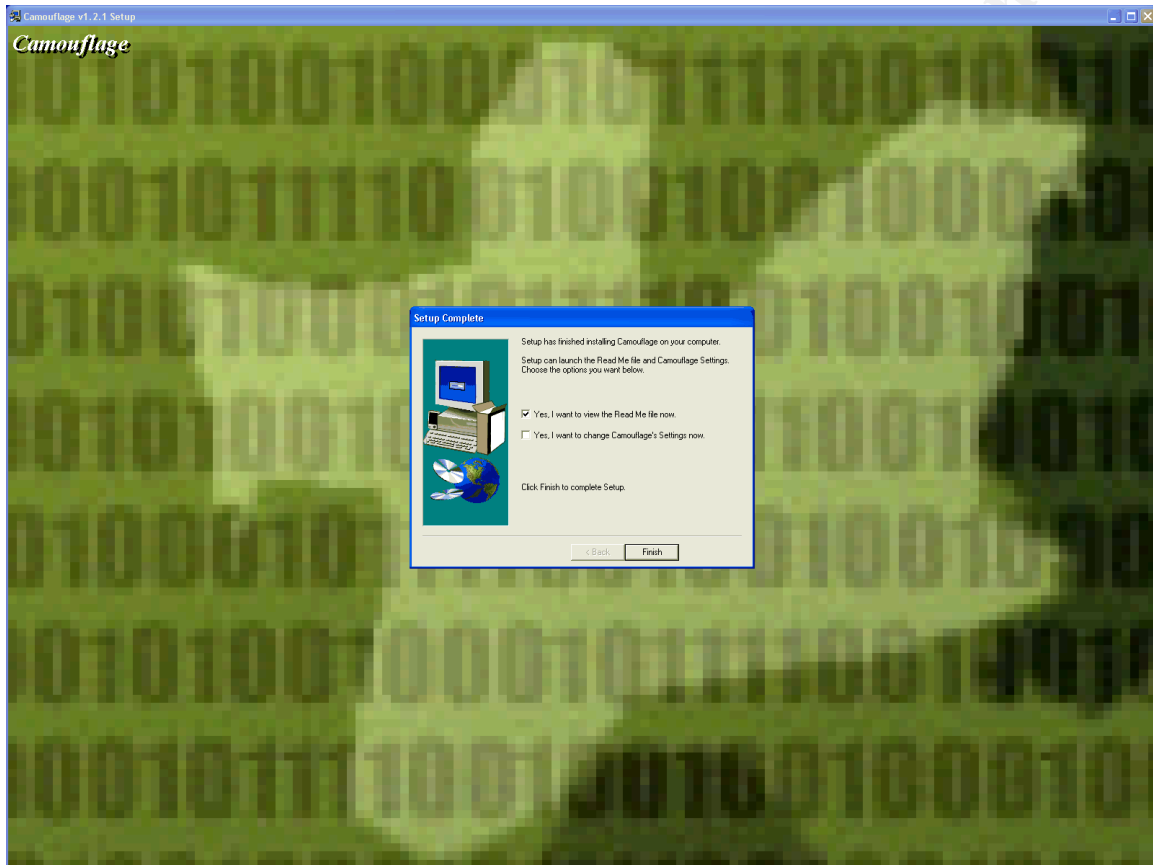
We move over to the windows system and install the software. For this

exercise, we are using an old Dell laptop running windows XP. This box was chosen because it is a spare box and can be removed from the network while we install and use the software we just downloaded. This will protect the rest of the network from any unknown side affects or malware that may accompany the camouflage software.

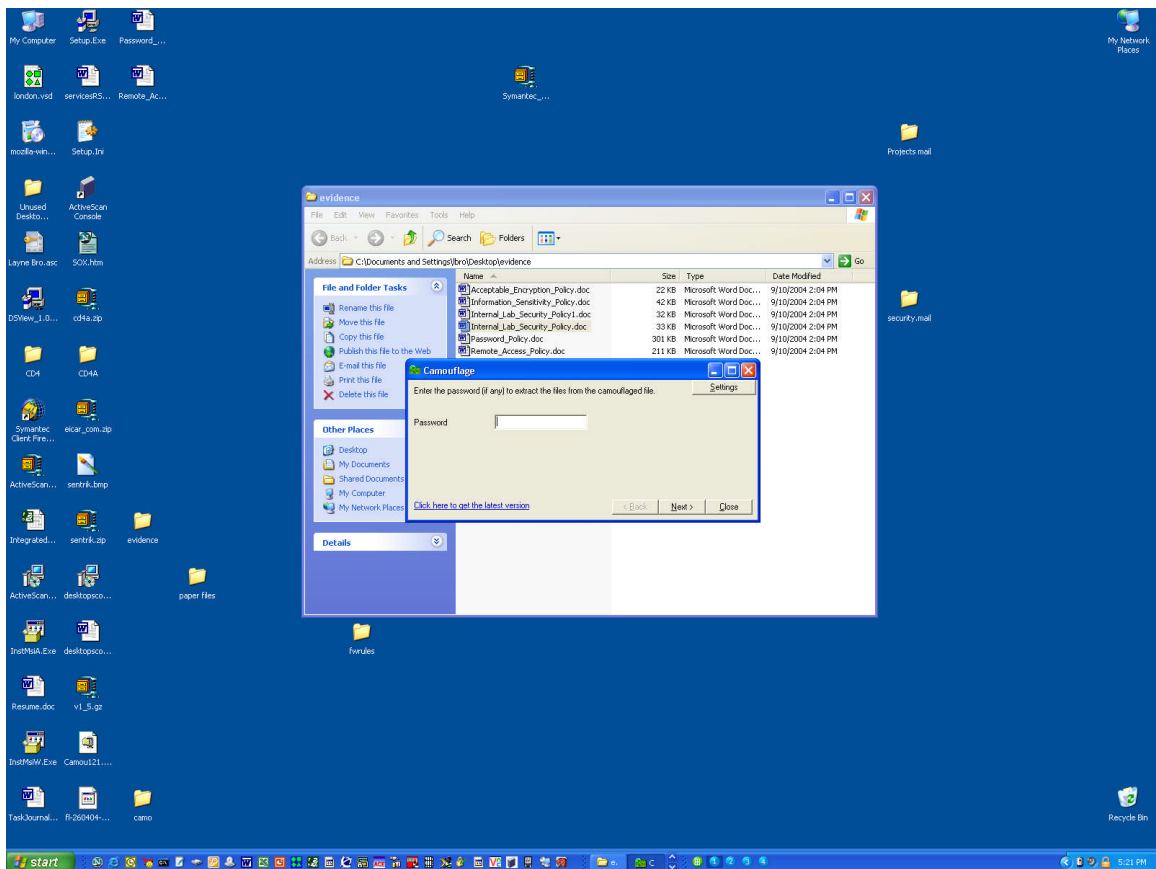
We go ahead and install the software. It seems rather “professional,” not like a traditional hacker tool.



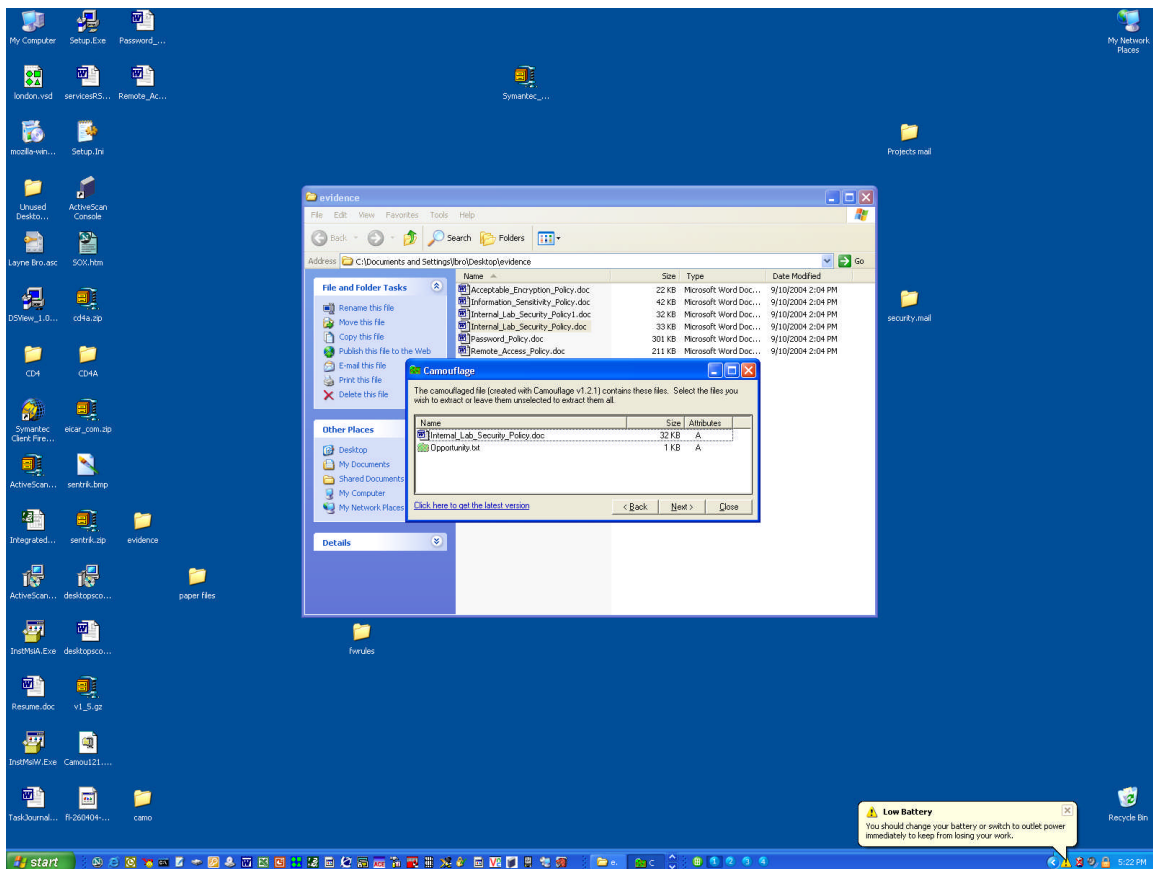
Interesting here at the end, where it allows you to “change camouflage settings now”. From the readme.txt file, it appears that you can rename the menu entries to hide the fact that camouflage is installed. This would be handy to prevent a casual user from noticing that the software has been installed. This is something we’ll need to mention to the Sys Admin group to watch.

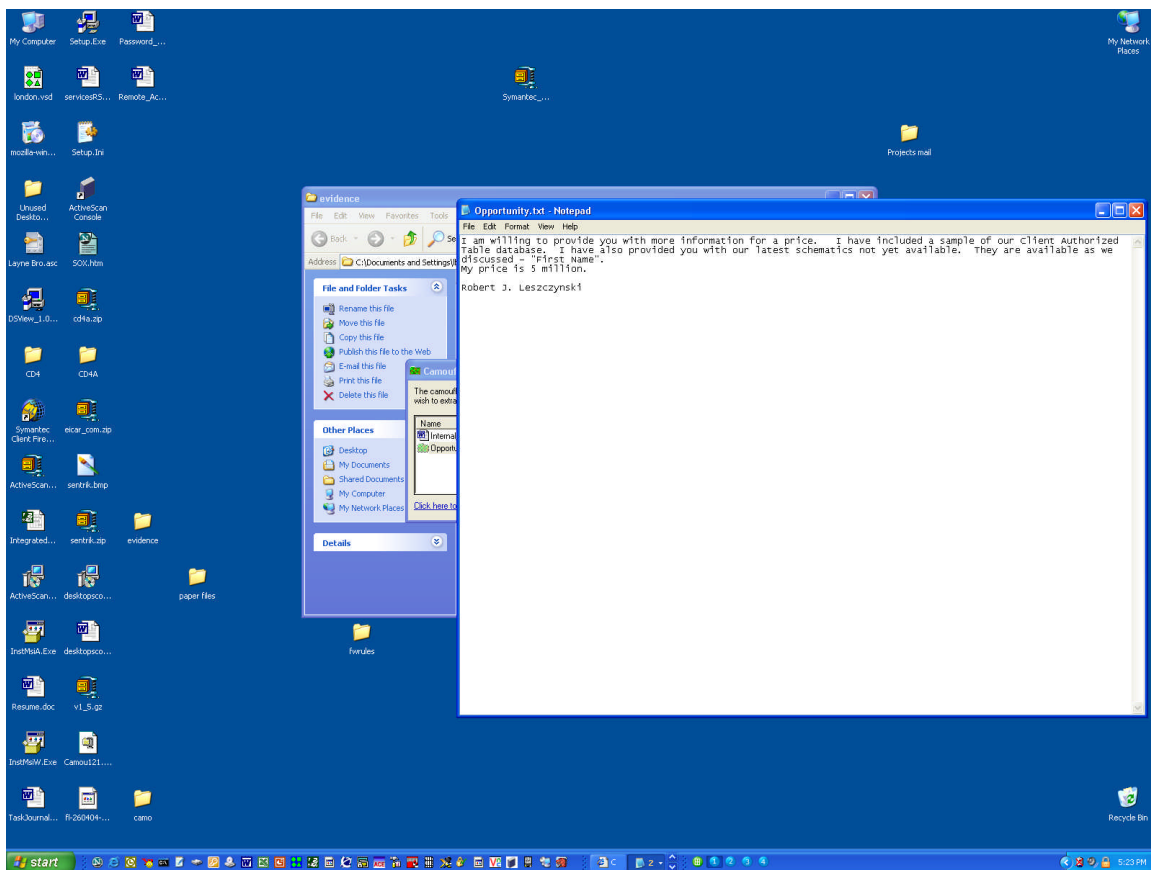


Now we'll go ahead and try to decrypt or "uncamouflage" the 6 files that we found on the disk. It prompts for a password when you select uncamouflage. For the first 3 files, I try a blank password and it fails.



On the fourth file, “Internal_Lab_Policy.doc”, the blank password succeeds. We can see that the actual policy has a file named Opportunity.txt encrypted and hidden inside of the MS Word file. We'll go ahead and extract this file.





I finish trying to uncamouflage the remaining files with a blank password with no success. However, according to the Opportunity.txt file, Robert has also supplied a sample of the Client Authorized Table database, as well as the latest schematics.

At this point, following the idea from "Opportunity.txt" that the other files are available "as we discussed - 'First Name'." I try to uncamouflage the other 5 files with Robert. No success. The software instructions do indicate that the password is case sensitive. I try every derivative of Robert that I can think of. I even try foreign language versions of Robert. No success. So I decide to go back to Autopsy to see if I can find out anything else about the files on this floppy.

First thing I do is to run a strings against the file we were able to Uncamouflage and look for any “tell-tale” signs of the camouflaging process.

I notice that after the last real word in the document that strings identifies, there are several random characters. Next I check one of the other documents – Acceptable_Encryption_Policy.doc. This file actually ends with the word Ballard.

The screenshot shows a web browser window with the address bar displaying `http://localhost:9999/autopsy?mod=1&submod=2&case=GCFA&host=Ballard&inv=Layne`. The browser's address bar also shows `GCFA:Ballard:images/v1_5`. The interface has a top navigation bar with tabs: **FILE ANALYSIS**, **KEYWORD SEARCH**, **FILE TYPE**, **IMAGE DETAILS**, **META DATA**, **DATA UNIT**, **HELP**, and **CLOSE**. Below the navigation bar, there are two main sections: **Directory Seek** and **Current Directory: a:**.

The **Directory Seek** section on the left includes a text input field for the directory name (currently `a:\`), a **VIEW** button, a **File Name Search** section with a text input field for a Perl regular expression, a **SEARCH** button, and buttons for **ALL DELETED FILES** and **EXPAND DIRECTORIES**.

The **Current Directory: a:** section on the right contains a table listing files and directories. The table has columns: **DEL**, **Type**, **NAME**, **WRITTEN**, **ACCESSED**, and **CREATED**. The files listed are:

DEL	Type	NAME	WRITTEN	ACCESSED	CREATED
✓	r/r	_ndex.htm	2004.04.23 10:53:56 (MDT)	2004.04.26 00:00:00 (MDT)	2004.04.26 09:47:36 (MST)
	r/r	Acceptable_Encryption_Policy.doc (ACCEPT-1.DOC)	2004.04.23 14:10:50 (MDT)	2004.04.26 00:00:00 (MDT)	2004.04.26 09:46:44 (MST)
✓	r/r	CamShell.dll (AMSHLL.DLL)	2001.02.03 19:44:16 (MST)	2004.04.26 00:00:00 (MDT)	2004.04.26 09:46:18 (MST)
	r/r	Information_Sensitivity_Policy.doc (INFORM-1.DOC)	2004.04.23 14:11:10 (MDT)	2004.04.26 00:00:00 (MDT)	2004.04.26 09:46:20 (MST)
	r/r	Internal_Lab_Security_Policy.doc (INTERN-2.DOC)	2004.04.22 16:31:06 (MDT)	2004.04.26 00:00:00 (MDT)	2004.04.26 09:46:24 (MST)
	r/r	Internal_Lab_Security_Policy1.doc (INTERN-1.DOC)	2004.04.22 16:31:06 (MDT)	2004.04.26 00:00:00 (MDT)	2004.04.26 09:46:22 (MST)

Below the table, there are buttons for **ADD NOTE** and **GENERATE MD5 LIST OF FILES**. Below the buttons, there is a section for **ASCII (display - report) * ASCII Strings (display - report) * Export * Add Note**. The **File Type: Microsoft Office Document** is displayed. The **Acceptable Encryption Policy.doc** file is selected, and its contents are displayed in the bottom section:

```
Acceptable Encryption Policy
Normal.dot
Microsoft Word 10.0
Ballard
Cisco Systems, Inc.
Acceptable Encryption Policy
Title
Microsoft Word Document
MSWordDoc
Word.Document.8
Acceptable Encryption Policy
Normal.dot
Microsoft Word 10.0
Ballard
Acceptable Encryption Policy
Title
Ballard Industries, Inc.
```

I check the other files looking for any pattern. It turns out that Internal_Security_Policy.doc, Password_Policy.doc, and Remote_Access_Policy.doc all have random data after the Company name that MS Word lists in the file properties as the "Company" for this document "Ballard Industries Inc." The other documents all end at the "Ballard Industries Inc." In fact, Password_Policy.doc and Remote_Access_Policy.doc both have lots of random data after this last MS Word entry of "Ballard Industries, Inc." This leads me to believe that these two documents both have the other images camouflaged inside of them.

The screenshot shows a web browser window with the address bar displaying 'http://localhost:9999/autopsy?mod=1&submod=2&case=GCFA&host=Ballard&inv=Layne'. The browser's address bar also shows 'GCFA:Ballard:images/v1_5'. The browser's menu bar includes 'The Sleuth', 'January 2004', 'Available Ports', 'Netscape.com', 'Mac Forensics', 'Fink', 'Apple', and 'News'. The browser's status bar shows 'GCFA:Ballard:images/v1_5'.

The main content area displays a file analysis tool with a tabbed interface. The tabs are 'FILE ANALYSIS', 'KEYWORD SEARCH', 'FILE TYPE', 'IMAGE DETAILS', 'META DATA', 'DATA UNIT', 'HELP', and 'CLOSE'. The 'FILE ANALYSIS' tab is selected.

The 'FILE ANALYSIS' tab shows a directory listing of files. The listing includes a 'Directory Seek' section on the left with a search box and a 'VIEW' button. Below this is a 'File Name Search' section with a search box and a 'SEARCH' button. At the bottom of the left sidebar are buttons for 'ALL DELETED FILES' and 'EXPAND DIRECTORIES'.

The main table lists files with columns for file name, date, and time. The files listed are:

File Name	Date	Time
r/r _ndex.htm	2004.04.23	10:53:56 (MDT)
r/r Acceptable_Encryption_Policy.doc (ACCEPT-1.DOC)	2004.04.23	14:10:50 (MDT)
r/r CamShell.dll (AMSHLL.DLL)	2001.02.03	19:44:16 (MST)
r/r Information_Sensitivity_Policy.doc (INFORM-1.DOC)	2004.04.23	14:11:10 (MDT)
r/r Internal_Lab_Security_Policy.doc (INTERN-2.DOC)	2004.04.22	16:31:06 (MDT)
r/r Internal_Lab_Security_Policy1.doc (INTERN-1.DOC)	2004.04.22	16:31:06 (MDT)
r/r Password_Policy.doc (PASSWO-1.DOC)	2004.04.23	11:55:26 (MDT)
r/r Remote_Access_Policy.doc (REMOTE-1.DOC)	2004.04.23	11:54:32 (MDT)
r/r RJL (Volume Label Entry)	2004.04.25	10:53:40 (MDT)

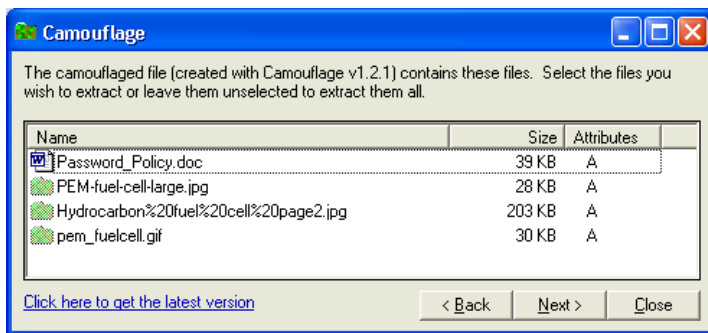
Below the table, there are links for 'ASCII (display - report)', 'ASCII Strings (display - report)', 'Export', and 'Add Note'. The file type is listed as 'File Type: Microsoft Office Document'.

The bottom section shows the metadata for the selected file, 'InfoSec Password Policy'. The metadata includes:

- Title: InfoSec Password Policy
- Microsoft Word Document
- MSWordDoc
- Word.Document.8
- InfoSec Password Policy
- Normal.dot
- Microsoft Word 10.0
- Ballard
- s42v
- Uy>k
- d{Hb
- _Lerni
- oQjgm
- 5HXN
- RV_B
- !61'
- yA}s>v1
- 0|2
- 5&|2'D|y

So I go back to trying to hack the password in Camouflage. This time I specifically target the two files I've identified as likely containing the hidden data. I once again try every spelling of Robert that I can think of. Then I try the First Names of the Companies Involved. I spend time searching the notes to see if we have any other names I can try. At this point, I decide that the First Name must be of something other than a person. Finally it dawns on me that the file names are names. I try the first word of each respective file name Password for Password_Policy.doc – and I have success. I can now pull out the other data that has been hidden.

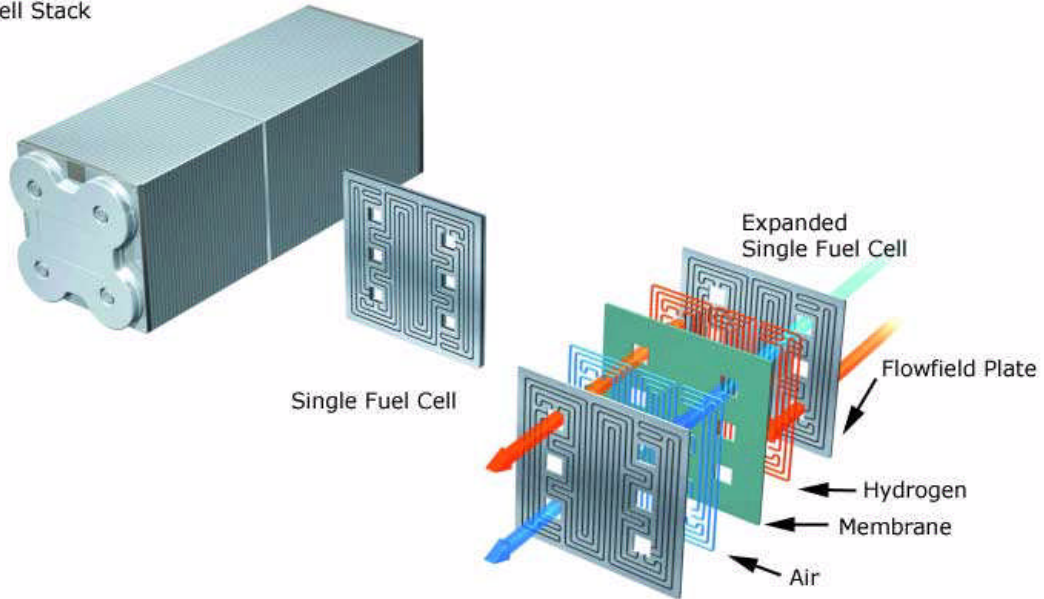
First we see the files hidden in Password_Policy.doc



And we can open them up and see their contents.

Design of a PEM Fuel Cell

Fuel Cell Stack



© SANS Institute 2000 - 2005

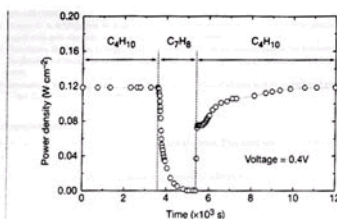


Figure 3 Effect of switching fuel type on the cell with the Cu-ceria composite anode at 973 K. The power density of the cell is shown as a function of time. The fuel was switched from *n*-butane (C_4H_{10}) to toluene (C_7H_8) and back to *n*-butane.

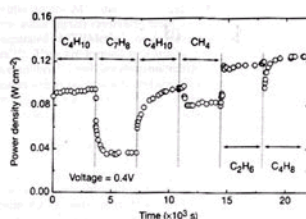
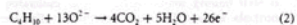
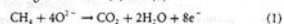


Figure 4 Effect of switching fuel type on the cell with the Cu-doped ceria composite anode at 973 K. The power density is shown as a function of time. The fuels were: *n*-butane (C_4H_{10}), toluene (C_7H_8), *n*-butane, methane (CH_4), ethane (C_2H_6), and 1-butene (C_4H_8).

higher temperature. Visual inspection of a cell after two days in *n*-butane at 1,073 K showed that the anode itself remained free of the tar deposits that covered the alumina walls.

Although it is possible that the power generated from *n*-butane fuels resulted from oxidation of H_2 —formed by gas-phase reactions of *n*-butane that produce hydrocarbons with a lower C:H ratio—other evidence shows that this is not the case. First, experiments were conducted in which the cell was charged with *n*-butane and then operated in a batch mode without flow. After 30 minutes of batch operation with the cell short-circuited, GC analysis showed that all of the *n*-butane in the cell had been converted completely to CO_2 and water. (Negligible amounts of CO_2 were formed in a similar experiment with an open circuit.) Second, analysis of the CO_2 formed under steady-state flow conditions, shown in Fig. 2, demonstrates that the rate of CO_2 formation increased linearly with the current density. (It was not possible for us to quantify the amount of water formed in our system.) Figure 2 includes data for both *n*-butane at 973 K, and methane at 973 K and 1,073 K. The lines in the figure were calculated assuming complete oxidation of methane (the dashed line) and *n*-butane (the solid line) to CO_2 and water according to reactions (1) and (2):



With methane, only trace levels of CO were observed along with CO_2 , so that the agreement between the data points and the calculation demonstrates consistency in the measurements and no leaks in the cell. With *n*-butane, simultaneous, gas-phase, free-radical reactions to give hydrocarbons with various C:H ratios make quantification more difficult; however, the data still suggest that complete oxidation is the primary reaction. Furthermore, the batch experiments show that the secondary products formed by gas-phase reactions are ultimately oxidized as well. Taken together, these results demonstrate the direct, electrocatalytic oxidation of a higher hydrocarbon in a SOFC.

Along with our observation of stable power generation with *n*-butane for 48 hours, Fig. 3 further demonstrates the stability of the composite anodes against coke formation. Aromatic molecules, such as toluene, are expected to be precursors to the formation of graphitic coke deposits. In Fig. 3, the power density was measured at 973 K and 0.4 V while the fuel was switched from dry *n*-butane, to 0.033 bar of toluene in He for 30 minutes, and back to dry *n*-butane. The data show that the performance decreased rapidly in the presence of toluene. Upon switching back to dry *n*-butane, however,

the current density returned to 0.12 W cm^{-2} after one hour. Because the return was not instantaneous, it appears that carbon formation occurred during exposure to toluene, but that the anode is self-cleaning. We note that the electrochemical oxidation of soot has been reported by others¹¹.

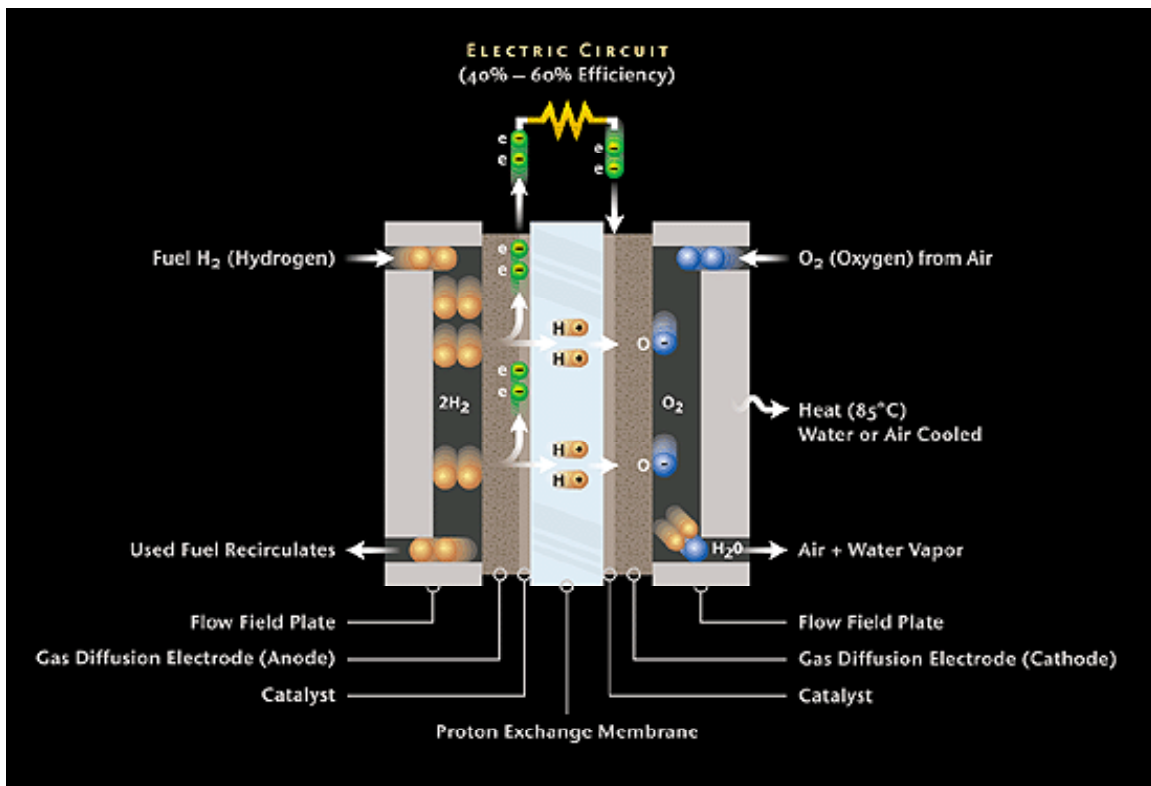
The data in Fig. 4 show that further improvements in cell performance can be achieved. For these experiments, samaria-doped ceria was substituted for ceria in the anode, and the current densities were measured at a potential of 0.4 V at 973 K. The power densities for H_2 and *n*-butane in this particular cell were approximately 20% lower than for the first cell, which is within the range of our ability to reproduce cells. However, the power densities achieved for some other fuels were significantly higher. In particular, stable power generation was now observed for toluene. Similarly, Fig. 4 shows that methane, ethane and 1-butene could be used as fuels to produce electrical energy. The data show transients for some of the fuels, which are at least partially due to switching.

The role of samaria in enhancing the results for toluene and some of the other hydrocarbons is uncertain. While samaria is used to enhance mixed (ionic and electronic) conductivity in ceria and could increase the active, three-phase boundary in the anode, samaria is also an active catalyst¹². Other improvements in the performance of SOFCs are possible. For example, the composite anodes could be easily attached to the cathode-supported, thin-film electrolytes that have been used by others to achieve very high power densities⁴. In addition to raising the power density, thinner electrolytes may also allow lower operating temperatures.

Additional research is clearly necessary for commercial development of fuel cells which generate electrical power directly from hydrocarbons; however, the work described here suggests that SOFCs have an intriguing future as portable, electric generators and possibly even as energy sources for transportation. The simplicity afforded by not having to reform the hydrocarbon fuels is a significant advantage of these cells. □

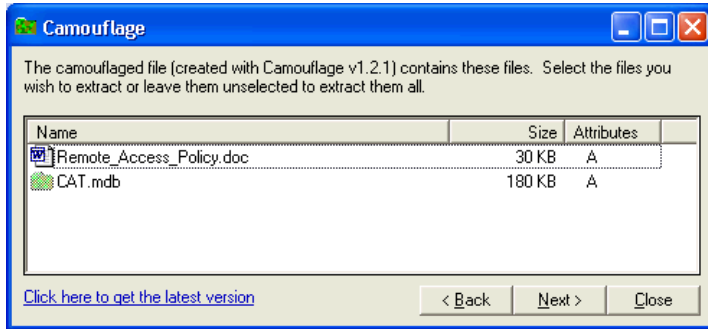
Received 12 September 1999; accepted 26 January 2000.

1. Steele, B. C. H. Burning on natural gas. *Nature* **400**, 620–621 (1999).
2. Service, R. F. Bringing fuel cells down to earth. *Science* **285**, 682–685 (1999).
3. Perry Murray, E., Tsai, T. & Barnett, S. A. A direct-methane fuel cell with a ceria-based anode. *Nature* **400**, 640–651 (1999).
4. Perna, E. S., Strubenschütz, J., Vohs, J. M. & Gorte, R. J. Ceria-based anodes for the direct oxidation of methane in solid oxide fuel cells. *Langmuir* **11**, 4832–4837 (1995).
5. Park, S., Craciun, R., Vohs, J. M. & Gorte, R. J. Direct oxidation of hydrocarbons in a solid oxide fuel cell I: methane oxidation. *J. Electrochem. Soc.* **146**, 3603–3605 (1999).
6. Steele, B. C. H., Kelly, L., Middleton, P. H. & Radkin, B. Oxidation of methane in solid-state electrochemical reactors. *Solid State Ionics* **28**, 1547–1552 (1988).
7. Lloyd, A. C. The power plant in your basement. *Sci. Am.* **281**(1), 80–86 (1999).



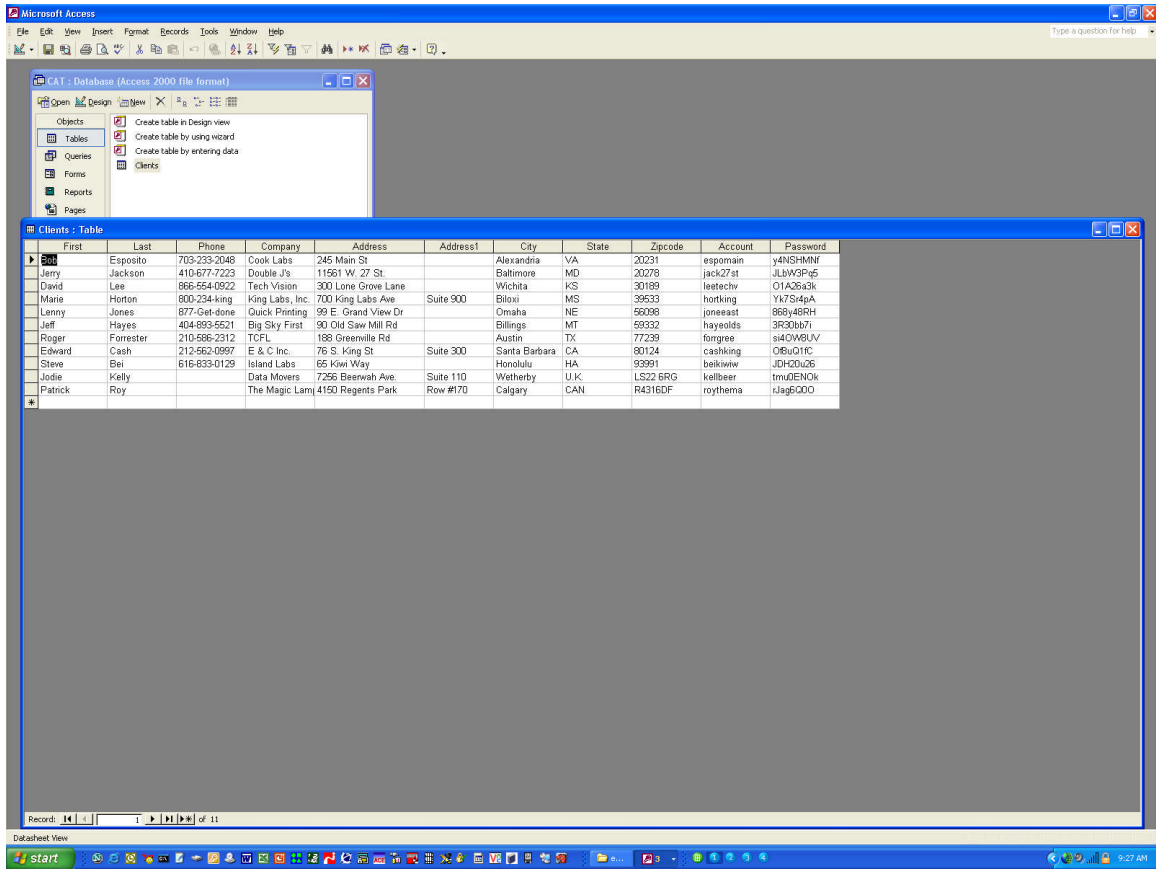
© SANS Institute 2000 - 2005

Then we can see the files hidden in Remote_Access_Policy.doc

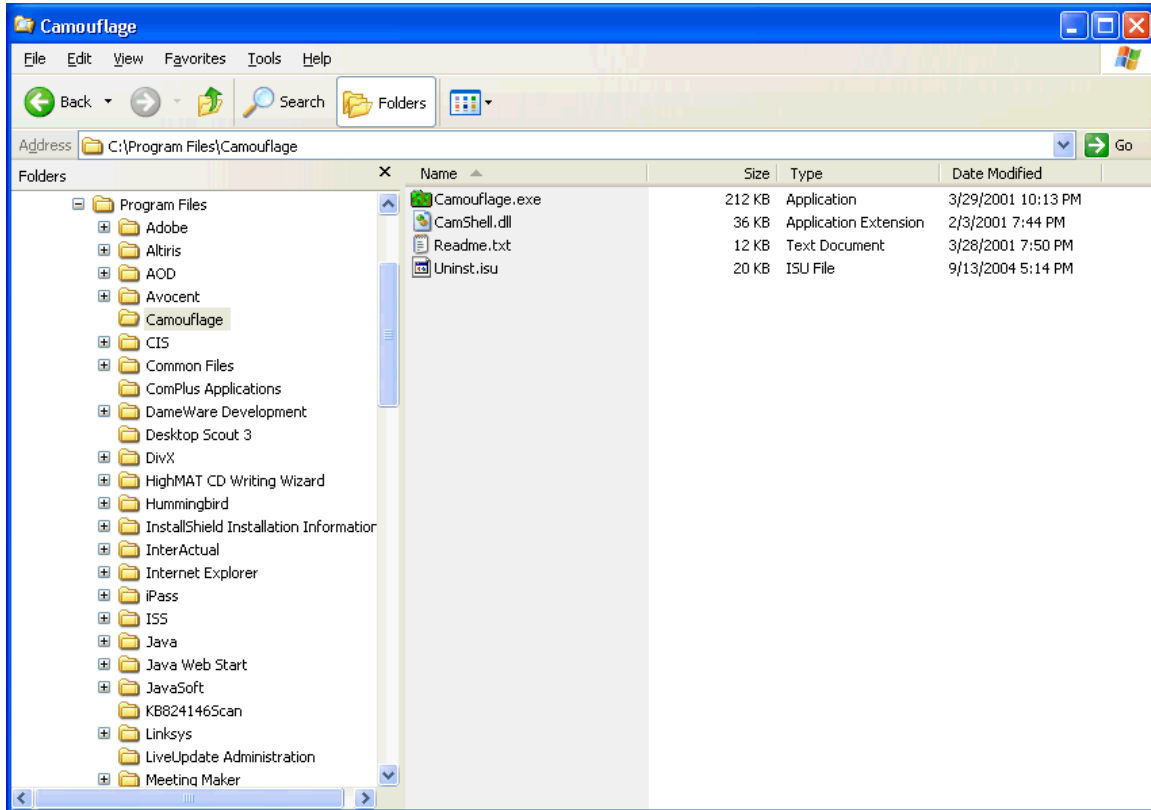


© SANS Institute 2000 - 2005

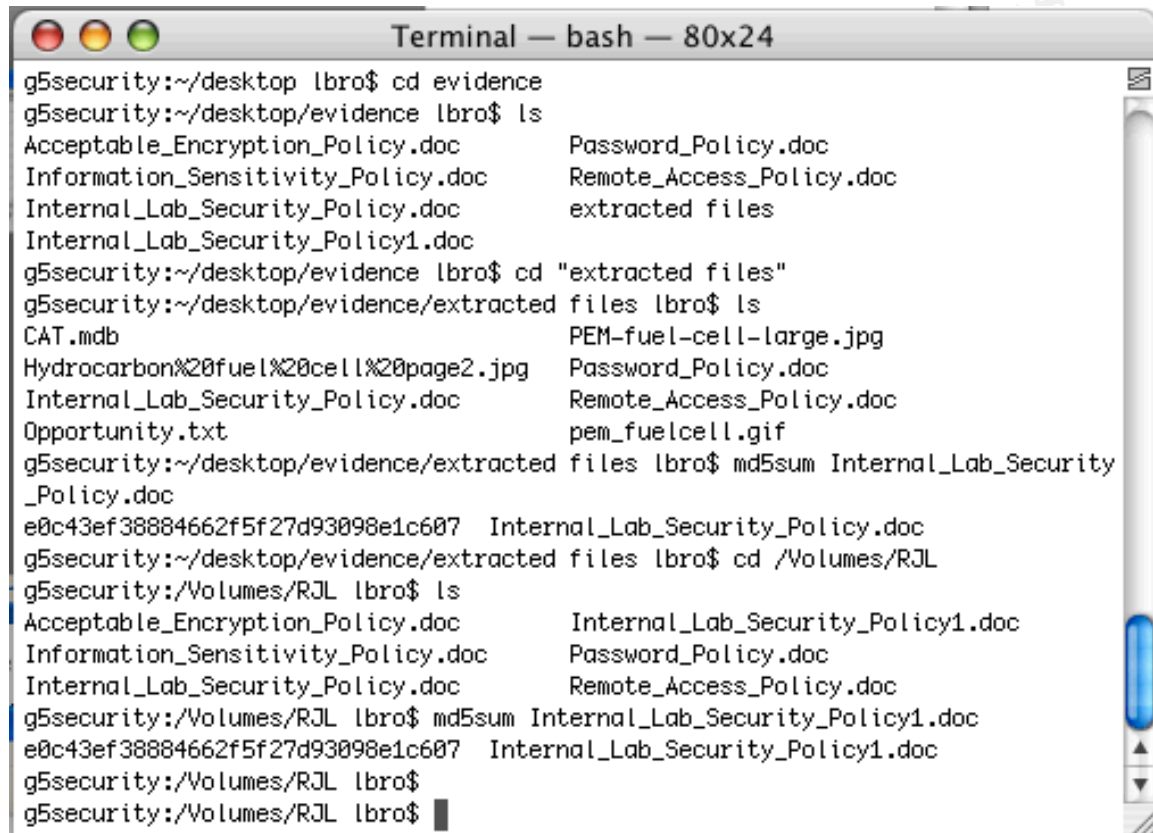
And we can open it up also.



Now I want to capture the files used by the Camouflage program.



It would be interesting to see if the file Internal_Lab_Security_Policy1.doc is identical to the Internal_Lab_Security_Policy.doc file that we extracted from within Internal_Lab_Security_Policy.doc. So we do an md5sum for comparison.

A screenshot of a macOS Terminal window titled "Terminal — bash — 80x24". The window shows a series of commands and their outputs. The user starts in the directory ~/desktop and navigates to a folder named 'lbrow'. They list the contents of 'lbrow', which includes several .doc files and a folder named 'extracted files'. They then navigate into 'extracted files' and list its contents, which includes a .mdb file, a .jpg file, another .doc file, another .doc file, a .txt file, and a .gif file. Next, they run an md5sum command on 'Internal_Lab_Security_Policy.doc' in the 'extracted files' directory, which returns the hash 'e0c43ef38884662f5f27d93098e1c607'. They then navigate to the root volume '/Volumes/RJL' and list its contents, which includes the same set of files as the 'lbrow' directory. Finally, they run an md5sum command on 'Internal_Lab_Security_Policy1.doc' in the '/Volumes/RJL' directory, which also returns the hash 'e0c43ef38884662f5f27d93098e1c607', confirming the files are identical.

```
g5security:~/desktop lbrow$ cd evidence
g5security:~/desktop/evidence lbrow$ ls
Acceptable_Encryption_Policy.doc      Password_Policy.doc
Information_Sensitivity_Policy.doc     Remote_Access_Policy.doc
Internal_Lab_Security_Policy.doc       extracted files
Internal_Lab_Security_Policy1.doc
g5security:~/desktop/evidence lbrow$ cd "extracted files"
g5security:~/desktop/evidence/extracted files lbrow$ ls
CAT.mdb                               PEM-fuel-cell-large.jpg
Hydrocarbon%20fuel%20cell%20page2.jpg Password_Policy.doc
Internal_Lab_Security_Policy.doc       Remote_Access_Policy.doc
Opportunity.txt                        pem_fuelcell.gif
g5security:~/desktop/evidence/extracted files lbrow$ md5sum Internal_Lab_Security
_Policy.doc
e0c43ef38884662f5f27d93098e1c607 Internal_Lab_Security_Policy.doc
g5security:~/desktop/evidence/extracted files lbrow$ cd /Volumes/RJL
g5security:/Volumes/RJL lbrow$ ls
Acceptable_Encryption_Policy.doc      Internal_Lab_Security_Policy1.doc
Information_Sensitivity_Policy.doc     Password_Policy.doc
Internal_Lab_Security_Policy.doc       Remote_Access_Policy.doc
g5security:/Volumes/RJL lbrow$ md5sum Internal_Lab_Security_Policy1.doc
e0c43ef38884662f5f27d93098e1c607 Internal_Lab_Security_Policy1.doc
g5security:/Volumes/RJL lbrow$
g5security:/Volumes/RJL lbrow$
```

The original file that was used to hide the demand letter from Robert is identical to another copy of the file on the disk.

This makes sense.

Robert Leszczynski downloaded a copy of Camouflage to enable him to hide the data he was trying to steal. He used the Camouflage software to take the schematics, encrypt them, and hide them inside of the Password_Policy.doc file. He also used Camouflage to encrypt and hide a sample of the Customer table inside of the Remote_Access_Policy.doc file. Finally, he created a simple notepad doc with his demands and offerings, encrypted it without a password, and hide it inside of the Internal_Lab_Security_Policy.doc. He then went and set all of the access and modify times on all of the policy docs to be identical – presumably to prevent anyone from noticing that they were different than the original policy documents.

It appears that Robert had made initial contact with someone to sell them the data to recreate this unique battery and also a list of the current customers.

However, I found no evidence that Robert had actually given them any data yet.

One frustrating factor about this case is my lack of ability to extract the program or anything else from the camshell.dll file. I ran a file command against this recovered file. It is listed as an html document, not a dll. Also, the deleted file index.htm is identical to the first 2 sectors of the camshell.dll file. In fact, index.htm and camshell.dll both start at sector 33 on the disk. Index.html is only in sectors 33 and 34, while camshell.dll continues on through sector 104. I tried extracting both files and uncamouflaging the camshell.dll file, however this didn't work with any password I could generate. I took another look at the strings output of this file, and had an unusual layout.

Sectors 33 – 34 html codes

Sectors 35 – 40 blank

Sectors 41 – 45 Data of some type

Sectors 46 - 78 Program data

Sectors 79 – 88 Blank

Sectors 89 – 100 data of some type

Sectors 101 – 104 blank.

After reviewing these sectors manually, it doesn't really appear that the camouflage.exe program was camouflaged and put inside of the index.htm file – you can still see the Visual Basic commands of the program. However, the sectors with raw data are interesting and I cannot currently explain them. Given more resources and time, and potentially access to more of Mr. Leszczynskis equipment, it would be reasonable to assume that we could extract more of the deleted data and develop even more of the picture.

Forensic Details:

Mr. Leszczynski used a program called Camouflage.exe to hide data inside of other files. This is a type of Steganography, which is defined as

“The art and science of hiding information by embedding messages within other, seemingly harmless messages. Steganography works by replacing bits of useless or unused data in regular computer files (such as graphics, sound, text, HTML, or even floppy disks) with bits of different, invisible information. This hidden information can be plain text, cipher text, or even images
(<http://www.webopedia.com/TERM/S/steganography.html>).

This program was used by Mr. Leszczynski sometime between April 22, 2004 @ 4:31PM mountain time and April 26th, 2004 at approximately 9:46AM Mountain time. In this time frame, Robert created the files used to hide data within, then encrypted and hid data within the original files, and then modified the last accessed times to make them all appear the same and hide which files he had actually modified and when. This program will take a file, and a password if desired, and encode them into the end of an existing document. Most software, like Microsoft Word, will only read the file until the MS end of file code and ignore all data after that point. The Camouflage program takes advantage of this “feature” of many software packages. Camouflage takes the file you want to hide, encrypts it along with the password you used, if any, and appends the data onto the end of an existing file. This is why the files with camouflaged data attached are larger than their original size. Also, the md5 sums of the original file and the same file with camouflaged data don’t match. In a file view under autopsy that displays all data of the file, we were able to see data after the end of the Microsoft Word formatted section of the file. This was a clue as to which files contained hidden data.

Program Identification:

The version of this program that we used was downloaded from camouflage.unfiction.com and not from <http://www.camouflage.freemove.co.uk> because this site is no longer available. We were able to download and use the file. Once I uncamouflaged the data from inside of Internal_Lab_Security_Policy.doc, I compared the now original Internal_Lab_Security_Policy.doc file with the Internal_Lab_Security_Policy1.doc file, which was believed to be the actual original from the company servers. Both of these files were identical as indicated by an md5sum comparison in the course of the investigation. This, along with the ability of the camouflage program I downloaded to uncamouflage files and reveal the version of camouflage used to create the camouflaged files

shows that this was the program used by Mr. Leszczynski.

Legal Implications:

Among any other laws that may have been broken by Mr. Leszczynski, we need to be concerned about California Senate Bill 1386. This bill specifically applies to personal data that is accessed or obtained by non-authorized persons. Quoting from the California state government website:

This bill, operative July 1, 2003, would require a state agency, or a person or business that conducts business in California, that owns or licenses computerized data that includes personal information, as defined, to disclose in specified ways, any breach of the security of the data, as defined, to any resident of California whose unencrypted personal information was, or is reasonably believed to have been, acquired by an unauthorized person. (http://info.sen.ca.gov/pub/01-02/bill/sen/sb_1351-1400/sb_1386_bill_20020926_chaptered.html)

Because Mr. Leszczynski was not authorized to work with customer data, we could be in violation of this law. To verify our violation, we simply have to look and see if any of our customers reside in California. We find that yes, Edward Cash is a customer from Santa Barbara California. To further investigate this potential issue, we continue with the law:

(e) For purposes of this section, "personal information" means an individual's first name or first initial and last name in combination with any one or more of the following data elements, when either the name or the data elements are not encrypted:

- (1) Social security number.*
- (2) Driver's license number or California Identification Card number.*
- (3) Account number, credit or debit card number, in combination with any required security code, access code, or password that would permit access to an individual's financial account.*

We do appear to have the full name and a password that was compromised by Mr. Leszczynski. So I believe we are in violation of this law and need to work with our legal team to determine the specific impact and violation, along with our response.

Additional Information:

California State Government SB1386 Law:

http://info.sen.ca.gov/pub/01-02/bill/sen/sb_1351-1400/sb_1386_bill_20020926_chaptered.html

Camouflage Home Page:

<http://camouflage.unfiction.com/>

Definition of steganography:

<http://www.webopedia.com/TERM/S/steganography.html>

© SANS Institute 2000 - 2005, Author retains full rights.

Citations

Definition of Steganography:

Webopedia. <http://www.webopedia.com/TERM/S/steganography.html>

California Senate Bill 1386:

Senator Peace, California Senate Bill 1386, Version dated Feb 12, 2002,
http://info.sen.ca.gov/pub/01-02/bill/sen/sb_1351-1400/sb_1386_bill_20020926_chaptered.html

© SANS Institute 2000 - 2005, Author retains full rights.

Part 2 Option 2

Scope:

This forensic tool validation will cover ssh as a data collection tool – specifically, using SSH as the data transport method of sending a disk image over a network to another system. SSH is Secure Shell and is used to make encrypted connections to servers running an SSH daemon. This test will cover the ability of an SSH client to transmit a disk image to a remote system over a network similar to methods utilizing netcat, windows shares, or other traditional processes. This test will show that using SSH as our transport method is accurate.

Tool Description:

SSH is being analyzed because it has many advantages over the other methods of transporting a bit level disk image from one computer to another, but I have never seen it listed as an image transport option.

SSH can be obtained from both commercial vendors and from open source sites. OpenSSH, an open source version, can be obtained from <http://www.openssh.org/>. This site has versions available for OpenBSD, FreeBSD, NetBSD, Linux, Solaris, AIX, IRIX, HP-UX and many other operating systems. The Helix CD also contains an SSH client. For this test, I'll be using an intel laptop, booted from the Helix 1.4 CD running OpenSSH 3.8 to create my image and sending it over a network to Mac OS X server running OpenSSH 3.6.1.

SSH is as fast of a transport method as Netcat without all the hassles of having to connect to the remote server and setup a netcat listener on a specific port. SSH is much faster than a windows share mapping, just like netcat.

By simplifying the setup of transporting an image to a specific server, many individuals can start the image creation process without the manual involvement of setting up. In many environments, non-security personnel must initiate the disk imaging process. The use of SSH is simpler to setup than netcat and much faster to complete than use of a windows share.

By speeding up the time it takes to collect an image, the Forensic analyst can start his analysis sooner. It also allows a mission critical system to be put back online faster after making the image or be available for recovery faster after making the image. In this day of always-available systems, it is important to return a company to normal operations as quickly as possible.

Use of SSH as our transport method will improve our response times both for the investigation and for the company to recover from an incident and allow us to use non-authorized and potentially non-trained individuals at the incident location to do the data collection.

Test Apparatus:

For my testing, I'll be using the SSH client that is already built into Helix 1.4 bootable CD. I will treat the intel based laptop as a system that has been compromised but is dead. So a static image can be taken by booting from the helix CD. The remote host accepting the disk image is my primary analysis workstation running Mac OS X 10.3.5 with the built in OpenSSH Server. This test will be limited to the SSH Client built into the Helix 1.4 CD.

To monitor the time expired for each image transmission, I will be using an atomic clock. This clock is synced with the NIST clock in boulder Colorado. This is the most accurate clock available for this testing (no stopwatch was available). All tests will have their start time and finish times, as defined by the atomic clock, recorded by the tester. We will also record the manual steps required to setup the transport method.

Environmental Conditions:

The connectivity for this test will be over a Local Area Network with a Cisco 2400 switch between the two computers. These two systems will be the only traffic on the switch. By limiting the switch to only traffic between these two computers, and using the same client and server for all my testing, we will be able to accurately compare the results. There should be no outside impacts to the time it takes to move data between the two systems. The only factor that will change is the method used to transport the data over the network, either a netcat connection, a windows share, or an SSH connection.

Description of the Procedure:

Target system is a standard Dell Latitude C 800 System with Windows XP installed. This system will be booted from the Helix 1.4 CD. The analysis workstation will be receiving the image directly from a dd command. This analysis system is a Mac Dual G5 desktop system with 4 G RAM, 1 x 120 G Internal hard drive containing the OS and 1 x 500 G external Lacie Bigger disk for capturing and analyzing disk images. The only activity on the G5 will be the capture of the disk image from the target system. The boxes will be connected to a simple network with a Cisco 2400 switch. We have verified that the G5 is running OpenSSH 3.6.1 and the Helix CD has OpenSSH 3.8 installed.

The process will be to make a single disk image of the entire disk, not to make individual partition images. Before beginning the disk image, we will calculate an md5sum for the drive and once the image is completed we will

calculate an md5sum for the disk image created on the G5. These two sums must be identical.

Criteria for Approval:

For the test results to be considered valid, the md5sum of the original disk and the disk images must be identical. If the sums do not match, the test is considered invalid and the results will be discarded.

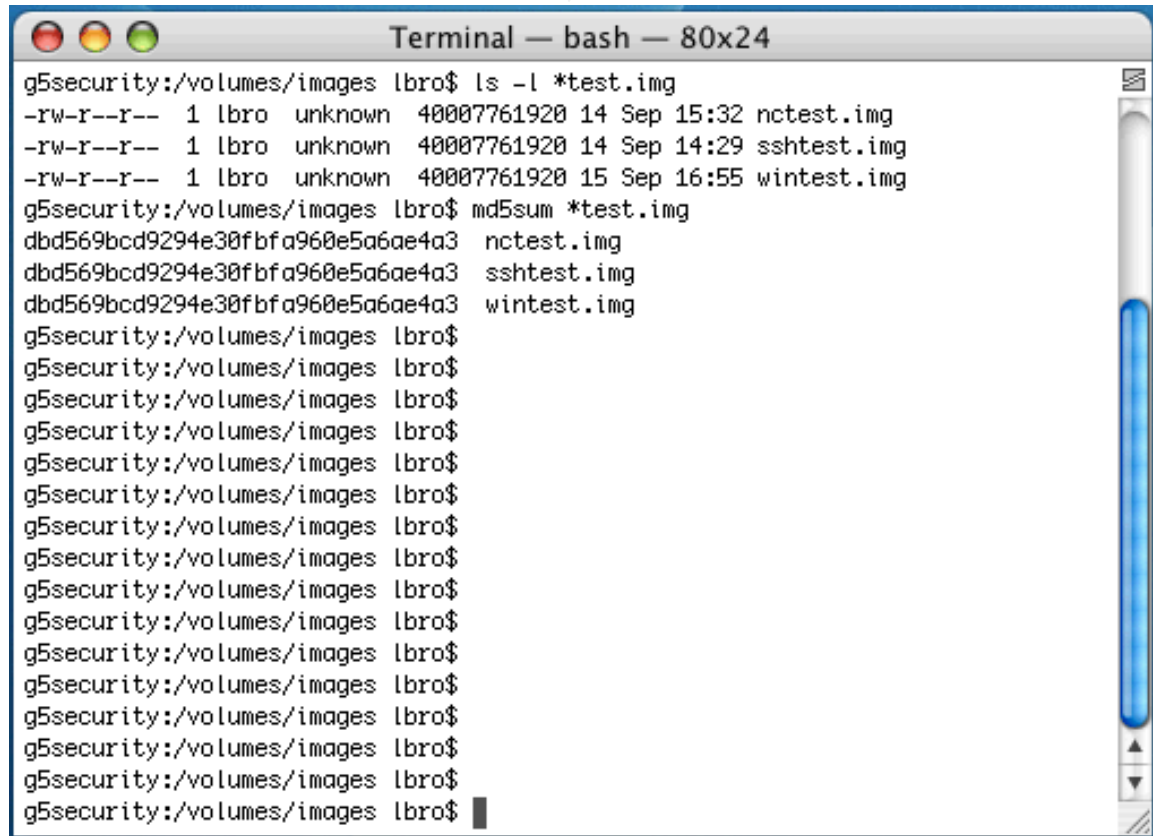
The tests will give us data in the form of “Amount of data” transferred in “Amount of Time.” Since we are keeping the amount of data consistent, we will be mainly concerned with the amount of time required for transmission of the images from the target to the host. We will also be concerned with the amount of manual intervention to setup the image transport. A configuration requiring no interaction by the image creator will be favored over one that requires manual steps to receive the disk image.

Data and Results:

Verification of md5sums.

Original md5sum: dbd569bcd9294e30fbfa960e5a6ae4a3

Disk images:

A screenshot of a terminal window titled "Terminal — bash — 80x24". The terminal shows a user at the prompt "g5security:/volumes/images lbro\$" running the command "ls -l *test.img". The output lists three files: "nctest.img", "sshtest.img", and "wintest.img", all with permissions "-rw-r--r--", owner "lbro", and size "40007761920". The user then runs "md5sum *test.img", and the output shows that all three files have the same md5sum: "dbd569bcd9294e30fbfa960e5a6ae4a3". The user then enters several empty prompts, which are not followed by any output.

```
g5security:/volumes/images lbro$ ls -l *test.img
-rw-r--r--  1 lbro  unknown  40007761920  14 Sep 15:32 nctest.img
-rw-r--r--  1 lbro  unknown  40007761920  14 Sep 14:29 sshtest.img
-rw-r--r--  1 lbro  unknown  40007761920  15 Sep 16:55 wintest.img
g5security:/volumes/images lbro$ md5sum *test.img
dbd569bcd9294e30fbfa960e5a6ae4a3  nctest.img
dbd569bcd9294e30fbfa960e5a6ae4a3  sshtest.img
dbd569bcd9294e30fbfa960e5a6ae4a3  wintest.img
g5security:/volumes/images lbro$
g5security:/volumes/images lbro$
g5security:/volumes/images lbro$
g5security:/volumes/images lbro$
g5security:/volumes/images lbro$
g5security:/volumes/images lbro$
g5security:/volumes/images lbro$
g5security:/volumes/images lbro$
g5security:/volumes/images lbro$
g5security:/volumes/images lbro$
g5security:/volumes/images lbro$
g5security:/volumes/images lbro$
g5security:/volumes/images lbro$
g5security:/volumes/images lbro$
g5security:/volumes/images lbro$
g5security:/volumes/images lbro$
g5security:/volumes/images lbro$
```

nctest is the Netcat test, sstest is the ssh test, and wintest is the windows share test

Image size: 40 G (40,007,761,920 bytes)

Netcat time to complete: 58 minutes 26 seconds (2:36:35 to 3:35:01)

SSH time to complete: 68 minutes 47 seconds (1:22:06 to 2:30:53)

Windows Share time to complete: 7 hours, 12 minutes, 37 seconds (3:45:00 to 10:57:37)

Windows share manual steps:

Possible to do with no manual setup

dd if=/dev/had of=\\g5security\images\WinTest.img

Netcat manual steps:

Access to the destination server is required.

Authority to issue netcat command is required.

Coordination of port chosen for listener on destination system and port used as destination from target system is required.

dd if=/dev/hda | nc g5security 4747

Prior work on g5security:

Nc -l -p 4747 > nctest.img

SSH Manual steps:

Possible to do with no manual setup.

dd if=/dev/hda | ssh user@g5security /bin/dd
of=/Volumes/Images/SSHTest.img

Analysis:

As the md5sum results show, all three methods of data transfer produce a valid image. Also, the netcat connection and the ssh connection were both fairly quick – about 1 hour each. The windows share connection was significantly longer. The 7+ hours required to do a windows share image are inappropriate for use in a production environment. However, the requirement of the netcat connection to manually setup a netcat listener is less than desirable also. The simplest and quickest way to create a remote disk image is through the use of ssh as the transport protocol.

Presentation:

The best way to present the data from this analysis would be to first display a screen shot showing that the images created by all three of the methods produce identical disk images. Then use of a chart showing the transfer times for each method should be developed. Lastly, a listing of the

steps required to perform the remote disk imaging should be developed. This will show that while Netcat and ssh are similar in speed, SSH can be used against any ssh server, similar to the ease of use of windows share.

Because this process relies on the dd command, and is simply a transport mechanism, the data will be preserved in a common format suitable for use in any forensics investigation. This is simply a better way to transfer the data of a dd command from the target system to the destination system.

The use of md5sums will be critical to prove that the image wasn't changed during transport. This is true no matter what mechanism is used.

Conclusion:

SSH is a superior tool for transport of dd output than either netcat or windows drive mappings. Because SSH is a common protocol, it is simple to setup an analysis workstation with an SSH Server running. This makes the connection as simple as a windows share. SSH also provides the speed of a netcat connection. This allows faster imaging of the target disk so that the suspect system can be imaged quickly and returned to the appropriate group for rebuilding, thus helping get the system back on line faster.

Because SSH is a secure protocol, it lends itself well to use in forensics. This tool will give Forensic Analysts the ability to use personnel at the site of the incident to create an image. It also allows the analyst to hand off manual repetitive parts of the disk imaging process to less costly resources. A Helix CD can be distributed to local administrators with instructions on how to initiate a disk image through the use of SSH, and no setup on the remote destination system is required. Personally, this means that at 2am, when an incident occurs in Hong Kong, the local System Administrator can begin a disk image without needing assistance from anyone in the US. Once the image is complete, the forensic analyst can be contacted to begin the process, instead of having to setup the communications (ie. Netcat) and then waiting around for the image to be completed (potentially several hours).

There are no improvements or changes that I would recommend currently to the SSH tool. Because it is a secure tool, it is easily used to improve forensic processes. I think this is a case where the forensics community benefits from an existing tool in another related discipline.