Baselines and Incident Handling

Chris Christianson

BASELINES AND INCIDENT HANDLING

GCIH Gold Certification

Chris Christianson, infosecsurvivor@gmail.com

Advisor: Pedro Bueno

Accepted: February 23rd 2007

Table of Contents

1. Introduction

Preparation is an important part of the incident handling process (SANS, n.d.); it is the only part of the process that an incident handler can do ahead of time. Preparing well can help an incident handler to identify and respond to an incident more quickly. It can also help him or her to be more efficient throughout the entire incident handling process.

While there are many things that an incident handler can do to prepare, one of the most commonly overlooked and underestimated things is the establishment of baselines. Why is the establishment of baselines so important in incident handling? What baselines should be established? How are they established? How do they aid an incident handler in the detection and monitoring phases of incident handling?

The purpose of this paper is to explain why the establishment of baselines is an important part of incident handling and how doing so can be a useful tool for an incident handler during incident handling process. Examples will be given regarding how to establish baselines on various types of systems and network equipment. Also, demonstrations will be given showing what to look for when attempting to identify, using

baselines, whether an incident has occurred.

2. Importance

What is a baseline? Merriam-Webster defines a baseline "as a line serving as a basis; *especially*: one of known measure or position used (as in surveying or navigation) to calculate or locate something. A usually initial set of critical observations or data used for comparison or a control" (Merriam-Webster, n.d.). As it applies to incident handling, a baseline is a snapshot of a system or the network—the way it normally works.

Why is the establishment of baselines so important in incident handling? Simply put, baselines can help an incident handler to determine whether or not an incident has occurred. A good incident handler needs to be able to detect incidents as quickly and efficiently as possible—something that is not always an easy task.

Often what happens in incident handling is that something is brought to a handler's attention that causes him or her to suspect that an incident has occurred. The incident handler then begins to investigate, and tries to determine if what has occurred is actually an incident. This can be a rather slow and time-consuming process. It may involve examining many kinds of different logs files, running processes, and the analysis of

network traffic. Often the handler may not even know exactly what it is for which he or she is looking. This is where baselines can help.

Most of the time the first question that must be asked concerns what happened or changed. Ascertaining whether something has changed is key for an incident handler in determining where to begin looking for possible problems. Once the change(s)made have been found, the incident handler can look into who or what made the change and why. Did a user or a system administrator make the change? Was it authorized? Or was the change made by something else—perhaps by some malicious code? Baselines can show what has changed and help in answering these important questions.

For example, a user reports that something is wrong with his or her system; specifically, it is running slower than normal. How does an incident handler know whether or not that system really is slower? What if the system is running slowly? What if it is running slowly because of some malicious code? This is when having a baseline helps. If the incident handler knows what the processor, hard drive, and network utilization is normally, he or she will be better able to determine whether or

not something is actually wrong. If he or she has a baseline, it will be possible to make this determination more quickly and easily.

Sometimes an incident may occur on a system and it may not even be noticeable by a user. Perhaps a tiny backdoor program is installed onto the system. The program is so small and takes so little of the system's resources that the user will probably never notice it. Yet, if a baseline has been established, an incident handler would know which programs, processes, and services are normally running on this system. That would enable the handler to detect the backdoor program.

Being a good incident handler requires the kind of attention shown by someone who is obsessed with his or her car. Such a person knows everything about the vehicle. He or she takes good care of it, knows where everything is on the vehicle, and understands how everything works. Such a person knows how the car is supposed to run. If something goes wrong with the car—if the car makes an odd sound, for instance—such a person would be aware. This kind of familiarity and attention would allow the person to detect the problem immediately. Baselines help an incident handler in the same way.

Baselines help an incident handler to know where everything is at on his or her network and system. It helps the handler to understand how it all works, and how it is supposed to run. If something is not as it should be, just like that person who is obsessed with the car, the incident handler knows about it. Is that system supposed to be running an FTP server? For what is this particular program used? Why is this system sending all this traffic to that system? An incident handler who has established baselines will be able to answer these questions quickly, determine whether or not an incident has occurred, and move on with the incident handling process.

The establishment of baselines is indeed an important part of the preparation stage of the incident handling process. The preparation stage involves obvious things such as creating a plan, organizing a team, and documenting. As noted above, preparation is the only part of the entire incident handling process that can be done ahead of time. It is, in fact, what an incident handler spends most of his or her time doing. Establishing baselines can be considered part of the task of documentation.

Often documentation, especially things such as the

establishing of baselines for system and network performance, is overlooked by system administrators. These matters are considered time-consuming and tedious tasks that serve little purpose. As will be demonstrated, however, nothing could be further from the truth. With a little ingenuity, a few tricks, and a bit of effort, establishing baselines is not a difficult task.

3. Baselines to Establish

In a perfect world, a system administrator would build a system from scratch using media of established high quality. Before the system is ever put into production, he or she would take a snapshot of the system in that perfect state. There would be a list of installed programs and running services. A list of installed hardware might also be useful. Anytime anything on the system is changed, a new snapshot would be taken.

The reality, however, is that this is not a perfect world. Rarely, if ever, does a system administrator have a perfect system with which to work. What, then, is an incident handler to do in such a situation? He or she uses the “Last Known Good” configuration so to speak.

Sometimes an incident handler is going to have to assume that, at this particular point in time, the system or network is good; in other words, it is running properly. At this point, a baseline can be established.

This is not perfect, to be sure, but a system administrator has to begin somewhere. As will be discussed later, once a system administrator discovers what is running on his or her systems and networks, he or she can work backwards. Gaining

insight into the network, the system administrator can then uninstall any unnecessary programs or services that aren't needed. He or she can clean up the systems and then establish newer, better baselines. From this point going forward, of course, he or she will at least know if something has changed.

With the understanding that a baseline can be established from the "Last Known Good" configuration, what kind of baselines should we establish? The following is a list of some useful baselines:

- List of installed programs and running services;
- Processor utilization, memory utilization, hard drive utilization, and network utilization; and
- Logs.

This list is by no means exhaustive; there are many other possible baselines that could be established. Which baselines a system administrator chooses to establish depends largely upon the administrator and his or her environment. The thing to be borne in mind is that it is important to establish what is running on the system and how.

The following sections discuss each of the above-named baselines in more detail.

List of Installed Programs and Running Services

It is helpful to have a list of the installed programs and services running on each system. This list should include the version numbers of programs, as well any hot fixes, that are installed. Such information is beneficial for a number of reasons. First, knowing which programs are installed and what services are running on a system will help an incident handler to understand what a system should be doing. With this information, an incident handler will also know what a system should not be doing.

Having a list of installed programs and running services will also help an incident handler to know if something has changed on the system; this can help an incident handler to determine weather or not a new, possibly unauthorized, program has been installed on the system. Should the system have program X installed on it? Was program X always previously installed? Is the system being used for program X's purpose? Is this a critical function of the system? Having a list of installed programs and running services will help an incident handler to answer these questions.

When one is looking at a long list of services and programs that are running, it may be difficult to determine the exact purpose of each application and process and whether or not it has always been there. When this information is made available ahead of time, assessing the situation is much easier. Baselines ease this task, allowing the incident handler to compare what was running previously vis-à-vis what is running now.

Information about different programs and services can be found at a number of different Internet sites; simply going to www.google.com and searching for the program or service in question will usually allow an incident handler to identify quickly what it is. The following is a list of some other Web sites where incident handlers can look up information about various programs and services:

- www.tasklist.org—A site that contains a list of processes;
- www.processid.com—Another site that contains a list of processes;
- www.answersthatwork.com/Tasklist_pages/tasklist.htm—Another site that contains a list of processes;
- http://www.liutilities.com/products/wintaskspro/proces

slibrary/—Another site that contains a list of process; this site can also scan for processes.

System Utilization

It is also important to obtain baselines for system utilization. Baselines for processor, memory, and hard drive utilization can be very useful to an incident handler in detecting weather or not something has occurred on a system.

For example, if a system's processor utilization suddenly jumps to 100% and nothing has changed on the system, this could be a sign of a virus or worm. In fact, this exact symptom would be something that might be observed on a system that had been infected by the Code Red worm. One of the telltale signs of this particular worm is high processor utilization (Cisco, 2001).

The same may be said for other system utilization baselines. One of the most valuable baselines that can be established is the baseline for network utilization. Seeing a large amount of traffic going between one system and another could be an indication that the system has been infected by some malware. That was the case with the SQL Slammer worm (Gerhards, 2003).

Logs

Logs are a valuable source of information to system administrators and incident handlers alike. While not commonly thought of as being useful for baselines, indeed they are. They contain all sorts of information that can help an incident handler determine what is normal and whether anything has changed on a system. For example, perhaps an incident handler is examining system logs and notices some invalid login attempts. Are these invalid login attempts normal? Was the invalid login attempt just a case of a user mistyping his or her password? Or was there a large number of login attempts? Were there invalid login attempts to multiple accounts? Was the login attempt in the middle of the night, when the user doesn't normally work? Establishing baselines—knowing what is normal—can help an incident handler to determine weather or not an incident has occurred.

Certain events may be unusual in themselves. The following Web sites contain information about various events that may appear in logs:

- www.eventid.net—Subscription-based service that

contains lists of different events; and

- www.microsoft.com/technet/support/ee/ee_advanced.aspx—Microsoft TechNet Events and Error Message Center.

4. How to Establish Baselines

(Windows, *nix, Cisco)

How does a system administrator or an incident handler go about establishing these various types of baselines on different systems? This section discusses some of the most common baselines that can be established on some popular network operating systems and devices.

Windows Systems

The following are some popular commands and utilities that can be used to get baselines of currently supported Windows Systems:

Performance Logs and Alerts

Performance Logs and Alerts can be used to monitor Windows Systems performance statistics. This program can be accessed by either of the following:

Start > Run > type ‘perfmon’

or

Start > Control Panel > Administrative Tools >

Performance

Performance Logs and Alerts open. By default there are three counters running: one for disk, one for memory, and one for processor utilization. However, these are not being logged to a file. In order to log these (or any other) settings to a file, these steps should be followed:

1. Right-click on “Counter Logs”;
2. Select “New Log Settings”;
3. Type a name for the “New Log Settings”;
4. Click “OK”;

The “Properties” dialog box opens.

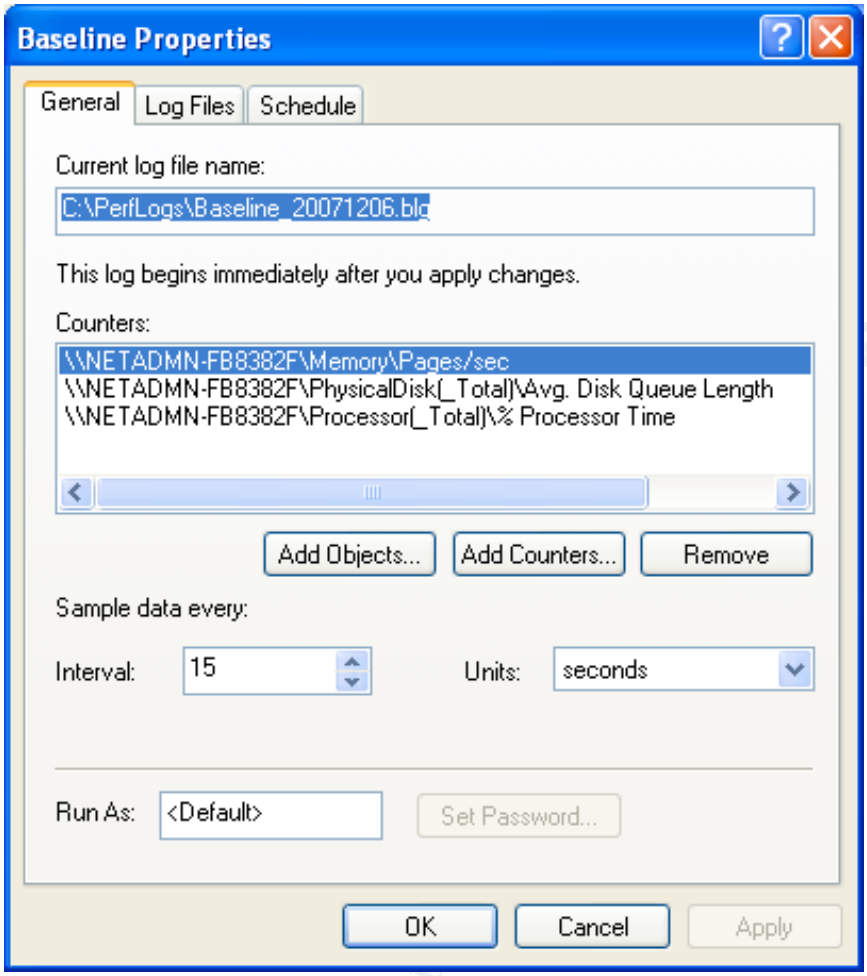

Figure 1. Properties Dialog Box

The Properties Dialog box has three tabs: “General,” “Log Files,” and “Schedule.” To add counters, click the “Add Counters” button. The following is a list of some of the most

useful counters for incident handling:

- Memory\Pages/sec;
- PhysicalDisk[_Total]\Avg. Disk Queue Length;
- Processor[_Total]\%Processor Time;
- Network Interface\Bytes Total/sec

After adding the desired counters, the log file needs to be configured. To do this, click on the “Log Files” tab. The “Log Files” settings are shown below:

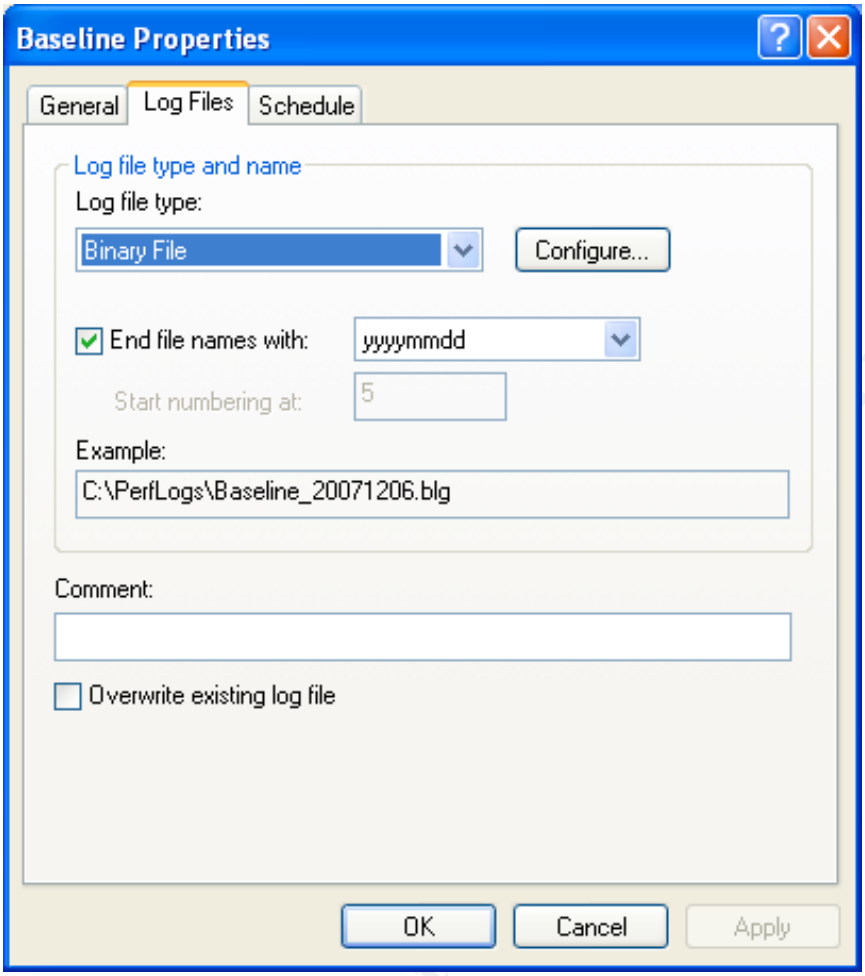

Figure 2. Log Files Tab

While there are a number of different possible settings, probably the easiest method for naming files is to end file names with “yyyymmdd” or any of the options that allow for the

use of the date. This will make finding log files later easier.

The “Schedule” tab allows the current log settings to be started and stopped, and a new log file to be created, all at the specified time. The settings shown below start a new log file everyday at 12:00 a.m.:

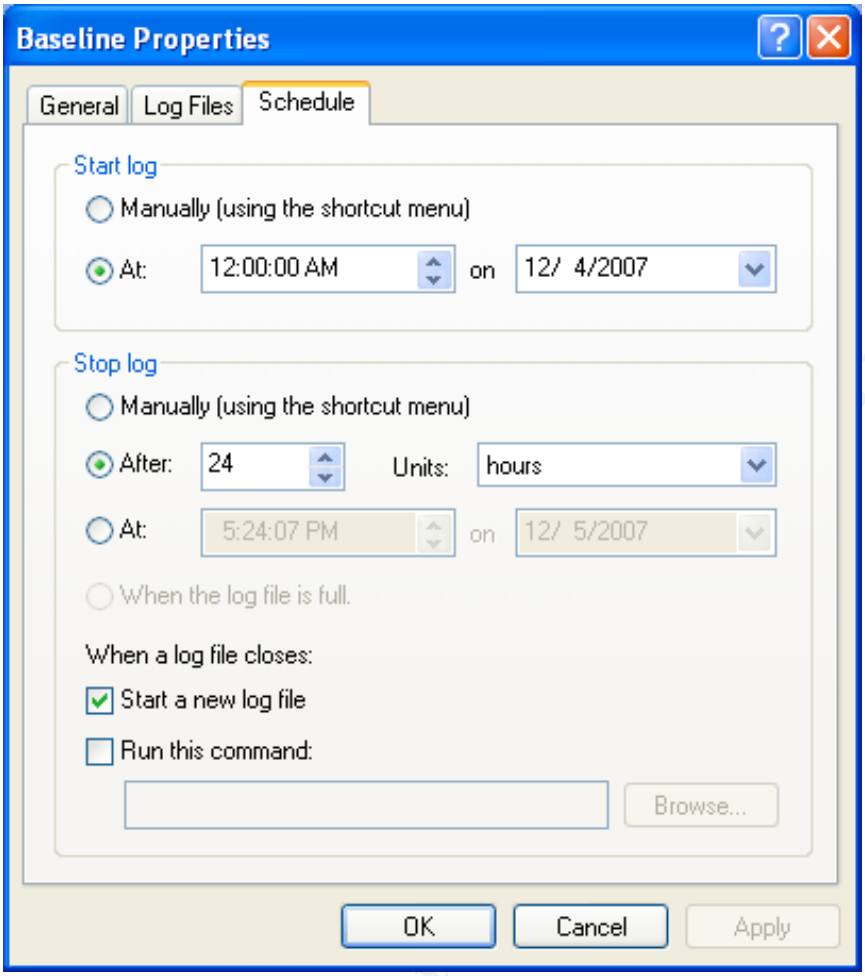

Figure 3. Schedule Tab

Another option for scheduling performance logs and alerts is to use Task Scheduler and the Logman utility. The following is the syntax for Logman:

```
logman.exe start <collection_name>
```

Where the “collection_name” is the name of the log settings.

These settings can also be copied from one system to another simply by saving them. They are saved as .htm/,html files. These files can be placed on Web server where they can then be accessed by other machines using the “New Log Setting From …” option.

Simple Network Management Protocol (SNMP)

Another method for gathering processor, memory, and network utilization statistics from Windows systems (as well as *nix systems and Cisco devices) is by using the Simple Network Management Protocol (SNMP).) The SNMP allows for the exchange of information between network devices and makes it possible for system administrators to manage and monitor network performance (Cisco, 2006). The protocol is supported by a number of different network management/monitoring programs. The following Web site contains more information about the SNMP:

http://www.cisco.com/univercd/cc/td/doc/cisintwk/ito_doc/snm

p.htm

From the managed system perspective, all that needs to be done to allow system utilization statistics to be collected is to enable the SNMP and then set the SNMP community name. To configure SNMP on a Windows systems, follow the following steps (Microsoft, 2006):

1. Start > Control Panel > Administrative Tools > Services;
2. In the right pane, double-click “SNMP Service”;
3. Click the “Traps” tab;
4. In the “Community Name” box, type the community name to which the system will send trap messages;
5. Click the “Add to List” button;
6. Under the ”Trap Destinations” box, click the “Add” button;
7. An SNMP Service Configuration Dialog appears; enter the system to which the system will send trap messages and click “Add”; and
8. Click the “OK” button.

While SNMP is a very useful for network management, it can also be a security risk if it is not properly configured. It is

very important that, when configuring the SNMP service, security settings are also set. These settings are located on the "Security" tab of the SNMP service. For more information on this, see the following link:

http://support.microsoft.com/kb/324263/

Windows Management Instrumentation Command-Line (WMIC) Tool

The Windows Management Instrumentation Command-Line (WMIC) is a command-line and scripting interface that simplifies the use of Windows Management Instrumentation (WMI) and systems managed through WMI (Microsoft Tech Net, 2008). WMI allows for the effective management of PC and server systems in an enterprise network (Microsoft, 2008). A great deal can be done with this tool, including the gathering of information for the establishment of baselines. Table 1 contains a list of some useful WMIC commands:

WMIC COMMANDS AND DESCRIPTIONS	
wmic.exe OS list full	Lists information about the operating system
wmic.exe USERACCOUNT list full	Lists information about user accounts
wmic.exe GROUP list	Lists information about groups

full	
wmic.exe SHARE list full	Lists information about shares
wmic.exe PROCESS list full	Lists information about processes
wmic.exe SYSDRIVERS list full	Lists information about drivers
wmic.exe SERVICE list full	Lists information about devices
wmic.exe JOB list full	Lists information about scheduled jobs
wmic.exe STARTUP list full	Lists information about startup programs
wmic.exe NICCONFIG list full	Lists information about network configuration

Table 1. WMIC Commands

This list of commands is by no means a complete list of all the information available using WMIC, but it is a good starting place for system administrators and incident handlers to get a picture of what is running on their systems. For more information on the WMIC, see the following link:

http://www.microsoft.com.nsatc.net/resources/documentation/windows/xp/all/proddocs/en-us/wmic.mspx?mfr=true

Other Built-in Windows Commands

There are also a number of other built-in Windows commands that are very useful for establishing baselines. Table 2 lists some of these commands.

Built-in Windows Commands	
ver	Displays the Windows version
ipconfig /all	Displays network configuration
netstat –ano	Displays open ports
route print	Displays routing table
dir %SYSTEMDRIVE% /A:H /S /ON /T:W /N	Displays hidden files with last-modified date
dir %SYSTEMDRIVE% /A:-D /S ON /T:W /N	Displays non-hidden files with last-modified date
at	Displays a list of scheduled jobs
tasklist	Displays a list of running applications (XP, 2003)
systeminfo	Displays basic system information (XP, 2003)

Table 2. Built-in Windows Commands

Everything that has been discussed thus far is built into Windows and installed by default when Windows is installed. The next commands are part of Windows, but they are not installed by default and are considered optional installations.

Support Tools

The following utilities are part of "Windows Support Tools." Usually these utilities are located on the Windows system installation CD in the "Support" directory. They can also be downloaded from Microsoft's Web site. Table 3 lists some of these Support Tools.

	Support Tools
netdiag /v	Displays networking configuration
diruse "C:\"	Lists byte and file count for C:\
diruse /S "%SYSTEMROOT%"	Lists byte and file count for Windows directory
diruse /S "%PROGRAMFILES%"	Lists byte and file count for Program Files directory
filever %SYSTEMDRIVE% /s /b /v	Lists all files, their versions, and attributes

Table 3. Support Tools

In order to install all of these tools, one must install Windows Support Tools with the "Complete" option. This option also installs another very useful utility, called WinDiff, which can be used for comparing different files and will be discussed in the next section.

Port Scanner or Vulnerability Scanner (Nmap)

The next baseline to be established is that concerning the ports that are open on a system. This can be accomplished using any port scanner or vulnerability scanner, such as Nmap. This is a fast free post scanner available at insecure.org. Scheduling Nmap, or any scanner for that matter, to run once a night or week, and then sending that output to a file, is all that is needed to establish this baseline.

The following command can be used to scan Windows, *nix systems and Cisco devices, and to send the output to a file:

```
Nmap <IP Address> > baseline.txt
```

It is important to keep in mind that Nmap is a very powerful tool and it should be installed on a system that is safe and secure. Care should be taken to make sure that output from the Nmap command is limited to authorized individuals. The output generated from this command, and the other commands as well, is valuable reconnaissance information for malicious attackers.

File Integrity Checker (MD5SUM/SHA1SUM)

The last baseline that needs to be established is that of a file integrity check. This can be done using any number of

commercial programs, such as Tripwire for Windows or, for free, using MD5SUM of SHA1SUM. The following examples create checksum for most of the critical files in the Windows directory, the System32 directory, the Windows Drivers directory, and the Etc directory.

```
sha1sum C:\Windows\*.dll
sha1sum C:\Windows\*.exe
sha1sum C\Windows\System32\*.com
sha1sum C:\Windows\System32\*.dll
sha1sum C:\Windows\System32\*.exe
sha1sum C:\Windows\System32\Drivers\*.sys
sha1sum C:\Windows\System32\Drivers\etc\*.*
```

Automation and the Results

All of the aforementioned commands can be placed into a batch file and scheduled to run daily, weekly, monthly, or whenever necessary. The best thing about all of this is that, after they are run the first time, there will at least be some sort of a baseline with which to work. From this time onward, if there is a change made to the system, one may simply run the batch file again to establish a new baseline.

The output of the all these commands or the batch file can be piped to file. For example:

baseline.bat > baseline_12-8-07.txt

These files can be copied to a network share or to an FTP server. Doing so will make the files easily available for analysis later, and will also prevent them from being deleted or otherwise tampered with. It is important to remember that, once a system has been compromised, one of the malicious attacker's goals will be to hide his or her tracks. It is, therefore, vitally important that the output generated from these commands be moved to another safe and highly secure system.

**nix Systems*

The same baselines that were established on Windows systems can also be established on *nix systems. The following section briefly covers some of the useful commands.

***nix Commands**	
uname –a	Displays *nix kernel version
cat /etc/issue	Displays linux distribution and version (Linux only)
less /etc/passwd	Displays users
grep :0: /etc/passwd	Displays user accounts with a UID = 0

ps –aux	Displays processes
find / -uid 0 –perm -4000 –print	Displays files owned by root (Linux only)
find / -name "…" –print	Displays files with dots and spaces
find / name ".. " –print	Displays files with dots and spaces
find / -name ". " -print	Displays files with dots and spaces
find /-name " " -print	Displays files with dots and spaces
ifconfig	Displays network configuration information
ip link | grep PROMISC	Displays network running in promiscuous mode (Linux only)
netstat –na	Displays open ports
arp –a	Displays ARP entries
crontab –u root -l	Displays cron jobs scheduled by root
cat /etc/crontab	Displays cron jobs (Linux only)
ls /etc/cron.*	Displays cron jobs (Linux only)
rpm –qa | less	Displays list of installed packages (Redhat, Mandrake, etc.)
dpkg –l	Displays list of installed packages (Debian, Ubuntu, etc.)
lsof	Displays open files

*Table 4. *nix Commands*

File Integrity Checker (Tripwire)

The last baseline that needs to be established is a file integrity check. This can be done using Tripwire, which is

available for free for *nix systems. After installing Tripwire and creating the Tripwire database, the following command can be used to compare files (McIntyre, 2001a; 2001b):

```
tripwire –m c
```

The following are links to several articles from *TechRepublic* explaining how to install and use Tripwire:

- http://articles.techrepublic.com.com/5100-6347-1053490.html; and
- http://articles.techrepublic.com.com/5100-6345-1053398.html.

Automation and the Results

Just as with Windows system, everything on *nix systems can be automated. This can be done by creating scripts (there are a number of different types of scripting technologies that can be used) and then scheduling the script to run using Cron.

Cisco Routers and Switches

Since Cisco routers and Switches are so prevalent, it is important that the method of establishing baselines for these devices also be covered here. Cisco's IOS is a bit different

than the operation of a workstation or server; the method of establishing baselines for it is, therefore, also a bit different. The following table lists some commands that can be useful for establishing baselines:

Cisco IOS Commands	
show clock detail	Displays detailed information about the clock
show version	Displays system hardware and software status
show running-config	Displays current configuration
show startup-config	Displays startup configuration
show ip route	Displays IP routing table
show ip arp	Displays IP ARP table
show users	Displays users connected via terminal sessions
show logging	Displays log
show ip interface	Displays IP information for interfaces
show interfaces	Displays interface information
show tcp brief all	Displays TCP connection information
show ip sockets	Displays IP connection information
show ip nat translations verbose	Displays IP NAT translation table
show ip cache flow	Displays IP route cache flows

show ip cef	Displays Cisco Express Forwarding table

Table 5. Cisco IOS Commands

Since such commands are usually run during a session, capturing the output of these commands simply requires turning on the logging for that session.

System Utilization

System utilization for processor, memory, and network is best collected from Cisco devices using SNMP. Again, SNMP can be easily set up on a device (Cisco, 2007). Doing so requires setting the SNMP community name; instructions for doing so on a Cisco device may be found on the following Web site:

http://www.cisco.com/univercd/cc/td/doc/product/software/ios122/122cgcr/ffun_c/fcfprt3/fcf014.htm

SNMP Management Tools

There are many commercial and open source SNMP management tools available to help System administrators and incident handlers in collecting SNMP data. Once configured, these tools can automatically gather, store, and analyze data from devices

all over the network. The following are free open-source SNMP management tools:

- Nagios; and
- OpenNMS.

5. Examples

As has been discussed previously, establishing baselines can help an incident handler to determine weather an incident has occurred or not. Once established, determining whether something has changed on a system—such as a new port being opened, a new program being installed, or high network utilization—is quicker and easier. This section gives some examples of baselines in action.

Example #1

In the first example, a baseline has been established for a Windows System. Note that, in order to simplify this example, only the output from the following commands were included in baseline.bat (SANS, 2006): 'wmic.exe PROCESS list brief' and 'netstat –ano' (See Appendix A - baseline.bat/)

```
*************************************************************
*********
****************************** BASELINE
*********************************
*************************************************************
*********

******************** wmic.exe PROCESS list brief
*************************

HandleCount Name                Priority  ProcessId  ThreadCount
```

```
WorkingSetSize
0             System Idle Process  0         0          1
28672
255           System               8         4          56
..245760
21            smss.exe             11        616        3
409600
431           csrss.exe            13        664        11
1912832
517           winlogon.exe         13        688        23
4739072
286           services.exe          9         732        18
4345856
355           lsass.exe             9         744        23
6217728
24            vmacthlp.exe          8         900        1
2220032
201           svchost.exe           8         916        19
4866048
269           svchost.exe           8         992        10
4169728
1419          svchost.exe           8         1084       74
28340224
77            svchost.exe           8         1152       6
3313664
197           svchost.exe           8         1264       14
4435968
111           spoolsv.exe           8         1408       14
4612096
142           pNSClient.exe         8         1604       5
5328896
278           explorer.exe          8         1796       11
16027648
158           snmp.exe              8         1956       5
3756032
115           svchost.exe           8         2012       6
4145152
53            VMwareService.exe    13         196        3
2973696
25            VMwareTray.exe        8         456        1
2928640
```

```
128           VMwareUser.exe     8          1188       2
6033408
149           msmsgs.exe         8          1204       4
884736
100           alg.exe            8          1304       6
3579904
28            wscntfy.exe        8          1596       1
2023424
136           wmiprvse.exe       8          1732       6
5050368
116           imapi.exe          8          2008       8
4034560
34            cmd.exe            8          648        1
2953216
237           wuauclt.exe        8          1176       8
17375232
39            ntvdm.exe          8          2428       3
2416640
153           wmiprvse.exe       8          2512       7
6402048
161           wuauclt.exe        8          2576       5
4079616
117           wmic.exe           8          2652       3
5672960

****************************** netstat -ano
******************************

Active Connections

  Proto  Local Address          Foreign Address        State
PID
  TCP    0.0.0.0:135            0.0.0.0:0              LISTENING
992
  TCP    0.0.0.0:445            0.0.0.0:0              LISTENING
4
  TCP    0.0.0.0:1248           0.0.0.0:0              LISTENING
1604
  TCP    127.0.0.1:1029         0.0.0.0:0              LISTENING
1304
  TCP    172.16.5.132:139       0.0.0.0:0              LISTENING
```

```
4
  UDP    0.0.0.0:161            *:*
1956
  UDP    0.0.0.0:445            *:*
4
  UDP    0.0.0.0:500            *:*
744
  UDP    0.0.0.0:1035           *:*
1152
  UDP    0.0.0.0:4500           *:*
744
  UDP    127.0.0.1:123          *:*
1084
  UDP    127.0.0.1:1900         *:*
1264
  UDP    172.16.5.132:123       *:*
1084
  UDP    172.16.5.132:137       *:*
4
  UDP    172.16.5.132:138       *:*
4
  UDP    172.16.5.132:1900      *:*
1264
```

This is the output from the Nmap scan.

```
Starting Nmap 4.20 ( http://insecure.org ) at 2008-01-10 12:04
PST
Interesting ports on 172.16.5.132:
Not shown: 9996 closed ports
PORT     STATE SERVICE
135/tcp  open  msrpc
139/tcp  open  netbios-ssn
445/tcp  open  microsoft-ds
1248/tcp open  hermes

Nmap finished: 1 IP address (1 host up) scanned in 3.949 seconds
```

This is the output from the same baseline.bat run again later.

```
************************************************************
*********
```

```
****************************** BASELINE
********************************
**********************************************************************
*********

****************** wmic.exe PROCESS list brief
************************

HandleCount  Name                 Priority  ProcessId
ThreadCount  WorkingSetSize
0            System Idle Process  0         0          1
28672
282          System               8         4          57
245760
21           smss.exe             11        616        3
409600
433          csrss.exe            13        664        11
1986560
513          winlogon.exe         13        688        22
4734976
288          services.exe         9         732        19
4354048
347          lsass.exe            9         744        22
1601536
24           vmacthlp.exe         8         900        1
2220032
198          svchost.exe          8         916        18
4857856
287          svchost.exe          8         992        11
4239360
1382         svchost.exe          8         1084       71
28053504
77           svchost.exe          8         1152       5
3301376
195          svchost.exe          8         1264       13
4431872
111          spoolsv.exe          8         1408       13
4608000
153          pNSClient.exe        8         1604       5
5353472
285          explorer.exe         8         1796       11
```

```
16031744
158          snmp.exe             8         1956       5
3756032
119          svchost.exe          8         2012       6
4153344
53           VMwareService.exe    13        196        3
2973696
25           VMwareTray.exe       8         456        1
2928640
128          VMwareUser.exe       8         1188       2
6033408
155          msmsgs.exe           8         1204       4
1007616
104          alg.exe              8         1304       6
3588096
28           wscntfy.exe          8         1596       1
2023424
138          wmiprvse.exe         8         1732       5
4898816
116          imapi.exe            8         2008       5
4001792
34           cmd.exe              8         648        1
2977792
237          wuauclt.exe          8         1176       8
17379328
39           ntvdm.exe            8         2428       3
2416640
158          wuauclt.exe          8         2576       4
4071424
44           tini.exe             8         3216       3
1888256
34           cmd.exe              8         3232       1
2482176
117          wmic.exe             8         3256       3
5668864
138          wmiprvse.exe         8         3304       6
5713920

****************************** netstat -ano
******************************
```

```
Active Connections

  Proto  Local Address          Foreign Address        State
PID
  TCP    0.0.0.0:135            0.0.0.0:0              LISTENING
992
  TCP    0.0.0.0:445            0.0.0.0:0              LISTENING
4
  TCP    0.0.0.0:1248           0.0.0.0:0              LISTENING
1604
  TCP    0.0.0.0:7777           0.0.0.0:0              LISTENING
3216
  TCP    127.0.0.1:1029         0.0.0.0:0              LISTENING
1304
  TCP    172.16.5.132:139       0.0.0.0:0              LISTENING
4
  UDP    0.0.0.0:161            *:*
1956
  UDP    0.0.0.0:445            *:*
4
  UDP    0.0.0.0:500            *:*
744
  UDP    0.0.0.0:1035           *:*
1152
  UDP    0.0.0.0:4500           *:*
744
  UDP    127.0.0.1:123          *:*
1084
  UDP    127.0.0.1:1900         *:*
1264
  UDP    172.16.5.132:123       *:*
1084
  UDP    172.16.5.132:137       *:*
4
  UDP    172.16.5.132:138       *:*
4
  UDP    172.16.5.132:1900      *:*
1264
```

This is the output from another Nmap scan.

```
Starting Nmap 4.20 ( http://insecure.org ) at 2008-01-10 12:05
PST
```

```
Interesting ports on 172.16.5.132:
Not shown: 9995 closed ports
PORT     STATE SERVICE
135/tcp  open  msrpc
139/tcp  open  netbios-ssn
445/tcp  open  microsoft-ds
1248/tcp open  hermes
7777/tcp open  unknown

Nmap finished: 1 IP address (1 host up) scanned in 3.997 seconds
```

It is worth nothing that the “wmic PROCESS list brief” command has a new entry, tini.exe. Looking at the output of the “netstat –ano” command reveals that there is a new port opened that wasn’t opened before—port 7777. That port is associated with Process ID 3216. Referring back to the output of the “wmic PROCESS list brief” command shows that Process ID 3216 belongs to tini.exe. Looking at the output of the “nmap” command also verifies that port 7777 was open.

Remember that the output from this baseline has been simplified. If a list of files and directories had been included in the output, “tini.exe” would have included on that list. Analyzing the firewall logs from the system might reveal connections to port 7777. If an antivirus program was installed, the system it would probably detect “tini.exe” as malicious software and quarantine the program; this would also appear in logs.

"Tini.exe" is a small backdoor program. The following Web site contains more information about tini.exe:

http://www.ntsecurity.nu/toolbox/tini/

WinDiff

WinDiff, a utility mentioned earlier that is available in the Windows "Support Tools," is useful for comparing files and directories. The following is a screenshot of WinDiff, comparing the output from the files created in this example. Notice how WinDiff highlights the lines that are different between the two files that are being compared. This makes it much easier to identify what has changed.

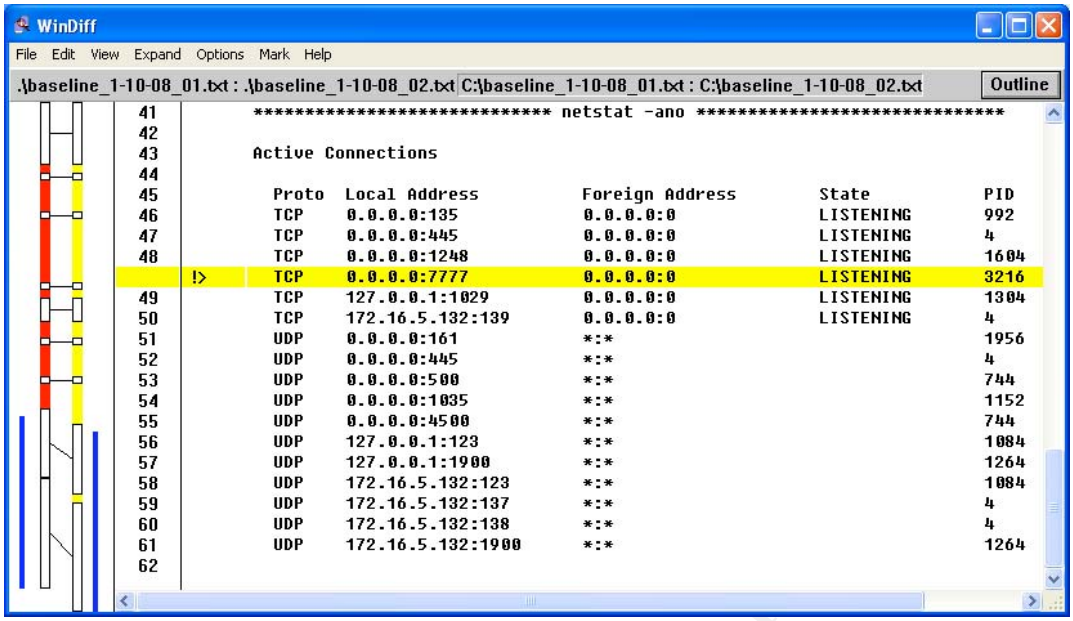

Figure 4. WinDiff

Ndiff

Another option for comparing the output of Nmap scans is a program called Ndiff. According to its authors, Ndiif can be used to, “compare putatively similar files, ignoring small numeric differences” (Beebe, 2004). The following example demonstrates how to use Ndiff to compare scan output:

First, create a baseline using the following command (PenguinSoft, 2000):

nmap –m baseline.nm <IP Address>

Next, scan the system later to see if anything has changed:

nmap –m 12-24-07.nm <IP Address>

Finally, compare the output of the scans using Ndiff:

ndiff baseline.nm 12-24-07.nm

The following site contains more information about using Ndiff:

http://www.penguin-soft.com/penguin/man/3/NDiff_Quickstart.html

Example #2

In the next example, a baseline was established of another Windows system. The following is a snapshot of the processor, memory, and network utilization of the system for a 1-hour period. It is typical of the system utilization during most times.

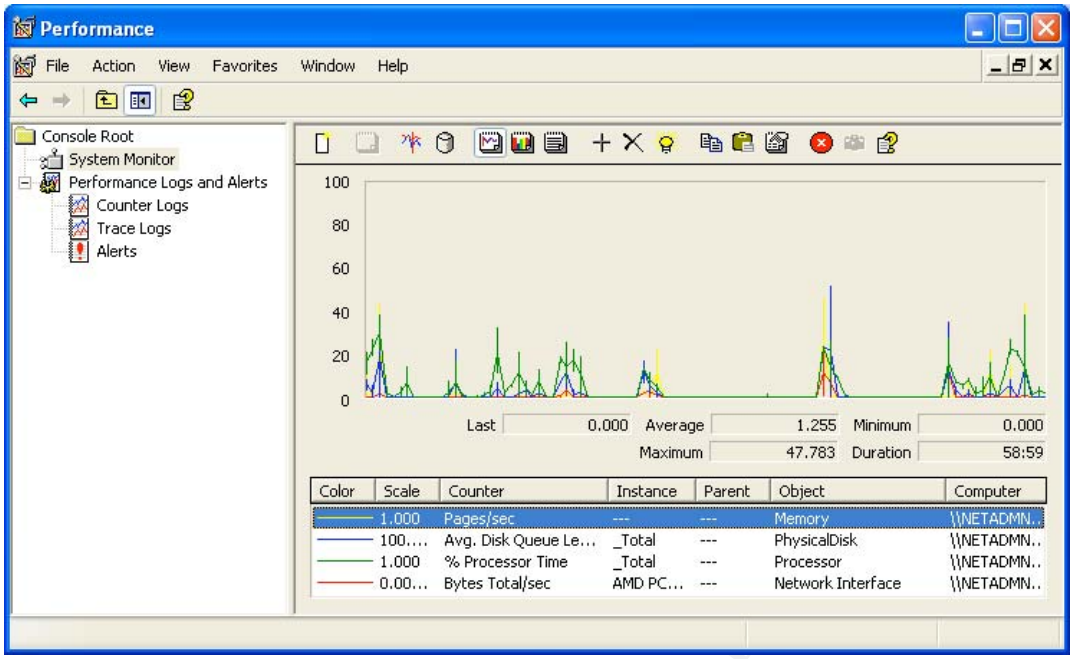

Figure 5. System Monitor Snapshot #1

This is another snapshot of the same system, later.

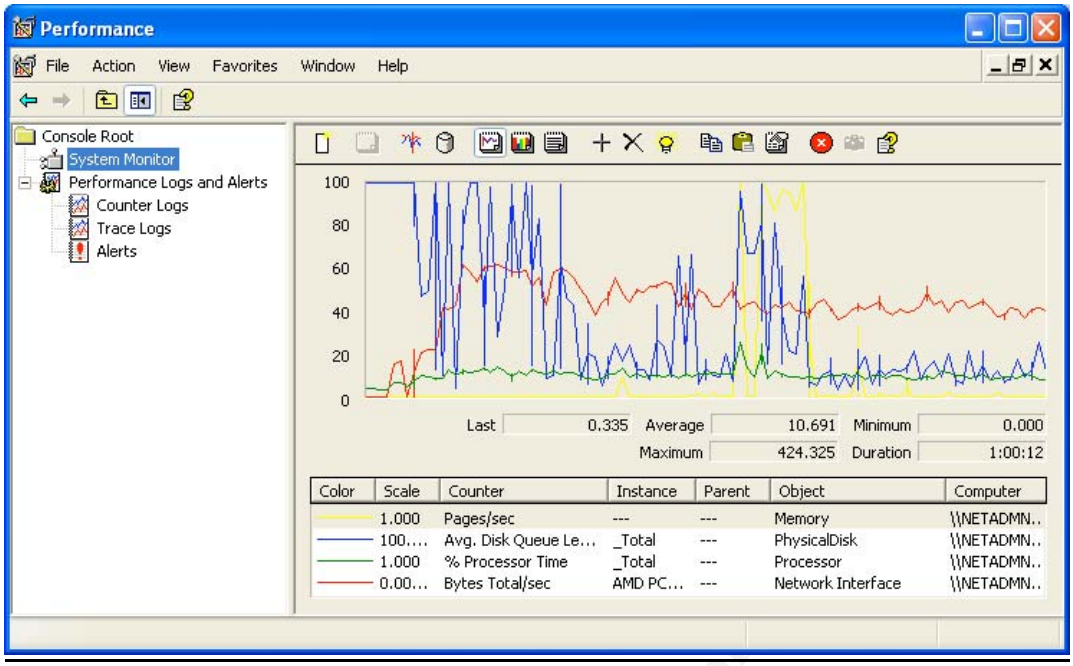

Figure 6. System Monitor Snapshot #2

Looking at Figure 6, several different things stand out. The "Average Disk Queue Length" is spasmodic, indicting a great deal of reading and writing to hard disk. The "Bytes Total/sec" is higher than normal, and it is higher than normal for a long period of time. It is important to remember that this is over the period of 1 hour. That is a lot of network traffic, compared to what would be normal for that period of time.

In this particular instance, the reason for the large

increase in traffic was that the user had installed a bittorrent client on their workstation and was downloading a large file. If other baselines had been established, they would have shown other indications of this activity, such as the bittorrent client being an installed program and a running process. New port numbers would have also appeared as open.

The point of this particular example is to show the benefits of establishing baselines for system utilization—in this case, network utilization. Doing so can help a system administrator or an incident handler to deal with a possible incident. Network utilization could also be monitored from routers and switches with the same result.

6. Conclusion

The examples given above are very simple, yet effective, demonstrations of how baselines can be used in incident handling. These baselines could have been established with any number of different tools; the examples could have been from any malware, or from of any number of different events. The results, however, would have been the same. The establishment of baselines makes an incident handler's job quicker and easier.

Without baselines, an incident handler would be left to guess, using his or her own intuition as regards whether or not the items in the examples above were normal.

It must also be acknowledged that baselines do not solve everything; they are just one part of a strategy of defense-in-depth. Even after they have been established, baselines can be fooled. In the first example, what if the output of the baseline were modified to hide "tini.exe"? What if a rootkit that hid the process were installed? In the second example, what if the network traffic were throttled? What if the bytes sent and received increased only marginally? In these cases, an incident handler might not notice anything out of the ordinary.

The great thing about baselines is that, if the network

traffic was throttled, probably what would happen is that something else would disturb the baseline. Perhaps there would be an unusual process running, for instance. The bottom line is that, even though baselines do not solve everything, they do give an incident handler another tool he or she can use to identify an incident.

There are also certain things for which establishing baselines can be rather difficult—such as network traffic, which may be very sporadic. Users may browse the Internet throughout the day, downloading files and doing many other tasks that generate network traffic as they go about their daily routines. Determining what is normal and what is not in such cases can be difficult. In the example used in this paper, it was fairly obvious that sustained network traffic of that magnitude was unusual.

Much of what has been discussed in this paper is a simple matter of good network and system administration. Yet, many times these things still are not put into practice. This can be for a number of reasons, including lack of time, lack of skills, and a lack of resources. Fortunately, each of these problems can be overcome with some effort.

As has been demonstrated herein, the establishment of baselines is not difficult. System and network administrators can be trained to do so. There is training available from a number of sources, including SANS, where system administrators can learn these valuable skills. Almost all vendors nowadays offer courses on administering their products. These courses usually include instruction in how to maintain and optimize system performance. Concerning resources, there are a number of free open-source solutions available. While these products may not have all of the extras of their commercial counterparts, they still are very effective for monitoring systems.

Having a change management control process in place is also important. Management should ensure that policies and procedures exist for change management, and that they are indeed being followed by staff. This will ensure that staff can quickly and easily determine what has changed on a network or system, and determine whether or not the change was authorized.

Also, the need for good documentation cannot be emphasized enough. Good documentation will help system administrators and incident handlers to determine quickly and easily important factors such as how the network is set up and configured; where

a particular system is; and how this particular program is supposed to be configured in order to communicate.

Preparation is an important part of the incident handling process. Preparing well can help an incident handler to identify and respond to an incident more quickly. It can also help him or her to be more efficient throughout the entire incident handling process. As has been shown, the establishment of baselines is an important part of incident handling.

7. References

Beebe, Nelson H. F. (2004). Ndiff: Compare putatively similar files, ignoring small numeric differences. Retrieved from the University of Utah Department of Mathematics Web site: http://www.math.utah.edu/~beebe/software/ndiff/

Cisco. (2001). Dealing with mallocfail and high CPU utilization resulting from the “code red” worm. Retrieved from http://www.cisco.com/warp/public/63/ts_codred_worm.shtml

Cisco. (2006). Simple Network Management Protocol (SNMP). Retrieved from http://www.cisco.com/univercd/cc/td/doc/cisintwk/ito_doc/snmp.htm

Cisco. (2007). Configuring SNMP support. Retrieved from http://www.cisco.com/univercd/cc/td/doc/product/software/ios122/122cgcr/ffun_c/fcfprt3/fcf014.htm

Gerhards, Rainer. (2003). SQL slammer lessons learned. Retrieved from the *Monitorware* Web site, http://www.monitorware.com/Common/en/Articles/SQLSlammer-Learnings.php

McIntyre, Jim. (2001a). Example 2: A sample Tripwire policy file. Retrieved from the *TechRepublic* Web site: http://articles.techrepublic.com.com/5100-6347-1053490.html

McIntyre, Jim (2001b). Using tripwire for filesystem integrity.

Part 2: Tripwire administration. Retrieved from the *TechRepublic* Web site: http://articles.techrepublic.com.com/5100-6345-1053398.html

Merriam-Webster Online Dictionary. (n.d.) Baseline. Retrieved from http://www.m-w.com/dictionary/baseline

Microsoft. (2006). How to: Configure Simple Network Management Protocol (SNMP) in Windows Server 2003. Retrieved from http://support.microsoft.com/kb/324263/

Microsoft. (2008). WMI: Windows Management Instrumentation. Retrieved from http://www.microsoft.com/whdc/system/pnppwr/wmi/default.mspx

Microsoft Tech Net. (2008). – Using the Windows Management Instrumentation Command-Line (WMIC) tool. Retrieved from http://www.microsoft.com.nsatc.net/resources/documentation/windows/xp/all/proddocs/en-us/wmic.mspx?mfr=true

SANS. (2006). Security 401: Security essentials. http://www.sans.org/training/description.php?mid=61&portal=b4b2467db19bd387b2ebefac4204431f

SANS. (n.d.) 504—Hacker techniques, exploits & incident handling. Retrieved from http://www.sans.org/training/description.php?mid=40&portal=b4b2467db19bd387b2ebefac42044

31f

PenguinSoft. (2000). NDiff_Quickstart: Tutorial for ndiff and related tools. Retrieved from http://www.penguin-soft.com/penguin/man/3/NDiff_Quickstart.html

8. Appendix A: baseline.bat

```
     @ECHO OFF
     ECHO
**************************************************************
*********
     ECHO **************************** BASELINE
*********************************
     ECHO
**************************************************************
*********
     REM wmic.exe OS list full
     REM wmic.exe USERACCOUNT list full
     REM wmic.exe GROUP list full
     REM wmic.exe SHARE list full
     ECHO.
     ECHO ******************* wmic.exe PROCESS list brief
************************
     ECHO.
     wmic.exe PROCESS list brief > 1.tmp
     REM wmic.exe SYSDRIVER list full
```

SOS Informatica 1/18/08 1:13 PM
Formatted: Portuguese (Brazil)

```
REM wmic.exe SERVICE list full
REM wmic.exe JOB list full
REM wmic.exe STARTUP list full
REM wmic.exe NICCONFIG list full
type 1.tmp
ECHO.
ECHO ***************************** netstat -ano
******************************
netstat -ano
ECHO.
REM ver
REM ipconfig /all
REM echo netstat -ano
REM route print
REM dir %SYSTEMDRIVE% /A:H /S /ON /T:W /N
REM dir %SYSTEMDRIVE% /A:-D /S ON /T:W /N
REM tasklist
REM systeminfo
REM netdiag /v
REM diruse ìC:\î
REM diruse /S ì%SYSTEMROOT%î
```

SOS Informatica 1/18/08 1:13 PM
Formatted: Portuguese (Brazil)

SOS Informatica 1/18/08 1:13 PM
Formatted: Portuguese (Brazil)

```
REM diruse /S ì%PROGRAMFILES%î
REM filever %SYSTEMDRIVE% /s /b /v
REM c:\sha1sum\sha1sum C:\Windows\*.dll
REM c:\sha1sum\sha1sum C:\Windows\*.exe
REM c:\sha1sum\sha1sum C:\Windows\System32\*.com
REM c:\sha1sum\sha1sum C:\Windows\System32\*.dll
REM c:\sha1sum\sha1sum C:\Windows\System32\*.exe
REM c:\sha1sum\sha1sum C:\Windows\System32\Drivers\*.sys
REM c:\sha1sum\sha1sum C:\Windows\System32\Drivers\etc\*.*
```

SOS Informatica 1/18/08 1:13 PM
Formatted: Portuguese (Brazil)