

NT/2K Incident Response Tools

H. Carvey 2001-08-15

NT/2K Incident Response Tools

by H. Carvey

last updated August 15, 2001

When responding to an incident, an investigator needs to know where to look for clues, as well as how to obtain the necessary information from the victim system. Many times, the investigator may not know exactly what she's looking for, and will have to collect some background information for analysis before continuing her search. However, the requirement to retrieve this information in a forensically sound manner still exists, as incident response investigations have an uncanny way of blossoming into full-fledged criminal investigations at just about the time you let your guard down.

Every organization is different, with a different infrastructure, different business requirements, and different security policies. As such, this paper is not intended as a guide to producing security policies or an incident response methodology. The purpose of this paper is to bring a wide variety of freely available software tools specific to NT and Win2K to the attention of the reader. However, this is by no means a complete and comprehensive list of such tools.

The tools presented in this paper are broken down three sections: Communications tools, tools for collecting volatile information, and tools for collecting non-volatile information. Each section will provide greater detail as to the type of information collected.

The intended method of employing these tools is to copy them to and run them from a CD. Ideally, the investigator will use this method to protect the tools themselves from being trojaned, deleted, or infected with viruses. The investigator should also consider including a "clean", known good copy of the command interpreter (cmd.exe) on the CD, as well.

One important resource that many investigators miss is Perl, which is freely available from [ActiveState](#). The investigator can use Perl to assist in parsing collected information, or as a glue language for combining and parsing commands. The investigator can also use Perl to create utilities that provide needed functionality that isn't found in other tools. These utilities can be compiled into standalone executables using [Perl2Exe](#), and then copied to the investigator's CD for use.

Communications Tools

When conducting an incident response investigation, the investigator needs to collect information about the current state of the 'victim' system, while altering the data on the victim system as little as possible. Some method must be used to collect information from the victim system and then save it on a forensic workstation for analysis. By far, the best tool for just this purpose is [netcat](#), a tool touted as a "Swiss army chainsaw". A netcat "listener" can be set up on the forensics workstation to collect information in a file with the following command:

```
c:\> nc -l -p 12345 > c:\case\victim\.dat
```

The information of interest is then sent from the victim system to the forensics workstation by using the following commands (where e: represents the CD-ROM drive):

```
e:\> | nc 12.34.56.78 12345
```

Should the investigator suspect that a sniffer is in use on the network, [Farm9's cryptcat](#) offers an excellent solution. Cryptcat is billed as "netcat enhanced with twofish encryption". Replacing "nc" in the commands above with the name of the cryptcat executable is all that is required to use cryptcat in place of netcat.

This method of getting information off of the victim system has an added benefit of being self-documenting. A key component of all computer investigations is documentation, specifically, what actions were taken at what time, and what was found. When using netcat to retrieve information from the victim system, the creation times of the files on the forensic workstation serve as a record of when the commands were run.

Collecting Volatile Information

Volatile information can best be described as information regarding the current state of the system that is not persistent, either over time or following a reboot. This is information that the investigator must obtain as soon as possible after the incident is identified and reported.

The first item of interest to an investigator is the processes running on the victim system. Quite a bit of information regarding processes can be obtained when the investigator uses a combination of several tools. [Pslist.exe](#), from [SysInternals](#), provides a comprehensive list of processes, PIDs, kernel and user time, and memory usage. [Pulist.exe](#), from the NT Resource

Kit, lists the PID and user for each process. [Fport.exe](#), from [FoundStone](#), lists the PID, process, and full path to the executable for the open TCP or UDP ports on the system. [Listdlls.exe](#), also from [SysInternals](#), lists the DLLs in use by each process. Listdlls.exe will also show the full path names of loaded DLLs, as well as indicate which loaded DLLs that have different version numbers than their corresponding on-disk files.

[Psloggedon.exe](#), from SysInternals, will give information regarding users logged on to the victim system locally, as well as via resource shares.

[Pstime.exe](#), another SysInternals tool, shows how long the system has been operating.

[Pclip.exe](#), from the [UnxUtils.zip](#) archive, sends the contents of the clipboard to STDOUT. Piping this command through netcat or cryptcat is relatively simple, as shown below:

```
e:\> pclip | nc 12.34.56.78 12345
```

The forensics workstation would be set up to receive the output as follows:

```
c:\> nc -l -p 12345 > c:\caseid\pclip.dat
```

The investigator should be sure to execute the "AT" command to see if the victim system has any jobs waiting to be executed.

The investigator will also need to collect information regarding network connections. This can be done by executing the "netstat -a" and "nbtstat -c" commands and piping the output through netcat or cryptcat to the forensics workstation. Variations on the use of net.exe will display information regarding files opened via the network and network sessions.

Other volatile information includes the actual network communications between the victim system and other systems. Several freely available "sniffer" programs exist for NT/2K platforms, and are relatively easy to set up on a forensics workstation or laptop. Starting with the [winpcap](#) libraries, the investigator can then opt for [WinDump](#), [Analyzer](#), [Ethereal](#), or even [snort](#). All of these tools can be used to capture network traffic, or read and translate tcpdump-style binaries of packet captures from other systems.

Collecting Non-volatile Information

Non-volatile information generally consists of configuration settings on the system that do not change over time or when the system is rebooted. The investigator may opt to collect this information in the same manner as volatile information, or by mapping drives from the victim system to the forensics workstation. Several factors should be considered at this point, the most important of which is, do any shares exist? By default, both NT and Win2K provide hidden administrative shares. However, this functionality can be overridden through the use of Registry key, or the shares may not be available as the Server service has been stopped. If the shares are available, they may be mapped to the forensics workstation, and the investigator must be sure to document that she has done so.

Files

The first and perhaps most important non-volatile information that the investigator should collect are the last modification, last access, creation (MAC) times for files on the victim system. A variety of tools exist to assist the investigator in collecting this information. For example, `afind.exe`, from FoundStone's [Forensic Toolkit](#), will allow the investigator to view last access times for files. However, a very simple and straightforward method of collecting this information is to use the [mac.pl](#) Perl script. Using this tool, the investigator will end up with a comma-delimited file (by saving this file with a ".csv" extension, it can be opened in Excel) containing file MAC times from the victim system. At this point, the investigator can use more intrusive methods to retrieve information from the victim system. Tasks such as generating checksums for files, or performing file signature analysis, will alter the last access times of the files, which is important information that the investigator must preserve.

Once the file MAC times are collected, the investigator may have determined that the 'attack' was of a somewhat sophisticated nature, and may opt to examine the victim system for alternate data streams (ADSs). ADSs are a feature of the NTFS file system that allows for the storage of data and executables in such a manner as to be invisible to the file tools Microsoft provides (`dir`, Windows Explorer, etc.). Frank Heyne provides an excellent CLI tool called [LADS](#) that is very easy to use. [Ads.pl](#) is also available. Both of these tools detect ADSs attached to files, as well as those associated with the directory listing.

The investigator may decide to perform file signature analysis. On Windows systems, the operating system associates files with executables based on the file extension. For example, files with the extension ".PPT" are associated with PowerPoint. Files can be "hidden" on the system by altering their extension, such as changing "mypic.jpg" to "mypic.txt". When

conducting file signature analysis, the investigator examines the first 20 bytes of a file, and checks to see if the file is associated with the correct file extension. Executable files, with extensions such as DLL, SYS, EXE, etc., contain the characters "MZ" in the first several bytes of the file. [Sig.pl](#) will allow the investigator to perform file signature analysis on files from the victim system. However, the investigator should have already collected MAC times from the files to be analyzed. Opening the file to retrieve even the first 20 bytes will alter the last access time for the file.

The EventLogs on the victim system may provide the investigator with some clues as to what happened. [DumpEvt.exe](#) is an excellent tool for this purpose, as is [dumpevt.pl](#). The Perl script has the added advantage of collecting the EventLog audit settings (as seen in User Manager on NT) as well. This information can be saved as a ".csv" file, for easy analysis via Excel. In addition to the EventLogs, the investigator should be sure to collect log files from applications running on the victim system. Often time, the log files from IIS can provide valuable information regarding the incident.

Other files may be of interest to the investigator. On NT, each profile on the system has a StartUp folder. On Win2K, the files are located in the "Documents and Settings" folder. While the files located in these directories are shortcuts, the files they link to can be very easily retrieved and may provide clues to the investigator. For example, a trojan may be initiated by writing to the StartUp folder of the user, rather than to the Registry. In some cases, a "Recent" folder can also be found within the user's profile directory. The files linked to in this folder can provide the investigator with clues regarding the files most recently accessed by the user. [Startup.pl](#) retrieves such information from remote NT systems, and can be easily modified to do the same for Win2K systems.

Should the investigator locate suspicious files on the victim system, they will need to be examined. The files can be copied off of the victim system via netcat/crypcat, by copying the file to a clean floppy disk, or via a drive mapped to the forensic workstation. The investigator should ensure that MAC times have been recorded for the files in question, as well as permissions on the files. File permissions can be retrieved using [cacls.exe](#), which is included on NT and Win2K systems. MD5 checksums of the files should be generated prior to and immediately after the files are copied, to ensure the integrity of the files. [Md5sum.exe](#), from the [UnxUtils.zip](#) archive, can be used for this purpose. Once the files have been copied to the forensics workstation, and the activity has been documented, the investigator can examine the files for clues as to their purpose. [Strings.exe](#) from SysInternals, and alternatively, [BinText](#)

from FoundStone, will display Unicode, ASCII, and resource strings embedded within the file. [Finfo.pl](#) attempts to retrieve information from the resource section of a file, such as the company name, file description, etc.

The Registry

The Registry of the victim system can be a gold mine of information to an investigator. For example, in order to be persistent across logins and reboots, many trojans, and other malware, tend to leave a footprint in the following Registry key:

```
HKLM\Software\Microsoft\Windows\CurrentVersion\Run
```

Also, the RunOnce, RunOnceEx, and RunServices subkeys (all located under the .. \CurrentVersion key) may contain suspicious entries.

The contents of other Registry keys may also be of interest to the investigator, depending upon the nature of the reported incident. For example, [autoruns.exe](#) from SysInternals checks Registry keys that are examined during the system startup and user logon processes.

However, many trojans can be configured to have names that are less suspicious than "NetBus", or "SubSeven Server". The investigator needs to maintain a running list of names used by various malware, as well as the particular Registry keys written to (BO2K likes to install itself as a service, leaving a footprint in HKEY_LOCAL_MACHINE\System\CurrentControlSet\Services) during the installation. This information can be obtained by regularly reviewing the web sites of anti-virus vendors. For example, the [F-Secure Virus Information](#) web site shows that the SirCam virus creates entries in the following Registry keys when infecting a system:

```
HKCR\exefile\shell\open\command
```

```
HKLM\Software\Microsoft\Windows\CurrentVersion\RunServices
```

Another option available to the investigator is to maintain text-based dumps of the Registry for comparison when an incident is identified. However, this option can be resource intensive when dealing with user workstations, rather than simply critical servers.

Other sources of information within the Registry include keys that maintain lists of most recently used documents, web sites typed into the Internet Explorer Address bar, etc. Another

key that may be of interest on NT systems is:

```
HKEY_CURRENT_USER\Software\Microsoft\Telnet
```

This Registry key maintains a list of the last ten hosts and ports with which the telnet.exe client was used to connect. This information can be used to determine if the telnet.exe client on the NT system was used to connect to hosts for malicious purposes, such as to take advantage of the /scripts/root.exe file left behind by the sadmind/IIS or Code Red II worms.

The contents of Registry keys can be easily queried using either a Perl script, or Microsoft's own [Registry Console Tool](#). The Registry Console Tool can be used in a batch file in order to query multiple keys, or run from the command line to obtain information from individual keys.

Also of use to the investigator is the fact that Registry keys have a value called the "LastWrite" time, which is similar to the last modification time of a file. Using this information, the investigator can determine approximately when certain activity occurred. [Keytime.pl](#) will allow the investigator to retrieve the LastWrite times from specific keys. With regards to the Telnet key mentioned above, the investigator would see the approximate time that the last site was accessed using the client.

The investigator should also collect information regarding the services on the victim system. The investigator will be most interested in the name and current state of the service, the service executable, and the account under which the service runs. This information is easily obtained via a Perl script such as [service.pl](#). The same technique can be used with regards to available shared resources on the victim system, using [share.pl](#).

Information regarding user accounts on the victim system, their group membership, and what privileges each account holds may provide the investigator with clues. [Rights.pl](#) provides this information.

Conclusion

Knowing where to look, and what tools to use, can go a long way toward assisting the investigator in determining what occurred with regards to a reported incident. There are a wide variety of freely available tools to choose from, as well as an extraordinary amount of utility and functionality available through Perl. If there is any information that an investigator needs to obtain from an NT or Win2K system, there is most likely already a tool available to do just that.

[Privacy Statement](#)

Copyright 2006, SecurityFocus