

# Win2K First Responder's Guide

H. Carvey 2002-09-05

## Introduction

When it comes to handling computer security incidents, proper first response handling of computer security incidents is second in importance only to incident prevention. Improper handling or collection of available information can do irreparable harm to an investigation. Investigators need to have a thorough understanding of what information they intend to collect, as well as the tools they can use and the effects those tools have on the system itself.

Investigators know that not every event reported will require a full investigation or lead to prosecution. Obviously, each incident will make different demands on investigators; however, incident handling personnel should not deviate from best practices and assume that different procedures should be used to handle an event. There are specific items of information that can be collected and analyzed quickly in order to determine what follow-up steps need to be taken. This article will offer a brief overview of some of the steps security administrators and incident handlers should take as part of the first response to security incidents. This article will focus on incidents in Microsoft Windows 2000, due to its popularity in both the corporate and server environments. Many of the general topics discussed in this article are applicable across other platforms, and many of the specific techniques and tools discussed can also be employed on NT and XP.

Throughout this article the terms "investigator" and "first responder" will be used. The former refers to the individual with overall investigative responsibilities, while the latter refers to the first person to approach and access the system once an incident or event has been reported.

## Step One: Protecting "Volatile Information"

When someone talks about "forensics" or "computer forensics", images of shutting down the suspect machine and making a bit-level image of the drive often come to mind. There are even discussions and quasi-religious feuds over the best methods for shutting a machine down. However, in shutting a Win2K machine down, a great deal of very valuable information can be lost. All data regarding network connections, processes, and memory contents is lost when the system is shut down. Due to this characteristic, this data is referred to as "volatile information".

Volatile information can be extremely important, not only in terms of determining what follow-up steps are required of the investigator, but also in terms of money. Many organizations measure the downtime of critical systems in dollars (tens or hundreds) per minute. Shutting down a system with a great deal of storage and files can be very, very expensive, particularly when sufficient information can be easily collected to determine the nature of the incident. Therefore, it is incumbent upon the first responder

(usually an administrator) to have a thorough understanding of what types of information can and should be retrieved prior to an incident being reported.

## Developing The Toolkit

Usually when an incident of immediate concern occurs, there is evidence of the steps leading up to the incident in log files, such as the EventLog (if auditing is enabled), or IIS or FTP logs. However, after the attack has occurred, there may be a process running, such as an IRC bot, or a Trojan. Information about this process, such as the full command line, path to the image, and user context that it's running in, can be easily retrieved using a combination of tools such as [pslist.exe](#), [handle.exe](#), and [listdlls.exe](#), all of which are available from [sysinternals](#). These tools can provide a great deal of information about the resources in use by the process, but the basic information already mentioned should be enough for an administrator or first responder to recognize anything suspicious or unusual.

More detailed information regarding processes is available using process-to-port mapping tools like [fport.exe](#) from [Foundstone](#). [Fport](#) will show which ports are opened by which processes, and the native `netstat -a` command will show which remote host, if any, is connected to that port. (Author's Note: `fport.exe` does not work on XP. Instead, the `-o` switch should be used with the `netstat` command to provide the PID of the process using the port in question. This is unique to XP, and does not apply to NT or 2K.)

Some of the information available from these tools may look relatively "normal", but may prove otherwise once additional information is collected. For example, variations of the `net` and `nbtstat` commands can reveal a great deal of information about NetBIOS connections to and from the system being examined. The following is a list of commands that should be executed to gather initial information:

```
net use
net session
net file
net share
net view
net user
net accounts
net localgroup
net start
nbtstat -c
nbtstat -n
nbtstat -s
```

The output of each of these commands should be redirected to a file, appending the information from subsequent commands to the file, or by creating a new file for each command. For detailed information

regarding each variation of the above commands, type the command, followed by "?", at the command prompt to obtain syntax information. In the case of the `net` commands, `net view /?` will provide an explanation of the information returned by the command. For `nbtstat`, type simply `nbtstat /?`.

The `netstat` and `arp` commands should be used to gather volatile information specific to the Internet and Ethernet connectivity of the suspect system:

```
netstat -an
netstat -r
arp -a
```

For instances in which system misuse is suspected, the first responder should be sure to retrieve the contents of the Clipboard, using `pclip.exe` from the [Unix Utilities archive](#). Important evidence may be stored on the Clipboard, including file contents or passwords. If there is a command prompt open on the screen or minimized to the Task Bar, the first responder should document this fact, and then execute the `doskey` (specifically, `doskey /history`) command within that command prompt to retrieve the command history. (Author's Note: This last item is very important, particularly when reviewing where information or evidence can be hidden on a system. In one instance when I was teaching a "live" forensics course for Win2K, I had half of the attendees set up a lab that the other half would have to "investigate". One of the steps in the lab setup was to open a command prompt, type in the prescribed commands, type "cls" to clear the screen, and then minimize the prompt. When the other attendees returned to the lab, one sat down at the "victim" station, and his first action was to close...Alt+F4...the prompt.)

The investigator may also wish to obtain information from the Registry of the suspect machine. While this information may not be considered volatile in nature, many times information from the Registry will assist the investigator in making decisions regarding the status of the investigation, whether or not shutting down the system to make an image of the drive justifies the cost of doing so, etc. One of the best tools available for collecting information regarding Registry entries is `reg.exe`, which is available in [the security.exe zipped archive](#). `Reg.exe` is a command line utility that can be used in batch files to retrieve information from the Registry. `Reg.exe` can be used to retrieve all entries from the Winlogon key by typing the following command:

```
C:\ security>reg query "Software\Microsoft\Windows NT\CurrentVersion\WinLogon"
```

Similarly, typing the following command can retrieve the contents of the "Run" key and all subkeys:

```
C:\ security>reg query Software\Microsoft\Windows\CurrentVersion\Run /s
```

`Reg.exe` can also be used to retrieve the data from specific Registry keys:

```
C:\ security>reg query "Software\Microsoft\Windows NT\CurrentVersion\AeDebug"
```

\Debugger?

Another very useful tool provided in the security.exe archive is auditpol.exe. This tool provides information regarding the current audit configuration of the system, letting the investigator see if auditing is enabled, and if so, what events are being audited. Again, though this information is not necessarily considered to be volatile, it may prove to be extremely useful to the investigator.

## Deploying The Toolkit

Once the investigator or first responder has an understanding of what information, volatile or otherwise, can be collected from a Win2K system, an appropriate toolkit should be developed and deployed to assist in collecting this information. By collecting the necessary tools and automating their usage through scripting, the process will be less prone to mistakes and deletions. Copying the tools to a diskette, for instance, and writing a batch file that launches the various tools has many benefits. First, the process is automated and can be easily updated by changing the batch file. Second, by using removable media such as a diskette, the output of the commands can be written to a flat file on the diskette itself, rather than tainting potential evidence by writing the file to the hard drive of the system. The investigator or first responder simply needs to ensure that the tools on the diskette leave enough room for the file to be written. Third, by automating the process, the first responder is less likely to make mistakes, such as mistyping a command or not providing the right switches in the arguments for the command.

Another method of retrieving information from a system without writing data to the hard drive is to pipe that information to another system via a network socket. One of the most popular programs that provides a facility for this is [Netcat](#), which is often referred to by security professionals as a "TCP/IP Swiss army chainsaw". The use of Netcat on Solaris and Linux systems has long been popular for such tasks as [creating drive images](#) over a network. The Win32 version of Netcat works very well on Win2K, and can be used to retrieve data from systems.

The process of using Netcat starts by setting up a Netcat listener on a forensics workstation:

```
C:\forensics>nc -l -p 30000 > forensics.data
```

The above command opens a port, and redirects anything received on that port to the file called "forensics.data". For the sake of this example, the IP address of the forensics workstation is 10.1.1.6.

Once the listener is set up, the first responder can insert his diskette containing tools and a batch file into the drive of the "victim" system. This time, however, the batch file will contain entries that send the output data from the commands to the remote system, rather than to a file on the diskette itself. The entries in the batch file will look like this:

```
net use | nc 10.1.1.6 30000  
net session | nc 10.1.1.6 30000
```

If the investigator suspects that a sniffer is active on his network, or would simply prefer to not pass the collected data over the network as plain text, an encrypted version of netcat called [Cryptcat](#) can be employed. Cryptcat is simply Netcat enhanced with TwoFish encryption, and is used exactly the same way Netcat is used. The only difference between the two is the addition of encryption to Cryptcat. (Author's Note: The default secret key for Cryptcat is reportedly "metallica", so for added security be sure to use the "-k" switch to change the default key.)

An alternative method of providing this capability to the first responder is to create a CD with the tools and batch file. This method has the benefit of providing the first responder with a greater range of tools on a single medium, rather than carrying around several diskettes. All of the tools used can be burned to the CD, including the tools that are native to the Win2K system, such as nbtstat.exe, netstat.exe, and net.exe. This provides added security by not only ensuring that "clean" versions of the tools are used, but by also ensuring that the tools themselves cannot be infected by viruses while they are in use.

The end result of the above Netcat (or Cryptcat) example will be a single large file on the forensics workstation containing all of the information that the first responder collected. This file can be broken up for analysis, or another agent can sit at the forensics workstation and coordinate stopping and restarting the listener with a new filename with the first responder.

An alternative to the use of Netcat or Cryptcat is to develop an application suite consisting of a server, and a client or clients that can be burned to a CD and employed by the first responder. These clients should be comprehensive and run automatically with minimal interaction from the first responder, other than to provide the IP address and port used by the server. The clients would connect to and send their information to the server via network sockets, and the server would handle the storage and even processing of the received data. This is the goal of the [Forensics Server Project](#), currently being developed by the author. The project consists of a server component, and three client components. The client components provide the following functionality:

1. Volatile data client that automatically retrieves data necessary for the investigation. This includes such things as data regarding the system itself, network connections, scheduled jobs, clipboard contents, and audit configuration data. Registry values that may impact the analysis of the rest of the data will also be collected automatically.
2. An external command client that allows the first responder to run any external command. This is particularly useful for collecting data using tools that provide a unique functionality, such as FoundStone's fport.exe.
3. A file client that automatically processes the collection of files. The client automatically collects MAC times, hashes, and other pertinent information from files before copying the binary contents of the

files to the server. The server stores the data, as well as the binary file contents, and verifies the hashes to ensure the integrity of the copied files.

The server component is responsible for storing the data sent to it by the clients, as well as documenting all activity and errors during the collection phase. Additionally, the server component will also include various correlation and analysis tools, such as [procdmp](#).

The purpose of the project is to provide a greater range of information collection, verification, and analysis to first responders and investigators during incident response activities. For example, rather than requiring the first responder to collect information on file hashes by typing in the filenames, the first responder can access the file client's GUI and select the files in question, and then simply click a button. The components of the project will automatically handle the collection of information, processing, copying, and verification of those files.

Questions regarding the status of the project should be directed to the author.

## **Conclusion**

Investigators can save themselves a great deal of time, effort, and stress by simply preparing for incidents ahead of time. Perhaps the most effective incident preparation measures include host and network hardening, in order to prevent incidents from happening. Hardening also ensures that incidents are 'noisy', in that valuable audit data will be generated. However, incidents will occur, and investigators and first responders need to be prepared to respond effectively. It is hoped that this article has provided readers with an overview of some of the steps that are required to respond to these incidents properly, as well as some of the tools that are available to carry out these steps.

## **Acknowledgements**

The author acknowledges Special Agent Jesse Kornblum, AFOSI, for his excellent [DFRWS paper](#).

[Privacy Statement](#)

Copyright 2006, SecurityFocus